

Arthur Gruber • Alan M Durham • Chuong Huynh • Hernando A del Portillo

# Bioinformatics in Tropical Disease Research

A Practical and Case-Study Approach

Last Updated: 2008



National Center for Biotechnology Information (US)  
Bethesda (MD)

National Center for Biotechnology Information (US), Bethesda (MD)

NLM Citation: Gruber A, Durham AM, Huynh C, et al., editors. *Bioinformatics in Tropical Disease Research: A Practical and Case-Study Approach* [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2008.

This book is intended to serve both as a textbook for short bioinformatics courses and as a base for a self-teaching endeavor. It is divided in two parts: A. Bioinformatics Techniques and B. Case Studies. Each chapter of the first part addresses a specific problem in bioinformatics and consists of a theoretical part and of a detailed tutorial with practical applications of that theory using software freely available on the Internet. All of the authors who were selected for this part of the book have extensive experience in teaching Bioinformatics, either at WHO-TDR and other short-term courses, at universities around the world, or at both. In the second part, we invited renowned researchers to write chapters that represent up-to-date reviews of particular human diseases, including biological aspects and bioinformatics approaches that helped to solve specific problems.

## Table of Contents

Preface.....	v
<b>A Practical Bioinformatics</b> .....	1
Chapter A01 Using Linux..... <i>Alan Mitchell Durham and Marco Dimas Gubitoso</i>	3
Chapter A02 Understanding Database Design..... <i>João Eduardo Ferreira and Osvaldo Kotaro Takai</i>	59
Chapter A05 Similarity Search..... <i>João Carlos Setubal and Ruediger Braeuning</i>	83
Chapter A06 Protein Structure, Modelling and Applications..... <i>Ardala Breda, Napoleão Fonseca Valadares, Osmar Norberto de Souza, and Richard Charles Garratt</i>	137
<b>B Case Studies</b> .....	171
Chapter B01 Control of Gene Expression in <i>Plasmodium</i> ..... <i>Mauro Ferreira de Azevedo and Hernando A. del Portillo</i>	173
Chapter B02 <i>Leishmania</i> Genomics: Where Do We Stand?..... <i>Silvia R.B Uliana, Jeronimo C. Ruiz, and Angela K. Cruz</i>	195
Chapter B03 Tuberculosis, Leprosy, and Other Mycobacterioses..... <i>Sylvia Cardoso Leão, Maria Isabel Romano, and Maria Jesus Garcia</i>	213
Chapter B04 Transgenic Vectors: <i>Anopheles</i> and <i>Aedes</i> ..... <i>Márcia Aparecida Sperança, Paulo Eduardo Martins Ribolla, and Margareth Lara Capurro</i>	233
Chapter B05 Proteomics Studies in <i>Trypanosoma cruzi</i> ..... <i>Ernesto S. Nakayasu and Igor C. Almeida</i>	249
Chapter B06 Expressed Sequence Tags (ESTs) and Gene Discovery: <i>Schistosoma mansoni</i> ..... <i>Ricardo DeMarco and Sergio Verjovski-Almeida</i>	261
Chapter B07 Alternative Splicing: Lessons from Cancer..... <i>Sandro J. de Souza and Anamaria A. Camargo</i>	273

## Preface

This book is the result of a long process that began with a series of Latin American Bioinformatics Training Courses for Tropical Disease Research. These courses, sponsored by WHO-TDR, took place in São Paulo, Brazil, in the years 2002, 2003, 2005, and 2006. They were designed as crash courses for biological researchers, trying to cover in 2 weeks enough information to enable biologists to continue their bioinformatics training independently.

During these courses, it became clear that, in spite of the growing number of books in bioinformatics, most of the books covered the issues either at a theoretical level that was too deep for beginners or at a level that was too superficial to convey the complexity of the issues. Additionally, we were not aware of any book that, for each theoretical subject, provided a detailed tutorial of free software that could be used in day-to-day research. Finally, in our experience, most of the researchers in biological subject areas feel intimidated when confronted with bioinformatics programs.

Hernando del Portillo, the initial course coordinator, then had the idea of creating such a book, taking advantage of the great human resources that were involved in the lectures from the various modules of the several WHO-TDR courses held in Brazil and other countries. He was also the person who got the group of editors together. Hernando, unfortunately, was able to be involved in only the first part of the editorial process, having to leave the book project before the last phase of the work. However, his contribution remains in the effort to select the authors, advocating for the case studies part, making the initial evaluations of the chapters, and co-authoring one of the chapters.

All of the editors of this book have extensive experience in both teaching and organizing bioinformatics courses. Arthur Gruber was part of the organizing committee of all the Brazilian courses, co-Primary Investigator (PI) of the last one, and instructor in short-term Bioinformatics courses in Brazil, South Africa, Peru, and Colombia. Alan Durham was part of the organizing committee of the first course in Brazil, co-PI of the second and third courses in Brazil, and PI of the 2006 course in Brazil; he was also part of the organizing committees and teaching staff in short-term courses in Brazil, South Africa, Thailand, and India. Chuong Huynh has been in the coordinating committee for all of the WHO-TDR courses around the world (five in Brazil, four in South Africa, four in India, and seven in Thailand) and also has been an instructor in all of them. Hernando del Portillo was the PI of the first three Brazilian courses. Finally, Alan, Arthur, and Hernando were part of the group that organized in 2002 one of the first PhD Bioinformatics Programs in Brazil.

This book is intended to serve both as a textbook for short bioinformatics courses and as a base for a self-teaching endeavor. It is divided in two parts: A. Bioinformatics Techniques and B. Case Studies. Each chapter of the first part addresses a specific problem in bioinformatics and consists of a theoretical part and of a detailed tutorial with practical applications of that theory using software freely available on the Internet. All of the authors who were selected for this part of the book have extensive experience in teaching Bioinformatics, either at WHO-TDR and other short-term courses, at universities around the world, or at both. In the second part, we invited renowned researchers to write chapters that represent up-to-date reviews of particular human diseases, including biological aspects and bioinformatics approaches that helped to solve specific problems.

The book is intended to be a continuous project, and we expect it to be regularly expanded and updated in the future. This characteristic and the desire to make the book widely available, particularly to researchers in developing countries, were key points that led to the decision of open web publishing, as opposed to a paper version.

We feel extremely fortunate with our choice of authors and expect this book to be very useful for both teachers and individual researchers. We thank all of the authors for their great work and for maintaining the motivation in spite of the delays during these last 2 years. Finally, we are deeply grateful to Belinda Beck and Laura Dean for their extremely competent and comprehensive editorial work.

March 2008

Arthur Gruber

Alan M. Durham

Chuong Huynh

# A. Practical Bioinformatics





## Chapter A01. Using Linux

Alan Mitchell Durham, PhD<sup>1</sup> and Marco Dimas Gubitoso, PhD<sup>2</sup>

Created: May 1, 2006.

Linux started in 1991 as an experiment on the processor 80386, from Intel. The idea was to make a UNIX-like system run efficiently on a microcomputer. UNIX was by that time a very popular system — and still is — for large machines and for scientific/academic research.

Its author, Linus Torvalds, posted a message on *Usenet* — a former kind of world wide forum — announcing he built the central part of a UNIX-like operating system and was making it available to anyone who would like to play with it. It was an adaptation of *Minix*, an educational and limited version of UNIX. The development since then was very quick, and as soon they were able to run programs, they could benefit from the large set of tools provided by the *GNU project*<sup>1</sup>.

The GNU project started a few years earlier by Richard Stallman of MIT, when he decided to develop a full UNIX clone under the concepts of *free software*. The idea of 'free' software is that one can copy, distribute and modify it as needed, without the restrictions of royalties or copyright violations, as a consequence, the programs are available for free — one can charge for copying the program but not for the program itself.<sup>2</sup> This is enforced by the "GNU General Public License", or GPL for short.

GNU was not complete, but had all the essential programs, except the kernel. Many applications like text editors, compilers and windowing systems were also available. The union of these programs with Linus' kernel gave birth to a full featured UNIX-like operating system: *Linux*.

The system is still evolving and everyday new programs are added, the vast majority under GPL. In particular, most of today' free bioinformatics software run in the Linux system.

## Distributions

Operating systems involve a large amount of programs, along with data, libraries and configuration files, which demand a well defined structure and organization. Each operating system has its own way of organizing the files, and for some of them, there may be more than one alternative.

This complexity makes it very difficult to make an installation without specific policies and helper programs. To solve this problem, the Linux community came up with the concept of distribution, which is essentially a set of programs wrapped in *packages*, tested to work together, so there is a guarantee that no conflicts will arise and it is possible to establish a simple installation and upgrading procedure.

There are many Linux distributions nowadays, but most of them are based on one of the following three:

**Slackware** - It is a more basic distribution, almost a "raw" one, directed to those users who like to experiment and test new features.

**Red Hat** - Intended to be easy to install, with ready solutions for most end users tasks. Has a set of configuration tools easy to use, at the expense of some flexibility. There are many derived distributions, like S.U.S.E and Mandriva

---

**Author Affiliations:** 1 Department of Computer Science; University of São Paulo; Brazil; Email: Alan.brazil@gmail.com. 2 Department of Computer Science; University of São Paulo, Brazil; Email: gubi@ime.usp.br.

1 GNU stands for GNU is Not UNIX

2 To preserve the rights of the author, Stallman came up with an innovative type of copyright: the *copyleft*.

**Debian** - Combines flexibility with ease of use and is one of the more popular nowadays. As with Red Hat, there are some derived distributions, Knoppix and Ubuntu being the most famous. The companion CD for this book is based on the Knoppix distribution.

The rest of this chapter is organized as follows. We first give an introduction to the Linux system and its basic concepts. Then we have a tutorial where you will learn about Linux with a hands-on approach. The tutorial is followed by a "Frequently Asked Questions" (FAQ) section, that you can use afterwards as a reference on how to perform different tasks in Linux. At the end there is a series of short exercises for practice.

## 1.1. UNIX and Linux Environments

### 1.1.1. Using Windows<sup>®</sup> to Understand Linux

Most books on operating systems introduce the basic concepts of the systems from scratch. In this chapter we take a different approach. Since most people have used computers before, and since the different versions of MS Windows<sup>®</sup> are installed in over 90 percent of the desktops nowadays, we will explain the main concepts of Linux trying to associate, as much as possible, with the Windows<sup>®</sup> counterparts.

Linux, like MS Windows<sup>®</sup>, is an operating system, that is, a software that controls the machine, running programs, managing devices like CD-ROM readers, hard disks, monitors, etc. However, Linux is a version of UNIX, a system that was conceived as a multi-user system since the beginning. Also, in the UNIX world, computers have been linked to networks since the 1970's. This means that these systems have had the possibility of multiple users and of the use of internet as part of their design since the early days.

There are important characteristics of Linux that we should emphasize:

- **Multiuser x personal systems:** Until the advent of Windows<sup>®</sup> NT, Microsoft systems were designed for *personal* computers. That means that, even if more than one person was using a computer, the system assumed that there was no need for protecting the data in the computer, or to prevent the user from doing any particular task. Linux, like Windows<sup>®</sup> NT, Windows<sup>®</sup> 2000 and Windows<sup>®</sup> XP, is a multiuser system. This means it is assumed that more than one person will use the system. Having more than one user means that the system needs to control the access to the computer and protect the data of one user from the other. To provide this separation a Linux system provides user *accounts*, and each account has a *password*. This used to be a novel concept for most users but, with the advent of Internet mail, most people have an e-mail account in some Internet site. The concept is exactly the same.

**The administrator:** In multi-user systems, many people will be using the computer, therefore important tasks involving the basic functioning of the system need to be reserved only to a special user, the *system administrator* (or *root*). The administrator is generally the only user that can install or remove software, create new user accounts and modify the configuration of the computer (like, for example, adding new disks). Administration of a Linux system is not a trivial task and requires some expertise, mainly if you are managing a Linux server. For machines used as personal workstations, the administration tasks can be reduced to a minimum, and can generally be performed by someone with less expertise.

- **Protection:** Protection of data and programs in Linux follows a very simple, yet very flexible, protocol. For each file, program, disk volume or device, there is a set of three permissions, associated with three different user categories.

User categories:

- **owner** (u) - usually the user that created the file (the owner, however, can be changed)
- **group** (g) - a file also belong to a group. User groups are created by the administrator and can have any number of users. A user can also participate in any number of groups.

- **other** (o) - any user that is not the owner and does not belong to the file's group.

#### Permissions

- **read** (r) - indicates that a file's content can be copied or read.
- **write** (w) - indicates that a file's contents can be modified or erased.
- **execute** (x) - indicates that a file can be run as a program. If set for a directory, indicates that the user can "enter" the directory.

Therefore, for each file or directory we have three categories of users: the owner, the users that belong to the file's (or directory's) group, and the rest of the world. Also, for each category there are three permissions: reading, writing and executing. This means each file has 9 different protection fields associated to it.

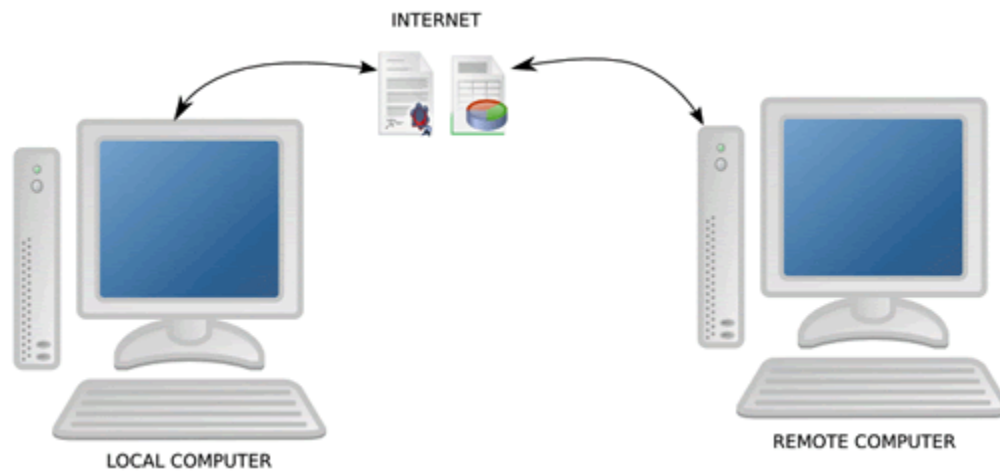
The flexibility of the Linux's protection scheme can be very useful. Let's consider some examples:

- A laboratory has three different research groups. Work has to be confidential among groups, but researchers in the same group can share information freely. The head researcher of the laboratory has full access to all information. In this situation we can have three user groups in Linux, each group containing the users associated with the respective research group. The account of the head researcher is included in all groups. All files have full access (reading, writing, executing) for the owner and for the group, but no access for the rest of the world.
- A researcher has a project that is confidential. He or she is the only one that should be allowed to modify the files of the project, or to execute the programs being developed. However, the head researcher of the lab should be able to access the files for inspection. In this case we need one user group with two accounts: the researcher's account and the lab head's account. All files in the project will have the researcher as the owner. The owner will have full access (read, write, execute), but the group will have only read access. The rest of the world will have no access to the file.
- A laboratory obtains a special license to run some software. However, the software should be run only in the computers of that particular laboratory. No user should be allowed to inspect, copy or modify the code. The file that contains the program should have the administrator as the user (or the account of the person responsible for the software). A group should be created for all users of the laboratory. The owner will have full access, the group will have only execution permission, and the rest of the world will have no permissions set.
- You decide to create a public folder to use for file exchange. Anyone can read files in the folder or copy files from it. In this case, you set the folder to have all permissions.
- To avoid making mistakes, if you have a file, you want to be sure it is not going to be deleted. One possible example is your income tax file, which you want to keep untouched after you finish it. In this case you can create a file that not even the owner has write permission. This means the file cannot be deleted either. To modify or remove the file you will need to change its permissions first.

The table below shows a summary of the examples above.

File Name	Owner		Group		Others		
GroupProject1	R	W	R	W			X
ConfidentialProject	R	W	X	R			
SpecialSoftware	R	W	X			X	
PublicFolder	R	W		R	W	R	W
IncomeTax	R						

The UNIX command to set permissions will be explained in the practical section of this chapter.



**Figure 1.1.** Data exchange. Knowing the address of the remote computer, you can send files or download files

## Data exchange

Normal file exchange in the MS Windows<sup>®</sup> world involves having access to an Internet site for downloads and having especially designed Internet pages for uploading information (like, for example, sending a file to be attached to an e-mail message in a web mail site). However, maintaining a web service is not an easy task and involves many security risks.

By contrast, any two Linux computers in the Internet can exchange information safely, independently of the fact that they maintain a web service or not. There are a series of programs that allow users to download and upload information across the Internet safely, provided that they have a regular user account and password in the remote computer (fig. 1.1).

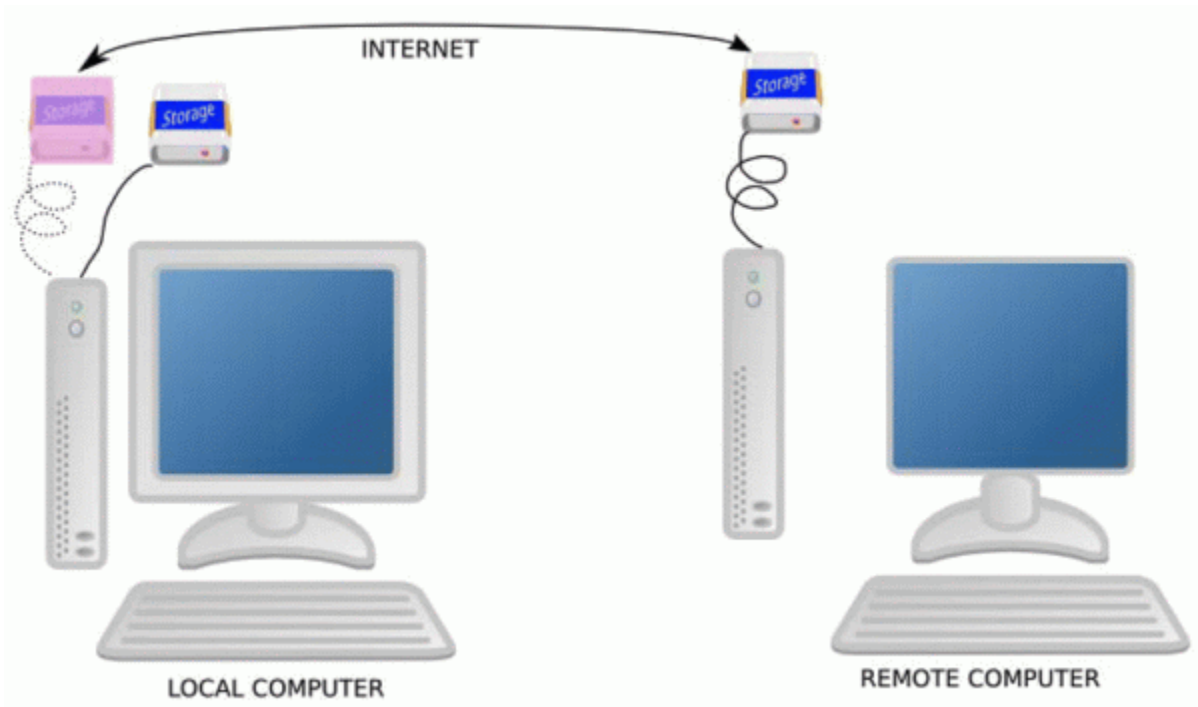
## Sharing disks

A common problem in laboratories with more than one computer is the fact that information generated when someone uses a specific machine is kept locally. MS Windows<sup>®</sup> allows you to share information across computers using the "network neighborhood" icon. Good network practice would involve requiring users to only store information in a single computer, a server. Having all information in a server means the users can access it from any other machine plugged on the net. In the Window's world, the administrators of the server can make folders available to users. A user can then "map" the remote folder in his local account. All information stored in the mapped folder is actually being stored remotely. This type of sharing involves some user knowledge (to map the remote folder) and discipline to always use the folder that is mapped to the server.

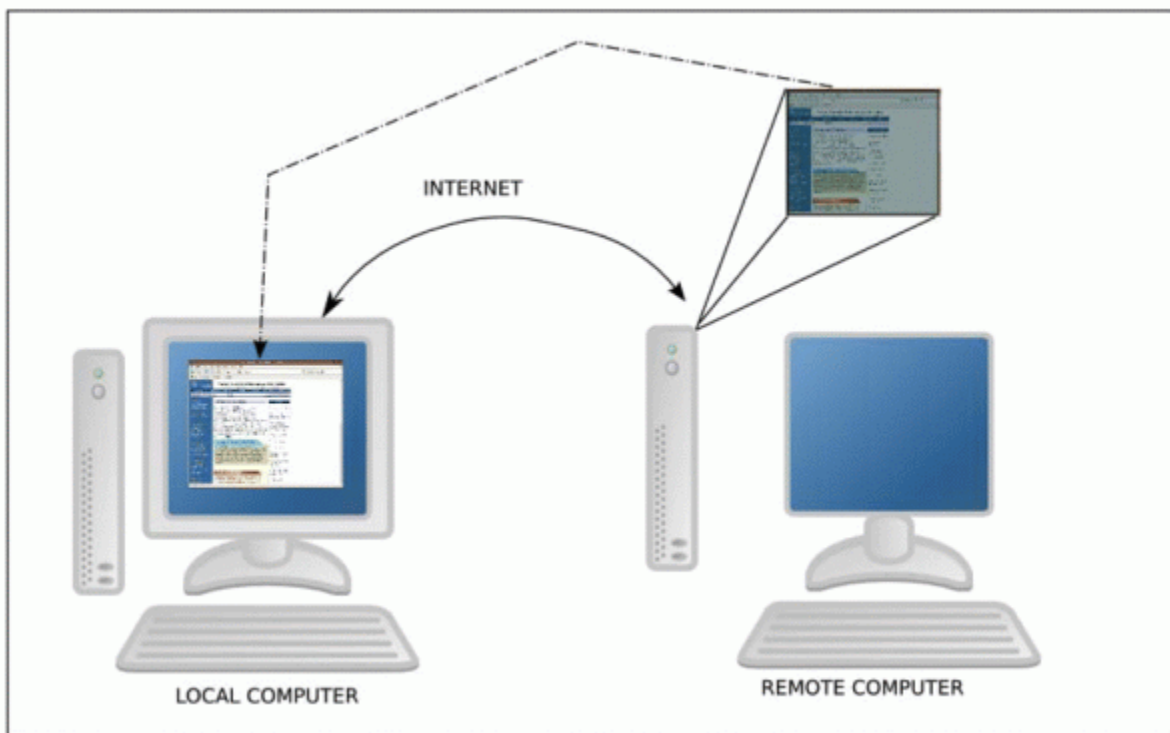
With Linux, remote directories can be mapped transparently into local folders of a computer. Actually, the whole user area can be mapped remotely. The user does not need to perform any tasks. Once logged in, all information stored by the user in his directories will actually be sent to the server. It is the system administrator's task to perform this mapping and this can be done for all users of the network. The sharing architecture can be changed without any need to inform the users. So, for example, a new server can be added to the network and some users' directories moved to the new servers without the need of notifying anyone else (fig. 1.2).

## Sharing processors

In the Windows<sup>®</sup> world some programs, when installed in the server, can be run remotely by users in some other computer on the local network. The user will only click on the program icon in the shared folder and the remote CPU will do the processing. However this feature has to be part of the program's code. Some programs



**Figure 1.2.** Sharing disks. The files that seem to be in a local disk are in fact in another computer - the user is unaware of this file sharing.



**Figure 1.3.** Users can log into other computers. The program will run on a remote computer, the screen is displayed in the local computer, while the data stay in the remote machine

will run remotely, others will still run locally. Most software in MS Windows<sup>®</sup> will not run remotely because they were originally conceived to only run locally.

In the Linux world, anyone can run some software in a remote computer, provided he/she has an account name and a password in the remote computer. Actually, logging in a remote computer is a standard procedure for Linux users. It is up to the user to decide which the best computer to run the software is (provided, of course, that the software is installed in that computer and that the user has an account there). There is no restriction imposed on the location of the computer. You can run software in your computer in Sri Lanka while visiting a research lab in Romania, with no need for special Linux configurations or installation. This is called in the Linux world a "remote login". The local computer serves only as a terminal, the local disk is not used, only the keyboard, mouse, and screen (fig. 1.3).

## 1.1.2 The graphical interfaces of Linux

After the advent of windowing systems, first popularized by the Apple Macintosh<sup>®</sup> computers in the 1980's, all other computer systems developed similar graphical user interfaces. In the beginning the differences were significant, but most systems seem now to have converged to very similar layouts. Linux, differently from MS Windows<sup>®</sup> and Apple Macintosh<sup>®</sup>, does not have only one layout, but in fact many different options of graphical interfaces. This means that the system that manages the graphical interface is separate from the operating system. The user can choose from a series of *window managers*, the one that best fulfills his/her needs. Another advantage is that Linux users can upgrade their window managers to newer versions without having to install a new version of Linux.

In this section we will introduce the *K Desktop Environment* (KDE), a very popular windows manager in Linux that is probably the one most similar to MS Windows<sup>®</sup>. In this section we will use Windows<sup>®</sup> as a base reference for describing KDE. However, we expect Macintosh<sup>®</sup> users to be also able to follow the description as well.

### The KDE Windows<sup>®</sup> manager

**KDE's taskbar and main menus.** KDE's graphical interface is probably the Linux's most user-friendly one. It is very similar in appearance to the MS Windows<sup>®</sup> interface with some characteristics of MacOS. Figure 1.4 shows a screenshot of KDE interface just after logging in. In the bottom of the screen we have the "taskbar", similar in spirit to the one implemented on MS Windows<sup>®</sup>, but with some particularities. The task bar is in the bottom of the screen, the "start" pop-up menu is activated by clicking the icon with letter "K".

We encourage the reader to experiment the other buttons. Moving the mouse over the various buttons will cause a description to pop up.

The number and names of programs available at the menu can vary immensely, depending on what is available in your version of Linux.

**Windows<sup>®</sup> software counterparts** Most programs available to Microsoft users have a similar counterpart in Linux. We will list below some of the most popular applications for Windows<sup>®</sup> and one counterpart in the Linux world.

**Windows<sup>®</sup> Explorer** - A file browser similar to the Windows<sup>®</sup> Explorer is Konqueror. With Konqueror, users can browse directories by clicking in the folder icons and by using the navigation buttons on the top part of Konqueror's windows. Images, sound files, text files and movies are automatically opened using appropriate applications. There is, however, one important difference; Konqueror can also be used as an Internet browser. When the application is initially launched, a help page is displayed in the browser. Internet browsing is started by typing a valid web address in the *location* text field, and file browsing is activated by clicking on the *home* icon, which opens up the user home directory in the local disk (equivalent to the My Documents folder of Windows<sup>®</sup> XP). A Google search in the Internet can be initiated directly by clicking the keywords in the search text box on the top right corner. Two interesting features of this browser are the ability of opening many "tabs", each one



**Figure 1.4.** KDE's graphical interface

with a different display (e.g. an internet address or directory). This facilitates copying files from one directory to another without the need of opening two windows.

**Internet Explorer** Even though Konqueror can be used to browse the Internet, there is another application that looks more similar to Microsoft's Internet Explorer: Mozilla Firefox. Firefox is the successor of Netscape, an Internet browser that for many years was Internet Explorer's main competitor. Like Konqueror, Firefox has a specific text box in the upper right corner that can be used to directly initiate a web search. Differently from Konqueror, this search can be performed not only by Google, but also by other search engines such as Yahoo, Amazon and Ebay (try clicking in the icon depicting a magnifying lens). Users can even add other search engines to the menu. There is also a version of Firefox for MS Windows<sup>®</sup>.

**Outlook Express** E-mail can be managed in Linux using Mozilla Thunderbird. As with the Internet browser, the interface of the e-mail client Thunderbird is similar enough to that of Outlook Express to make any lengthy descriptions unnecessary.

A MS Windows<sup>®</sup> version of Thunderbird is also available.

**Microsoft Office** Microsoft Office's users can perform similar tasks in Linux using two distinct Linux packages: OpenOffice and KDE Office. Programs of both packages can open and save files in Microsoft-compatible formats (.doc, .ppt, .xls).

- **OpenOffice.org** This system, developed in the Java language by Sun Microsystems can be also installed in MS Windows<sup>®</sup> systems. The main programs available are (more details can be found at <http://www.openoffice.org>):
  - Write - a word processor, similar to MS Word
  - Calc - a spreadsheet similar to MS Excel



Figure 1.5. Konqueror

- Impress - a presentation manager similar to MS PowerPoint
- Base - a database manager similar to MS Access

**KDE Office** - This system is part of the KDE window manager package. There are also Windows<sup>®</sup> versions of all programs. The main programs available are (more details can be found at <http://www.koffice.org>):

- KWrite - a word processor very similar to MS Word,
- KSpread - a spreadsheet similar to MS Excel
- KPresenter - a presentation manager similar to MS PowerPoint
- KExi - a database manager similar to MS Access

**Notepad** Notepad is a text editor. This is different from MS Word, which is a text processor, that is, a program to create formatted written documents, not just plain text. Many people in the Windows<sup>®</sup> platform use MS Word to edit both formatted documents and unformatted text. Text editors are much faster and lightweight than word processors. In addition, if your file is not a .doc document, it is just easier to use Notepad directly. The main problem with notepad is the absolute lack of nice editing features. The Linux operating system is proficuous in text editors. Probably the most flexible text editing system is Emacs (or Xemacs, a slight different version). Emacs provides all the basic text editing functions but is also a configurable editor and can be used to read e-mail, run programs, compile computer programs. A much simpler alternative to Emacs is Kate, very similar in spirit to Notepad.





Figure 1.6. Firefox

## Other Graphical User Interfaces (GUIs)

One of the nice characteristics of Linux is the extent to which a user can customize his installation. Different from other operating systems for personal machines, Linux has many windows managers. We have presented KDE, but there are other options. The most popular alternative to KDE is Gnome. Gnome has a cleaner interface but still has the same working principles, with a main taskbar and pull up menus. Other options of window managers do not include a taskbar, just pop-up menus prompted by the mouse buttons. These include WindowMaker, IceVM, and others.

### 1.1.3 Command Line

Windows systems, or graphical user interfaces (GUI) are very convenient to perform some tasks with just a few clicks of the mouse. Nevertheless, there are many situations where a GUI is not the best choice.

For instance, if you have to run the same program on a large set of files, which are in different directories, using just the GUI is very cumbersome and error prone. You would have to move the mouse all the time and find the right icons for the files you want, one at a time. Some specific programs have been developed to simplify these monotonous tasks, but this is not the general rule, and they do not cover all the possible cases.

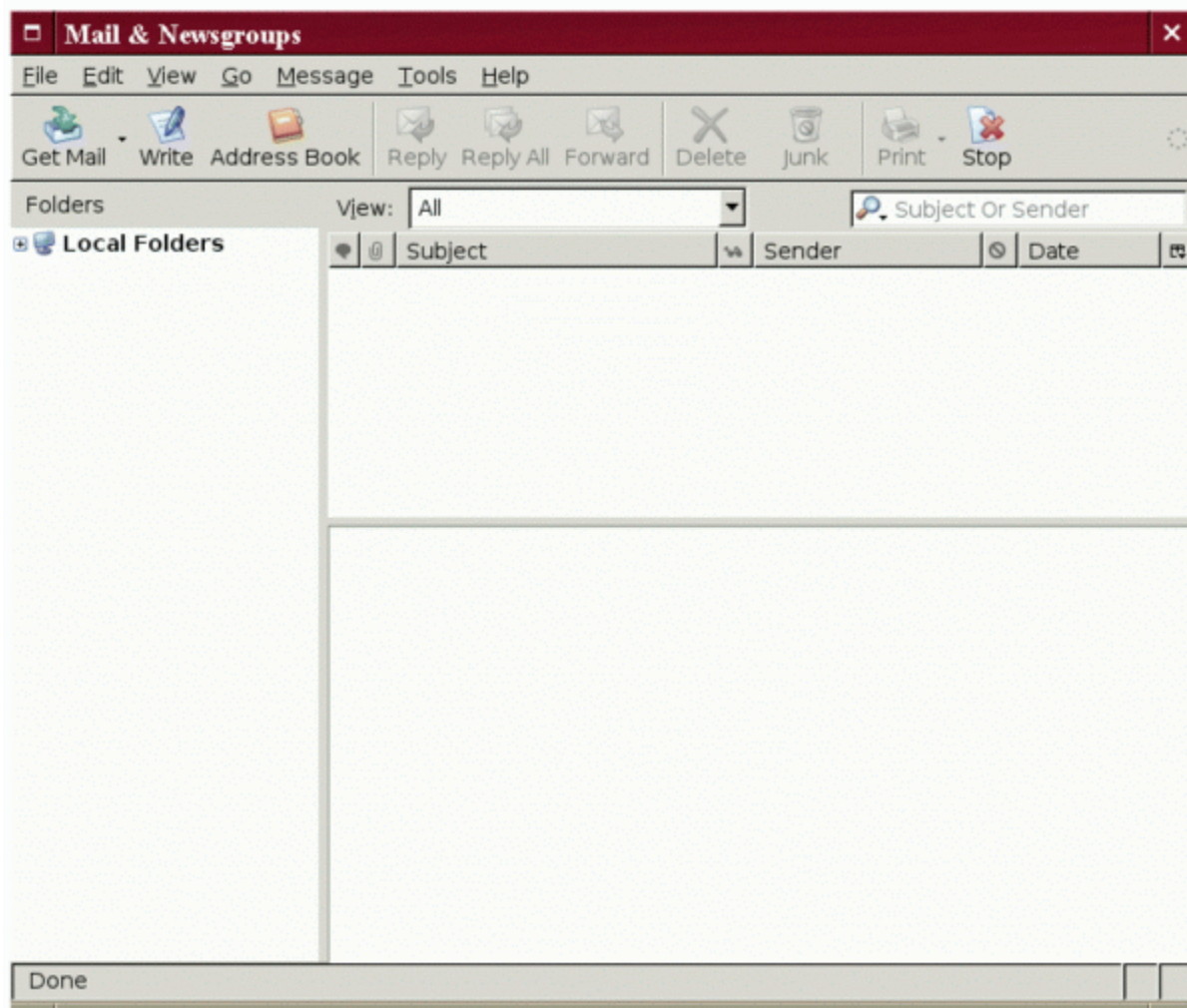


Figure 1.7. Thunderbird

Another situation is when the programs you are running take a long time to run, and you need to check the output or success of the execution before proceeding. You would need then to check the screen all the time to see if anything went wrong and take the appropriate decision.

Basically, although a GUI is very good for interactive processing, it is hard to automate tasks, especially if they involve repetitive tasks.

The alternative is using a *shell*, a text interface consisting basically of a virtual terminal where you type the tasks to be done using a set of commands and some special features, presented later in this chapter. Writing the commands gives you the ability to specify the tasks with much more detail, and to perform job control — like repetition and error checking, since the interface has a language specially crafted for this purpose.

Even though you have to type the commands, you do not lose the interactivity. In fact, you can launch graphic applications and collect their results without any problem. Mastering just the basics of command line interaction gives you a noticeable increase of productivity for the following reasons:

- The possibility of writing complex commands, easily coupling the execution of many applications.
- The simplicity to customize execution. It is more complicated in a GUI to state all possible execution patterns, and, if you want to change the execution pattern or the execution parameters, you will have to

open a 'preferences' window and re-select a series of many buttons. Using command line, you just need to type the correct options directly.

- You can save the command lines you used in a text file and run them again later. You can even include conditions that change the execution behavior when needed.
- It is very easy to create "combo" applications by connecting existing ones without the need of further programming.

The graphic user interface provides an abstraction at a higher level. As a consequence, it is possible to find solutions to standard situations much faster, but does not allow detailed control of the application. When a finer control is needed, in many occasions it is not possible to stay at an abstract level.

## 1.2. The Shell and Command Line Interaction

### 1.2.1. Some initial concepts

#### Shells: bash

To interact with the computer, you need a program which can understand your requests and translate them into actions to be executed by the machine. On most current machines this is done by a graphical interface with menus and icons: if you want to run a program, just click or double-click on the corresponding icon. In the same fashion, if you want to delete a file, just drag the corresponding icon to a trash can.

The alternative is a *text* interface called a *shell*, where the commands are typed and the results are presented in text form as well. In fact, this was the standard method of interaction with computers before the graphic interface became possible that, due to its strength, is the main method on UNIX systems.

In essence, a shell is a program runner which allows the user to control the execution, direct results, combine programs and even develop small applications in a very quick way. In this section we will present some concepts needed to use the Linux shell.

**A single directory tree: no device names** In the MS Windows<sup>®</sup> system each physical device (CD-ROMs, hard disks, USB drives, etc.) is identified by a letter of the alphabet. It is common, for example, to have the main hard disk under "C:", the CD-ROM drive under "D:" and so on. The complete address of a file looks similar to

```
C:\Documents and Settings\John\My Documents\Research\thesis.doc
```

Linux uses a different approach. There are no device letters, so in fact the user is unaware of the physical location of his/her files. All directories and files are under a single root directory. The structure of the directory hierarchy is controlled by the administrator. Thus, the complete address of any file starts at the root directory, as follows in the example below:

```
/home/employee1/Research/thesis.doc
```

This has many advantages. In particular, users do not need to know of any changes in the systems' devices. For example, if a project turns out to use lots of disk space, we can buy a new hard disk and assign to this new disk the home directory of the project, copying all the old information into the new disk without the users ever knowing what happened. Their information will be at the same address, even if it may be now being stored in a different hard disk.

**The PATH environment variable: finding programs** As in MS Windows<sup>®</sup>, programs can be installed in any directory in Linux. In the Linux bash shell, you can tell the system which directories can contain programs.

When the user types a command, the system automatically searches in the directories specified by the user. The places where the shell looks for programs are specified in the variable called PATH (a variable is a place where the system stores information). In all Linux installations some directories are already part of the PATH. These are generally in /usr/bin and /usr/local/bin. Almost all standard Linux applications are located in these two directories. However, any user can add new directories to his/her path and run other programs without having to search for them. You can even download a new version of a program in the system and run your version by default, as opposed using the one available to the other users. To do this, you only have to include the directory where you stored the program in the PATH system variable **before** the system directories.

**Auto completion** Another nice feature of the bash shell is auto-completion. It is normally very tiresome to type complete file addresses or even program names. In the bash shell, when the user starts typing and presses the <TAB> key, the system tries to complete the typing. If it is a program name (first word to be typed), bash searches the directories in the path to find which programs start with the letters typed. Otherwise bash looks in the partial path specified by the user to search for completions. If there is only one possible completion, pressing the "tab" key (we will refer to it as <TAB>) will cause the appropriate word to appear, if there is more than one completion, a sound will be emitted. Pressing <TAB> the second time will display all possible completions.

**Job control:** When using command line, it is important to understand the concept of "process". A process is a running program. There can be many processes running the same program (for example, when you open more than one Windows<sup>®</sup> Explorer window). A process can be in two states: *running* and *suspended*. When a process is *suspended*, no work is performed nor lost. A running process just executes the program tasks normally. Processes can be *killed*, that is, the user can determine that the processing will be interrupted and end in a certain moment (this is equivalent to click in the **X** icon of a window).

**History** The shell maintains a list of all the previous commands that the user has typed. This list is called the "history". The history can be useful to reissue long commands or to confirm which programs were executed and in which order.

**Data flow:** Linux programs, by default, get input from the keyboard and send output data to the screen. The keyboard is the *Standard Input*, or STDIN, and the screen is the *Standard Output*, or STDOUT. All basic shell commands use this principle. A nice feature of command line interactions is that you can have the output of a program in STDOUT be fed directly to the STDIN of another program, creating a UNIX *pipe*. For example, suppose you have two programs: one that reads sequences from a multi-FASTA file and outputs only the sequence names, the other that reads lines and order them alphabetically. Both programs are useful on their own, but if you connect the output of the first program (selecting only the sequence names) to the input of the second (sorting alphabetically), you can read a multi-FASTA file and have as a result the sequence names, ordered. These two programs exist in Linux and this task can be performed, as we will see later. It is important to note that, for pipes to work correctly, the second program needs to be able to read the output of the first program, that is, the format of one program's output has to be compatible with the other program's input.

We have seen some concepts that are important to keep in mind when we start learning about the Linux commands. The next section will present a set of important Linux commands in a learn-by-doing style.

## 1.2.2 Learning by doing

In this section we will explain the main Linux commands and actions through a tutorial. At each step a task will be proposed and you will be taught how to perform that task. We strongly recommend that you read this chapter while performing the actions in the companion Linux system. This system will run in almost any PC and you do not need to modify your computer to do it. Also, the system will have all the files and the environment needed to follow the tutorial. To use the companion CD please follow the instructions below:

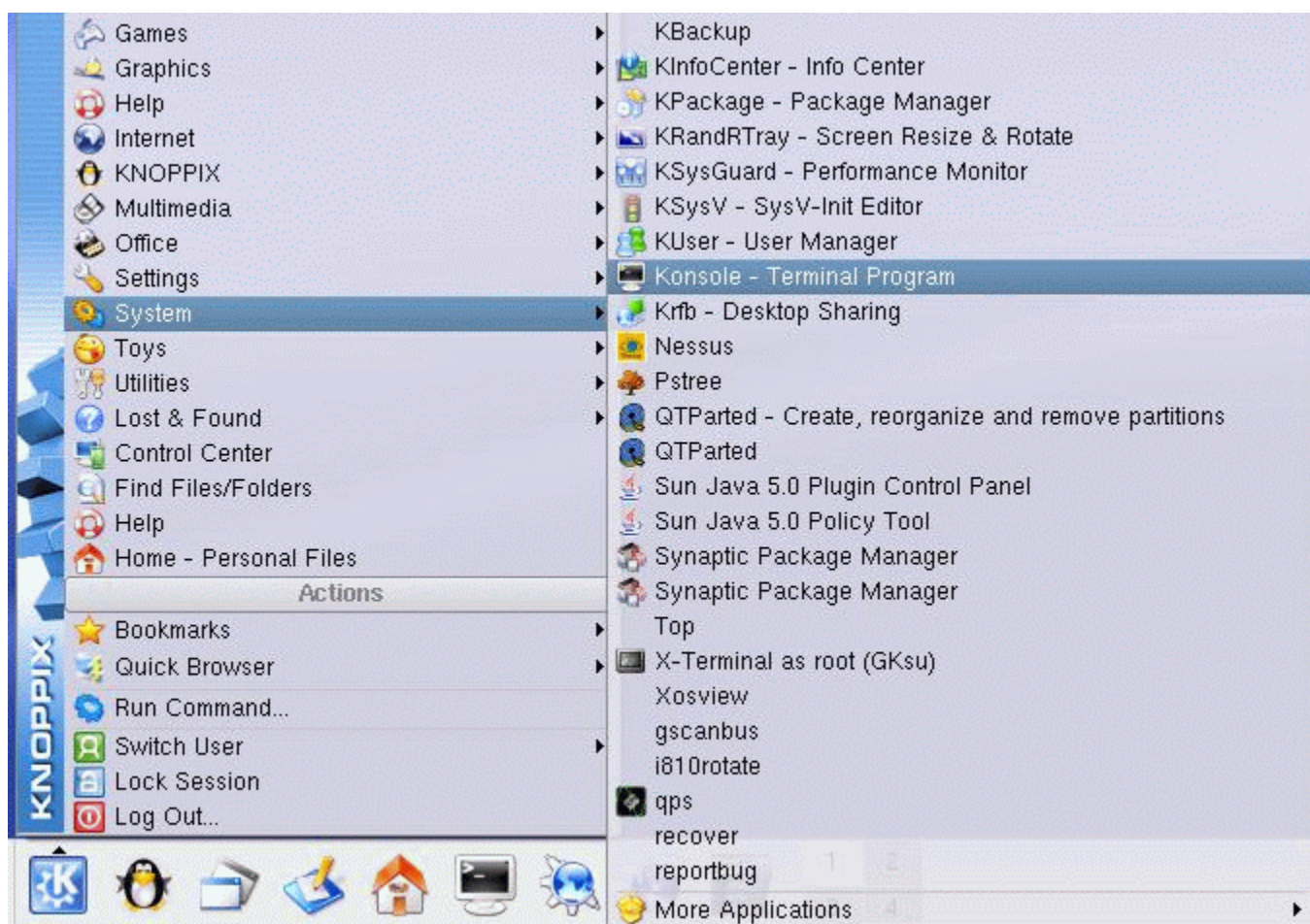
1. Download the CD ISO image from one of the two sites below (whichever is accessible):  
<http://clinmaldb.usp.br/bioinfobook/bookDVD.iso>  
<http://www.coccidia.icb.usp.br/bioinfobook/bookDVD.iso>
2. Follow the instructions described at one of the two sites below (whichever is accessible):  
<http://clinmaldb.usp.br/bioinfobook/instructions.html> :  
<http://www.coccidia.icb.usp.br/bioinfobook/instructions.html>
3. After following all the instructions you should have your computer running our version of Knoppix, ready for the tutorial.

## Logging in

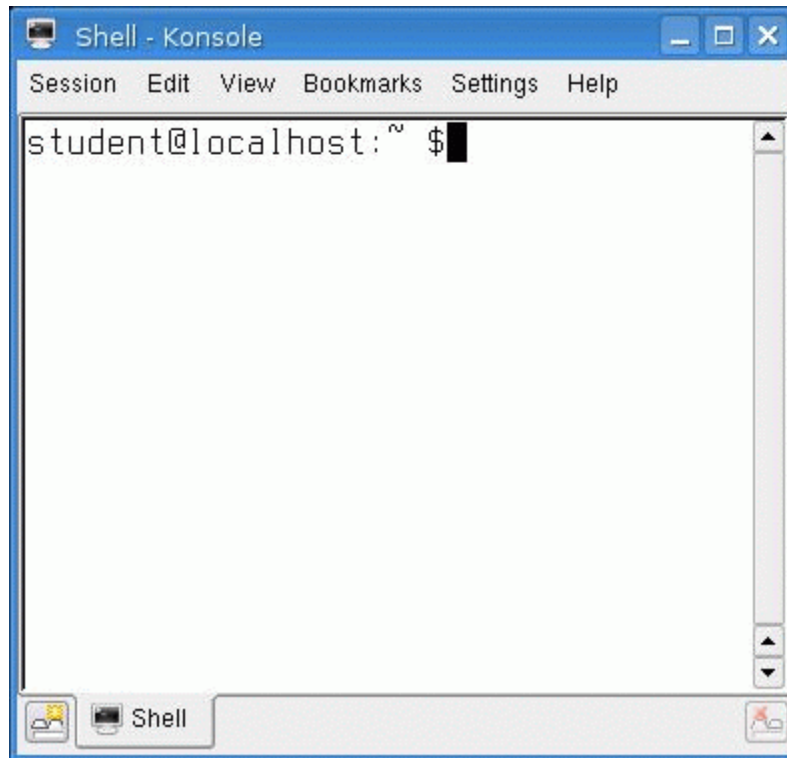
Since Linux is a multiuser system, access is controlled. Before you start doing any work, you have first to log in. For this purpose, you should type your username and password in the presentation screen (see the figure below). For this tutorial, you should log in as the user *student*, with the password *student*. After logging in, the initial screen for the KDE environment will appear



**Launching a shell** In this tutorial, we will be working almost entirely from the command line. To use the shell, you need to start a terminal window. Click on the terminal icon in the bottom bar. Alternatively, click on the icon with a K letter (similar to the MS Windows<sup>®</sup> Start button) on the left bottom corner of your screen. This will open up a menu. Select 'System' and then 'Konsole - Terminal Program' (see figure below illustrating the second option):



A window will pop up. This window is almost entirely blank, with only one line being displayed. What you see is the *prompt*. The prompt indicates that the shell is ready to receive a command. The prompt also displays some useful information. You should have the following prompt: `student@localhost:~:$` The first part of the prompt is the username ("student"), followed by "@" and the machine name ("localhost"), ":", the current directory (in our case, "~", the home directory), and finally "\$". You can see the shell window below (note colors in the shell may vary):



## Creating a work environment (a simple directory structure)

When you first log in, you are at your *home directory*. Each user has his/her own home directory. This directory can be located anywhere in the system, but in most Linux systems it is under the */home* directory. In our case, our home directory is */home/student*. If you use Konqueror to inspect your home directory (K menu, Internet submenu, Konqueror entry; type the "home" icon.) you will see the directory is initially empty. We could create a directory structure directly from Konqueror, but we will use the shell command line instead.

For our tutorial we want the following subdirectories:

- recentFiles
- results
- internetSite

**Task:** Create three subdirectories: *recentFiles*, *results* and *internetSite*.

**Comments:** To create directories in the command line you use the command *mkdir* (from MaKe DIRectory). The only thing you need to specify is the new directory's name. The command can be used to create one or more directories.

**Issuing the command:** We will create the new directories using *mkdir* twice, once to create a single directory, and once to create the remaining two. You should type the following two lines in the shell

```
mkdir recentFiles
mkdir results internetSite.
```

If you typed the command correctly, now you should have three subdirectories in the */home/student* directory.

**IMPORTANT NOTE:** In most Microsoft Windows<sup>®</sup> systems, the computer does not discriminate between upper and lower case letters, but Linux and UNIX do. So, throughout this tutorial, be careful not to change uppercase letters for lowercase letters and vice-versa, as the commands will not work.

## Paths and the current directory

Similarly to when you are in Windows<sup>®</sup> Explorer, in the shell, at any given time, you are inside some directory. When the shell is initially created, you are at your *home directory*, in our case `/home/student`. This is called your *current directory*. You can refer to any file in your current directory by using only the file's name. To refer to any other file that is not located in your current directory, you should give a *path*, that is, directions on how to navigate either from the current directory or from the root directory to the desired file. Paths that start in the current directory are called *relative paths*, and the ones that start in the root directory are called *absolute paths*. Absolute paths always start with `/`, otherwise you have a relative path. You can change the current directory using the `cd` command followed by a path to a directory. Examples of the command:

- `cd /home/instructor` - absolute path
- `cd Desktop` - relative path
- `cd localdirectory/anotherdirectory` - relative path

There are some important nicknames for specific directories that you should remember (we list them inside quotes below):

- `"."` - denotes the current directory
- `"~"` - denotes your home directory
- `"~username"` - denotes the home directory of user *username*.
- `".."` - denotes one directory up, that is the directory on which your current directory is.

Please remember these nicknames, as they will be used below.

## Creating a local copy of an Internet site

**Task:** Copy the hypertext (HTML) version of the tutorial and FAQ of this chapter into your local machine's *Tutorials* directory.

**Comments:** A very useful Linux program is `wget`. With this program you can mirror a whole Internet site in your computer with just one single command. `wget` works also if the site requires a password. To use `wget` you need to know the address of the main page you want to copy. If the page is password-protected, you also need a valid username and password. Using `wget`, you can either copy only the page, all the links, all the links in the links, and so forth. You can also copy only the pages that refer to the links in the same Internet server (so you can, for example, avoid copying advertising links from a commercial page).

**Issuing the command:** First we need to go to the *internetSite* directory.

```
cd internetSite
```

Now we should copy the pages of an internet site. The main internet page for this tutorial is located at

[http://www.coccidia.icb.usp.br/linux\\_tutorial/index.html](http://www.coccidia.icb.usp.br/linux_tutorial/index.html).

To copy the site completely we will call `wget` with the "mirror" option (`-m`) and with the `-nH` option<sup>3</sup>. Therefore, to copy the site completely you need to write

---

<sup>3</sup> If `wget` is invoked without this option, it will create the directory path `www/bionfo/org/br/linux_tutorial/index.pl`. within the *Tutorials* directory.



```
wget -nH -m http://www.coccidia.icb.usp.br/linux\_tutorial/index.html
```

If you typed the command correctly, not only the file *index.html* will be copied into your directory, but also all other files related to the links of this file. This means that all the pages addressed in the links, as well as the pages these pages point to will be downloaded. If you use your local browser to open the file *index.html* directory, you will now be able to find your local copy of the tutorial page. Try the links and you will see that all the respective pages have now link referring to local addresses.

**Important** - When you type a very long command line, some parts of this command may be typed incorrectly, in which case your command will not work. If you make a mistake, it will probably fall in one of the categories below:

- You typed wrongly one of the options. In this case, a long text displaying all the correct options of the command will appear.
- You typed a wrong internet address (the first part of the http address: "<http://www.coccidia.icb.usp.br>") – in this case the error message should have the text

Resolving *theActualAdrrssYouTyped* ... failed: Name or service not known.

- You typed the second part of the address wrongly (after the first single "/", that is, "Linux\_tutorial/index.html"). In this case the error message should include the text:

*HTTP request sent, awaiting response... 404 Not Found:*

**Sites with passwords:** Many sites in the internet require passwords to show some part of their material. Wget can retrieve pages protected by passwords. If you know a valid user and its password you can use the options – *http-user=* and *-http-passwd=*. The book site has a part that is password protected. Try the command

```
wget -nH -m http://coccidia.icb.usp.br/linux\_tutorial/protected/protected\_page.html
```

You will get back an error message stating that you need a password to access the page:

*HTTP request sent, awaiting response... 401 Authorization Required*

*Authorization failed.*

To retrieve the information you can use the user *bookreader* with the password *bookworm*. Let's try (**warning:** the command written below should be typed in a single line, we use two for lack of space):

```
wget -nH -m --http-user=bookreader --http-password=bookworm http://coccidia.icb.usp.br/linux\_tutorial/protected/protected\_page.html
```

Now, if you typed correctly, you should have retrieved the *Linux\_tutorial* directory in your home directory. Inside this directory you will find the file *protected\_page.html*.

Important: this is an even longer command line than the previous one. Apart from all the errors previously described, another possible error can occur if you typed either the user or the password wrongly. In this case the error will be the same as in the previous one:

*HTTP request sent, awaiting response... 401 Authorization Required*

*Authorization failed.*

## Finding files in a local directory

**Task:** Search at the directory /home/tutorial/sequences and its subdirectories, looking for all files with the extension ".fasta"

**Comments:** Searching for files with specific names or parts of the name is a very common task. Very frequently users download some files from the internet and then realize later that they are not located where they were supposed to be. Command-line UNIX has a file search program similar to the "search" facility of Windows<sup>®</sup> Explorer. Konqueror also have a similar facility. However, using the command line can be much faster than opening a browser and navigating through a set of menus. To perform this task, the UNIX shell provides the program find. In this program, the user specifies the name of the file to be searched and the starting location point of the search (a directory). As a result, find will show the path of all the files matching the request. You can either specify the complete file name or use wildcards to specify only parts of the name.

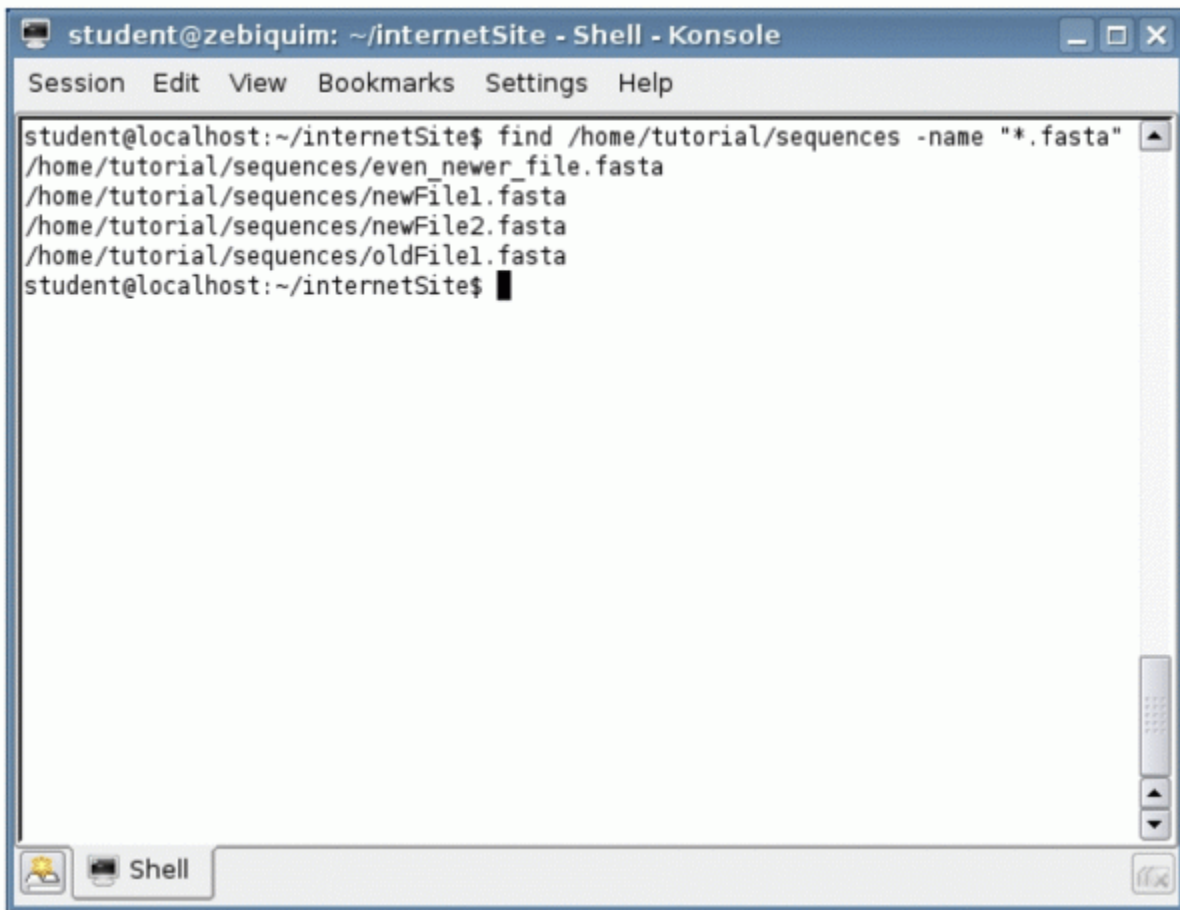
**Issuing the command:** To find a file, we use the find program. This program needs two parameters, the directory on which to start the search, and the name of the file to be searched. You can use wildcards to describe filenames, where these wildcards may include the following characters:

- \* — This wildcard designates any character or series of characters, or even no character at all. In our case, writing \*.fasta indicates names that start with any characters whatsoever, but necessarily end with ".fasta".
- ? — The question mark will designate any single character. Writing "?asta" will match "fasta", "pasta", ".asta", and so on.
- [list of characters] — This expression works just like the '?' wildcard, but instead of indicating *any* character, it will match one of the characters in a list. A range may be indicated if you use a '-'. For instance, "[fp]asta[0-9]" will designate "fasta0, fasta1, fasta2, ..., fasta9, pasta0, ..., pasta9".

Wildcards can be used anywhere in the specification of a name to be searched. To perform the task described in this item you should type the command

```
find /home/tutorial/sequences -name "*.fasta"
```

The result should be the listing:

A screenshot of a Linux terminal window titled "student@zebiquim: ~/internetSite - Shell - Konsole". The window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The terminal content shows a user running the command "find /home/tutorial/sequences -name '\*.fasta'" which returns four file paths: "/home/tutorial/sequences/even\_newer\_file.fasta", "/home/tutorial/sequences/newFile1.fasta", "/home/tutorial/sequences/newFile2.fasta", and "/home/tutorial/sequences/oldFile1.fasta". The prompt "student@localhost:~/internetSite\$" is visible at the end of the output. The terminal window has a scrollbar on the right and a taskbar at the bottom with a "Shell" icon.

## [Checking file information, moving files]

**Task:** Check the date and size of the fasta files you found in the previous items, and copy the files that were created after December 2005 to the directory /home/student/recentFiles

**Comments:** There is a command in Linux just to list file names, ls, and a command just to copy files, cp. There are many options the user can specify when issuing the ls command. In particular, we can ask for a *complete* listing, where not only the names of the files are shown, but also their respective permissions, size, and date of the last modification. Also, the copy command (cp) can be used to copy one or more files. When we specify more than one file to be copied, the destination of the copy command needs to be a directory. In this case, all files will be copied under the same name as the original ones.

**Issuing the commands:** You need first to issue the ls command with the "-l" option (for *long*):

```
ls -l /home/tutorial/sequences
```

The result should as below:

```

student@zebiquim: ~/internetSite - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/internetSite$ ls -l /home/tutorial/sequences/
total 15
-rw-r--r-- 1 tutorial tutorial 476 2007-08-14 09:56 even_newer_file.fasta
-rw-r--r-- 1 tutorial tutorial 5890 2006-08-20 15:11 newFile1.fasta
-rw-r--r-- 1 tutorial tutorial 1247 2006-08-20 15:11 newFile2.fasta
-rw-r--r-- 1 tutorial tutorial 5660 2005-08-20 13:08 oldFile1.fasta
student@localhost:~/internetSite$

```

The files that were created after December 2005 are

- newFile1.fasta
- newFile2.fasta
- even\_newer\_file.fasta

Next you can issue a single copy command to copy all three files into the directory `/home/student/recentFiles`. Since we need to copy three files, it will be tiresome to type the complete path for all the files that we want to copy. The easiest thing to do is to "go" to the `/home/tutorial/sequences` directory, so we can only type the names of the files (that is, their *relative path*), thus avoiding to type long addresses many times. However, we need to use the complete path for the destination directory.

```
cd /home/tutorial/sequences
```

```
cp newFile1.fasta newFile2.fasta even_newer_file.fasta /home/student/recentFiles
```

Finally, to make sure that the files were copied correctly, check the "recentFiles" directory.

```
ls -l /home/student/recentFiles
```

Please note that in the listing, all files are listed as having been created today.

### [Inspecting file contents (more, less)]

**Task:** Inspect the files you just copied, checking if they are really all FASTA files.

**Comments:** Sometimes we need to give a "quick look" at a file, just to check its content. One option is to load it onto a text editor program (such as, for example, Notepad). However, in Linux there is an easier way of looking inside a file, the `more` command. This command, will display the content of a file on your terminal screen, one

page at a time (where "page" is exactly the current size of the terminal window). To go to the next page you just press the space bar, to move down only line only, type the enter key, to end the program, type the "q" key. There is a more sophisticated version of more called less (don't mind the name, it is "computer scientist humor"). Less has more options, like searching the text for specific strings and moving back in the text. More is always available in Linux systems, whereas less is not always present in all Linux distributions. For more details on less, check the Linux online manual.

**Issuing the commands:** To inspect a file, you need to type the more command, followed by the name of the file. To make things easier, we will first go to your recentFiles directory:

```
cd /home/student/recentFiles
```

You can then issue the more command to many files at a time, in this case, more will browse each file at a time. To see one file at a time (for example, newFile1.fasta), you should type:

```
more newFile1.fasta
```

Your terminal screen should show something like (if you did not resize your terminal):



```
kterm
>gi|56797977|emb|AJ871406.1| Plasmodium falciparum mRNA for Pdx2 protein
ATGTCAGAAATAACTATAGGGGTATTATCACTACAAGGTGATTTTGAACCGCATATAAATCATTTTATCA
AATTACAAATACCGTCGTTAAATATTATCCAAAGTAAGAAATGTTTCAATGATTTGGGACTATGTGACGGGCT
TGTAATTCDDAGGTGGAGAAATCCACAACGTACGTCGATGTTGTGCTTATGAAATGATACCTTATATAAT
GCTTTAGTACATTTTCATTTCATGTGCTAAAAAGCCAAATTTGGGGCCTTGTGCAGGTTGTATTCCTTAT
CTAAGAAATGTAGAAATATAAACTTTATAGCAATTTTGGAAATAAATTTTCTTTTGGAGGATTGGATAT
AACTATATGTAGAAATTTTATGATCACAAATGATAGTTTATATGCTCATTAAACATAATATCTGAT
AGTAGTGCTTTTAAAGAAAGACTTACAGCGGCCGTCATAGGGCACCTTATATAGAGAAATATTATCAG
ATGAAGTAAAGTACTTGTACATTTTCACATGAATCATATGGCCAAATATTATAGCAGCCGTTGAACA
GAATATTTGTTTAGGCACAGTTTCCATCCAGATTTATGCCACATACCGCTTTTCAACATATTTTTAT
GAGAAGGTTAAAAATTACAATATTCATAA
>gi|4507020|ref|NM_000342.1| Modified Homo sapiens solute carrier family 4, anion exchanger,
member 1 (erythrocyte membrane protein band 3, Diego blood group) (SLC4A1), mRNA, introduced
by Gubialan
CAGCGGCTGCAGGACTTCACCAAGGGACCCGAGGGCTCGTGAGCAGGGACCCGCGGTGCGGGTTATGCTG
GGGGTCAGATCACCGTAGACAACCTGGACACTCAGGACCCAGCCATGGAGGAGCTGCAGGATGATTATGA
AGACATGATGGAGGAGAACTCGAGCAGGAGGAAATGAAAGACCCAGACATCCCCGAGTCCCAGATGGAG
CCTCCCTGGCTGGAGTGGCCAAACCAACCTGCTAGACAGGTTTATCTTTGAGAGCCAGATCCGGCCTCAGGA
CCGAGAGGAGCTGCTCCGGGCCCTGCTGCTTAAACACAGCCACGCTGGAGAGCTGGAGGGCCCTGGGGGGT
GTGAAGCCTCAGTCCCTGACACGCTCTGGGGATCCCTCACAGCCTCTGCTCCCCAAGACCTCCCTCACTGG
AGACACAGCTCTTCTGTGAGCAGGGAGATGGGGCACAGAGGGGACTCACCATCTGGAAATCTGGAAAA
GATCCCCCGGATTCAGAGGGCCAGCTTGGTGTCTAGTGGGCCGCGCCGACTTCCCTGGAGCAGCCGGTGTG
GGCTTCGTGAGGCTGCAGGAGGCGAGCGGAGCTGGAGGGCGGTGGAGCTGCCGGTGCCTATACGCTTCCCTCT
TTGTGTTGCTGGGGACTGAGGGCCCCACATCGATTACACCCAGCTTGGCCGGGCTGCTGCCACCCCTCAT
--Mais--(31%)
```

Try typing the "enter" key and the space bar to inspect the file and see what happens. Another option is to inspect more than one file with a single command such as:

```
more newFile1.fasta newFile2.fasta even_newer_file.fasta
```

Initially, you have almost the same output as in the previous command, but when the first page of the file is displayed, before the actual contents, there are five lines indicating the name of the file being displayed:

```

kterm
newFile1.fasta
>gi|56797977|emb|AJ871406.1| Plasmodium falciparum mRNA for Pdx2 protein
ATGTCAGAAATAACTATAGGGGTATTATCACTACAAGGTGATTTTGACCCGCATATAAATCATTTTATCA
AATTACAATACCGTCGTTAAATATTTAATCCDAGTAAGAAATGTTTCATGATTTGGGACTATGTGACGGGCT
TGTAATTCAGGTGGAGAAATCCACAACGTACGTCGATGTTGTGCTTATGAAATGATACCTTATATAAT
GCTTTAGTACATTTCAATTCATGTGCTAFAAAGCCCAATTTGGGGCACCTTGTCAGGTTGTATTCCTTAT
CTAAGAAATGTAGAAATATAAACTTTATAGCAATTTTGGAATAAATTTCTTTTGGAGGATTGGATAT
AACTATATGTAGAAATTTTATGGATCACAATATGATAGTTTTATATGCTCATTAAACATAATATCTGAT
AGTAGTGCTTTTAAAGAAAGACTTACACGCGGCCGTCATAAGGGCACCTTATATAAGAGAAATATTATCAG
ATGAAGTAAAGTACTTGCATACATTTTCACATGAATCATATGGCCCAATATATAGCAGCCGTTGAACA
GAATAATTTGTTTAGGCACAGTTTCCATCCAGAAATTTGCCCACATACCGCTTTTCAACATATTTTTAT
GAGAAGGTTAAAAATTAACAATATTCATAA
>gi|4507020|ref|NM_000342.1| Modified Homo sapiens solute carrier family 4, anion exchanger,
member 1 (erythrocyte membrane protein band 3, Diego blood group) (SLC4A1), mRNA, intruded
by Gubialan
CAGCGGCTGCAGGACTTACCAGGGACCCGAGGGCTCGTGAGCAGGGACCCGCGGTGCGGGTTATGCTG
GGGGCTCAGATCACCGTAGACAACCTGGACACTCAGGACCACGCCATGGAGGAGCTGCAGGATGATTATGA
AGACATGATGGAGGAGAAATCTGGAGCAGGAGGAATATGAAGACCCAGACATCCCCGAGTCCCAGATGGAG
CCTCCCTGGCTGGAGTGGCCAAACCAACCTGCTAGACAGGTTTATCTTTGAGAGCCAGATCCGGCCTCAGGA
CCGAGAGGAGCTGCTCCGGGCCCTGCTGCTTAAACACAGCCACGCTGGAGAGCTGGAGGGCCCTGGGGGGT
GTGAAGCCTGACGTCCTGACACGCTCTGGGGATCCCTCACAGCCCTGCTCCCCAACACCTCCACTGG
AGACACAGCTCTTCTGTGAGCAGGGAGATGGGGCACAGAGGGCACTACCATCTGGAAATTCGGAAAA
--Mais-- (27%)

```

Also, after you reach the end of the first file, the contents of the next one will appear (try hitting the spacebar until the next file appears). Please note that, at the last page of a file, there is a text in the bottom of the page indicating the next file to be displayed.

After you finish your inspection, you will see that the file "even\_newer\_file.fasta" is not actually a FASTA file, but rather contains a normal text.

After quitting more, try the less command for file *newFile1.fasta*. Please note that you can move backwards using the "page up" key. Also, the up arrow and down arrow keys will work accordingly.

```
less newFile1.fasta
```

Use less only for a single file, using it for more than one file at a time is considerably more complex.

## [Changing file names]

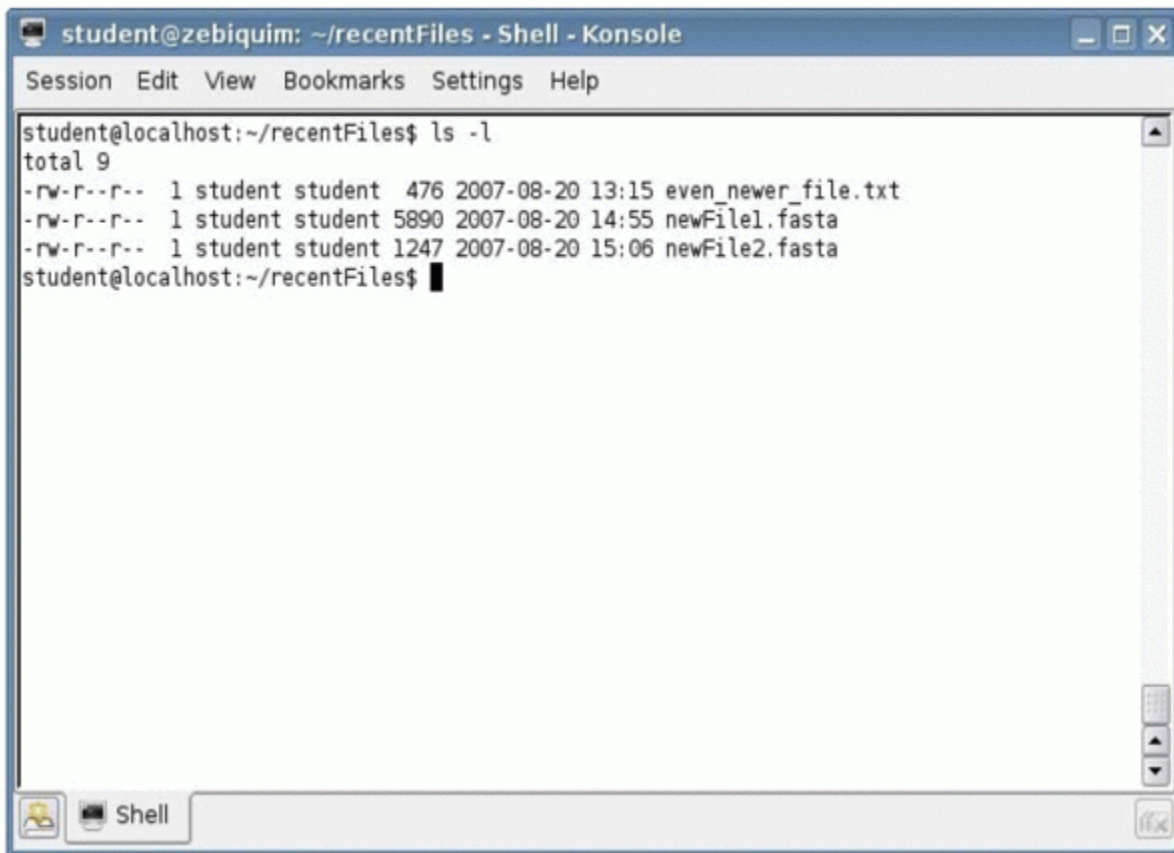
**Task:** In the previous task, we inspected three files and found out that the file "even\_newer\_file.fasta" is misnamed and does not contain genetic sequences in FASTA format. Now we will change the file's name to "even\_newer\_file.txt".

**Comments:** In UNIX there is no "rename" command. Instead, the "move" command, named mv, is used. This command can either be used to change a file's location or be used to simply change a file's name. In this command, the user specifies the old name and the new name of a file. Either the old name or the new name can be relative paths or complete paths. The effect of the command is both renaming and moving the file contents.

**Issuing the commands:** For this task, simply type:

```
mv even_newer_file.fasta even_newer_file.txt
```

Now if you list your files (using the "ls -l" command) you will see that your file had its name changed. Please note that the date of the file remains unchanged (important: the files' dates will be your current date and will not match the figure below).



```
student@zebiquim: ~/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ ls -l
total 9
-rw-r--r-- 1 student student 476 2007-08-20 13:15 even_newer_file.txt
-rw-r--r-- 1 student student 5890 2007-08-20 14:55 newFile1.fasta
-rw-r--r-- 1 student student 1247 2007-08-20 15:06 newFile2.fasta
student@localhost:~/recentFiles$
```

## [Checking files for specific content]

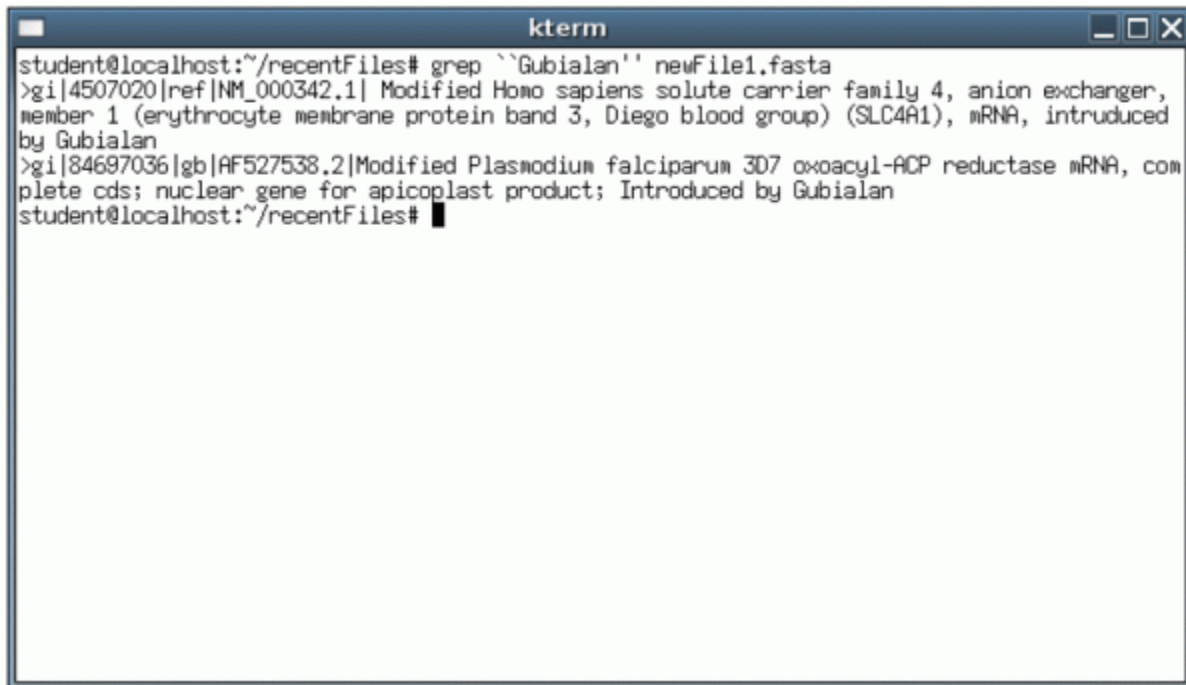
**Task:** Find all FASTA sequences whose headers contain the word "Gubialan".

**Comments:** There is a very powerful search program in UNIX called `grep`. This program is used to inspect the contents of single or multiple files looking for specific patterns of characters. The program examines a file and prints all the lines that contain the searched pattern (in our case, the word "Gubialan"). If more than a single file is specified in the search, the name of each file is printed before each line, so the user can easily identify which lines came from which files. We will use both forms in this exercise

**Issuing the commands:** If we want to search for the word "Gubialan" in the file `newFile1.fasta` we should type

```
grep "Gubialan" newFile1.fasta
```

This file contains two entries where the name appears in the header. The output of this command should be as displayed below:



```

student@localhost:~/recentFiles# grep `Gubialan` newFile1.fasta
>gi|4507020|ref|NM_000342.1| Modified Homo sapiens solute carrier family 4, anion exchanger, member 1 (erythrocyte membrane protein band 3, Diego blood group) (SLC4A1), mRNA, introduced by Gubialan
>gi|84697036|gb|AF527538.2|Modified Plasmodium falciparum 3D7 oxoacyl-ACP reductase mRNA, complete cds; nuclear gene for apicoplast product; Introduced by Gubialan
student@localhost:~/recentFiles# █

```

We could proceed and reissue the command for the other two files, substituting "newFile1.fasta" and by "newFile2.fasta". However, we can do this all in one single command line:

```
grep "Gubialan" newFile1.fasta newFile2.fasta
```

Notice that we now have many lines, all preceded by the file names.

A third way of issuing this command is to use "wildcards". Wildcards are characters with special meaning in the shell command line. One that is particularly useful is "\*". Remember that, when "\*" appears in the middle of the command lines, it stands for "any characters, any length". When this character appears, the Linux shell verifies which are the possible completions for the expression and expands them inside the command line. In other words, if we type

```
grep "Gubialan" newFile*.fasta
```

The meaning of "newFile\*.fasta" is

"any existing file with a name starting with 'newFile' and ending with '.fasta'".

Therefore, we will have the same command line as before, since "\*" can in this case be successfully substituted by "1" and "2". Similarly, you can type an even shorter command line:

```
grep "Gubialan" newFile*
```

In this case, Linux will check every possible completion and issue, once again, the command. This time, "\*" is substituted by "1.fasta" and "2.fasta". One important note: if there are other files with names starting with "newFile", they would also have been included in the command line.

## [Checking file sizes]

**Task:** Check the number of lines of each of the three FASTA files.

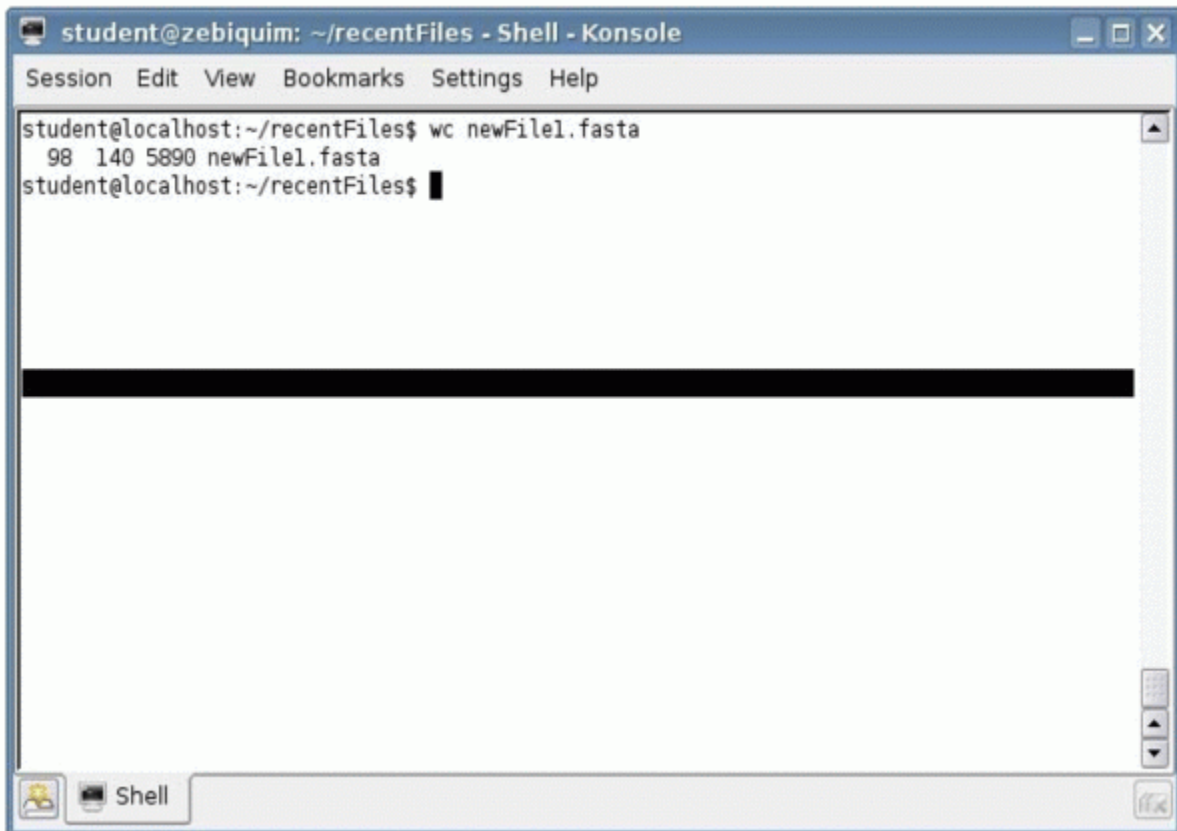


**Comments:** The Linux shell offers a command `wc` (from Word Count) to count the size of the files. It displays not only the number of bytes (characters) that the file contains, but also the number of words and the number of lines.

**Issuing the commands:** To count the number of characters, words and lines of the file "newFile1.fasta", we just need to type:

```
wc newFile1.fasta
```

As a result, three numbers will appear in the screen, corresponding, respectively, to the number of lines, words, and characters in the file. As we can see from the output, the newFile1.fasta file has 5890 characters, 140 words, and 98 lines:

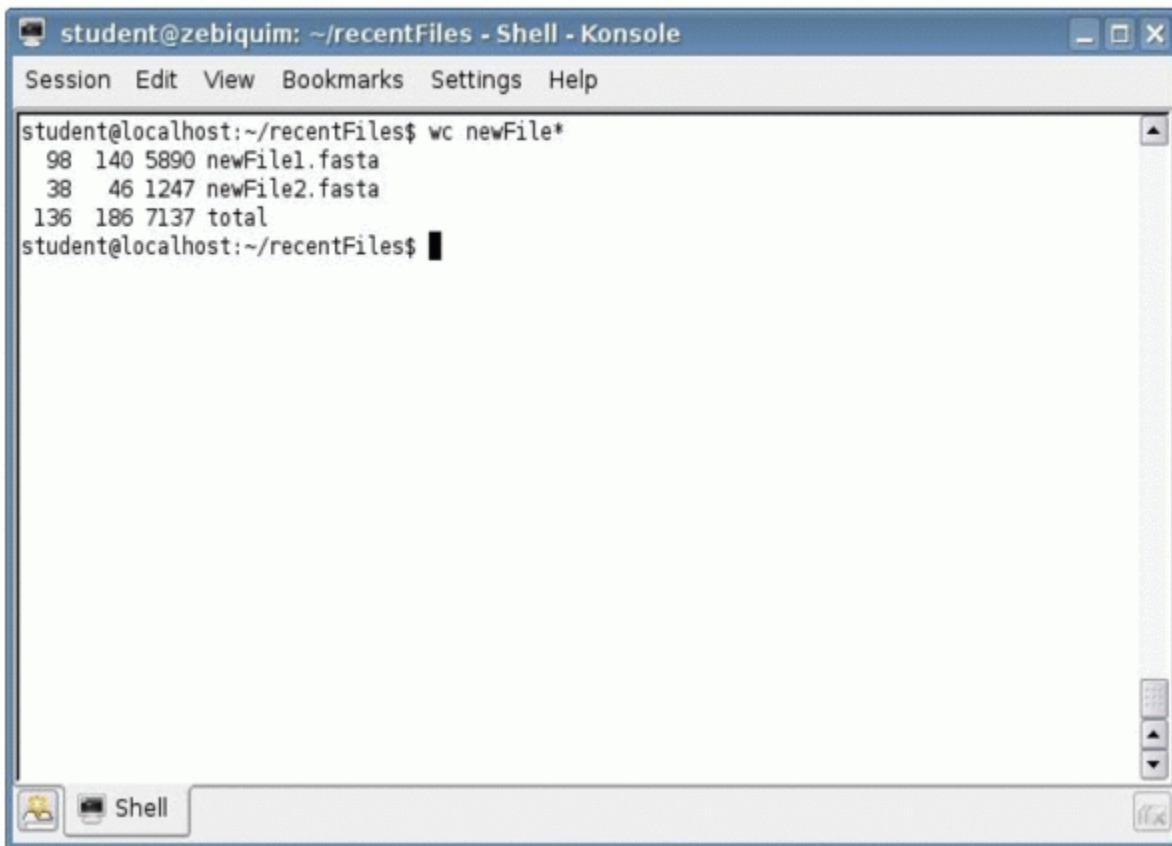
A screenshot of a Linux terminal window titled "student@zebiquim: ~/recentFiles - Shell - Konsole". The terminal shows the command "wc newFile1.fasta" being executed. The output is "98 140 5890 newFile1.fasta". The terminal prompt is "student@localhost:~/recentFiles\$".

```
student@zebiquim: ~/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ wc newFile1.fasta
98 140 5890 newFile1.fasta
student@localhost:~/recentFiles$
```

We also use `wc` for a batch of files by typing all the file names or by using wildcards:

```
wc newFile*
```

As you can see in the output below, in this case the program not only shows the counts of each file, but also displays the sum of all respective values.



```
student@zebiquim: ~/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ wc newFile*
 98 140 5890 newFile1.fasta
 38  46 1247 newFile2.fasta
136 186 7137 total
student@localhost:~/recentFiles$
```

## [Using grep to select lines in a file]

**Task:** Look at all the FASTA headers of the FASTA files.

**Comments:** It is very common to have a file or a set of files where you want only to look at specific lines. To do this in MS Windows<sup>®</sup>, users generally open the file in a text editor and then use a "find" command to look for some specific word(s) present in the requested lines. This can be tiresome and will not work properly for very large files, since most text editors in Windows<sup>®</sup> cannot handle very large files. The Linux's grep command, on the other hand, can handle files of any size. In addition, the user can select lines he/she is interested on by specifying some of their content. The challenge is to find out the distinct pattern we are searching for. In the case of the FASTA header lines, this is an easy job, since we only have to count all lines containing the ">" character.

**Issuing the commands:** To see all FASTA headers of the file newFile1.fasta, we just need to type:

```
grep ">" newFile1.fasta
```

We can also look at the FASTA headers of all FASTA files of our example using wildcards:

```
grep ">" newFile*.fasta
```

The results in the second case will be:

```

student@localhost:~/recentFiles$ grep ">" newFile*
newFile1.fasta:>gi|56797977|emb|AJ871406.1| Plasmodium falciparum mRNA for Pdx2 protein
newFile1.fasta:>gi|4507020|ref|NM_000342.1| Modified Homo sapiens solute carrier family 4,
anion exchanger, member 1 (erythrocyte membrane protein band 3, Diego blood group) (SLC4A
1), mRNA, introduced by Gubialan
newFile1.fasta:>gi|84697036|gb|AF527538.2|Modified Plasmodium falciparum 3D7 oxoacyl-ACP r
eductase mRNA, complete cds; nuclear gene for apicoplast product; Introduced by Gubialan
newFile1.fasta:>seq6
newFile1.fasta:>sequence5
newFile1.fasta:>sequence 3
newFile2.fasta:>seq1; sequenced by Gubialan
newFile2.fasta:>seq2
newFile2.fasta:>sequence 3
newFile2.fasta:>seq4; sequenced by Gubialan
newFile2.fasta:>sequence5
newFile2.fasta:>seq6
student@localhost:~/recentFiles$

```

By visual inspection, we can see that there are a total of 12 FASTA headers, and therefore, 9 sequences in our files.

### [Using grep and wc together]

**Task:** Check how many sequences are contained in each FASTA file.

**Comments:** This task was actually performed manually in the previous item. However, in this case, it was easy to manually count the number of sequences because this number was very small. However, if files are big and have a large number of sequences, manual counting becomes tiresome and error prone. To automatically count the lines output by the grep command, we can "couple" it to the wc command.

In the Linux shell, we can directly connect the output of a program into the input of another program. We do this by using *pipes*. Pipes are like intermediate files, but much faster and simpler to specify. A pipe in the shell is specified using the "|" character. If we want the output of program p1 to be used directly as the input of program p2, we need to "connect" them using a pipe: p1 | p2.

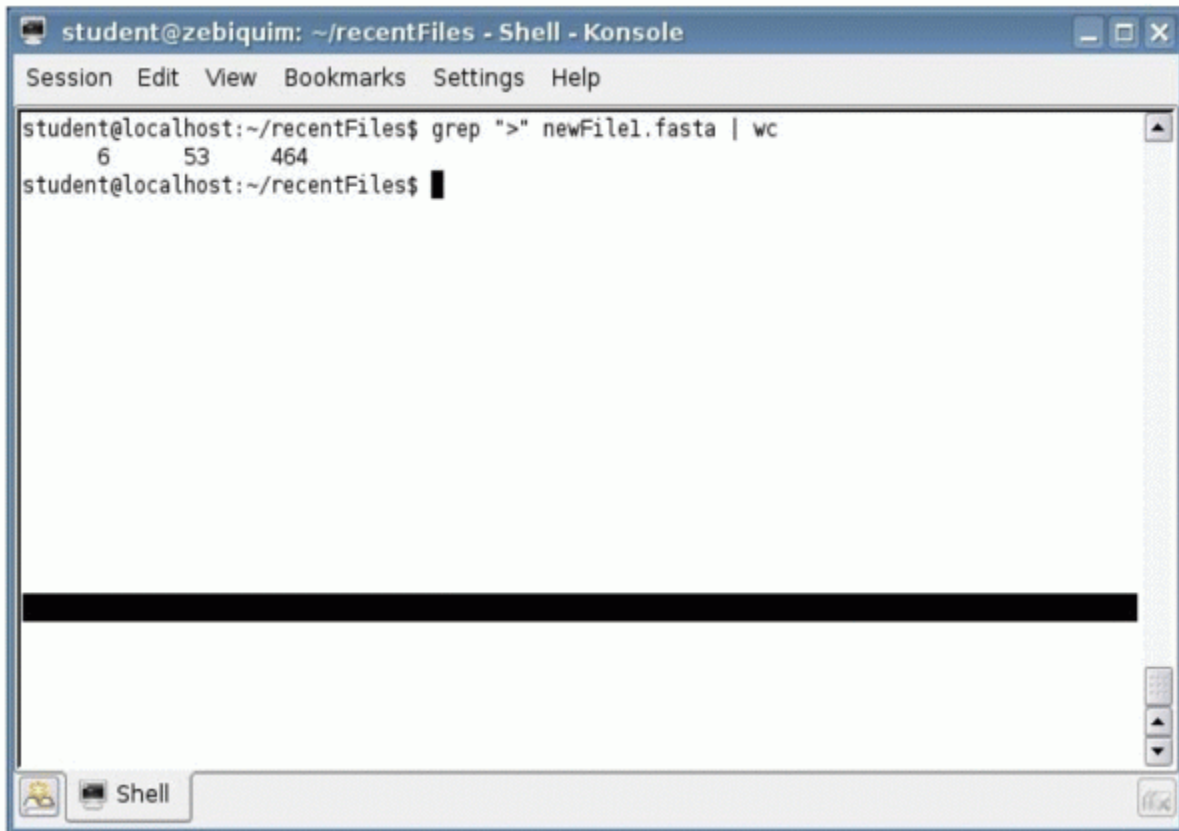
**Issuing the commands:** We want to count how many lines are being displayed in the output of the command:

```
grep ">" newFile1.fasta
```

So we only need to "pipe it" to the command wc:

```
grep ">" newFile1.fasta | wc
```

The output will then show us that newFile1.fasta contains 6 sequences :



The image shows a terminal window titled "student@zebiquim: ~/recentFiles - Shell - Konsole". The window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The terminal content shows the following command and output:

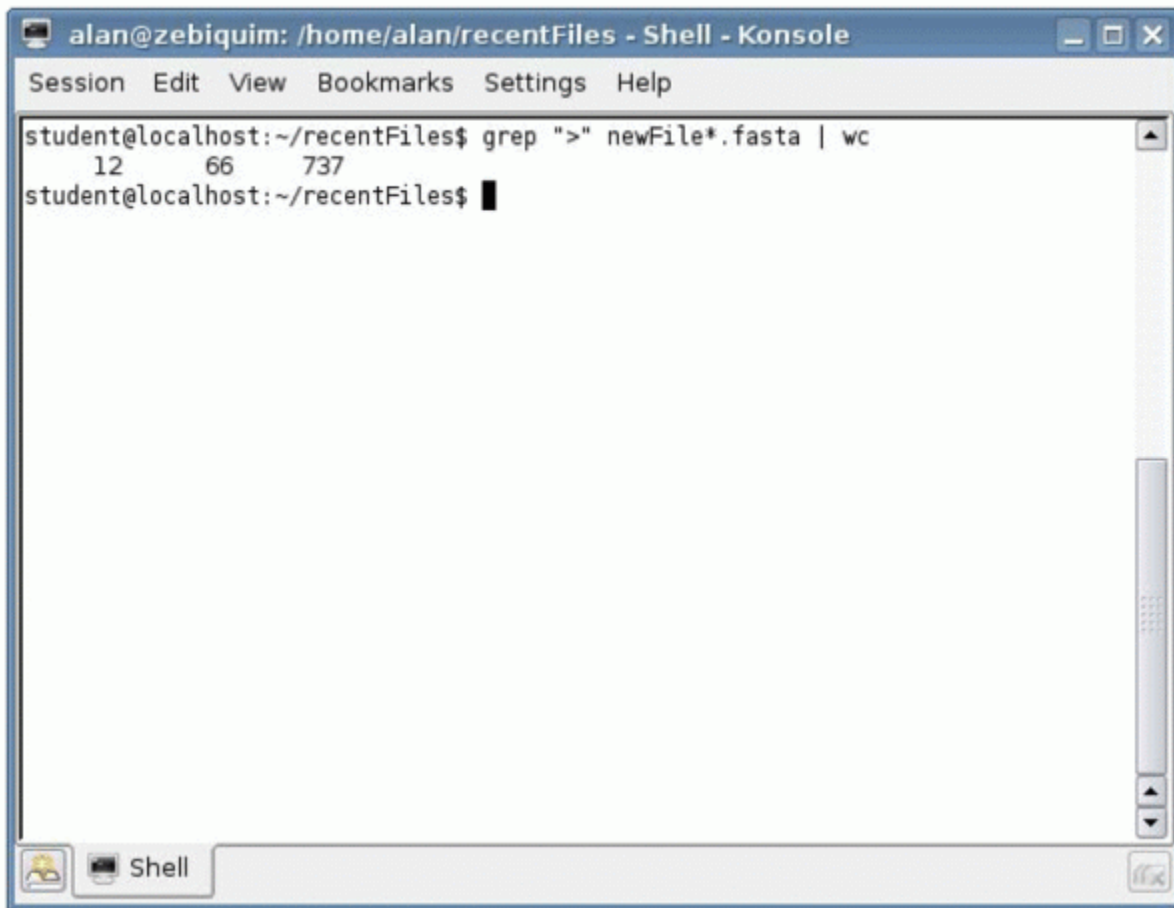
```
student@localhost:~/recentFiles$ grep ">" newFile1.fasta | wc
    6    53   464
student@localhost:~/recentFiles$
```

The terminal window also features a status bar at the bottom with a "Shell" icon and a "Shell" label.

We can also do this task for all FASTA files by using wildcards:

```
grep ">" newFile*.fasta | wc
```

The output will show us how many FASTA headers are present in both our files:

A screenshot of a Linux terminal window titled "alan@zebiquim: /home/alan/recentFiles - Shell - Konsole". The terminal shows a command being executed: "student@localhost:~/recentFiles\$ grep ">" newFile\*.fasta | wc". The output of the command is displayed on the next line: "12 66 737". The terminal prompt "student@localhost:~/recentFiles\$" is visible again on the line below. The terminal window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The bottom of the window shows a taskbar with a "Shell" icon and a close button.

### [Using grep and wc together]

**Task:** Check how many FASTA sequences contain the term "Gubialan" in the.

**Comments:** We can use more than one pipe in a single command line. This means that we can connect two, three, four, or any number of programs in a single processing command. This is particularly useful when we have filter programs like grep, so the user can perform many successive filtering steps, all in a single command line.

**Issuing the commands:** In this case, we want to perform two filterings and one counting:

First we need to select the FASTA header lines from the files:

```
grep ">" newFile*.fasta
```

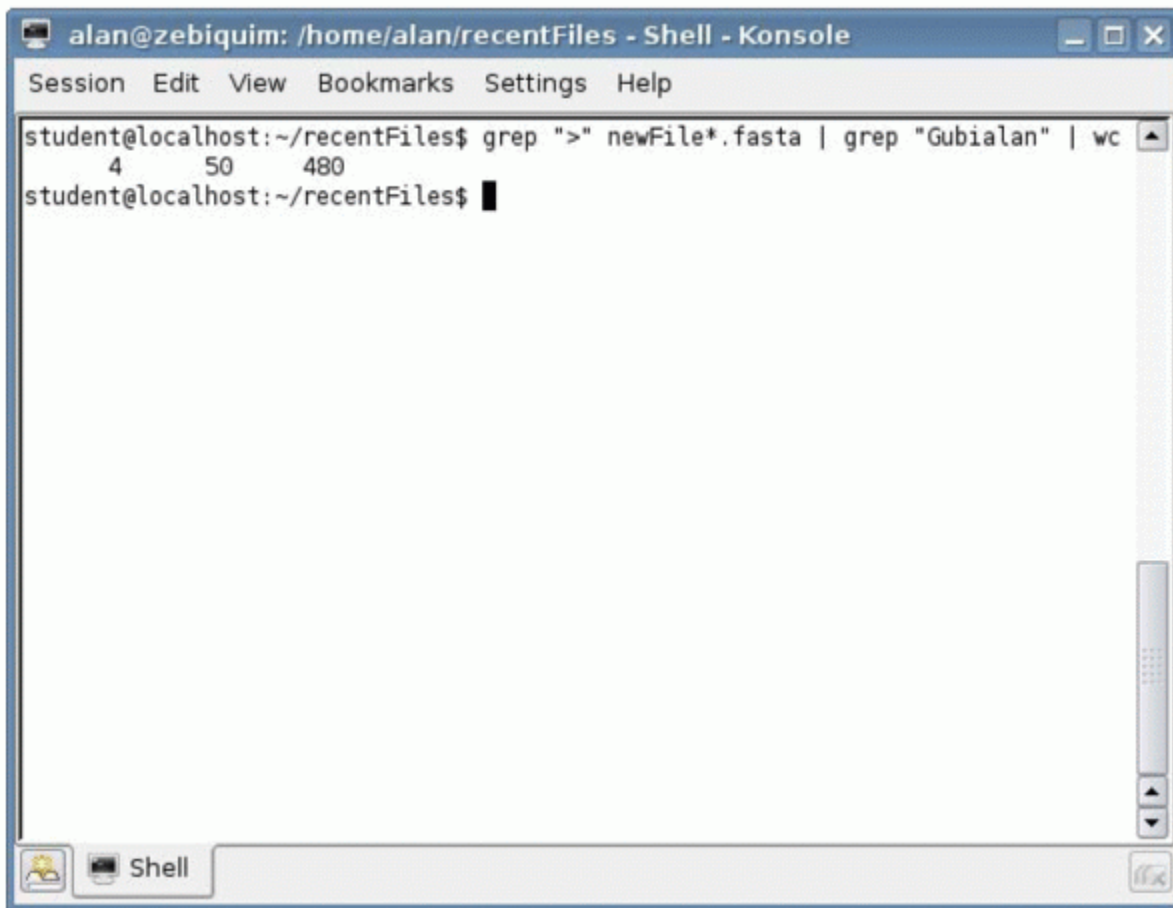
Next, we select only those lines that contain the word "Gubialan":

```
grep ">" newFile*.fasta | grep "Gubialan"
```

Finally we want to count these lines

```
grep ">" newFile*.fasta | grep "Gubialan" | wc
```

The result will show the number of FASTA sequences that have the word Gubialan in the header:

A screenshot of a terminal window titled "alan@zebiquim: /home/alan/recentFiles - Shell - Konsole". The window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The terminal content shows a command: "student@localhost:~/recentFiles\$ grep ">" newFile\*.fasta | grep "Gubialan" | wc". The output is displayed as a table with three columns: "4", "50", and "480". The prompt "student@localhost:~/recentFiles\$" is shown again on the next line.

```
alan@zebiquim: /home/alan/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ grep ">" newFile*.fasta | grep "Gubialan" | wc
  4    50   480
student@localhost:~/recentFiles$
```

### [Concatenating files, saving a program's output in a file]

**Task:** Create a new file named *all.fasta*, with the contents of all three FASTA files.

**Comments:** One common way to join the contents of different files is to use a text editor and "cut and paste" tools, in a process that is tiresome, error prone, and cannot be applied to a large number of files or to files with large sizes. In Linux, joining files can be performed in an extremely fast way by using the `cat` command. This command displays on the screen the contents of a file. It is similar to `more`, already seen above, but the contents are displayed at once, with no pause after each page. The command `cat` can also be used to display the contents of multiple files, through the use of wildcards. However, to fulfill the task of creating a new file, we still need another element, an "output redirection". In the shell, we can save the screen output of any command in a file by *redirecting* this output. To redirect the output of a command to a file, we need to add, at the end of the command, the ">" character, followed by the name of the file.

**Issuing the commands:** We will use `cat` and wildcards to make the system output the contents of all the files at once:

```
cat newFile*
```

However, if you try the command above you will notice that the contents of all the files are really put together, but they are all displayed in the screen. To create the file "all.fasta" we need now to *redirect* this output to the file:

```
cat newFile* > all.fasta
```

You can now inspect your newly created file and verify that it contains indeed the sequences of all previous files. For this task, you can either use more:

```
more all.fasta
```

Or, for example, check only the FASTA headers to see if all the ones from the three files are in all.fasta:

```
grep ">" all.fasta
```

### [Searching files by name]

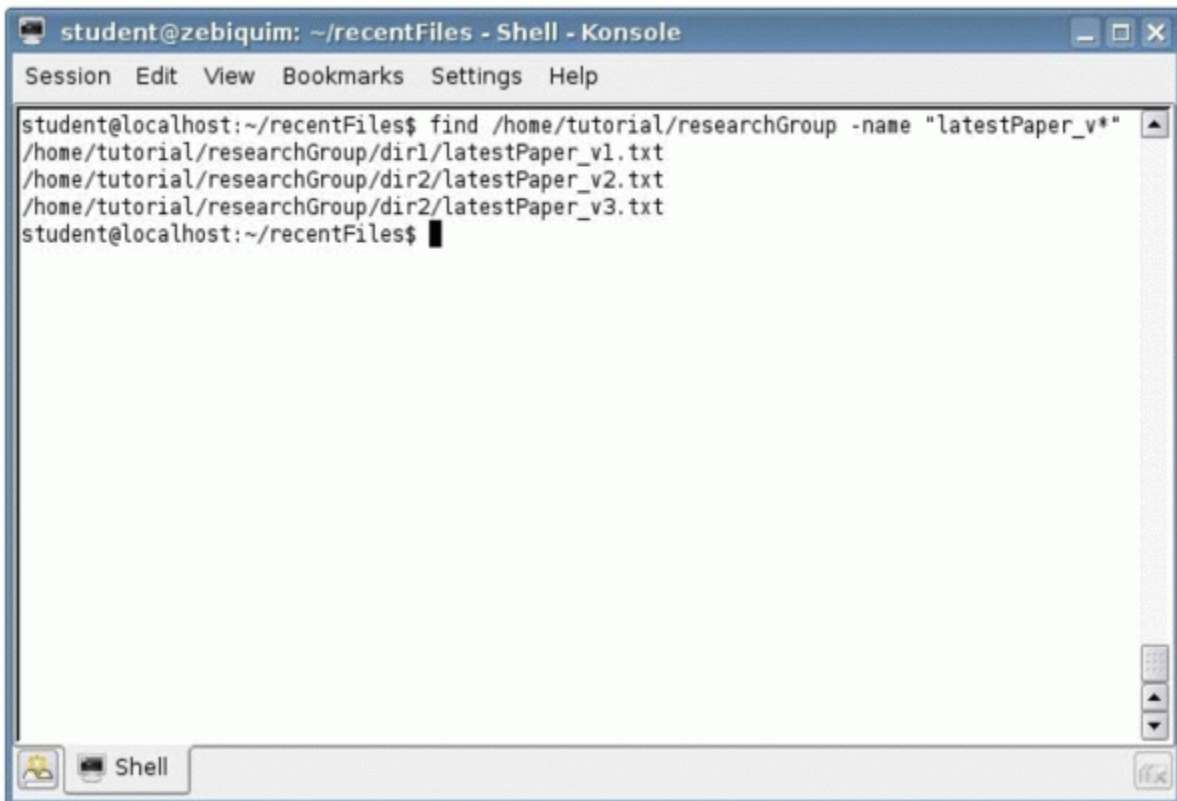
**Task:** Find all the files with names starting with "latestPaper\_v" in the directory /home/student/tutorial/researchGroup and its subdirectories.

**Comments:** To perform this task the UNIX shell has the program find.

**Issuing the commands:** We want to find all files starting with "latestPaper\_v", so will accept anything after this initial part of the name. To specify that, we just need to put the wildcard at the end of the command. We also want to start the search at the directory "/home/student/tutorial/researchGroup", therefore you should type the following command:

```
find /home/tutorial/researchGroup -name "latestPaper_v*"
```

As a result , you will see the file names listed with their complete path:

A screenshot of a terminal window titled "student@zebiquim: ~/recentFiles - Shell - Konsole". The window has a menu bar with "Session", "Edit", "View", "Bookmarks", "Settings", and "Help". The terminal content shows the command `find /home/tutorial/researchGroup -name "latestPaper_v*"` being executed, resulting in three lines of output: `/home/tutorial/researchGroup/dir1/latestPaper_v1.txt`, `/home/tutorial/researchGroup/dir2/latestPaper_v2.txt`, and `/home/tutorial/researchGroup/dir2/latestPaper_v3.txt`. The prompt `student@localhost:~/recentFiles$` is visible at the end of the output. The terminal window has a standard Linux desktop environment interface with window controls and a taskbar at the bottom.

## [Copying files]

**Task:** Create the directory "papers" in your home directory. Copy the files you found in the previous item into the new "papers" directory. There should be 3 files, one in each of the subdirectories of /home/student/tutorial/researchGroup

**Comments:** When copying files, we can also use wildcards. It is important to remember that, if we use wildcards, we are probably specifying more than one file to be copied, so the destination has to be a directory.

**Issuing the commands:** First, go to the home directory by using the command `cd` with no argument, and then create the new directory using the command `mkdir`:

```
cd
mkdir papers
```

The three files the we have found in the previous item are:

```
"/home/student/tutorial/researchGroup/dir1/latestPaper_v1.txt"
```

```
"/home/student/tutorial/researchGroup/dir1/latestPaper_v2.txt"
```

and

```
"/home/student/tutorial/researchGroup/dir2/latestPaper_v3.txt".
```

There are many ways in which we can perform the copy, depending if we use *relative paths* or *absolute paths* and on our use of wildcards. We will show three.

In the first one we use only relative paths and no wildcard (you should type all text in a single command line, we use more than one line here due to lack of space):

```
cp ../tutorial/researchGroup/dir1/latestPaper_v1.txt ../tutorial/researchGroup/dir2/latestPaper_v2.txt ../
tutorial/researchGroup/dir2/latestPaper_v3.txt papers
```

Now we use absolute paths for the source files (again, a single command line):

```
cp /home/tutorial/researchGroup/dir1/latestPaper_v1.txt /home/tutorial/researchGroup/dir2/
latestPaper_v2.txt /home/tutorial/researchGroup/dir2/latestPaper_v3.txt papers
```

Finally, we can try a shorter version, with wildcards:

```
cp /home/tutorial/researchGroup/dir*/latestPaper_v* papers
```

Please note that, in this last case, wildcards were used twice, the first time to describe the directory names and the second time to specify the files.

## [Comparing the contents of files]

**Task:** Check the differences between the files *latestPaper\_v1.txt* and *latestPaper\_v2.txt*.

**Comments:** Files are constantly changing. Sometimes, to avoid losing important information due to bad editing, users maintain many versions of a file instead of always modifying the same file. This way, we avoid the



damage of editing mistakes. In another common possibility (as it is the case of this chapter), two people are working in different parts of the same file. Finally, we have the case where we just need to know the differences between two arbitrary files. We can do this in Linux using the `diff` command. We type the command name, the names of the two files, and a description of the differences is shown on the terminal window.

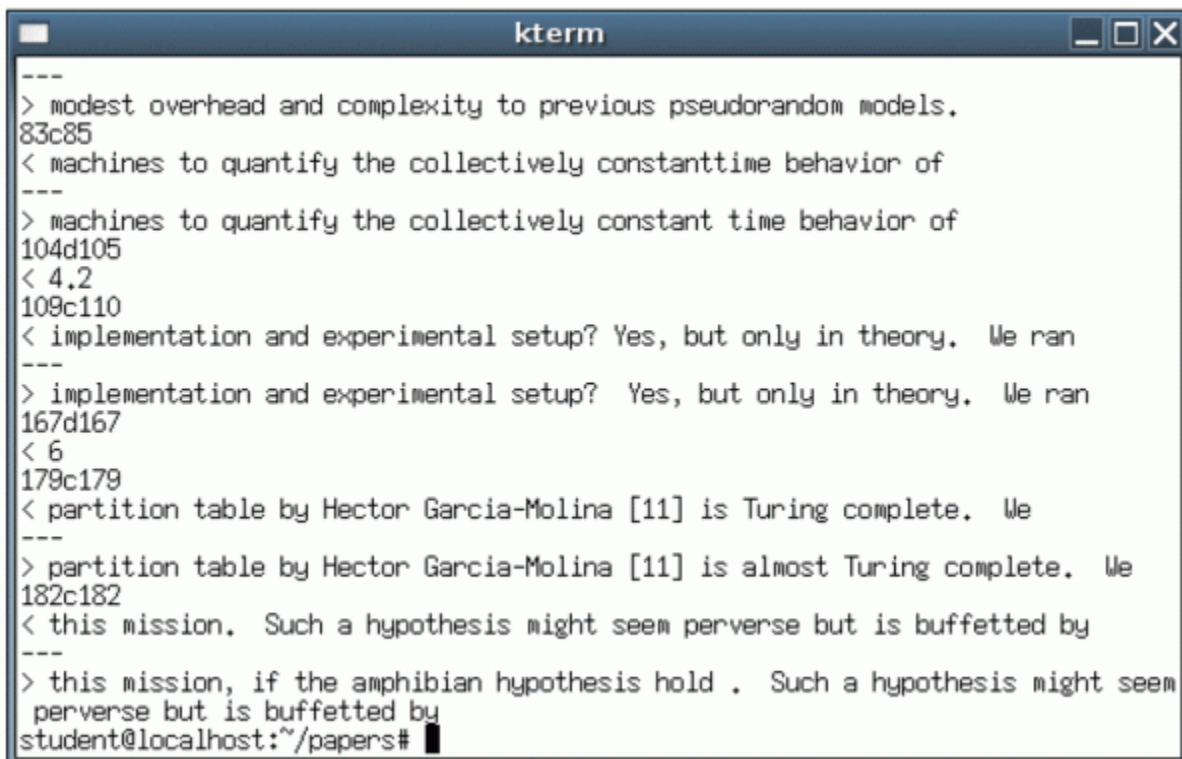
**Issuing the commands:** To find out the differences between the files, let's move first into directory `"/home/student/tutorial/researchGroup/papers"`:

```
cd papers
```

Now, to find out the differences between files `"latestPaper_v1.txt"` and `"latestPaper_v2.txt"`, we will use the `diff` command:

```
diff latestPaper_v1.txt latestPaper_v2.txt
```

You will see an output like the one displayed below:



```
-----
> modest overhead and complexity to previous pseudorandom models.
83c85
< machines to quantify the collectively constanttime behavior of
-----
> machines to quantify the collectively constant time behavior of
104d105
< 4.2
109c110
< implementation and experimental setup? Yes, but only in theory. We ran
-----
> implementation and experimental setup? Yes, but only in theory. We ran
167d167
< 6
179c179
< partition table by Hector Garcia-Molina [11] is Turing complete. We
-----
> partition table by Hector Garcia-Molina [11] is almost Turing complete. We
182c182
< this mission. Such a hypothesis might seem perverse but is buffeted by
-----
> this mission, if the amphibian hypothesis hold . Such a hypothesis might seem
perverse but is buffeted by
student@localhost:~/papers#
```

The output of the program will only show the difference between the files. First, the line numbers involved are described. In our output, the first difference starts with `"83c85"` which means, line eighty-three of the first file, line eighty-five of the second file. Next, the contents of the lines of the first file preceded by `"<"` and the contents of the lines of the second file preceded by `">"`. The next differences are similarly described.

## [Returning to the home directory]

**Task:** Go back to the sequence directory.

**Comments:** None.

**Issuing the commands:** To go back to your home directory you just need to type the `cd` command with no arguments.

```
cd
```

### [Checking for new sequences in FASTA files]

**Task:** Find out the names of the sequences of `newFile2.fasta` that are not present in `newFile1.fasta`. All the sequences will have a similar name, with a common prefix and a number. Verify the names of the sequences and check if there is a sequence missing (`grep`, `sort`, `more`)

**Comments:** We need to compare the names of sequences. So far we know how to select the sequence names using `grep`, how to store the results of a program in a file using `>` and how to compare the contents of two files using `diff`. If we want to know the sequences that are in `newFile1.fasta` and not in `newFile2.fasta`, we will first use `grep` to isolate the sequence names in two files "`newFile1.names`" and "`newFile2.names`" (these are example names, any name can be used). Then, we can use the command `diff` in the new files:

**Comments:** To make things easier, we will first go to the directory `recentFiles`:

```
cd recentFiles
```

Now we will create the new files containing the names of the sequences contained in `newFile1.fasta` and `newFile2.fasta`:

```
grep ">" newFile1.fasta > newFile1.names
```

```
grep ">" newFile2.fasta > newFile2.names
```

Finally, we will check the differences of the name files:

```
diff newFile1.names newFile2.names
```

### [Checking for new sequences in FASTA files with unordered sequences]

**Task:** If you examine closely the output of the `diff` command of the last item you will notice a problem: some sequences that are in both files are reported as differences (`seq6`, and `sequence5`). Try to correct the problem.

**Comments:** The problem is that `diff` reads both files line by line and check the differences also line by line. If the sequences are not in the same order in both files the results will be misleading. Therefore, before comparing the sequence names, we need to sort both files to be sure the names appear in the same order. The UNIX shell provides a sorting command `sort` that will read a file and print the lines of this file in alphabetical order. If, after selecting the sequence names, we sort them before saving in the names files, then the procedure should work appropriately.

**Issuing the commands:** We will perform almost the same task as before, but we can now "pipe" the result of the `grep` program into `sort` before storing them. We will store the results under new file names for clarity.

```
grep ">" newFile1.fasta | sort > newFile1.names.sorted
```

```
grep ">" newFile2.fasta | sort > newFile2.names.sorted
```

```
diff newFile1.names.sorted newFile2.names.sorted
```

Check the results and you will see that now there are no common sequences displayed in the `diff` output.

## [Running programs, output redirection]

**Task:** Run the program `quickprocess` for the file `all.fasta`. Now run it again and save the output and the error messages in different files, redirect the error output to `quickprocess.error` and the rest of the output to `all.quickprocess.fasta`.

**Comments:** Running a specific program in the command line is like running a shell command (all the shell commands we have seen are actually Linux programs). However, when programs generate a lot of screen output in Linux, we can save this output in files. To understand this, we need first to explain the concept of *standard output* and *error output*. In Linux, programs have three types of output: file output, standard output and error output. Normally, whatever is sent to standard output and to error output is shown on the shell window. However, the user can **redirect** either or both outputs to files. To redirect a program's standard output to a file we use the ">" character, and to redirect the error output to a file, we use the "2>" characters (no space between them).

**Issuing the commands:** We want to run program `quickprocess` on the file `all.fasta`. This program was downloaded when you performed the `wget` command and is in directory "programs". Initially we will run the program "quickprocess" normally:

```
~tutorial/programs/quickprocess all.fasta
```

As you notice, the output will quickly run out of your shell window. Now we will try to run `quickprocess` again and save the standard output in `quickprocess.out`, and the error output in `quickprocess.error`:

```
~/tutorial/programs/quickprocess all.fasta > quickprocess.out 2> quickprocess.error
```

You can now use `more` or `less` to inspect each output<sup>4</sup>.

## [Running a program, and killing it]

**Task:** You should run program `slowprocess` for the file `all.fasta`. This is a slow processing program and it will not end soon. Next, you should kill the program.

**Comments:** In Linux you can kill any program that is being run from the shell. This is an important feature when some programs take much more time than previously anticipated or behave erroneously. To stop a program you only have to type `control-c` (that is, press, at the same time, the "ctrl" and the "c" keys).

**Issuing the commands:** To start the program `slowprocess` for the file `all.fasta` just type:

```
~tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error
```

You will notice that the shell will "freeze", meaning that the program is running and that the shell is waiting for an output or for the program to finish. To finish it just type `ctrl-c`.

## [Running processes in the background]

**Task:** Run program `slowprocess`, stop it momentarily, make it continue in the background.

**Comments:** Linux actually runs many programs at the same time. You can see that, for example, when you have a clock program in your taskbar, it is running at the same time that your are using the shell. In the previous

---

<sup>4</sup> Remember that you should use the space bar to move down a page in `more` and in `less` and press the "q" key to quit

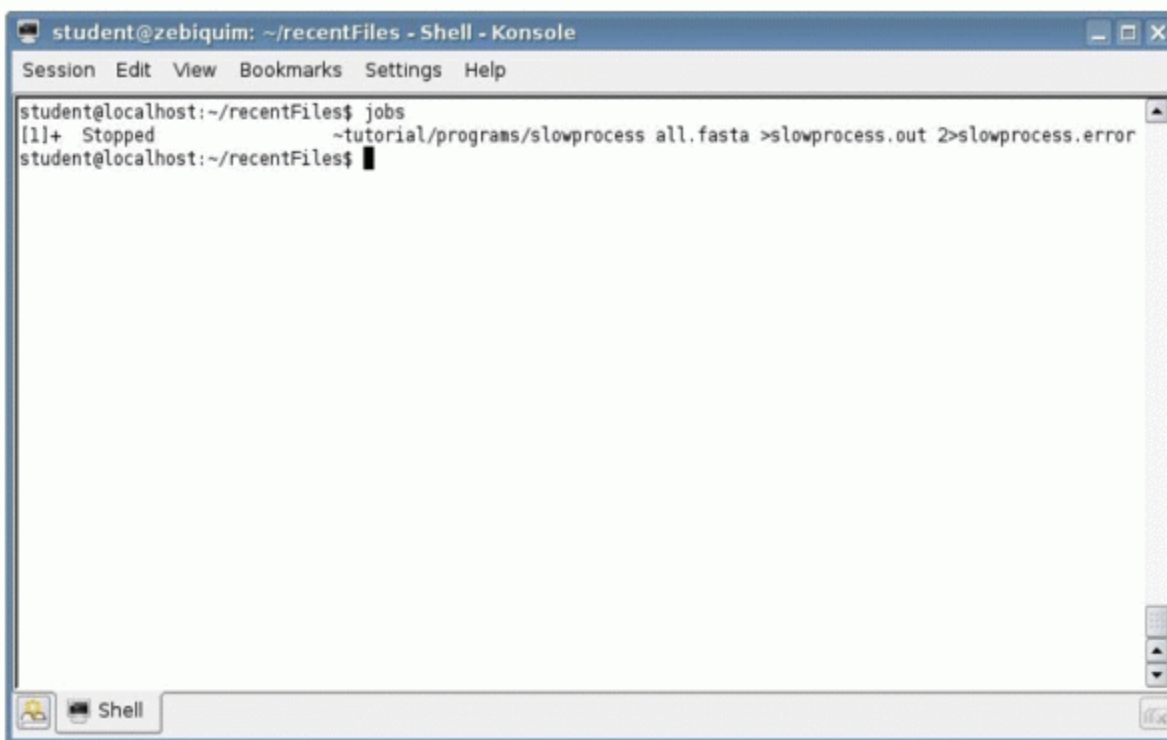
exercise you have terminated the execution of a program in the shell by typing `ctrl-c`. Alternatively, you can stop the program temporarily, and later resume its execution, exactly at the point where it was stopped. Once a program is stopped in the shell, you can restart it normally, or make it run in the *background*. Programs running in the background behave normally, but you can keep using the shell and typing new commands as the original program runs. This means you can have many programs running in the shell's background. The shell also offers the command "jobs", to check which programs are currently running or sleeping in the shell.

**Issuing the commands:** Start the program `slowprocess` with `all.fasta` as input 5:

```
~/tutorial/programs/slowprocess all.fasta > slowprocess.out 2> slowprocess.error
```

Now, in order to suspend the running program, you should type `ctrl-z` (that is, press, at the same time, the "ctrl" and the "z" keys). To suspend a program is to put it to "sleep". You will notice that, after typing `ctrl-z` you can again type shell commands. To check if the program is sleeping try typing `jobs`

Your output should look like:



```
student@zebiquim: ~/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ jobs
[1]+  Stopped                  ~/tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error
student@localhost:~/recentFiles$
```

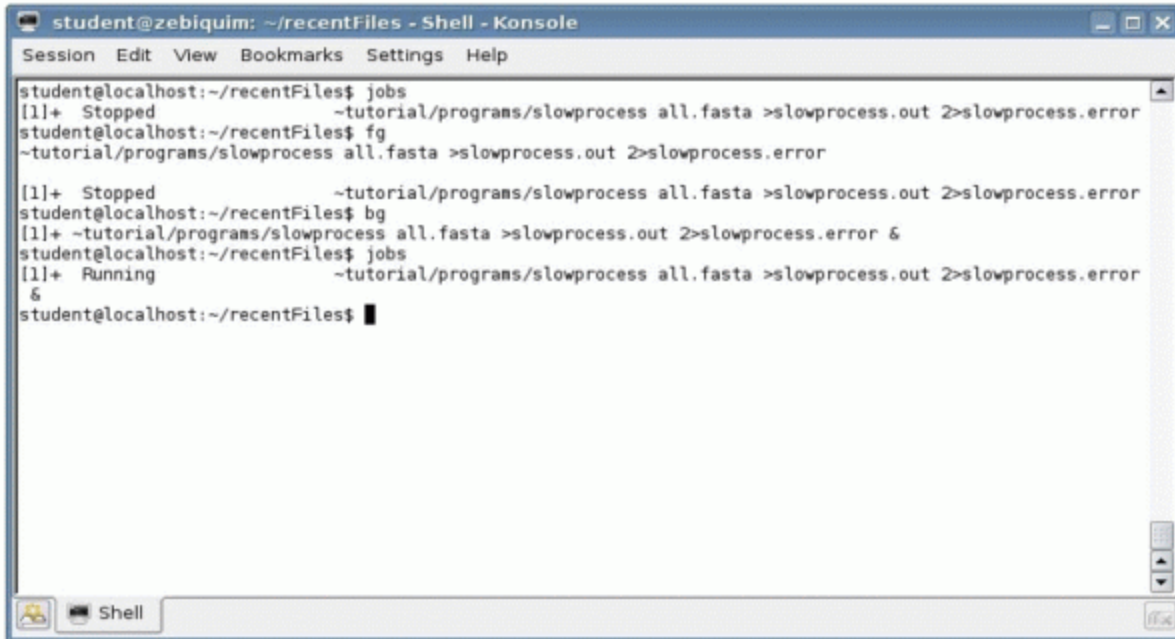
Indicating that the program `slowprocess` is currently stopped, but has not been killed. To resume the program you should type:

```
fg
```

Which will put the program in "foreground", freezing the shell again. As an alternative, you can type `ctrl-z` again and then the command

```
bg
```

This command will send the program to run in the "background". If you use the jobs command again, you will see that the program slowprocess is now running:



```

student@zebiquim: ~/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ jobs
[1]+  Stopped                  -tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error
student@localhost:~/recentFiles$ fg
-tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error

[1]+  Stopped                  -tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error
student@localhost:~/recentFiles$ bg
[1]+ -tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error &
student@localhost:~/recentFiles$ jobs
[1]+  Running                  -tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error
&
student@localhost:~/recentFiles$ █

```

### [Kill process, start process slowprocess directly in the background, error output in file]

**Task:** Kill the running process slowprocess, restart the process directly in the background, storing the output in file *slowprocess.error*.

**Comments:** We have seen how to put a running process in the background. However, we can start a process directly in the background, saving some typing. If we want a process to run directly in the background, we should type "&" at the end of the command line (just before typing "enter"). The shell will resume immediately and the process will run at the same time, in the background.

**Issuing the commands:** To kill a process running in the background we use the *kill* command, followed by the sign "%" before the job number, in our case:

```
kill %1
```

Now we want the error output of our program to go into file "slowprocess.error". To do this, we will use the error output redirection ("2:"). We also want to start our process directly in the background, so we need to type "&" at the end of the command line:

```
~tutorial/programs/slowprocess >slowprocess.out 2> slowprocess.error &
```

You can check now if the process is actually running with the "jobs" command. The result should be something like:



```
kterm
student@localhost:~/recentFiles$ ~/tutorial/programs/slowprocess all.fasta > slowprocess.out 2> slowprocess.error &
[1] 16067
student@localhost:~/recentFiles$ . jobs
[1]+  Running                  ~/tutorial/programs/slowprocess all.fasta >slowprocess.out 2>slowprocess.error &
student@localhost:~/recentFiles$
```

### [Check CPU usage]

**Task:** Check how much of the computer's capacity is being used by the program `slowprocess`.

**Comments:** We have seen the command `jobs` to list all the processes being run from the shell. However, this program only shows **what** is running, not how much of the computer's capacity is being allocated to it. Linux offers a program that will show how much of the computer's memory and CPU time is being used by the programs at a given moment. It is important to note, however, that this program will check ALL programs being run in the computer, including the programs you are running and, potentially, those that are run by other users too.

**Issuing the commands:** The program `slowprocess` should still be running. To check how much processing power it is using, you should type

```
top
```

Your shell window should show something like follows below:

```

alan@zebiquim: /home/alan - Shell - Konsole <2>
Session Edit View Bookmarks Settings Help
top - 16:27:23 up 5:19, 1 user, load average: 0.52, 0.38, 0.41
Tasks: 114 total, 2 running, 111 sleeping, 0 stopped, 1 zombie
Cpu(s): 4.0% us, 3.7% sy, 91.4% ni, 0.0% id, 0.0% wa, 0.0% hi, 1.0% si
Mem: 2076212k total, 1970164k used, 106048k free, 30296k buffers
Swap: 4883752k total, 8864k used, 4874888k free, 1521164k cached

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM    TIME+  COMMAND
 7647 student   35   10  5144 1896 1260 R  91.9  0.1   0:36.83 slowprocess
 4524 root       15    0  101m  64m 5240 S   5.7  3.2   5:39.42 Xorg
 6706 alan      15    0 33776  18m  13m S   1.0  0.9   0:25.06 kicker
 7609 alan      16    0  6528 4408 1484 S   1.0  0.2   0:00.53 xv
 7036 alan      15    0  207m 108m  41m S   0.3  5.4   4:29.17 soffice.bin
    1 root       16    0  1564   528  460 S   0.0  0.0   0:00.99 init
    2 root       34   19    0    0    0 S   0.0  0.0   0:00.01 ksoftirqd/0
    3 root       RT    0    0    0    0 S   0.0  0.0   0:00.00 watchdog/0
    4 root       10   -5    0    0    0 S   0.0  0.0   0:00.15 events/0
    5 root       10   -5    0    0    0 S   0.0  0.0   0:00.00 khelper
    6 root       10   -5    0    0    0 S   0.0  0.0   0:00.00 kthread
    8 root       10   -5    0    0    0 S   0.0  0.0   0:00.30 kblockd/0
    9 root       20   -5    0    0    0 S   0.0  0.0   0:00.00 kacpid
  125 root       15    0    0    0    0 S   0.0  0.0   0:00.21 pdflush
  127 root       18   -5    0    0    0 S   0.0  0.0   0:00.00 aio/0
  126 root       15    0    0    0    0 S   0.0  0.0   0:00.66 kswapd0
   715 root       10   -5    0    0    0 S   0.0  0.0   0:00.00 kseriod

```

You will see 12 columns. The first one will show the process id number. This number is designated and used by the system. The second column is "USER", which identifies the user that started the program. The 9th column, "%CPU", shows how much of the computer's processor is being used by each program. The 10th column, "%MEM" shows how much of the computer's memory is being used by each program. The 11th column, "TIME+", shows how much time has elapsed since the program started. Finally, the last column displays the shell command associated with each program. For example, in our figure we can see that

- slowprocess id number is 16067
- it is using 96.9 percent of the computers CPU (this is a lot of processing power)
- it is using only 0.1 percent of the memory
- at the time this screenshot was taken, the program had been running for more more than 47 seconds

It is important to note that, with the exception of the COMMAND column, what you see in your terminal will be different numbers, since you will be using a different computer. Notice also that the columns "%CPU", "%MEM" and "TIME+" can change continuously, reflecting dynamically the undergoing processes on your computer.

### [Checking the final lines of a file]

**Task:** Check interactively final lines of the file *slowandbad.error*, when you see "something wrong" printed, kill the program.

**Comments:** In the field of Bioinformatics, it is not uncommon to have programs that run for a long time. This can be due to at least three different reasons: there is some other program using all the computer's resources, your own program may be slow and consuming computer's resources (thus running for a long time), or something may be wrong with your program. When this last case happens, generally some message will appear at the end of the program's output. An easy way to check the end of an output is to send it to a file and inspect the last lines of that file. This way, you can at the same time keep all output saved in a file for later detailed

inspection, and avoid browsing the whole file to visualize the last lines. To inspect the last lines of a file we use the command `tail`.

**Issuing the commands:** The program `slowAndBadProcess` was designed to misbehave. However, it will take some time for this to happen. When it happens, the program will print "something wrong" in the error output. You should first start the `slowandbadprocess` program, sending its error output to `slowandbad.error` and the standard output the `slowandbad.out`:

```
~tutorial/programs/slowandbadprocess all.fasta > slowandbad.out 2> slowandbad.error&
```

To check the last lines of that file, you should type the command:

```
tail slowandbad.error
```

Keep doing once every minute until you see that "something wrong" is printed in the last line.

### [Killing a program running in the background]

**Task:** Kill the program `slowandbadprocess`

**Comments:** When a program starts use `top` and `kill` to finish its execution, especially if it consumes a lot of system's resources, which is the case of `slowandbadprocess`. Once you know you need to kill a program, you should first check its shell process id number. You can do this by using the command `jobs`. Once you know the number, you can use the `kill` command to terminate that program.

**Issuing the commands:** First check which is the process number by using the `jobs` command:

```
jobs
```

Your output can be slightly different depending on what you have done in your shell so far. The important thing is to register which is the number of the job `slowAndBadProcess`, in our case it is 1. Now, you can kill the program by using the `kill` command, and then check if it was killed using the `jobs` command:

```
kill %1
```

```
jobs
```

When we run these programs we have the following output:



```

student@zebiquim: ~/recentFiles - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~/recentFiles$ ~tutorial/programs/slowandbadprocess all.fasta > slowandbad.out 2> slowandbad.error&
[1] 7640
student@localhost:~/recentFiles$ tail slowandbad.error
Warning!: Name 'gi|56797977|emb|AJ871406.1| Plasmodium falciparum mRNA for Pdx2 protein'
does not follow the SLP (Silly Useless Pattern)!!!
something wrong: No match for size 12 ==>running forever
student@localhost:~/recentFiles$ jobs
[1]+  Running                  ~tutorial/programs/slowandbadprocess all.fasta >slowandbad.out 2>slowandbad.e
rror &
student@localhost:~/recentFiles$ kill %1
student@localhost:~/recentFiles$ jobs
[1]+  Terminated             ~tutorial/programs/slowandbadprocess all.fasta >slowandbad.out 2>slowandbad.e
rror
student@localhost:~/recentFiles$ █

```

Please note that we need to precede the number of the process by the "%" symbol.

### [Wait for a running program to finish]

**Task:** Wait for a slow process to finish running.

**Comments:** Since the program is running on background and you have the shell free for use, you should run a command that will keep checking if the program is still running. You can use `top`, for this and keep checking the list it produces until `slowprocess` is not listed anymore.

**Issuing the commands:** Just type `top` and keep checking. When `slowprocess` is not on the table anymore, exit `top` by typing the letter "q". To be sure you can try also the command `jobs` which will list all the programs you started on the shell, even if they have stopped.

### [Copying a file from an user account in a remote computer]

**Task:** Copy to your computer the file `lotsaseqs.fasta`. This file is in the computer `localhost` in the home directory of user `visitor`.

**Comments:** File transfer is probably one of the most common computer tasks today. Millions of people surf the Internet and download all kinds of files. Files are also exchanged using e-mail. However, it is not trivial to make a file available in the Internet. Also, there is always a limit on the size of files that can be transmitted by e-mail. Linux permits transferring ANY file of ANY size between two Linux/UNIX computers that are connected in the Internet provided the user has the appropriate permissions. You just need a user name and password in both machines. This is particularly useful when you are away from your computer and find out that you need one file that you did not put available in the internet. Copying files between different machines is not much different from copying files in the same machine. However, you need to specify some extra information, that is, the internet address of the remote computer and the user account you are going to use in that computer (remember that Linux organizes all system security around user accounts). The command to perform remote copy is `scp` (from Secure CoPy).

**Issuing the commands:** As we said, the files can be in any computer, provided you have access to an account name and password. To simplify here, in our exercise, we are bringing a file from the computer `localhost`, which happens to be the same computer you are in. Since you need an account we provided a visitor account named "visitor". To copy the file you should issue the command (do not forget the ".", that is the destination directory)

```
scp visitor@localhost:lotsasequences.fasta .
```

The format of the command will always be the same, first `scp` the command name, then, when specifying the source file you type the account, "@", the computer address, ":", and the name of the file, followed by the destination, in our case the current directory (remember, if we put a directory as the destination of a copy command, files are copied with the same name). It is very important that you do not put any spaces when describing the source file (you can see that in the command above).

When it is the first time you try to connect to a specific machine, the system will ask you if it should proceed with the following message:

```
The authenticity of host 'localhost (127.0.0.1)' can't be established.
RSA key fingerprint is 2b:bc:07:30:e1:b8:29:4d:ba:98:fe:e0:44:91:94:3e.
Are you sure you want to continue connecting (yes/no)?
```

If this is the case, just type "yes" then the "enter" key.

Next, the computer will ask for a password. This is only natural, since the computer is checking if you are really authorized to use the account. Type now the password "visitor" (do not type the quote symbols!). The system will show the progress of the copying and will return to the shell. Now you can check if the file was really copied:

```
ls -l lotsasequences.fasta
```

You will see that the file has just been created.

### [Counting FASTA sequences in a file]

**Task:** Check how many sequences were downloaded with the new file.

**Comments:** We have performed this task before, we need to use `grep` to select the FASTA header lines, and then use `wc` to count the number of selected lines.

**Issuing the commands:** We can run the two commands, connecting them with a Linux "pipe".

```
grep ">" lotsasequences.fasta | wc
```

Remember, the first number is the line count, which is also the number of sequences in our new file.

### [Connecting to a remote machine]

**Task:** Connect to an account in a remote Linux/Unix machine, check the files in the home directory, start the *konqueror* browser and run a program. In this exercise you should connect to user name *visitor* (password: *visitor*) in the machine *localhost*.

**Comments:** One of the very nice characteristics of the Linux/Unix world is the connectivity. Once you have an user account in a machine connected to the internet, you can log in that machine from anywhere in the planet, sometimes even from MS Windows<sup>®</sup> machines. You will connect always through a shell. If you have a high speed connection you can even run programs that have a graphical interface. To connect remotely *from* Linux machine, you should use the program `ssh`. Similar to the `scp` command, you will need to specify the user account and the machine address that you are trying to use in the form `user_name@machine_internet_address`. If you

want to start a graphical program remotely, you will need to use the `-X` option for the command, but remember that you need a fast Internet connection for that to be usable. In our exercise our "remote" machine will be the same as our local machine, but if you get a user account on any remote computer this should work as well.

**Issuing the command:** We first connect to the remote machine using the simple version of the `ssh` command

```
ssh visitor@localhost
```

If you are logging in this machine for the first time, you may get a warning message. Just type "yes" and proceed. Please notice that, once you are logged in, the *prompt* (the text displayed before your cursor) changes from "student@localhost:~\$" to "visitor@localhost:~\$", indicating you are now user "visitor" in machine "localhost". You are now connected to the "remote" machine. Try listing the local files:

```
ls -l
```

You will notice that the files listed are not anymore those of your user account. As you have done with `ls`, you can run any program that is available in the remote machine. Before proceeding, exit from the remote machine by typing

```
exit
```

You should notice that your prompt now indicates that you have returned to your account. Now try to connect again with the `-X` option. That should start a graphical connection which will enable use to run graphical interfaces. Since, for convenience, our "remote" and "local" machines are the same, we can be sure that the connection is fast enough. To log in with the `-X` option, type

```
ssh -X visitor@localhost
```

After typing the password again (*visitor*), you can now start the graphical program `konqueror`. To keep your shell in use, we will start the program running in the background (by adding "&" at the end of the command):

```
konqueror &
```

Not you are running a program in a remote machine, your local computer is used only to display the results, but the remote machine's CPU does the actual processing. The process to quit your remote session is the same just type

```
exit
```

You will notice that, when you exit, `konqueror` will be killed automatically too.

## [Changing permissions of a file]

**Task:** Change the permissions of files *latestPaper\_v2.txt*, *newFile1.fasta*, *newFile2.fasta* and *all.fasta*. You will make *latestPaper\_v2.txt* only available for reading by you and the group. *newFile1.fasta*, *newFile2.fasta* should be protected from any access by others (including user of the same group), and *all.fasta* should be a file where everyone can write.

**Comments:** To change the permissions of files in Linux we use the command `chmod`. As we have said in the introduction, we have three types of permission (execution, reading and writing), along with three categories of users (owner, group, and others).

The general form of the `chmod` command is:

```
chmod who<operation>permission
```

Where *who* designates which group(s) of users we want to set the permission, *<operation>* indicates if we want to add or remove a permission and *permission* indicates which permission(s) we want to set. Below we describe the possible values of each one:

*who*

u:owner

g: users of the same group

o: all users excluding the owner and the ones of the same group

a: all users

*operation*

+ : add permission

- : remove permission

*permissions*

x : execute

r: read

w: write

So, for example, if we want the owner of the file to be able to write in that file we should specify "u+w" (owner, add permission, writing).

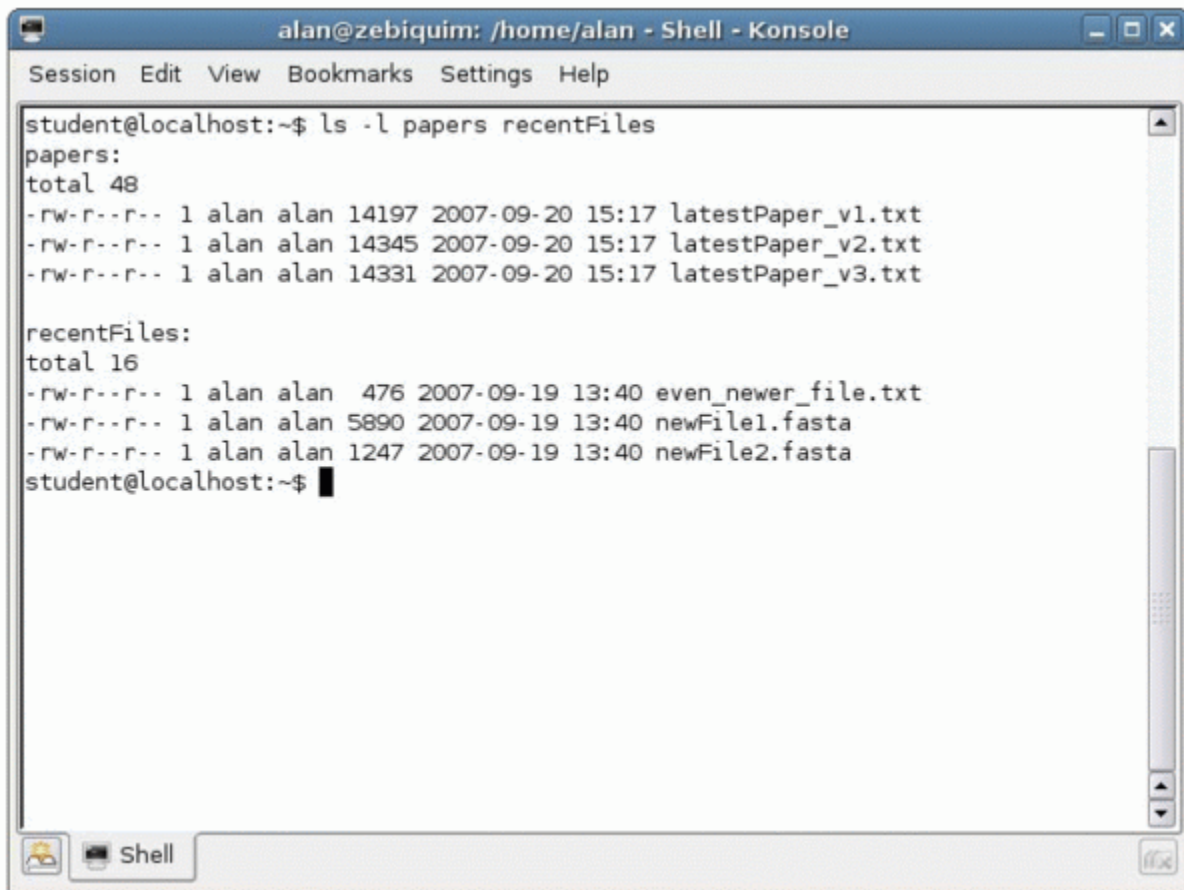
The execution permission deserves a more detailed comment for the interested reader. This permission is used in two cases. First, when we have a computer program that we want to run, the file that corresponds to that program needs execution permission. The second case is for directories. Execution permission means we can include this directory in a path. To clarify this concept we will use a metaphor: we can say that a directory is like a door to a room, where there are possibly other doors (directories) and files. If you have **reading** permission to a directory, you can "turn on the light" in the room and check what is in it, that is you can "see" which are the files and directories in it. If you have **writing** permission to a directory, this means you can add new subdirectories (adding other doors), adding new files to that directory, or even changing the names of the files. If you have **execution** permission, then you can "go through" the room to one of its doors. In other words, you need execution permission to a directory if you want to access any of its sub-directories. This provides a very flexible protection mechanism. You can have directories where users have execution permission, but no reading or writing permission, for example. In this case, they can access files in sub-directories, if they know the complete path, but they will NOT be able to investigate what is stored in this intermediate directory, that is they need to know beforehand which are the name the directory they want to use. This gives the user more freedom in organizing their home area, separating protection issues from organization issues as much as possible.

There are other ways of setting permissions using `chmod`, the interested reader should check the Linux manual.

**Issuing the command:** In order to avoid too much work, we should first check the file permissions using the `ls -l` command for the directories *papers* and *recentFiles*. To be sure we are all in the same place, first we will go to the home directory.

```
cd
```

```
ls -l papers recentFiles
```



```
alan@zebiquim: /home/alan - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~$ ls -l papers recentFiles
papers:
total 48
-rw-r--r-- 1 alan alan 14197 2007-09-20 15:17 latestPaper_v1.txt
-rw-r--r-- 1 alan alan 14345 2007-09-20 15:17 latestPaper_v2.txt
-rw-r--r-- 1 alan alan 14331 2007-09-20 15:17 latestPaper_v3.txt

recentFiles:
total 16
-rw-r--r-- 1 alan alan 476 2007-09-19 13:40 even_newer_file.txt
-rw-r--r-- 1 alan alan 5890 2007-09-19 13:40 newFile1.fasta
-rw-r--r-- 1 alan alan 1247 2007-09-19 13:40 newFile2.fasta
student@localhost:~$
```

As we can see above, everyone has reading permission, and only the owner has writing permission to all these files. We will first change the permissions of file *latestPaper\_v2.txt*. We need to remove reading permission for the rest of the world, and also remove writing permission to the owner:

```
chmod o-r papers/latestPaper_v2.txt
```

```
chmod u-w papers/latestPaper_v2.txt
```

Next we want to change the permissions of *newFile1.fasta* and *newFile2.fasta*. We need to remove reading permission for the group and for the rest of the world. All this can be performed in a single command

```
chmod go-w recentFiles/newFile*.fasta
```

Finally, we want to add universal writing permission to *all.fasta*:

```
chmod a+w recentFiles/all.fasta
```

You can now list the files again and verify their permissions and, as a test, try to remove the file *latestPaper\_v2.txt*. You should get an error message (see figure)

```
ls -l papers recentFiles
```

```
rm papers/latestPaper_v2.txt
```

```

alan@zebiquim: /home/alan - Shell - Konsole
Session Edit View Bookmarks Settings Help
student@localhost:~$ chmod o-r papers/latestPaper_v2.txt
student@localhost:~$ chmod u-w papers/latestPaper_v2.txt
student@localhost:~$ chmod go-w recentFiles/newFile*.fasta
student@localhost:~$ chmod a+w recentFiles/all.fasta
student@localhost:~$ ls -l papers recentFiles
papers:
total 48
-rw-r--r-- 1 alan alan 14197 2007-09-20 15:17 latestPaper_v1.txt
-r--r----- 1 alan alan 14345 2007-09-20 15:17 latestPaper_v2.txt
-rw-r--r-- 1 alan alan 14331 2007-09-20 15:17 latestPaper_v3.txt

recentFiles:
total 24
-rw-rw-rw- 1 alan alan 7137 2007-09-20 15:43 all.fasta
-rw-r--r-- 1 alan alan 476 2007-09-19 13:40 even_newer_file.txt
-rw-r--r-- 1 alan alan 5890 2007-09-19 13:40 newFile1.fasta
-rw-r--r-- 1 alan alan 1247 2007-09-19 13:40 newFile2.fasta
student@localhost:~$ rm papers/latestPaper_v2.txt
rm: remove write-protected regular file `papers/latestPaper_v2.txt'?

```

### 1.2.3 Frequently Asked Questions (FAQ)

This section contains a description of the UNIX commands in the form of a FAQ. Titles will be things like: 'how to copy, move and remove files', 'finding files by name', etc.

UNIX programs that will be discussed are:

basic: ls, cd, mkdir, rmdir, mv, rm, cp (recursive), more or less find, grep, top (ps), scp, ssh, emacs, nedit, apropos, man, wc, wget, sort, cat, diff

- **FILES AND DIRECTORIES**
  - **What is a directory or folder?**

*Directory* and *folder* are two names for the same thing. Think of a directory as a special place where you can put files or even other directories. This resource helps you to organize your files and should be used extensively.

Any permanent information is located inside a directory. The main directory is called 'root' and referenced by '/'. Directories within other directories are indicated by separating the names with '/'. For instance, '/var/spool/mail' refers to the folder 'mail' which is located inside 'spool' which in turn is inside 'var'. 'var' is in the root directory.

When you are using the computer, you are always 'located' at one directory — we call it your *current directory*. When you log in, you are placed in your 'home' directory, which usually is /home/[your login name].

A file, and even a folder, may appear in more than one directory, using a 'shortcut' or 'link', see the ln command.

- **How to list files and directories?** The command ls shows all the files and directories located in the current directory.

If you want to list another directory, just pass it as an argument to ls:

```
ls /etc
```

```
ls /var/log
```

```
ls /home/student
```

You can also restrict the list to a subset, using '\*' for any part of the name, '?' for any letter and '['...]' for some a limited set of characters.<sup>6</sup>

```
ls *.fasta
```

```
ls hepatitis[ABC].seq
```

```
ls protocols/x?.prt
```

- **How to create a directory?** To create a new directory use the command mkdir and the directory name as its argument. You can create more than one folder in the same command.

```
mkdir MyNewDirectory
```

```
mkdir Project1 Project2
```

```
mkdir MyNewDirectory/Sequences
```

- **How to remove a directory?** Use rmdir, but be aware that the directory must be empty. You can remove more than one in the same command line.

```
rmdir Project1 Project2
```

---

<sup>6</sup> This is in fact a characteristic of the shell. You can pass arguments like this to any program, when using the command line.

```
rmdir MyNewDirectory/Sequences
```

```
rmdir MyNewDirectory
```

You can also remove a directory and all its contents quickly by using `rm -rf`, but this is *very dangerous*, since you can potentially erase large amount of data without the possibility of recovery. Only try this one if you are **absolutely sure** of what you want to do.

```
rm -rf MyNewDirectory
```

- **What is the current directory and how I select one?** The *current* directory is the one you operate by default — it is your 'working directory'. For instance, when you execute `ls` without arguments, it will list the files in the current directory. You can verify which is your current directory with the command `pwd` (print working directory).

To select another working directory use the command `cd` (change directory). Just use the new folder as an argument.

```
cd ..
```

```
cd /etc
```

```
cd /usr/share/doc
```

- **How to remove a file?** The command `rm` permanently removes a file — use with care! You can specify sets of files exactly in the same way you do with the `ls` command.
- **How to move a file from one directory to another?** Use the `mv` command. Pass the list of files and/or folders you want to move in the command line. The last argument must be the destination folder. You can specify sets of files and folders as you do in the `ls` command.
- **How to rename a file?** Use the `mv` command (the same used to move files). The first argument is the name of the file or folder and the second must be the new name.
- **How to copy files?** You can copy files using the `cp` command. The last argument is the destination. You can copy a set of files to a new directory or make a new copy of a single file.

```
mkdir Fastas
```

```
cp *.fasta Fastas/
```

If you use the `'-r'` option you can make a recursive copy. This means you can copy a directory and all its subdirectories to a new location.

```
cp -r directory1 targetName
```



- **How to find a file in the system?** The simplest way to find a file is using the locate command, but it is not always implemented. It is fast and uses a database to get the information. This database is updated periodically and may not be up to date when you execute the program.

The safer, but much slower method, is using the program find. It actually searches all directories for the name you provide.

Since find can do a lot more than just look for files, its syntax is a bit complicated, but the basics are easy to master. Here we will show how to use it to find a file by its name. To do that, just follow this template:

```
find folder -name 'file name'
```

*folder* is the starting directory for the search, find will search it and all subdirectories within it. *file name* is the name of the file you want to find, you can use the '\*', '?' and '[...]' constructs as you do with 'ls'. You must be careful with shell interference: always surround arguments with (').

## CONTENTS OF FILES

There are many types of files: images, sounds, formatted or raw text, and so on. Special types need special viewers (images for instance). The available viewers depend greatly on the installed system. The following table shows the most popular viewers commonly found in Linux systems.

type	to view
images	display
movies	mplayer, totem
sound	mplayer, play
postscript	gv
pdf	gv, acroread, xpdf
doc, xls, pps	openoffice

The rest of this section will cover text files

- **How to view the contents of files?** To view the contents of text files, there are a few options. cat just copy the file to the output. If the file is short, this a quick and good way.

more displays the file content, one page at a time. You can browse the pages back and forth and even search for patterns.

less is just another version of more, a little bit more sophisticated. It is not always present, and in some implementations both programs are the same.

- **How to search a file?** grep will take any string as its first argument and print all the lines which contain that string in the files given by the following arguments.

```
grep AAAAAAA *.fasta
```

The string may be a 'regular expression', which is a compact way to describe a set of strings — see the grep man page for details.

- **How to check the number of lines, words and chars?** The program `wc` (word count) prints three numbers: the number of (complete) lines, words and characters for each of the files given in the arguments. If more than one file is given, an extra line with the total counts is appended.

```
wc *.fasta
```

```
wc tutorial/researchGroup/dir*/latestPaper_v*
```

`ls -l` will give the total size of the files, among other information, like ownership and last modification date.

- **How to sort the contents?** `sort` will print the lines sorted in lexicographic order, if you use `'-n'` they will be sorted by numerically.

```
grep '>' *.fasta | sort
```

The comparison starts at the beginning of each line. To sort by another part of the line, use `-k position` as an argument. *position* indicates the word in the line you want to use. 1 is first, 2 is second, and so on.

Words are separated by whitespaces. This option is particularly useful if you want to sort a table by one of its columns. To use another separator instead of whitespace, specify it with the `'-t'` option.

```
ls -l *.fasta | sort -n -k 5
```

- **How to compare files?** The best way to compare two large and similar text files is to print only the differences between them. This is exactly what the `diff` command does. It receives two files in the command line and prints the missing, added and changed lines, if the second file is seen as a modification of the first.

Among the several output formats, the most interesting are the detailed (which is the default), the unified (option `-u`), and the 'side-by-side' (option `-y`).

```
diff New.fasta Old.fasta
```

```
diff -u New.fasta Old.fasta
```

```
diff -k New.fasta Old.fasta
```

- **Editing a file.** There is a large number of text editors available in UNIX. They range from simple line editing tools like `ed` to very sophisticated programs like `emacs`. The most popular among UNIX users are `vi` and `emacs`, but a good start would be `kate`.

`kate` is quite powerful but still easy to operate editor. All operations can be done by using menus and the interface is quite intuitive.

`emacs` is a very powerful and flexible editor and has the advantage of being present almost everywhere, but it requires a longer time to really master it. It is worth to experiment the basics and check if you like it.

## PROCESSES

- **What is a process?**

A process is a program being executed. A *program* is a piece of software resting on some disk and which you can run. When you 'start' the program, it is copied to the main memory for execution.

You may have more than one instance of the same program running at the same time, that is, more than one process associated to the same code.

- **How to list the active processes?**

Just like `ls` lists the files, `ps` will list the processes of the current user, but a more interesting tool is the `top`: it displays all active processes in a table which is dynamically updated every few seconds (the refresh rate can be adjusted with the option `-d`).

```
ps
```

```
ps -ugx
```

```
top
```

```
top -d 2
```

- **How to terminate a process?**

If the process has a text interface, you can normally terminate it by pressing *Control-C* at point where the program asks for data.

In a graphic environment, there is usually a button located at the title bar which terminates the process. Clicking the right button of the mouse also presents a menu with a termination option.

In any situation, you can use the `kill` command on a terminal. Just type `kill` followed by the number of the job associated to the program. Alternatively, you can find the process id number using `top`, and use the `kill` command with the process number. You should use the `-9` option if this does not work:

```
kill %1
```

```
kill 1243
```

```
kill -9 1243
```

- **How to start more than one process?**

When you start a process from within a terminal, it 'occupies' the terminal and control all the input and output there — we say that it is executing in *foreground*.

To get access to the terminal again, you can signal the process to 'sleep' by typing *Control-Z*. You can then issue a command to make the process resume its operation. There are two possibilities: `fg` puts the process in foreground again, but `bg` resumes the process without giving the terminal control back to it, allowing you to issue other commands at the same time.

Another, much more direct, way, is to add an `'&'` right after each command you want to run in background. You can start several processes at the same time this way.

```
slowprocess &
```

```
program1 & program2 & program3
```

## NETWORK

- **What is an URL?**

Information is available in the Internet in several different ways. It may be a web page, a video or audio stream, a program you can execute remotely or simply a set of files that can be accessed using some *protocol*.

The protocol is just a set of rules a program must follow in order to get the information. The http protocol, for instance, dictates the steps a browser must follow to receive a valid web page. Fortunately, you do not need to know how a protocol works (this is done by the program you are using), but you should know under which protocol the information you need is made available.

An *URL* (Universal Resource Locator) is a way to completely specify the requested information. It has the following structure:

*protocol://address/contents*

These are some examples:

- <http://www.google.com/> A web page.
- <ftp://ftp.us.debian.org/welcome.msg> A file in a *ftp* server.
- <file:///etc/protocols> A file on your local machine.

**How to connect to other machines** The best and safer way to establish a connection is by using the ssh command. It may not be commonly available, but is becoming more and more popular.

To use it, just type:

*ssh login@machine*

*login* must be a valid user name on the machine you want to connect. If the login is not specified, the name of the current user machine will be used instead. After you enter your password successfully, the terminal you are using becomes a terminal from the remote machine.

If you use the option **-X** **and** if this is allowed by the remote server, you will even be able to export the X-Window interface and open graphical programs remotely.

*ssh myusername@remote.machine.org*

*ssh -X myusername@remote.machine.org*

**Fetching files from the network** If you are willing to retrieve files from an account you have on a remote machine, the scp program is the safer choice. Its usage is very similar to that of cp, the difference being that you must precede the source or destination with the name of the remote machine, followed by a colon (:), like this

*scp login@machine:this that*

As with the ssh command, *login* is your username on the remote machine. scp can be used to copy files from either direction. Use scp -r to copy directories recursively.

If, on the other hand, you are willing to fetch files from a server, use the wget command instead. You must know the *URL* of the files you want to retrieve, then just type at the prompt:

*wget -nH url*

The `-nH` option tells `wget` not to create subdirectories for each host you access.

## 1.2.4 Exercises

### Short exercises

Let's practice! Here is a series of very short exercises, which use one or two commands. Try them in sequence and refer to the previous section or the manual in case of doubt.

1. **ls:** List all files in the current directory whose names are longer than 3 characters
2. **ls:** List all files in the current directory whose names contain the letter 'a'.
3. **ls:** List all files in the current directory whose names have exactly 3 characters.
4. **mkdir:** Create a subdirectory called Funny in the current directory and move into it.
5. **mkdir** Create another subdirectory, called Silly, in the parent directory.
6. **ls, output redirection:** Generate a file with the list of all files in the parent directory, and call it bozo.
7. **cat:** Print the contents of bozo file on screen.
8. **mv:** Move file bozo from the current directory to directory Silly, created above.
9. **rm:** Remove Funny, but remember you are still "in" it.
10. **cp:** In the directory Silly, make a copy of bozo and call it clown.
11. **ls, grep:** List all files in the /etc directory which were created or modified in November, 2005.
12. **ls, grep, wc:** How many of them are there? Count them automatically.
13. **cd, ls, output redirection:** Go into diectory Silly of your home directory (if you are not there already) and create a file named passpatou with the output of `ls /etc`.
14. **cat, output redirection:** Append the contents of passpatou to bozo.
15. **cat, output redirection, mv:** Put the contents of passpatou at the *beginning* of clown (you will need a temporary file).
16. **mkdir, mv, rmdir:** Create directory example3b, move all files from directory example3 into the new directory, remove directory example3
17. **wget:** Copy the book's home page in your home directory (<http://www.bioinfobook.org/index.html>),
18. **wc** Check how many words there are in file examples/article1.txt
19. **diff, ls, >** Compare the contents of the directories example1 and example2 to check which are the files that are not in both directories
20. **grep, wc** Check how many files in directory example1 were not in example2
21. **cd:** a) Go to the directory /examples. b)Now show two different ways to go back to your home directory.
22. **sort, more** List the names in file examples/names in alphabetical order, one page at a time
23. **background processing, top, kill:** Run programs slowandbadprocess and slowprocess in the background. Check which one uses more CPU and kill it. Remember if you want to kill a process use the command "jobs" to check the process number
24. **tail:** Check the last 20 lines of file example1/interestingText.

Answers:

1. `ls ?????*`
2. `ls *a*`
3. `ls ???`
4. `mkdir Funny`  
`cd Funny`
5. `mkdir ../Silly`

6. `ls .. > bozo`
7. `cat bozo`
8. `mv bozo ../Silly/bozo`
9. `cd ..`  
`rmdir Funny`
10. `cd Silly`  
`cp bozo clown`
11. `ls -l /etc | grep "2005-11"`
12. `ls -l /etc | grep "2005-11" | wc`
13. `cd ~/Silly`  
`ls /etc > passpatou`
14. `cat passpatou >> bozo`
15. `cat passpatou clown > auxFile ; mv auxFile clown`
16. 2 possibilites:
  - a. `mkdir example3b`  
`mv example3/* example3b`  
`rmdir example3`
  - b. `mv example3 example3b`
17. `wget -nH http://www.bioinfobook.org/index.html`
18. `wc examples/article1.txt`
19. `ls example1 > file1`  
`ls example2 > file 2`  
`diff file1 file2`
20. `diff example1 example2 | grep "verb<"|wc)`
21.
  - a. `cd examples`
  - b. `cd`  
*or*  
`cd ..`
22. `sort examples/names | more`
23. `slowandbadprocess &`  
`slowprocess&`  
`top`  
`jobs`  
`kill $1`

24. `tail -n 20`





## Chapter A02. Understanding Database Design

João Eduardo Ferreira<sup>1</sup> and Osvaldo Kotaro Takai<sup>2</sup>

Created: September 12, 2007.

### Context

Today, more than in any other moment in history, public and private institutions depend on the ability to keep precious, up-to-date data regarding their activities in order to manage business and research, as well as to continue being competitive in market. In particular, bioinformatics applications often generate very large data sets that are stored through flat files and spreadsheet formats. In recent years, with the advance of bioinformatics and biologic techniques, the amount of data stored in computer systems has grown exponentially. Hence, it is essential that bioinformatics researchers employ tools to simplify the management of data, thereby allowing for the quick extraction and integration of useful information. In particular, the use of data mining techniques requires data to be stored in structured databases. In a database, information is more readily accessible and easy to query, good examples in the Tropical Diseases area are PlasmoDB and ToxoDB. However, data stored in simple files is still the rule rather than the exception in the bioinformatics area. This chapter will present the main concepts of database design and query, and will help the reader to understand the power of the database approach to storing data, as well as how queries can be constructed from a specification of database architecture.

The set of tools designed and employed for this purpose is known as a **Database Management System (DBMS)**.

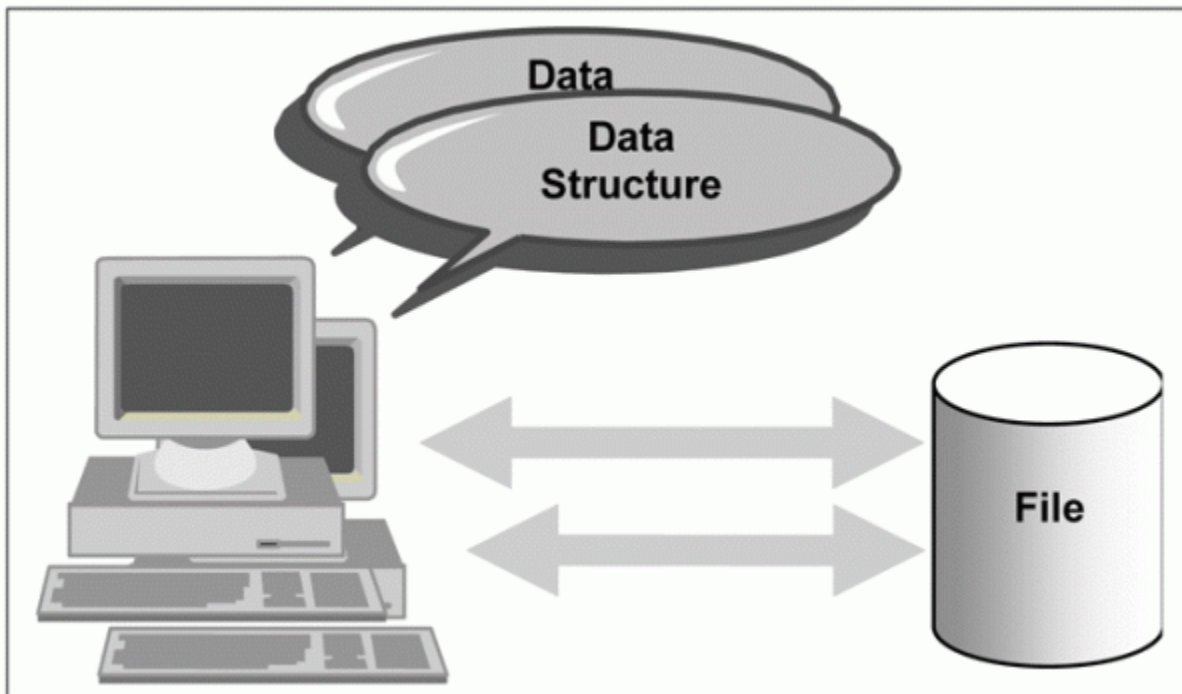
We will use a simple and practical example of a database of clinical data to illustrate the main concepts related to the creation and the maintenance of a database. In Section 1, we will give a brief introduction to DBMS's.

Section 2 discusses the conceptual design of databases. Section 3 presents the *Structured Query Language (SQL)*, a standard language used to create and manipulate databases. In Section 4 we present the Relational Model of data that is used to implement the database design.

### 1. Introduction

In the early years of computer technology, the main purpose of computer programs was to store and manipulate data. These programs recorded their data on disk according to their own structures. All programs that were not familiar with the structure of the data were unable to use them.

If various programs needed to share data from the same file, they would have to know and manipulate the same structures. The figure below gives a good picture of these situations.



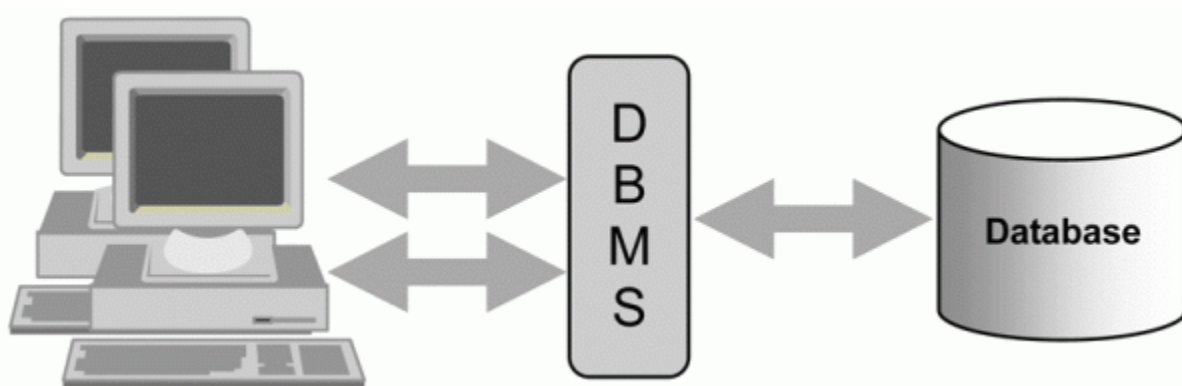
If any one of these programs needed to change the data structure, all other programs (which were accessing the same file) would have to be changed, even if the alteration took place in data that was not manipulated by all the programs. This led to a major problem: *guaranteeing the unity of the data structures among the different programs due to the existence of redundancies.*

In order to avoid this problem, an intermediate system was set up to convert the data from the format they were recorded in the file to the specific format recognized by each program.

With this intermediate program, the following occurs:

- The programs "see" only the data that they have to interact with;
- The programs don't need to know the details of the physical recording of their data;
- The programs don't have to be modified when the data structure is modified;
- The alterations are concentrated in this intermediate system.

Eventually, these intermediate systems were able to manage many different files. This group of files was named **Database** and the intermediate system became known as **Database Management System** (or **DBMS**). The next figure shows how a DBMS accesses data.



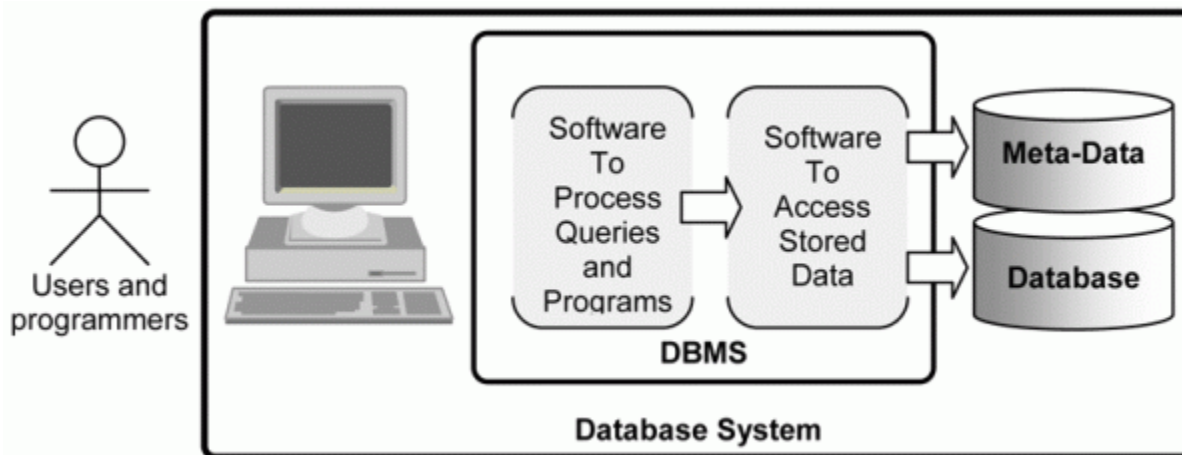
A **database** is currently defined as *a collection of coherent data that is logically related with some associated meaning.* A chaotic collection of data cannot be called a database. A database is designed, constructed and

populated with data to meet a specific purpose and public. Furthermore, a database represents some aspect of the real world, sometimes referred to as **miniworld**. Changes in the miniworld are reflected in the database.

In other words, a database has a source from which all the data originate, some degree of interaction with events in the real world and a public that has genuine interest in the contents of the database. The DBMS is the software that incorporates functions that define, recover and change the data within a database.

- Designing a database involves specifying the types of data to be stored in the database;
- Setting up a database is the process by which the data is stored in some means of storage controlled by the DBMS;
- Manipulating a database means using functions such as queries to recover specific data, making changes to the database to reflect changes in the miniworld (insertions, updates and removals), and creating reports.

The first commercial Database Management System was released in the late 1960's and was based on primitive archiving systems available at the time. These DBMS's did not control the competing accesses of different users or processes. The DBMS's evolved from these archiving systems with on-disk storage, creating new data structures with the aim of storing information. Sometime later, the DBMS's began to use different types of representation, or **data models**, to describe the structure of the data found in their databases. This description of databases according to a data model is called **meta data**. The group of application programs which use a DBMS is called a **Database System**. The figure below presents a general diagram of a Database System and how it interacts with users.



## 1.1 Describing and storing data in DBMS's

For a DBMS to be able to store and recover information from a database system, this information must have a structure. The description of this structure, or **scheme**, is maintained by the DBMS in a database referred to as Meta Data.

Applications in the real world are usually rich in concepts and there are various associations among these concepts. In a university's academic system, for example, there will likely be concepts such as: Courses, Outlines, Classes and Students; and associations between these concepts: Courses have Outlines, Outlines are made up of Lectures and Students enroll in Courses. These concepts and associations, which constitute the **application scheme**, should be captured from the miniworld (the university) and properly stored in a DBMS.

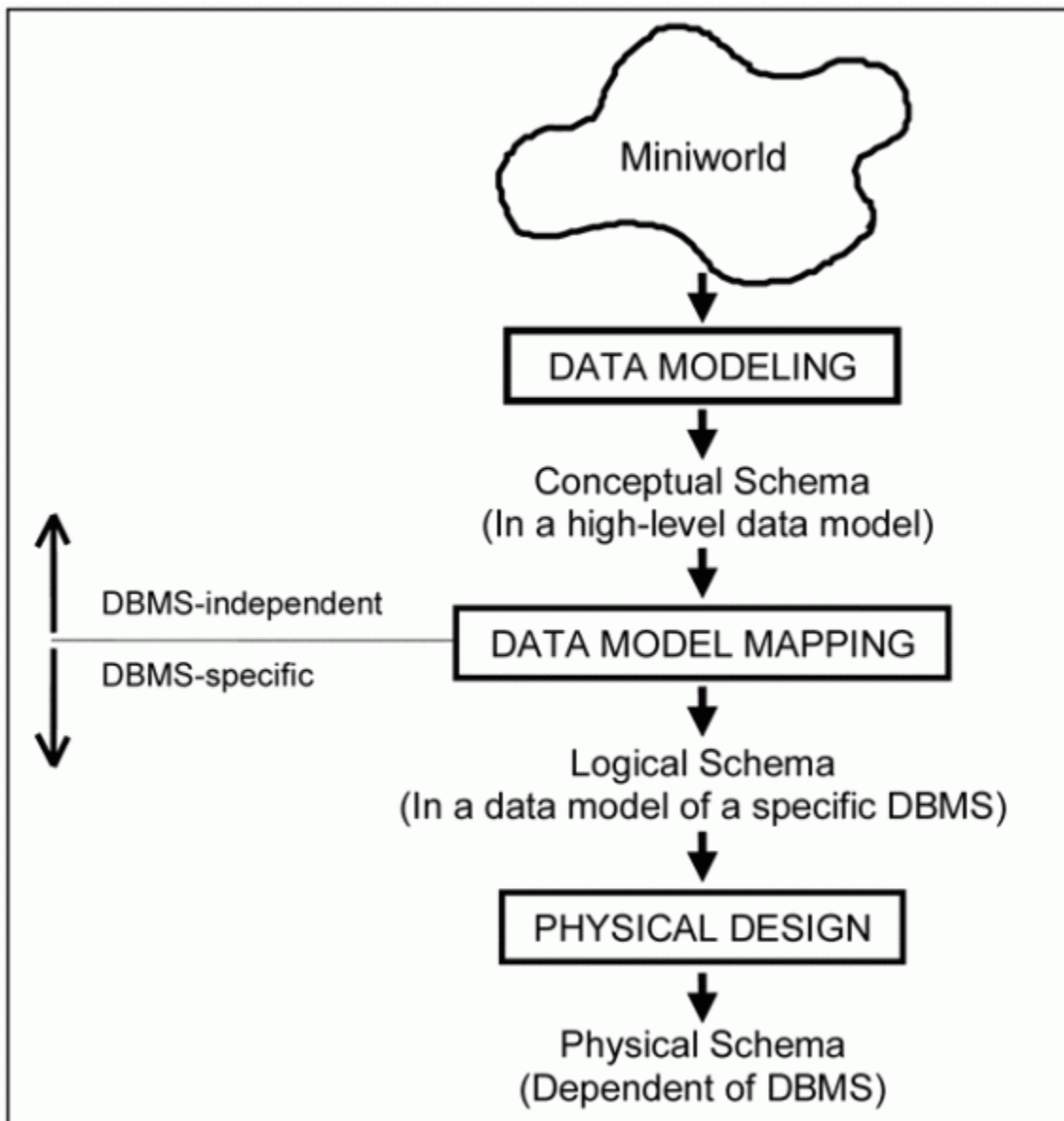
Unfortunately, the database technology that is currently available does not allow high-level representations of an application scheme, such as the one in the example above, to be directly represented in the DBMS's, because

these can only be understood by human beings. A DBMS requires a high-level scheme (or conceptual scheme) to be converted into a representation that can be manipulated by the computer (the logical scheme).

The application scheme used by the DBMS is obtained through a three-step approach, known as a Database Design:

- a. Data Modeling: the process used to conceive the Conceptual Scheme;
- b. Mapping the Data Model: the process by which the Conceptual Scheme is transformed into the Logical Scheme;
- c. Physical Design: the process of building the Physical Scheme (the application scheme that is manipulated by the DBMS) from the Logical Scheme.

These steps are illustrated below:



The first step is executed by database designers, which make use of a group of concepts and rules in order to guide them along the task of structuring the information from the miniworld. This group of Concepts and Rules is the **Conceptual Scheme**.

The second step, which is also performed by the database designer, transforms the Conceptual Scheme into the Logical Scheme according to the **Logical Data Model** employed by a specific DBMS (such as the Relational Model, which will be discussed in section 3).

The third step involves adding aspects of implementation to the Physical Scheme (indexing and file structure, transactions and concurrency control, optimization, recovery in case of failure, protection mechanisms, partitioning and data grouping) in order to obtain the Physical Scheme. This chapter will not cover the Physical Schemes as it is the subject of advanced database courses.

## 1.2 The need for the Data Modeling Approach

In order to illustrate the concepts regarding database design, let us use a hypothetical application of a clinical database. The clinical database should store, for example, patient information, samples and DNA sequences. Let us suppose that the designers initially came up with the following description of the miniworld:

- The following patient information should be controlled: identification code<sup>1</sup>, gender, date of birth, city of birth, country;
- From the samples, the most important information is: sample code, country of origin of the sample, body part from which it was taken, number of sequences generated by a given sample; the sequence data is represented by the sequence's identification, the region of the genome, the length of the sequence and its respective DNA sequence;
- A patient may have more than one sample associated to him/her. On the other hand, a sample is associated to only one patient.
- A sample may have many sequences associated to it; however, a sequence is associated exclusively to one sample.

In the next sections, we will present the database concepts with which database designers must be familiar in order to create an application's conceptual model.

A first, straightforward way of representing our problem is shown in the Table below, which depicts all data belonging to patients, samples and sequences:

Code	Gender	Birthday	Country	Code Sample	Body Organ	Collection Date	Code Sequence	Genomic Region	DNA Sequence
0001	M	01/01/1970	Brazil	001	Liver	10/10/2005	1111	ENV	'atgcctgacttgctacccttggaatcga
0001	M	01/01/1970	Brazil	001	Liver	10/10/2005	1112	POT	atgcctgacttgctacccttggaacttaa'
0001	M	01/01/1970	Brazil	001	Liver	10/10/2005	1113	null	aatcga...tactttactttaccataacttaa'
0001	M	01/01/1970	Brazil	002	spleen	05/05/2005	2222	null	accataacttaa accataacttaa
0001	M	01/01/1970	Brazil	002	spleen	05/05/2005	2223	POT	tacttactttactttactttaccataacttaa
0001	M	01/01/1970	Brazil	002	spleen	05/05/2005	2223	POT	tacttactttactttactttaccataacttaa
2040	F	02/02/1980	Angola	005	lung	02/02/2006	3333	ENV	'atgcctgacta...tactttactttaccataacttaa
2040	F	02/02/1980	Angola	005	lung	02/02/2006	3334	Null	aatcga...tactttactttaccataacttaa'

As we can see in the table, one patient has one or more samples. Each sample has one or more sequences. In this table we have data replications for patients and samples for each sequence. For example, the patient 0001 has six

<sup>1</sup> In clinical databases, patient ID is normally done indirectly. Hence, instead of giving explicit reference to the patient by name and last name, a code is used to indirectly identify the patient.

replicates and two samples. His samples 001 and 002 are replicated three times, for each sequence data. The patient 2040 has three replicates for one sample that has one sequence.

This table is a non-optimized data structured, which present many disadvantages related mostly to the redundancy in the storage of data (partially repeated rows). In particular we can mention:

- larger disk storage
- consistency control – repeated information has to be the same, a potentially complex task
- difficulty in the updating process, in particular when updates must be propagated to different rows (for example, correcting a mistake in the patient's gender)
- clarity on query results: for example, a query of which patients have samples collected from the liver will return patient 001 three times.

The next table summarizes the main characteristics of a system that is developed using a DBMS as compared to a system that is developed using the traditional archiving system.

Traditional File Processing	DBMS	Advantage of DBMS
Data definitions is part of applications programs code	Meta-Data	Redundancy control
Application-data Interdependence	Application-data Independence	Redundancy control Maintainability
Data representations in physical level	Conceptual representation by data and programs	Maintainability
Specifics modules implements each vision	Multiple vision	Query facilities

Although the advantages of using DBMS's are attractive, there are two situations in which they are not recommended:

- When the data and applications are simple and stable – for example, in the case of an address book;
- When an extreme access speed is required. In this case, the applications should store the data directly in the RAM (*Random Access Memory*).

Section 2 will show the correct way to partition this table through the Entity-Relationship data modeling approach and a more complete case study of patients, samples and sequences.

## 2. Data Modeling Using the Entity-Relationship Model

The **Entity-Relationship Model** (ERM) is a data model designed for the purpose of representing the semantics associated with data from the miniworld. The ERM is used in the **conceptual design** stage, where the **conceptual scheme** of the application's database is conceived. Its concepts are intuitive, which ensures that the database designers capture the concepts associated with the application data without the interference of specific database implementation technology. The conceptual scheme that is created based on the ERM is called **Entity-Relationship Diagram** (ERD). The basic concepts of ERM are essential to understand how a database can be effectively queried by users.

**ERM:** Group of modeling concepts and elements that the database designer needs to know.

**ERD:** The Result of the modeling process executed by the data designer that is familiar with ERM.

## 2.1 Entities, Attributes and Groups of Entities

### Entity

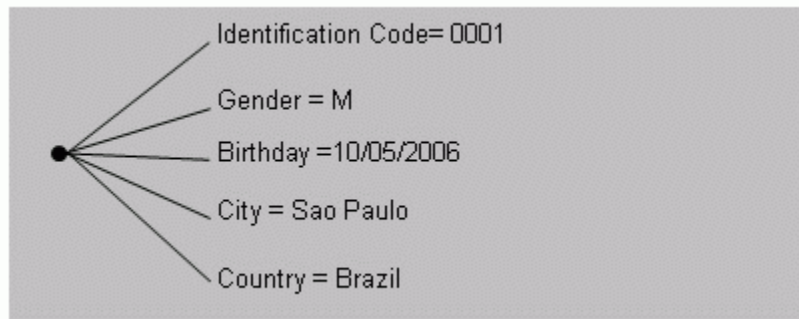
The most basic object represented by the ERM is the **entity**. An entity is something in the real world that exists independently.

For example individual patients, samples and sequences in a clinical database are Entities.

### Attribute

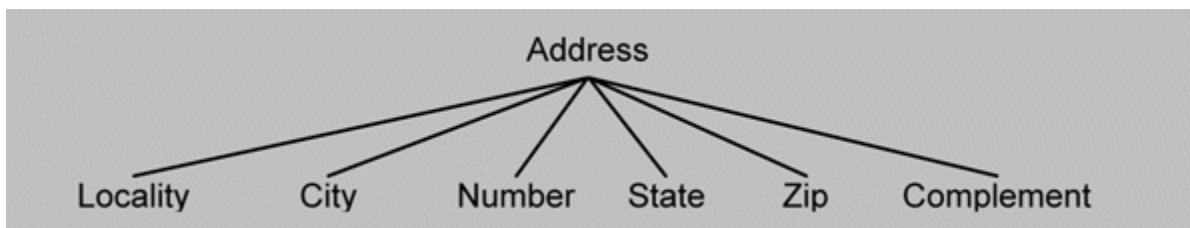
Each Entity contains specific information known as **attributes**.

For example, a patient from the clinical database can be described by his attributes: identification code, gender, data of birth, city of birth and country. Each attribute has a value, (attribute value). For example attribute gender can have the values "male" or "female". Below, we can see Patient entity (e1) with its attributes.



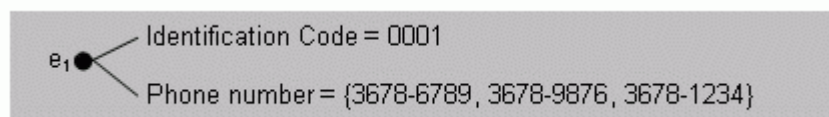
### Composite Attributes

Some attributes may be composed of many parts, or sub-attributes. For example, the attribute *Address*, depicted next, is composed of other attributes, such as city, number, locality, state, zip and complement. Such attributes are known as **Composite Attributes**.



### Multi-valued Attributes

Many attributes have only one value (**uni-valued**). However, there are some attributes that may be composed by a group of values (**multi-valued**). For example, as you can see next, a patient (e<sub>1</sub>) has more than one telephone number, so *Telephone Numbers* is a multi-valued attribute.



## Derived Attributes

Some information about an entity can be obtained by processing the attributes. For example, we can obtain a patient's age from its Date of Birth. This information is called derived attributes. Thus, age is a derived attribute where:

- $\text{age} = \text{Current Date} - \text{Date of Birth}$ .

## Null Value

Sometimes an attribute of a specific record may have an undefined value. In these cases, a **null value** is given to this attribute. For example, not all patients have addresses with an apartment number. In that case, the attribute apartment number will have the value *null*. Another case where null value can be applied is when the value is unknown. For example, a patient may not know exactly when he/she was born. Hence, a null value is used.

**Null Value** is used whenever the value of an attribute **does not apply** or when its value is **unknown**.  
**Attention!** Null Value is not the same as zero (0).

## Key-Attribute

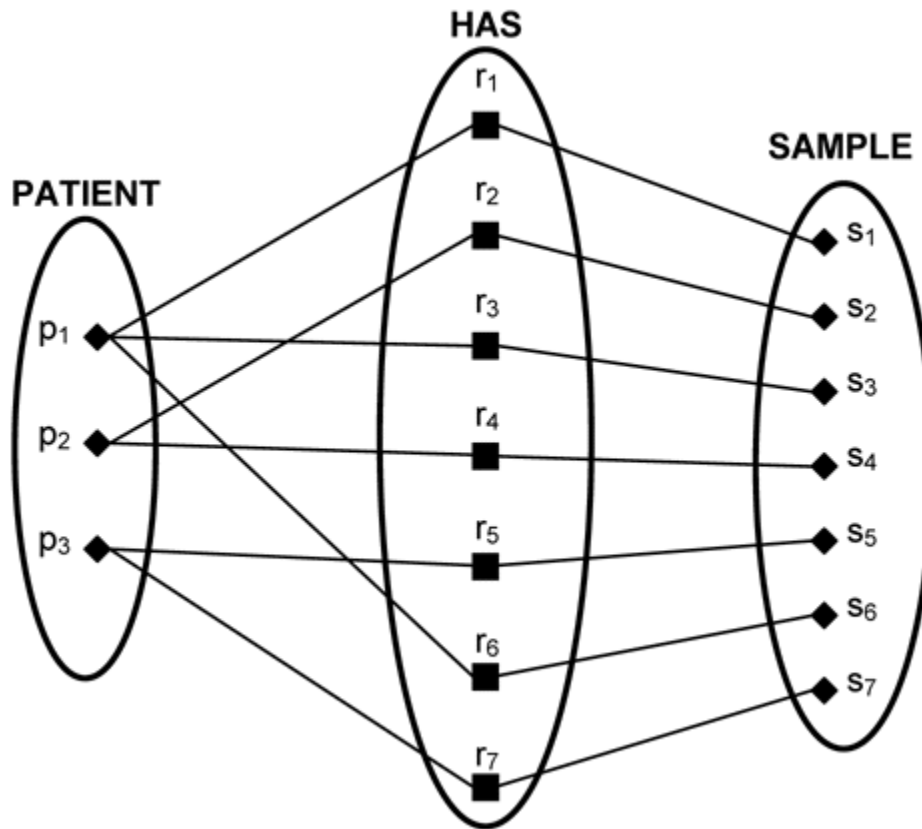
Entities must have an attribute whose value that make them unique in the database. We call these attributes key-attributes. All Entity Types must have at least one key-attribute; it doesn't matter if this attribute is simple or composite.

The values of a key-attribute must be unique. For example, one key-attribute of entity type Patient would be the identification code.

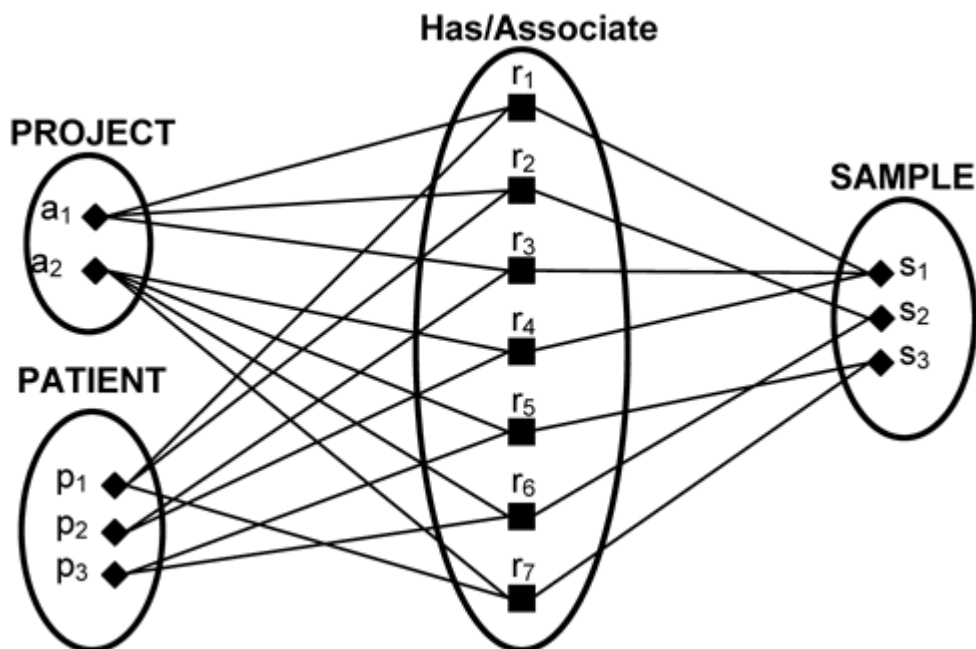
## 2.2 Relationships and Groups of Relationships

What makes the Entity-relationship model so flexible and powerful is the fact that we can establish relations between entities. For example, we can associate samples to patients. To see how this can be expressed in our model, consider the figure below:





Above, you can observe that entity  $p_1$ , from the entity type "Patient", is related to entity  $s_1$ , from the entity type "Sample", by way of the relationship  $r_1$ . A *relationship type* is just a group of similar relationships. In our example, we have the relationship "Has" that associates sequences to patients. Relationships can be established among more than just two entities. For example, Patients may have certain Samples associated with certain Projects. The next figure shows a ternary relationship type (we say relationship type of level 3) that relates patients, samples and projects: Has/Associate.



**Be Careful!** A ternary relationship type cannot be substituted by three binary relationship types.

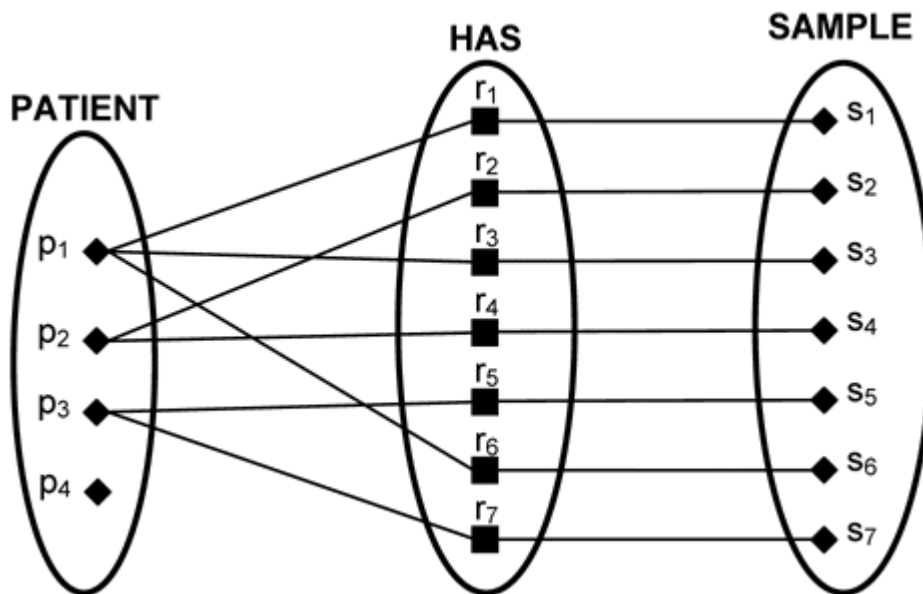
There can be relationship types of any level, but level 2 relationship types are more common.

## Cardinality Ratio

The cardinality ratio specifies the number of relationship instances in which an entity can participate. The binary relationship type Patient HAS Sample, from Figure 2.6, has a cardinality ratio of 1:N (read: *one-to-“N”*). This means that for every Patient entity, we can associate any number of Sample entities. However, a Sample entity can only be related to a single Patient. The most common cardinality ratios for binary relationship types are 1:1 (one-to-one), 1:N (one-to-“N”) and M:N (“M”-to-“N”).

## Constraints on Relationship Types

The cardinality ratio of a relationship establishes two types of constraints on these relationships: *participation constraint* and *structural constraint*. A nice feature of DBMS software is that they automatically enforce these constraints, that is, entities and relationships cannot be entered into the database if the constraints are not satisfied. The next figure illustrated a relationship with cardinality 1:N:



## Constraint on Participation

The constraint on participation specifies whether the existence of an entity defines its participation on a relationship. There are two types of constraints on participation: *total* and *partial*. For example, if the clinical database establishes a rule that every registered Sample must belong to a Patient, then the entity Sample can only exist if it participates in a relationship instance HAS. The participation of Sample in HAS is called total, which means that every Sample entity must be related to a Patient entity via the relationship type HAS. The total constraint on participation is sometimes referred to as existential dependence. Also, it is unreasonable to impose that every Patient be obliged to have a Sample when he/she is registered on the clinical database. Hence, the patient's participation in the HAS relationship is partial. This means that some entities in the group of entities Patient may be related to an entity Sample via HAS, but not necessarily all of them.

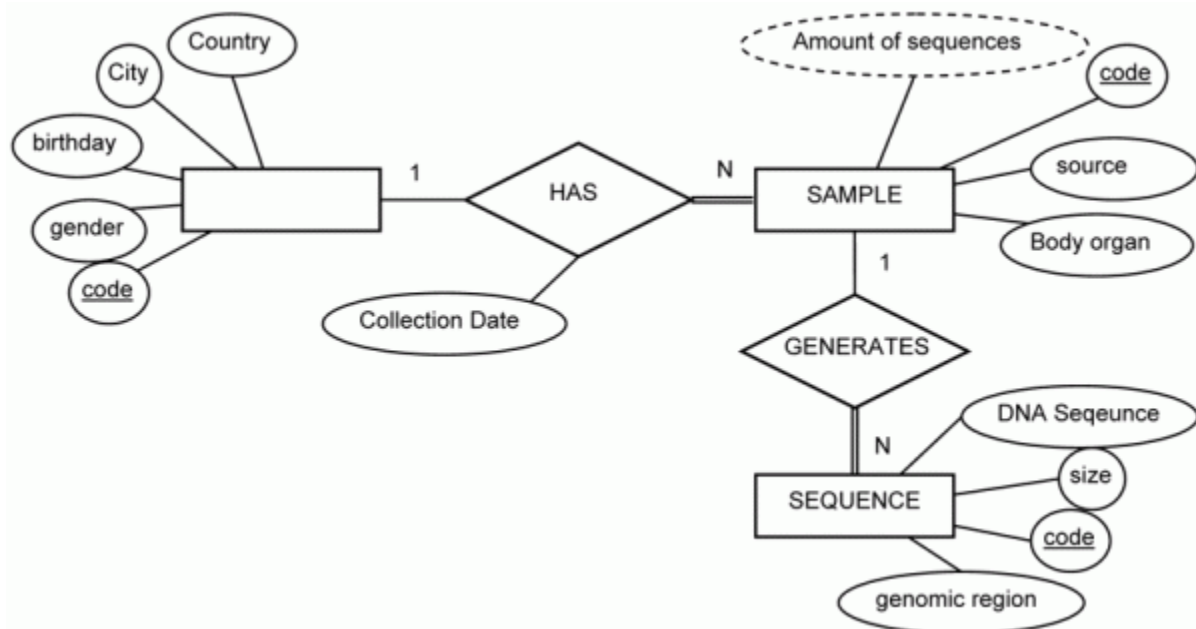
## The **Cardinality Ratio** and the **Constraint on Participation** depend on Rules from the miniworld.

### Structural Constraint

The **Structural Constraint** specifies the **minimum** and **maximum** number of relationships on which an entity should participate. For example, in the relationship type "Patient HAS Sample", a patient may not have a sample; therefore, minimum = 0. However, a Patient can have as many samples as necessary depending on the project. We say that by stating that maximum = N. Hence, the structural constraint of the Patient in relation to the relationship HAS is (0, N). On the other hand, a Sample can only exist if it belongs to one single Patient; hence, minimum = 1 and maximum = 1. Therefore, the structural constraint of Sample in relation to the relationship type HAS is (1,1).

### Relationship Attribute

Relationship Types can also have Attributes. For example, the date when the sample was collected (Collection Date) from a given patient should be represented in the relationship type HAS. Also sequences are associated with blood samples in the relationship GENERATES, as illustrated next:



In this case, the relationship GENERATES does not have attributes.

## 2.3 Conceptual Design of databases with Entity-Relationship Models

The previous section introduced some of the main concepts involved in the Entity-Relationship Model. The creation and specification of the Entity-Relationship model is called the *Conceptual Design*. During this process we identify entity types, their attributes, the key-attributes, the relationships their cardinalities and constraints.

In our example we have three entity types (key attributes are underlined):

- **PATIENT** - identification code, gender, birthday, city, country.
- **SAMPLE** - sample code, country of origin, body part, number of sequences.
- **SEQUENCE** - sequence code, region of the genome, size, DNA sequence.

The relationship types were also identified, together with the cardinality ratios, constraint on participation and attributes:

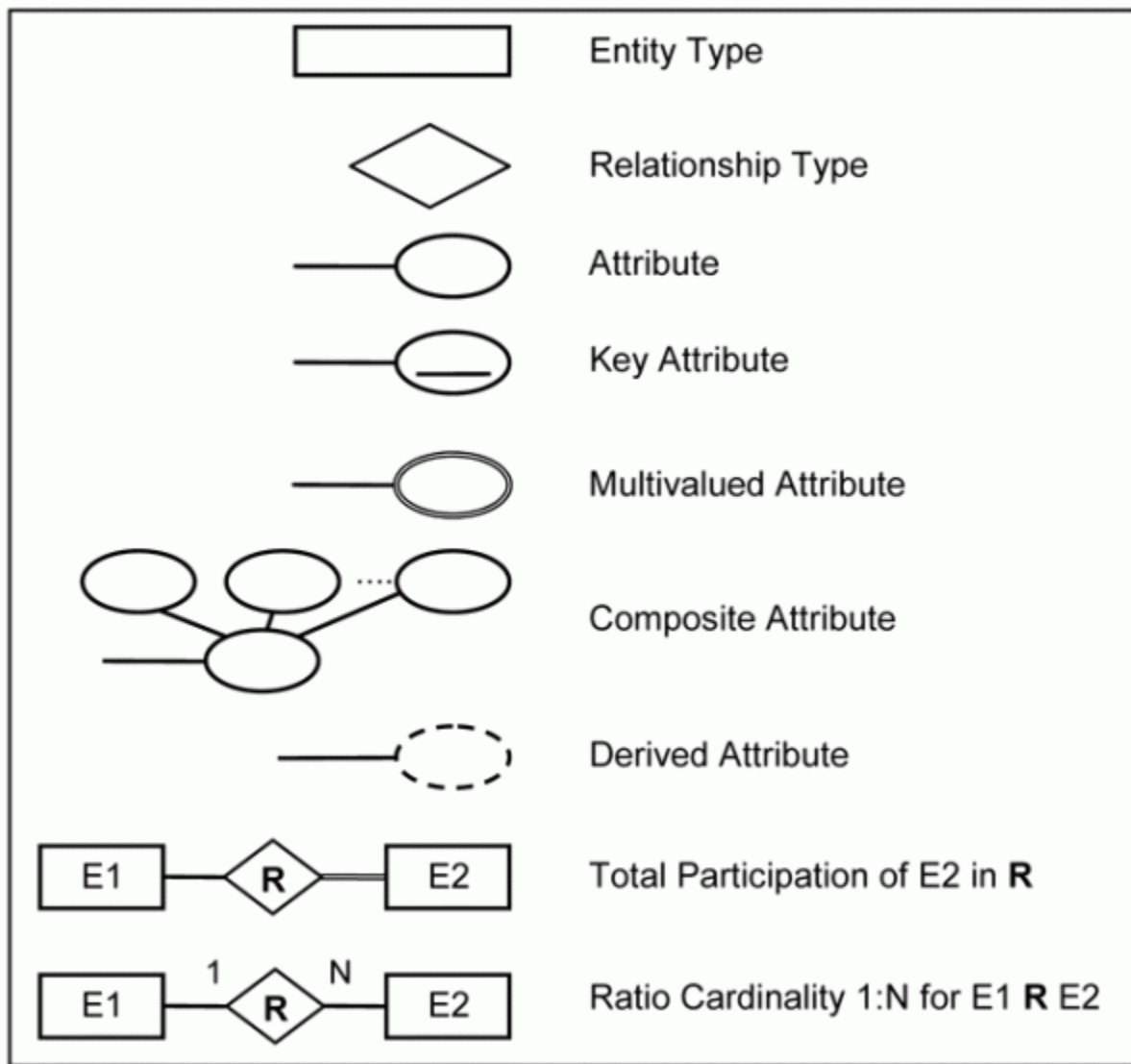
- PATIENT **HAS** SAMPLE:
  - Cardinality Ratio (**1:N**), as:
    - 1 Patient can have N Samples.
    - 1 Sample belongs to **1** Patient.
  - Constraint on Participation:
    - For the Patient: **Partial**, as the patient is not obliged to have a sample in order to be registered in the database.
    - For the Sample: **Total**, as a sample necessarily must be associated to a Patient in order to be registered in the database.
  - Attributes: collection date of the sample.
- SAMPLE **GENERATES** SEQUENCE
  - Cardinality Ratio (**1:N**), as:
    - 1 Sample can have N Sequences.
    - 1 Sequence belongs to **1** Sample.
  - Constraint on Participation:
    - For the Sample: **Partial**, as a Sample can be registered although there may be no Sequence to associate it with.
    - For the Sequence: **Total**, as there must be an existing Sample in the database from which the sequence have been generated.
  - Attributes: none.

## Entity-Relationship Diagram (ERD)

The Entity-Relationship Diagram (ERD) represents the Entity-Relationship model of a database. It is one of the most common forms of documenting a database's structure. The ability to read ERDs can help biologists understand the information and relations that are stored in any database and is also the key to understanding which are the querying possibilities of that database. The previous figure illustrates an ERD for the clinical database scheme. The entity types Patient, Sample and Sequence are depicted inside rectangles. Relationship types REALIZE, GENERATE are shown in diamonds linked to the participating entity types. Attributes are shown in ovals connected to the entity types and relationship types. The key-attributes are underlined. Derived attributes are depicted in ovals with dotted lines.

Also, the figure shows the cardinality ratios for each relationship type. The cardinality ratio of Patient **HAS** Sample is 1:N and of Sample **GENERATES** Sequence is 1:N. The partial constraints on participation are specified by simple lines. The parallel lines denote total participation (existential dependency).

Finally, this section ends with the next figure, which shows a notational graph summary of Entity-Relationship Models. Further details can be found in Elmasri and Navathe 2004.



In this section, the basic concepts and notations of the ERM were introduced taking into account a specific application, clinical databases, for an easier illustration of the concepts and also to show that an application's conceptual model depends on the rules and constraints of the miniworld, as established by specialists in each respective area.

When designing a database, the next phase is the logical design of the database, which makes use of the application's conceptual model in order to build the application's logical model. The logical data model selected in this case is the Relational Model, which will be explained in section 4. Next, we will explain how to query a database based on an ER diagram, since this is the ultimate goal of a database, and assuming that most readers will not be involved in designing databases, but rather in exploring the information stored in them.

### 3. Querying in the Relational data Model

Using the ER diagram presented above a physical database was created to illustrate the use of queries. Details of this database creation is shown in Section 4. In the current section we will explore the benefits and flexibility of queries on a database that was carefully designed. We will use an actual database populated according to the values depicted in the three tables below:

**PATIENT.**

<u>Code</u>	Gender	Birthday	City	Country
0001	M	1980-01-21	São Paulo	Brazil
1000	F	1990-08-09	Manaus	Brazil
9876	M	1975-04-30	Cuiaba	Brazil
0101	F	1960-02-02	São Paulo	Brazil

**SAMPLE.**

<u>Code</u>	Source	BodyOrgan	DateOfCollect	PatientFk
0001	Brazil	Null	2006-01-01	0001
0011	Brazil	Null	2006-01-02	1000
0015	Angola	Null	2006-01-03	1000
1010	South Africa	Liver	2006-01-04	0001
1050	Mocambique	spleen	2006-01-05	9876
2069	Brazil	head	2006-01-06	0101
9876	USA	lung	2006-01-07	9876

**SEQUENCE.**

<u>Code</u>	GenomicRegion	Size	DNASequences	SampleFk
1111	Null	535	'AGCCACCAGCGCAACATGA...CTCAGCTGAAGAAG'	0011
2222	ENV	1035	'TGC GTTACTTTAAATTGTA...GAGACCTTCAGACCA'	0015
3333	POT	1018	'GGCGAACGGGTGAGTAAC...TTATCGCTAGTTACC'	1010
4444	ENV	1044	'TGC GTTACTTTAAATTGTA...AGACCTTCAGACCA'	0015
5555	Null	1193	'TACTTCAACTACGAGAACC...ATAAAAATTACTTTG'	0015

We will begin by studying the simplest type of query. In this type of query, all you have to verify is whether the tables were filled out correctly in the relational database. The SQL language has a basic command to recover information in a relational database: the SELECT command. In general, the results of registration queries in SQL are tables.

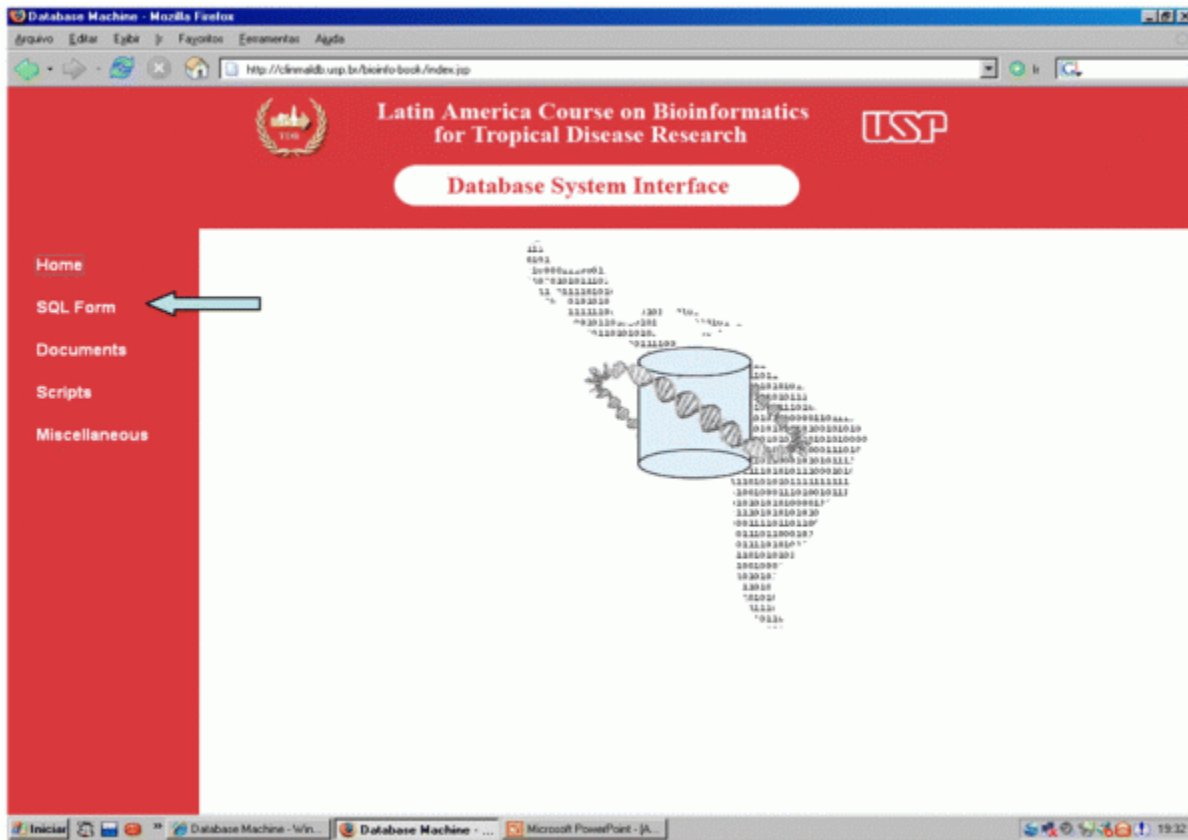
The basic type of the SELECT command is made up of three clauses: SELECT, FROM and WHERE:

**SELECT** <list of attributes>

**FROM** <list of tables>

**WHERE** <condition>

The condition is an expression that filters the data that will be retrieved by the query. To illustrate the use of the SELECT command, we will make use of the database of clinical data again, assuming that it has been populated using the database's Internet page (available at <http://clinmaldb.usp.br/bioinfo-book>):



You should access the link "SQL Form", which will give you the query page, where you can run all queries described below through copy and paste mechanisms for each SQL expression and execute them using the "Execute" button. The first query is a command to show the existing entries (see section 4) in the relation PATIENT. It is described using the SQL expression<sup>2</sup>:

Query 1: **SELECT \* FROM PATIENT**

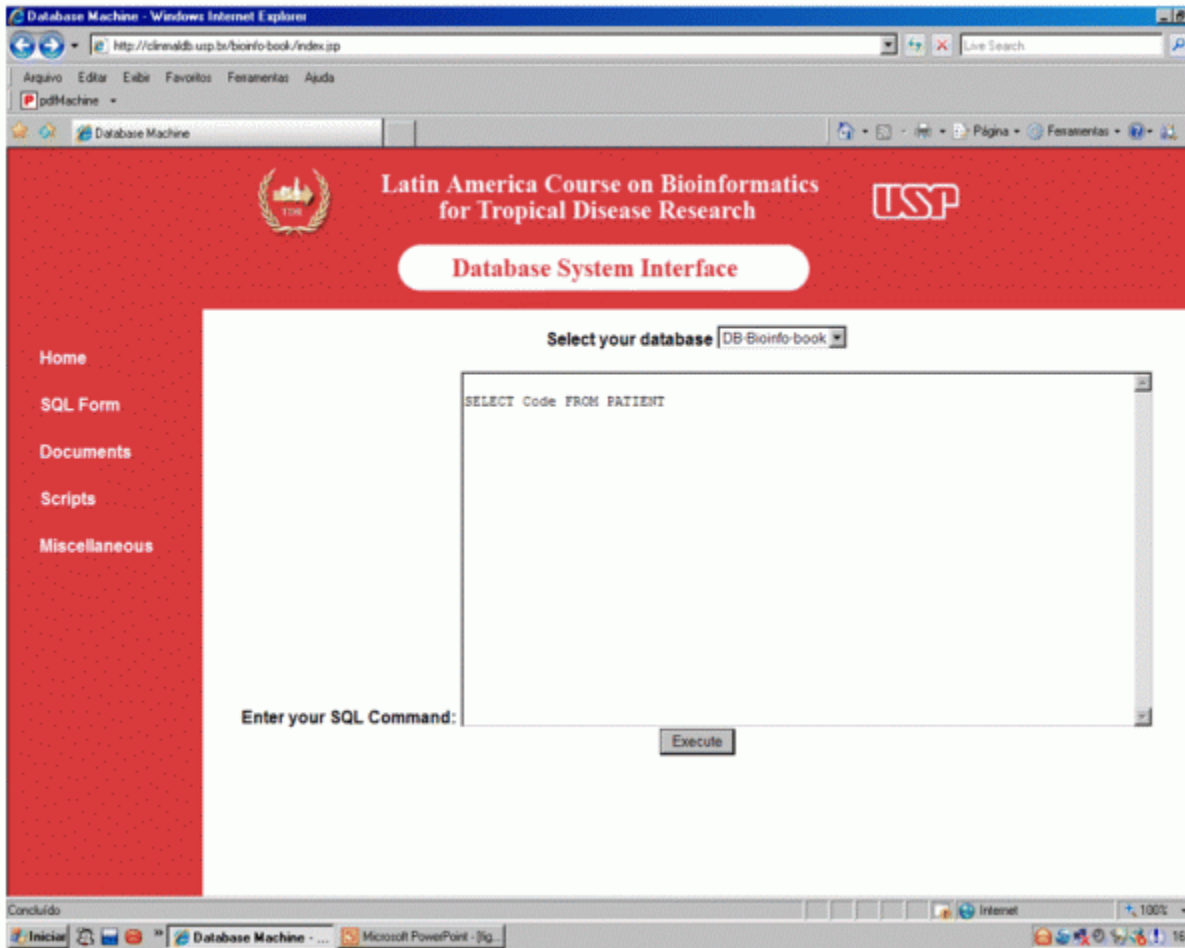
Executing this command in SQL Form Internet environment, it should return all the values of the relation PATIENT which were above. In case you are interested in getting only the values of the attribute PatientCode in the relation PATIENT, click the "New query" link at the bottom of the results and issue the query:

Query 2:

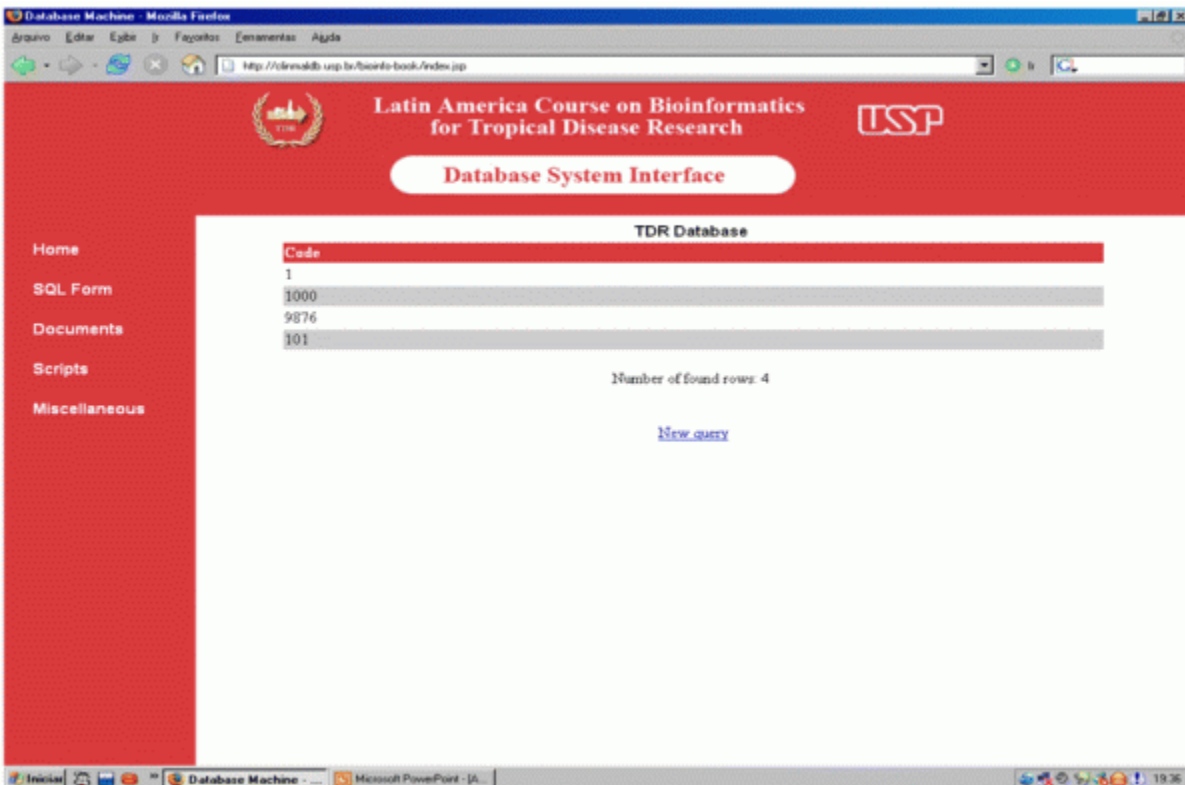
**SELECT Code FROM PATIENT**

To execute this command, type the text above in the SQL form page of the database:.

<sup>2</sup> The asterisk indicates that the SELECT command should return the values of all the attributes of the relation PATIENT.



After typing the text, click on the "Execute" button and you should get the response as shown below:





In the examples given previously, we did not employ the clause WHERE. This clause is used to filter the information in a query. The conditions specified in a WHERE clause can be a disjunction or a conjunction of comparisons of attribute values. For example: imagine you are interested in obtaining the list of code numbers for patients of the female gender. This query would be formulated as:

Query 3:

```
SELECT Code FROM PATIENT WHERE Gender = 'M'
```

We can have more complex conditions after the WHERE clause. By using a conjunction (AND) of comparisons we can issue the following queries:

Query 4: Find from all patients, those that are female and whose birthday is before 1980-01-01.

```
SELECT Code  
FROM PATIENT  
WHERE Gender = 'M' and Birthday < '1980-01-01'
```

Query 5: Find all DNA sequence with GenomicRegion = Pol and size above 200 bp.

```
SELECT DNASequences  
FROM SEQUENCE  
WHERE Size > 200 and GenomicRegion = 'POL'
```

You can use the internet environment described above to issue each of these queries and compare the results with the previous one and with the database contents.

Queries do not need to be restricted to just one table. The JOIN operation is used for combining information of two or more tables. A "Join" is built using a special clause condition in WHERE. If we want to join the information belonging to PATIENT and SAMPLE tables, we need to define a join condition. Normally this is done through key and foreign key connections. In this case the FROM clause will mention two tables and the attributes will refer to which table they belong:

Query 6: Find the country of patient whose sample code is 15.

```
SELECT PATIENT.Country  
FROM PATIENT, SAMPLE  
WHERE SAMPLE.Code = 15 and  
PATIENT.Code = SAMPLE.PatientFK
```

Another example, this time using two joins conditions. One for PATIENT and SAMPLE, and other for SAMPLE and SEQUENCE:

Query 7: Find the Gender of Patient, Source of Sample and GenomicRegion of Sequence whose size is longer than 800 bp.

```
SELECT Gender, Country, GenomicRegion
```

```

FROM PATIENT, SAMPLE, SEQUENCE
WHERE SEQUENCE.Size > 800
and
PATIENT.Code = SAMPLE.PatientFK
and
SAMPLE.Code = SEQUENCE.SampleFK

```

Finally we will show how to obtain the results in a user-specified order. The order of the results can either be numerical or alphabetical. In SQL, this is possible using the operator ORDER BY.

Query 8: Find the city and birthday of Patients that have samples collected in 2006-01-03, and show it ordered by Patient's birthday.

```

SELECT PATIENT.City, PATIENT.Birthday
FROM PATIENT, SAMPLE
WHERE SAMPLE.DateOfCollect = '2006-01-03' and
PATIENT.Code = SAMPLE.PatientFK
ORDER BY PATIENT.Birthday

```

For a better understanding of the SQL language, re-execute these basic queries and try other queries using PATIENT, SAMPLE and SEQUENCE in the SQL Form environment.

## 4. Relational Data Model

As we have seen above, SQL queries can be a powerful instrument for accessing the information contained in a database. SQL queries can only be used if we implement a Relational Database. To do this, we need to build the *Relational Data Model*. **Relational Data Model** is a logical data model used to develop logical database designs<sup>3</sup>.

The Relational Model represents data from a database as *relations*. The word *relation* is used to mean "a list or group of information" and not in the sense of "association or relationship".

Every relation can be understood as a table or as a simple registration file. However, keep in mind the following observation: during the logical design phase of the database, we recommend that you employ the word *relation* instead of *table*, since the word *table* is normally employed in the context of a specific Relational DBMS in the physical design phase.

An example illustrated in the next figure shows an extension of the relation **PATIENT**, with its **attributes** and **attribute values**.

---

<sup>3</sup> The Database Management Systems (DBMS) that use the Relational Model are called Relational DBMS's.

The diagram shows a table with five columns: Code, Gender, Birthday, City, and Country. The rows contain data for five different records. Annotations include:
 

- Attribute:** An arrow points to the 'Birthday' column header.
- Tuple:** An arrow points to the entire row containing the record with Code '0003'.
- Value:** An arrow points to the specific value '02/02/1980' in the Birthday column of the row with Code '2040'.

Code	Gender	Birthday	City	Country
0001	M	01/01/1970	São Paulo	Brasil
0003	F	03/03/2003	Manaus	Brasil
1010	M	02/02/2002	Rio de Janeiro	Brasil
2040	M	02/02/1980	Angola	Angola

Each line of the relation is called **tuple** and each column is called an **attribute**. The **type of data** of the **values** that an **attribute** can assume is termed the **domain** of the attribute.

- It's important to emphasize that the Relational Model always takes into account that the attributes' values are indivisible, that is, they're **atomic**.

## The Specification of a database using SQL

The Relational Data Model is also specified using the Data Definition Language (DDL), part of the SQL standard. In addition, as we will see below, we can use the SQL language to specify a series of integrity constraints on the database that will be maintained automatically by the DBMS.

Unfortunately, depending on the DBMS, the SQL language provided can be a mixture of ANSI SQL89, SQL92 and SQL99 standards. Furthermore, some developers of DBMS'S provide extensions that do not use the SQL language standard (such as an AUTOINCREMENT clause that does not exist in any standard ANSI SQL), which may make the portability of applications that use these extensions more difficult. For this reason, it is important for the designer to know how to evaluate the costs and the benefits of using non-standard SQL constructions in any given application in a database.

Before we start, we should talk about the concepts of *primary key* and *foreign key*. The primary key of a relational table uniquely identifies each record in the table. It can be a normal attribute that is guaranteed to be unique (such as Social Security Number in a table with no more than one record per person), A foreign key is an attribute that points to the primary key of another table. The purpose of the foreign key is to ensure referential integrity of the data. In other words, only values that are supposed to appear in the database are permitted.

Let us consider the logical scheme of the relational database of clinical data, presented in the next box. For example: the Patient Code (marked with FK) of the relation SAMPLE is the foreign key that corresponds to the primary key Code in the relation PATIENT.

```

PATIENT(Code, Gender, Birthday, City, Country)
SAMPLE (Code, Source, BodyRegion, AmountOfSequences,
DateOfCollect, PatientFk)
SEQUENCE (Code, GenomicRegion, Size, DNASquence, SampleFk)
  
```

Integrity constraints can be defined at the moment the relations in the relational database are created. For example: below, we can see the definition of the integrity constraints during the creation of the relation PATIENT by using the command: CREATE TABLE in the SQL language.

The command indicates the creation of the table PATIENT instead of the relation PATIENT. The SQL language is used in the database's physical design with the purpose of materializing relations in physical tables.

```
CREATE TABLE PATIENT (  
    Code          VARCHAR(04) NOT NULL,  
    GENDER        CHAR(1),  
    Birthday      DATE,  
    City          VARCHAR(10),  
    Country       VARCHAR (02),  
    CONSTRAINT PatientPK  
        PRIMARY KEY (Code)  
);
```

The command CREATE TABLE initially specifies the name of the relation and its set of attributes. Each attribute has a name, a type that specifies the domain of valid values and may have some attribute constraint such as NOT NULL.

In the SQL code above, the *entity integrity constraint* was ensured by indicating NOT NULL to the Code attribute. The *key integrity constraint* was ensured by the following clause: **CONSTRAINT PatientPK PRIMARY KEY (Code)**.

The clause, called PatientPK, indicates that the attribute *Code* is a primary key; therefore, the key's integrity constraint will be ensured by the DBMS

The referential integrity constraint is ensured by the clause FOREIGN KEY. For example: look at the code below, which shows the creation of the relation SAMPLE.

```
CREATE TABLE SAMPLE (  
    Code VARCHAR(20) NOT NULL,  
    Source VARCHAR(30),  
    BodyOrgan VARCHAR(30),  
    CollectionDate DATE,  
    PatientFk VARCHAR(04),  
    CONSTRAINT pk_sample PRIMARY KEY (Code),  
    CONSTRAINT fk_patient FOREIGN KEY (PatientFk)  
    REFERENCES PATIENT(Code)  
    ON DELETE CASCADE  
    ON UPDATE CASCADE  
);
```

The `fk_patient` constraint indicates that the attribute `PatientFk` is a foreign key that makes reference to the primary key `PATIENT(Code)` in the relation `PATIENT`. Moreover, the clause `ON DELETE CASCADE`, indicates that if a given Patient is removed from the database, the Sample tuples will be removed. The clause `ON UPDATE CASCADE` indicates that, when the attribute `PatientCode` of a Patient is changed, the attributes that made reference to the given patient will also be changes accordingly.

Similarly, we can define the relation for the sequence data in the following way:

```

CREATE TABLE SEQUENCE (
    Code    VARCHAR(20)    NOT NULL,
    GenomicRegion VARCHAR(30),
    Size     VARCHAR(30),
    DNASequence VARCHAR(4000),
    SampleFk   VARCHAR(20),
    CONSTRAINT pk_sequence
        PRIMARY KEY (CodigoSequencia),
    CONSTRAINT fk_sequence
        FOREIGN KEY (SampleFk) REFERENCES SAMPLE(Code)
        ON DELETE CASCADE
        ON UPDATE CASCADE
)

```

It's interesting to verify that the constraints imposed upon the relations work appropriately. In order to do this, we will present some commands in the SQL language that will allow you to make insertions and removals of tuples in a relation.

The box below shows the insertion of a new Patient by using the command INSERT. Notice that all the attribute values were filled out in the sequence in which the attributes were defined during the creation of the relation PATIENT. In this way, the first value, '0001', corresponds to the attribute PatientCode, the second value, 'M', corresponds to the attribute Gender, up until the last value, 'Brazil', which corresponds to the attribute value Country.

```

INSERT INTO PATIENT
VALUES      ('0001', 'M', '1980-01-21', 'Sao Paulo',

```

Many times, you don't have to fill out all the attribute values in a tuple. In these cases, the command INSERT allows you to specify the attributes whose values will be filled out.

There are variations in the INSERT command that can be seen in Elmasri and Navathe 2004. With these variations of the INSERT command, you can see that the DBMS guarantees the three integrity restrictions reviewed in this unity (key, entity and referential). Details regarding the creation and insert scripts of relations can be found menu option Scripts which is available on the web at <http://clinmaldb.usp.br/bionfo-book>.

## 5. Final Considerations

This Chapter was an introduction to the main aspects of the conceptual, logical and physical designs of a database. Section 1 was an introduction to the historical background of how Database Management Systems were developed. In Section 2, we discussed the ERM model for conceiving the conceptual data design. In Section 3, we showed examples of possible queries and in Section 4 we explained the **Relational Model**, which it is supported by Relational Database Manage Systems widely found in the market.

In this chapter, the process of mapping the conceptual scheme to the relational scheme was introduced. A simplified process was applied to generate the relational data scheme of the clinical database example. The SQL queries used were very basic constructions.

To construct more complex databases and queries, one needs to be familiar with Relational Algebra and Relational Calculus, and also requires a deeper knowledge of SQL. It is important to emphasize that this chapter is only an introduction to database design and the SQL languages. For a more in-depth understanding the user is referred to the *Further Readings* section below.

## Further readings

1. Elmasri, R.; Navathe, S. B. Fundamentals of Database Systems. Pearson (Addison Wesley), 4<sup>th</sup> ed., 2004.
2. Korth, H.; Silberschatz. Database Systems. Third Edition. Makron Books, 1998.
3. Raghu Ramakrishnan e Johannes Gehrke, Database Management Systems, Second Edition, McGraw-Hill, 2000.
4. Teorey, T.J., Lighthouse S., Nadeau, T. Database Modeling and Design, 4<sup>th</sup>. ed., Morgan Kaufmann Publishers, Inc, San Francisco, 2006.
5. Beaulieu A., Learning SQL ,O'Reilly Media, Inc., 2005.
6. Molinaro A., SQL Cookbook, O'Reilly Media, Inc., 2005
7. Taylor A. G., SQL For Dummies, For Dummies Publisher; 6<sup>th</sup> ed., 2006





## Chapter A05. Similarity Search

João Carlos Setubal, PhD<sup>1</sup> and Ruediger Braeuning, PhD<sup>2</sup>

Created: February 10, 2006; Updated: March 9, 2007.

João Carlos Setubal has authored the theoretical part of this chapter and Ruediger Braeuning has authored the tutorial.

### Why Sequence Comparison Is Important

Consider the following protein sequence:

```
GDVEKGKKIFIMKCSQCHTVEKGGKHKHTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE
```

It is the sequence of a small protein found in **humans**, called cytochrome *c*. Its function is to carry electrons in the cell. The full description from the SwissProt database (<http://us.expasy.org>, also mirrored at other sites) is:

Electron carrier protein. The oxidized form of the cytochrome *c* heme group can accept an electron from the heme group of the cytochrome *c*1 subunit of cytochrome reductase. Cytochrome *c* then transfers this electron to the cytochrome oxidase complex, the final protein carrier in the mitochondrial electron-transport chain.

This description, although irrelevant in what follows, serves the useful purpose to remind us that proteins are three-dimensional structures that perform specific biochemical roles in a cell. It is easy to forget this fundamental fact when dealing with the strings of letters that represent proteins (or DNA), as I do in this entire chapter. Now consider the following sequence:

```
GDVEKGKKIFVQKCAQCHTVEKGGKHKHTGPNLHGLFGRKTGQAAGFSYTDANKNKGITWG
EDTLMEYLENPKKYIPGTKMIFAGIKKKGERADLIAYLKKATNE
```

It is the sequence of a protein found in **mice**, whose function is the same as that of the human protein, and therefore is also known as cytochrome *c*. A cursory look at these two sequences suggests that they are very similar. In fact, one could play the children's game "spot the differences" and, with some patience, visually determine all differences that exist between these two sequences. For example, the first difference is in the 11th residue: in the human sequence, we find an I, whereas in the mouse sequence there is a V.

One way to be more systematic about finding the differences is to *align* these two sequences. This means placing one on top of the other, so that it is easier to tell which positions are the same and which positions are different. Here is this alignment, with the human sequence on top and the mouse sequence on the bottom:

```

H: GDVEKGKKIFIMKCSQCHTVEKGGKHKHTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
M: GDVEKGKKIFVQKCAQCHTVEKGGKHKHTGPNLHGLFGRKTGQAAGFSYTDANKNKGITWG
*****  ** ***** ***** ***** ***** ***** ***** **

H: EDLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKATNE
M: EDLMEYLENPKKYIPGTKMIFAGIKKKGERADLIAYLKATNE
***** ***** ***** ***** ***** ***** *****

```

Notice that the alignment had to be broken into two sections, because the sequences are too long to fit in a single line. Notice also that there are asterisks below the second sequence to show positions where there is a match between a residue (or character) in the first sequence and a character in the second sequence. Thus, we can immediately see that there are exactly nine positions where these two sequences differ, of a total of 104 positions. This gives us our first *measure of similarity* between these two sequences: they are 91% (95/104) identical.

Intuition suggests that this is a high level of similarity, and indeed it is. It is no coincidence that two proteins, although from two different species, have the same function with highly similar sequences. Humans and mice are mammals, and they share an ancestor that probably existed about 80 million years ago. It is likely that this ancestor had a cytochrome *c* protein, and the sequences above are descendants of that primitive sequence, just as each species is as a whole (but please note the caveats in the next section). Many of the protein sequences in mice and humans are similar for this reason, and this gives us a powerful tool to find clues about protein function. In the case of cytochrome *c*, independent bench experiments determined the function of each of the two proteins. For many other newly discovered proteins, however, we can formulate hypotheses for their function based on similarity with known proteins. This is a primary motivation for comparing sequences.

If sequence similarities were restricted to closely related species, such as humans and mice, sequence comparison would be of limited use. Fortunately, this is not the case, as can be shown by comparing the human cytochrome *c* sequence to cytochrome *c* sequences from other organisms.

Let us start with *Drosophila melanogaster*, the **fruit fly**. Here is its sequence:

```

GVPAGDVEKGKKLFFVQRCQAQCHTVEAGGKHKVGPNLHGLIGRKTGQAAGFAYTDANKAKG
ITWNEDTLFEYLENPKKYIPGTKMIFAGLKKPNERGDLIAYLKSATK

```

This time it is not so straightforward to align this sequence to the human sequence. The first problem is that this sequence has 107 characters, instead of 104. So it is not immediately clear how to align one on top of the other, as we did before. But again, some visual inspection suggests that the sequences are very similar. For example, we could notice that both sequences have the strings GDVEKGKK at or near the beginning.

On the basis of this observation, we could create the following alignment:

```

H: GDVEKGKKIFIMKCSQCHTVEKGGKHKHTGPNLHGLFGRKTGQAPGYSYTAANKNKG
D: GVPAGDVEKGKKLFFVQRCQAQCHTVEAGGKHKVGPNLHGLIGRKTGQAAGFAYTDANKAKG
***** * * ***** ***** ***** ***** ***** ***** **

H: I IWGEDLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKATNE
D: ITWNEDTLFEYLENPKKYIPGTKMIFAGLKKPNERGDLIAYLKSATK
* * ***** ***** ***** ***** ***** ***** *****

```

We have obtained an alignment where there are 80 identical positions. Again, this level of similarity is very high. If we did not know what the function of the *D. melanogaster* protein was, we would be able to hypothesize that it has the same function as the one from human, although humans and flies are very different organisms.

Now let us see what we can do with the cytochrome *c* sequence from *Caenorhabditis elegans*, a worm and well-studied model organism. Here is its sequence:

```
SDI PAGDYEKGGKVKYQRCLQCHVVDSTATKTGPTLHGVI GRTSGTVSGFDYSAANKNKG
VVWTKETLFEYLLNPKKYI PGTKMVFAGLKKADERADLIKYIEVESAKSL
```

This sequence has 107 characters, so we have again the problem of differing lengths. Our method of visual inspection may suggest local similarities, but to obtain a good alignment, it will not suffice to simply shift one sequence with respect to the other as we did with the human and fly comparison. Here is what happens if we do this (we show only the first part of the alignment):

```
H:      GDVEKGKKIFIMKCSQCHTVEKGGKHKGTGPNLHGLFGRKTGQAPGYSYTAANKNKG
C: SDI PAGDYEKGGKVKYQRCLQCHVVDSTATKTGPTLHGVI GRTSGTVSGFDYSAANKNKG
      ***** * * * * *
```

Notice that there are several matches in the beginning, but they then virtually disappear. However, visual inspection shows that there would be several matches in the second half of this alignment if only we could shift the *C. elegans* sequence by one position, in the middle, thus:

```
H:      GDVEKGKKIFIMKCSQCHTVEKGGKHKGTGPNLHGLFGRKTGQAPGYSYTAANKNKG
C: SDI PAGDYEKGGKVKYQRCLQCHVVDSTAT-KTGPTLHGVI GRTSGTVSGFDYSAANKNKG
      ***** * * * * * ***** ** * * * * * *****
```

Notice the introduction of a "space" character (represented by the hyphen) to accomplish the desired shift, aligning it with an H residue in the human sequence. The rest of the alignment becomes:

```
H:      I IWGEDTLMEYLENPKKYI PGTKMIFVGIKKKEERADLIAYLKKATNE
C:      VVWTKETLFEYLLNPKKYI PGTKMVFAGLKKADERADLIKYIEVESAKSL
      * * * * * ***** * * * * * ***** *
```

We have obtained an alignment with 59 matches, which is still very good, again showing the high degree of *conservation* of this particular protein sequence, even between organisms so different as humans and worms. However, from an alignment-construction point of view, two basic questions arise from this last example. **How can we be sure that the alignment shown is the best one?** Maybe there is another way of introducing spaces that will give us more than 59 matches. The second question is: **What method should be used to create[AG1] alignments?** Thus far, we have done it manually. If we had to deal with longer sequences, then doing a manual alignment and using a trial-and-error method would clearly be very tedious and prone to error. These two questions suggest that we need a rigorous way for defining good alignments (and in particular, the best alignment); and we need an *algorithm*, that is, a mathematical procedure that can be implemented in a computer program, to find the best alignment. This motivates the sections that follow.

## Caveats Regarding Similar Sequences, Evolutionary History, and Gene Function

In the previous section, we made the statement that highly similar protein sequences are probably derived from a common ancestor sequence, and that they also probably have the same molecular function. Before we plunge into technical details regarding sequence similarity, it is important to qualify these statements.

The keyword in this qualification is "probably". We are making inferences about past history, and therefore we can never be 100% sure. Indeed, there are known cases of similar sequences that do not share a common ancestor. This can happen due to chance (more on that in a later section) or due to a phenomenon called *convergent evolution* (whereby two sequences or characteristics evolve to become similar but follow different

paths, as is the case of wings in bats and birds). But it is an observed fact that the more two sequences are similar, the higher the probability that they are derived from a common ancestor.

Note also that one should be very careful about generalizing the evolutionary history of particular genes to that of the species as a whole to which the genes belong. The cytochrome *c* example of the last section suggests that the evolutionary history of this protein reflects the evolutionary history of the different species mentioned. This is probably true for cytochrome *c*, but it is by no means always the case with other proteins. It is known that the genome of all organisms sometimes incorporates DNA from other organisms, a phenomenon called *lateral* (or *horizontal*) *transfer*, as opposed to the more traditional mode of *vertical* descent, from parent to offspring. When such an event happens, and the foreign DNA contains genes (as it usually does), the gene in question will have a very different evolutionary history compared with the genes that have been vertically transmitted.

Finally, note that although sequence similarity is usually a good indication of the same protein function, the converse is not true. There are plenty of examples of proteins with the same function that share no significant similarity, even at the amino acid level.

## What Is Sequence Similarity?

We now address how to define a measure of similarity between sequences. In fact, we have already used one such measure without defining it: *the percent identity* similarity measure. In this measure, similarity between two sequences is defined as the maximum number of matches that can be found in an alignment between the sequences divided by the alignment length. The notion of *maximum number of matches* implies that we need the *best alignment* between the sequences; in other words, the best alignment is the one that yields the maximum number of matches.

When we compared the human and mouse sequences above, we obtained the value 91% for their similarity under this measure. How can we be sure that it is correct (i.e., the maximum number of matches)? That alignment was so simple to create that a manual inspection should be enough to convince us that it is the best possible alignment. For more complicated alignments, we will need an algorithm to find the best alignment, but we will defer the topic of algorithms for the next section. Assuming that the other alignments shown in the previous section do yield the maximum possible number of matches, we would obtain the following similarity values: human *vs.* fly:  $80/108 = 74\%$ ; human *vs.* worm:  $59/111 = 53\%$ . Notice that the denominator changes in all three cases, because each alignment has its own length. Note also that this simple similarity measure can in principle be used to quantify the *evolutionary distance* that separates humans from mice, flies, and worms. Evolutionary analysis based on sequence comparison is a fascinating topic, but it is beyond the scope of this chapter.

Percent similarity is a reasonable measure of similarity, but there is another, more sophisticated one that is more widely used. It takes into account the fact that when aligning sequences, we very often to use spaces, as was the case with the human *vs.* worm comparison of the last section. The use of spaces introduces a third case of single-character alignment. The first two are the *match* and the *mismatch*. The third one is when a character is matched to a space. It should be intuitive that an alignment between a character and a space is an event different in kind from an alignment between two different characters (a mismatch). A mismatch implies a *substitution*, that is, one residue was replaced by another one over the course of evolution (or perhaps *both* were replaced). A space implies either a *deletion* in the sequence that needs it in the alignment or an *insertion* in the other sequence. Evolutionarily speaking, insertions/deletions are events different from substitutions and, therefore, should be considered in different ways when evaluating alignments.

In the percent identity similarity measure, we simply counted the number of matches. This is equivalent to saying that each match counts for +1 and each mismatch or space counts for 0. If we want to treat mismatches and insertions/deletions (also known as *indels*) differently, then we need different numbers for them. It is known

that indels are rarer events than substitutions. Therefore we will assign  $-1$  to mismatches and  $-2$  to indels (we use negative values to stress the fact that these are *penalties*, which contribute to decrease the total similarity score).

What we have just created is a *scoring system* for alignments. Whenever two sequences are compared and their similarity is computed, there has to be an underlying scoring system. The scoring system just described is very simple, but it has been shown to be good enough for most nucleotide (or DNA) alignments. Let us see a simple example:

GATCTCA-GTAATA
GA-CTAATGTA-TA

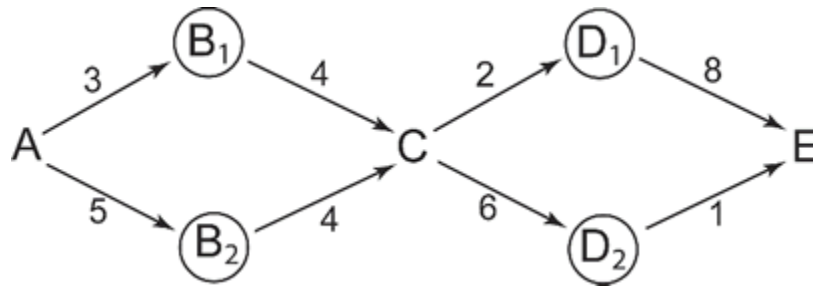
In this alignment, we are using vertical bars to denote matches. Notice that there are three indels and one mismatch. Using the scoring system defined above, we obtain the value  $1 + 1 - 2 + 1 + 1 - 1 + 1 - 2 + 1 + 1 + 1 - 2 + 1 + 1 = 3$ . What is this value? If this is the maximum possible score for any alignment between these two sequences, we will say that it is their *similarity* under the new measure. Finding this kind of best alignment requires an algorithm, just as it does for percent identity.

The measure of similarity just described is so widely used that it does not have a special name, as percent identity does. It is simply referred to as *the similarity* in many textbooks. However, a cautionary remark has to be given here concerning the word *homology*. In many scientific papers and published literature in general, it is common to say that two sequences have such-and-such *percent homology*, or that one sequence "has homology" to another. In most cases, the authors in fact are making statements about the *similarity* of the sequences being compared, and not about their homology. Two sequences are homologous when they share a common ancestor, that is, they are phylogenetically related. Two highly similar sequences may indeed be homologous (as is the case of the human and mouse cytochrome *c* sequences seen before), and high similarity is a strong indication of homology. However, one should never lose sight of the fact that the numerical quantity associated with the comparison is similarity (under various guises), and that homology is a hypothesis about their shared evolutionary history. The term *homology search*, frequently used when doing sequence comparison in sequence databases, can, in light of this explanation, be understood as a search for similar sequences that, if similar enough, will likely be homologous.

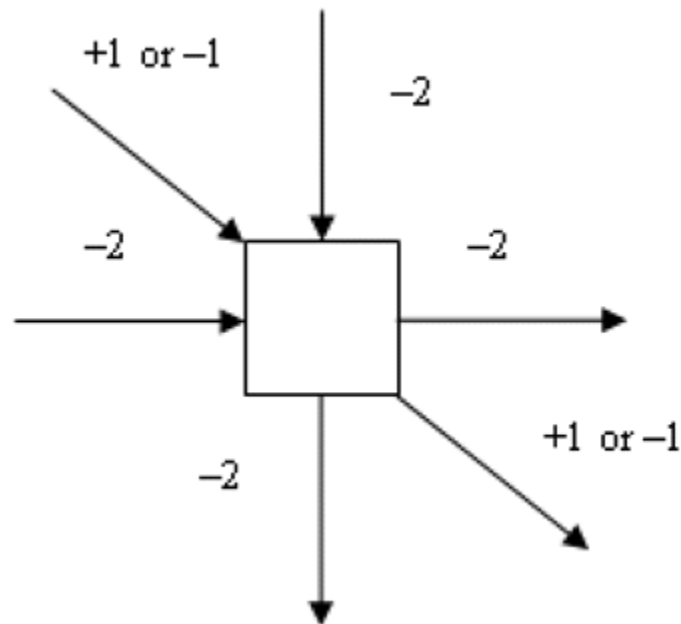
Another important comment here is about "degree of similarity". We have said that the human and mouse sequences are highly similar. But what does "high" mean? What number has to be achieved to be considered high? In the case of percent identity, we have an intuition (90% seems high and 10% seems low), but in the case of the value 3 obtained in the nucleotide comparison shown above, we are more or less at a loss to state whether it is high or low. This is an extremely important question in the study of sequence comparison, and as we shall see, its answer lies in the statistical significance of the value obtained. But before we go into that, we have to address the similarity computation question.

## Finding the Similarity Between Two Sequences by Dynamic Programming

We will present an overview of the algorithm that finds the best alignment between two sequences under the  $+1/-1/-2$  scoring scheme defined in the previous section. Recall that an algorithm is a mathematical procedure with well-defined steps that runs in  $[AG^2]$  a finite time and is guaranteed to yield a certain result (the similarity and the corresponding alignment, in our case). An algorithm is different from a *computer program*; a program is the embodiment (or *implementation*) of an algorithm in a particular programming language and to be run on particular computers.



**Figure 1.** General idea of how dynamic programming works. The problem is to find the shortest path from  $A$  to  $E$ . Numbers on arrows denote the distance between the two points linked. We initially find the shortest way from  $A$  to  $C$  (which is through  $B_1$  and is 7 units long). This information is stored. We then worry about the best way from  $C$  to  $E$ . When we find it ( $CD_2E$ ), the whole solution ( $AB_1CD_2E$ ) is given by using the previous intermediary result ( $AB_1C$ ).

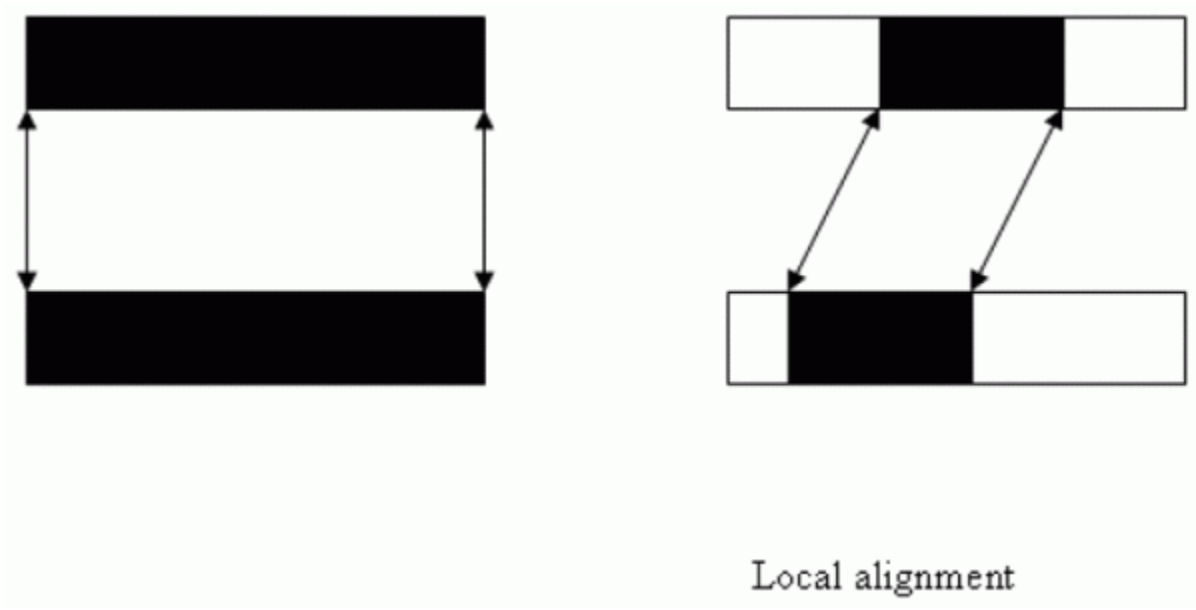


**Figure 2.** A generic cell of the dynamic programming matrix for sequence comparison.

The algorithm we will describe relies on a design technique called *dynamic programming* (DP). This is a historical name that has essentially nothing to do with the way the technique works. DP is used in many bioinformatics algorithms, and references to it abound in the published literature. So it is worth having a general idea of the concept.

DP works basically by constructing a table to store intermediary results that are used to obtain the final result. Here is the general idea, explained through a toy problem. The problem is to go from point  $A$  to point  $E$  using the shortest path. It is the case that point  $C$  lies between  $A$  and  $E$ . If we know the best (shortest) way to get from  $A$  to  $C$ , then what remains to be solved is how to go from  $C$  to  $E$ . In DP, we would store in a table the  $A$ - $C$  solution and then use it later to decide the best way to go from  $A$  to  $E$ . See Figure 1.

In the case of sequence alignment, DP works by trying to align one sequence with parts of the other sequence. These "intermediary" alignments correspond to intermediary "nodes" in problems similar to those illustrated in Figure 1. Let us explain through an example. Suppose we want to align sequence  $s = \text{GATC}$  to sequence  $t = \text{ATA}$ . In terms of what happens with the *last* column of the alignment, there are exactly three possibilities: either we have  $C$  aligned with a space, or  $C$  aligned with  $A$ , or  $A$  aligned with a space. There can be no other possibility.



**Figure 3.** Schematic illustration of global and local alignments. The boxes represent sequences, and the shaded parts are aligned.

Each of them will have its score  $(-2, -1, -2, \text{ respectively})$ . Which one we should choose will depend on the best possible intermediary alignment that each implies. Let us see what this means.

If we align C with a space, the score of the final alignment would be  $-2$  plus the score of the best alignment between GAT and ATA. If we align C with A, the score of the final alignment would be  $-1$  plus the score of the best alignment between GAT and AT. Finally, if we align A with a space, the score of the final alignment would be  $-2$  plus the score of the best alignment between GATC and AT. Because in DP we keep a table with all intermediary results, it is a simple matter to choose the best one: it is simply the maximum value among all three sums considered.

The overall situation is best visualized by a table:

	G	A	T	C	
0	-2	-4	-6	-8	
A	-2	-1	-1	-3	-5
T	-4	-3	-2	0	-2
A	-6	-5	-2	-2	-1

The **first row** contains sequence  $s$ , and the **first column** contains sequence  $t$ . The **second row** contains the values obtained assuming that *all* characters of  $s$  are aligned with spaces. There is an analogous situation for the **second column**. The other entries are the result of systematic computation, as was described above for the last character. The exact same reasoning applies to the alignment decision to be done for the  $i$ th character in  $s$  and the  $j$ th character in  $t$ . The similarity will be the number obtained for the bottom right cell in the matrix ( $-1$  in this case). It corresponds to the alignment:



The algorithm to obtain the similarity simply fills out this table in the systematic way described. The time it will take will be proportional to the size of the table (in terms of the number of cells), which is clearly the length of  $s$

plus 1 multiplied by the length of  $t$  plus 1. If  $s$  and  $t$  are of approximately the same length, then the algorithm can be said to run in *quadratic* time with respect to the lengths of the input sequences. This has important consequences, as we shall see later on.

There is a relationship between the shortest path toy problem that we presented above and the sequence comparison problem that goes deeper than the fact that both are problems that can be solved by DP. Each cell in the DP matrix can be thought of as connected to the neighboring cells by labeled "arrows", much in the same way that nodes in the shortest path problem are connected by labeled arrows. A generic matrix cell can be depicted as in Figure 2.

The numeric label on each arrow gives the "distance" from (or to) the neighboring cell. The numbers are exactly those of the scoring scheme that we have described. The diagonal arrows have two labels because the "distance" will depend on whether the two characters are the same (score +1) or different (score -1). In this framework, the filling of the table is equivalent to finding the *longest* path (as opposed to shortest, in the toy example above) from the top left cell to the bottom right cell. It is longest because similarity is a maximum value.

Filling out the table only gives the similarity (a number). To retrieve the actual alignment, we need another procedure. After the table has been completed and starting from the bottom right cell, all we need to do is to determine which previous cell was chosen in determining that cell's value, exactly as we described above. In the example, the value -1 in the bottom right cell was the result of  $0 - 1$ , where the 0 comes from the cell diagonally across. This means that we aligned C with A. To learn the next position of the alignment, we go to the cell containing 0 and ask the same question, and so on. This is called the *traceback*.

We will now give an overview of a more sophisticated algorithm, which is closer to the one actually used in practice when sequence comparison is done by dynamic programming. The difference with respect to the algorithm just described lies with the way a series of consecutive spaces (also called *gaps*) are treated. It has been observed that certain homologous proteins differ from each other not only by substitutions or by single indels but also by *blocks* of insertions and deletions. Consider the alignment:

H	S	G	I	A	G	-	-	-	R	V	G	G	T	L	S	A	A	S
H	S	S	I	I	G	L	P	Q	L	V	G	G	T	L	Q	P	A	S
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

This alignment has three consecutive spaces in the top sequence. It probably reflects a single deletion event in which all three corresponding residues were lost by the protein (alternatively, there could have been an insertion in the bottom sequence). If we were to charge for each space separately, this would mean treating each space as a *separate evolutionary event*. In general, we do not want to do that. We want to charge it as if it were a single event. One solution would be to charge just -2 for the whole gap. However, the longer the gap is, the less likely it is that it was indeed a single event. We want a formula that will enable us to charge more for longer gaps. One such formula that is widely used is:

$$p(k) = h + gk$$

where  $p(k)$  is the amount to be charged;  $k$  is the gap length (number of spaces); and  $h$  and  $g$  are constants, such as -2 and -1. Using these values, the formula means that for a gap with a single space, we will charge -3; for a gap with two spaces, we will charge -4, and so on. The constant  $h$  corresponds to the value we charge to *open a gap*; the value of  $g$  determines the cost based on the gap length. Notice how the formula yields a different result for a gap than what we would get if we were charging for each space separately. For  $k = 10$ , we would charge -20 if each space were separately charged; with the formula, the cost goes down to -12. Such a way of charging gaps is known as an *affine penalty function*.



We have to change the algorithm described above if we use an affine gap penalty function, and here is a brief overview about how it is done. Important details will be omitted, so do not expect a full understanding of the algorithm. As before, there is a table to store results; but instead of a single table, there are three separate tables. Each table corresponds to one of the three possible cases of alignment of a given column (character in  $s$  with space, or character in  $s$  with character in  $t$ , or character in  $t$  with space). The separation in three tables allows us to distinguish the case where we have a single space from the case where the space is simply the first in a gap. This distinction is crucial for the application of the affine formula.

Although the algorithm requires three tables instead of one, its running time remains quadratic in the length of the input sequences. In other words, there is no significant increase in running time because of the use of the affine gap penalty function as compared with the more simple way of charging spaces individually; this fact has accounts for the widespread use of DP with affine gap penalty functions.

The two algorithms described will perform a *global* alignment between two sequences. This means that all of sequence  $s$  has to be aligned with all of sequence  $t$ . In most cases, just a *local* alignment is desired. A local alignment is an alignment of a region of sequence  $s$  (usually called a *substring* or *subsequence*, although the two concepts are mathematically different) and a region of sequence  $t$ . Figure 3 illustrates schematically global and local alignments. In many cases, the best similarity score between two sequences is obtained by a local alignment instead of a global alignment; however, there are other cases in which the correct alignment to be obtained is global, even if its score may not be as good as the best local alignment between the same sequences. Yet another form of alignment is known as *semi-global*. In this case, we want to find the best alignment between a *prefix* of a sequence (the initial part) with the *suffix* (the final part) of another sequence. Such an alignment is needed in DNA fragment assembly and roughly corresponds to finding puzzle pieces (DNA fragments) that interlock. It is possible to easily modify the two algorithms described so that they will compute local or semi-global alignments; we do not describe such modifications here.

The dynamic programming algorithm with an affine gap penalty function was developed by Smith and Waterman (6) and is therefore known as the *Smith-Waterman algorithm*. Many efficient and open-source implementations of the Smith-Waterman algorithm can be obtained through the Internet.

## Amino Acid Scoring Systems

When comparing protein sequences, simple scoring schemes, such as +1 for a match, -1 for a mismatch, and -2 for a space, are not appropriate. To understand why, let us once again refer to the cytochrome  $c$  alignment between the human and worm sequences we have presented in the first section, which we copy here for convenience.

H:	GDVEKGKKI	FIMKCSQCHTVEKGGKHK	TGPNLHGLFGRKTGQAPGYSYTAANKNKG	
C:	SDI	PAGDYEKGGKVYKQRCLQCHVVDSTAT	-KTGPTLHGVI	GRTSGTVSGFDYSAANKNKG
	*****	* * * * *	* * * * * * * * * * * * * * *	

Let us look at the mismatches that appear after the first string of matches. In the first mismatch, we have I (isoleucine) aligned with V (valine). The biochemical properties and sizes of I and V are similar (both are hydrophobic and aliphatic). The amino acids in the next mismatched pair, F (phenylalanine) and Y (tyrosine), are also similar in size and properties. For the following pair, I and K (lysine), this is no longer true. These two have different properties and sizes. It would be advantageous to have a scoring system that reflects these similarities and differences; the reason lies with the way the evolution of biological molecules proceeds. If a nucleotide mutation in a codon causes change in the corresponding amino acid, that mutation may or may not persist, depending on the relationship between the original amino acid and the new one. For instance, mutations that result in an amino acid of size similar to the original one, or with similar biochemical properties, are more likely to succeed (meaning that the mutations still yield a sequence that is successfully translated into a working

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	9	-1	-1	-3	0	-3	-3	-3	-4	-3	-3	-3	-3	-1	-1	-1	-1	-2	-2	-2
S	-1	4	1	-1	1	0	1	0	0	0	-1	-1	0	-1	-2	-2	-2	-2	-2	-3
T	-1	1	4	1	-1	1	0	1	0	0	0	-1	0	-1	-2	-2	-2	-2	-2	-3
P	-3	-1	1	7	-1	-2	-1	-1	-1	-1	-2	-2	-1	-2	-3	-3	-2	-4	-3	-4
A	0	1	-1	-1	4	0	-1	-2	-1	-1	-2	-1	-1	-1	-1	-1	-2	-2	-2	-3
G	-3	0	1	-2	0	6	-2	-1	-2	-2	-2	-2	-2	-3	-4	-4	0	-3	-3	-2
N	-3	1	0	-2	-2	0	6	1	0	0	-1	0	0	-2	-3	-3	-3	-3	-2	-4
D	-3	0	1	-1	-2	-1	1	6	2	0	-1	-2	-1	-3	-3	-4	-3	-3	-3	-4
E	-4	0	0	-1	-1	-2	0	2	5	2	0	0	1	-2	-3	-3	-3	-3	-2	-3
Q	-3	0	0	-1	-1	-2	0	0	2	5	0	1	1	0	-3	-2	-2	-3	-1	-2
H	-3	-1	0	-2	-2	-2	1	1	0	0	8	0	-1	-2	-3	-3	-2	-1	2	-2
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5	2	-1	-3	-2	-3	-3	-2	-3
K	-3	0	0	-1	-1	-2	0	-1	1	1	-1	2	5	-1	-3	-2	-3	-3	-2	-3
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5	1	2	-2	0	-1	-1
I	-1	-2	-2	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4	2	1	0	-1	-3
L	-1	-2	-2	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4	3	0	-1	-2
V	-1	-2	-2	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4	-1	-1	-3
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6	3	1
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7	2
W	-2	-3	-3	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11

**Figure 4.** BLOSUM62 Substitution Matrix; see source, <ftp://ftp.ncbi.nlm.nih.gov/blast/matrices>

protein). Therefore, when aligning protein sequences, we would like to give a higher score to an alignment between I and V than to one aligning I and K.

Intense and detailed studies of proteins known to be homologous have been carried out over the years. The outcome of these studies has been a number of *substitution matrices* that can be used to score individual amino acid alignments when comparing protein sequences. One of the most commonly used of such matrices is called BLOSUM62. It appears in Figure 4 (BLOSUM comes from BLOck Substitution Matrix, and these matrices were originally proposed by Henikoff and Henikoff (15)).

Let us understand this matrix. First of all, it is easy to use it to determine the score of a given pair of aligned amino acids. Just find one amino acid in a row and then the other amino acid in a column. The corresponding entry in the matrix gives the score for their alignment (the matrix is symmetric, so it does not matter which of the two amino acids you pick to find the row). So, for example, for I and V, we get a score of +1, whereas for I and K, the score is -3. Notice that the diagonal contains the highest values (the self scores). These values are used for matches. It may seem strange that these values are not all equal, but this reflects the fact that over the course of evolution, it may be the case that after a series of substitutions, an amino acid can revert back to what it was originally. The higher the probability of this happening, the lower the self score of this amino acid (in other words: amino acids that can be more easily substituted by other amino acids will have lower self scores than those that are less easily substituted). So tryptophan (W), the largest of all amino acids, has the highest self-score, and this means that if we see a W aligned with a W, it is more probable (with respect to the other amino acids) that no substitution at all happened at this position. Tryptophan's low substitution rate may reflect the unique role it plays in protein folding because of its relatively large size. [AG3]

If we use BLOSUM62 to score a human vs. worm cytochrome *c* sequence alignment, this is what we get (only part of the alignment is shown):

```

H: GDVEKGKKIFIMKCSQCHTVEKGGKHKGTGPNLHGLFGRKTGQAPGYSYTAANKNKGI IWG
   GD EKGKK++ +C QCH V+      KTGP LHG+ GR +G   G+ Y+AANKNKG++W
C: GDYEKGKKVYQRCLQCHVVDSTAT-KTGPTLHGVIGRSGTVSGFDYSAANKNKGVVWT

```

We are presenting here yet another format to display alignments. The original sequences appear in the topmost and bottommost rows. In the middle are shown the matches (by repeating the matched character) as well as the positive mismatches, or simply **positives** (denoted by the + sign), which are those single-character alignments that have positive scores according to the substitution matrix that was used. This shows that the similarity between the human and the worm sequence is even better than what we obtained when we considered only a percent identity scoring system. This increase in accuracy is highly desirable, especially when comparing distantly related organisms, as is the case of humans and worms.

Ignoring the unmatched residues at the start of the human sequence (refer back to the original alignment in the first section), the human/worm alignment has a score of 339 when using BLOSUM62. We will see later on that more important than this score (usually called the **raw score**) is the statistical significance of the score.

Why does BLOSUM62 have a number in its name? Because evolution takes place over many millions of years, the kinds of substitutions observed between two sequences will differ, depending on how far apart they are, evolutionarily speaking. Certain kinds of substitutions will be more probable for proteins that have diverged for a long time as compared with proteins that have diverged more recently. The number in the matrix reflects this fact (62 means 62% identity; and, roughly speaking, it is a measure of the *minimum* similarity among the proteins or protein segments used to derive the matrix). Higher BLOSUM matrix numbers (e.g., BLOSUM80) are derived from more recently diverged proteins, and lower numbers (e.g., BLOSUM45) are derived from more distant proteins. This implies that substitution penalties in higher-numbered BLOSUM matrices are lower than in lowered-numbered ones. Of course it may be the case that we do not know beforehand how far apart, evolutionarily, are the two sequences that we want to compare. In those cases, it is thought that BLOSUM62 is a good compromise. [AG4]

Another set of substitution matrices used in sequence comparison is called PAM (for Point (or Percent) Accepted Mutations, described in Dayhoff, Schwartz and Orcutt (14)). PAM matrices give an estimate of substitution rates among amino acids, assuming a certain percentage of amino acid changes. For example, the PAM1 matrix assumes one change for every 100 amino acids on average; the PAM250 matrix assumes 250 changes for every 100 amino acids on average. Thus, the numbers in the PAM matrix are directly related to evolutionary distance (which is the opposite of BLOSUM matrices, where the numbers are inversely related to evolutionary distance). PAM matrices other than PAM1 were not computed from real data, but extrapolated from PAM1. Historically PAM matrices were proposed before BLOSUM matrices, and for general sequence comparison they are considered less reliable than BLOSUM matrices. [AG5]

Explaining in detail how these matrices are built is beyond the scope of this chapter. Please see the references at the end.

## Sequence Databases and Statistical Significance of Alignments

Because of technological advances in DNA sequencing, an enormous amount of sequences for various organisms has been generated in the past 20 years. Moreover, the pace of sequencing is still exponential, which means that the current number of available sequences, however large, will be small compared with what we can expect to have a few years from now.

This situation has led to the creation of the so-called *public sequence databases*. These are DNA and protein sequence repositories, of which there are many around the world. One of the most important is called **GenBank** and is maintained by the National Center for Biotechnology Information (<http://ncbi.nlm.nih.gov>). A separate

chapter would be needed to give a full description of the sequence data kept in GenBank and the ways it can be accessed. More information can be found in the chapter about GenBank in the [NCBI Handbook](#). For the purposes of this chapter, however, we will simply consider that GenBank provides a catalog of sequences containing millions of entries; we will call this catalog simply "the database", and the process of comparing one sequence against all sequences in the catalog as the *database search problem*.

This introduction sets the stage for the problem that we would like to consider in this section: what to do if we have a new sequence  $s$  and want to know whether it is similar to another sequence present in the database. This has become an extremely important problem, precisely because of the usefulness of sequence comparison as indicated at the beginning of this chapter. Given what we have already covered, there is a simple (but naïve, as it will turn out) answer to the problem: simply compare  $s$  with every sequence  $t$  present in the database using the dynamic programming algorithm that we have learned. This approach, however, immediately raises two significant questions (1): how long will this comparison take? and (2) how can we tell which database sequences are *significantly similar* to our sequence? Let us address each of these problems in turn.

Let us do a back-of-the-envelope calculation to see what kinds of running times we would get, were we to apply the dynamic programming algorithm for pairwise sequence comparison to the database search problem. Let us assume that each pairwise comparison takes one-tenth of a second (a fair assumption in a comparison of two 1-kb-long DNA sequences, for a very good implementation of the Smith-Waterman algorithm, running on a top-of-the-line desktop), and that the database contains one million sequences (also fair). Then, to do just one query to the database (that is, compare one sequence with all others) would take us 100,000 seconds, or almost 28 hours. In today's world, this is unacceptably long. This calculation does not even take into account that the Smith-Waterman algorithm runs in a time proportional to the product of the sequences being compared. In other words, for sequences twice as long, the algorithm will take four times as long, roughly speaking. This state of affairs was recognized early on in the history of sequence databases and led to the development of more efficient algorithms for database sequence comparison. The best known of these is called BLAST and will be the subject of the next section. Before we describe BLAST, however, we will turn to the second question raised above.

We have mentioned two similarity concepts: the percent identity and the similarity based on a scoring system. Percent identity is reasonably intuitive, as already pointed out: if two sequences are sufficiently long and have 90% identity, then there are excellent grounds to believe that this is not by chance (in other words, that the genes that they represent are homologous; but remember the caveats from the second section). On the other hand, a number such as 20% identity seems too low. To see this, let us determine the expected percent identity between two totally random DNA sequences of exactly the same length. If we do not allow gaps, then there is just one possible global alignment. In each column of this alignment, there are exactly 16 possibilities (4 possible nucleotides above times 4 possible nucleotides below). Of those 16 possibilities, 4 produce identities. So we can expect that in each column, there is a 25% chance of a match, and this is the expected percent identity in an ungapped alignment between two random sequences of the same length. Although this reasoning is extremely simplified, it can be concluded that for DNA sequences, an optimal alignment with percent identity not much larger than 25% is indicative that the two sequences are not related.

We can be more precise about these statements, but it turns out that percent identity is less used and therefore less important than similarity based on a scoring system. However, in such a system, the similarity number is much less intuitive. The score between the two cytochrome  $c$  sequences obtained with the BLOSUM62 matrix is 339, as we saw in the previous section. What does this number mean? Is it good or bad? Just as we did for the percent identity, we need to compare this number to numbers that would have been obtained by chance alignments. In other words, we have to attach some *statistical significance* to alignments. A statistically significant alignment will be one that is unlikely to have arisen just by chance. The more significant the alignment is (and the significance must be quantified), the higher the probability is that it is *not* a chance alignment.

The theory of statistical significance of alignments is beyond the scope of this chapter. We will limit ourselves to describing, in the next section, how the BLAST algorithm for the database search problem deals with statistical significance of the alignments it finds.

## BLAST

BLAST is an algorithm/program for sequence comparison first published in 1990 (7). The name is of course catchy; moreover, it is an acronym for *basic local alignment search tool*. As already mentioned, it was developed specifically for database searches, although it can be used to compare any two sequences. BLAST is much more efficient than the Smith-Waterman algorithm, but this efficiency is achieved at a small price in terms of quality of results.

In this section, we give an overview of BLAST. It should be enough to give readers a basic understanding of how BLAST works, but it comes nowhere near a full explanation of the BLAST algorithm and associated statistical theory. Interested readers should refer to the bibliography to learn more or to the chapter about BLAST in the [NCBI Handbook](#).

Before we start with technical details about BLAST, it is worth mentioning a few facts about it. In 1997, a new version of BLAST was published (8). The two references cited are among the most widely cited references in the technical literature of science. This is an indication of the usefulness of BLAST (and more generally of sequence comparison) for modern research in molecular biology and related fields. Another fact is that the concepts of algorithm and program are intimately connected for BLAST. As explained above, the concepts of algorithm and program are different. Usually, there will be an algorithm with many different implementations of it in particular programming languages. In the case of BLAST, the two notions go together: for practical purposes, there is no distinction between the algorithm and the program. Finally, BLAST has become a *family of programs*, and in this chapter we will be primarily interested in its most basic versions. Please note also that the BLAST program can be run on the NCBI website (as well as from many other sites), as well as downloaded (free of charge) from the same site and run locally.

The first technically important fact about BLAST is that, as its name says, it is a local alignment algorithm. Another important fact is that BLAST is not guaranteed to find the optimal alignment (best score) for any given comparison. This is the price paid for its efficiency. Let us understand what this means. We have seen that the DP algorithm *is* guaranteed to find the *best* alignment between any two sequences given a scoring scheme. As we have also seen, the best alignment is not necessarily statistically significant. On the other hand, there may be more than one statistically significant alignment between two sequences. This can happen because we are dealing here with local alignments. One "small" local alignment may already be significant, and if it is enlarged, its similarity may increase and so probably does its significance. In the context of BLAST, this means that although for a particular pairwise comparison BLAST is not guaranteed to always find the best alignment, almost always it will find a statistically significant alignment between the two sequences if one exists. Practice has shown that even with this caveat, BLAST is extremely useful, and the price of its efficiency is certainly worth paying. (Technically speaking, the lack of a guarantee to always find the best alignment means that BLAST is a *heuristic*, as opposed to an *exact algorithm*, as is the case of dynamic programming.)

BLAST is based on the concept of *seed alignment*. Given an alignment between two highly related sequences (such as the cytochrome *c* sequence in human and fly), it becomes immediately apparent that they share regions of consecutive matches (in that example, there are seven regions with five or more matches). The idea behind BLAST is based on this observation: given two sequences, first try to find a segment of consecutive matches (or a segment of high score, when comparing protein sequences) between the two; and then try to extend this alignment (the seed alignment) to the left and to the right. The extension proceeds until the inclusion of additional columns to the alignment fails to improve the overall score. If the current score is above a certain threshold, the resulting alignment (a *high scoring segment pair* (HSP) in BLAST terminology) is output.

Much of the speed of BLAST comes from a highly efficient way of finding seed alignments. The description that follows applies to protein sequences. The key concept here is (again in BLAST terminology) the *word*. A word is a string of  $k$  letters. Given a sequence of length  $n$ , it will contain  $n - k + 1$  words of length  $k$ . For example, the sequence QCHTVEKGGKHK contains the following words of size  $k = 3$ :

QCH,CHT,HTV,TVE,VEK,EKG,KGG,GGK,GKH,KHK.

All  $k$ -words for a given query can be easily determined; and for each such  $k$ -word, all  $k$ -words that have an alignment score with it above a certain threshold (using an amino acid scoring matrix) can also be efficiently found and stored (let's call this the *query word list*). All  $k$ -words for all sequences in a database can be precomputed and also stored in a special table. Thereafter, it becomes a matter of finding which of the database words appear in the query word list. Using sophisticated computer science data structures and programming tricks, BLAST is able to efficiently find all  $k$ -words of the query that have an alignment score above a certain threshold. The time BLAST takes in this process grows linearly with the size of the query word list as well with the size of the database. In the case of a protein sequence search, the default value for  $k$  is 3. For nucleotide comparisons,  $k = 11$  (although the method for finding high-scoring word pairs is different).

The running time complexity of BLAST depends not only on the time spent finding seed alignments but also on the time extending or combining these alignments. I am not aware of a rigorous running time complexity analysis of the complete BLAST algorithm. In practice, however, it runs very fast, perhaps 50 times faster than a good Smith-Waterman implementation for a typical protein-protein comparison.

We will now explain some additional key concepts that users of BLAST should know about. To help assess the quality and significance of an alignment, BLAST reports three types of scores: the **raw score**, the **bit score**, and the **expect value** or **e-value**. The raw score is simply the score of the alignment based on the scoring system and scoring matrix used. The bit score is computed from the raw score using a formula that normalizes it with respect to the scoring system; therefore bit scores from searches using different scoring systems are comparable. Finally, the e-value gives the statistical significance of the alignment. Its meaning is: the number of alignments of the same size that could be expected to be obtained purely by chance from a database of a given size using the given scoring matrix. In other words, an e-value of 1 means that we could reasonably expect that the alignment we obtained is due to chance and, therefore, is not biologically meaningful. An e-value of 0.00001 (which is  $10^{-5}$ , denoted by e-5 in BLAST) suggests that the alignment obtained has a much lower probability of having arisen by chance. An e-value of e-100 suggests that the database hit that was found is almost certainly a homolog.

Note that the concept of e-value is not specific to BLAST. It is perfectly possible to compute e-values for alignments obtained by dynamic programming. It is beyond the scope of this chapter to explain how bit scores and e-values are derived, but let us see what scores and e-values we obtain for the cytochrome *c* alignments that we have been using as examples (which were obtained by BLAST). First, the **human** and **mouse** comparison (the format follows usual BLAST reports):

```

Score = 203 bits (517), Expect = 3e-53
Identities = 95/104 (91%), Positives = 98/104 (94%)

Query: 1   GDVEKGKKIFIMKCSQCHTVEKGGKHKHTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
60
          GDVEKGKKIF+ KC+QCHTVEKGGKHKHTGPNLHGLFGRKTGQA G+SYT ANKNKGI WG
Sbjct: 1   GDVEKGKKIFVQKCAQCHTVEKGGKHKHTGPNLHGLFGRKTGQAAGFSYTDANKNKGITWG
60

Query: 61  EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE 104
          EDTLMEYLENPKKYIPGTKMIF GIKKK ERADLIAYLKKATNE
Sbjct: 61  EDTLMEYLENPKKYIPGTKMIFAGIKKKGERADLIAYLKKATNE 104

```

For this alignment, we obtained a raw score of 517, a bit score of 203, and an e-value of  $3e^{-53}$ , or  $3 \times 10^{-53}$ , as can be seen in the first two lines. Note that hits in the database are called "subject sequences" in BLAST terminology.

The next alignment is **human** and **fly**:

```
Score = 176 bits (447), Expect = 2e-45
Identities = 80/102 (78%), Positives = 87/102 (85%)

Query: 1  GDVEKGKKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
60
          GDVEKGKK+F+ +C+QCHTVE GGHK GPNLHGL GRKTGQA G++YT ANK KGI W
Sbjct: 5  GDVEKGKKLFVQRCACHTVEAGGKHKVGPNLHGLIGRKTGQAAGFAYTDANKAKGITWN
64

Query: 61 EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKAT 102
          EDTL EYLENPKKYIPGTKMIF G+KK ER DLIAYLK AT
Sbjct: 65 EDTLFEYLENPKKYIPGTKMIFAGLKKPNERGDLIAYLKSAT 106
```

Finally, **human** and **worm**:

```
Score = 135 bits (339), Expect = 5e-33
Identities = 59/99 (59%), Positives = 76/99 (76%), Gaps = 1/99 (1%)

Query: 1  GDVEKGKKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWG
60
          GD EKGKK++ +C QCH V+      KTGP LHG+ GR +G  G+ Y+AANKNKG++W
Sbjct: 6  GDYEKGKKVYKQRCLQCHVVDSTAT-KTGPTLHGVIIGRTSGTVSGFDYSAANKNKGVVWT
64

Query: 61 EDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLK 99
          ++TL EYL NPKKYIPGTKM+F G+KK +ERADLI Y++
Sbjct: 65 KETLFEYLLNPKKYIPGTKMVFAGLKKADERADLIKYIE 103
```

Notice how the e-values are increasingly *larger* with increased phylogenetic distance between the organisms (although the absolute value of the exponents -53, 45, 33 in the examples- decreases; this is the source of some confusion). But even at  $5e^{-33}$ , the alignment is still highly significant, which allows us to say with confidence that the two sequences should be homologs. e-values up to  $e^{-5}$  in general suggest homology. From  $e^{-5}$  to  $e^{-1}$ , the alignments have to be considered with extreme caution. e-values greater than 0.1 in general suggest that the respective alignments are due to chance. However, please remember that there are a lot of parameters involved (database size, gap costs, scoring matrix, and so on), so what you get is highly dependent on the values used for all of these parameters. In particular, e-values resulting from BLAST searches using different databases are not directly comparable, because the e-value depends on the database size. There are formulas that allow the conversion of an e-value obtained in a query against a database of a certain size to a new value assuming a different database size; such formulas (not covered here) allow an e-value to be given for the BLAST comparison of one sequence against just one other sequence, assuming for example the entire GenBank as the database.

Another important concept for BLAST users is the notion of *sequence complexity*. Analysis of database sequences has shown that certain subsequences contain repetitive elements or appear with relatively high frequency as parts of other sequences. Because of this, it is not uncommon for the local alignment of such sequences or regions to be due entirely to chance. Such sequences are given the designation of *low complexity sequences*. An example of a low complexity DNA sequence is AAAAAA (which is more or less obvious); an example of a low complexity amino acid sequence is AGNLLGRNVVVVGAG (which is far from obvious). To avoid the (possibly) artificial increase in score in an alignment that would be caused by the presence of low complexity sequences, BLAST provides a *low complexity filter*, meaning that even if present, a sequence classified as low complexity

does not contribute to the total score. When the low-complexity filter is used and a low-complexity sequence is present in the alignment, it appears *masked* in the result with the character N (if nucleotide) or X (if amino acid); or the low complexity region is highlighted in a different color and in lowercase. See an example below.

```
KMVARVFIPFAFSLTFSIFRNINAVLAPFXXXXXXXXXXXXXXXXXXXXYFLSFLAQLPV
++V VFIPFA + +++FR INA ++P L S L A + GLL S YFL F +AQ+PV
RIVLSVFIPFAAGYYLAYLFRITINAAISPALASEFGLDAAETGLLASVYFLVFGVAQIPV
```

Note that sometimes the use of the low-complexity filter will unduly lower the score; this can happen when the entire alignment is not due to chance, and both query and subject sequences possess a matching region classified as low complexity. To avoid this problem, BLAST provides a *soft masking* alternative: low-complexity sequences are masked during the seed alignment phase but are present normally during alignment extension.

The reader should bear in mind that even with a stringent e-value threshold and use of the low-complexity filter, there can still be false matches (also called false positives). A partial example is given below:

```
Score = 219 bits (557), Expect = 3e-55
Identities = 144/449 (32%), Positives = 243/449 (54%), Gaps = 11/449 (2%)

Query  50  NVRILDLSRQKFAVFPKEIWELEYLEILKLEENRITVLPREINKLKNLKELYLNGNKLTI  109
        N+R L L+      P + L YLE L L +N +T +P ++ LKNL+ L L N+LTI
Sbjct  229  NMRQLLLNSNHIDTLPSPGLEHLRYLETLSLGNMLTYIPDSLSSLKNLRILNLEYNQITI  288

Query  110  VPKEIWELENLTILRELENNRISTLPKEIEKSKNLQELDLRGNRLVTLPEGIGELKLEEL  169
        K + L L L L N I +LPKE+ + KNL+ L + N+L L I +L ++EL
Sbjct  289  FSKSLCFLPKLNSLNLGTGMIGSLPKEVRELKNLESLLMDHNKLTFLAVEIFQLPKIKEL  348
```

Only the first two rows of the alignment are given. Notice that the e-value appears to be very good. But then note that there seems to be a high frequency of leucines in both the query and the subject. In fact, both query and subject are leucine-rich repeat proteins, although one of them is found in the bacterium *Leptospira interrogans* and the other is found in mouse (*Mus musculus*). It is unlikely that these two proteins are homologs. Turning on the low-complexity filter will not eliminate this false positive, as can be seen below (low-complexity sequences appear in lowercase; only the first two rows of the alignment are shown):

```
Score = 144 bits (362), Expect = 1e-32
Identities = 143/453 (31%), Positives = 239/453 (52%), Gaps = 19/453 (4%)

Query  50  NVRILDLSRQKFAVFPkeiweleyleilkleenRITVLPREInklknlkelylنگnkltI  109
        N+R L L+      P + L YLE L L +N +T +P ++ LKNL+ L L N+LTI
Sbjct  229  NMRQLLLNSNHIDTLPSPGLEHLRYLETLSLGNMLTYIPDSLSSLKNLRILNLEYNQITI  288

Query  110  VPKEIWELENLTILRELENNRISTLPKEIEKSKNLQELDLRGNRLVTLpegigelkleeel  169
        K + L L L L N I +LPKE+ + KNL+ L + N+L L I +L ++EL
Sbjct  289  FSKSLCFLPKLNSLNLGTGMIGSLPKEVRELKNLESLLMDHNKLTFLAVEIFQLPKIKEL  348
```

The score has decreased, but the e-value still suggests the alignment is significant.

As we have seen, BLAST has an *extension phase*, when seed alignments are extended with the goal of improving the score. During this phase, gaps may be introduced, and gap inclusion incurs penalties on the score given by an affine function, as described in the section Finding the Similarity. BLAST users can change the values of the constants for gap opening ( $G$ ) and gap extension ( $E$ ). The default values for proteins are  $G = 11$  and  $E = 1$ . (BLAST convention gives these constants as positive values, but they are penalties exactly as described in the section Finding the Similarity.) However, only certain combinations of values for  $G$  and  $E$  are allowed, and they depend on the scoring matrix being used. For example, in the case of BLOSUM62,  $G$  can vary in the range



[6,12], whereas  $E$  can be either 1 or 2. Such limitations are necessary to preserve the statistical properties of alignments.

We finish this section by mentioning BLAST types and BLAST variants. Biological sequences are of two kinds: nucleotides and proteins. BLASTN is the type of BLAST used to compare nucleotide sequences. BLASTP is used to compare protein sequences. In addition to these comparisons, it turns out that it is extremely useful to be able to compare nucleotide sequences to protein sequences. Today it is much easier and cheaper to sequence DNA than it is to sequence a protein directly. Therefore, a crude (but effective) method to determine whether a DNA sequence codes for a protein is to translate the DNA sequence in all of its six frames and compare each of those translations to known protein sequences. BLASTX is a type of BLAST that does exactly that: given a DNA sequence, it translates it in all six frames and compares each translation to the protein sequences in the database. TBLASTN is similar, but the query is assumed to be protein and the database is assumed to be in nucleotides. Finally, TBLASTX compares query nucleotide sequences against database nucleotide sequences but translates *both* query and database sequences in all 6 frames before comparing them. It is the most costly (in computer time) of the BLAST family of programs, as one would expect.

In addition to these types of BLAST, there are many BLAST variants. These are programs inspired by BLAST but that use different algorithms and/or perform specialized sequence comparisons. Two noteworthy ones are:

- **MegaBLAST:** It is for nucleotide comparisons only, and it is aimed at comparing sequences that are already known to be similar. In such cases, MegaBLAST can be much faster than BLASTN and is able to handle much longer sequences.
- **PSI-BLAST:** It stands for position-specific iterated BLAST and is meant for protein sequences. The full understanding of this variant requires the concept of *position specific scoring matrix*, which is beyond the scope of this chapter. However, the idea is as follows: given a set of related protein sequences (a family), it is possible to align all of them together (see below for mulTiple alignment) and capture the variations observed in each column (or position) in a matrix. With such a matrix, we can search for additional members of the family in a much more sensitive fashion than we would with each member of the family individually as a query. We can then use the members found to refine the matrix and repeat the process. This *iterative* search process will likely increase the sensitivity even further. This is the processing that PSI-BLAST offers.

## A Note on Global Alignments

We have seen the difference between global and local alignments; we have also seen that BLAST finds local alignments. For most sequence comparison needs, a local alignment is all that is needed; and in many cases BLAST (and several other local alignment programs) returns alignments that are "nearly" global (in the sense that the alignment includes nearly the whole of query and subject). That was the case with the cytochrome *c* alignments that we have shown previously. There are, however, situations where a strict global alignment is required. One such example is the comparison of proteins from the same protein family. We need to know how different members of the family relate to each other in an exact quantitative fashion; if one member is shorter than another, there has to be a penalty that will account for the unmatched portion of the longer one. Only with a global alignment will we be able to achieve this.

## Sequence Comparison for Very Large Sequences

As we have seen, the running time of the dynamic programming algorithm for sequence comparison is proportional to the product of the lengths of the sequences being compared. The main reason for this, as we explained, is the size of the DP matrix. This matrix needs to be stored in the main computer memory (RAM) and takes a lot of space for large sequences. Although there are certain tricks that can decrease the amount of memory required, when comparing very large sequences, the bottleneck in using the DP algorithm is usually

lack of memory and not excessive time. This means that even when we can afford the time to execute the DP algorithm, our computer may not have enough memory to run it. This in turn means a limit on the sizes of sequences that we may be able to compare using DP.

The BLAST algorithm/program was not designed to compare large sequences, either. We have already mentioned the variant MegaBLAST, which is able to handle much larger sequences. However, what exactly do we mean by "large sequences"?

The typical protein has around 300 amino acids, but there are some that are 10 times longer. A prokaryotic protein-coding nucleotide sequence for a 3,000-long protein sequence will be 12 kb long. Both DP and BLAST can handle these kinds of sequences without any problems. Problems begin to appear when we want to compare full genomes, or entire chromosomes. These are sequences that have millions of nucleotides, and DP, BLAST, and even megaBLAST will "choke" if asked to compare sequences this long.

Given that full genome sequences have become common, and that comparing them is just as important as it is to compare individual genes, what are we to do? The answer lies in the clever use of special computer science techniques and data structures. One such structure is the *suffix tree*. It is beyond the scope of this chapter to explain what a suffix tree is. Suffice it to say that it allows memory- and time-efficient detection of *exact* substring matches between two sequences. It is an idea similar to the seed alignment of BLAST, but with a completely different implementation.

A well-known program that uses the suffix tree technique is called MUMmer (10). This program finds all Maximal Unique Matches (MUMs) with a user-defined minimum length between two DNA sequences. The important thing to remember is that a MUM is an *exact* match. But just as is the case of BLAST, experience has shown that long approximate matches (those that contain mismatches and indels) contain regions where exact matches are found. So MUMmer creates the whole alignment by finding only the component exact matches (the program also tries to link neighboring exact matches by applying the Smith-Waterman algorithm). Thanks to a carefully crafted suffix tree implementation, MUMmer is able to align whole bacterial genomes or entire eukaryotic chromosomes in a matter of minutes using reasonable amounts of computer memory (4 Gigabytes seems to be enough for most applications), assuming the default length of a MUM (25 nucleotides).

## Multiple Sequence Comparison

This section will give a basic introduction to the vast subject of multiple sequence comparison (MSC). MSC is a natural extension to the concept of pairwise comparison that we have dealt with so far. Given  $k > 2$  sequences, we want to know how similar they are to one another and obtain a multiple alignment, which is the obvious extension of the pairwise alignment that we are already familiar with. For example, Figure 5 shows a multiple alignment of several cytochrome *c* sequences.

There are two questions that we will address here. The first is: **Why would anyone want to compare multiple sequences and create a multiple alignment?** Can't we get by with pairwise alignments only? The second is (assuming there is a good answer to the first!): **How can we create a multiple alignment, and preferably a "good" one?**

For the first question, the answer depends on the kinds of sequences that one wants to compare. A simple case is illustrated by the multiple alignment in Figure 5. From the pairwise alignments we *already know* that the input sequences are related. The primary reason we want to align them all together is to see what they have in common. Such information is not easy to derive from the pairwise alignments. For example, a simple visual inspection of Figure 5 immediately tells us that all cytochrome *c* sequences have one section of 11 amino acids that is exactly the same in all of them (NPKKYIPGTKM), although the sequences belong to extremely different biological species. Such information can only be obtained by a multiple alignment and is extremely valuable to

```

C : ----SDIPAGDYEGKGVYKQRCLQCHVVDSTAT-KTGPTLHGVIIGRTSGTVSGFDYSAA
Y : ----TEFKAGSAAKKGATLTKRCLQCHTVEKGGPHKVGPNLHGIFGRHSGQAEGYSYTD
A : MASFDEAPPGNPKAGEKIFRTKCAQCHTVEKGAHGKQGNLNGLFGRQSGTTPGYSYSAA
D : -----GVPAGDVEKGGKLFVQRCAQCHTVEAGGKHKVGNLHGLIGRKTGQAAGFAYTDA
H : -----GDVEKGGKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTA
M : -----GDVEKGGKIFVQKCAQCHTVEKGGKHKTGPNLHGLFGRKTGQAAGFYSYTD
      * . : * . : : . * * * . * : . * * * . * : : : * * : * . * : * : *

C : NKNKGVVWTKETLFEYLLNPKKYIPGTKMVFAGLKKADERADLIKYIEVESAKSL
Y : NIKKNVLWDENNMSEYLINPKKYIPGTKMAFGGLKKEKDRNDLITYLKKACE---
A : NKSMAVNWEEKTYDYLLNPKKYIPGTKMVFPGLKKPQDRADLIAYLKEGTA---
D : NKAAGITWNEDTLFEYLENPKKYIPGTKMIFAGLKKPNERGDLIAYLKSATK---
H : NKNKGIWGEDTLMEYLENPKKYIPGTKMIFVGIKKKEERADLIAYLKKATNE--
M : NKNKGIWGEDTLMEYLENPKKYIPGTKMIFAGIKKKGERADLIAYLKKATNE--
      * : * : . : : : * * * * * * * * * * * * * * * * * * * * * : * * * * * : :

```

**Figure 5.** Multiple alignment of cytochrome *c* sequences. C, *Caenorhabditis elegans* (the worm); Y, *Saccharomyces cerevisiae* (yeast); A, *Arabidopsis thaliana* (a model plant); D, *Drosophila melanogaster* (the fly); H, *Homo sapiens*; M, *Mus musculus* (mouse). The format used is "clustalW"; the last row indicates columns where there is conservation. There are three cases: all residues in the column are the same (denoted by \*); conserved "strong groups" (denoted by the colon); conserved "weak groups" (denoted by the dot). The strong and weak groups refer to groups of amino acids in a substitution matrix called Gonnet PAM250.

those studying the evolution of proteins. Incidentally, such studies usually lead researchers to build phylogenetic trees, and the basis of most phylogenetic tree reconstruction methods is a multiple alignment.

Another situation arises when we are not sure whether certain sequences are indeed related to one another. Each pairwise score or *e*-value may be weak. By comparing all sequences together, we may discover that all of them share certain regions, which will give us much more confidence that they are indeed related. This situation arises when we are comparing homologous sequences from species that are quite distant to each other (as is the case with the species used in the multiple alignment of Figure 5) but that are not so well conserved (as is *not* the case with cytochrome *c*, which is very conserved even among distant species, as the alignment shows us).

Because of the need for multiple sequence alignments, we will give a brief sketch of the methods used to perform such comparisons. The simple dynamic programming algorithm that we have described can be extended to compare  $k > 2$  sequences. We will not explain how this can be done; please see the references. The resulting time and space complexity are proportional to at least  $2^k n^k$ , where  $n$  is the length of each sequence (assuming all are of the same length). This means that DP is impractical in most cases. The alert reader will probably now ask: isn't there the equivalent of BLAST for multiple sequence comparison? The answer is, yes and no.

It is "no" because there is no such thing as a BLAST-based or BLAST-inspired program for multiple sequence comparison. The answer is "yes" in the sense that there are *heuristic* programs for MSC that are able to tackle the MSC problem for relatively large values of  $k$  (for example, 50) and relatively large values of  $n$  (for example, 5 kb). However, such programs do not have guarantees that they will find the optimal multiple alignment, more or less in the same sense that BLAST is not guaranteed to find the optimal pairwise alignment.

One additional fact about multiple alignments worth knowing (at this introductory level) is how they are scored. The most basic scoring system for multiple alignments is an extension of the pairwise scoring system that we have described and is called *sum of pairs*. For each column in the alignment, we add all pairwise scores of characters present in that column. So for example, if a column has two isoleucines and one valine, the score of that column will be  $\text{score}(I,I) + \text{score}(I,V) + \text{score}(I,V)$ . The one exception is that if there are spaces in a column,

we do not charge for the situation in which a space is paired with another space. The total score of the alignment will be the sum of the column scores.

There are several programs that perform multiple alignments. Three of the best known are: clustalW (11), T-coffee (12), and MUSCLE (13). The introduction given in this chapter should be enough for the interested reader to be able to understand the outputs of these programs. Please note, however, that these programs are all heuristics, and that humans are still able in some cases to improve multiple sequence alignment results given by these programs by editing the alignments directly. To do so, however, requires considerable expertise in the structure and evolution of proteins.

## Practical Section

The following are solved exercises as well as **Tips** and tricks for the application of the theoretical knowledge covered in the theoretical portion of this chapter.

### 1. Why sequence comparison is important

#### Identity, similarity, percent identity

Q1.1: Consider the following alignment of the first part of two hemoglobin sequences, one human (H), one mouse (M):

```

H: MVLSPADKTN
   ||||..||:|
M: MVLSGEDKSN

```

Legend:

| identical

: very similar

. similar

As a first measure of similarity, what is the percent identity?

A1.1:

The percent identity is 70% (7 matches over 10 positions).

Q1.2:

You see 4 different alignments reported. Which alignment is the best?

100% (10/10)

91% (95/104)

74% (80/108)

53% (59/111)

A1.2:

It is important to not only consider the percent identity but also the length of the alignment. In our case the best alignment would be b). This is why alignment programs report scores (not shown here) which take both (matches/mismatches and length of the alignment) into consideration.

As a general rule of thumb we can consider the following alignments significant:

For DNA sequences:

more than 70 % identity over more than 100 bases.

For proteins:

more than 25 % identity over more than 100 amino acids.

Most importantly: have a look at the actual alignment and decide if it has any biological significance. Amino acid sequences after all make up three-dimensional proteins. A very good alignment of two coil regions tells you that these two proteins look similar whereas a good alignment of the catalytic residues tells you that the proteins probably have the same function.

## 2. What is sequence similarity

### Score DNA sequences

Q2.1:

Calculate the score for the following alignment:

```

1  GATAGACACAGACCGGTGGCATTGTGG  27
   |   |   |   ||   |   ||   ||   |
1  GTCGGGAAGAGATAACTCCGATGGTTG  27

```

Each match scores 5 points, each mismatch costs a penalty of 4 points.

A2.1:

The score is 0.

12 matches ( $12 \times 5 = 60$ ) plus 15 mismatches ( $15 \times -4 = -60$ ).

### Homology

Q2.2:

Is it correct to say that human and mouse cytochrome c are 91% homolog?

A2.2:

Homology is a hypothesis about a common ancestry of two sequences. Two sequences may have descended from a common ancestor or not. In the former case, they are homologues, and in the latter not. Therefore, there is no sense in stating homology in terms of percentage. It is either true or false. In addition, homology cannot be inferred using sequence similarity as the only criterion.

### 3. Finding the similarity between two sequences by dynamic programming

#### Gap penalties

Q3.1:

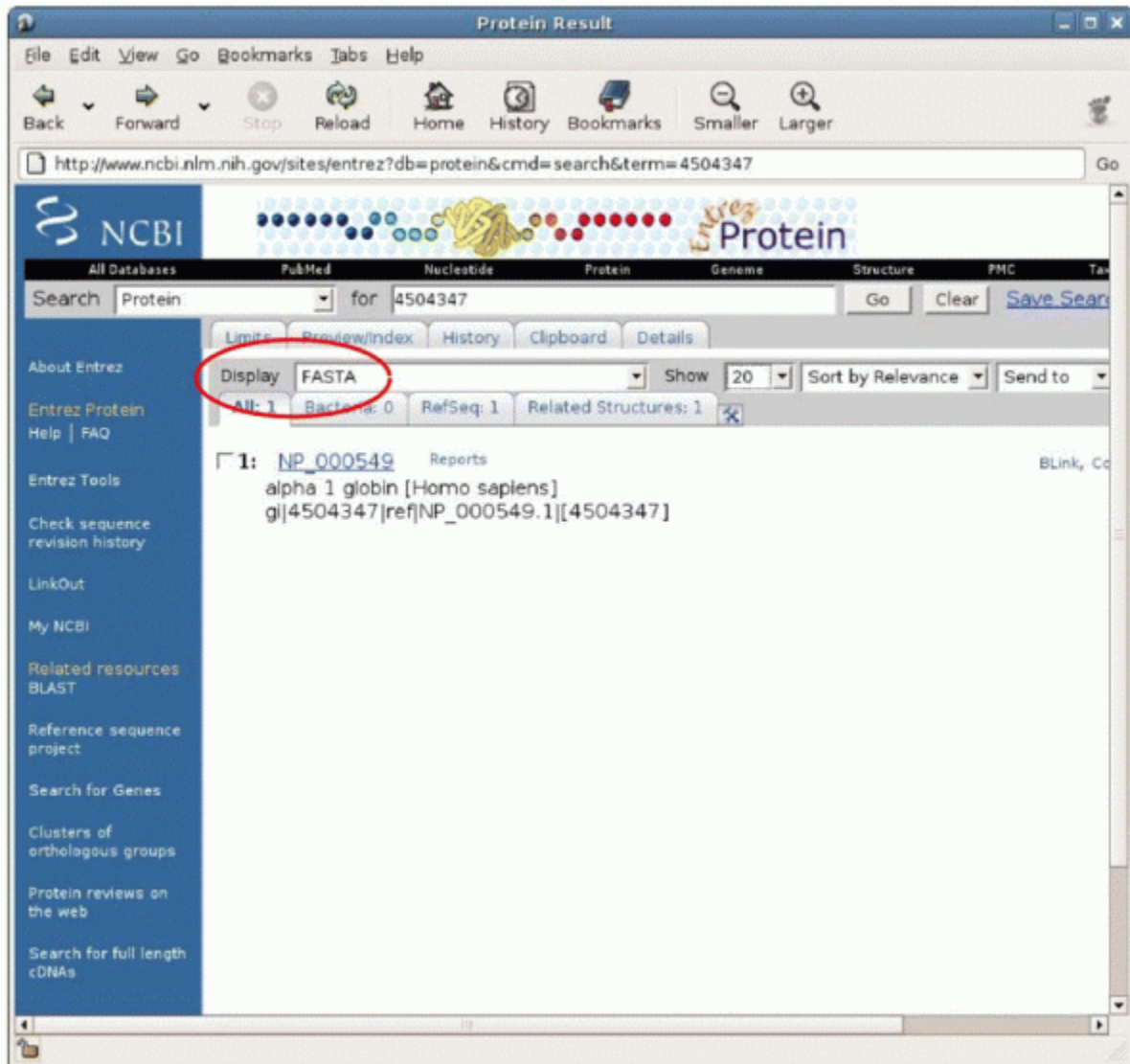
Now let's look at a complete alignment, but this time you will do the alignment of the following hemoglobin sequences:

```
>gi|4504347|ref|NP_000549.1| alpha 1 globin [Homo sapiens]
MVLSPADKTNVKAAWGKVGAAHAGEYGAEALERMFSLFPTTKTYFPHFDLSHGSAQVKGHGKKVADALTNA
VAHVDDMPNALSALSSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSK
YR
>gi|145301578|ref|NP_032244.1| hemoglobin alpha 1 chain [Mus musculus]
MVLSGEDKSNIAAWGKIGGHGAEYGAEALERMFASFPPTTKTYFPHFDVSHGSAQVKGHGKKVADALASA
AGHLDDLPGALSALSSDLHAHKLRVDPVNFKLLSHCLLVTLASHHPADFTPAVHASLDKFLASVSTVLTSK
YR
```

Either copy them from this document or retrieve them from the NCBI (<http://www.ncbi.nlm.nih.gov/>). At the NCBI search 'protein' for either the GI number (e.g. 4504347) or the accession number (e.g. NP\_000549.1):

The screenshot shows the NCBI homepage in a web browser. The search bar contains the text 'Protein' and the GI number '4504347'. The search results page is displayed, showing the NCBI logo, navigation links, and a search bar. The search results are filtered by 'Protein' and the GI number '4504347' is entered in the search box. The search results page shows the NCBI logo, navigation links, and a search bar. The search results are filtered by 'Protein' and the GI number '4504347' is entered in the search box.

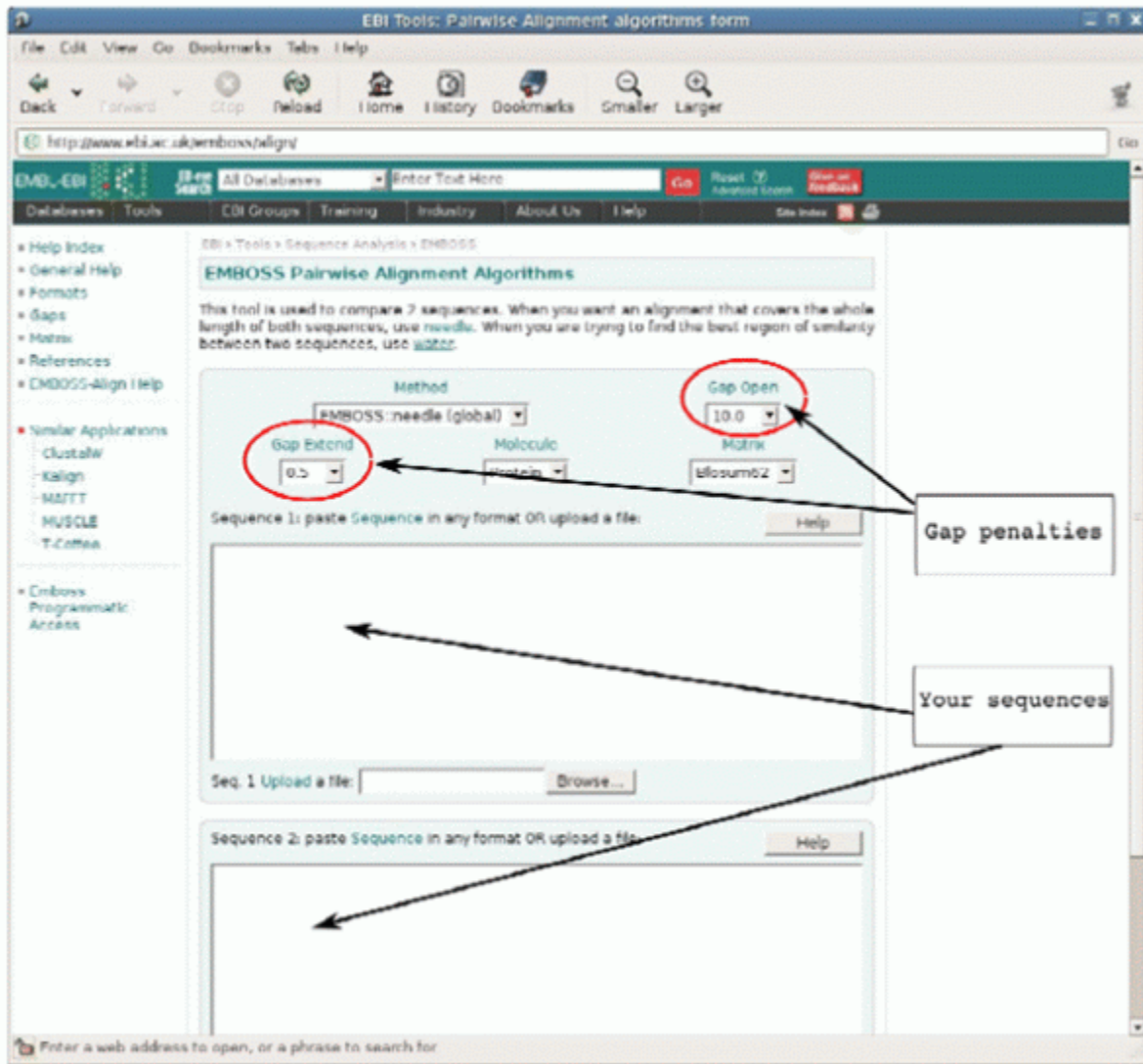
On the results page set the display to `FASTA`:



Align the two sequences over their complete length (global alignment), using **needle** (<http://www.ebi.ac.uk/emboss/align/>), an implementation of the Needleman-Wunsch algorithm.

Perform the alignment a number of times, varying the parameters in the following way:

- stick to the default parameters - Gap\_penalty: 10.0, Extend\_penalty: 0.5
- be conservative (i.e. gaps are a very rare event) - Gap\_penalty: 100.0, Extend\_penalty: 10.0
- be very flexible (i.e. gaps can occur frequently) - Gap\_penalty: 1.0, Extend\_penalty: 0.1



Have a look at the resulting alignments (percent identity, percent similarity, percent gaps, score).

Note that each result of **'needle'** tells you exactly, what you have done, thus making it easy to reproduce these results.

Don't forget that we are looking at proteins. Their function is based on their 3D structure. Introducing gaps in an alignment means that at that position an insertion happened. An insertion in a helix structure is much more disruptive than in a coil structure.

What do you observe?

A3.1:

a)





```
#
# Length: 142
# Identity: 122/142 (85.9%)
# Similarity: 131/142 (92.3%)
# Gaps: 0/142 ( 0.0%)
# Score: 648.0
```

This is a conservative approach. As a) already didn't contain any gaps nothing changes here. This means our alignment is quite robust.

```
c)
# Gap_penalty: 1.0
# Extend_penalty: 0.1
```

```
#
# Length: 146
# Identity: 124/146 (84.9%)
# Similarity: 133/146 (91.1%)
# Gaps: 8/146 ( 5.5%)
# Score: 656.0
```

This is a flexible approach. A number of gaps get introduced. The score of this alignment is higher than in a) or b) but does this make biological sense? In a) and b) we postulate that insertion/deletions are rare and differences in the sequences are due to mutations. In c) we say that insertions/deletions happen quite frequently. It's your biological knowledge that in this case has us preferring a)/b) over c).

## Local alignment

Q3.2:

Use our mouse (GI 145301578) and human (GI 4504347) hemoglobin sequences. Find the best regions of similarity between the two sequences using the program `'water'` (<http://www.ebi.ac.uk/emboss/align/>), an implementation of the Smith-Waterman algorithm. Compare the results with the alignments generated with the program `'needle'`.

A3.2:

The two sequences can be nicely aligned over their entire length. The best region of similarity stretches from one end of the sequence to the other. In this case global and local alignments present the same result.

Q3.3:

Let's stick with mouse hemoglobin and align mRNA (GI:145301577) and genomic (GI:49899) sequence using `'water'` (<http://www.ebi.ac.uk/emboss/align/>). Vary the gap penalty in the following way:

- a. stick to the default parameters
- b. be conservative (i.e. gaps are a very rare event)

c. be very flexible (i.e. gaps can occur frequently)

What do you observe?

A3.3:

There are regions of aligned characters and regions of gaps. These correspond, respectively, to exons and introns.

a)

Gap\_penalty: 10.0

Extend\_penalty: 0.5

3 exons, 2 introns, nicely separated

b)

Gap\_penalty: 100.0

Extend\_penalty: 10.0

1 exon, 1 intron, matches, mismatches and gaps all mixed.

c)

Open gap penalty 1.0

Gap extension penalty 0.1

3 exons, 2 introns, nicely separated

Tip

Sim4 (<http://pbil.univ-lyon1.fr/sim4.php>) is a specialized tool for aligning expressed DNA sequences with their genomic sequences. A slightly newer tool is Exonerate. It can be used at <http://sss.hgc.jp/> as part of the Sequence Similarity Search service of the Human Genome Centre, Japan. The program runs faster than sim4 but with the same accuracy.

Make sure that you are using alignment tools appropriate for your organism of research. E.g. sim4 and exonerate are meant to be used for eukaryotes with their intron/exon structure and not for prokaryotes.

## Global alignment

Q3.4:

Now use the mouse mRNA (GI:145301577) and genomic (GI:49899) sequences for hemoglobin and align them using `'needle'` (<http://www.ebi.ac.uk/emboss/align/>). Vary the gap penalty:

stick to the default parameters

be conservative (i.e. gaps are a very rare event)

be very flexible (i.e. gaps can occur frequently)

What do you observe?

A3.4:

There are regions of aligned characters and regions of gaps. These correspond to exons and introns.

a)

```
# Gap_penalty: 10.0
```

```
# Extend_penalty: 0.5
```

Alignment covers the complete length of both sequences.

3 exons, 2 introns, nicely separated

b)

```
# Gap_penalty: 100.0
```

```
# Extend_penalty: 10.0
```

To be more conservative we assign high gap penalties. Now the alignment doesn't cover the complete length of both sequences.

1 exon, matches, mismatches and gaps all mixed.

c)

```
# Gap_penalty: 1.0
```

```
# Extend_penalty: 0.5
```

To be more flexible we reduce the gap penalties to very small values. Now, the alignment covers the complete length of both sequences.

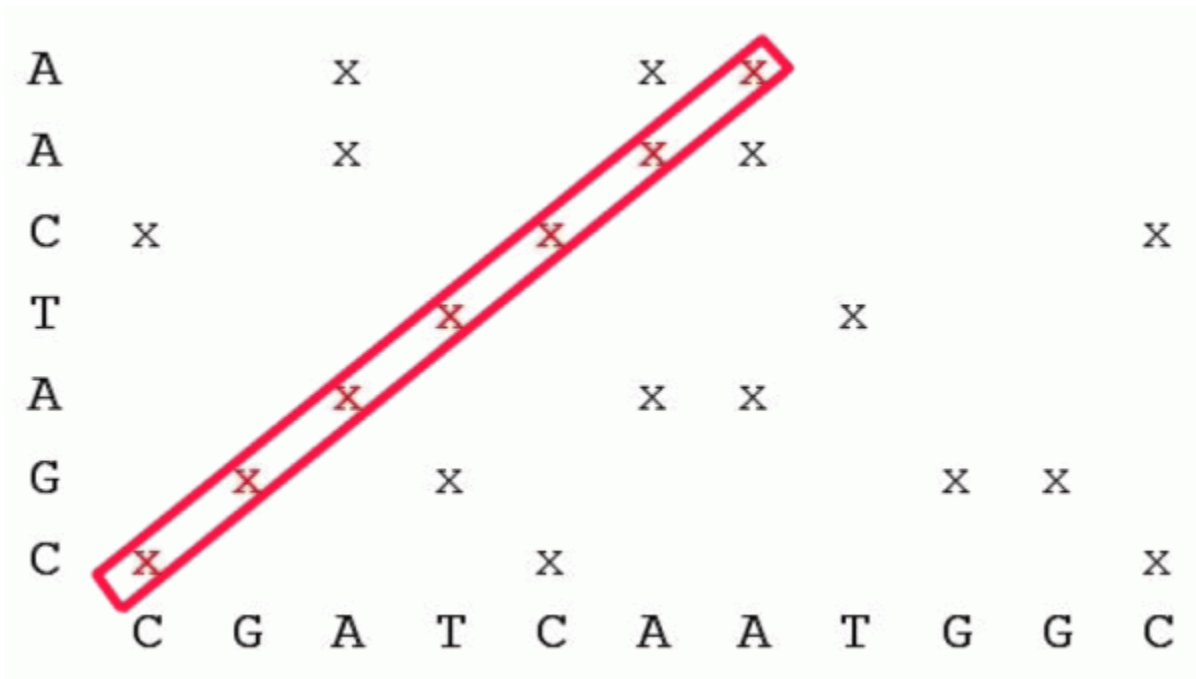
3 exons, 2 introns, nicely separated

NOTE: The scores used in b) and c) are examples and should not be considered universal values for being conservative or flexible.

## Dotplot

A great tool to give you an overview of all possible alignments of two sequences is Dotlet (<http://www.isrec.isb-sib.ch/java/dotlet/Dotlet.html>). It takes your sequences and writes one sequence along the x-axis and the other one along the y-axis. Next, it simply checks what residues both sequences have in common and puts a dot down for each match. A perfect alignment consists of an uninterrupted line of dots (matches). It allows you to consider neighbouring residues by specifying a word size.

To illustrate the underlying principle check this little example (each match is represented by the letter x):



The best local alignment is the longest diagonal (represented in red for clarity) which correspond to the alignment:

```

CGATCAA
| | | | |
CGATCAA
    
```

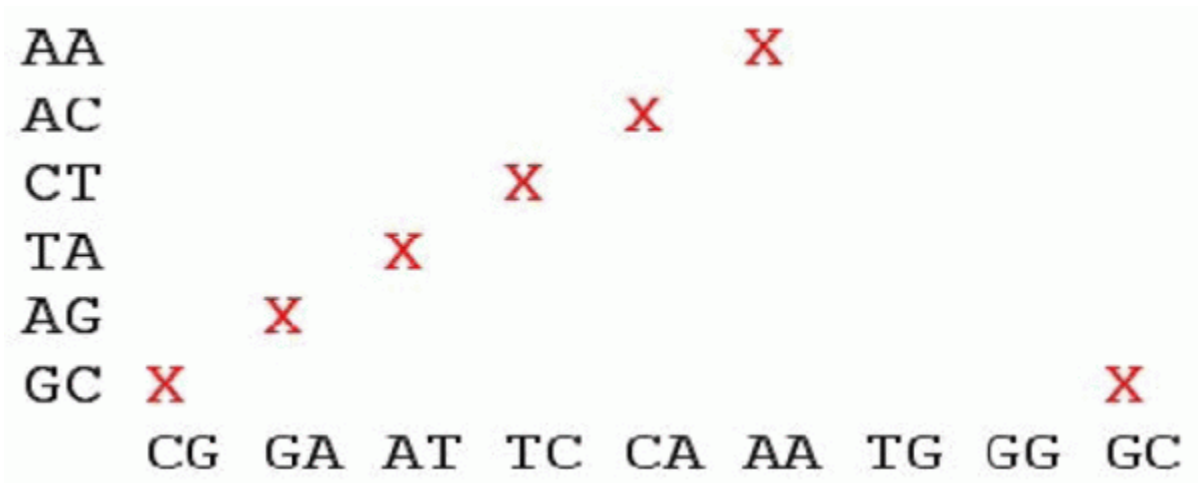
Dotplots of single bases tend to have too many positions filled up. We can use the same technique using a sliding window of bases, instead of comparing one base at a time. In the sliding window version, a sequence is scanned and converted into tuples and a comparison of tuples is compared instead of single bases. In the example above, using a window of two bases, we will have, for the first sequence the pairs

AA, AC, CT, TA, AG, GC

and for the second sequence the pairs

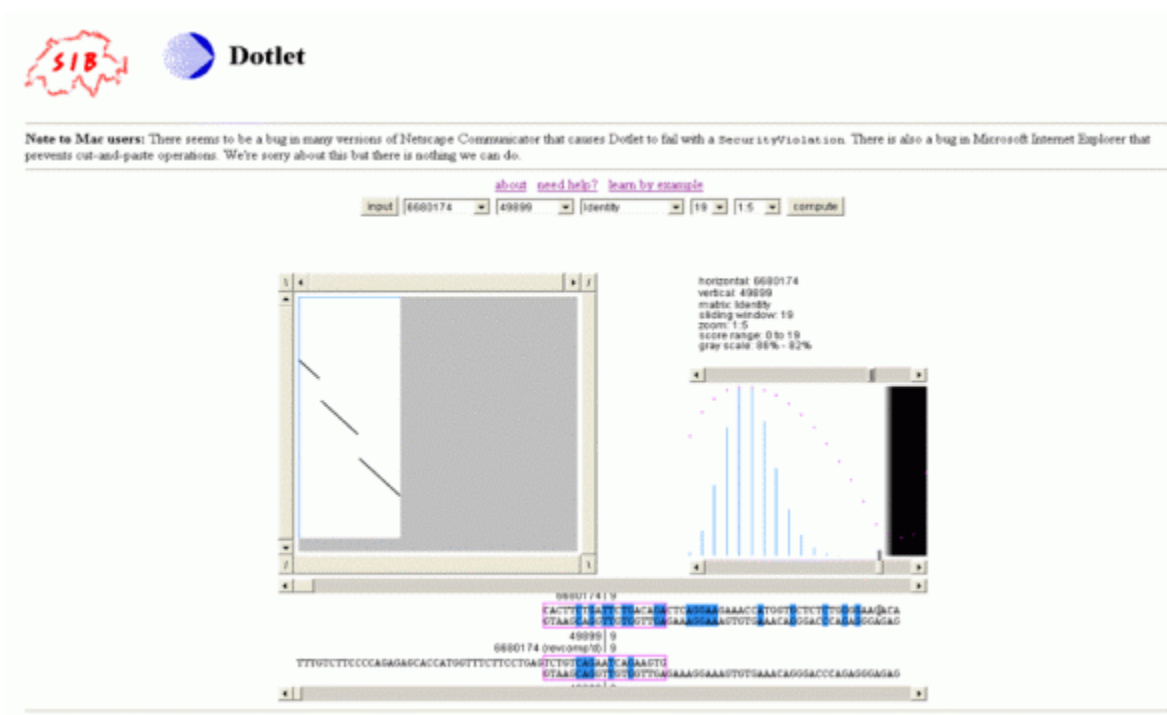
CG, GA, AT, TC, AA, TG, GG, GC

The effect of using a window is to lower the resolution of the dotplot. This is particularly useful if you are aligning two large sequences and are interested only in larger alignments. In our example, the dotplot with a window of two bases will be



Please note that an x is put in the diagram if there is a regular match or if the pair matches its reverse (NOT reverse complementary)

For our mRNA and genomic mouse hemoglobin sequences, we get the following result:



For more details, instructions and examples check the `need help?' and `learn by example' links.

## 4. Amino acid scoring systems

### BLOSUM number

Q4.1:

In Exercise 3.1, you aligned mouse (GI:145301578) and human (GI:4504347) hemoglobin sequences using the default scoring matrix BLOSUM62. Knowing that these hemoglobins diverged recently you want to fine-tune your alignment by using another scoring matrix. Would you go for a BLOSUM matrix with a higher or one with a lower number?

A4.1:

Recall what you've learned in the theoretical part of this chapter. Matrices help the computer to calculate the best alignment. A high BLOSUM matrix was created from a group of protein sequences with a high percent of similarity. In other words: the higher this percentage, the closer these sequences are (or the more recent they diverged). The better a chosen matrix fits your sequences, the better your result will be. While the choice of the appropriate scoring matrix increases the specificity and sensitivity of your alignments, it is still up to you to verify the biological significance of the corresponding result. If we would be looking for long and weak alignments we would choose a low BLOSUM matrix like BLOSUM40.

## Matrices

Q4.2:

Use our mouse (GI:145301578) and human (GI:4504347) hemoglobin sequences. Align the two sequences over their complete length (global alignment), using `'needle'` (<http://www.ebi.ac.uk/emboss/align/>).

`'Needle'`, as implemented by EMBOSS, allows you to use three different BLOSUM matrices: 62, 50 and 40. Other matrices are available in other implementations. Perform the alignment a number of times, varying the scoring matrix every time:

- stick to the default BLOSUM62 matrix
- proteins have diverged some time ago (use a BLOSUM slightly lower than 62)
- proteins have diverged a long time ago (use the lowest BLOSUM available)

For proteins that diverged very recently use a BLOSUM higher than 62.

**Note:** `'Needle'` does not offer matrices higher than BLOSUM62

Have a look at the resulting alignments (percent identity, percent similarity, percent gaps, score).

Note that each result of `'needle'` tells you exactly, what you have done, thus making it easy to reproduce these results.

Next, repeat the exercise with the human hemoglobin (GI:4504347) and a worm globin (GI:17541936).

A4.2:

Mouse/human (diverged 100 million years ago)

# Matrix: EBLOSUM62

# Identity: 123/142 (86.6%)

# Similarity: 131/142 (92.3%)

# Gaps: 0/142 ( 0.0%)

# Matrix: EBLOSUM50

# Identity: 123/142 (86.6%)

# Similarity: 131/142 (92.3%)

# Gaps: 0/142 ( 0.0%)

# Matrix: EBLOSUM40

# Identity: 123/142 (86.6%)  
 # Similarity: 135/142 (95.1%)  
 # Gaps: 0/142 ( 0.0%)

a), b), c) do not really differ because our sequences can be aligned very robustly. Note that BLOSUM62 considers an Alanine (A) - Glycine (G) pair similar '.' whereas BLOSUM40 considers this very similar '..'.

Human/worm (diverged 1000 million years ago)

# Matrix: EBLOSUM62  
 # Identity: 29/370 ( 7.8%)  
 # Similarity: 49/370 (13.2%)  
 # Gaps: 256/370 (69.2%)  
 # Matrix: EBLOSUM50  
 # Identity: 33/372 ( 8.9%)  
 # Similarity: 57/372 (15.3%)  
 # Gaps: 260/372 (69.9%)  
 # Matrix: EBLOSUM40  
 # Identity: 33/372 ( 8.9%)  
 # Similarity: 64/372 (17.2%)  
 # Gaps: 260/372 (69.9%)

Notice that the position of gaps in the different alignments changes. The lowest BLOSUM available (BLOSUM40) gives the best results as humans and worms diverged much longer ago than humans and mice.

## 5. Sequence databases and statistical significance of alignments

### Tip

It is advisable to search databases of the highest quality available and to restrict the search, if possible, just to the data you are interested in.

Let's say, for example, that we want to find human proteins similar to a mouse hemoglobin. We would then restrict the search to *H. sapiens* and use a high quality, manually curated, protein database like SWISS-PROT.

For DNA sequences I would suggest to leave out low quality high throughput sequences like ESTs and first try with a database like NR, which is non-redundant (i.e. it doesn't contain duplicate sequences).

## 6. BLAST

At <http://www.ncbi.nlm.nih.gov/BLAST/> you will find a variety of BLAST programs. We can distinguish between BLAST for assembled genomes, basic BLAST and specialized BLAST.

Q6.1:



Use mouse cytochrome c protein (GI: 6681095) and search SWISS-PROT for related sequences. Report sequence similarity between mouse and chicken cytochrome c.

A6.1:

On the main BLAST page select 'protein blast' from the 'Basic BLAST' section. The BLASTP page opens up. You can run BLAST with as little or as much customization as you want. Each option is explained (click on the little question marks). Enter the GI number, select the database (SWISS-PROT) and make sure blastp is the selected program:

The screenshot shows the NCBI BLASTP search interface. The 'Enter Query Sequence' field contains the accession number '6681095'. The 'Database' dropdown is set to 'Swissprot protein sequences (swissprot)'. The 'Algorithm' dropdown is set to 'blastp (protein-protein BLAST)'. The 'BLAST' button is highlighted in yellow. A note at the bottom states: 'Note: Parameter values that differ from the default are highlighted in yellow'.

Notice that parameters that differ from the default (in our case the database choice) are highlighted in yellow and will be remembered for your next database search.

Hit the 'BLAST' button to start the database search. You will get a confirmation message, which gets refreshed in regular intervals till the job is completed. Once the search is finished you get presented with a graphical overview of the results. This graphic is followed by a table which contains the same information. The sequence names are linked to their database entries, the scores are linked to the respective alignment with the query sequence, and the little coloured boxes at the end of the lines are links to related entries in other databases. As cytochrome c is a well conserved protein we get lots of very good hits covering the complete length of our query sequence. A perfect alignment has a high score, a low E-value and covers the complete query sequence.



Find chicken cytochrome c:

```
>sp|P67881|CYC_CHICK Cytochrome c
Length=105

Score = 180 bits (456), Expect = 1e-45, Method: Composition-based stats.
Identities = 97/105 (92%), Positives = 101/105 (96%), Gaps = 0/105 (0%)

Query 1   MGDVEKGKKIFVQKCAQCHTVEKGGKHKHTGPNLHGLFGRKTGQAAGFSYTDANKNKGITW 60
          MGD+EKGKKIFVQKC+QCHTVEKGGKHKHTGPNLHGLFGRKTGQA GFSYTDANKNKGITW
Sbjct 1   MGDIEKGKKIFVQKCSQCHTVEKGGKHKHTGPNLHGLFGRKTGQAEGFSYTDANKNKGITW 60

Query 61  GEDTLMEYLENPKKYIPGTKMIFAGIKKKGERADLIAYLKATNE 105
          GEDTLMEYLENPKKYIPGTKMIFAGIKKK ER DLIAYLK AT++
Sbjct 61  GEDTLMEYLENPKKYIPGTKMIFAGIKKKSERVDLIAYLKDATSK 105
```

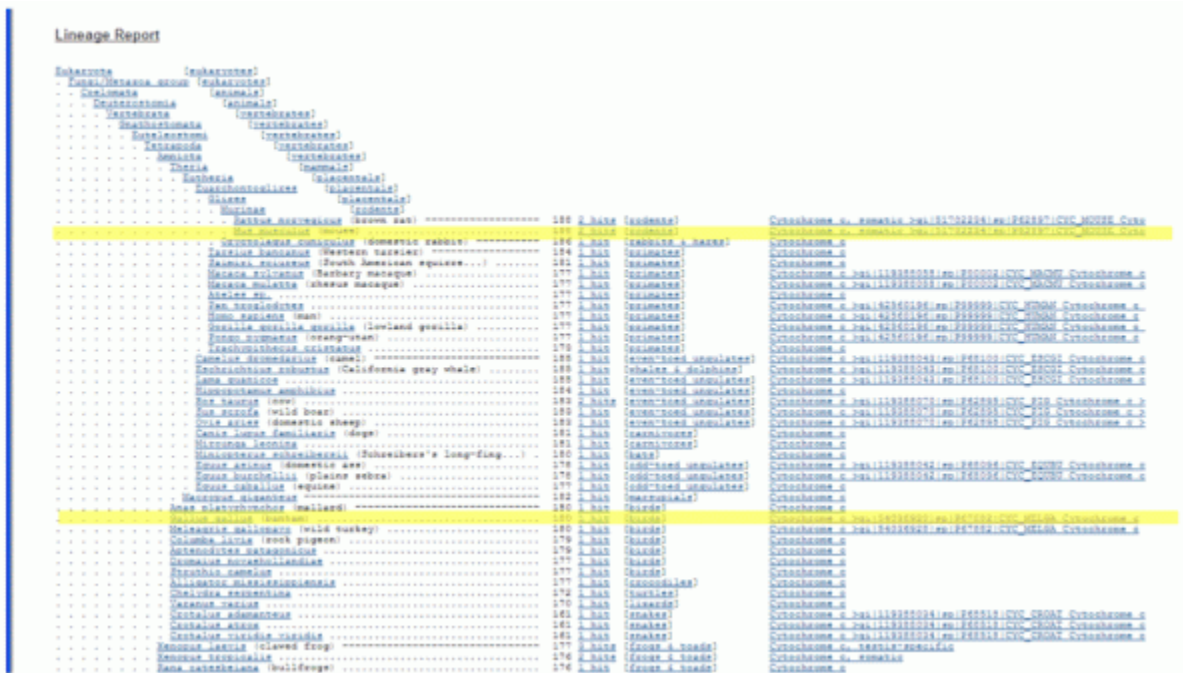
The two sequences are 96% similar.

Other options to explore on the BLAST results page:

Other options to explore on the BLAST results page:

- Taxonomy reports

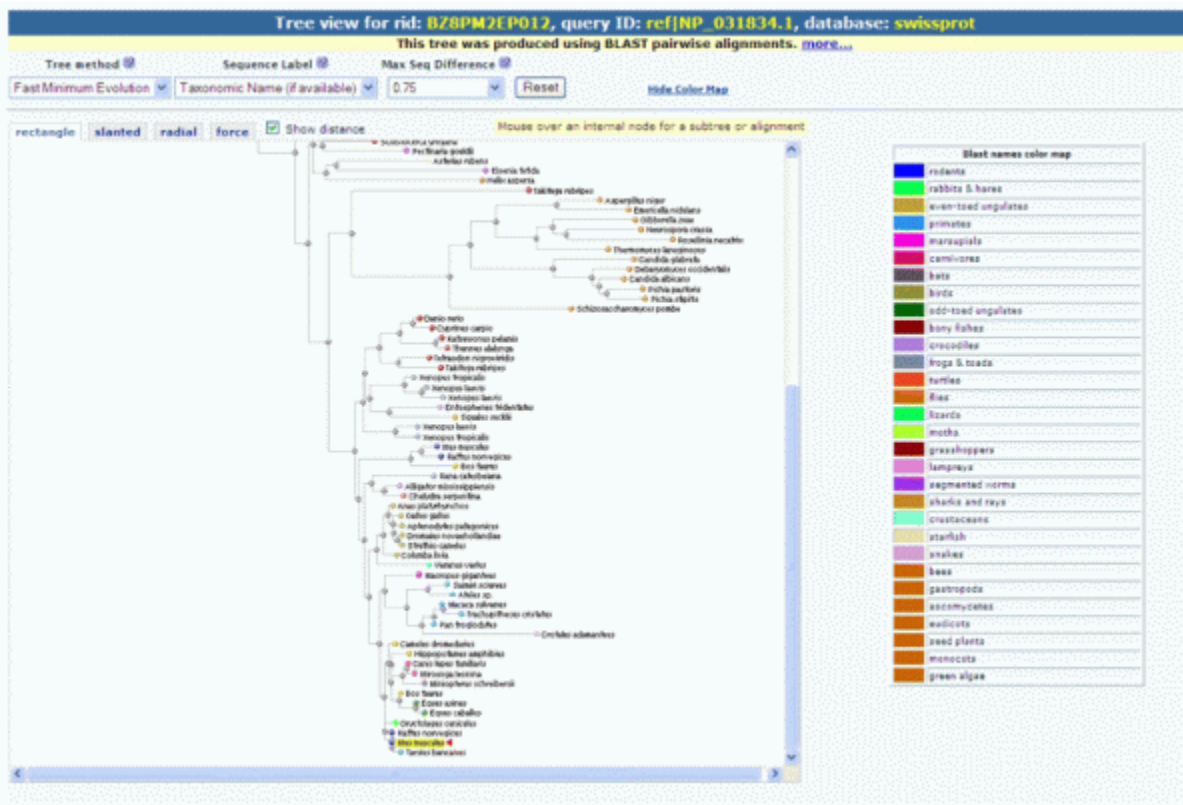
Click on 'Taxonomy reports' to see how the database hits are distributed across different lineages, organisms, or taxa.



Mouse and chicken have been manually highlighted yellow.

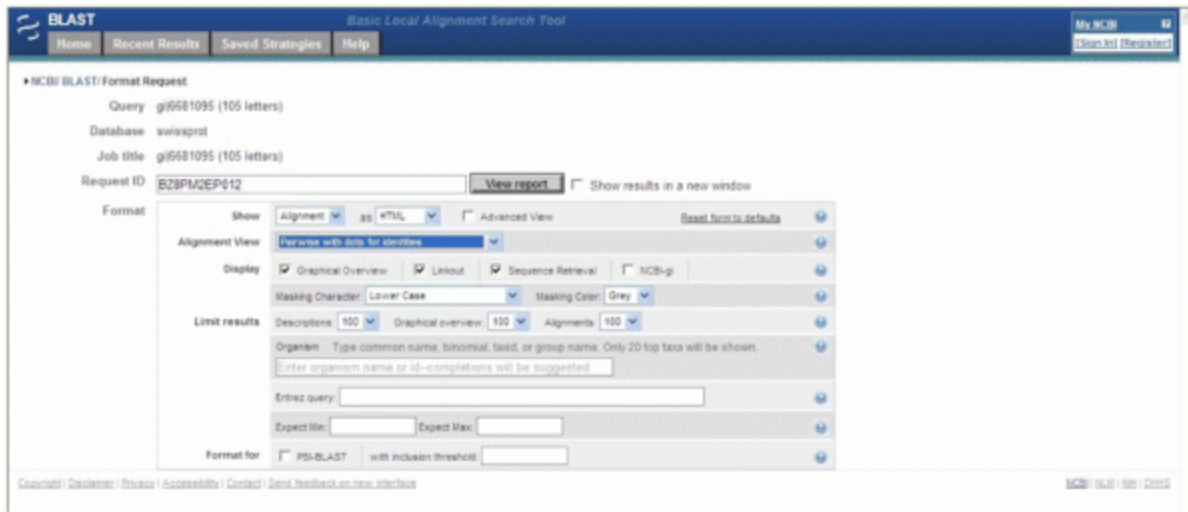
b. Distance tree of results

Click on 'Distance tree of results' to see how similar the different database hits are to each other:



c. Choose a different format for the results

To see only differences between your query sequence and database sequences, click on 'Reformat these Results'. On the new page choose 'Pairwise with dots for identities' as the 'Alignment view':



The result is:

```

Length=105

Score = 180 bits (456), Expect = 1e-45, Method: Composition-based stats.
Identities = 97/105 (92%), Positives = 101/105 (96%), Gaps = 0/105 (0%)

Query 1  MGDVEKGKKIFVQKCAQCHTVEKGGKHKHTGPNLHGLFGRKTGQAAGFSYTDANKNKGITW 60
Sbjct 1  ...I.....S.....E.....60

Query 61  GEDTLMYELENPKKYIPGTKMIFAGIKKKGERADLIAYLKKATNE 105
Sbjct 61  .....S..V.....D..SK 105

```

## Tip

When you look at the various alignments a BLAST search produces,

first	
	look at the statistical significance (E-value)
then	
	at the score
then	
	at the percent identity
and finally	
	at the distribution of identical residues.

## blast2seq

Q6.2:

Use our hemoglobin sequences from above (GI:4504347, GI:145301578) and find the right BLAST flavour (<http://www.ncbi.nlm.nih.gov/BLAST/>) to do a pairwise alignment of them.

Does the resulting alignment differ from the alignments we got using 'needle'/'water'?

A6.2:

The option to use is 'Align two sequences using BLAST (bl2seq)' (check the 'Specialized BLAST' section at the bottom of the main BLAST page). As a rule, blast2seq will return results very similar to 'water', since they are both performing local alignments. In our particular case, the result will also be similar to that of needle, even though this last tool performs a global alignment. This is due to the fact that our sequences are very similar over their entire length, making the "global" and the "local" alignments alike.

## Word size

Q6.3:

The 'word size' parameter determines the sensitivity of your similarity search. BLAST has a quick look at your query sequence and each database sequence. They have to have at least a stretch of sequence of 'word size' in common (100% identity) to be considered for an alignment.

Your research found the sequence TTCTGA to be very interesting. Let's see if 'word size' influences if BLAST finds this sequence. Use pen and paper and write down all BLAST words of length 11 for the following query sequence: **CACTTCTGATTCTGA**

A6.3:

**CACTTCTGATTCTGA** is composed of the following 11 letter words:

CACTTCTGATT  
 ACTTCTGATTC  
 CTTCTGATTCT  
 TTCTGATTCTG  
 TCTG**ATTCTGA**

Our sequence of interest (**TTCTGA**) is only 6 nucleotides long. BLAST will only find it if your query sequence and a database sequences have the above word **TCTGATTCTGA** in common. If the database sequence reads e.g. **TCTGGTTCTGA** this will not be the case despite the fact that the sequences have **TTCTGA** in common.

Q6.4:

Now decrease the word size to 6 and write down all the words.

A6.4:

**CACTTCTGATTCTGA** is composed of the following 6 letter words:

CACTTC  
 ACTTCT  
 CTTCTG  
 TTCTGA  
 TCTGAT  
 CTGATT  
 TGATTC  
 GATTCT  
 ATTCTG  
**TTCTGA**

This time you can be guaranteed that if your query sequence and a database sequence have **TTCTGA** in common it will be found.

Note: The more sensitive (i.e. the smaller **'word size'**) a search is the longer it will take to finish. The BLAST variant megablast uses a **'word size'** of 28 to increase its speed. Megablast is used for highly similar sequences and can therefore compromise some sensitivity. As a rule of thumb: do not use a **'word size'** bigger than the smallest feature you are looking for.

### Bit score, raw score, E-value

Q6.5:

You've been reported bit score, raw score, E-value for the following 5 alignments.

Score = 155 bits (393), Expect = 4e-37

Identities = 79/150 (52%), Positives = 101/150 (67%), Gaps = 1/150 (0%)

Score = 74.3 bits (181), Expect = 2e-12

Identities = 36/105 (34%), Positives = 57/105 (54%), Gaps = 0/105 (0%)

Score = 347 bits (889), Expect = 1e-94

Identities = 167/167 (100%), Positives = 167/167 (100%), Gaps = 0/167 (0%)

d). Score = 189 bits (480), Expect = 3e-47

Identities = 88/151 (58%), Positives = 114/151 (75%), Gaps = 1/151 (0%)

Score = 89.7 bits (221), Expect = 4e-17

Identities = 52/127 (40%), Positives = 68/127 (53%), Gaps = 5/127 (3%)

Sort them by relevance as BLAST would do. Using the rule of thumb, which alignments are significant?

A6.5:

The BLAST order would be c), d), a), e), b).

All alignments above are significant and worthwhile assessing the degree of similarity and the possibility of homology.

Remember the rule of thumb:

- E-value smaller than  $e^{-5}$  are usually significant.
- We are unsure if the e-value is between  $e^{-5}$  and  $e^{-1}$ .
- E-values higher than 0.1 are usually not significant.

## Filtering

Q6.6:

You are looking for all similar nucleic sequences in human. Choose the correct BLAST flavour (<http://www.ncbi.nlm.nih.gov/BLAST/>) and search the following sequence against 'human genomic plus transcript':

```
>gi|1036804|gb|U36764.1|HSU36764 Human TGF-beta receptor interacting
protein 1 mRNA, complete cds
GGCACGAGGTTGCGGCCTTCCTCGCGTCACCGCCGGGATGAAGCCGATCCTACTGCAGGGCCATGAGCGG
TCCATTACGCAGATTAAGTATAACCGCGAAGGAGACCTCCTCTTACTGTGGCCAAGGACCCTATCGTCA
ATGTATGGTACTCTGTGAATGGTGAGAGGCTGGGCACCTACATGGGCCATACCGGAGCTGTGTGGTGTGT
GGACGCTGACTGGGACACCAAGCATGTCCTCACTGGCTCAGCTGACAACAGCTGTCGTCTCTGGGACTGT
GAAACAGGAAAGCAGCTGGCCCTTCTCAAGACCAATTTCGGCTGTCCGGACCTGCGGTTTTGACTTTGGGG
GCAACATCATCATGTTCTCCACGGACAAGCAGATGGGCTACCAGTGCTTTGTGAGCTTTTTGACCTGCG
GGATCCGAGCCAGATTGACAACAATGAGCCCTACATGAAGATCCCTTGCAATGACTCTAAAATCACCAGT
GCTGTTTGGGGACCCCTGGGGGAGTGCATCATCGCTGGCCATGAGAGTGGAGAGCTCAACCAGTATAGTG
CCAAGTCTGGAGAGGTGTTGGTGAATGTTAAGGAGCACTCCCGGCAGATCAACGACATCCAGTTATCCAG
GGACATGACCATGTTTGTGACCGCGTCCAAGGACAACACAGCCAAGCTTTTTGACTCCACAACCTTTGAA
CATCAGAAGACTTTCCGGACAGAACGTCCTGTCAACTCAGCTGCCCTCTCCCCAACTATGACCATGTGG
TCCTGGGCGGTGGTCAGGAAGCCATGGATGTAACCACAACCTCCACCAGGATTGGCAAGTTTGAGGCCAG
GTTCTTCCATTTGGCCTTTGAAGAAGAGTTTGAAGAGTCAAGGGTCACTTTGGACCTATCAACAGTGTT
GCCTTCCATCCTGATGGCAAGAGCTACAGCAGCGGCGGAAGATGGTTACGTCCGTATCCATTACTTCG
ACCCACAGTACTTCGAATTTGAGTTTGAAGGCTTAAGAAGCTGGATCTCCTGCCGGGCGTGGTGGCTCATG
CCTGTAATCCCACCACTTTTTTTAAGGCAGGCGGATCACCTGAGGTCAGGAGTTAAGACCAGCCTGAC
CAACATGGAGAAACTCGTCTCTACTAAAAATACAAAAATACAAAAATTAGCCAGGCATGGTGGCACACGC
CTATAGTCCCAGCTACTCAGGAGGCTGAGGCAGGAGAATCACTTGAACCCAGGAGGCATAGGTTGCAGTG
AGCTGAGATCACGTCATTGCACTCCATCCTGAGCCACAAGAGCAAACTCCGTCTCAAAAAAAAAAAAA
```

- Switch all filtering options on
- Switch filtering options off

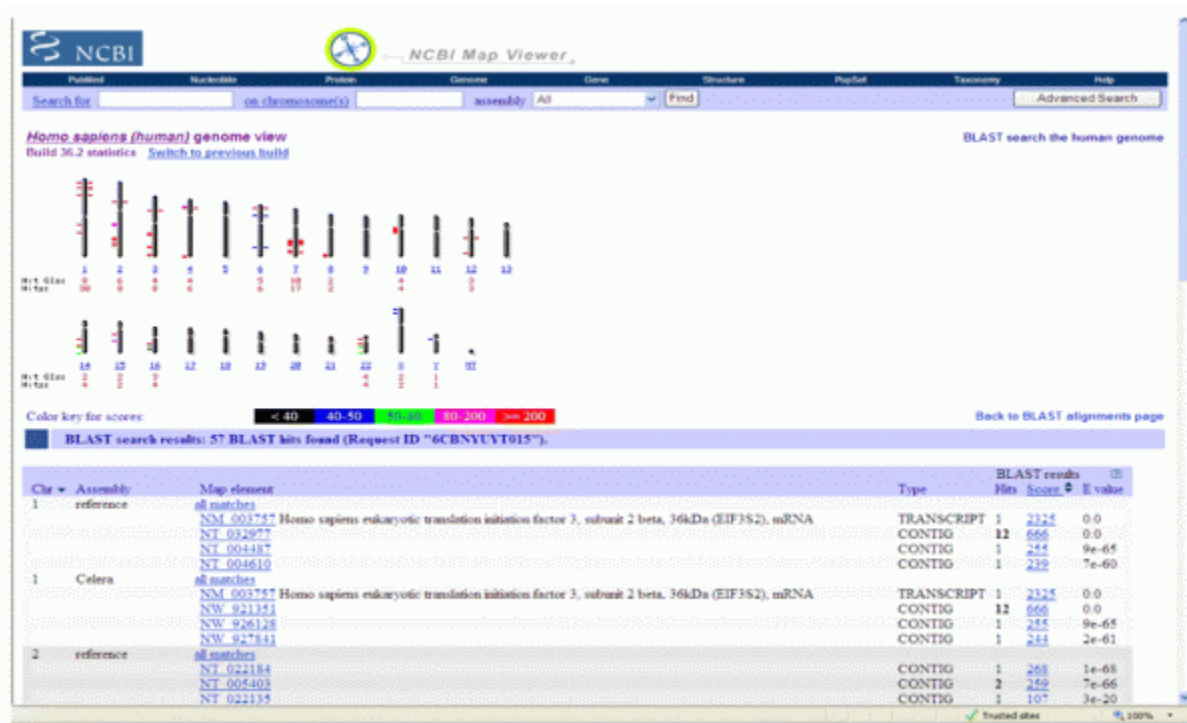
How do the results differ?

A6.6:

Choose BLAST against the assembled human genome. Then choose megablast for a quick overview or blastn for higher sensitivity.

Our query sequence contained some low complexity regions. We are normally not interested in these repeats but rather in sequences that code for specific functions. With filtering switched on we concentrate on functional similarity, whereas with filtering switched off we concentrate on finding other sequences with repeats.

a) BLAST reports some hits. Click on the 'Genome View' button to see how the database hits are distributed across the human genome:



After some time you will get:

'ERROR: An error has occurred on the server, Too many HSPs to save all'.

Filtering allowed us to get a manageable number of database hits.

## BLASTN

Q6.7:

Search (<http://www.ncbi.nlm.nih.gov/BLAST/>) the NR database for similar sequences to the mouse hemoglobin sequence. What do you get reported? At what e-value would you draw a line between significant and non-significant hits?

A6.7:

We hit some sequences that are described as hemoglobin. Most sequences are clones without functional description. At least the first 100 hits are relevant. Their E-value is below  $e^{-5}$  and they all cover more than 100 residues.

Tip

Be very careful when you get hits on multiple chromosomes of the same genome as this normally doesn't indicate functional relationship.

Q6.8:

In a book you read about the reconstruction of extinct species. The author gives a DNA sequence and claims that it is from a mammoth. Please check (<http://www.ncbi.nlm.nih.gov/BLAST/>) the following sequence:



```
TCGCGCGTTTTTCGGTGATGACGGTGAAAACCTCTGACACATGCAGCTCCCGGAGACGGTCACAGCTTGTCT
GTAAGCGGATGCCGGGAGCAGACAAGCCCGTCAGGGCGCGTCAGCGGGTGTGGCGGGTGTTCGGGGCTGG
CTTAACTATGCGGCATCAGAGCAGATTGTAAGTGCAGAGTGCACCATATGCGGTGTGAAATACCGCACAGAT
GCGTAAGGAGAAAATACCGCATCAGGCGCCATTCGCCATTCAGGCTGCGCAACTGTTGGGAAGGGCGATC
GGTGCGGGCCTCTTCGCTATTACGCCAGCTGGCGAAAGGGGGATGTGCTGCAAGGCGATTAAGTTGGGTA
ACGCCAGGGTTTTCCAGTCACGACGTTGTAACACGACGGCCAGTGCCAAGCTTGCATGCCTGCAGGTCG
ACTCTAGAGGATCCCCGGGTACCGAGCTCGAATTCGTAATCATGGTCATAGCTGTTTCCCTGTGTGAAATT
GTTATCCGCTCACAATTCACACAACATACGAGCCGGAAGCATAAAGTGTAAGCCTGGGGTGCCTAATG
AGTGAGCTAACTCACATTAATTGCGTTGCGCTCACTGCCCCTTCCAGTCGGGAAACCTGTCGTGCCAG
CTGCATTAATGAATCGGCCAACGCGCGGGGAGAGGCGGTTTGCCTATTGGGCGCTCTTCCGCTTCCCTCGC
TCACTGACTCGCTGCGCTCGGTCGTTTCGGCTGCGGCGAGCGGTATCAGCTCACTCAAAGGCGGTAATACG
GTTATCCACAGAATCAGGGGATAACGCAGGAAAGAACATGTGAGCAAAGGCCAGCAAAGGCCAGGAAC
CGTAAAAGGCCGCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCTGACGAGCATCACAAAATCGAC
GCTCAAGTCAGAGGT
```

A6.8:

The author must have made a mistake as only cloning vectors get reported as hits.

## BLASTP

Q6.9:

Search (<http://www.ncbi.nlm.nih.gov/BLAST/>) SWISS-PROT for similar sequences to the mouse hemoglobin sequence. What do you get reported? At what e-value would you draw a line between significant and non-significant hits?

A6.9:

You get back lots of other hemoglobin sequences. At least the first 500 hits are relevant. Their E-value is below  $e^{-5}$  and they all cover more than 50 residues.

Tip

Look for matches which cover structurally relevant distances. As a rule of thumb domains are 50 residues in size.

## BLASTX

Q6.10:

You obtained the following DNA sequence:

```
CTTGATCGAGGCTGCCAACACATTGATGACATCGCCGGCACCCCTCTCCAAGCTCAGCGACCTCCATGCC
CACAAGCTCCGCGTGGACCCTGTCAACTTCAAAGTGAGTGTCTGGGAAGGGGTGACCCGCCCTGCCCCCA
CCGAGCCCCGTCTTGGGCCATGCGGCCACCCCTCGCCTCACCCCTCGCTCATCCTCTCCTTTGCCTT
GCAGCTCCTGGGCCAATGCTTCTGGTGGTGGTGGCCATC
```

You are interested in similar sequences that have a high-quality annotation. You decide to give SWISS-PROT a try. Which is the appropriate flavour of BLAST (<http://www.ncbi.nlm.nih.gov/BLAST/>)? What function does our DNA sequence have?

A6.10:

From the 'basic BLAST' section choose BLASTX, as it translates your sequence in all six reading frames, looking for similarities at the amino-acid level. The sequence is a chicken hemoglobin sequence.

## TBLASTN

Q6.11:

You managed to sequence the mouse hemoglobin protein. Now you want to know on which chromosome the gene coding for this protein is located. Use TBLASTN from the 'Basic BLAST' section at <http://www.ncbi.nlm.nih.gov/BLAST/> to compare your protein (GI:145301578) against NR translated in all six reading frames. Where is the mouse hemoglobin gene located?

A6.11:

When you click on the small blue box containing a G letter, just after the E-value of the first hit, you get information about the genomic location, which in this case is chromosome 11.

## Protein vs. DNA sequences

In all these searches keep in mind that DNA sequences are less conserved than amino acid sequences, which in turn are less conserved than the structures of the corresponding proteins.

If you want to predict the function of a sequence and you know the open reading frame, use the encoded protein sequence for similarity searches. If you are interested in synonymous (silent) and non-synonymous mutations use the DNA sequences.

## MegaBLAST

Q6.12:

We know that besides the mouse hemoglobin nucleotide sequence (GI:6680174), there are more hemoglobins in the NR database. To quickly pull them out we use MegaBLAST (<http://www.ncbi.nlm.nih.gov/BLAST/>). Search NR for similar sequences to the mouse hemoglobin mRNA sequence. What do you get reported? At what e-value would you draw a line between significant and non-significant hits?

A6.12:

We get lots of hemoglobin sequences from different species reported. At least the first 100 hits are relevant. Their E-value is below  $e^{-5}$  and they all cover more than 100 residues.

## PSIBLAST

Q6.13:

You wonder what other protein families are closely related to hemoglobins. To find out you decide to use PSI-BLAST with mouse haemoglobin as query sequence (select 'protein blast' from the 'Basic BLAST' section, then select PSI-BLAST on the new webpage). PSI-BLAST performs a normal BLAST to find sequences similar to your query sequence (mouse hemoglobin) and combines them into a broader, more flexible description (hemoglobin protein family). It then uses this broader description to search for related sequences. This process can be

repeated as often as you like (or until no additional new related sequences can be found). Which families seem to be closely related to hemoglobins?

A6.13:

The first iteration of PSI-BLAST is a normal BLAST and reports exclusively hemoglobins. Click on 'Run PSI-Blast iteration 2' to summaries the hits and use this summary (also called position-specific scoring matrix) for the next round of searches. The second iteration reports some new sequences (yellow 'NEW' tag). When having a look at them you will see that there are now also myoglobins, cytoglobins, and globins included.

Note: Keep in mind that incorporation of unrelated sequences contaminates the results and will lead to more and more unrelated sequences being found.

## **BLAST for primers**

Q6.14:

You found the optimal set of primers for a PCR experiment:

LEFT PRIMER AAGGAAAAGGCCAGTAAGGC

RIGHT PRIMER CTCCCAAGCCTCCTGTAG

To make sure they only prime at the desired site you decide to check that.

Which human gene are we working on? Are you happy with the specificity of the primers?

A6.14:

Tip

BLAST searches both strands for the best regions of similarity. Therefore, we do not have to perform two searches with our primer pair, but can simply concatenate the forward and reverse primer and check them in a single search:

>primer

AAGGAAAAGGCCAGTAAGGCCTCCCAAGCCTCCTGTAG

Go to the BLAST program selection guide. To do that click 'Help' on the main BLAST page:

BLAST: Basic Local Alignment and Search Tool

File Edit View Go Bookmarks Tabs Help

Back Forward Stop Reload Home History Bookmarks Smaller Larger

http://www.ncbi.nlm.nih.gov/BLAST/

BLAST Home Recent Results Saved Strategies **Help** My NCBI [Sign In] [Register]

NCBI/BLAST Home

BLAST finds regions of similarity between biological sequences. [more...](#)

[Learn more](#) about how to use the new BLAST design.

**BLAST Assembled Genomes**

Choose a species genome to search, or [list all genomic BLAST databases](#).

- Human
- Mouse
- Rat
- Arabidopsis thaliana*
- Oryza sativa*
- Ros taurus*
- Danio rerio*
- Drosophila melanogaster*
- Gallus gallus*
- Pan troglodytes*
- Microbes
- Apis mellifera*

**Basic BLAST**

Choose a BLAST program to run.

<a href="#">nucleotide blast</a>	Search a <b>nucleotide</b> database using a <b>nucleotide</b> query Algorithms: <a href="#">blastn</a> , <a href="#">megablast</a> , <a href="#">discontiguous megablast</a>
<a href="#">protein blast</a>	Search <b>protein</b> database using a <b>protein</b> query Algorithms: <a href="#">blastp</a> , <a href="#">psi-blast</a> , <a href="#">phi-blast</a>
<a href="#">blastx</a>	Search <b>protein</b> database using a <b>translated nucleotide</b> query
<a href="#">tblastn</a>	Search <b>translated nucleotide</b> database using a <b>protein</b> query
<a href="#">tblastx</a>	Search <b>translated nucleotide</b> database using a <b>translated nucleotide</b> query

**Specialized BLAST**

Choose a type of specialized search (or database name in parentheses.)

- Search [trace archives](#)
- Find [conserved domains](#) in your sequence (cds)
- Find sequences with similar [conserved domain architecture](#) (cdart)
- Search sequences that have [gene expression profiles](#) (GEO)
- Search [immunoglobulins](#) (IgBLAST)
- Search for [SNPs](#) (snp)

Enter a web address to open, or a phrase to search for

News

**New Human and Mouse pre-indexed databases**

megablast  
2007-09-04 10:55:00

[More BLAST news...](#)

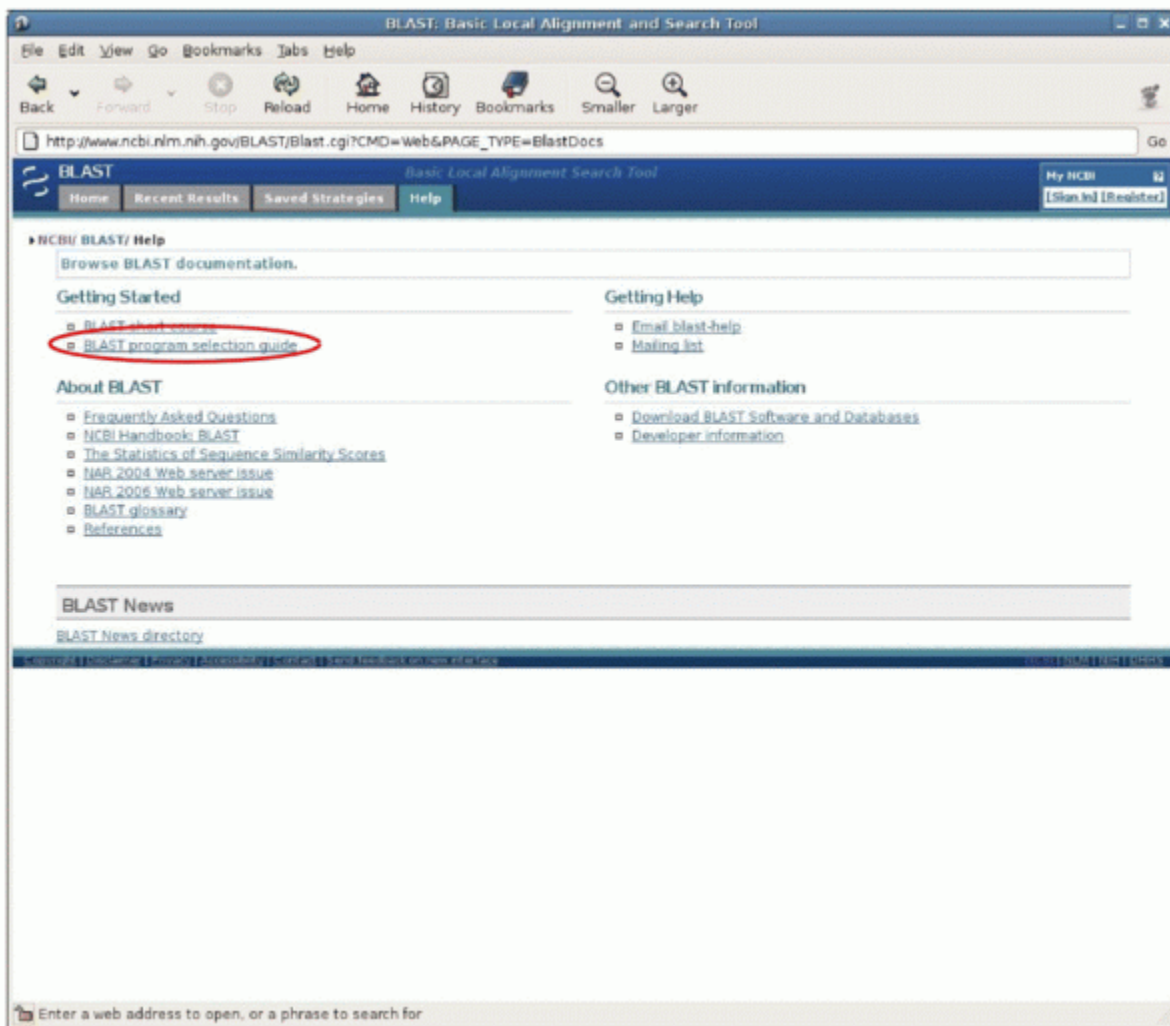
Tip of the Day

**How to save custom search pages.**

So you have made a few BLAST searches and after adjusting the database, organism limits and maybe a few Algorithm Parameters you arrive at what you think is a good search strategy. Do you want to have to fiddle with pull down menus or remember all the changes you made the next time to want to run a similar search? Now you can use "Saved Strategies" to and always have a saved search template.

[Here lies...](#)

Then select 'BLAST program selection guide':



Then, follow the "Program selection table" link:

BLAST program selection guide

File Edit View Go Bookmarks Tabs Help

Back Forward Stop Reload Home History Bookmarks Smaller Larger

http://www.ncbi.nlm.nih.gov/blast/producttable.shtml

NCBI – BLAST Latest news: New BLAST design to be released on April 16, 2007

**BLAST Program Selection Guide**  
User Service  
NCBI, NLM, NIH  
8500 Rockville Pike, Bethesda, MD 20894

**Table of Content**

1. [Introduction](#)
2. [BLAST Database Content](#)
3. [Program Selection Table](#)
4. [Explanation for the program choices given in Tables 3.1 and 3.2](#)
5. [Explanation for the program choices given in Tables 3.3](#)
6. [Explanation on Special Purpose Pages](#)
7. [Appendices](#)

**1. Introduction**

NCBI has provided BLAST sequence analysis services for over a decade. For many users, the first question they often face is *'Which BLAST program should I use?'* In order to help users arrive at an answer to this question, we created this "BLAST Program Selection Guide."

This document first introduces the BLAST databases available from NCBI (in Section 2). The actual guide (Section 3) divides BLAST searches into several categories according to the *nature* and *size* of the input query and the primary goal of the search. Starting from the query sequence column on the left and cross-referencing to the right, a user will arrive at the specific BLAST program(s) best suited for that search.

This document is also available in [PDF](#) (163,516 bytes).

**2. BLAST Database Content**

A BLAST search has four components: query, database, program, and search purpose/goal. To discuss effective BLAST program selection, we first need to know what databases are available and what sequences these databases contain. In this section, we will first take a look at the common BLAST databases. According to their content, they are grouped into nucleotide and protein

The 'NUCLEOTIDE' table lists 'Find primer binding sites or map short contiguous motifs'. Click on 'Search for short.nearly exact matches'.

**3. Program Selection Tables**

The appropriate selection of a BLAST program for a given search is influenced by the following three factors **1)** the nature of the query, **2)** the purpose of the search, and **3)** the database intended as the target of the search and its availability. The following tables provide recommendations on how to make this selection.

Length <sup>1</sup>	Database	Purpose	Program	Explanation
20 bp or longer 28 bp or above for megablast	Nucleotide	Identify the query sequence	<a href="#">discontiguous megablast</a> , <a href="#">megablast</a> , or <a href="#">blastn</a>	<a href="#">Learn more</a> ...
		Find sequences similar to query sequence	<a href="#">discontiguous megablast</a> or <a href="#">blastn</a>	<a href="#">Learn more</a> ...
	Peptide	Find similar sequence from the Trace archive	<a href="#">Trace megablast</a> , or <a href="#">Trace discontiguous megablast</a>	<a href="#">Learn more</a> ...
		Find similar proteins to translated query in a translated database	<a href="#">Translated BLAST (tblastx)</a>	<a href="#">Learn more</a> ...
7 - 20 bp	Nucleotide	Find similar proteins to translated query in a protein database	<a href="#">Translated BLAST (tblastx)</a>	<a href="#">Learn more</a> ...
		Find primer binding sites or map short contiguous motifs	<a href="#">Search for short, nearly exact matches</a>	<a href="#">Learn more</a> ...

NOTE:  
<sup>1</sup> The cut-off is only a recommendation. For short queries, one is more likely to get matches if the

A BLASTN window opens. To speed up the search you can restrict yourself to human sequences ('Human genomic + transcript'). Note that as adjustment to short query sequences the expectation value is increased and the word size is decreased. In the results window check for hits covering the complete query sequence. You can see that your complete primer sequence gets reported as forward and reverse primer together with flanking features (target gene): fibroblast growth factor. Our target gene is on chromosome 12 and only here do both primers bind sufficiently close together.

### Tip

Check where your results are positioned on the human genome by clicking the 'Genome View' button (just above the BLAST graphical results). To find the correct hits check for high scoring hits (use the color key or the table of results).

## More BLAST

For more information and exercises check the following resources (in that order):

Blast Quick Start (click on P or H for the self scoring exercises) (<http://www.ncbi.nlm.nih.gov/Class/minicourses/blastex2.html>)

Blast information guide (<http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/information3.html>)

The Statistics of Sequence Similarity Scores (<http://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-1.html>)

## More Tips

### Tip

For more complicated cases you should always use a specialized alignment tool and not a database search tool like BLAST which is optimized for speed and not for accuracy.

### Tip

When reporting a very important result show the alignment which you have refined using an alignment tool as well as manual curation. All other information such as scores and E-values are just a more or less crude summary.

### Tip

If you want to keep updated on similar sequences for your query sequence you can sign up for the free service myNCBI (<http://www.ncbi.nlm.nih.gov/entrez/cubby.fcgi?call=QueryExt.CubbyQuery..ShowAll>).

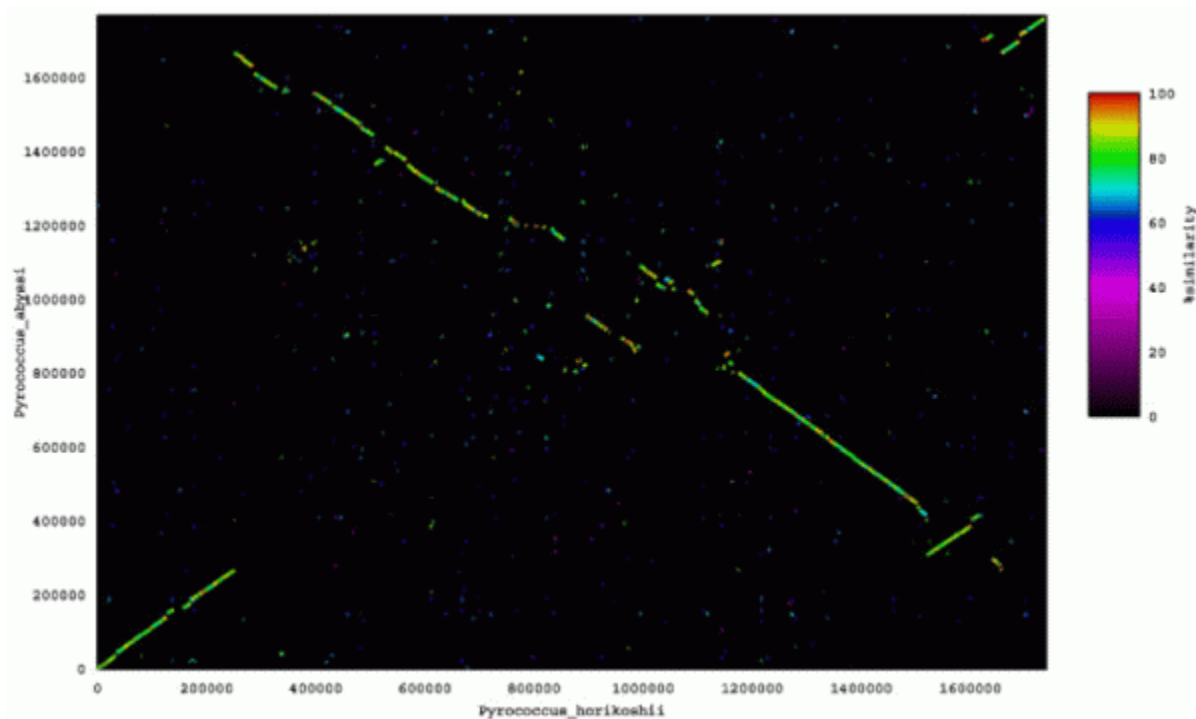
### Tip

You can restrict your results not just to an organism, but to any dataset you would like. Just use NCBI's search tool Entrez to specify what you are looking for, and then choose 'Limit by entrez query' from the advanced BLAST options. Click on the link to get instructions as to how exactly this works.

## 7. Sequence comparison for very large sequences

To perform rapid whole genome alignments, one can use Mummer. Mummer is not available as a web service, but can be downloaded for free from <http://sourceforge.net/projects/mummer>. Note that it cannot be installed on Windows. Once installed on a UNIX-like operating system you simply type in: `mummer sequence1 sequence2' to get two sequences aligned.

The following is a sample screenshot from Mummer's webpage and shows the comparison of two bacterial genomes:





## 8. Multiple sequence alignment

### Where to find ClustalW, T-coffee, MUSCLE

ClustalW can be used online at <http://www.ebi.ac.uk/clustalw/>. You can also download the software from the same site.

- Help Index
- General Help
- Formats
- Gaps
- Matrix
- References
- ClustalW Help
- ClustalW FAQ
- Jalview Help
- Scores Table
- Alignment
- Guide Tree
- Colours

---

- Similar Applications
  - ▶ Muscle
  - ▶ T-Coffee

**ClustalW**

**Submission Form**

ClustalW is a general purpose multiple sequence alignment program for DNA or proteins. It produces biologically meaningful multiple sequence alignments of divergent sequences. It calculates the best match for the selected sequences, and lines them up so that the identities, similarities and differences can be seen. Evolutionary relationships can be seen via viewing Cladograms or Phylograms. **New users, please read the FAQ.**

>> Download Software

YOUR EMAIL	ALIGNMENT TITLE	RESULTS	ALIGNMENT	CPU MODE
<input type="text" value=""/>	<input type="text" value="Sequence"/>	<input type="text" value="interactive"/>	<input type="text" value="full"/>	<input type="text" value="single"/>
KTUP (WORD SIZE)	WINDOW LENGTH	SCORE TYPE	TOPDIAG	PAIRGAP
<input type="text" value="def"/>	<input type="text" value="def"/>	<input type="text" value="percent"/>	<input type="text" value="def"/>	<input type="text" value="def"/>
MATRIX	GAP OPEN	END GAPS	GAP EXTENSION	GAP DISTANCES
<input type="text" value="def"/>	<input type="text" value="def"/>	<input type="text" value="def"/>	<input type="text" value="def"/>	<input type="text" value="def"/>

OUTPUT		PHYLOGENETIC TREE		
OUTPUT FORMAT	OUTPUT ORDER	TREE TYPE	CORRECT DIST.	IGNORE GAPS
<input type="text" value="aln w/numbers"/>	<input type="text" value="aligned"/>	<input type="text" value="none"/>	<input type="text" value="off"/>	<input type="text" value="off"/>

Enter or Paste a set of Sequences in any supported format: Help

Upload a file:

**If you plan to use these services during a course please contact us using the email below. Please read the [FAQ](#) before seeking help from our support staff.**

T-coffee can be used online at <http://www.ch.embnet.org/software/TCoffee.html> :

**ch.EMBnet.org**

Home Services Courses Links Contacts

## T-COFFEE

This program is more accurate than ClustalW for sequences with less than 30% identity, but it is slower...

For any question please contact the author: [Cedric Notredame](#)

Please cite:  
"T-Coffee: A novel method for multiple sequence alignments"  
C. Notredame, D. Higgins, J. Heringa  
*Journal of Molecular Biology*, **302**, 205-217, (2000)

Valid format for input is: FASTA(Pearson)  
max number of sequences = 30  
max total length of sequences = 10000  
[Help page](#)

Output order: Input

Sequence type:  Protein  DNA

Input sequences:  
(see above for valid formats)

Run T-COFFEE Clear Input

MUSCLE can be used online at <http://www.bioinformatics.nl/tools/muscle.html>:

### Tip

From the MUSCLE webpage: "Evaluated on the BaliBASE benchmark (which includes 142 reference alignments and more than 1,000 sequences), MUSCLE delivers a higher score than T-Coffee (until now considered the most accurate technique) while requiring only 20 seconds of CPU time on a typical desktop computer. By comparison, T-Coffee requires 1,500 seconds. The popular ClustalW algorithm takes 170 seconds, but it generates a lower score (reflecting less accurate alignments) than either MUSCLE or T-Coffee." BALibase can be found at <http://www-igbmc.u-strasbg.fr/BioInfo/BaliBASE/>

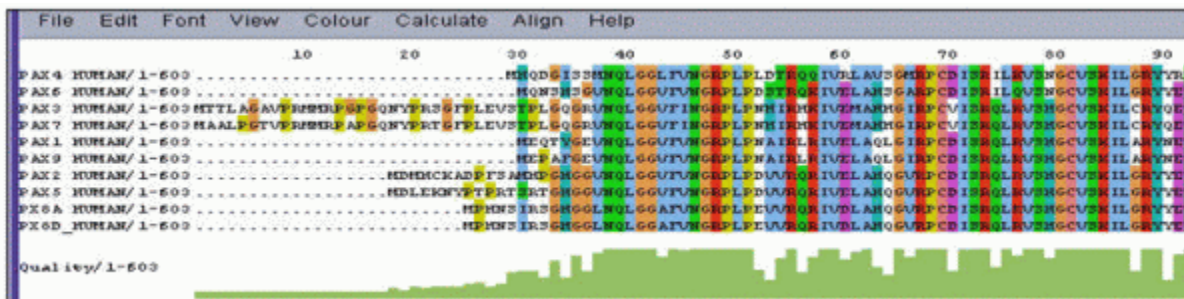
## How to edit alignments

Computers do not do biology. If you are the biological expert, you should use your expertise to correct mistakes. Often there are several possible alignments with the same score and you can make sure the correct one is chosen. A free of charge alignment editor is BioEdit (<http://www.mbio.ncsu.edu/BioEdit/bioedit.html>) which runs on Windows. A screenshot of the program interface is shown below:



BioEdit even allows you to toggle between DNA or amino acid sequence view.

Another free alignment editor is Jalview (<http://www.jalview.org/>), which runs on all platforms. One of the possible views looks like this and gives a quality measuring for each position within the alignment:



## Search with family descriptions (signatures, patterns, HMMS)

MuTiple sequences in an alignment can be regarded as family members. If database searches for other similar sequences are not satisfactory, one can summarize a family in a variety of ways. The simplest one is to create a consensus sequence by representing each column in the muTiple sequence alignment by its most frequent residue. To allow more flexibility as well as flexible distances between key residues, signatures/patterns can be described. To include the environment of each residue, Hidden Markow Models (HMMS) are used. For all these approaches specialized tools and databases are available. To get started visit <http://ca.expasy.org/> and check the 'Databases' and 'Tools and software packages' section.

## General Tips

- Remember the server, the database, and the program version you used (in case you or somebody else wants to reproduce the results).
- Write down the ID (identification) and AC (accession) number of your sequences.
- Write down program parameters. Often these can be found in the results of programs or in log files. If this is not the case, write them down or make a screenshot.
- Check results by using different tools which rely on different methods, change the program parameters. Artefacts tend to be not robust.
- Use up to date databases.
- Be aware that programs make mistakes and databases contain mistakes, therefore be an educated user.

## References

### Further Readings

Having a BLAST with bioinformatics (and avoiding BLASTphemy), *Genome Biology* 2001, 2(10):reviews2002.1–2002.10, (<http://genomebiology.com/2001/2/10/reviews/2002.1>).

Bioinformatics - A Beginner's Guide, JM Claverie and C Notredame, Wiley, USA, 2003.

### Bibliography

There are many sources that give additional details about sequence comparison algorithms. Below is a basic list of relevant general books that contain one or more chapters about sequence comparison. Some of these books were used to prepare this chapter.

The Smith-Waterman algorithm is described in reference 6:

The original BLAST papers are described in references 7 and 8:

Reference 9 is a whole book about BLAST

NCBI's website (<http://ncbi.nlm.nih.gov>) provides a wealth of information about BLAST.

The MUMmer program is described in reference 10

References 11-13 describe the mulTiple alignment programs mentioned in the text.

The primary references for PAM and BLOSUM amino acid scoring matrices, respectively, are #14 and #15

1. Setubal J, Meidanis J. *Introduction to Computational Molecular Biology*. PWS; 1997.
2. Gusfield D. *Algorithms on Strings, Trees, and Sequences*. Cambridge University Press; 1997.
3. Durbin R, Eddy S, Krogh A, Mitchison G. *Biological Sequence Analysis*. Cambridge University Press; 1998. [b7].
4. Jones N, Pevzner P. *An Introduction to Bioinformatics Algorithms*. MIT Press; 2004. [b8].
5. Mount D. *Bioinformatics*. 2nd edition. Cold Spring Harbor Laboratory Press; 2005. [b9].
6. Smith TF, Waterman MS. Identification of common molecular subsequences. *J Mol Biol.* 1981;147:195–197. PubMed. PubMed PMID: 7265238.
7. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. A basic local alignment search tool. *J Mol Biol.* 1990;215:403–410. PubMed. PubMed PMID: 2231712.
8. Altschul SF, Madden TL, Shaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* 1997;25:3389–3402. PubMed. PubMed PMID: 9254694.
9. Korff, I, Yandell, M, Blendel. J. *BLAST*. O'Reilly; 2003. [b10].

10. Kurtz S, Phillippy A, Delcher AL, Smoot M, Shumway M, Antonescu C, Salzberg SL. Versatile and open software for comparing large genomes. *Genome Biol.* 2004;5(2):R12. PubMed.
11. Chenna R, Sugawara H, Koike T, Lopez R, Gibson TJ, Higgins DG, Thompson JD. Multiple sequence alignment with the Clustal series of programs. *Nucleic Acids Res.* 2003;31(13):3497–3500. PubMed. PubMed PMID: 12824352.
12. Notredame C, Higgins DG, Heringa J. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *J Mol Biol.* 2000;302(1):205–217. PubMed. PubMed PMID: 10964570.
13. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.* 2004;32(5):1792–1797. PubMed. PubMed PMID: 15034147.
14. Dayhoff MO, Schwartz RM, Orcutt BC. A model of evolutionary change in proteins. *Atlas of Protein Sequence and Structure.* 1978;5:345–352. [b11].
15. Henikoff S, Henikoff JG. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A.* 1992;89:10915–10919. PubMed.[b12]. PubMed PMID: 1438297.

## Chapter A06. Protein Structure, Modelling and Applications

Ardala Breda,<sup>1</sup> Napoleão Fonseca Valadares,<sup>2</sup> Osmar Norberto de Souza,<sup>3</sup> and Richard Charles Garratt<sup>4</sup>

Created: May 1, 2006; Updated: September 14, 2007.

### 1. Why Is It Important to Study Proteins?

In the drama of life on a molecular scale, proteins are where the action is (1).

Proteins are molecular devices, in the nanometer scale, where biological function is exerted (1). They are the building blocks of all cells in our bodies and in all living creatures of all kingdoms. Although the information necessary for life to go on is encoded by the DNA molecule, the dynamic process of life maintenance, replication, defense and reproduction are carried out by proteins.

There are twenty natural amino acids, whose frequency is higher than other special ones with particular functions. These twenty amino acids can be grouped together forming polypeptide chains, or proteins, in different ways determined by the genetic code and limited by stereochemical properties. These proteins may have a constitutive or transient cell expression in regard to its functions. It is worth mentioning that efforts are underway to make proteins of unnatural amino acids as well (2, 3).

The Gene Ontology Project (4) uses three categories to describe a gene product or protein. Molecular Function is the category that describes the tasks performed by individual proteins and can be broadly divided into twelve subcategories; namely cellular processes, metabolism, DNA replication/modification, transcription/translation, intracellular signaling, cell-cell communication, protein folding/degradation, transport, multifunctional proteins, cytoskeletal/structural, defense and immunity, and miscellaneous functions. Structural proteins, for example, are responsible for the cell integrity and cell overall shape (5), their functions varies from composing the cytoskeleton (6) to assembling transmembrane ion channels (7,8), a structure essential for cell osmolarity and even for synaptic information flux. Not only the DNA molecule cannot replicate itself without protein machinery such as the transcription complex, or transcription bubble (9); but also the mitosis and meiosis events of cell duplication and gamete production cannot go on without proteins performing crossing-over events and chromosome segregation (10). The immune system, responsible for our body's defense, is based on structure recognition, on differentiation of self from non-self; this can only be possible through specialized cells that can bind and identify what is foreign to the body. Such recognition processes occurs via protein-protein interactions on the surface of the immune system's cells, where binding affinity can determine if an immune response will or will not be initiated, and also where non-appropriated protein interactions or recognition can cause auto-immune diseases (11, 12). Biochemical reactions of cell breathing, oxygen and carbonic gas transport, food absorption, energy usage, energy storage, heat or cold physiological reactions, or any life process one can figure out, is basically carried out by a protein, or a protein complex. All processes taking place in a living organism have proteins acting somewhere, in a precise way, developed under the pressures of natural selection.

These are only some examples of protein function. A remarkable fact is that all tasks they can perform are based on a common principle, the twenty amino acids that can form a protein. That is the reason why studying proteins, their composition, structure, dynamics and function, is so important. We must understand how these molecules fold, how they assemble into complexes, how they function if we wish to answer questions such as

why we have cancer, why we grow old, why we get sick, how can we find cures for many diseases, why life as we know it has evolved in this way and on this planet and not anywhere else, at least for the moment.

All proteins functions are dependent on their structure, which, in turn, depends on physical and chemical parameters. This is other important fact on studying these molecules; classical biological, physical, chemical, mathematical and informatics sciences have been working together in a new area known as bioinformatics to allow a new level of knowledge about life organization (13).

## 2. Explosion of Biological Sequence and Structure Data

Proteins have traditionally being studied individually. A protein of interest had its coding sequence identified and cloned in a proper expression vector. Hence, provided that cloning, expression and purification were successful, enough quantities of pure proteins could be employed in biochemical experiments or used to prepare solutions for NMR spectroscopy or to grow crystals for structure determination by X-ray crystallography. With the sequencing technology advances in the early 80's, the paradigm shifted from studying single proteins to whole set of proteins of an organism (1, 14).

From the second half of the 90's we can observe an exponential growth of the biological sequence and structure data, mainly due to the Genome Projects underway in different countries around the world, both on public and private industries (Fig. 1). As of April 2007, there were 1952 genome projects, of which 343 are of published complete genomes, 993 and 588 are of ongoing prokaryotic and eukaryotic genomes, respectively, and 28 of metagenomes ([www.genomesonline.org](http://www.genomesonline.org)(15)).

These data area stored in databases or data banks whose number increases every year. According to Galperin (16) there are 858 databases, 139 more than the previous year, available to the public. The amount of data produced urged the necessity for fast and reliable ways of accessing, retrieving, researching and understanding these data (13, 17). Bioinformatics tools are used to perform such tasks in all biological databases, of which GenBank (18) at NCBI and the RCSB/PDB (19) are the most representative for sequences and structures, respectively.

### 2.1 The GenBank® Database

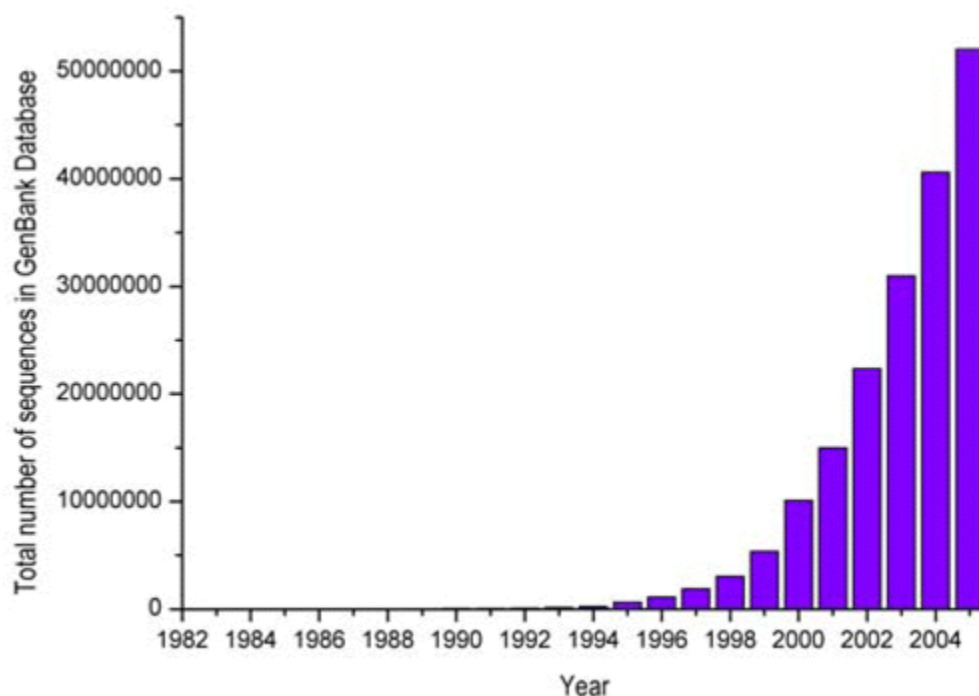
GenBank, possibly the most widely known biological primary database, contains publicly available DNA sequences for more than 205,000 different species, obtained largely through submissions from individual laboratories and batch submissions from large-scale sequencing projects. Together with the EMBL Data Library in Europe and the DNA Data Bank of Japan, they make the three most important sequence databases with daily exchange of data, designed to provide and encourage access within the scientific community to the most up to date and comprehensive DNA sequence information, ensuring worldwide coverage (18).

As of December 15, 2005 there are 52,016,762 sequences in GenBank®. A BLAST (20) search on February 8, 2006 revealed that out of the number above 3,284,262 sequences are non-redundant (nr) coding sequences (CDS), excluding environmental samples. The nr sequences are updated regularly; hence its value can change from one week to another.

### 2.2 The RCSB/PDB Database

The Research Collaboratory for Structural Bioinformatics (RCSB) Protein Data Bank (PDB) provides an information web resource to biological macromolecular structures (19). It includes tools and resources for understanding the relationship between sequence, structure and function of biological macromolecules. As of February 7, 2006 there were 35,026 structures in the RCSB/PDB, of which 33,429 are proteins (Fig. 2), including those making up complexes with nucleic acids [[Summary Table of Released Entries](#)].



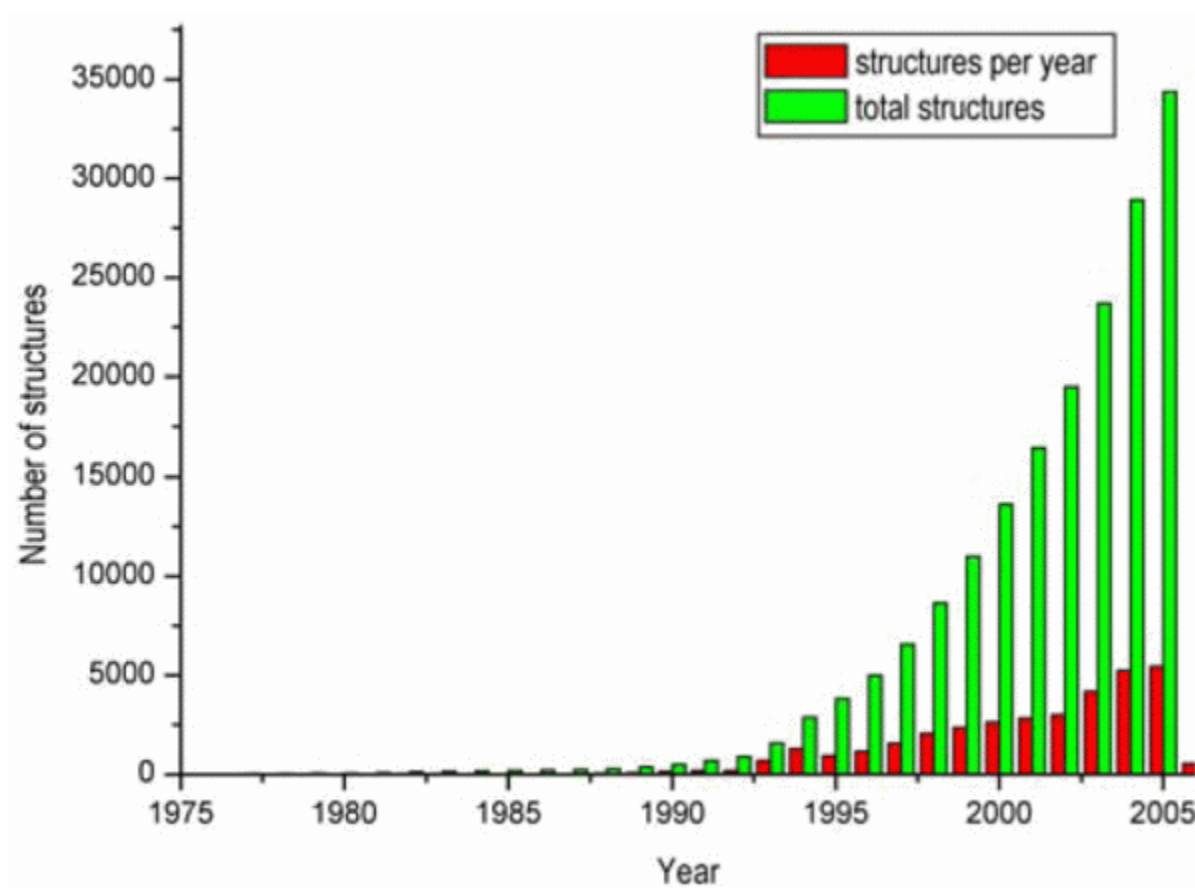


**Figure 1.** Number of sequences available in the GenBank as of December 15, 2005. The biological data explosion in mid 90's can be easily seen with the exponential growth from 1995. For a detailed description of the complete data set, please see <http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html>.

As in the GenBank nr sequences, if we use a 90% sequence identity cutoff to eliminate redundancy in the protein structure data bank we end up with 12,611 clusters or distinct sets of structures [Current Statistics on Redundancy in the RCSB Protein Data Bank]. If we now assume this number represents the different types of known protein folds and compare it to the different types of known protein sequences, from the nr database of GenBank®, we conclude that only 0.38% of the available protein sequences have known folds. Using 1028, the current number of folds determined by SCOP [As Folds Defined By SCOP, (21, 22)], the number of protein sequences with known folds, 0.03%, is considerably smaller. This is in part due to the faster rate with which protein sequences can be determined (from CDS) as compared to the determination of novel protein 3D structures. [Comment: Also, many protein sequences have similar 3D folds, and many of these sequences have little sequence similarity to other sequences with the same 3D fold. This is one main reason why the number of SCOP folds is much less than that predicted by simple sequence similarity.] Clearly, computational methods that are complementary to experimental methods for the determination of protein structures are necessary, and these can be provided by structural bioinformatics efforts [reviewed by 23–25].

According to the Oxford English Dictionary, Structural Bioinformatics is conceptualizing biology in terms of molecules, in the sense of physical chemistry and applying informatics techniques, derived from disciplines such as mathematics, computer science and statistics, to understand, organize and explore the structural information associated to these molecules on a large scale (13). There are several computational methods for protein structure determination, including homology modeling (26), fold recognition via threading (27), and *ab initio* methods (28).

The huge increase in the amount of sequence and structure data of proteins together with advances in experimental and computational, bioinformatics methods, are improving our knowledge about the relationship between protein sequence, structure, dynamics and function. This knowledge, in turn, is being used to understand how proteins interact with other molecules such as small molecules or ligands that can become a drug candidate. In the following sections the hierarchy of protein structure and its application to drug design in



**Figure 2.** Growth of the total number of structures in the RCSB/PDB data base (Kouranov et al., 2006). The exponential growth follows the same pattern of Fig 1.

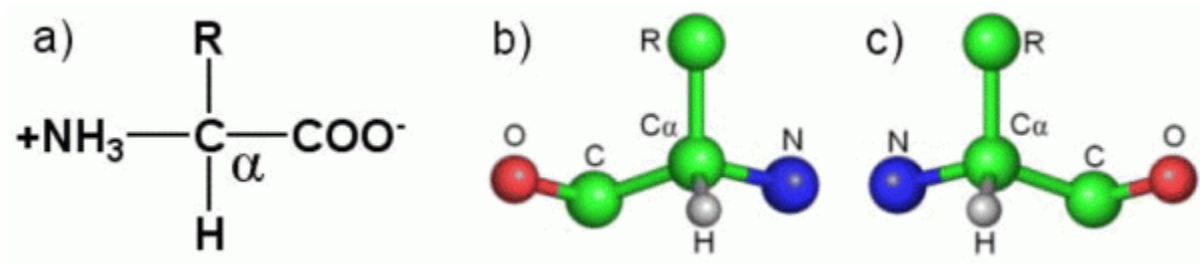
tropical disease will be described. The last section is an exercise that will enable the reader or student to apply several bioinformatics tools starting from a partial DNA sequence or protein accession number and yielding a model of its structure and function.

### 3. Basic Principles of Protein Structure

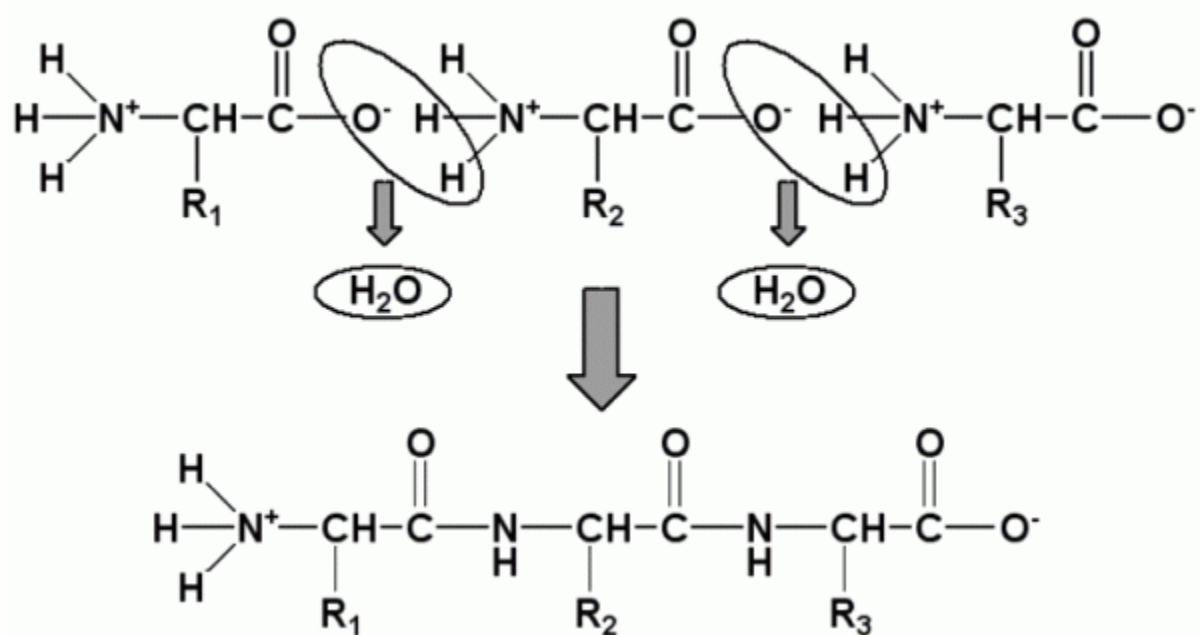
When an all atom model of a protein structure is seen for the first time (be it a figure, a three-dimensional model or a computer graphics representation) it may appear a daunting task to decipher any underlying pattern within it. After all, such structures typically contain thousands if not tens- or hundreds-of-thousands of atoms. For this reason, it is often considered convenient to simplify the problem by using a hierarchical description of protein structure in which successive layers of the hierarchy describe increasingly more complex levels of organization. Typically four levels are used and referred to as the primary, secondary, tertiary and quaternary structure of a protein. It is often useful to include two additional intervening levels between the secondary and tertiary structures, which are referred to as super secondary structures and domains. Many excellent textbooks exist on the subject which should be consulted for more detailed information (1, 29–31). Here, we will attempt only to give an outline in order to emphasize the importance of the study of the three-dimensional structure of proteins for the drug design process which will be described towards the end of this chapter.

#### 3.1 Primary structure

The primary structure of a protein originally referred to its complete covalent structure but is more frequently interpreted as being the sequence of amino acids of each polypeptide chain of which the protein is composed.



**Figure 3. The amino acids.** In (a) the generic structure of an  $\alpha$ -amino acid is given, in which only the R group (side chain) varies from one type to another. In (b) and (c) the difference between L- and D- amino acids respectively is shown (one being the mirror image of the other).



**Figure 4.** A polypeptide chain is generated by a series of condensation reactions, *in vivo* normally occurring within the ribosome during protein synthesis. The individual amino acids are shown in their zwitterionic form, in which the amino and carboxylic acid groups are oppositely charged. This is the most abundant form at neutral pH.

These are often one and the same thing but disulphide bonds and other rarer types of covalent bond formed between amino acid side chains are not directly encoded by the sequence itself.

A polypeptide chain is a unidimensional heteropolymer composed of amino acid residues. There are basically only twenty naturally occurring amino acids which are directly encoded by the corresponding gene, although in exceptional cases stop codons can be used for the incorporation of two additional amino acids (selenocysteine and pyrrolysine (32)). All of these amino acids are  $\alpha$ -amino acids which possess the generic structure given in Figure 3a. Common to all such amino acids is the amino group, carboxylic acid group and hydrogen bound to the central carbon atom (the  $\alpha$  carbon). Only the R group (also known as the side chain) differs from one amino acid to another and it varies in terms of size, polarity, hydrophobicity, charge, shape, volume etc. With the twenty different amino acids available, nature is able to produce the wide diversity of functions which proteins perform in living organisms.

It should be noted that the  $\alpha$ -carbon is tetrahedral and in general is bound to four different chemical moieties. As such it is asymmetric (a chiral center) and two different enantiomers (D and L) for each amino acid exist (Fig 3b). Amongst the naturally occurring amino acids, the only exception is the amino acid glycine, whose R-group

is a hydrogen atom, making the  $\alpha$ -carbon symmetric. This confers a series of important conformational properties on this amino acid which are often essential for the maintenance of a given structure. For this reason, critical glycines are often conserved amongst members of a given protein family. The remaining amino acids are (with very few exceptions) always found to be L-amino acids. This has important consequences for the chiral structures observed in proteins at all levels of the hierarchy. Only two other chiral centers exist within the twenty amino acids, these are the  $\beta$ -carbons within the side-chains of the amino acids threonine and isoleucine, which also exist as only one of the two possible enantiomers.

A polypeptide chain is generated by a series of condensation reactions between the carboxyl group of one amino acid and the amino group of the next, yielding a covalent bond (Fig. 4). This is an amide bond which is given the trivial name of a peptide bond in the case of polypeptide chains. The union of two amino acids results in a dipeptide, which possesses a free amino group (N-terminus) and carboxyl group (C-terminus) allowing the condensation reaction to continue *ad infinitum* in principle, in both directions (Fig. 4). After undergoing condensation, the amino acid is termed a residue (to denote 'that which is left' after condensation). Given the existence of the 20 naturally occurring amino acids, for a polypeptide of length  $n$  there are  $20^n$  possible amino acid sequences. Since  $n$  is typically of the order of hundreds and may even be tens-of-thousands, the number of theoretically possible polypeptide sequences is literally astronomical. The vast majority do not currently exist (indeed, there is insufficient matter in the universe) and never have existed before (as there has been insufficient time to generate them). The polypeptides we observe in the extant species are the products selected from a tiny fraction of all possible combinations that were expressed along the evolutive process.

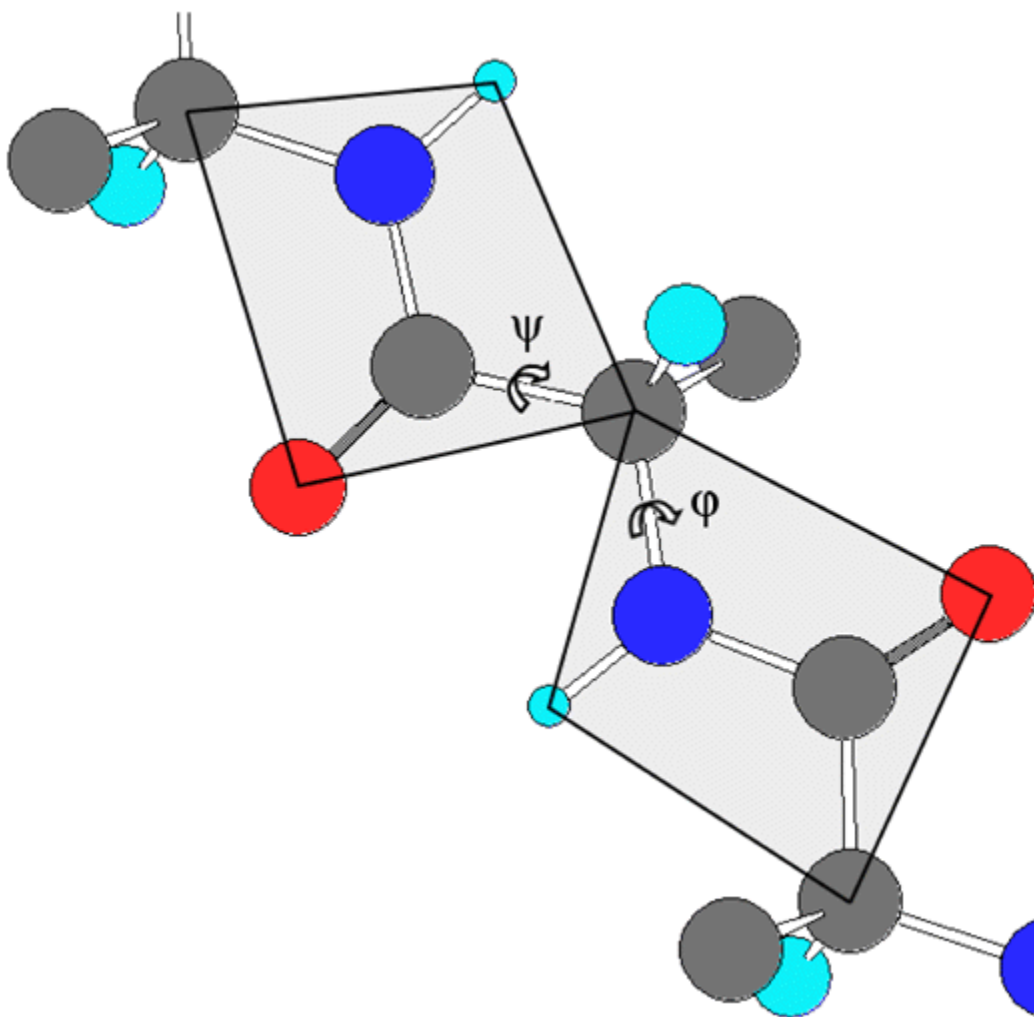
### 3.1.1 The conformational properties of a polypeptide chain

Due to the chemical nature of the amide bond, the peptide linkage between two amino acids is subject to the phenomenon of resonance, meaning that it acquires the characteristics of a partial double bond. The C-N distance (1.32Å) is shorter than a normal single bond and longer than a normal double bond. Furthermore, this introduces rigidity into the structure as the bond is no longer freely rotatable. The consequence is that the  $\alpha$ -carbon atoms of two adjacent amino acids, together with the carbonyl (C=O) and NH groups of the peptide group itself, all lie within the same plane (Fig. 5). The torsion (dihedral) angle associated with this bond is termed  $\omega$ , which in order to be planar must be either  $180^\circ$  (*trans*) or  $0^\circ$  (*cis*). *Cis* peptide bonds are only common when the amino acid proline lies on the C-terminal side of the bond. This is due to the unusual nature of the proline side chain which forms a covalent bond with its own mainchain nitrogen generating a closed ring. As such proline is strictly speaking a secondary amino acid and neither *trans* nor *cis* are particularly energetically favorable. Approximately one third adopt the *cis* conformation and these are frequently conserved amongst protein family members (33).

Polypeptides necessarily obey standard stereochemistry and therefore all bond lengths and angles are effectively fixed, varying only minimally around their standard values. Therefore, the only source of conformational freedom that the polypeptide possesses comes from the torsional rotation around its single bonds. We have already seen that the peptide bond is not freely rotatable and therefore  $\omega$  is effectively fixed, normally in the *trans* configuration. There are only two remaining single bonds per residue along the main chain, and from these bonds one may associate the torsion angles  $\phi$  and  $\psi$  (Fig. 5), which are defined to be  $0^\circ$  when in the conformation *trans*. The conformation of the mainchain of a given residue may therefore be described in terms of these two parameters, which may be conveniently represented in terms of a two-dimensional coordinate system (the Ramachandran plot), in which both  $\phi$  and  $\psi$  vary between  $-180$  and  $+180^\circ$ .

## 3.2 Secondary structure

Not all combinations of  $\phi$  and  $\psi$  are stereochemically possible, since many (such as  $0^\circ/0^\circ$  for example) lead to steric hindrance. In fact, as shown originally by Ramachandran, only about one third of  $\phi/\psi$  space is stereochemically accessible to amino acid residues in a real polypeptide (34).

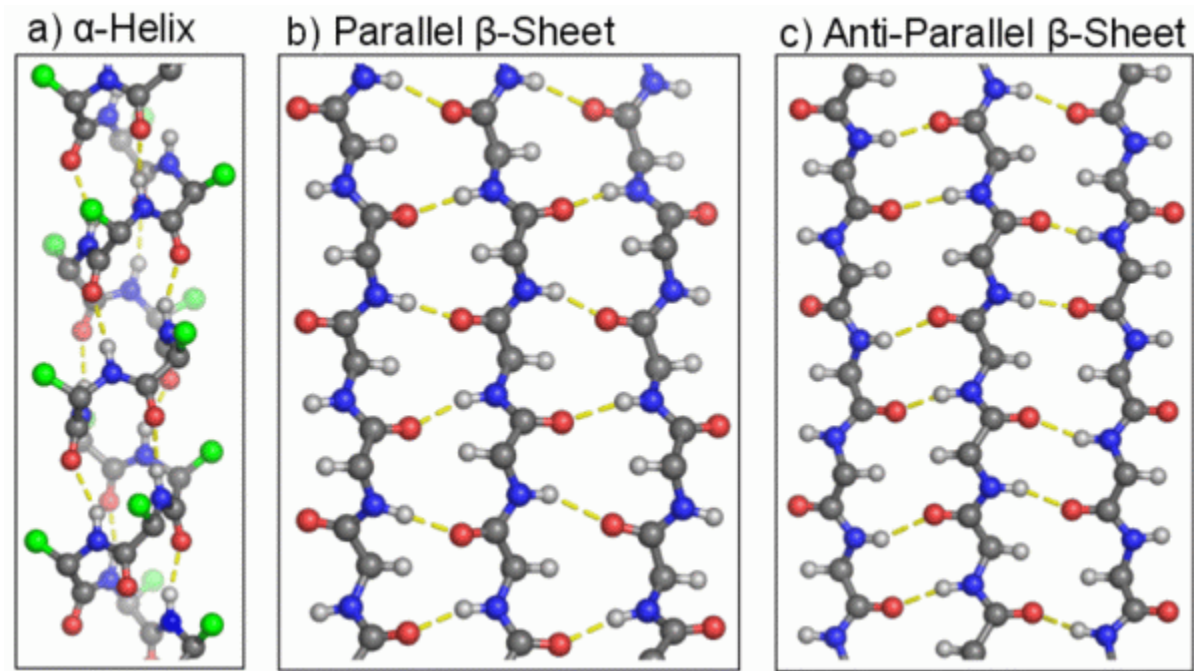


**Figure 5.** A short section of polypeptide chain showing the planar peptide groups and identifying the torsion angles  $\phi$  and  $\psi$ .

If the  $\phi$  and  $\psi$  angles are repeated systematically for all residues within a stretch of polypeptide, the result will inevitably be a helix. An infinite number of types of helix are theoretically possible, depending on the combination of  $\phi$  and  $\psi$ , and they may be conveniently characterized by two parameters,  $n$  and  $d$ , corresponding respectively to the number of residues per helical turn and the shift parallel to the helical axis per residue. There are convenient mathematical relationships which relate  $\phi$  and  $\psi$  to  $n$  and  $d$ .

However, of all the theoretically possible helices, only a limited number occur with any frequency in protein structures. Those which are most important in the case of globular proteins correspond to the  $\alpha$ -helix ( $\phi \approx -63^\circ$ ,  $\psi \approx -42^\circ$ ,  $n = +3.6$ ,  $d = 1.5\text{\AA}$ ) and the  $\beta$ -strand ( $\phi \approx -120^\circ$  and  $\psi \approx +135^\circ$ ,  $n = -2.3$ ,  $d = 3.3\text{\AA}$ ). These are the most important types of secondary structure and correspond to small stretches of consecutive residues. The negative  $n$  value in the case of the  $\beta$ -strand indicates that it is a left-handed helix, whereas the  $\alpha$ -helix is right-handed. The fact that the  $n$  parameter is close to 2 for  $\beta$ -strands (the minimum value theoretically possible) means that strands are in fact very stretched out helices with a narrow main chain diameter. Consequently the side chains emerge alternately on opposite sides of the strand, giving rise to a zig-zag or pleated appearance.

The  $\alpha$ -helix (Fig. 6) is particularly abundant in both globular and some fibrous proteins due to its inherent stability. This arises as a consequence of several factors including the presence of consecutive hydrogen bonds between the carbonyl of residue  $i$  and the NH of residue  $i+4$ , the radius of the helix which is compatible with Van de Waals contacts across the helical axis and the possibility of forming ion pairs (salt bridges) between



**Figure 6.** The most common types of secondary structure.



**Figure 7.** Flat ( $n=2$ ) and twisted  $\beta$ -sheets (left and right respectively)

oppositely charged residues separated by 3 or 4 residues along the sequence. The hydrogen bonds are particularly strong due to the ideal separation between donor (nitrogen) and acceptor (oxygen) atoms as well as due to their linearity and dipolar alignment. Alternative but similar helices, such as the  $3_{10}$  and  $\pi$  helices are much less common due to the weaker nature of the stabilizing forces described above. Autonomous  $3_{10}$  helices do exist, but they are much more commonly observed as the first or final turn of an  $\alpha$ -helix (35).  $\pi$ -helices may be a little more common than originally thought (36).

$\alpha$ -helices in general present a series of common physicochemical and structural properties. They tend to be amphipathic, with a hydrophobic face pointing towards the protein core and a hydrophilic face directed towards aqueous solvent (this may be reversed in the case of membrane proteins). Competition between water and the amide groups of the protein backbone tends to lengthen the hydrogen bonds on the hydrophilic face, leading to helix curvature (37). The alignment of the individual dipoles of the peptide units leads to a macroscopic dipole associated with the whole helix which is typically of the order of 0.5 units of positive charge at the N-terminus and 0.5 units of negative charge at the C-terminus (38). Together with the free NH groups within the first turn,

this makes the N-termini of  $\alpha$ -helices ideal binding sites for anions such as phosphate. In the absence of such an anion, frequently the N-terminus is capped by the side chain of the first helical residue (the N-cap) which is most often an asparagine or aspartic acid (39). The most common amino acid encountered at the second position is a proline, meaning that the sequence Asn-Pro would, in principle, be a strong helix initiation signal.

$\beta$ -strands are unable to form internal hydrogen bonds. They assemble into sheets by the formation of hydrogen bonds between strands (Fig. 6). Sheets may be parallel, antiparallel or mixed in nature depending on the relative orientation of the strands of which they are composed. The  $\phi$  and  $\psi$  angles vary from one type to another and  $\beta$ -sheets are capable of assuming a wide array of architectures (40). One characteristic common to almost all  $\beta$ -sheets is that they are twisted. When viewed perpendicular to the strands, the sheet has a left-handed twist and when viewed parallel to the strands it is right-handed. The latter may appear a contradiction given the left-handed twist of the individual strands (see above), but results from the fact that only every second residue (1, 3, 5, 7 etc or 2, 4, 6 etc.) contribute hydrogen bonds to the same side of the strand. Planar sheets (with  $n = 2$ ) do exist but are rare compared with twisted ones. Figure 7 shows examples of both flat and twisted  $\beta$ -sheets.

Amongst the different architectures that  $\beta$ -sheets form are parallel (rare) and anti-parallel (common) coiled coils, parallel and anti-parallel saddles (hyperbolic paraboloids), 8-stranded parallel barrels and anti-parallel barrels with a highly variable number of coiled strands, helicoids of three strands etc. (40)

In globular proteins the  $\alpha$ -helices and  $\beta$ -strands will often traverse the structure. Since both have approximately linear axes, clearly other types of structure are needed in order to cause chain reversal. These are termed turns and loops. Turns are composed of a well defined number of residues.  $\delta$ -turns are composed of two residues,  $\gamma$ -turns three,  $\beta$ -turns four,  $\alpha$ -turns five and  $\pi$ -turns six (41). The most well-known and widely described are the  $\beta$ -turns, which come in several different forms distinguishable by the  $\phi$  and  $\psi$  angles of the second and third residues of the turn. In most (but not all) definitions, a hydrogen bond exists between the carbonyl of the first residue and the NH of the fourth. Several classifications have been proposed for  $\beta$ -turns and types I and II are the most common (see 41). However types I' and II' are observed in  $\beta$ -hairpins (a super-secondary structure composed of two anti-parallel  $\beta$ -strands) due to the compatibility of the sheet twist with that of the turn itself (42). In general, turns can not be represented by a single point in  $\phi/\psi$  space (unlike the helices described above). Indeed many of the  $\beta$ -turns require one of the internal residues (2 or 3) to adopt 'prohibited'  $\phi/\psi$  angles, imposing a requirement for a glycine at specific positions (41). The type of turn may sometimes therefore, be easily predicted from the sequence of its component residues.

Loops are more difficult to classify as they are larger than turns and can therefore adopt a wide variety of conformations. Different to turns, they generally present a hydrophobic core and, at least in the case of  $\Omega$ -loops, are as compact as other parts of the structure.

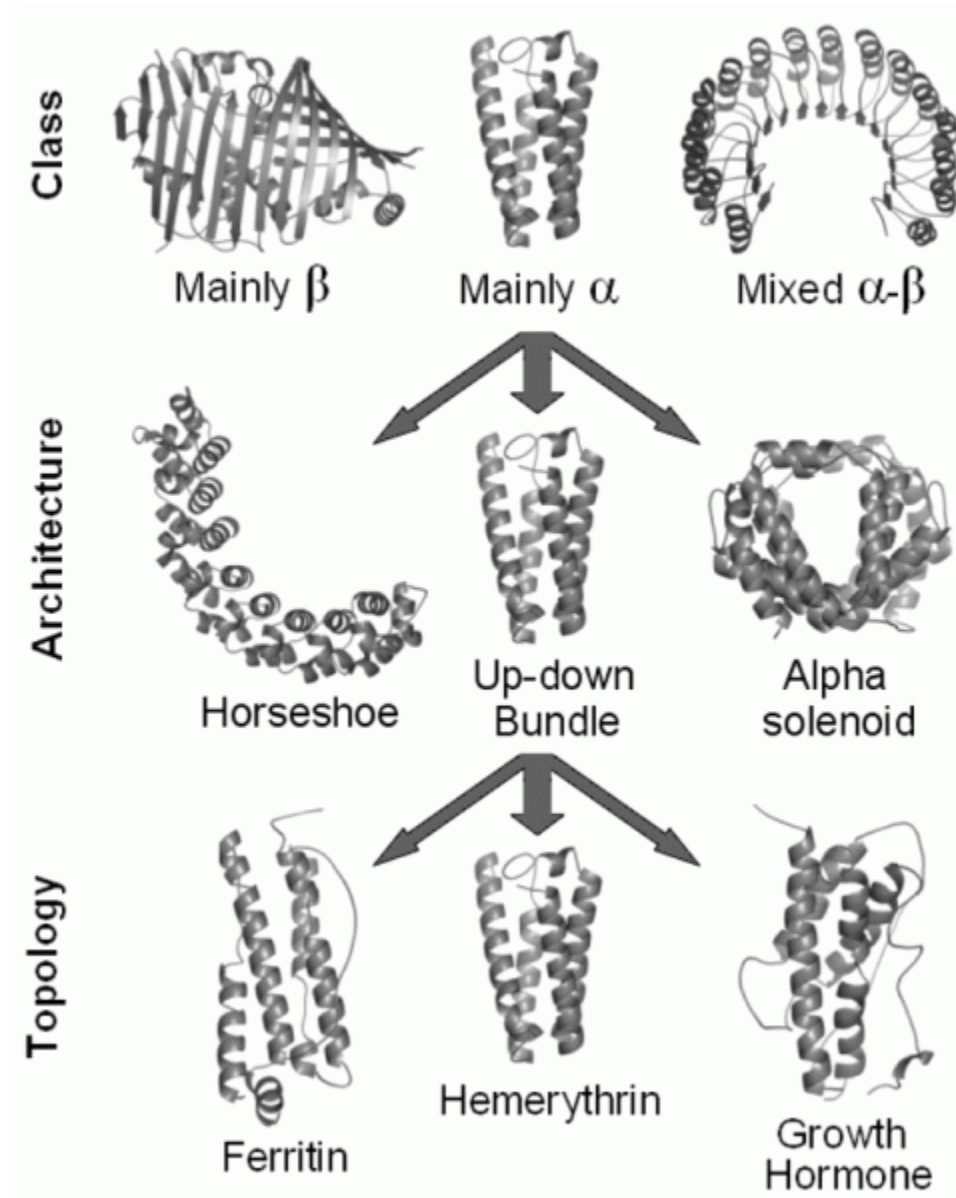
### 3.3 Tertiary structure

In order to form globular structures, the individual elements of secondary structure of a polypeptide must pack against one another in order to form a stable, compact and biologically active tertiary structure. The ways in which secondary structures most commonly pack has been widely studied (43–45). The most simple models which aim to explain experimental observations are based on the interdigitation of side chains. For example, the knobs-into-holes approach or ridges-into-grooves model (43) explain equally well the most commonly observed packing angles between two  $\alpha$ -helices ( $\Omega = -50^\circ$  and  $20^\circ$ ). A similar approach can be used for the packing of  $\beta$ -sheets against one another or of helices against sheets (44, 45).

The final result of the folding process is the full three-dimensional structure of the polypeptide. Clearly, this depends on the amino acid sequence and therefore on the atomic details of the structure. However, for soluble proteins this will almost inevitably result in the generation of a hydrophobic core in which apolar residues tend to cluster their side chains within the protein's interior, leaving hydrophilic residues exposed to solvent. For



**Figure 8.** Different representations of the same protein structure (Alzheimer's amyloid precursor protein). All atoms except hydrogens (left), Ca- trace (center) and ribbon (right).



**Figure 9.** Part of the CATH classification. Examples from the upper most levels in the hierarchy (Class, Architecture and Topology) are shown.

descriptive purposes it is useful to simplify the structure by removing the atomic detail and considering only the trace of the mainchain in space (see Fig. 8). This is often referred to as the fold of the protein. Many classifications of protein folds exist and two of the most widely used are SCOP (22) and CATH (46), both of



which use a hierarchical approach. The highest level of the hierarchy classifies folds into classes, based on their overall secondary structure content (mainly  $\alpha$ , mainly  $\beta$ ,  $\alpha/\beta$ , irregular). In the case of the CATH classification (Fig. 9), the second layer of the hierarchy deals with different architectures within a given class, of which there are currently approximately 30 major types. For example, within the  $\alpha$ -class folds may be classified at the architectural level as  $\alpha$ -bundles,  $\alpha$ -non-bundles,  $\alpha$ -horseshoes,  $\alpha$ -solenoids and  $\alpha/\alpha$  barrels. Within the  $\beta$ -class, there are  $\beta$ -rolls,  $\beta$ -barrels,  $\beta$ -clams,  $\beta$ -sandwiches  $\beta$ -trefoils,  $\beta$ -prisms (parallel and orthogonal)  $\beta$ -propellers (with different numbers of blades), different types of  $\beta$ -solenoids (or  $\beta$ -helices) etc. Within the  $\alpha/\beta$ -class we find  $\alpha/\beta$  barrels (TIM barrels),  $\alpha/\beta$  sandwiches (with varying numbers of layers and types), the  $\alpha/\beta$ -box,  $\alpha/\beta$ -horseshoe,  $\alpha/\beta$ -propeller etc. The architecture describes the overall arrangement of secondary structural elements but without concern for their connectivity.

At the level of topology, a finer description is given within a given architecture in which the connectivity of the secondary structures is taken into account. For example, even considering a relatively simple architecture, that of an  $\alpha$ -bundle, for example a four-helix bundle, there is a surprising number of different topologies (ways of orientating the helices with respect to each other and then connecting them). In fact there are 48 different topologies theoretically possible for a four-helix bundle (47). However, probably for reasons of simplicity of folding, nature has favored the all anti-parallel bundle in which two diagonally related helices are orientated parallel and the remainder anti-parallel. These topologies are more common than one would expect by chance.

There is a general belief that the total number of folds is relatively limited (perhaps a few thousand) and it is expected that as the result of structural genomics initiatives, within the next few years our knowledge of the universe of possible folds used by soluble proteins at least will be reasonably complete (48). Already it is clear that some folds are more common than others and these have been termed superfolds by some authors (49). They include the TIM barrel, the jelly-roll and the up-and-down 4-helix bundle.

With the accumulation of new protein structures at an ever increasing rate, determined by X-ray diffraction and multi-dimensional NMR, it has been possible to derive a series of empirical rules which govern the possible ways in which a polypeptide may fold. Although many such rules remain founded on a purely empirical basis, some of them can probably be explained by appealing to the need for efficient folding kinetics. The most well established rule is that which determines that connections between elements of secondary structure of the type  $\beta x \beta$  (where  $\beta$  refers to a  $\beta$ -strand and  $x$  may be anything, an  $\alpha$ -helix, a loop or another  $\beta$ -strand) should be right-handed (50). There are very few exceptions to this rule, but one spectacular example is the left-handed  $\beta$ -helix as seen for example in UDP-N-acetylglucosamine acyltransferase (51). Other connectivity rules prohibit the crossing-over of loop connections and the formation of knots for example. However, as with most empirical rules, exceptions can always be found and recently some dramatically knotted structures have been described (52).

It should be noted that proteins are inherently chiral molecules and this has important consequences for the interactions they make. Many enzymes for example are stereoselective with respect to their substrates due to inherent asymmetry in their active sites. For example, when an artificial HIV protease was synthesized from D instead of L amino acids, the result was an active protease which was specific for a substrate of the opposite enantiomer (53). The consequences of the asymmetric  $\alpha$ -carbon atom can be seen at most if not all levels of protein structure. It is not by chance that the right-handed  $\alpha$ -helix is energetically favored over its left-handed partner, nor that  $\beta$ -strands are left-handed (right handed when only every second residue is viewed) or their sheets left-handed (when viewed perpendicular to the strands). Furthermore, the packing of right-handed  $\alpha$ -helices using side chain interdigitation leads to the common packing angle of  $20^\circ$ , meaning that when these helices coil around one another to form coiled coils (as seen in keratin, myosin and leucine zippers for example) inevitably such structures will be left-handed. At the next level up, the right handed connections found in  $\beta x \beta$  units together with the twist on  $\beta$ -strands themselves means that all  $(\beta\alpha)_8$  barrels (TIM barrels) have the same chirality. Indeed, the barrel will always 'roll up' the same way because of the combination of strand twist and

sheet shear (the shifting of one strand with respect to its neighbors which results in the inclination of the strands with respect to the barrel axis) (1, 40).

### 3.4 Quaternary structure

Some proteins (a minority) are composed only of a single polypeptide chain. The remainder are homo- or hetero-oligomers or macromolecular assemblies or polymers. In a recent evaluation of the proteins encoded by the *E. coli* genome, only about 20% were expected to be monomeric (54). Dimers were the most common, representing nearly 40% of all proteins, followed by tetramers (21%). When two subunits (polypeptide chains) interact to form an interface, two possibilities exist; the interface may be either isologous or heterologous. In the first case, the association leads to the formation of a symmetric dimer possessing a two-fold (diad) axis, in which if a region A on the first subunit interacts with a region B on the second, then region A of the second will necessarily interact with region B on the first (1). In the case of heterologous interactions this is not the case. They may be either symmetric, leading to ring-like structures possessing symmetry axes of higher order ( $>2$ ) or asymmetric.

Almost all oligomeric structures possess internal symmetry. Given that the individual subunits themselves are asymmetric and chiral (because they are made of L-amino acids) this means that the symmetry can only arise by the association of subunits around pure rotation axes. In principle, these can be of any order and many examples including 2-fold, 3-fold, 4-fold, 5-fold, 6-fold, 7-fold, 9-fold and 11-fold axes (and maybe others) have been observed in nature. Three fundamentally different arrangements are possible. Cyclic structures (built up of a ring of subunits employing one type of heterologous interface generally lead to pore-like structures, particularly when the number of subunits becomes large. Examples include neuraminidase (4-fold) and the light-harvesting complex (11-fold). Dihedral structures, besides possessing one principle axis, also have a series of perpendicular 2-fold axes. For example, most tetramers are built using  $222$  symmetry (three mutually perpendicular 2-fold axes) rather than a single 4-fold axis, as found in neuraminidase. Dihedral symmetry gives more options for creating different interfaces between subunits, and this richness is often taken advantage of in allosterically regulated enzymes. Together with a higher order principal axis, dihedral symmetry can be used for creating proteins with internal cavities accessible via channels. Examples include the chaperone GroEL (55) and the proteasome (56), both of which have 7-fold axes.

The third option is the use of cubic or icosahedral symmetry in which a series of three fold axes are accompanied by non-perpendicular two-fold, four-fold or five-fold axes (or combinations of these). The resulting structures are generally approximately spherical and are used as storage molecules (such as ferritin for example) or as viral capsids. Furthermore, translational symmetry can be added to the rotation axes to produce helical symmetry as seen in microtubules, tobacco mosaic virus and other filamentous structure.

What are the advantages of building oligomeric proteins and why are they so common? Several answers exist. Sometimes it is just a question of being big, as for example in the case of plasma proteins which, in order to avoid ultra-filtration by the kidneys, must be greater than 60KDa. Being large can often also increase stability and reduce the surface area/volume ratio, thus making the protein less susceptible to proteolysis. Being oligomeric also gives rise to the potential for cooperativity in enzymes and transport molecules, as well as being an efficient way of building large structures with limited genetic material, such as in spherical viruses for example. The potential of having multiple identical binding sites within a single molecule (as in immunoglobulins for example) introduces the possibility for a cross-linking activity and as mentioned previously many oligomeric structures serve to generate cavities with a specific biological function.

### 3.5 A final comment on protein structure

The reason for interest in the full three-dimensional structure of a protein is generally to understand its biological activity or to interfere with it, as for example in the case of drug design (see below). In both cases a

merely topological description is clearly insufficient because biological activity depends on atomic detail and often apparently subtle modifications to a structure can dramatically affect its activity. Here we do not have space for full descriptions of even a limited number of protein families. However, in the following section it should become evident that the knowledge of a high resolution atomic structure of a target protein is an essential prerequisite for structure based drug design.

## 4. The Use of Protein Structures in Inhibitor Discovery and Design

Before the development of techniques for obtaining the 3D structure of proteins, drugs were discovered by systematic modification of compounds with known biological activities that were discovered by chance or random screening, based on trial and error. It was a time and money consuming approach, and even when employing all medicinal chemistry's methods to reduce the amount of compounds to be synthesized and tested, these approaches required exhaustive procedures of syntheses and evaluation. Traditional medicinal chemists focused their attention in understanding the relationships between structures and activities of known active compounds and proposing modifications that would improve some biological property of the molecule (57, 58).

Medicinal Chemists took immediate advantage of the rapid development in computing and computer graphics in particular, and developed techniques for visualizing, manipulating and superposing 3D structures of active compounds in order to obtain faster information about the structure-activities relationship of known active compounds. Different computer based Quantitative Structure-Activities Relationship (QSAR) approaches like HQSAR, CoMFA and related 3D QSAR approaches became popular amongst medicinal chemists.

Simultaneously, significant advances in molecular and structural biology were made and currently tens of thousands of 3D protein structures are available. This represents an enormous amount of structural information about medicinally relevant receptors, furthermore structures of protein complexes with small ligands are also an invaluable source of information about the favorable interactions that stabilize the ligand's association with the binding pocket of the protein (59).

Recently, powerful desktop computers have become affordable, together with an ever increasing diversity of free software for academic purposes as well as a marked improvement in data transfer through internet access. Thus, it is now feasible to work at home on powerful workstations, in close contact with academic research groups throughout the world, and taking full advantage of up-to-date software for visualizing molecules, performing molecular modeling and even to process crystallographic data and solve structures.

In spite of such unquestionable progress, structure-based ligand design still faces several major challenges and limitations such as a lack of availability of 3D structures for many important receptors, particularly membrane-bound receptors, adequate computational approaches to deal with induced-fit in ligand-receptor interactions or water molecules in the binding site and selectivity between different isoforms of the same receptor or enzyme. Furthermore, collaboration with synthetic chemists and experimental biologists will probably continue to be a critical determinant of success (59).

Drugs are ligands that not only fit into the binding pocket of the target protein, but also are absorbed, transported, distributed to the right body compartment, as well as ideally being stable to metabolism, non-toxic, free of side effects and chemically stable in the formulation. So it is clear that the ligand design stage of the drug discovery process is just one of the initial steps in a multidisciplinary effort that usually requires several rounds of refinement between the identification of an active principle and the selection of a viable drug candidate. Structural based design can assist this continuous process by pinpointing opportunities for structural modifications that do not interfere with binding, but may improve affinity and selectivity, and which may favorably alter the properties of the molecule, such as solubility, hydrophobicity and ionization state (59).

Finding a lead compound, optimizing its properties and turning it into an approved drug takes an enormous amount of time and money. It is a quest that demands an intense and cooperative effort between multidisciplinary discovery teams (60).

In some cases the careful analysis of a protein-ligand structure can be enough to suggest modifications in the ligand to improve affinity. More structures of the same protein with different ligands can make it easier. Molecular modeling tools can model a protein's structure based on a previously known homologous structure, so it is possible to use the structural information to design ligands for proteins for which no structure is known. Virtual screening can provide ligand structures completely different from the already known ligands. QSAR models can aid the design of modifications to improve a biological property of a ligand and can predict activities with a reasonable error. Docking procedures can be used to model binding modes and estimate binding energies.

## 4.1 Applications of Protein Structures

Predicting the binding modes and affinities of different compounds as they interact with protein binding sites is the main goal of structure-based drug design. Computational approaches to this problem are usually called docking. Figure 10 shows schematically one approach to this problem as implemented in the program DOCK (61).

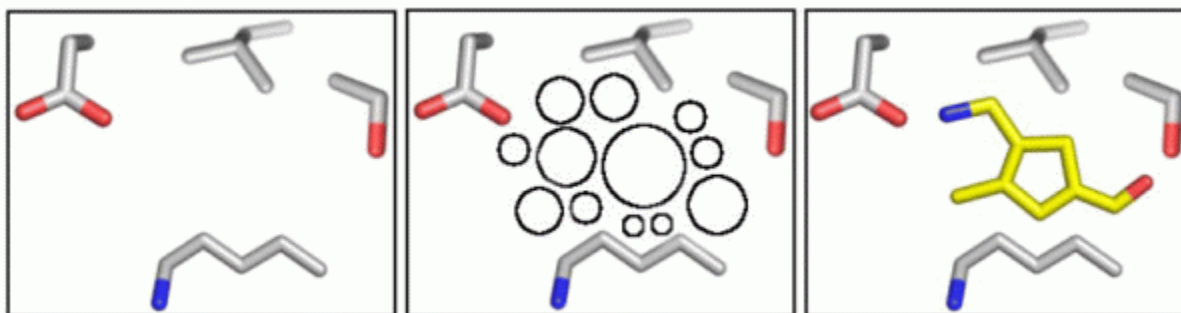
The key characteristic of a good docking program is its ability to reproduce the experimental binding modes of ligands (62). A large number of programs have been written to address this area. In general, the programs operate as follows: A large number of conformations are generated for the small molecule, either prior to docking or during the docking routine. Each conformation is positioned in the active site in a variety of orientations, the combination of conformation and orientation being known as a "pose". This sampling procedure is usually based genetic algorithms, Monte Carlo simulation, simulated annealing or distance geometry (61, 62). Many poses are selected and ranked by a scoring function, in order to determine the best overall pose.

In general, the potential flexibility within both the ligand and residues of the binding site represents a significant computational challenge. To make the docking problem tractable, most programs address adequately the question of ligand flexibility, but assume that the binding site is rigid. Also, some programs partially address protein flexibility. The scoring functions developed to date usually use either a classical molecular mechanics force field, an empirical function describing terms such as hydrogen-bonding and lipophilicity, or a "knowledge-based" potential derived from analysis of protein-ligand complexes (62, 63).

Another important characteristic of a good docking program is the ability of its scoring functions to score and rank multiple ligands against a single target according to their experimental binding affinities (62). This task is more problematic (64), the scoring functions used in docking programs must contain many approximations, since they have to be fast enough to evaluate extremely large numbers of possible poses. For instance, they do not usually take explicit account of entropic changes due to displacement of waters from the active site or of reduction in degrees of freedom in the ligand. Usually a scoring function that can select the correct pose for one ligand will not necessarily rank a series of potential ligands in the correct order (63).

An important use of protein-ligand docking programs is virtual screening, in which large libraries of compounds are docked into a target binding site and scored. For this purpose, the dockings need to be even quicker. Speeding up a docking protocol is often done at the cost of sampling fewer binding modes, and, as a result, reduces the success rates. It is therefore important that search parameters are chosen that give docking speeds useful for virtual screening applications, with an acceptable loss in docking accuracy (63).

To overcome the "ranking problem" researchers have employed different methods ranging from brute force, which means using highly expensive computer clusters to do all the heavy computational work, to using more



**Figure 10.** Schematic representation of the DOCK search algorithm. Spheres are calculated over the surface of the receptor site and then filtered to keep only the largest sphere associated with each surface atom. Ligand atoms are fitted into the site by matching the spheres' center.

elegant approaches like consensus scoring. This method consists of using a docking program to do the conformational search and rank the compounds, and employ other docking tools' scoring functions to reevaluate the top results. Only the top scored compounds common to each scoring function will be identified as candidates for bioassay. Compared to single scoring procedures, it was shown that false positives in virtual library screening were largely reduced and the hit-rates were improved. There are many different ways of performing consensus scoring (64).

Some docking tools also have another useful characteristic, the estimation of binding energy. However, this technique still demands a very high computational power and most projects can not afford it. Usually only the best ligands found on virtual screening are submitted to this procedure. Note that even though these calculations are not very accurate, if it can provide the correct rankings of candidate molecules, it will provide valuable information.

#### 4.1.1 Getting started with Virtual Screening

Virtual screening results are highly dependent on database and target preparation. Databases need to be tuned, and that means a considerable effort should be put into filtering out molecules with undesirable physical properties and chemical functionalities such as high molecular weight, reactive groups, too many rotating bonds and non suitable lipophilicity (65). Tautomeric states, ionization and charges should be also observed, as well as the possible enantiomers arising from the chiral centers. Finally, one must decide if one wants a database of drug-like or lead-like compounds.

Before preparing the receptor, it is necessary to check if it is adequate for a virtual screening procedure. It is common to have available more than one structure of the same protein in complex with different ligands or without any ligand bound. It is up to the medicinal chemist to choose between these different options. As a guideline, the resolution must be as high as possible. Another important step is to read the pdb file of the receptor. Useful information can be found about the crystallographic parameters employed, notes from the author about residues with low electron density and even mutations in the protein. Also programs such as Procheck and Whatcheck can be used to analyze the stereochemistry and chemical environment of each residue of the protein.

After choosing the most appropriate receptor structure, usually the addition of hydrogen atoms is required, followed by minimization of the potential energy to avoid steric clashes. Some changes must be carefully done on the binding site, like assigning the right protonation state to ionizable residues, checking the tautomeric states of the histidines, the possible flips of the glutamines and asparagines and the orientations of the hydrogen atoms of the hydroxyl groups on the serine and threonine residues (65).

Remember that virtual screening procedures usually take days to finish and if a mistake is made in the preparation stages, the entire work may be compromised. Also one of the most important procedures is the visual inspection of the best ranked compounds in the context of the binding site. Serendipity, the principle that "chance only favors the prepared mind" in the words of Louis Pasteur, plays an important role in the whole virtual screening process especially during the visual inspection stage.

### 4.1.2 Protein structure and 3D-QSAR

Protein structures can be used in many ways to improve the results of Quantitative Structure-Activity Relationship (QSAR) techniques. Two of the most common techniques are the Comparative Molecular Field Analysis (CoMFA) and the Comparative Molecular Similarity Indices Analysis (CoMSIA). Both methods assume that the ligands have a common binding mode, their 3D structures are modeled and superposed and Partial Least Square (PLS) analysis is performed to try to find a correlation between the compounds' biological activities and their 3D structures (66).

These methods strongly rely on a correct superposition of the ligand structures and, if the structure of the target protein is known, a valid approach is to dock the ligands into the protein binding site and use the result as a "structural alignment". The protein structural information can also be used to aid the interpretation of the results.

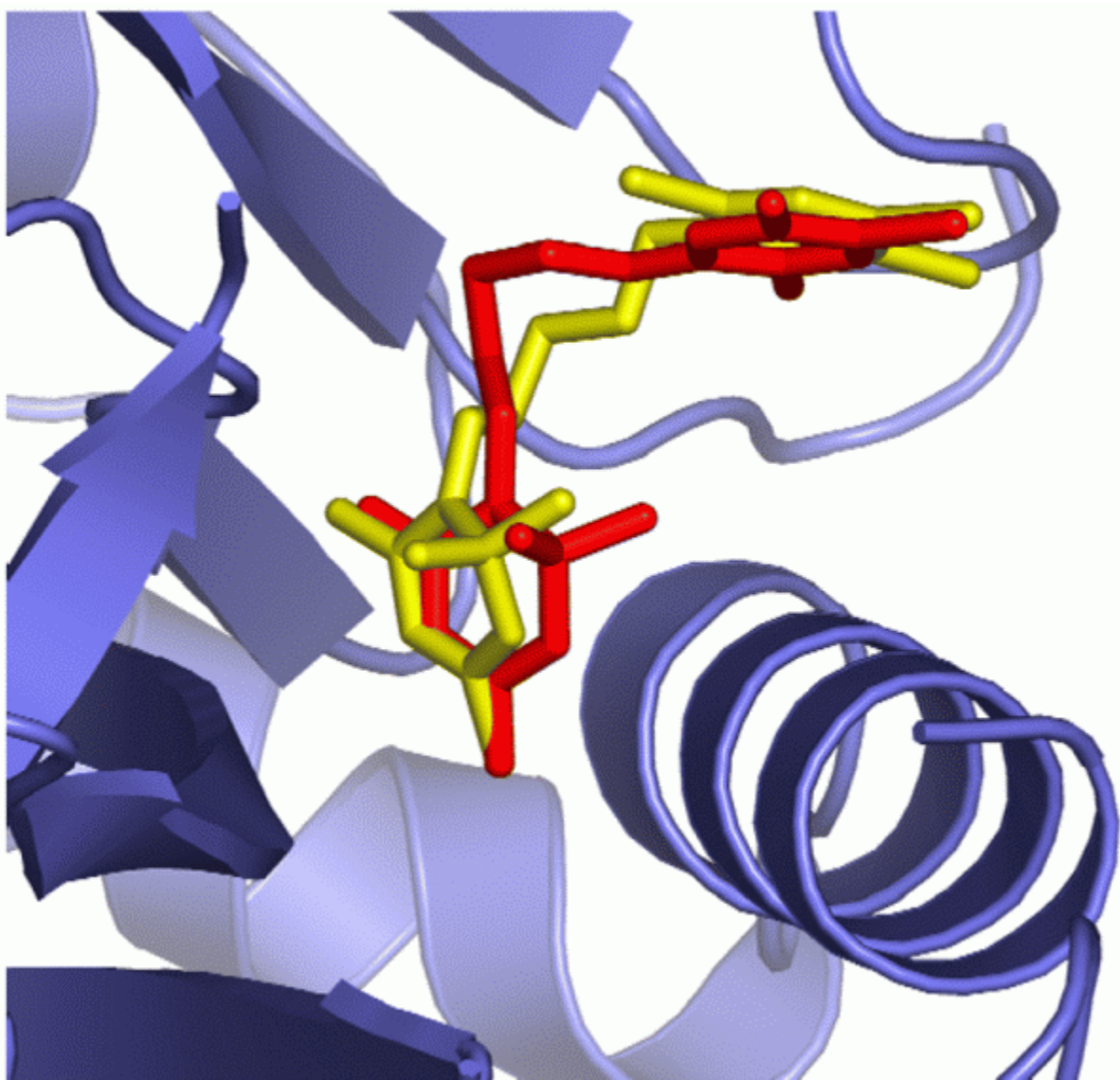
Another QSAR method, Comparative Binding Energy (COMBINA) requires at least one structure of a ligand-protein complex and experimental binding data for the training set of ligand-protein complexes. The training set is used for model development only and not for evaluation of the model, which must be performed using an independent set of data. The structure of the known ligand-protein complex is used to model all remaining complexes of the training set (in case of no structural data being available), these complexes are superimposed and energy is minimized. The interaction energy between ligand and receptor in each complex is computed and decomposed on the basis of physical properties and location in the molecule. PLS analysis is then used to build a model correlating the biological binding data with the sum of weighted, selected components of the interaction energy (66).

## 4.2 Structure-based Drug design targeting Tropical Diseases

Tropical diseases are responsible for some of the most severe worldwide health problems. Hundreds of millions individuals are affected by these diseases (67–71) killing millions of people annually. Structural differences between host and parasite can be exploited aiming to develop a selective drug against the parasites, without causing damage to the host.

An example of this approach is the dihydrofolate reductase-thymidylate synthase (DHFR-TS) enzyme (Fig. 11). Although the human host presents this enzyme in its biochemical pathway, the human enzyme differs structurally from the parasite enzyme, and these differences can be exploited in order to design selective inhibitors against the parasites' homologous enzyme. The bifunctional enzyme DHFR-TS from parasites such as *Toxoplasma gondii*, *Cryptosporidium parvum* and four different species of *Plasmodium* plays a crucial role in pyrimidine biosynthesis and has been used as a target for structure-based drug design.

The structure of DHFR-TS from *Leishmania major*, and the recent determination of the structures from *Cryptosporidium hominis* and *P. falciparum* (67) can now shed some light on the binding mode of wild type DHFR-TS and mutants resistant to current inhibitors, such as methotrexate and dapson (Lapdap™). These structural data offer the possibility to develop novel inhibitors with higher potency and selectivity, and these data would be useful for molecular modeling DHFRs from other species of parasites for which structures are not available yet (72). Furthermore, the widespread resistance to currently available antimalarial drugs has prompted an increase in the search for new targets, some of them already in the validation stages (72).



**Figure 11.** Docking of the synthetic ligand Wr99210 into the binding site of wild-type *Plasmodium falciparum* DHFR (pdb ID: 1j3i). The program GOLD (red) was able to reproduce the crystallographic position of the ligand (yellow) (62).

DHFR inhibitors are largely ineffective for the control of trypanosomatid infections, partly due to the presence of pteridine reductase (PTR1), which is involved in folate/pterin metabolism. Crystal structures of the PTR1 from *Leishmania tarentolae* and *Trypanosoma cruzi*, and a series of crystallographic analyses of the *Leishmania major*'s enzyme in binary and ternary complexes identified important interactions that can be exploited in a structure-based development of novel enzyme inhibitors with potential therapeutic value (73).

Moreover, other approaches are being considered in order to tackle the development of novel drugs against trypanosomatides. The crystal structure and mechanism of trans-sialidase, an enzyme that appears to be critical for *T. cruzi* survival and cell invasion capability is now available. The structure provides the first step to the structure-based design of novel inhibitors with potential therapeutic applications (74).

Also, the 2.0Å resolution structure of UDP-galactose 4'-epimerase from the protozoan parasite *Trypanosoma brucei*, the causing agent of the African sleeping sickness in humans, and the wasting disease nagana in cattle, was recently determined. A structural comparison with the human enzyme identified a potentially important

difference in the substrate-binding cleft that might assist a structure-based approach to the development of *TbGalE* specific inhibitors (68).

Other enzymes that have been considered as good targets for antiparasitic drug development are the Glutathione *S*-transferases, a family of detoxification enzymes that catalyze the conjugation of glutathione with various endogenous and xenobiotic electrophiles. Studies have identified antischistosomal (75), antimalarial and antifilarial (69) activity of compounds known for their GST inhibiting activity (70).

Crystal structures of the Glutathione *S*-transferase of the filarial worm *Onchocerca volvulus*, responsible for the debilitating disease onchocerciasis, were recently obtained in high resolution (1.5Å and 1.8Å). The structural analysis of the enzyme shows that it is a suitable drug target for intervention in filarial infections, with the potential advantage of not compromising the human host (69).

Other structure-based efforts to change the tropical diseases panorama are being made. An example is the recent study that used available structures and gene sequences of the enzyme glyoxalase I to create a model of the *Leishmania donovani* enzyme (76). This work suggested that the differences between the human and the *L. donovani* structures could be exploited for structure-based drug designing of selective inhibitors against the parasite's enzyme.

Also, structural analysis of the *Mycobacterium tuberculosis* type II dehydroquinase suggested modifications on known inhibitors that led to the synthesis of a new inhibitor over than 180 times more potent than the prototype inhibitor. Docking studies indicated key electrostatic binding interactions between the inhibitor and the enzyme. Structural studies of enzyme-inhibitor complexes are underway and should be considered for the design of the next generation of inhibitors (71).

## 5. Practical Section: Computational Characterization of a Gene Product from a Partial Gene Sequence Only

The objective of this exercise is to bring the reader or student in contact with freely available online bioinformatics tools used to computationally characterize a complete protein starting with only a partial DNA sequence of its corresponding gene. Through a set of guiding questions, with suggestions, this exercise will involve searching the GenBank for the whole gene sequence, translating it into the corresponding protein sequence, and searching for the protein structure (here it can either be assumed that the student is already familiar with the outputs from the distinct BLAST programs or that learning will be carried out through the exercise). If no structure is available, then the student should search for similar sequences with 30% or greater identity that have known 3D structures. Depending on the course length, the student can proceed by determining the protein structure by homology modelling or simply by using the known 3D structure to infer structural similarity and function for his/her protein. The student is encouraged to download any known protein 3D structure file to his/her computer and visualize it using a locally installed biological macromolecule viewer such as SwissPDBViewer (77) or VMD (78). This will be a unique opportunity for the student to get familiar with protein architecture. The same experiment can be easily changed to investigate a whole genome as well. The set of questions can be used to guide the student through the writing up of an article describing the gene and its product. This exercise should be conducted by a lecturer or tutor in bioinformatics.

### 5.1 Molecular Biology or Genomics Laboratory Experiment

In a molecular biology or genomics laboratory a DNA fragment, supposedly corresponding to a gene, was identified. Its first 30 base pairs have been sequenced and determined to be 5'-TGACACAACAAGGACGCACATGACAGGA -3'. Answer the questions below through Bioinformatics experiments.



### 5.1.1 Gene Characterization

1 - Can the putative gene be completely and unambiguously characterized from the partial DNA sequence given above? How?

Yes. The gene can be completely and unambiguously characterized, but the solution will require a more careful analysis than it did a couple of years ago. The reason is simple; there were much less sequences than there are now.

Solution:

Since we have a nucleotide sequence as our initial data set, we must go to the NCBI BLAST page, select the nucleotide blast program and search the nucleotide database using a nucleotide query. The page will look like this:

This BLASTN interface is easy to understand. It is divided into four major sections.

1. - The first allows the user to paste the query sequence for analysis. Our query sequence containing 30 nucleotides, highlighted in the red rectangle, was pasted into the search box.
2. - The second section permits the choice of a database to be searched and optional sequence range coordinates to the search. We will search the non-redundant (nr) nucleotide (nt) database in this exercise. Note that this is no longer the default option. That is why it is highlighted in yellow.
3. - The third section offers optimization alternatives to the search. We will use standard parameters and settings for our search. We will search for highly similar sequences using megablast.

4. - This section shows a summary of our search features that were defined in the previous sections. We will now mark the option to show the results in a new window. This option is very useful, since it allows us to re-run BLAST searches with different sequences and/or parameters while maintaining the former results.

Now we are ready to perform our search, just click on BLAST icon. After a few seconds the results page will appear in a new window. Assuming that you are familiar with a BLAST output, only the part of the output list of hits and alignments will be illustrated here.

Output list of hits:

Legend for links to other resources: [U](#) UniGene [E](#) GEO [G](#) Gene [S](#) Structure [M](#) Map Viewer

Sequences producing significant alignments:  
(Click headers to sort columns)

Accession	Description	Max score	Total score	Query coverage	E value	Max ident	Links
<a href="#">CP000717.1</a>	<i>Mycobacterium tuberculosis</i> F11, complete genome	60.0	60.0	100%	2e-07	100%	
<a href="#">CP000611.1</a>	<i>Mycobacterium tuberculosis</i> H37Ra, complete genome	60.0	60.0	100%	2e-07	100%	
<a href="#">AM408590.1</a>	<i>Mycobacterium bovis</i> BCG Pasteur 1173P2, complete genome	60.0	60.0	100%	2e-07	100%	
<a href="#">AF000516.2</a>	<i>Mycobacterium tuberculosis</i> CDC1551, complete genome	60.0	60.0	100%	2e-07	100%	
<a href="#">BX842576.1</a>	<i>Mycobacterium tuberculosis</i> H37Rv complete genome: segment 5/13	60.0	60.0	100%	2e-07	100%	
<a href="#">BX248339.1</a>	<i>Mycobacterium bovis</i> subsp. <i>bovis</i> AF2122/97 complete genome: sear	60.0	60.0	100%	2e-07	100%	
<a href="#">U02492.1</a>	<i>Mycobacterium tuberculosis</i> H37Rv isoniazid and ethionamide target p	60.0	60.0	100%	2e-07	100%	<a href="#">G</a>
<a href="#">U41388.1</a>	<i>Mycobacterium bovis</i> putative ketoacyl ACP reductase and enoyl ACP	60.0	60.0	100%	2e-07	100%	
<a href="#">U66801.1</a>	<i>Mycobacterium tuberculosis</i> 3-ketoacyl reductase (fabG) gene, compl	58.0	58.0	96%	8e-07	100%	<a href="#">G</a>
<a href="#">AC122924.2</a>	<i>Mus musculus</i> BAC clone RP23-39M16 from 12, complete sequence	38.2	38.2	76%	0.76	95%	
<a href="#">AL772249.6</a>	Mouse DNA sequence from clone RP23-430H1 on chromosome 2 Cont	38.2	38.2	63%	0.76	100%	
<a href="#">AC200520.3</a>	<i>Pongo pygmaeus abelii</i> BAC clone CH276-247I19 from chromosome 1	36.2	36.2	60%	3.0	100%	
<a href="#">AC198645.4</a>	<i>Pongo pygmaeus abelii</i> BAC clone CH276-79N12 from chromosome 8	36.2	36.2	60%	3.0	100%	
<a href="#">CP000454.1</a>	<i>Arthrobacter</i> sp. FB24, complete genome	36.2	36.2	73%	3.0	95%	
<a href="#">AC159018.3</a>	<i>Pan troglodytes</i> BAC clone CH251-533F3 from chromosome unknown.	36.2	36.2	60%	3.0	100%	
<a href="#">AC097467.1</a>	<i>Homo sapiens</i> BAC clone RP11-27G13 from 4, complete sequence	36.2	36.2	73%	3.0	95%	
<a href="#">AC158591.2</a>	<i>Pan troglodytes</i> BAC clone CH251-526G18 from chromosome unknow	36.2	36.2	60%	3.0	100%	
<a href="#">AP001360.5</a>	<i>Homo sapiens</i> genomic DNA, chromosome 11 clone:RP11-831A10, co	36.2	36.2	60%	3.0	100%	

You may notice that only the first 11 hits present very low E-values (below  $10^{-5}$ ), which are most possibly significant. If we wish to be more stringent, demanding a 100% coverage in the sequence alignment, only the first eight hits are relevant for our search and the characterization of our sequence.

The first six hits are related to complete genome sequences which makes it more difficult for us to identify our target sequence. We hope to find the putative gene directly or DNA sequences containing the putative gene. Hence, the only direct and easy options left are hits seven and eight.

This straightforward analysis have shown that the partial DNA sequence we are seeking to identify most likely is part of a gene belonging to either *Mycobacterium tuberculosis* or to *Mycobacterium bovis*, or both.

How to solve this possible ambiguity? We must now look carefully at the sequence alignments that follow the list of hits for these two hits we have selected. The results are:

```

> gb|U02492.1|MTU02492 G Mycobacterium tuberculosis H37Rv isoniazid and ethionamide target
protein (inhA) gene, complete cds
Length=832

Score = 60.0 bits (30), Expect = 2e-07
Identities = 30/30 (100%), Gaps = 0/30 (0%)
Strand=Plus/Plus

Query 1  TGACACAACACAAGGACGCACATGACAGGA  30
          |||
Sbjct 1  TGACACAACACAAGGACGCACATGACAGGA  30

> gb|U41388.1|MSU41388 Mycobacterium bovis putative ketoacyl ACP reductase and enoyl
ACP reductase (inhA) gene, complete cds
Length=1789

Score = 60.0 bits (30), Expect = 2e-07
Identities = 30/30 (100%), Gaps = 0/30 (0%)
Strand=Plus/Plus

Query 1  TGACACAACACAAGGACGCACATGACAGGA  30
          |||
Sbjct 904 TGACACAACACAAGGACGCACATGACAGGA  933

```

The alignment statistics is identical for both hits.

For the hit whose GenBank accession number is U02492.1, our query sequence begins exactly at the first position of the hit sequence.

For the hit whose GenBank accession number is U41388.1 the 5' end of our query sequence matches residue 904 on the hit sequence.

If you click on the accession number links, you will be redirected to the corresponding annotation of these sequences. As suggested by the annotation, the second one is a much longer nucleotide sequence (1789 bp) containing not only one, but two putative genes, one of them starting at base 904 which is similar to that found in the first hit. Also, the second hit corresponds to genes that are putative, that is, with no experimental confirmation.

From these analyses we can now proceed to the next question.

2 – Does the partial DNA sequence above correspond to a gene? If so, what is its GenBank accession number?

Yes. According to the analyses in 1, the 30-nucleotide partial DNA sequence most likely corresponds to a confirmed gene. Its accession number is best represented by U02492.1.

3 – What is the gene name?

The gene name can be obtained from the annotation in the hit list or by searching the NCBI nucleotide database with the GenBank accession number U02492.1. Another way is to click on the link associated with the entry U02492.1 with the mouse right hand button and open it in a new window. The top of the resulting page will be as follow:

The screenshot shows the NCBI Nucleotide database interface. The search results for U02492.1 are displayed. The entry is for the *inhA* gene from *Mycobacterium tuberculosis* H37Rv. The feature table shows the following information:

LOCUS	MTU02492	832 bp	DNA	linear	BCT 26-JAN-1994
DEFINITION	Mycobacterium tuberculosis H37Rv isoniazid and ethionamide target protein ( <i>inhA</i> ) gene, complete cds.				
ACCESSION	U02492				
VERSION	U02492.1 GI:407313				
KEYWORDS	-				
SOURCE	Mycobacterium tuberculosis H37Rv				
ORGANISM	<i>Mycobacterium tuberculosis</i> H37Rv				
	Bacteria; Actinobacteria; Actinobacteridae; Actinomycetales; Corynebacterineae; Mycobacteriaceae; Mycobacterium; Mycobacterium tuberculosis complex.				
REFERENCE	1 (bases 1 to 832)				
AUTHORS	Banerjee, A., Dubnau, E., Quemard, A., Belasubramanian, V., Um, K. S., Nilson, T., Collins, D., de Lisle, G. and Jacobs, W. R. Jr.				
TITLE	<i>inhA</i> , a gene encoding a target for isoniazid and ethionamide in <i>Mycobacterium tuberculosis</i>				
JOURNAL	Science 263 (5144), 227-230 (1994)				
PUBMED	<a href="#">5284673</a>				
REFERENCE	2 (bases 1 to 832)				
AUTHORS	Banerjee, A.				
TITLE	Direct Submission				
JOURNAL	Submitted (11-OCT-1993) Aash Banerjee, Microbiology and Immunology, Albert Einstein College of Medicine, 1300 Morris Park Avenue, Bronx, NY 10461, U.S.A				
FEATURES	Location/Qualifiers				
source	1..832				
	/organism="Mycobacterium tuberculosis H37Rv"				
	/mol_type="unassigned DNA"				
	/strain="H37Rv"				
	/db_xref="taxon:83332"				
	/clone="pYUB 315"				
	/clone_lib="MTB library by L. Pascopella"				
<a href="#">CDS</a>	11..16				
<a href="#">ORF</a>	22..831				
	/gene="inhA"				

Looking carefully at this GenBank flat file we can obtain all the information about the nucleotide entry U02492.1. We can clearly see all over, but more specifically in the FEATURES section that the gene name is *inhA*.

4 – What is the gene size in base pairs?

In the FEATURE section, in the ninth line, the link gene shows that the *inhA* gene starts at base 22 and ends at base 831, thus totaling 810 base pairs.

5 – Which organism(s) does the gene belong to?

The analyses showed that the gene can be found in both bacteria *M. tuberculosis* and *M. bovis*.

### 5.1.2 Gene Product or Protein Characterization

6 – How many amino acids does the gene product contain?

Again, looking carefully at the U02492.1 entry, we can easily find the answer to this question or by a simple mathematical calculation. For example, if the gene contains 810 bp, the last three bases compose the stop codon, so the protein coding region is 807 bp long. After dividing this number by three (the number of bases in a codon) we obtain 269 amino acids. However, very often the annotation information is not available, and the only information we have is the nucleotide sequence. Thus, how can we find out the protein sequence and, consequently, its corresponding number of amino acids?

We need to translate our *inhA* gene sequence into its corresponding protein. There are several different ways to perform this task. We will use one of them, a tool called *Translate* (<http://us.expasy.org/tools/dna.html>), which

can be accessed through the ExPASy web site. This tool translates a DNA (RNA) sequence into its six (three in each direction) different possible open reading frames (ORFs). The *Translate* interface looks like this:

ExPASy Home page Site Map Search ExPASy Contact us Proteomics tools

Search Swiss-Prot/TrEMBL for Go Clear

## Translate tool

Translate is a tool which allows the translation of a nucleotide (DNA/RNA) sequence to a protein sequence.

Please enter a DNA or RNA sequence in the box below (numbers and blanks are ignored).

```
TGACACAACACAAGGACGCACATGACAGGACTGCTGGACGGCAAACGGATTCTGGTTAGCGGAATCATCA
CCGACTCGTCGATCGCGTTTCACATCGCACGGGTAGCCCAAGGAGCAGGGCGCCCAAGCTGGTGTCAACGG
GTTGACCCGGCTGCGGCTGATTCAGCGCATCACGACCGGCTGCCGGCAAAGCCCCGCTGCTCGAACTC
GACGTGCAAAACGAGGAGCACCTGGCCAGCTTGGCCGCGCGGGTGACCGAGCGATCGGGCCGGCAACA
AGCTCGACGGGGTGGTGCATTGATTGGGTTTCATGCCGACAGACCGGGATGGGCATCAACCCGTTCTTCGA
CGCGCCCTACCGGGATGTGTCCAAGGGCATCCACATCTCGGCGTATTGATGCTTCGATGGCCAAGGGC
CTGCTGCCGATCATGAACCCCGGAGGTTCCATCGTCGSCATGGACTTCGACCCGAGCCGGCGATGCCGG
CCTACAACCTGGATGACGGTCGCCAAGAGCCGTTGGAGTCGGTCAACAGGTTGTTGGCCGCGAGGCCGG
CAAGTACGGTGTGCGTTTCAATCTCGTTGCCGACGGCCCTATCCGGACGCTGGCGATGAGTGGATCGTC
GGCGGTGCGCTCGGCGAGGAGGCCGGCCAGATCCAGCTGCTCGAGGAGGGCTGGGATCAGCGCGCTC
CGATCGGCTGGAAATGAAGGATGCGACGCCGGTCCCAAGACGGTGTGCGCGCTGCTGTGACTGGCT
GCCGGGACCAACCGGTTGACATCATCTACGCCGACGGCGGCGCACACCAATTGCTCTAGA
```

Output format: Verbose ("Met", "Stop", spaces between residues)

Reset or TRANSLATE SEQUENCE

Our nucleotide sequence, corresponding to the GenBank entry U02492.1, was copied and pasted for analysis (red rectangle). After clicking on TRANSLATE SEQUENCE we obtain:

## Translate Tool - Results of translation

Please select one of the following frames:

### [5' Frame 1](#)

Stop H N T R T H M **Met** T G L L D G K R I L V S G I I T D S S I A F H I A R V A Q E Q G A Q L V L T G F D R L R L I Q R I T D R L P A K A P L L E L D V Q  
N E E H L A S L A G R V T E A I G A G N K L D G V V H S I G F **Met** P Q T G **Met** G I N P F F D A P Y A D V S K G I H I S A Y S Y A S **Met** A K A L L P I  
**Met** N P G G S I V G **Met** D F D P S R A **Met** P A Y N W **Met** T V A K S A L E S V N R F V A R E A G K Y G V R S N L V A A G P I R T L A **Met** S A I V G  
G A L G E E A G A Q I Q L L E E G W D Q R A P I G W N **Met** K D A T P V A K T V C A L L S D W L P A T T G D I I Y A D G G A H T Q L L **Stop** X X X X  
X X X X X

### [5' Frame 2](#)

D T T Q G R T **Stop** Q D C W T A N G F W L A E S S P T R R S R F T S H G **Stop** P R S R A P S W C S P G S T G C G **Stop** F S A S P T G C R Q R P R C S N  
S T C K T R S T W P A W P A G **Stop** P R R S G R A T S S T G W C I R L G S C R R P G W A S T R S S T R P T R **Met** C P R A S T S R R I R **Met** L R W P R  
R C C R S **Stop** T P E V P S S A W T S T R A G R C R P T T G **Stop** R S P R A R W S R S T G S W R A R P A S T V C V R I S L P Q A L S G R W R **Stop** V R S  
S A V R S A R R P A R S S C S R R A G I S A L R S A G T **Stop** R **Met** R R R S P R R C A R C C L T G C R R P R V T S S T P T A A R T P N C S R X X X X  
X X X X

### [5' Frame 3](#)

T Q H K D A H D R T A G R Q T D S G **Stop** R N H H R L V D R V S H R T G S P G A G R P A G A H R V R P A A A D S A H H R P A A G K G P A A R T R R  
A K R G A P G Q L G R P G D R G D R G G Q Q A R R G G A F D W V H A A D R D G H Q P V L R R A L R G C V Q G H P H L G V F V C F D G Q G A A A  
D H E P R R F H R R H G L R P E P G D A G L Q L D D G R Q E R V G V G Q Q V R G A R G R Q V R C A F E S R C R R P Y P D A G D E C D R R R C A R R  
G R R P D P A A R G G L G S A R S D R L E H E G C D A G R Q D G V R A A V **Stop** L A A G D H G **Stop** H H L R R R R R A H P I A L X X X X X X X X X X

### [3' Frame 1](#)

X X X X X X S X S R A I G C A R R R R R R **Stop** C H P W S P A A S Q T A A R T P S W R P A S H P S C S S R S E R A D P S P P R A A G S G R R P P R R A  
H R R R S H S S P A S G **Stop** G L R Q R D S N A H R T C R P R A P R T C **Stop** P T P T R S W R P S S S C R P A S P G S G R S P C R R W N L R G S **Stop** S  
A A A P W S K H T N T P R C G C P W T H P R R A R R R T G **Stop** C P S R S A A **Stop** T Q S N A P P R R A C C P P R S P R S P G R P S W P G A P R F A  
R R V R A A G P L P A A G R **Stop** C A E S A A A G R T R **Stop** A P A G R P A P G L P V R C E T R T S R **Stop** **Stop** F R **Stop** P E S V C R P A V L S C A S  
L C C V

### [3' Frame 2](#)

X X X X X X X L E Q L G V R A A V G V D D V T R G R R Q P V R Q R A H R L G D R R R I L H V P A D R S A L I P A L L E Q L D L G A G L L A E R T  
A D R T H R Q R P D R A C G N E I R T H T V L A G L A R H E P V D R L Q R A L G D R H P V V G R H R P A R V E V H A D D G T S G V H D R Q Q R L  
G H R S I R R D V D A L G H I R V G R V E E R V D A H P G L R H E P N R **Met** H H P V E L V A R P D R L G H P A G Q A G Q V L L V L H V E F E Q R  
G L C R Q P V G D A L N Q P Q P V E P G E H Q L G A L L L G Y P C D V K R D R R V G D D S A N Q N P F A V Q Q S C H V R P C V V S

### [3' Frame 3](#)

X X X X X I X X **Stop** S N W V C A P P S A **Stop** **Met** **Met** S P V V A G S Q S D S S A H T V L A T G V A S F **Met** F Q P I G A R **Stop** S Q P S S S S W I W A  
P A S S P A P T I A I A S V R I G P A A T R F E R T P Y L P A S R A T N L L T D S N A L L A T V I Q L **Stop** A G I A R L G S K S **Met** P T **Met** E P P G  
F **Met** I G S S A L A I E A Y E Y A E **Met** W **Met** P L D T S A **Stop** G A S K N G L **Met** P I P V C G **Met** N P I E C T T P S S L L P A P I A S V T R P A K L A  
R C S S F C T S S S S G A F A G S R S V **Met** R **Stop** I S R S R S N P V S T W A P C S W A T R A **Met** **Stop** N A I D E S V **Met** I P L T R I R L P S S S P  
V **Met** C V L V L C

Our task now is to find out which of the six frames corresponds to the *inhA* gene product. In other words, we are asking what is the correct ORF. Often the correct ORF is the longest amongst the six possible ones. If we follow this assumption (which not always is correct!), the longest ORF amongst the six found by the *Translate* tool ORF 1 (5'-3'Frame 1) is the right answer. Clicking over the link 5'-3'Frame 1 will show us the conceptual translation in one-letter amino acid code, as follows:

## Translate Tool

Please select one of the "Methionine" or one of the highlighted residues following a **Stop** codon (or the beginning of the sequence).

This will create a virtual Swiss-Prot entry, comprising the residues from your chosen start position up to the following **Stop** codon.

HNTRTH**M**TGLLDGKRILVSGIITDSSIAFHARVAQEQGAQLVLTGFDRRLRIQRITDRLPAKAPLLELDVQ  
EEHLASLAGRVTEAIGAGNKLDGVVHSIGF**M**PQTGMGINPFFDAPYADVSKGIHISAYSASYAS**M**AKALLPIMN  
PGGSIVGMDFDPSRAMPAYNWM**T**VAKSALESVNRFFVAREAGKYGVRSNLVAAGPIRTLAM**S**AIVGGALGE  
EAGAQLLEEWDQRAPIGW**N**MKDATPVAKTVCALLSDWLPATTGDIIYADGGAHTQLL **Stop**

[ExPASy Home page](#) [Site Map](#) [Search ExPASy](#) [Contact us](#) [Proteomics tools](#)

Once identified, the ORF can be translated into its corresponding protein or amino acid sequence. In most prokaryote species, an ORF starts with an ATG coding for methionine (Met or M) and ends with a stop codon (TAA, TAG or TGA). Now, clicking on the first Met residue will give us our protein sequence in a format similar to the GenBank flatfile, including the number of amino acids. In this case, 269 (see figure below).

## Virtual Sequence: VIRT25514

```

ID   VIRT25514                Unreviewed;          269 AA.
AC   VIRT25514;
DE   Translation of nucleotide sequence generated on ExPASy
DE   on 06-Aug-2007 by 201.54.129.80.
CC   -!- This virtual protein sequence will automatically be deleted
CC   from the server after a few days.
CC   5.73 PI.
DR   SWISS-2DPAGE; VIRT25514; VIRTUAL.
SQ   SEQUENCE 269 AA; 28528 MW; F161D6D6A631CA08 CRC64.
      MTGLLDGKRI LVSGIITDSS IAFHIARVAQ EQGAQLVLTG FDRLRLIQRI TDRLPAKAPL
      LELDVQNEEH LASLAGRVTE AIGAGNKLDG VVHSIGFMPQ TGMGINPFFD APYADVSKGI
      HISAYSYASM AKALLPIMNP GGSIVGMDFD PSRAMPAYNW MTVAKSALES VNRFVAREAG
      KYGVRNLVA AGPIRTLAMS AIVGGALGEE AGAQIQLLEE GWDQRAPIGW NMKDATPVAK
      TVCALLSDWL PATTGDIIYA DGGHAHTQLL
//

```

If we continue and click on the *FASTA format* link at the bottom of the page, we obtain our protein sequence in the the FASTA format, ready to be used by other programs.

```

>VIRT25514|VIRT25514 Translation of nucleotide sequence generated on ExPASy on 06-Aug-2007 by 201.54.129.80
MTGLLDGKRILVSGIITDSSIAFHARVAQEQGAQLVLTGFDRRLRLIQRITDRLPAKAPL
LELDVQNEEHLASLAGRVTEAIGAGNKLDGVVHSIGFMPQ TGMGINPFFDAPYADVSKGI
HISAYSYASMAKALLPIMNPGGSIVGMDFDPSRAMPAYNWMTVAKSALESVNRFVAREAG
KYGVRNLVAAGPIRTLAMSAIVGGALGEEAGAQIQLLEEGWDQRAPIGWNMMDATPVAK
TVCALLSDWLPATTGDIIYADGGHAHTQLL

```

7 – What is the gene product name and possible function?

So far we have worked with a DNA sequence. Now we will query the NCBI protein databases with the protein sequence obtained above and discover its possible function. For this purpose, we will go back to the basic BLAST page at NCBI (<http://www.ncbi.nlm.nih.gov/BLAST/>) but, at this time, we will select the link "protein blast", which will lead us to the BLASTp page. The BLASTp page at NCBI will look like this:

The screenshot shows the NCBI BLAST web interface. The top navigation bar includes 'Home', 'Recent Results', 'Saved Strategies', and 'Help'. The main content area is titled 'Enter Query Sequence' and contains a text input field with a protein sequence: `MTGLDGRILVSGIITDSEIAPNARVAQGGQQLVLTGFDRLRLIQRITDRLPAKAPL  
LELDVQNEEHLASLAGRVTEAIGAGNKLDVYHSIGFMPQTGMGINDFFDAPYADVSEGI  
HISAYSASHAKALLPIMFPGGSIVQMDFFSRAMPATNRMIVAKSALESYVNFVAREAG  
KYGVRSHLVAAGPVRTLMSAIVGGALGELTLAGAQIQLEEGHQRAPIGWRRKDATPVAK  
IVCALLSENLPAITGDIIVADGGHTQLL`. This sequence is highlighted in yellow. Below the input field are options for 'Or, upload file' and 'Job Title'. The 'Choose Search Set' section has a dropdown menu for 'Database' set to 'Non-redundant protein sequences (nr)'. The 'Program Selection' section has radio buttons for 'blastp (protein-protein BLAST)', 'PSI-BLAST (Position-Specific Iterated BLAST)', and 'PHI-BLAST (Pattern Hit Initiated BLAST)'. The 'blastp' option is selected. At the bottom, there is a 'BLAST' button and a checkbox for 'Show results in a new window'. A note at the bottom right states: 'Note: Parameter values that differ from the default are highlighted in yellow'.

Our query sequence is already inserted in the search box. BLASTp standard parameters for protein-protein comparison are being used such as the substitution matrix BLOSUM62 and the nr database. For further details click in [Algorithm parameters](#) link at the bottom of the page.

The output list of results is shown for the top-21 hits:



Sequences producing significant alignments:		Score (Bits)	E Value	
<a href="#">ref NP_216000.1 </a>	enoyl-(acyl carrier protein) reductase [Myco...	<a href="#">519</a>	7e-146	<b>G</b>
<a href="#">pdb 2AQH A</a>	Chain A, Crystal Structure Of Isoniazid-Resistant ...	<a href="#">518</a>	1e-145	<b>S</b>
<a href="#">gb AA54545.1 </a>	InhA [Mycobacterium tuberculosis]	<a href="#">518</a>	1e-145	
<a href="#">pdb 2AQI A</a>	Chain A, Crystal Structure Of Isoniazid-Resistant ...	<a href="#">517</a>	3e-145	<b>S</b>
<a href="#">gb AAD26114.1 </a>	mutant NADH-dependent 2-trans enoyl-acyl carri...	<a href="#">517</a>	3e-145	
<a href="#">pdb 1BVR A</a>	Chain A, M.Tb. Enoyl-Acp Reductase (InhA) In Compl...	<a href="#">517</a>	3e-145	<b>S</b>
<a href="#">pdb 2AQK A</a>	Chain A, Crystal Structure Of Isoniazid-Resistant ...	<a href="#">517</a>	3e-145	<b>S</b>
<a href="#">pdb 1ENY </a>	Chain , Structural Genomics, Psi, Protein Structu...	<a href="#">514</a>	2e-144	<b>S</b>
<a href="#">pdb 1ENZ </a>	Chain , Structural Genomics, Psi, Protein Structu...	<a href="#">512</a>	7e-144	<b>S</b>
<a href="#">ref YP_905485.1 </a>	enoyl-[acyl-carrier-protein] reductase, InhA...	<a href="#">496</a>	8e-139	<b>G</b>
<a href="#">ref YP_882476.1 </a>	[NADH] enoyl-[acyl-carrier-protein] reductas...	<a href="#">473</a>	4e-132	<b>G</b>
<a href="#">ref NP_960144.1 </a>	enoyl-(acyl carrier protein) reductase [Myco...	<a href="#">473</a>	7e-132	<b>G</b>
<a href="#">ref NP_302227.1 </a>	enoyl-(acyl carrier protein) reductase [Myco...	<a href="#">472</a>	1e-131	<b>G</b>
<a href="#">pdb 2NTV A</a>	Chain A, Mycobacterium Leprae Inha Bound With Pth-...	<a href="#">470</a>	4e-131	<b>S</b>
<a href="#">ref YP_887466.1 </a>	[NADH] enoyl-[acyl-carrier-protein] reductas...	<a href="#">468</a>	1e-130	<b>G</b>
<a href="#">emb CAD27905.1 </a>	inhA protein [Mycobacterium avium subsp. paratub...	<a href="#">467</a>	3e-130	
<a href="#">ref YP_001134919.1 </a>	Enoyl-(acyl-carrier-protein) reductase (N...	<a href="#">466</a>	6e-130	<b>G</b>
<a href="#">ref YP_639621.1 </a>	Enoyl-(acyl-carrier protein) reductase (NADH...	<a href="#">464</a>	3e-129	<b>G</b>
<a href="#">ref YP_953566.1 </a>	Enoyl-(acyl-carrier-protein) reductase (NADH...	<a href="#">461</a>	2e-128	<b>G</b>
<a href="#">gb ABK54061.1 </a>	NADH-dependent enoyl-acyl-carrier-protein redu...	<a href="#">454</a>	2e-126	
<a href="#">sp O07400 INHA MYCAV</a>	Enoyl-[acyl-carrier-protein] reductase [...	<a href="#">438</a>	2e-121	

The bit score and the  $E$  values indicate that we found our protein. Moreover, the red squares on the right with the letter "S" inside indicate that our proteins most likely have a 3D structure (S) which will help to answer the next questions.

To increase our confidence that the hits correspond to homologs of our query protein, we must check if the alignment covers the whole query protein sequence. The figure below shows the first alignment in the list of results.

Score = 519 bits (1336), Expect = 7e-146, Method: Composition-based stats. Identities = 269/269 (100%), Positives = 269/269 (100%), Gaps = 0/269 (0%)					
Query	1	MTGLLDGKRILVSGIITDSSIAFHARVAQEQGAQLVLTGFDRRLRIQRITDRLPAKAPL	60		
Sbjct	1	MTGLLDGKRILVSGIITDSSIAFHARVAQEQGAQLVLTGFDRRLRIQRITDRLPAKAPL	60		
Query	61	LELDVQNEEHLASLAGRVTEAIGAGNKLDGVVHSIGFMPQTMGINPFFDAPYADVSKGI	120		
Sbjct	61	LELDVQNEEHLASLAGRVTEAIGAGNKLDGVVHSIGFMPQTMGINPFFDAPYADVSKGI	120		
Query	121	HISAYSYASMAKALLPIMNPGGSIVGMDFDPSRAMPAYNWMTVAKSALESVNRVAREAG	180		
Sbjct	121	HISAYSYASMAKALLPIMNPGGSIVGMDFDPSRAMPAYNWMTVAKSALESVNRVAREAG	180		
Query	181	KYGVRNLVAAGPIRTLAMSAIVGGALGEEAGAQIQLLEEGWDQRAPIGWNMKDATPVAK	240		
Sbjct	181	KYGVRNLVAAGPIRTLAMSAIVGGALGEEAGAQIQLLEEGWDQRAPIGWNMKDATPVAK	240		
Query	241	TVCALLSDWLPATTGDIIYADGGAHTQLL	269		
Sbjct	241	TVCALLSDWLPATTGDIIYADGGAHTQLL	269		

Inspecting the annotation, we can see the our query protein is called Enoyl (Acyl Carrier Protein or ACP) Reductase of *Mycobacterium tuberculosis* strain H37Rv or InhA. To obtain a more complete answer we can access the protein annotation through its GenBank accession number NP\_21600, similarly as we did with the gene accession number. The FEATURES section of the resulting archive is:

```

FEATURES             Location/Qualifiers
    source            1..269
                     /organism="Mycobacterium tuberculosis H37Rv"
                     /strain="H37Rv"
                     /db_xref="taxon:83332"
    Protein           1..269
                     /product="enoyl-(acyl carrier protein) reductase"
                     /EC_number="1.3.1.9"
                     /function="THIS ISOZYME IS INVOLVED IN MYCOLIC ACID
                     BIOSYNTHESIS. SECOND REDUCTIVE STEP IN FATTY ACID
                     BIOSYNTHESIS. INVOLVED IN THE RESISTANCE AGAINST THE
                     ANTITUBERCULOSIS DRUGS ISONIAZID AND ETHIONAMIDE
                     [CATALYTIC ACTIVITY: ACYL-[ACYL-CARRIER PROTEIN] + NAD(+)
                     = TRANS-2,3-DEHYDROACYL-[ACYL-CARRIER PROTEIN] + NADH]."
                     /calculated_mol_wt=28397
    Region           1..269
                     /region_name="PRK07889"
                     /note="enoyl-(acyl carrier protein) reductase; PRK07889"
                     /db_xref="CDD:76326"
    Region           2..269
                     /region_name="FabI"
                     /note="Enoyl-[acyl-carrier-protein]; COG0623"
                     /db_xref="CDD:30968"
    CDS              1..269
                     /gene="inhA"
                     /locus_tag="Rv1484"
                     /coded_by="NC_000962.2:1674202..1675011"
                     /experiment="experimental evidence, no additional details
                     recorded"
                     /note="Catalyzes a key regulatory step in fatty acid
                     biosynthesis"
                     /transl_table=11
                     /db_xref="GeneID:886523"

```

Since *M. tuberculosis* is a bacterium, this enzyme most likely functions in the cytoplasm and is involved in the biosynthesis of mycolic acid, an essential component of its cell wall. The blue links of the qualifiers can provide the curious reader with even more information about this enzyme.

8 – Is this gene present in humans? What consequences could this fact have for a possible application of gene product for a structure-based drug discovery?

To answer this question, you can run a BLASTp similarity search of this protein sequence against a database of human RefSeq proteins. Also, it is necessary to read about the fatty acid biosynthesis in humans and bacteria. A quick answer is: no, it is not present in humans. Hence, it is theoretically an ideal drug target against tuberculosis. In fact, the reader will find out that this enzyme has been proven to be a *bonafide* target of the drug isoniazid (INH).

9 – Does the gene product have a 3D structure? If so, what is its RCSB/PDB identification number? Describe the gene product architecture.

As we can see from the BLASTp output above, there are now several 3D structures for the InhA enzyme, enzyme-NADH complex, tertiary complexes involving drug candidates, and with the NADH-INH adduct which inhibits the enzyme. Historically, the first InhA structure to be determined was that with PDB ID 1ENY (the 8<sup>th</sup> entry in BLASTp's hit list), so we will investigate this particular one. To see the 3D structure, you should now go to PDBs search page (<http://www.rcsb.org/pdb/home/home.do>). To do this, enter the PDB code of the protein (1ENY) in the search bar at the top of the page and click the button "Site Search". The initial page of the result will look as follows:

RCSB PDB PROTEIN DATA BANK

A MEMBER OF THE PDB

An Information Portal to Biological Macromolecular Structures

As of Tuesday Aug 07, 2007 there are 45055 Structures | PDB Statistics

CONTACT US | HELP | PRINT PAGE

PDB ID or keyword Author 1ENY Site Search Advanced Search

Home Search Structure Queries

Are you missing data updates? The PDB archive has moved to <ftp://ftp.wwpdb.org>. For more information click here.

Some chains and/or residues have been updated. Click here for details, or here for details about the remediation process

Help Structure Summary Biology & Chemistry Materials & Methods Sequence Details Geometry Remediation

1ENY DOI 10.2210/pdb1eny/pdb

Rad - Derived Information

Title CRYSTAL STRUCTURE AND FUNCTION OF THE ISONIAZID TARGET OF MYCOBACTERIUM TUBERCULOSIS

Authors Dessen, A., Quemard, A., Blanchard, J.S., Jacobs Jr., W.R., Sacchettini, J.C., TB Structural Genomics Consortium (TBSGC)

Primary Citation Dessen, A., Quemard, A., Blanchard, J.S., Jacobs Jr., W.R., Sacchettini, J.C. Crystal structure and function of the isoniazid target of Mycobacterium tuberculosis. Science v267 pp 1638-1641, 1995 [Abstract]

History Deposition 1995-01-27 Release 1996-01-29

Experimental Method Type X-RAY DIFFRACTION Data N/A

Parameters	Resolution(Å)	R-value	R-Free	Space Group
	2.20	0.196 (obs.)	n/a	P 6 <sub>2</sub> 2 2

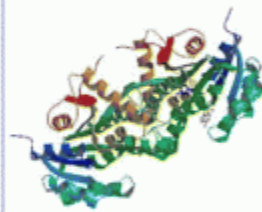
Unit Cell

Length [Å]	a	100.14	b	100.14	c	140.45
Angles [°]	alpha	90.00	beta	90.00	gamma	120.00

Molecular Description Asymmetric Unit Polymer: 1 Molecule: ENOYL-ACYL CARRIER PROTEIN (ACP) REDUCTASE Chains: A

Images and Visualization

Biological Molecule



Display Options

- KING
- Jmol
- WebMol
- MBT SimpleViewer\*
- MBT Protein Workshop
- QuickPDB
- All Images

\* Capable of displaying biological molecules.

Classification Oxidoreductase

Source Polymer: 1 Scientific Name: **Mycobacterium tuberculosis** Expression system: **Escherichia coli**

Ligand Chemical Component	Identifier	Name	Formula	Drug Similarity	Hapten Similarity	Ligand Structure	Ligand Interaction
NAD	NICOTINAMIDE-ADENINE-DINUCLEOTIDE	C <sub>21</sub> H <sub>27</sub> N <sub>7</sub> O <sub>14</sub> P <sub>2</sub>	[View] [View]				

SCOP Classification (version 1.71)

Domain Info	Class	Fold	Superfamily	Family	Domain	Species
d1eny_	Alpha and beta proteins (a/b)	NAD(P)-binding Rossmann-fold domains	NAD(P)-binding Rossmann-fold domains	Tyrosine-dependent oxidoreductases	Enoyl-ACP reductase	Mycobacterium tuberculosis, TB, gene InhA

CATH Classification (version v2.1.0)

Domain	Class	Architecture	Topology	Homology
1eny000	Alpha Beta	3-Layer(aba) Sandwich	Rossmann fold	NAD(P)-binding Rossmann-like Domain

GO Terms

Polymer	Molecular Function	Biological Process	Cellular Component
ENOYL-ACYL CARRIER PROTEIN (ACP) REDUCTASE (1ENY:A)	• none	• none	• none

© RCSB Protein Data Bank

We can obtain all the available information about the 3D structure of this enzyme by browsing through the links or download the PDB file to a local directory in our computer and work with our preferred molecular modeling and visualization package. For instance, we can see in the classification section that this enzyme has a 3-layer ( $\alpha\beta\alpha$ ) sandwich architecture according to the CATH classification. We will visualize this next.

Some of the visualization software can be accessed directly from the page illustrated above. They can be located in the bottom of the section "Images and Visualization". Another alternative is the site *First Glance in Jmol* (<http://molvis.sdsc.edu/fgj/index.htm>). Go to the site, enter 1ENY in the blank box and click the Submit button:



**FirstGlance in Jmol**  
A simple tool for macromolecular visualization. ([More...](#))

Enter a **PDB identification code** here:

Example: enter **1d66**

Or visit the [Gallery of Interactive Molecules](#) or the [Snapshots Gallery](#).

Or enter a [molecule's URL](#), or [upload your own PDB file](#).

Use <http://firstglance.jmol.org> for links and bookmarks. [More...](#)  
FirstGlance may display from other URL's, but these are *not guaranteed to be permanent!*

---

[What Is FirstGlance in Jmol?](#)  
[How to show a molecule in FirstGlance from your website.](#)  
[All About FirstGlance in Jmol version 1.0.](#)  
Troubleshooting, copyright, licenses, acknowledgements, design & history, technical information, etc.

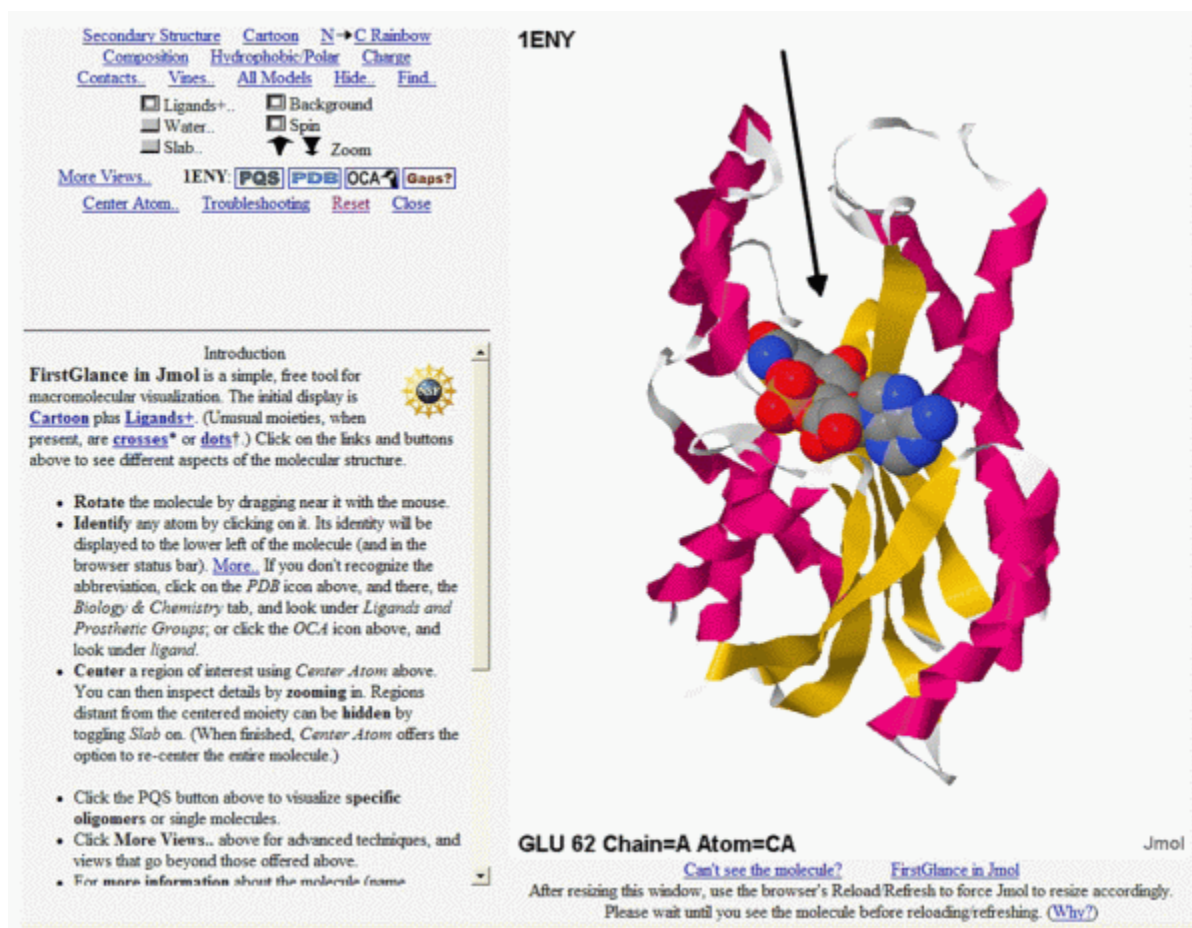
[Adoptions](#) [Mirror Sites](#)  
[Advanced Options](#)

---

Feedback to [the FirstGlance Development Team](#)

As the name suggests, we can do many simple visualization manipulations that will help us to understand the molecule structure, interactions and function. Note: the use of molecular visualization software, no matter how simple it is, demands at least a general knowledge of the object to be visualized. Hence, to be able to run this software, the user must understand the basic principles of protein structure and function.

After submitting the 1ENY PDB code, and waiting a little bit to load the JAVA machine, we obtain:



[Secondary Structure](#) [Cartoon](#) [N→C Rainbow](#)  
[Composition](#) [Hydrophobic/Polar](#) [Charge](#)  
[Contacts](#) [Vines](#) [All Models](#) [Hide](#) [Find](#)

Ligands+..  Background  
 Water..  Spin  
 Slab..

[More Views..](#) 1ENY: [PQS](#) [PDB](#) [OCA](#) [Gaps?](#)  
[Center Atom..](#) [Troubleshooting](#) [Reset](#) [Close](#)

---

**Introduction**

**FirstGlance in Jmol** is a simple, free tool for macromolecular visualization. The initial display is **Cartoon plus Ligands+**. (Unusual moieties, when present, are **crosses\*** or **dots\***.) Click on the links and buttons above to see **different** aspects of the molecular structure.

- **Rotate** the molecule by dragging near it with the mouse.
- **Identify** any atom by clicking on it. Its identity will be displayed to the lower left of the molecule (and in the browser status bar). [More...](#) If you don't recognize the abbreviation, click on the *PDB* icon above, and there, the *Biology & Chemistry* tab, and look under *Ligands and Prosthetic Groups*; or click the *OCA* icon above, and look under *ligand*.
- **Center** a region of interest using *Center Atom* above. You can then inspect details by **zooming in**. Regions distant from the centered moiety can be **hidden** by toggling *Slab* on. (When finished, *Center Atom* offers the option to re-center the entire molecule.)
- Click the *PQS* button above to visualize **specific oligomers** or single molecules.
- Click *More Views..* above for advanced techniques, and views that go beyond those offered above.
- For **more information** about the molecule's name

**GLU 62 Chain=A Atom=CA** Jmol

[Can't see the molecule?](#) [FirstGlance in Jmol](#)  
After resizing this window, use the browser's Reload/Refresh to force Jmol to resize accordingly.  
Please wait until you see the molecule before reloading/refreshing. ([Why?](#))

Here the InhA structure appears static, but the site will initially display the image rolling in order to present an even better idea of its three-dimensionality. Pressing the left mouse button and moving it left and right and up and down allows the user to keep full control of the rotation of the molecule. Also, pressing the left mouse button at the same time as the control or alt keys, and moving the mouse, will change the zoom.

It's possible to visualize the molecule using different representations. Above, the enzyme backbone is represented as ribbons where  $\alpha$  helices and strands of the  $\beta$  sheet are colored red and yellow, respectively. Turns and loops are in grey. In this view angle, we can see  $\alpha$  helices at left, a central  $\beta$  sheet, and more  $\alpha$  helices at right. That is why this enzyme has an architecture of the 3-layer- $\alpha\beta\alpha$  sandwich type. The two  $\alpha$  helices layers are the "bread slices" and the  $\beta$  sheet, the sandwich's filling!

We can also observe the space-filling model of the NADH coenzyme colored according to the CPK rules (nitrogen in blue, carbon in grey, oxygen in red, phosphorus in orange). This coenzyme is the key to the InhA function.

In the figure, the added black arrow points to the position where the substrate will fit inside the enzyme. Drugs that inhibit this enzyme will most likely bind to this region.

10 – If the gene product does not have a 3D structure, would it be possible to infer its structure from the closest homologues with a known 3D structure? Explain the answer in details.

Remember, all the information we gathered so far started from a very small nucleotide sequence. Some basic knowledge of molecular biology and the usage of only a few of over 1200 bioinformatics websites are available in a Web Server Edition of the Nucleic Acids Research journal ([http://nar.oxfordjournals.org/content/vol35/suppl\\_2/index.shtml?etoc](http://nar.oxfordjournals.org/content/vol35/suppl_2/index.shtml?etoc)).

Fortunately, we found a 3D structure for our 30 nucleotide sequence. If our protein did not have a structure, we could use the principle of comparative or homology modeling to build a 3D structure for it. This principle states that if a protein sequence of unknown structure (target protein) has a sequence identity greater than or equal to 30% with another protein with known structure (template), over its whole length, then the target protein 3D structure can be modeled based on one or on a combination of several template molecules (26). This is possible because homolog proteins descend from a common ancestor and are likely to present the same structure and function. Caution note: This is correct in general, but there are exceptions though.

## References

- (1). Lesk, A. M. Introduction to Protein Architecture. OUP, Oxford, 2001.
- (2). Xie J., Schultz P. G. An expanding genetic code. *Methods*. 2005;36:226–238.
- (3). Anthony-Cahill S. J., Griffith M. C., Noren C. J., Suich D. J., Schultz P. G. Site-specific mutagenesis with unnatural amino acids. *Trends Biochem. Sci.* 1989;14:400–403. PubMed PMID: 2683258.
- (4). Ashburner M., Ball C.A., Blake J. A., Botstein D., Butler H., Cherry J. M., Davis A. P., Dolinski K., Dwight S. S., Eppig J. T., Harris M. A., Hill D. P., Issel-Tarver L., Kasarskis A., Lewis S., Matese J. C., Richardson J. E., Ringwald M., Rubin G. M., Sherlock G. Gene Ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nature Genet.* 2000;25:25–29. PubMed PMID: 10802651.
- (5). Cabeen M.T., Jacobs-Wagner C. Bacterial cell shape. *Nat. Rev. Microbiol.* 2005;3:601–610. PubMed PMID: 16012516.
- (6). Kostin S., Hein S., Arnon E., Scholz D., Schaper J. The Cytoskeleton and Related Proteins in the Human Failing Heart. *Heart Failure Reviews*. 2000;5:271–280. PubMed PMID: 16228910.
- (7). Kunzelmann K. Ion Channels and Cancer. *J. Membr. Biol.* 2005;205:159–73. PubMed PMID: 16362504.
- (8). Hille B., Armstrong C. M., MacKinnon R. Ion channels: from idea to reality. *Nat. Med.* 1999;5:1105–1109. PubMed PMID: 10502800.

- (9). Green M. R. Eukaryotic transcription activation: right on target. *Mol. Cell.* 2005;18:399–402. PubMed PMID: 15893723.
- (10). Bloom K. Chromosome segregation: seeing is believing. *Curr. Biol.* 15:R500–503.
- (11). Delves P. J., Roitt I. M. The immune system. Second of two parts. *N. Engl. J. Med.* 2000a;343:108–117. PubMed PMID: 10891520.
- (12). Delves P. J., Roitt I. M. The immune system. First of two parts. *N. Engl. J. Med.* 2000b;343:37–49. PubMed PMID: 10882768.
- (13). Luscombe N.M., Greenbaum D., Gerstein M. What is Bioinformatics? A proposed definition and overview of the field. *Method Inform. Med.* 2001;40:346–358.
- (14). Burley S. K., Almo S. C., Bonanno J. B., Capel M., Chance M. R., Gaasterland T., Lin D., Sali A., Studier F. W., Swaminathan S. Structural genomics: beyond the Human Genome Project. *Nat. Genet.* 1999;23:151–157. PubMed PMID: 10508510.
- (15). Liolios K., Tavernarakis T., Hugenholtz P., Kyrpides N. C. The Genomes On Line Database (GOLD) v.2: a monitor of genome projects worldwide. *Nucl. Acids Res.* 2006;34:D332–D334.
- (16). Galperin M. Y. The Molecular Biology Database Collection: 2006 update. *Nucl. Acids Res.* 2006;34:D3–D5.
- (17). Ouzonis C. A., Valencia A. Early bioinformatics: the birth of a discipline – a personal view. *Bioinformatics.* 2003;19:2176–2190. PubMed PMID: 14630646.
- (18). Benson D. A., Karsch-Mizrachi I., Lipman D. J., Ostell J., Wheeler D. L. GenBank. *Nucl. Acids Res.* 2006;34:D16–D20.
- (19). Kouranov A., Xie L., de la Cruz J., Chen L., Westbrook J, Bourne P. E., Berman H. M. The RCSB PDB information portal for structural genomics. *Nucl. Acids Res.* 2006;34:D302–D305.
- (20). Altschul S. F., Madden T. L., Schaffer A. A., Zhang J., Zhang Z., Miller W., Lipman D. J. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nuc. Acids Res.* 1997;25:3389–3402.
- (21). Andreeva A., Howorth D., Brenner S. E., Hubbard T. J. P., Chothia C. G., Murzin A. G. SCOP database in 2004: refinements integrate structure and sequence family data. *Nuc. Acids. Res.* 2004;32:D226–D229.
- (22). Murzin A. G., Brenner S. E., Hubbard T., Chothia C. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* 1995;247:536–540. PubMed PMID: 7723011.
- (23). Edwards Y. J., Cottage A. Bioinformatics methods to predict protein structure and function. A practical approach. *Mol. Biotechnol.* 2003;23:139–166. PubMed PMID: 12632698.
- (24). Moulton J. A decade of CASP: progress, bottlenecks and prognosis in protein structure prediction. *Curr. Opin. Struct. Biol.* 2005;15:285–289. PubMed PMID: 15939584.
- (25). Rajpal D. K. Understanding biology through bioinformatics. *Int. J Toxicol.* 2005;24:147–52. PubMed PMID: 16040566.
- (26). Martí-Renom M. A., Stuart A., Fiser A., Sánchez R., Melo F., Sali A. Comparative protein structure modeling of genes and genomes. *Annu. Rev. Biophys. Biomol. Struct.* 2000;29:291–325. PubMed PMID: 10940251.
- (27). Godzik A. Fold recognition methods. *Methods Biochem. Anal.* 2003;44:525–546.
- (28). Bonneau R., Baker D.A. Ab initio protein structure prediction: progress and prospects. *Annu. Rev. Biophys. Biomol. Struct.* 2001;30:173–189. PubMed PMID: 11340057.
- (29). Creighton, T. *Proteins: Structures and Molecular Properties*, 2<sup>nd</sup> Edition, W.H. Freeman, 1992.
- (30). Branden, C & Tooze, J. *Introduction to protein Structure*, 2<sup>nd</sup> Edition, Garland Publishing, New York, 1999.
- (31). Petsko, G. A. & Ringe, D. *Protein Structure and Function*, New Science Press, 2004.
- (32). Zhang Y, Baranov P.V., Atkins J.F., Gladyshev V.N. Selenocysteine and pyrrolysine use dissimilar decoding strategies. *J. Biol. Chem.* 2005;280:20740–20751. PubMed PMID: 15788401.
- (33). MacArthur M.W., Thornton J.M. Influence of proline residues on protein conformation. *J. Mol. Biol.* 1991;218:397–412. PubMed PMID: 2010917.

- (34). Ramakrishna, C., Ramachandran G.N. Stereochemical criteria for polypeptide and protein chain conformations. II. Allowed conformations for a pair of peptide units. *Biophys. J.* 1965;5:909–933.
- (35). Karpen M.E., Dehaseth P.L., Neet K.E. Differences In The Amino-Acid Distributions Of 310-Helices And Alpha-Helices. *Prot Sci.* 1992;1:1333–1342.
- (36). Fodje M,N, Al-Karadaghi S. Occurrence, conformational features and amino acid propensities for the pi-helix. *Prot. Eng.* 2002;15:353–358.
- (37). Blundell T., Barlow D., Borkakoti N., Thornton J. Solvent-induced distortions and the curvature of alpha-helices. *Nature.* 1983;306:281–283. PubMed PMID: 6646210.
- (38). Hol W.G.J., Vanduijnen P.T., Berendsen H.J.C. Alpha-Helix Dipole And Properties Of Proteins . *Nature.* 1978;273:443–446. PubMed PMID: 661956.
- (39). Richardson J.S., Richardson D.C. Amino acid preferences for specific locations at the ends of alpha helices. *Science.* 1988;240:1648–1652. PubMed PMID: 3381086.
- (40). Salemme F.R. Structural properties of protein beta-sheets. *Prog. Biophys. Mol. Biol.* 1983;42:95–133. PubMed PMID: 6359272.
- (41). Chou K-C. Prediction of tight turns and their types in proteins. *Anal. Biochem.* 2000;286:1–16. PubMed PMID: 11038267.
- (42). Sibanda B.L., Blundell T.L., Thornton J.M. Conformation of beta-hairpin protein structures. *J. Mol. Biol.* 1989;206:759–777. PubMed PMID: 2500530.
- (43). Chothia C., Levitt M., Richardson D. Helix to helix packing in proteins. *J. Mol. Biol.* 1981;145:215–250. PubMed PMID: 7265198.
- (44). Chothia C., Janin J. Relative orientation of close-packed beta-pleated sheets in proteins. *Proc. Natl. Acad. Sci.* 1981;78:4146–4150. PubMed PMID: 16593054.
- (45). Janin J., Chothia C. Packing of alpha-helices onto beta-pleated sheets and the anatomy of alpha/beta proteins. *J. Mol. Biol.* 1980;143:95–128. PubMed PMID: 7003165.
- (46). Orengo C.A., Michie A.D., Jones S., Jones D.T., Swindells M.B., Thornton J.M. CATH - a hierarchic classification of protein domain structures. *Structure.* 1997;5:1093–1108. PubMed PMID: 9309224.
- (47). Presnell S.R., Cohen F.E. Topological distribution of four alpha-helix bundles. *Proc. Natl. Acad. Sci.* 1989;86:6592–6596. PubMed PMID: 2771946.
- (48). Teichmann S.A., Chothia C., Gerstein M. Advances in Structural Genomics. *Curr. Opin. Str. Biol.* 1999;9:390–399.
- (49). Orengo C.A., Jones D.T., Thornton J.M. Protein superfamilies and domain superfolds. *Nature.* 1994;372:631–634. PubMed PMID: 7990952.
- (50). Richardson J.S. Handedness of crossover connections in beta-sheets. *Proc. Natl. Acad. Sci.* 1976;73:2619–2623. PubMed PMID: 183204.
- (51). Raetz C.R., Roderick S.L. A left-handed parallel beta helix in the structure of UDP-N-acetylglucosamine acyltransferase. *Science.* 1995;270:997–1000. PubMed PMID: 7481807.
- (52). Taylor W.R. A deeply knotted protein structure and how it might fold. *Nature.* 2000;406:916–919. PubMed PMID: 10972297.
- (53). Milton R.C.D., Milton S.C.F, Kent S. B. H. Total chemical synthesis of a d-enzyme: the enantiomers of HIV-1 protease show demonstration of reciprocal chiral substrate specificity. *Science.* 1992;256:1445–1448. PubMed PMID: 1604320.
- (54). Goodsell D.S., Olson A.J. Structural Symmetry and Protein Function. *Annu. Rev. Biophys. Biomol. Struct.* 2000;29:105–153. PubMed PMID: 10940245.
- (55). Braig K., Otwinowski Z., Hegde R., Boisvert D.C., Joachimiak A., Horwich A.L., Sigler P.B. The crystal structure of the bacterial chaperonin GroEL at 2.8 Å. *Nature.* 371:578–586.
- (56). Seemuller E., Lupas A., Stock D., Lowe J., Huber R., Baumeister W. Proteasome from *Thermoplasma acidophilum*: a threonine protease. *Science.* 1995;268:579–582. PubMed PMID: 7725107.
- (57). Hugo Kubinye - 3D QSAR in Drug Design: Theory Methods and Applications - Chapter: Molecular Superposition for Rational Design, page 200–201.
- (58). Pandi Veerapandian - Structure-Based drug design -1997

- (59). Klaus Gubernator & Hans-Joachim Böhm - Structure-based Ligand Design -1998.
- (60). Myers S., Baker A. Drug discovery - an operating model for a new era. *Nature Biotechnology*. 2001;19:727–730.
- (61). Kuntz I.D. Structure-based strategies for drug design and discovery. *Science*. 1992;257:1078–1082. PubMed PMID: 1509259.
- (62). Verdonk M. L., Cole J. C., Hartshorn M. J., Murray C. W., Taylor R.D. Improved Protein–Ligand Docking Using GOLD. *PROTEINS: Structure, Function, and Genetics*. 2003;52:609–623.
- (63). Vigers G. P. A, Rizzi J. P. Multiple Active Site Corrections for Docking and Virtual Screening. *J. Med. Chem*. 2004;47:80–89. PubMed PMID: 14695822.
- (64). Wang R., Wang S. How Does Consensus Scoring Work for Virtual Library Screening? An Idealized Computer Experiment. *J. Chem. Inf. Comput. Sci*. 2001;41:1422–1426. PubMed PMID: 11604043.
- (65). Lyne, P. D. Structure-based virtual screening: an overview. *Drug Discovery Today* Vol. 7, No. 20. 2002.
- (66). Wade, C. R., Henrich, S., Wang, t. Using 3D protein structures to derive 3D-QSARs. *Drug Discovery Today*, Vol.1, No 3. 2004.
- (67). Anderson, A. C. Targeting DHFR in parasitic protozoa. *DDT*, Vol. 10, No 2. 2005.
- (68). Shaw M.P. et al. High-resolution crystal structure of *Trypanosoma brucei* UDP-galactose 4\_-epimerase: a potential target for structure-based development of novel trypanocides. *Molecular & Biochemical Parasitology*. 2003;126:173–180. PubMed PMID: 12615316.
- (69). Perbandt M. Structure of the Major Cytosolic Glutathione S-Transferase from the Parasitic Nematode *Onchocerca volvulus*. *J.B.C.* 2005;280(13):12630–12636.
- (70). Nathan S. T. et al. Structure of glutathione S-transferase of the filarial parasite *Wuchereria bancrofti*: a target for drug development against adult worm. *J. Mol. Model*. 2005;11:194–199. PubMed PMID: 15864673.
- (71). Sánchez-Sixto C. et al. Structure-Based Design, Synthesis, and Biological Evaluation of Inhibitors of *Mycobacterium tuberculosis* Type II Dehydroquinase. *J. Med. Chem*. 2005;48:4871–4881. PubMed PMID: 16033267.
- (72). Olliaro P. L., Yuthavong Y. An Overview of Chemotherapeutic Targets for Antimalarial Drug Discovery. *Pharmacol. Ther*. 1999;81(2):91–110. PubMed PMID: 10190581.
- (73). Schuttelkopff A. W. et al. Structures of *Leishmania major* pteridine reductase complexes reveal the active site features important for ligand binding and to guide inhibitor design. *J. Mol. Biol*. 2005;352:105–116. PubMed PMID: 16055151.
- (74). Buschiazzo A. et al. The Crystal Structure and Mode of Action of Trans-Sialidase, a Key Enzyme in *Trypanosoma cruzi* Pathogenesis. *Molecular Cell*. 2002;10:757–768. PubMed PMID: 12419220.
- (75). Jao, S. C. et al. (2005) Design of potent inhibitors for *Schistosoma japonica* glutathione S-transferase. *Bioorg. Med. Chem*. ARTICLE IN PRESS, 2005.
- (76). Padmanabhan P.K. et al. Glyoxalase I from *Leishmania donovani*: A potential target for anti-parasite drug. *Biochemical and Biophysical Research Communications*. 2005;337:1237–1248. PubMed PMID: 16236261.
- (77). Guex N., Peitsch M.C. SWISS-MODEL and The Swiss-PdbViewer: An environment for comparative protein modeling. *Electrophoresis*. 1997;18:2714–2723. PubMed PMID: 9504803.
- (78). Humphrey W., Dalke A., Schulten K. VMD - Visual Molecular Dynamics. *J. Molec. Graphics*. 1996;14:33–38.



## **B. Case Studies**



## Chapter B01. Control of Gene Expression in *Plasmodium*

Mauro Ferreira de Azevedo<sup>1</sup> and Hernando A. del Portillo<sup>2</sup>

Created: July 17, 2006; Updated: May 11, 2007.

Malaria parasites have more than 10 stages of cellular differentiation and invade at least four types of cells in two different hosts with a considerable variation in temperature between them. All of this complex biology depends on the efficient control of gene expression, about which our knowledge still has many shortcomings. Although this parasite has some general mechanisms in common with yeast and higher eukaryotes, many aspects of its genetic regulation seem to be specific to this genus: (i) during the asexual blood stages, the parasites seem to turn on a rigid, viral-like program of early, middle, and late genes expressed as a cascade of continuous events; (ii) it seems likely that malaria parasites have acquired unique and yet-to-be-described transcription factors; (iii) antisense transcription has been described in about 10% of the coding genome, clearly indicating as-yet-undefined, post-transcriptional control mechanisms; and (iv) control gene expression of the var subtelomeric multigene family involves a gene-specific cross-talk between intron and exon, as well as epigenetic mechanisms to control allelic exclusion. Here, we review our present knowledge on control of gene expression in malaria parasites and illustrate the importance of bioinformatics in advancing our knowledge in this area, with illustrative examples on promoters, transcription factors, and the transcriptome analysis of the intraerythrocytic developmental cycle.

### General Aspects

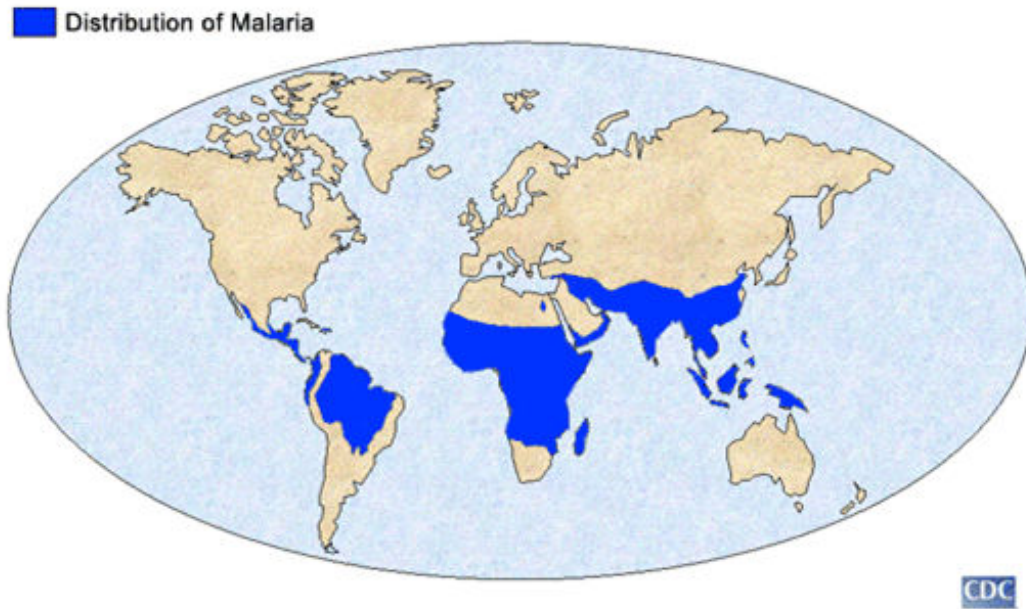
Malaria is the most important parasitic disease in the world. Each year, 300 to 500 million people are infected, and more than 1 million people, mostly children under 5 years of age, die. More than 40% of the world's population live in high-risk areas including sub-Saharan Africa, South and South-East Asia, Indonesia, and South and Central America (Figure 1) (1).

The causative agent is a parasite of the phylum Apicomplexa, genus *Plasmodium*. Although there are more than 100 species of this genus, only *P. falciparum*, *P. malariae*, *P. ovale*, and *P. vivax* infect humans. More than 99% of infections are caused by *P. falciparum* and *P. vivax*.

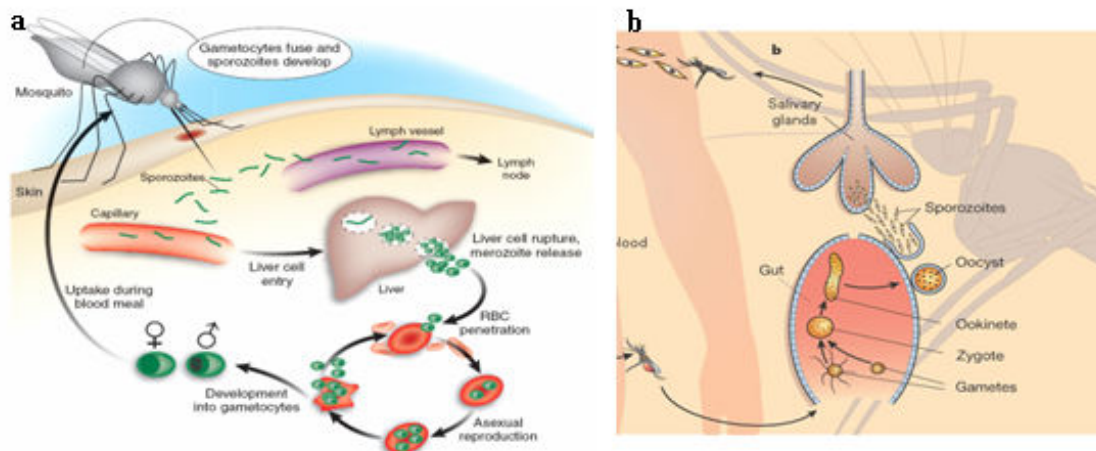
The geographic distribution of these two malaria species is very different. *P. falciparum* is the species most prevalent in sub-Saharan Africa, where the highest number of deaths occurs. *P. vivax* is widely distributed and is the most frequently found species in the majority of regions, except Africa. A genetic characteristic of the Negro population is one of the reasons that *P. vivax* is less prevalent in Africa. A mutation in the gene encoding the “Duffy” antigen confers the Duffy-negative phenotype to this population. *P. vivax* depends on this receptor to invade the erythrocyte and is incapable of infecting Duffy-negative red blood cells (2, 3). *P. falciparum*, however, has alternative mechanisms for invasion, independent of binding to the “Duffy” protein (4).

### Life Cycle

The life cycles of the different malaria species are very complex (Figure 2). An infected female *Anopheles* mosquito bites the vertebrate host, injecting parasite forms, known as sporozoites, into the dermis. Intense movement in the dermis allows some of the parasites to encounter blood and lymphatic vessels, thereby causing a decrease in movement. The parasites that enter the lymphatic system invade the closest lymph node and transform into a stage similar to the hepatic stage. However, they are unlikely to complete the process of division



**Figure 1.** Geographical distribution of human malaria worldwide. Obtained from CDC [http://www.cdc.gov/malaria/distribution\\_epi/distribution.htm](http://www.cdc.gov/malaria/distribution_epi/distribution.htm).



**Figure 2.** Life cycle. a. The life cycle in the vertebrate host, demonstrating the modifications proposed by Amino *et al.* (2006) (5). Extracted from Jones *et al.* (2006) (104). b. Life cycle in the mosquito ([http://www.sanger.ac.uk/PostGenomics/plasmodium/presentations/plasmodium\\_lifecycle.shtml](http://www.sanger.ac.uk/PostGenomics/plasmodium/presentations/plasmodium_lifecycle.shtml)).

or reach cell maturity. This extraerythrocytic stage was described recently by Amino *et al.* (2006) (5), but its importance in establishing infection and the hosts' immune response is still unknown. Parasites that enter the blood vessels are carried in the blood and reach the hepatocytes, invading these cells. Here, they are modified to a more rounded form, which initiates the process of asexual reproduction, forming the hepatic schizont, which now contains thousands of merozoites that are liberated into the host circulation. In *P. vivax*, after the sporozoites have invaded the hepatocytes, they can differentiate into a dormant stage called a hypnozoite. These can lie dormant for months and re-initiate replication when the original disease has already been cured, leading to another infection, known as relapse. In this way, patients cured of the disease can develop a new infection. Independent of this particular life stage of *P. vivax*, once merozoites are released into the bloodstream, they quickly invade red blood cells, initiating the asexual intraerythrocytic cycle, which causes the disease pathology.

The parasite forms a parasitophorous vacuole, where it develops into the first stage, known as the ring stage. This feeds on the contents of the red blood cell and transforms into a trophozoite. This stage begins the process of cell division known as schizogony and transforms into the schizont. This contains the new merozoites that rupture the red blood cell and are released into the bloodstream to infect new cells, continuing the disease cycle. Some parasites do not begin the process of cell division and instead, differentiate into the sexual form of the parasite, the male and female gametocytes that will be ingested by a mosquito during its blood meal. Once in the digestive tract of the insect, the red blood cells are digested, releasing the male and female gametes. The male gametes (microgametes) undergo three rounds of mitosis, giving rise to eight cells, a phenomenon known as exflagellation, and are transformed into individual male gametes. Fertilization of the gametes occurs forming a zygote, the only diploid stage of the parasite. This differentiates into the ookinete, an amoeboid form able to cross the peritrophic membrane, and lodges in the internal wall of the digestive tract between the epithelium and the basal lamina. It forms a protective envelope and transforms into an oocyst. This undergoes the first reducing meiosis, followed by many stages of mitosis, releasing haploid sporozoites into the haemolymph of the insect. Sporozoites migrate, and some of them actively penetrate the salivary gland of the mosquito until a new blood meal is done, and another individual may then be infected.

## ***Plasmodium falciparum* and *Plasmodium vivax***

*P. falciparum* and *P. vivax* are the most important human malaria species. The pathology caused by infection by each of these parasites is very distinct. Although *P. vivax* infects only reticulocytes, *P. falciparum* infects young and mature erythrocytes indiscriminately, producing much higher parasitemias. *P. falciparum* has a cytoadhering phenotype where mature forms of the parasite adhere to the endothelium, escaping clearance by the spleen. Because of this phenotype, mostly ring stages are observed on examination of peripheral blood. *P. vivax* does not have a similar phenotype, and it is unknown how clearance by the spleen is avoided. Recently, it has been speculated that *P. vivax* parasites cytoadhere to the barrier cells of the spleen, escaping spleen macrophage clearance and installing themselves into a reticulocyte-rich area (6). These characteristics mean that *P. falciparum* malaria is much more lethal than *P. vivax*, which rarely kills its host but is also a chronic and debilitating disease.

The types of study that can be carried out using these two parasites have different limitations. *P. falciparum* can be continuously cultured easily *in vitro*, allowing application of methods such as: *in vitro* drug testing; inhibition of invasion using specific antibody serum; and transfection and genetic manipulation. All of these methods are severely limited in *P. vivax* because of the absence of a simple, reproducible method of *in vitro* continuous culture. In fact, this limitation is reflected in the large amount of data that have been gathered from the genome, transcriptome, and proteome of *P. falciparum* as compared with *P. vivax*.

## **Genome**

In the last two decades, considerable advances in the techniques of DNA sequencing and partial or complete sequences of hundreds of genomes have been published (<http://www.ncbi.nih.gov/entrez/query.fcgi?db=genomeprj>). Although the genome probably contains most of the information necessary for the development of an organism, the function and regulation of expression of predicted genes are not immediately apparent after publication of DNA sequence data. This information requires much more time and resources than the initial genome projects and also depends on advances in post-genome technology, such as bioinformatics and reverse genetics, that permit the formation and functional testing of models for biological problems.

The study of malaria has benefited from the advance of genome projects, *Plasmodium* spp. being one of the parasites with the greatest number of species and strains whose genomes have been or are being sequenced. The genomes of *P. falciparum* and *P. yoelli* (a species that infects rodents) have been sequenced, and their first drafts were published 4 years ago (7, 8). Partial sequences of the *P. chabaudi* and *P. berghei* genomes, and comparison

with the other two genomes of the genus, were published recently (9). The genome of *P. vivax* is in the final stages of annotation and should be published in the Spring/Summer of 2007 (<http://www.tigr.org/tdb/e2k1/pva1/>). As well as these, total or partial sequencing projects of the genomes of *P. knowlesi*, *P. gallinaceum*, and *P. reichenowi* are under way. The sequencing data and annotation of these genomes, as well as information about gene expression and protein interactions, are available in databases such as GenBank, GeneDB, and PlasmoDB, the last being the official database of the *Plasmodium* spp. genomes.

The *P. falciparum* genome project was undertaken as a consortium of the sequencing centers The Institute for Genomic Research (TIGR), the Sanger Center, and Stanford University. The strategy used involved separating the 14 chromosomes by pulsed field electrophoresis and then sequencing each one using the “shotgun” method. This entails cloning smaller parts of the chromosomes into plasmid vectors and then sequencing these clones using vector-specific oligonucleotides. In some cases, larger clones, constructed by using yeast artificial chromosomes (YACs), were also used to help with mapping. Chromosomes 1, 3, 4, 5-9, and 13 were sequenced by the Sanger Center; chromosomes 2, 10, 11, and 14 by TIGR; and chromosome 12 by Stanford University.

The haploid genome of *P. falciparum* has more than 23 million base pairs (Mb) and a total AT content of 80.62%. The AT content in the coding regions is 76.23%, increasing to almost 90% in introns and intergenic regions. At the last data update, 5,440 genes had been annotated, 5,374 of which encode proteins. Of these, more than 50% contain introns, and 63% were annotated as hypothetical proteins without known function ([http://www.sanger.ac.uk/Projects/P\\_falciparum/genome.overview.shtml](http://www.sanger.ac.uk/Projects/P_falciparum/genome.overview.shtml)). Of the RNA genes, 45 transfer RNA (tRNA) and 20 ribosomal (rRNA) genes were identified (7).

The genome of *P. vivax* has yet to be published, but preliminary data indicate an AT content of about 55% and a total size of about 30 Mbp. The subtelomeric regions—where many of the genes are involved in virulence and/or antigenic variation, such as the *vir* gene family (10), are found—have a higher AT content of about 80%. In the central chromosome regions, synteny with the *P. falciparum* genome is high, as is the similarity between the genes of the two species.

Analysis of the genomes of the two rodent species of malaria, *P. berghei* and *P. chabaudi*, together with the data published previously for *P. yoelli* and *P. falciparum*, demonstrates that about 80% of the rodent malaria genes have orthologs in *P. falciparum*, possibly forming a group of universal plasmodial genes (9). Most of these genes are localized in the central regions of the chromosomes, with the subtelomeric regions being much more diverse.

The data reviewed above show that the genome sequences of the most important malaria parasites are readily available for mining and comparisons. Of general interest, comparisons between syntenic chromosome regions of some species of *Plasmodium* can be performed using tools available at the new PlasmoDB site, and several programs such as Artemis allow genome visualizations and annotations.

## Transcriptome

Some years before the *P. falciparum* genome was published, the first large-scale studies of the transcriptome were begun using “microarray” technology. In spite of working with limited data from the pre-genomic era, these studies were capable of identifying many differences in abundance of transcription between trophozoites and gametocytes (11) and between the asexual intraerythrocytic stages (12).

As the sequencing data available for *Plasmodium* spp. increased, new studies on transcriptomes appeared in much greater scale. The transcriptome, analyzed by microarray for 48 hours during the intraerythrocytic life cycle of *P. falciparum*, with resolution of 1 hour, revealed that the parasite has ridged regulation, which seems unique in eukaryotes (13). Indeed, most of the genes are transcribed during this phase of the life cycle, and about three-quarters of these are expressed only once (see below). Another study (14) analyzed the transcriptome of six stages of the intraerythrocytic cycle, as well as merozoites, sporozoites, and gametocytes, using microarray. By

using methods with less temporal resolution but analyzing transcripts from a greater variety of stages, this study found that less than one-half the parasites' genes (49%) are regulated at the transcription level.

Analysis of transcripts from less abundant parasite stages, such as sporozoites and hepatic forms, has also been undertaken. Subtractive cDNA libraries, obtained from sporozoite and merozoite RNA, were used to identify 25 sporozoite-specific transcripts possibly involved in invasion (15). Preliminary data from transcriptomes of hepatic stages of the parasite indicate an absence of specific transcripts and a change from the expression profile of sporozoites to that of blood-stage forms (16-18). The use of different methodology to isolate the parasites in these three studies leads to results that are difficult to compare. This may reflect on the presence of different profiles, depending on the time of infection of the hepatocyte and consequent maturity of the parasites.

An increase in the sensitivity of the microarray technique has permitted analysis of the transcriptome of parasites isolated from patients. In these studies, it was demonstrated that the pattern expression of most genes is equivalent to parasites maintained in culture. However, some genes behave differently, perhaps reflecting differences in the biology of the parasites in different conditions. Among these are the genes that encode surface antigens and proteins involved in sexual differentiation that are overexpressed *in vivo* (19, 20). It is possible that these differences in expression represent a greater need for the parasite to escape the host immune response and that *in vivo* conditions favor greater production of gametocytes for infecting mosquitoes.

The advent of microarray technology thus faces a computational challenge to analyze this large amount of data because, for instance, the single experiment on the transcriptome of the intraerythrocytic developmental cycle (IDC) at 1-hour resolution generated close to 350,000 data. Indeed, many computational, statistical, and bioinformatics algorithms have been developed to analyze microarray data (21).

## Proteome

An increase in the sensitivity of mass spectrometry techniques and improving methods for cultivation and purification of parasites have allowed considerable advances in studies of the *Plasmodium* spp. proteome.

Data from the *P. falciparum* proteome (22) and *P. berghei* (9) confirm most of the microarray data from Bozdech *et al.* (2003) (13). More than one-half of the proteins in the parasite are expressed in only one stage, and for the genes of known function, there is a high correlation between the genes expressed and metabolic and cellular activities of the parasite at that stage. Some stage-specific genes belong to multigene families whose members are expressed at different stages in a strategically specific manner, such as involvement in invasion, indicating that the parasite might use a limited repertoire of strategies to survive.

Genes that were thought to be specific to the intraerythrocytic cycle, such as the genes *var*, *rifin*, *stevor*, and the super family *pir*, which include the *vir* genes of *P. vivax*, are transcribed and often translated in parasite stages in the mosquito (9, 22). It is possible that some members of these families have an important function for survival of these stages, or that the control of expression is simply more relaxed.

The proteome of less abundant stages, such as the gametocytes and sporozoites, has also been determined, demonstrating many differences between the repertoire of proteins in these stages and the asexual blood forms (23, 24). Purification of male and female gametocytes was performed by fluorescence activated cell sorting (FACS) of parasites that express the *gfp* gene controlled by active promoters in each of the stages (25). The data produced in this study demonstrate a very distinct repertoire in the gametocytes compared with the other parasite stages. The male gametocytes have the most specialized proteome, perhaps reflecting the intense differentiation process that occurs during exflagellation.

Organelles specific to this phylum, such as rhoptries, are essential for invasion. Rhoptries have been purified, and the proteins have been identified by mass spectrometry. A study by Bradley *et al.* (2005) (26) identified some of

the proteins present in the rhoptries of *T. gondii*. They have orthologous genes in *P. falciparum*, indicating the possibility of common mechanisms of invasion. A similar study was done on *P. falciparum* (27).

The data reviewed above indicate that malaria parasites are able to express mostly non-redundant protein repertoires at each particular stage, reinforcing the idea of a tight control of gene expression. On the other hand, data from the transcriptome of the IDC at 1-hour resolution indicates a simple program for controlling gene expression during the intraerythrocytic developmental cycle where more than 75% of the genes of the parasites are transcribed. Understanding how the parasite controls gene expression at each differentiation stage is essential to the development of alternative strategies and might shed light into unique mechanisms acquired during the evolution of malaria parasites.

## Control of Gene Expression

Malaria parasites have more than 10 stages of cellular differentiation, precisely regulate differentiation and replication, and invade at least four types of cells in two different hosts with a considerable variation in temperature between them. All of this complex biology depends on an efficient control of gene expression—and what we know of gene expression control still has many shortcomings. Although this parasite has some general mechanisms in common with yeast and higher eukaryotes, many aspects of their genetic regulation seem to be specific to this genus.

## Chromatin

In nucleated organisms, DNA is compartmentalized in the nucleus, and this provides the first barrier to gene activation. Transcription factors are produced in the cytoplasm and must enter the nucleus to exercise their functions. Inside the nucleus, the DNA is packed and compartmentalized into chromatin that has regions with different levels of packaging. Heterochromatin is highly compact, and thus it is difficult to transcribe genes in these regions. In contrast, euchromatin is more relaxed, and genes are more easily transcribed. There is movement in the compaction of chromatin, allowing genes in a region of heterochromatin to be changed to a region of euchromatin and activated. One of the first steps in activation of transcription is the unpacking of the DNA, allowing access of the proteins needed to initiate transcription. This process frequently involves the acetylation of histones in the promoter region, making them negatively charged and decreasing their affinity for the DNA.

In *Plasmodium*, the chromatin DNA is structured in the form of nucleosomes of approximately 155 bp (28, 29). The genes that code for each of the structural histones (H2A, H2B, H3, and H4) and also for four histone variants are present in the genome (30). The gene that codes for histone H1 has never been found in any of the Apicomplexa organisms, indicating that they may have a different mechanism from other eukaryotes for organizing their nucleosomes (31). It has also been observed that this protein is very divergent among other protozoans and higher eukaryotes, with the absence of the amino-terminal and globular domains in the more primitive organisms (32). It is possible that the Apicomplexa H1 histone has a very distinct structure, making it difficult to be predicted. Proteins responsible for nucleosome assembly have also been identified (33).

It has been observed that many parasite promoters are regulated in a similar manner in the chromosomal and episomal context. This might indicate that the organization of chromatin does not have a great influence on gene regulation. However, Horrocks *et al.* (2002) (34) demonstrated that episomal plasmids are organized in the form of nucleosomes and that the structure and maintenance of the pattern of gene regulation of the promoters depend on the parasites passing through the S-phase of mitosis. More evidence, such as the presence of proteins involved in the acetylation of histones (such as the ADA2-GCN5 complex, which is conserved in eukaryotes), suggests an important role for the regulation of the DNA structure in the parasite (35). The deacetylation of histones represses transcription. Orthologs to histone deacetylases such as HDAC (36) and Sir2 (37), responsible



for repression of gene expression in yeast, have been identified in *P. falciparum*. Sir2 appears to be involved in the formation of heterochromatin and silencing of some types of *var* genes (38, 39).

## Promoters

The promoters of *Plasmodium* spp. have a bi-partite structure with a core promoter followed by regulatory elements (40, 41). Transcription is monocistronic (42), and the genes that encode proteins are transcribed by RNA polymerase II (43). Data from the transcription analyses demonstrate the great importance of transcriptional control. Many promoters characterized by transient transfection experiments express the reporter gene with the same temporal pattern as the endogenous genes. Some, such as the dihydrofolate reductase (*dhfr*) and the elongation factor alpha one (*ef1- $\alpha$* ) promoters, are active during all of the intraerythrocytic cycle. Others are active during specific stages, such as merozoite surface protein 2 (*msp2*), apical membrane antigen 1 (*ama1*), *Pfs16* and *Pfs25* that are active during early schizogony, in mature schizonts, in gametocytes, and in gametes, respectively (44-46).

*Plasmodium* spp. promoters have distinct characteristics, because the transcription machinery of the parasite does not recognize the promoters of other eukaryotes. Yet, the mechanism of transcriptional regulation appears to be conserved within the genus because promoters from different species are functional in heterologous transfection experiments (41, 47-49). Unexpectedly, however, a recent report on the functional analysis of *P. vivax* promoters demonstrated that unlike all other *Plasmodium* spp., promoters of *P. vivax* are poorly or not recognized by the *P. falciparum* transcription machinery (50). These results suggest the existence of *P. vivax*-specific transcription regulatory elements.

Binding sites for various eukaryotic transcription factors are found in the sequence of different *Plasmodium* spp. promoters. Functional analyses detecting DNA-protein interactions, either by transient transfection experiments with reporter genes or by electrophoretic mobility shift assays (EMSAs), were unable to prove whether these sites were recognized by parasite proteins. On the contrary, it was observed that the important sequences involved in promoter activation do not demonstrate similarities with the motifs involved in the regulation of other organisms (46, 51-57). A few functional motifs shared with other eukaryotes have been characterized, such as the "TATA box" (58) responsible for recruitment of the core transcription complex and the long adenine- and thymidine-rich sequences (poly (dT) poly (dA)) (59) that change the conformation of chromatin, making it more accessible to transcription factors.

The data reviewed above clearly demonstrate the scarce knowledge on promoters gathered in malaria through functional analysis, mainly because of the large technical difficulties of such analyses. Bioinformatics approaches are now accelerating discovery of this key aspect of control of gene expression in malaria parasites (see "Post-Transcriptional Control", below).

## Basal Transcription Complex

Initiation of transcription is much more complex in eukaryotes than in bacteria. Instead of a single type of RNA polymerase, able to recognize promoters via the sigma subunit, three different classes of polymerase transcribe different types of genes. RNA polymerase I transcribes ribosomal RNA genes, type II transcribes genes encoding proteins, and type III transcribes transfer RNAs and other small RNAs. These enzymes act as a complex of 8-14 subunits of up to 500 kilodaltons (kD). Recognition of promoter regions is not carried out directly by RNA polymerase but depends on the help of specific proteins, known as transcription factors. These recognize specific sites and bind to DNA and, together with the RNA polymerase, form the core complex for transcription.

*P. falciparum* and other species from the genus contain nuclear genes that code for polypeptide chains of the three types of polymerase. The largest of these polypeptides has variable regions longer than those found in other eukaryotes (60) and contains a repeat motif rich in serine (61). The carboxyl-terminal domain (CTD) of polymerase II has a very divergent amino-acid sequence compared with other eukaryotes. However, a structure

similar to the hepta-peptide repeat, SPTSPSY, is present, indicating that the conserved mechanism of phosphorylation of this region is used for initiation of transcription (62, 63). The mitochondrial RNA polymerase has similarities with the RNA polymerase of the bacteriophage T3/T7 (64) and is encoded by nuclear genes in many organisms. The parasite also contains another RNA polymerase with bacterial characteristics, that of the Apicoplast. The gene encoding one of its polypeptide chains is found in the genome of this organelle (65).

For each of the RNA polymerases to be able to transcribe a gene, it is necessary that transcription factors bind to the promoter region of the DNA. Of the three types of RNA polymerase, pol II needs the greatest number of transcription factors. In the region of the promoter closest to the site of initiation of transcription, the core promoter elements are found, such as the “TATA box”, the “CAAT box”, transcription initiators (INR), and some others. The *trans* elements of the core transcription complex bind to this region of the promoter. These elements are composed principally of TFIID, TFIIA, TFIIB, TFIIF, RNA polymerase, TFIIE, TFIIH, and TFIIJ. Of note, malaria parasites seem to contain much lower transcription factors than crown eukaryotes, including factors associated with the basal transcription complex, which has been mostly conserved throughout evolution (see below).

## Capping

After initiation of transcription, when the recently transcribed RNA reaches a size of about 20-30 nucleotides, it undergoes its first modification, known as “capping”. This process occurs using three basic reactions consisting of: 1) hydrolysis of the triphosphate at the 5' extremity by a RNA 5' triphosphatase, leaving a diphosphate; 2) addition of a guanidine monophosphate group (GMP) in a 5'-to-3' direction to this extremity by the action of a guanyltransferase; and 3) methylation of the last guanine at position N7 by a methyltransferase.

The mechanism of “capping” is different in animals and yeast. Animals have the triphosphatase and the guanyltransferase encoded by a single gene, giving rise to a bifunctional protein, whereas in yeast, three different proteins encoded by individual genes are used. The “capping” is related to protection and stability of the RNA, transport to the cytoplasm, and translation. In *Plasmodium* spp. and other protozoa such as *Trypanosoma brucei* and *Giardia lamblia*, the “capping” mechanism is similar to that of yeast, with the first two enzymes being encoded by different genes (66, 67). However, the methyltransferase has not yet been characterized and may have distinct characteristics from the groups already studied.

## Splicing

At the end of transcription, and in some cases, before this, the messenger RNA (mRNA) should have its introns removed by “splicing”, leaving an open reading frame (ORF) for translation. In many eukaryotes, this is an important regulation point. During the splicing of some genes, part of the exons can be removed with the introns, modifying the reading frame and allowing translation of different isoforms of the protein. This is a regulated process, and the transcripts can be processed in a stage- or tissue-specific form, depending on the needs of the cell.

In *P. falciparum*, this mechanism of regulation appears to be present. Knapp *et al.* (1991) (68) demonstrated that a mRNA from a gene with an unknown function is processed into at least three different transcripts. Species of this genus appear to have certain conserved loci in relation to gene organization and structure, including the number and position of introns and splicing alternatives. This indicates the presence of common control mechanisms (69, 70). In some cases, the alternative splicing may not alter the reading frame. This occurs in the B7 gene, where transcript processing in the asexual and sexual blood stages gives rise to 5' untranslated regions (5'UTR) of different sizes (71). It is possible that this influences the efficiency of translation or the mRNA stability. Although these few examples demonstrate that alternative splicing occurs in *Plasmodium*, the extent of such phenomenon in controlling gene expression in the different developmental stages remains to be determined.

## Termination

The end of transcription in eukaryotes depends on the interaction of various proteins with the recently synthesized mRNA and polymerase II and is accompanied by a second modification in the mRNA, polyadenylation. In the 3' untranslated regions (3'UTR) of the genes, conserved sequences are used as binding sites for a protein complex that cleaves the RNA, 10-30 nucleotides from this site, and adds a ribonucleotide tail of adenines (poly A). It is not yet known how the polymerase, which continues to transcribe the gene, stops transcription and is released from the DNA. Recently, a model was proposed where the yeast protein Rat1 and its cofactor Rai1 bind directly to the 5' extremity of the RNA that continues being transcribed after cleavage of the mRNA and is degraded by the activity of 5'-3' exonuclease until it meets with the RNA polymerase, which, by an unknown method, releases the DNA (72).

As in many eukaryotes, the end of transcription in *Plasmodium* spp. occurs at multiple sites, and the poly A tail is added by cleavage and polyadenylation (73). The elements responsible for this process are not well characterized. Canonical sites of polyadenylation of the type ATTAAA and AATAAA are found in multiple copies in the 3'UTR of various genes, but it is unknown whether these are recognized by the transcription machinery of the parasite in the same manner as other eukaryotes. In some cases, it is probable that the AATAA pentamer is recognized as the site of polyadenylation, which can occur 1-30 nucleotides downstream of this site (74).

For the *P. gallinaceum* gene that encodes for the gametocyte antigen Pgs28, it was demonstrated that the poly A tail is added around 20 nucleotides after the fifth canonic polyadenylation site (AATAAA/ATTAAA) and that, before this, there is an important U-rich element (75). This 3'UTR was used in transient transfection experiments in human cells, being recognized by our own transcription machinery, but in a different manner than that occurring in the parasite in relation to the polyadenylation site and the important *cis* elements for this process (76). Transient transfection experiments demonstrated that the presence of a 3'UTR from the parasite genus is necessary for expression of the reporter genes. Distinct mechanisms for regulation of the termination of transcription and processing of mRNA must be involved, because the parasite does not recognize 3'UTRs from other eukaryotes (77, 78).

## Transcription Factors

In comparison to other organisms, *Plasmodium* spp. appears to have a distinct and perhaps more limited repertoire of proteins involved in gene regulation. Thus, a recent study, undertaken by Coulson *et al.* (2004) (79), identified fewer transcription factors than those of crown eukaryotes using different algorithms and bioinformatics (see below). The initial annotation of the genome did not identify certain components of the core transcription complex, such as TFIIA, the beta subunits of TFIIE and TFIIF, and other factors associated with TFIID and TFIIH. On the basis of these data and analysis of the *T. gondii* genome, it was initially proposed that the core transcription complex in Apicomplexa organisms is simpler than that of other eukaryote groups (31). Improvements in bioinformatic techniques have since then led to the identification of more factors (80) and proteins with complex DNA-binding domains. These data indicate that *P. falciparum*, and perhaps other Apicomplexa, have more genes involved in regulation of transcription than was previously predicted (81). The difficulty in annotating transcription factors in *Plasmodium* spp. is probably attributable to divergence in the sequence of its domains compared with known groups.

Of the proteins that form the core transcription complex, only the "TATA binding protein" (TBP) has been functionally characterized and has an amino acid sequence very divergent from the TBP of other organisms. However, the structure is similar and is capable of binding to the conserved sequences TATAA and TGTA (58, 82). Another *trans* element that has been characterized is a protein relating to the Myb factors. This is regulated transcriptionally during the intraerythrocytic cycle and can bind conserved *cis* elements present in different promoters of the parasite (83).

The low number of transcription factors present in the parasite is contrasted by the high numbers of *cis* elements present in each promoter (84). In yeast for example, most of the promoters are regulated by one or two *cis* elements. Considering that the combination of these two factors (*cis* and *trans*) determines the diversity of regulation in the cell, it is possible that this is maintained in the parasite using a few transcription factors that bind in combination to various *cis* elements in the promoters.

## Post-Transcriptional Control

The 3'UTR has a role in the regulation of gene expression in many eukaryotes. In animals, they determine the temporal expression of numerous genes, particularly during embryonic development. Their secondary structure and interaction with other proteins influence the processing, transport, stability, and efficiency of translation of the transcript. In *Plasmodium* spp., a great deal of evidence indicates the existence of important post-transcriptional control, particularly during the passage from the vertebrate host cycle to its development in the mosquito.

The first evidence of post-transcriptional control came from studies of the *P. berghei* gene *Pbs21*, which encodes a protein localized in the membrane of female gametocytes, zygotes, and oocytes. As well as being detected in these parasite stages, transcripts of this gene are also present in female gametocytes, where they are not translated, demonstrating post-transcriptional control (85). Orthologs of this gene have been found in *P. falciparum* and other parasites of this genus. Other genes that undergo post-transcriptional control are those that encode dihydrofolate reductase and thymidylate synthetase (*dhfr-ts*). In animals, these two enzymes are encoded by independent genes, contrary to the mechanism in protozoans. Each of these proteins can bind its corresponding mRNA and inhibit its translation. In the presence of substrates or competitive inhibitors, these proteins lose the capacity to bind to RNA, and translation increases. In *P. falciparum*, the mRNA binding site is not located close to the active site of the enzyme, which therefore has the same affinity for the mRNA in its active and inactive forms. This characteristic of post-transcriptional control impedes the production of more proteins in a situation of greater necessity, for example in the presence of inhibitors, imposing a much lower tolerance of antifolate drugs than in human cells (86).

Protein families involved in RNA binding are highly represented in the *P. falciparum* genome (79). Among these are the Puf proteins that bind the nano-responsive elements (NRE), present in the 3'UTR of genes of various eukaryotes and inhibit RNA translation. Two genes that encode members of this gene family have been found in the *P. falciparum* genome and as orthologs in other species of the genus. They are overexpressed in gametocytes (87), and functional analysis demonstrates that both proteins are capable of binding to *Drosophila* mRNA that contains the NRE sequence in its 3'UTR, indicating its possible role in post-transcriptional control of genes involved in the development of sexual stages of the parasite (88).

When the transcriptome and the proteome of *P. berghei* were compared, new gametocyte-transcribed genes that have their translation inhibited were found (9). Analysis of the 3'UTR of these genes revealed a conserved motif of 47 nucleotides that contain repetitions of the NRE sequence. These data reinforce the importance of post-transcriptional control in gametocytes and assign the role of recognizing specific elements in the 3'UTR of genes that should have their translation inhibited to the Puf proteins.

Direct evidence of the importance of translational control on gene expression in malaria has been recently and elegantly demonstrated in gametocytes in *P. berghei* (89). Using reverse genetics, the gene encoding the DOZI helicase of *Plasmodium berghei* was knocked-out, and its disruption inhibited the formation of the ribonucleoprotein complexes, diverting at least 370 transcripts for degradation.

Another mechanism of post-transcriptional control is “antisense” RNA, which involves the *cis* or *trans* synthesis of an RNA complementary to the mRNA of some genes. This RNA binds to the corresponding mRNA silencing

its translation, “splicing”, or diminishing its stability. Antisense RNA has been found in all the large groups of organisms and has been most studied in bacteria (90).

Some evidence points to the existence of an antisense silencing mechanism in *Plasmodium* spp. Analysis of the transcriptome of intraerythrocytic stages of the parasite by “serial analysis of gene expression” (SAGE) revealed that more than 10% of the genes have antisense transcripts (91, 92). Later, it was determined that they are transcribed by an RNA polymerase sensitive to  $\alpha$ -amanitina, a characteristic typical of RNA polymerase II (93). The antisense RNA mechanism is functional in transfection experiments for silencing of specific genes, such as *Cla9* (94).

## Non-Coding RNAs

The pattern of gene expression used by the *Plasmodium* genus differs in many aspects to most eukaryotes. Genes that are normally transcribed constitutively, such as histones, actin, and calmodulin, are transcribed in a stage-specific manner. Probably the most surprising difference in regulation of a group of genes in the parasite, compared with other eukaryotes, is the ribosomal RNA (rRNA) genes. Instead of the standard configuration of three subunits (18S, 5S, and 28S) repeated dozens or hundreds of times in sequence, it has around 20 genes, distributed on different chromosomes, often with some of the subunits missing. In addition, two or three structurally distinct forms of rRNA are present, and their expression is regulated. Type A rRNA is transcribed in the blood stages of the parasite, and type S is predominantly transcribed in the mosquito stages of the parasite (95, 96). The exact function of this type of mechanism is unknown, but knockout of one of the S-type genes in *P. berghei* inhibits development in the mosquito in a dose-dependent manner (97). The A-type sequence is more similar to that of other eukaryotes, whereas type S is more divergent, appearing similar to the bacterial ribosome in some regions (98). This type of mechanism of expression of ribosomes is unique, because organisms such as plants and bacteria regulate the type of ribosomes by altering the protein component.

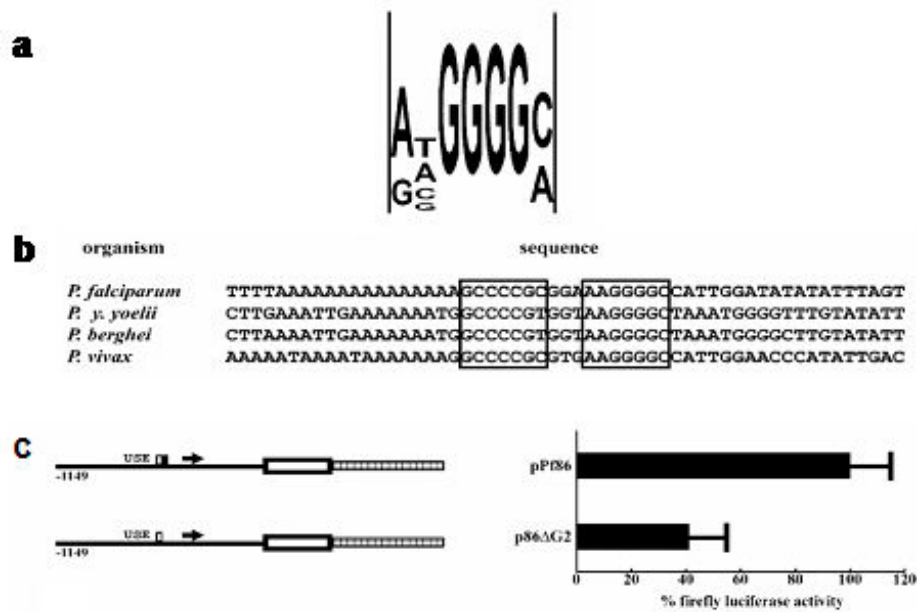
Another group of non-coding RNA also has a peculiar structure in *Plasmodium*. Unlike most major eukaryotic organisms that have redundancy in the most commonly used codons, *P. falciparum* has a tRNA for almost all codons as single copy (7). Transfection experiments with genes containing a much lower AT content than the parasites demonstrate that these are translated without apparent necessity for optimization of the codons. These data may indicate that the parasite is capable of translating GC- and AT-rich codons with the same efficiency, unlike most organisms studied.

As has happened with practically all the genome, transcriptome, and proteome projects, the data generated by the studies in *Plasmodium* spp. have generated many more questions than answers as well as new lines of research. Among these, the use of computational tools to advance our scarce knowledge on control of gene expression is a brand new field that promises to rapidly advance our scarce knowledge in this area. A few illustrative examples on promoters, transcription factors, and transcriptomes will demonstrate its importance.

## Bioinformatics and Gene Expression in *Plasmodium*

### Promoter Elements

Although functional/structural analyses of a few promoter regions from *Plasmodium* have revealed that they possess the common bipartite structure of promoters from other eukaryotes, analysis of intergenic regions has failed to identify canonical sequences such as CCATT and SP1 boxes. The laboratory of Dyann Wirth, as reported by Militello and coworkers (57), was the first to use a bioinformatics approach to gain insight into predicted regulatory elements in intergenic regions of *P. falciparum*. To do so, they analyzed the upstream intergenic regions of the heat shock protein genes as a model using the TRANSFAC (99) database on the [Biobase](#) homepage, which contains data on transcription factors, their binding sites, and regulated genes. TRANSFAC uses Match, a web-based tool to search for transcription factors by weight matrix search, and Patch, another



**Figure 3.** A functional G-rich palindromic element is conserved among the intergenic regions of *hsp* genes in *Plasmodium*. a. G-rich element. b. G-rich palindromic element in different malaria species. c. A reporter plasmid containing the intergenic region of the *hsp* gene of *P. falciparum* and driving the expression of the luciferase reporter gene is functional. Data and figures obtained with permission from Dyanne Wirth.

web-based tool that identifies transcription factors by pattern matching. TRANSFAC thus allows making predictions of poorly characterized gene promoters. Interestingly, except for TATA boxes, whose prediction is questionable in malaria because of the high AT content of intergenic regions (close to 90%), non-other, common elements present in crown eukaryotes were identified in these *Plasmodium* upstream regions, suggesting species-specific regulatory elements. To further look for such elements, all intergenic regions of the *hsp* genes were analyzed using *AlignAce* to find common repeated elements to which a particular value was given (100, 101). Strikingly, a high-score, G-rich palindromic element was found in the intergenic regions of the *hsp* genes (Figure 3a). To determine whether such element was conserved in other *Plasmodium* species, the coding and upstream intergenic regions of the *hsp86* gene of *P. yoellii*, *P. berghei*, and *P. vivax* were retrieved using TBLASTN and BLASTN, with the *P. falciparum* *hsp86* amino acid sequence used as a query. The genomic sequence of the *hsp86* locus of these four *Plasmodium* species were aligned, and significantly, the same G-rich element was observed in the upstream regions in these species, strongly suggesting that it represented an important regulatory element for *hsp* gene expression maintained through evolution (Figure 3b).

To prove these computational findings, recombinant plasmids containing the luciferase reporter gene under the control of the *P. falciparum* *hsp86* upstream intergenic region, as well as mutated versions lacking the G-rich element, proved this one is required for gene expression of the *hsp86* gene (Figure 3c). Similar approaches are now being used to analyze other promoter regions in *P. falciparum*, such as *var* gene promoters. Most relevant, because of the high AT content of intergenic regions as well as a putative role of long stretches of polypyrimidines (i.e., low complexity regions) in basal transcription (59), the development of new algorithms to solve this intrinsic problem of promoter predictions in malaria parasites represents an interesting computational challenge.

## Transcription Factors

Controlled gene expression in higher eukaryotes is achieved mainly through a plethora of Transcription Associated Proteins (TAP), which are mainly taxon specific. Strikingly, the first annotated complete genome

draft from *P. falciparum* failed to predict several TAPs commonly described in other eukaryotes (7). This suggested that either malaria parasites contain significantly lower TAPs than other eukaryotes, or that because of the high AT content of the genome, their prediction required a more sophisticated computational analysis. To address this issue, Coulson and coworkers (2004) (79) devised two different computational approaches to globally search for *P. falciparum* TAPs. In the first approach, they searched the [TRANSFAC](#) and [SWISS-PROT](#) databases for sequences retrieved from “transcription” in their keyword or gene description fields. Next, sequences were linked to their corresponding Protein Families (Pfam) database annotations; in this way, they obtained 51 profile-Hidden Markov Models (HMMs) of protein domains involved in transcription. HMMER version 2.2 was then used to search for *P. falciparum* sequences using the 51 HMMs. In addition, seven other databases from crown eukaryotic organisms, which included *Schizosaccharomyces pombe*, *Saccharomyces cerevisiae*, *Arabidopsis thaliana*, *Caenorhabditis elegans*, *Anopheles gambiae*, *Drosophila melanogaster*, and *Homo sapiens*, were similarly searched with these HMMs. Interestingly, only 17 models had matches with *P. falciparum* sequences corresponding to 69 unique proteins. In contrast, a similar analysis revealed 40 models matching *Arabidopsis thaliana* sequences.

In the second approach, they queried the *P. falciparum* genome as well as the genome of the same crown eukaryotes genomes against 17,054 TAPs. Clustering of TAPs and homologs by similarity revealed 84 clusters containing at least one *P. falciparum* gene and one TAP pertaining to 95 genes, of which only 8 were also identified by the HMMs. Together, these approaches thus revealed a total of 156 malarial TAPs, overall representing 3% of the genome. This is in striking contrast to the other higher eukaryotes similarly analyzed, which revealed an average of 10% of their genomes corresponding to TAPs. The complete list of TAPs as well as information on PlasmoDB ID, gene description, and expression at different developmental stages of malaria parasites are available from the [Supplementary Material](#).

Lack of predictions of the TFIID complex, essential for the formation of the transcription preinitiation complex at promoter regions of Pol II genes, encouraged a search for new computational methods to look for such factors. Analysis of *P. falciparum* protein domains revealed that regions of low complexity were rarely found within globular hydrophobic domains (80). Thus, a new computational, two-dimensional program called Hydrophobic Cluster Analysis (HCA), which is not sensitive to gaps, was generated and used to search for globular domains of *P. falciparum* having similarity to domains conserved in protein sequences representing different subunits of the basal transcription factors and cofactors of humans and yeast (80). Importantly, 33 putative transcription factors, of which only 10 had been predicted using HMMs, were identified using HCA (Table 1). Thus, the combination of profile-based searches along with HCA promises to be a general method to identify TAPs in a genome scale for malaria parasites.

Several other important aspects on control of gene expression of malaria parasites were revealed by these analyses. First, unlike crown eukaryotes, malaria parasites do not contain large paralogous families of TAPs; in fact, this lack of gene duplications is not limited to TAPs but to other internal genes, in contrast to subtelomeric genes, where malaria have clustered multigene families involved in immune escape. Second, malaria parasites contain homologs of all 12 subunits of RNA polymerase II, TBP protein, and TFIIB, indicating that the basal transcription machinery is similar to that of crown eukaryotes. Yet, most components of the TFIID complex, as well as other factors essential for forming the initiation complex of RNA Pol II genes, were not predicted (see below). Third, TAPs involved in RNA stability and translation, such as the CCCH-type Zn finger, were overrepresented in the genome of *P. falciparum*, reinforcing the idea of an important post-transcriptional control of gene expression. Last, TAPs involved in acetylation/deacetylation were identified, indicating that epigenetic mechanisms can also be involved in control of gene expression in malaria. Recent elegant evidence has indeed demonstrated epigenetic control of *var* gene expression (38, 39).

In summary, using regular similarity analysis (7), HMMs (79), and profile-based searches/HCA (80), there are still about 40% of general transcription factors found in crown eukaryotes that are not predicted in the genome

of *P. falciparum*. Thus, either these factors have evolved to a point that none of these computational methods is able to predict them, or basal transcription is different in malaria parasites. If so, these differences can be rationally exploited for alternative control strategies against malaria.

**Table 1.** General transcription factors predicted in *Plasmodium falciparum*.

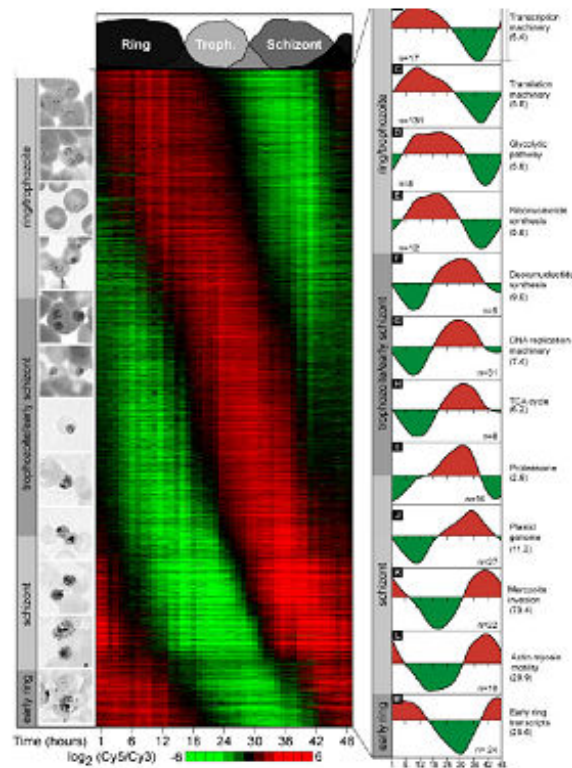
Factors	<i>H. sapiens</i>	<i>S. cerevisiae</i>	<i>P. falciparum</i>	Nuclear signals prediction		Expression pattern	
				NLS	NES	Micro-array	Proteomic data
<b>TFIIA <math>\alpha</math></b>	P52655	P32773 (TOA1)	<b><u>MAL7P1.78 +</u></b>	-	-	G	-
<b>TFIIA <math>\beta</math></b>			<b><u>PFL2435w +*</u></b>	-	-	T,Sc	-
<b>TFIIA <math>\gamma</math></b>	P52657	P32774 (TOA2)	<b><u>PFL1630 *</u></b>	+	-	G	-
<b>TFIIB</b>	Q00403	P29055	FFA0525w	-	-	All stages	-
<b>TFIID TBP</b>	P20226	P13393	FFB0305w	-	-	R	-
<b>TFIID TAF1</b>	P21675 (TAF250)	P46677 (TAF145)	<b><u>PFL1645w</u></b>	+	-	SLT	S,G
<b>TFIID TAF2</b>	gi4507347 (TAF150)	P23255 (TAF150)	<b><u>MAL7P1.134</u></b>	+	+	R,T	\$
<b>TFIID TAF5</b>	Q15542 (TAF100)	P38129 (TAF90)	?				
<b>TFIID TAF7</b>	Q15545 (TAF55)	Q05021 (TAF67)	<b><u>PFL1425w</u></b>	+	+	R,S	-
<b>TFIID TAF14</b>	P42568 (ENL/JAF-9)	P35189 (TAF30)	?				
<b>TFIID TAF4</b>	Q00268 (TAF135)	P50105 (TAF48)	?				
<b>TFIID TAF12</b>	Q16514 (TAF20)	Q03761 (TAF68/61)	?				
<b>TFIID TAF6</b>	P49948 (TAF80)	P53040 (TAF60)	?				
<b>TFIID TAF9</b>	Q16594 (TAF31)	Q05027 (TAF17)	?				
<b>TFIID TAF11</b>	Q15544 (TAF28)	Q04226 (TAF40)	?				
<b>TFIID TAF13</b>	Q15543 (TAF18)	P11747 (TAF19)	?				
<b>TFIID TAF3</b>	gi13374079 (TAF140)	Q12297 (TAF47)	?				
<b>TFIID TAF8</b>	gi31323620 (TAF43)	Q03750 (TAF45)	?				
<b>TFIID TAF10</b>	Q12962 (TAF30)	Q12030 (TAF25)	<b><u>PFE1110w</u></b>	-	+	R, Sc	-
<b>TFIIE <math>\alpha</math></b>	P29083	P36100	<b><u>MAL7P1.86 +</u></b>	+	+	Sc	-
<b>TFIIE <math>\beta</math></b>	P29084	P36145	<b><u>MAL13P1.260 *</u></b>	+	-	ND	-
<b>TFIIF <math>\alpha</math></b>	P35269 (RAP74)	P41895 (Tfg1)	?				
<b>TFIIF <math>\beta</math></b>	P13984 (RAP30)	P41896 (Tfg2)	<b><u>PFL1_0458 *</u></b>	-	+	R,G	-
<b>TFIIH core p62/TFB1</b>	P32780 (p62)	P32776 (TFB1)	<b><u>MAL3P7.42 +*</u></b> <b><u>(Chr2:cbac_258)</u></b>	+	+	R,T	-
<b>TFIIH core p52/TFB2</b>	Q92759 (p52)	gi6325135 (TFB2)	PFL2125c	+	+	R,T,Sc	-
<b>TFIIH core p44/SSL1</b>	Q13888 (p44)	Q04673 (SSL1)	MAL13P1.76	+	+	R,T	-
<b>TFIIH core p34/TFB4</b>	Q13889 (p34)	Gi6325313	PFL3_0279	-	-	T	-
<b>TFIIH core TFB5</b>	G55665883	Gi13129164	<b><u>PFL4_0298</u></b>	-	-	R, T, G	-
<b>TFIIH core XPB/SSL2-RAD25</b>	P19447 (XPB)	Q00578 (SSL2/RAD25)	FF10_0369	+	-	G	\$
<b>TFIIH XPD/RAD3</b>	P18074 (XPD)	P06839 (RAD3)	PFL1650w	+	+	R,T,G	G
<b>TFIIH CAK MAT1/TFB3</b>	P51948 (MAT1)	Gi6320668 (TFB3)	FFB0610c	+	-	R,T	-
<b>TFIIH CAK Cdk7/KIN28</b>	P50613 (CDK7)	P06242 (KIN28)	??				
<b>TFIIH CAK Cyclin H/CCL1</b>	P51946 (cyclinH)	P37366 (CCL1)	??				

The general transcription factors which were identified in this report are underlined and shown in bold; \* and + indicate similarities which were identified in this report after assessing PSI-BLAST marginal similarities at the sequence 2D level or after extending the comparison at the 2D level outside the limits primarily defined by PSI-BLAST, respectively. The Swiss-Prot accession numbers are given for the human and yeast sequences. When a Swiss-Prot identifier is not available, the GenBank identifier (gi) is indicated in italics instead. The references of the human TAF3 and TAF8 sequences can be found in [50] and [100], respectively. \$ : see text for comments on these putative homologues of Cdk7/KIN28 and cyclin H/CCL1. The presence of nuclear localization sequences (NLS [102]), nuclear export sequences (NES [103]), and the expression of these predicted general transcription factors using microarray and proteomics on different parasite stages ([9–11, 89]) are indicated. G: gametocyte, T: trophozoite, LT: late trophozoite, Sc: schizonte, R: ring, (-): absent, (+): present, (?): cannot be found. Table and legend reproduced with permission from Isabelle Callebaut (80).

## Transcriptome Analysis of the Intraerythrocytic Developmental Cycle

Pathology in malaria is mostly associated with the intraerythrocytic developmental cycle (IDC), where sequestration of infected red blood cells in the deep capillaries of internal organs, as well as synchronous rupture of infected red blood cells and release of merozoites, can induce obstruction of blood flow, edema, cerebral malaria, and anemia, among others. Interestingly, during these stages, malaria parasites have a cyclic developmental program in which, upon entrance of merozoites inside red blood cells, the parasites differentiate into a stage known as "ring", which is surrounded by a parasitophorous vacuole. After a period of 12–18 hours, the parasites enter into a highly metabolic growth stage known as a trophozoite, which lasts for about 16 h, after





**Figure 4.** Overview of the *P. falciparum* IDC transcriptome (13). (A) A phaseogram of the IDC transcriptome was created by ordering the transcriptional profiles for 2,712 genes by phase of expression along the y-axis. The characteristic stages of intraerythrocytic parasite morphology are shown on the left, aligned with the corresponding phase of peak gene expression. (B–M) The temporal ordering of biochemical processes and functions is shown on the right. Each graph corresponds to the average expression profile for the genes in each set, and the eman peak-to-through amplitude is shown in parentheses. Figure and legend reproduced with permission from Joe DeRisi.

which DNA synthesis starts in a stage known as the schizont. After nuclear divisions inside a syncytium, cytoplasmic membranes are synthesized, red blood cells burst, and free individual merozoites are liberated into the circulation to invade new red blood cells (Figure 2). In a seminal work carried out at the laboratory of Joe DeRisi at the University of California, San Francisco, the entire transcriptome analysis of the asexual blood stages at 1-hour scale resolution was undertaken (13). To do so, this group had previously developed a web-based program called ArrayOligoSelector that allowed them to construct a 70-base-long oligonucleotide microarray representing the entire *P. falciparum* genome as it had been released since 2000 and annotated as 2002 (7). This program takes into account different parameters including uniqueness, user-defined elements to be avoided, self-binding, complexity, and GC content and is [freely available for downloading](#). A total of 7,462 oligonucleotides representing 5,409 annotated open reading frames were printed into this first-generation microarray. Next, they grew parasites into a 5-liter culture bioreactor, and after a double round of synchronization by sorbitol, they removed two 5-ml aliquots at each hour for 48 hours. One of the aliquots was pooled with all others, making a reference pool labeled with Cy-3, and the other was labeled with Cy-5 and used to analyze the expression profile at each particular hour of the IDC. Thus, at 1-hour scale resolution, they were able to determine those genes that were expressed higher than the reference pool for the entire IDC. Unexpectedly, analysis of the results revealed that close to 85% of the genes were expressed during the IDC, and that of those close to 75% were expressed only once in a periodic manner, having a single maximum and a single minimum of 1-1.5 hours. In sharp contrast, a similar analysis of the expression profiles of the yeast cell cycle revealed that only 3% of the genes were expressed periodically (102). The periodic expression allowed them to

use Fourier Analysis to order expression profiles of the IDC into a phaseogram according to their time and phase of expression (Figure 4). Results demonstrated that malaria parasites have a cascade of expression lacking clear boundaries throughout the IDC, and that genes expressed at each particular phase could be grouped into genes with similar biological functions.

A striking example of this similar phase/similar function was with genes expressed during the late schizogony and involved in merozoite invasion. Thus, during the merozoite stages, the parasites have to expose a series of proteins at their surface that will allow them to specifically interact and enter new red blood cells, and these surface antigens have become prime candidates for the development of vaccines. Strikingly, several merozoite surface proteins—AMA1, EBA, and RESA, all of which have experimental evidence supporting their roles as potential vaccine candidates—were expressed at the same phase, along with many other coding genes, the functions of which are unknown and which expand the list of putative vaccine candidates. Similar results and lists were generated for the entire phaseogram. Because close to 60% of the *P. falciparum* genome represents hypothetical proteins of unknown function, this analysis and lists of genes have potentially and significantly advanced malaria research by assigning function of hypothetical proteins by phase of expression during the IDC; yet, formal proof-of-principle of any one particular hypothetical protein expressed at a particular phase and from which we can assign a similar function as the ones annotated is still lacking. This seminal paper thus illustrates the power of mathematical-computational analysis to advance our biological knowledge of the asexual blood stages of malaria parasites. Moreover, these data indicate that there is a tight, simple, virus-like control of gene expression during the IDC in spite of its complex cycle involving three differentiation stages. It is important to recall, however, that to obtain such data and interpretation, the authors used only 2,714 ORFs of the 5,440 predicted genes. Thus, close to 50% of the entire coding genome of *P. falciparum* was represented or was not periodically expressed (i.e., not amenable to Fourier analysis) on this particular array.

An alternative systems biology approach to create probabilistic genetics networks (PGNs) based on the entire transcriptome of the IDC from these same data was reported by Barrera and co-workers (103). Irrespective of periodically or non-periodically expressed genes, they built a Markov model, which was able to create PGNs based on the interactions on individual or double-genes throughout the IDC. Validation of the model was shown by creating a PGN of glycolysis and by demonstrating that genes in this PGN were biologically sound, and that key enzymes, such as isomerase and which had oligonucleotides representing them and not giving a periodic signal, were included into this PGN. Similar to the Fourier analysis, formal proof of these PGNs still awaits further experimentation. Regardless, this different analysis of the same set of data clearly exemplifies the importance of computational biology, statistics, and mathematics in advancing biologically meaningful results from data on transcriptomes.

## Concluding Remarks

As reviewed in this chapter, control of gene expression of malaria parasites seems unique among crown eukaryotes. Although significant advances in our knowledge of this fundamental aspect of the biology of malaria parasites have been achieved through functional assays, they remain technically difficult. Bioinformatics and computational approaches are rapidly advancing our knowledge in this area. Indeed, as data on genomes, transcriptomes, and proteomes of several *Plasmodium* species are expanding, new challenges will include crossing borders of all these “omes” to integrate a view of the parasites as a dynamic system. An integrative approach involving biology, mathematics, statistics, and computational biology to develop rational approaches for alternative control strategies will be needed.

## References

1. Snow RW, Guerra CA, Noor AM, Myint HY, Hay SI. The global distribution of clinical episodes of *Plasmodium falciparum* malaria. *Nature*. 2005;434(7030):214–217. PubMed PMID: 15759000.

2. Miller LH, Mason SJ, Clyde DF, McGinniss MH. The resistance factor to *Plasmodium vivax* in blacks. The Duffy-blood-group genotype, FyFy. *N Engl J Med.* 1976;295(6):302–304. PubMed PMID: 778616.
3. Barnwell JW, Wertheimer SP. *Plasmodium vivax*: merozoite antigens, the Duffy blood group, and erythrocyte invasion. *Prog Clin Biol Res.* 1989;313:1–11. PubMed PMID: 2675110.
4. Baum J, Maier AG, Good RT, Simpson KM, Cowman AF. Invasion by *P. falciparum* merozoites suggests a hierarchy of molecular interactions. *PLoS Pathog.* 2005;1(4):e37. PubMed PMID: 16362075.
5. Amino R, Thiberge S, Martin B, Celli S, Shorte S, Frischknecht F, Menard R. Quantitative imaging of *Plasmodium* transmission from mosquito to mammal. *Nat Med.* 2006;12(2):220–224. PubMed PMID: 16429144.
6. del Portillo HA, Lanzer M, Rodriguez-Malaga S, Zavala F, Fernandez-Becerra C. Variant genes and the spleen in *Plasmodium vivax* malaria. *Int J Parasitol.* 2004;34(13-14):1547–1554. PubMed PMID: 15582531.
7. Gardner MJ, Hall N, Fung E, White O, Berriman M, Hyman RW, Carlton JM, Pain A, Nelson KE, Bowman S, Paulsen IT, James K, Eisen JA, Rutherford K, Salzberg SL, Craig A, Kyes S, Chan MS, Nene V, Shallom SJ, Suh B, Peterson J, Angiuoli S, Pertea M, Allen J, Selengut J, Haft D, Mather MW, Vaidya AB, Martin DM, Fairlamb AH, Fraunholz MJ, Roos DS, Ralph SA, McFadden GI, Cummings LM, Subramanian GM, Mungai C, Venter JC, Carucci DJ, Hoffman SL, Newbold C, Davis RW, Fraser CM, Barrell B. Genome sequence of the human malaria parasite *Plasmodium falciparum*. *Nature.* 2002;419(6906):498–511. PubMed PMID: 12368864.
8. Carlton JM, Angiuoli SV, Suh BB, Kooij TW, Pertea M, Silva JC et al. Genome sequence and comparative analysis of the model rodent malaria parasite *Plasmodium yoelii yoelii*. *Nature.* 2002;419(6906):512–519. PubMed PMID: 12368865.
9. Hall N, Karras M, Raine JD, Carlton JM, Kooij TW, Berriman M et al. A comprehensive survey of the *Plasmodium* life cycle by genomic, transcriptomic, and proteomic analyses. *Science.* 2005;307(5706):82–86. PubMed PMID: 15637271.
10. del Portillo HA, Fernandez-Becerra C, Bowman S, Oliver K, Preuss M, Sanchez CP et al. A superfamily of variant genes encoded in the subtelomeric region of *Plasmodium vivax*. *Nature.* 2001;410(6830):839–842. PubMed PMID: 11298455.
11. Hayward RE, Derisi JL, Alfadhli S, Kaslow DC, Brown PO, Rathod PK. Shotgun DNA microarrays and stage-specific gene expression in *Plasmodium falciparum* malaria. *Mol Microbiol.* 2000;35(1):6–14. PubMed PMID: 10632873.
12. Ben Mamoun C, Gluzman IY, Hott C, MacMillan SK, Amarakone AS, Anderson DL et al. Co-ordinated programme of gene expression during asexual intraerythrocytic development of the human malaria parasite *Plasmodium falciparum* revealed by microarray analysis. *Mol Microbiol.* 2001;39(1):26–36. PubMed PMID: 11123685.
13. Bozdech Z, Llinas M, Pulliam BL, Wong ED, Zhu J, DeRisi JL. The transcriptome of the intraerythrocytic developmental cycle of *Plasmodium falciparum*. *PLoS Biol.* 2003;1(1):E5. PubMed PMID: 12929205.
14. Le Roch KG, Zhou Y, Blair PL, Grainger M, Moch JK, Haynes JD et al. Discovery of gene function by expression profiling of the malaria parasite life cycle. *Science.* 2003;301(5639):1503–1508. PubMed PMID: 12893887.
15. Kaiser K, Matuschewski K, Camargo N, Ross J, Kappe SH. Differential transcriptome profiling identifies *Plasmodium* genes encoding pre-erythrocytic stage-specific proteins. *Mol Microbiol.* 2004;51(5):1221–1232. PubMed PMID: 14982620.
16. Gruner AC, Hez-Deroubaix S, Snounou G, Hall N, Bouchier C, Letourneur F et al. Insights into the *P. y. yoelii* hepatic stage transcriptome reveal complex transcriptional patterns. *Mol Biochem Parasitol.* 2005;142(2):184–192. PubMed PMID: 15913805.
17. Sacci JB, Azad AF. Gene expression analysis during liver stage development of *Plasmodium*. *Int J Parasitol.* 2002;32(13):1551–1557. PubMed PMID: 12435439.
18. Wang Q, Brown S, Roos DS, Nussenzweig V, Bhanot P. Transcriptome of axenic liver stages of *Plasmodium yoelii*. *Mol Biochem Parasitol.* 2004;137(1):161–168. PubMed PMID: 15279962.

19. Daily JP, Le Roch KG, Sarr O, Fang X, Zhou Y, Ndir O.et al. In vivo transcriptional profiling of *Plasmodium falciparum*. *Malar J*. 2004;3:30. PubMed PMID: 15296511.
20. Daily JP, Le Roch KG, Sarr O, Ndiaye D, Lukens A, Zhou Y.et al. In vivo transcriptome of *Plasmodium falciparum* reveals overexpression of transcripts that encode surface proteins. *J Infect Dis*. 2005;191(7): 1196–1203. PubMed PMID: 15747257.
21. Quackenbush J. Computational analysis of microarray data. *Nat Rev Genet*. 2001;2:418–427. PubMed PMID: 11389458.
22. Florens L, Washburn MP, Raine JD, Anthony RM, Grainger M, Haynes JD.et al. A proteomic view of the *Plasmodium falciparum* life cycle. *Nature*. 2002;419(6906):520–526. PubMed PMID: 12368866.
23. Blair PL, Carucci DJ. Functional proteome and expression analysis of sporozoites and hepatic stages of malaria development. *Curr Top Microbiol Immunol*. 2005;295:417–438. PubMed PMID: 16265900.
24. Lasonder E, Ishihama Y, Andersen JS, Vermunt AM, Pain A, Sauerwein RW.et al. Analysis of the *Plasmodium falciparum* proteome by high-accuracy mass spectrometry. *Nature*. 2002;419(6906):537–542. PubMed PMID: 12368870.
25. Khan SM, Franke-Fayard B, Mair GR, Lasonder E, Janse CJ, Mann M.et al. Proteome analysis of separated male and female gametocytes reveals novel sex-specific *Plasmodium* biology. *Cell*. 2005;121(5):675–687. PubMed PMID: 15935755.
26. Bradley PJ, Ward C, Cheng SJ, Alexander DL, Collier S, Coombs GH.et al. Proteomic analysis of rhoptry organelles reveals many novel constituents for host-parasite interactions in *Toxoplasma gondii*. *J Biol Chem*. 2005;280(40):34245–34258. PubMed PMID: 16002398.
27. Sam-Yellowe TY, Florens L, Wang T, Raine JD, Carucci DJ, Sinden R.et al. Proteome analysis of rhoptry-enriched fractions isolated from *Plasmodium* merozoites. *J Proteome Res*. 2004;3(5):995–1001. PubMed PMID: 15473688.
28. Cary C, Lamont D, Dalton JP, Doerig C. *Plasmodium falciparum* chromatin: nucleosomal organisation and histone-like proteins. *Parasitol Res*. 1994;80(3):255–258. PubMed PMID: 8036241.
29. Lanzer M, de Bruin D, Wertheimer SP, Ravetch JV. Transcriptional and nucleosomal characterization of a subtelomeric gene cluster flanking a site of chromosomal rearrangements in *Plasmodium falciparum*. *Nucleic Acids Res*. 1994;22(20):4176–4182. PubMed PMID: 7937144.
30. Miao J, Fan Q, Cui L, Li J. The malaria parasite *Plasmodium falciparum* histones: Organization, expression, and acetylation. *Gene*. 2006;369:53–65. PubMed PMID: 16410041.
31. Meissner M, Soldati D. The transcription machinery and the molecular toolbox to control gene expression in *Toxoplasma gondii* and other protozoan parasites. *Microbes Infect*. 2005;7(13):1376–1384. PubMed PMID: 16087378.
32. Kasinsky HE, Lewis JD, Dacks JB, Ausio J. Origin of H1 linker histones. *FASEB J*. 2001;15(1):34–42. PubMed PMID: 11149891.
33. Chandra BR, Olivieri A, Silvestrini F, Alano P, Sharma A. Biochemical characterization of the two nucleosome assembly proteins from *Plasmodium falciparum*. *Mol Biochem Parasitol*. 2005;142(2):237–247. PubMed PMID: 15899528.
34. Horrocks P, Pinches R, Kriek N, Newbold C. Stage-specific promoter activity from stably maintained episomes in *Plasmodium falciparum*. *Int J Parasitol*. 2002;32(10):1203–1206. PubMed PMID: 12204219.
35. Fan Q, An L, Cui L. PfADA2, a *Plasmodium falciparum* homologue of the transcriptional coactivator ADA2 and its in vivo association with the histone acetyltransferase PfGCN5. *Gene*. 2004;336(2):251–261. PubMed PMID: 15246536.
36. Joshi MB, Lin DT, Chiang PH, Goldman ND, Fujioka H, Aikawa M.et al. Molecular cloning and nuclear localization of a histone deacetylase homologue in *Plasmodium falciparum*. *Mol Biochem Parasitol*. 1999;99(1):11–19. PubMed PMID: 10215020.
37. Scherf A, Figueiredo LM, Freitas-Junior LH. *Plasmodium* telomeres: a pathogen's perspective. *Curr Opin Microbiol*. 2001;4(4):409–414. PubMed PMID: 11495803.

38. Duraisingh MT, Voss TS, Marty AJ, Duffy MF, Good RT, Thompson JK. et al. Heterochromatin silencing and locus repositioning linked to regulation of virulence genes in *Plasmodium falciparum*. *Cell*. 2005;121(1):13–24. PubMed PMID: 15820675.
39. Freitas-Junior LH, Hernandez-Rivas R, Ralph SA, Montiel-Condado D, Ruvalcaba-Salazar OK, Rojas-Meza AP. et al. Telomeric heterochromatin propagation and histone acetylation control mutually exclusive expression of antigenic variation genes in malaria parasites. *Cell*. 2005;121(1):25–36. PubMed PMID: 15820676.
40. Wu Y, Sifri CD, Lei HH, Su XZ, Wellems TE. Transfection of *Plasmodium falciparum* within human red blood cells. *Proc Natl Acad Sci U S A*. 1995;92(4):973–977. PubMed PMID: 7862676.
41. Crabb BS, Cowman AF. Characterization of promoters and stable transfection by homologous and nonhomologous recombination in *Plasmodium falciparum*. *Proc Natl Acad Sci U S A*. 1996;93(14):7289–7294. PubMed PMID: 8692985.
42. Lanzer M, Wertheimer SP, de Bruin D, Ravetch JV. Chromatin structure determines the sites of chromosome breakages in *Plasmodium falciparum*. *Nucleic Acids Res*. 1994;22(15):3099–3103. PubMed PMID: 8065922.
43. Lanzer M, de Bruin D, Ravetch JV. Transcription mapping of a 100 kb locus of *Plasmodium falciparum* identifies an intergenic region in which transcription terminates and reinitiates. *EMBO J*. 1992;11(5):1949–1955. PubMed PMID: 1374714.
44. Wickham ME, Thompson JK, Cowman AF. Characterisation of the merozoite surface protein-2 promoter using stable and transient transfection in *Plasmodium falciparum*. *Mol Biochem Parasitol*. 2003;129(2):147–156. PubMed PMID: 12850259.
45. de Koning-Ward TF, Speranca MA, Waters AP, Janse CJ. Analysis of stage specificity of promoters in *Plasmodium berghei* using luciferase as a reporter. *Mol Biochem Parasitol*. 1999;100(1):141–146. PubMed PMID: 10377003.
46. Dechering KJ, Kaan AM, Mbacham W, Wirth DF, Eling W, Konings RN. et al. Isolation and functional characterization of two distinct sexual-stage-specific promoters of the human malaria parasite *Plasmodium falciparum*. *Mol Cell Biol*. 1999;19(2):967–978. PubMed PMID: 9891033.
47. van der Wel AM, Tomas AM, Kocken CH, Malhotra P, Janse CJ, Waters AP. et al. Transfection of the primate malaria parasite *Plasmodium knowlesi* using entirely heterologous constructs. *J Exp Med*. 1997;185(8):1499–1503. PubMed PMID: 9126931.
48. Mota MM, Thathy V, Nussenzweig RS, Nussenzweig V. Gene targeting in the rodent malaria parasite *Plasmodium yoelii*. *Mol Biochem Parasitol*. 2001;113(2):271–278. PubMed PMID: 11295181.
49. Fernandez-Becerra C, de Azevedo MF, Yamamoto MM, del Portillo HA. *Plasmodium falciparum*: new vector with bi-directional promoter activity to stably express transgenes. *Exp Parasitol*. 2003;103(1-2):88–91. PubMed PMID: 12810052.
50. Ferreira de Azevedo M, del Portillo HA. *Mal J*. 2007;6:20. PubMed PMID: 16462797.
51. Horrocks P, Lanzer M. Mutational analysis identifies a five base pair cis-acting sequence essential for GBP130 promoter activity in *Plasmodium falciparum*. *Mol Biochem Parasitol*. 1999;99(1):77–87. PubMed PMID: 10215026.
52. Mbacham WF, Chow CS, Daily J, Golightly LM, Wirth DF. Deletion analysis of the 5' flanking sequence of the *Plasmodium gallinaceum* sexual stage specific gene pgs28 suggests a bipartite arrangement of cis-control elements. *Mol Biochem Parasitol*. 2001;113(1):183–187. PubMed PMID: 11254967.
53. Porter ME. The DNA polymerase delta promoter from *Plasmodium falciparum* contains an unusually long 5' untranslated region and intrinsic DNA curvature. *Mol Biochem Parasitol*. 2001;114(2):249–255. PubMed PMID: 11378205.
54. Osta M, Gannoun-Zaki L, Bonnefoy S, Roy C, Vial HJ. A 24 bp cis-acting element essential for the transcriptional activity of *Plasmodium falciparum* CDP-diacylglycerol synthase gene promoter. *Mol Biochem Parasitol*. 2002;121(1):87–98. PubMed PMID: 11985865.

55. Voss TS, Kaestli M, Vogel D, Bopp S, Beck HP. Identification of nuclear proteins that interact differentially with *Plasmodium falciparum* var gene promoters. *Mol Microbiol.* 2003;48(6):1593–1607. PubMed PMID: 12791141.
56. Chow CS, Wirth DF. Linker scanning mutagenesis of the *Plasmodium gallinaceum* sexual stage specific gene *pgs28* reveals a novel downstream cis-control element. *Mol Biochem Parasitol.* 2003;129(2):199–208. PubMed PMID: 12850264.
57. Militello KT, Dodge M, Bethke L, Wirth DF. Identification of regulatory elements in the *Plasmodium falciparum* genome. *Mol Biochem Parasitol.* 2004;134(1):75–88. PubMed PMID: 14747145.
58. Ruvalcaba-Salazar OK, del Carmen Ramirez-Estudillo M, Montiel-Condado D, Recillas-Targa F, Vargas M, Hernandez-Rivas R. Recombinant and native *Plasmodium falciparum* TATA-binding-protein binds to a specific TATA box element in promoter regions. *Mol Biochem Parasitol.* 2005;140(2):183–196. PubMed PMID: 15760658.
59. Polson HE, Blackman MJ. A role for poly(dA)poly(dT) tracts in directing activity of the *Plasmodium falciparum* calmodulin gene promoter. *Mol Biochem Parasitol.* 2005;141(2):179–189. PubMed PMID: 15850701.
60. Bzik DJ. The structure and role of RNA polymerases in *Plasmodium*. *Parasitol Today.* 1991;7(8):211–214. PubMed PMID: 15463499.
61. Fox BA, Li WB, Tanaka M, Inselburg J, Bzik DJ. Molecular characterization of the largest subunit of *Plasmodium falciparum* RNA polymerase I. *Mol Biochem Parasitol.* 1993;61(1):37–48. PubMed PMID: 8259131.
62. Li WB, Bzik DJ, Gu HM, Tanaka M, Fox BA, Inselburg J. An enlarged largest subunit of *Plasmodium falciparum* RNA polymerase II defines conserved and variable RNA polymerase domains. *Nucleic Acids Res.* 1989;17(23):9621–9636. PubMed PMID: 2690004.
63. Giesecke H, Barale JC, Langsley G, Cornelissen AW. The C-terminal domain of RNA polymerase II of the malaria parasite *Plasmodium berghei*. *Biochem Biophys Res Commun.* 1991;180(3):1350–1355. PubMed PMID: 1840489.
64. Li J, Maga JA, Cermakian N, Cedergren R, Feagin JE. Identification and characterization of a *Plasmodium falciparum* RNA polymerase gene with similarity to mitochondrial RNA polymerases. *Mol Biochem Parasitol.* 2001;113(2):261–269. PubMed PMID: 11295180.
65. Gardner MJ, Goldman N, Barnett P, Moore PW, Rangachari K, Strath M. et al. Phylogenetic analysis of the *rpoB* gene from the plastid-like DNA of *Plasmodium falciparum*. *Mol Biochem Parasitol.* 1994;66(2):221–231. PubMed PMID: 7808472.
66. Ho CK, Shuman S. A yeast-like mRNA capping apparatus in *Plasmodium falciparum*. *Proc Natl Acad Sci U S A.* 2001;98(6):3050–3055. PubMed PMID: 11248030.
67. Hausmann S, Altura MA, Witmer M, Singer SM, Elmendorf HG, Shuman S. Yeast-like mRNA capping apparatus in *Giardia lamblia*. *J Biol Chem.* 2005;280(13):12077–12086. PubMed PMID: 15556935.
68. Knapp B, Nau U, Hundt E, Kupper HA. Demonstration of alternative splicing of a pre-mRNA expressed in the blood stage form of *Plasmodium falciparum*. *J Biol Chem.* 1991;266(11):7148–7154. PubMed PMID: 1707880.
69. van Lin LH, Pace T, Janse CJ, Birago C, Ramesar J, Picci L. et al. Interspecies conservation of gene order and intron-exon structure in a genomic locus of high gene density and complexity in *Plasmodium*. *Nucleic Acids Res.* 2001;29(10):2059–2068. PubMed PMID: 11353075.
70. Singh N, Preiser P, Renia L, Balu B, Barnwell J, Blair P. et al. Conservation and developmental control of alternative splicing in *maebl* among malaria parasites. *J Mol Biol.* 2004;343(3):589–599. PubMed PMID: 15465047.
71. Pace T, Birago C, Janse CJ, Picci L, Ponzi M. Developmental regulation of a *Plasmodium* gene involves the generation of stage-specific 5' untranslated sequences. *Mol Biochem Parasitol.* 1998;97(1-2):45–53. PubMed PMID: 9879886.

72. Kim M, Krogan NJ, Vasiljeva L, Rando OJ, Nedea E, Greenblatt JF. et al. The yeast Rat1 exonuclease promotes transcription termination by RNA polymerase II. *Nature*. 2004;432(7016):517–522. PubMed PMID: 15565157.
73. Ruvolo V, Altszuler R, Levitt A. The transcript encoding the circumsporozoite antigen of *Plasmodium berghei* utilizes heterogeneous polyadenylation sites. *Mol Biochem Parasitol*. 1993;57(1):137–150. PubMed PMID: 8093973.
74. Lanzer M, Wertheimer SP, de Bruin D, Ravetch JV. *Plasmodium*: control of gene expression in malaria parasites. *Exp Parasitol*. 1993;77(1):121–128. PubMed PMID: 8344402.
75. Golightly LM, Mbacham W, Daily J, Wirth DF. 3' UTR elements enhance expression of Pgs28, an ookinete protein of *Plasmodium gallinaceum*. *Mol Biochem Parasitol*. 2000;105(1):61–70. PubMed PMID: 10613699.
76. Shue P, Brown SV, Cann H, Singer EF, Appleby S, Golightly LM. The 3' UTR elements of *P. gallinaceum* protein Pgs28 are functionally distinct from those of human cells. *Mol Biochem Parasitol*. 2004;137(2):355–359. PubMed PMID: 15383307.
77. Wu Y, Kirkman LA, Wellems TE. Transformation of *Plasmodium falciparum* malaria parasites by homologous integration of plasmids that confer resistance to pyrimethamine. *Proc Natl Acad Sci U S A*. 1996;93(3):1130–1134. PubMed PMID: 8577727.
78. Goonewardene R, Daily J, Kaslow D, Sullivan TJ, Duffy P, Carter R. et al. Transfection of the malaria parasite and expression of firefly luciferase. *Proc Natl Acad Sci U S A*. 1993;90(11):5234–5236. PubMed PMID: 8506371.
79. Coulson RM, Hall N, Ouzounis CA. Comparative genomics of transcriptional control in the human malaria parasite *Plasmodium falciparum*. *Genome Res*. 2004;14(8):1548–1554. PubMed PMID: 15256513.
80. Callebaut I, Prat K, Meurice E, Mornon JP, Tomavo S. Prediction of the general transcription factors associated with RNA polymerase II in *Plasmodium falciparum*: conserved features and differences relative to other eukaryotes. *BMC Genomics*. 2005;6:100. PubMed PMID: 16042788.
81. Balaji S, Babu MM, Iyer LM, Aravind L. Discovery of the principal specific transcription factors of Apicomplexa and their implication for the evolution of the AP2-integrase DNA binding domains. *Nucleic Acids Res*. 2005;33(13):3994–4006. PubMed PMID: 16040597.
82. McAndrew MB, Read M, Sims PF, Hyde JE. Characterisation of the gene encoding an unusually divergent TATA-binding protein (TBP) from the extremely A+T-rich human malaria parasite *Plasmodium falciparum*. *Gene*. 1993;124(2):165–171. PubMed PMID: 8444340.
83. Boschet C, Gissot M, Briquet S, Hamid Z, Claudel-Renard C, Vaquero C. Characterization of PfMyb1 transcription factor during erythrocytic development of 3D7 and F12 *Plasmodium falciparum* clones. *Mol Biochem Parasitol*. 2004;138(1):159–163. PubMed PMID: 15500927.
84. van Noort V, Huynen MA. Combinatorial gene regulation in *Plasmodium falciparum*. *Trends Genet*. 2006;22(2):73–78. PubMed PMID: 16380193.
85. Paton MG, Barker GC, Matsuoka H, Ramesar J, Janse CJ, Waters AP. et al. Structure and expression of a post-transcriptionally regulated malaria gene encoding a surface protein from the sexual stages of *Plasmodium berghei*. *Mol Biochem Parasitol*. 1993;59(2):263–275. PubMed PMID: 8341324.
86. Zhang K, Rathod PK. Divergent regulation of dihydrofolate reductase between malaria parasite and human host. *Science*. 2002;296(5567):545–547. PubMed PMID: 11964483.
87. Cui L, Fan Q, Li J. The malaria parasite *Plasmodium falciparum* encodes members of the Puf RNA-binding protein family with conserved RNA binding activity. *Nucleic Acids Res*. 2002;30(21):4607–4617. PubMed PMID: 12409450.
88. Fan Q, Li J, Kariuki M, Cui L. Characterization of PfPuf2, member of the Puf family RNA-binding proteins from the malaria parasite *Plasmodium falciparum*. *DNA Cell Biol*. 2004;23(11):753–760. PubMed PMID: 15585133.
89. Mair GR, BraksGarver LS, Wiegant JC, Hall N, Dirks RW, Khan SM, Dimopoulos G, Janse CJ, Waters AP. Regulation of sexual development of *Plasmodium* by translational repression. *Science*. 2006;313:667–669. PubMed PMID: 16462797.

90. Brantl S. Antisense-RNA regulation and RNA interference. *Biochim Biophys Acta*. 2002;1575(1-3):15–25. PubMed PMID: 12020814.
91. Gunasekera AM, Patankar S, Schug J, Eisen G, Kissinger J, Roos D. et al. Widespread distribution of antisense transcripts in the *Plasmodium falciparum* genome. *Mol Biochem Parasitol*. 2004;136(1):35–42. PubMed PMID: 15138065.
92. Patankar S, Munasinghe A, Shoaibi A, Cummings LM, Wirth DF. Serial analysis of gene expression in *Plasmodium falciparum* reveals the global expression profile of erythrocytic stages and the presence of anti-sense transcripts in the malarial parasite. *Mol Biol Cell*. 2001;12(10):3114–3125. PubMed PMID: 11598196.
93. Militello KT, Patel V, Chessler AD, Fisher JK, Kasper JM, Gunasekera A. et al. RNA polymerase II synthesizes antisense RNA in *Plasmodium falciparum*. *RNA*. 2005;11(4):365–370. PubMed PMID: 15703443.
94. Gardiner DL, Holt DC, Thomas EA, Kemp DJ, Trenholme KR. Inhibition of *Plasmodium falciparum* clag9 gene function by antisense RNA. *Mol Biochem Parasitol*. 2000;110(1):33–41. PubMed PMID: 10989143.
95. Gunderson JH, Sogin ML, Wollett G, Hollingdale M, de la Cruz VF, Waters AP. et al. Structurally distinct, stage-specific ribosomes occur in *Plasmodium*. *Science*. 1987;238(4829):933–937. PubMed PMID: 3672135.
96. Waters AP, Syin C, McCutchan TF. Developmental regulation of stage-specific ribosome populations in *Plasmodium*. *Nature*. 1989;342(6248):438–440. PubMed PMID: 2586613.
97. van Spaendonk RM, Ramesar J, van Wigcheren A, Eling W, Beetsma AL, van Gemert GJ. et al. Functional equivalence of structurally distinct ribosomes in the malaria parasite, *Plasmodium berghei*. *J Biol Chem*. 2001;276(25):22638–22647. PubMed PMID: 11292830.
98. Rogers MJ, Gutell RR, Damberger SH, Li J, McConkey GA, Waters AP. et al. Structural features of the large subunit rRNA expressed in *Plasmodium falciparum* sporozoites that distinguish it from the asexually expressed subunit rRNA. *RNA*. 1996;2(2):134–145. PubMed PMID: 8601280.
99. Wingender E, Dietze P, Karas H, Knuppel R. TRANSFAC: a database on transcription factors and their DNA binding sites. *Nucleic Acids Res*. 1996;24(1) PubMed PMID: 8601280.
100. Roth FP, Hughes JD, Estep PW, Church GM. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat Biotechnol*. 1998;16(10):939–945. PubMed PMID: 9788350.
101. Hughes JD, Estep PW, Tavazoie S, Church GM. Computational identification of cis-regulatory elements associated with groups of functionally related genes in *Saccharomyces cerevisiae*. *J Mol Biol*. 2000;296(5): 1205–14. PubMed PMID: 10698627.
102. Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, Futcher B. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol Biol Cell*. 1998;9:3273–3297. PubMed PMID: 9843569.
103. Barrera ,et al. 2004.
104. Jones MK, Good MF. Malaria parasites up close. *Nat Med*. 2006;12(2):170–171. PubMed PMID: 16462797.



## Chapter B02. *Leishmania* Genomics: Where Do We Stand?

Silvia R.B Uliana, MD, PhD,<sup>1</sup> Jeronimo C. Ruiz, PhD,<sup>2</sup> and Angela K. Cruz, PhD<sup>3</sup>

Created: October 12, 2006; Updated: August 24, 2007.

### A Short History of the *Leishmania* Genome Project

More than 20 different species of the genus *Leishmania* are known to be pathogenic to humans. These protozoan parasites are kinetoplastids of the Trypanosomatidae family and depend on a female Phlebotominae insect and on a variety of vertebrate hosts to complete its vital cycle. Collectively, *Leishmania* spp. are responsible for one of the world's major communicable diseases, which lead the World Health Organization to include leishmaniasis among the six major diseases targeted for intensive research and control efforts. It is estimated that more than 2 million new cases of leishmaniasis occur each year in 88 countries (<http://www.who.int/health-topics/leishmaniasis.htm>). Leishmaniasis is a spectral disease with multifaceted clinical manifestations, varying from mild and frequently self-healing cutaneous lesions to severe mucocutaneous ulcers or to visceral manifestations, which can lead to death.

Besides its unquestionable medical relevance, this ancient eukaryote is an important biological model. *Leishmania* are mostly diploid organisms with no apparent sexual cycle displaying some uncommon biochemical, genetic, and morphological features that are either unique to the order Kinetoplastida or are more frequently used by these organisms than by any other. These features include a unique mitochondrial DNA organization, the kinetoplast, extensive mitochondrial DNA editing (1), glycosomes (2), polycistronic transcription (3), trans-splicing (4, 5), and GPI anchoring of membrane proteins, among others (6-8).

Because leishmaniasis is a Third World endemic disease, there is almost no interest on the part of the pharmaceutical companies to search for low-toxicity drugs or invest in the development of safe and effective vaccines to fight the disease. Therefore, deciphering the genomic sequence of the parasite was taken as an option to optimize approaches and accelerate the understanding of the biology of this powerful human parasite and its interaction with the host. The ambience of several successful, ongoing genome projects, particularly those of pathogenic organisms, was the background to put forward a genome project for *Leishmania* and other relevant Trypanosomatids (9).

The first Parasite Genome Network Planning Meeting was held in April 1994 in Rio de Janeiro, Brazil, and was sponsored by FIOCRUZ and UNICEF/UNDP/World Bank/WHO Special Programme for Research and Training in Tropical Diseases (TDR). The Genome Projects of *Trypanosoma cruzi*, *Trypanosoma brucei*, and *Leishmania major* were launched at the Meeting, and reference strains for each one of the parasites were chosen for the sequencing project. A *Leishmania* Genome Network (LGN) was established, and several laboratories from all over the world got involved in the initiative.

The LGN major aims were: 1) to develop a physical map of the parasite's genome; 2) to curate, analyze, and disseminate parasite genome data; and 3) to characterize the entire genomic sequence of the organism. These core objectives would allow progress in different areas from discovery of new drug targets, diagnostic and/or vaccine approaches, and improvement of knowledge on the parasite biology to disseminate expertise in genomics to different countries.

Pulsed-field gel electrophoresis (PFGE) was used to generate "molecular karyotypes" for distinct *Leishmania* species and strains. A combination of PFGE and hybridization analysis revealed that an observed chromosome

size variation was not a consequence of clonal heterogeneity of strains, and that the variation was relatively small (10 to 20%) and mainly attributable to expansion and deletion of subtelomeric regions (10). In landmark studies, Wincker and colleagues (10) defined 36 physical linkage groups corresponding to the complete set of chromosomes, suggesting a genome size of 35 Mb. They not only showed that these physical linkage groups were highly conserved in Old World species but were also mostly conserved in New World species (11). The LGN group extended the analysis to construct the *Leishmania major* Friedlin (LmjF) reference strain molecular karyotype (12). These data are evidence of the conservation of syntenic (conservation of gene order) groups among species. Understanding the extent of conservation in genome organization was a very important step at that time, confirming that the use of a single reference strain for all *Leishmania* species was an appropriate decision.

Another important work front during the mid- and late-1990s was the construction of a physical map for the LmjF chromosomes. This approach was chosen to allow the selection of a tileset of clones necessary to span the entire genome. The map was built from cloned DNAs of large insert genomic libraries (13). The selected clones would then be sequenced, and a large proportion of the mapping phase of the genome project involved the systematic choice of contig end-probing for contig-joining activities. At that point, given hardware and software features and the lack of funds, full LmjF genome shotgun sequencing was not considered a feasible strategy.

A parallel activity during the early years of the *Leishmania* Genome Project was the sequencing of expressed genomic libraries, seen at that time as a fast and simple means toward gene discovery. Single-pass sequencing of clones from cDNA libraries from different life cycle stages was conducted, generating more than 2,500 Expressed Sequence Tags (ESTs). EST sequencing was originally identified as an efficient, cost-effective method for gene discovery in relatively large genomes, such as the protozoan ones. Nevertheless, other studies demonstrated that *T. brucei* and *Leishmania* gene-dense genomes and non-intronic genes would make genomic sequencing a much better option, even for gene discovery purposes (14, 15).

The complete sequencing of chromosome 1 was initiated in 1996, as a pilot project of physical map-based sequencing, and was published by Myler and coworkers (16) in 1999. A striking polarity of putative transcription was shown and confirmed for the genomes of *Leishmania*, *T. brucei*, and *T. cruzi*. The so-called Directional Gene Clusters (DGCs), a unique characteristic of gene organization of Trypanosomatids to be discussed further (below), may be a consequence of the polycistronic mode of transcription and the unusual linkage between transcription and RNA processing involving a coupled mechanism of *trans*-splicing and polyadenylation (17, 18).

The rest of the genome came afterwards, and the 33.6-Mb sequence was obtained by a combination of approaches ranging from hierarchical sequencing strategy, or the clone-by-clone approach, and Whole Chromosome Shotgun (WGS), which involves an initial separation step of individual or co-migrating chromosomes by PFGE. *In silico* analysis of the *L. major* genome predicted 911 RNA genes, 39 pseudogenes, and 8,370 protein-coding genes. The entire *Leishmania* genome sequence was published along with the *Trypanosoma brucei* and *T. cruzi* genomes in the July 2005 Science issue (19). This marks the conquest of an international effort that, as already said (20), must not be taken as a final achievement but as the beginning of several possible investigation routes.

The *Leishmania* spp. genomes are now being exploited with a comparative genomics approach: the genomes of *L. infantum* and *L. braziliensis* have been completely sequenced and annotated (21). It is worthwhile mentioning that sequencing of each of these two genomes was accomplished in less than 1 year. Moreover, all the recent software and hardware improvements allowed all of the sequencing to be completed by the whole genomic DNA shotgun sequencing approach.

## Particular Features of *Leishmania* Genome Organization in Comparison with Other Eukaryotic Genomes

Mapping and sequencing of the *L. major* genome allowed the definition of 36 discrete chromosomes composing the genome of this protozoan parasite. Different *Leishmania* species have 34 to 36 chromosomes, varying in size from 268 to 2,680 kb. As mentioned above, chromosomes do not condense during cell division, and the genome is largely diploid and homozygous.

The 33.6-Mb haploid genome of *L. major* has an overall G+C content of 59.7%. The 8,370 predicted protein-coding genes are densely arranged and occupy about 48% of the genome (19) (<http://www.genedb.org/genedb/leish/>). Many protein-coding genes are grouped in multicopy families represented in tandem arrays.

*L. major* genome is relatively poor in repetitive sequences when compared with the other Trypanosomatids (19, 22). Interspersed elements are not as frequent, and active retrotransposable elements have not been found (23). Major repeated elements in *L. major* are found exclusively in subtelomeric regions (24). As mentioned above, contraction and expansion of these subtelomeric regions are responsible for the size variation observed between lines and chromosome homologs (25).

*Leishmania* telomeres contain variable numbers of the hexameric repeat 5'-TAGGGT-3', variable numbers of the octameric repeat 5'-TGGTCATG-3', and a sequence immediately adjacent to the telomere, common to all *Leishmania* species and named *Leishmania* conserved telomere-associated sequence (LCTAS) (26). The LCTAS can be present as a single copy at the end of the chromosome or in repeated copies intercalated by the telomere end-most hexameric repeat. In the subtelomeric region, some other, less conserved repeated elements are found, followed by a barren region (27).

One of the most remarkable and particular characteristics of *Leishmania* and *Trypanosoma* genomes is the organization of genes into long arrays of tandem coding sequences, directionally positioned and transcribed into polycistronic primary transcripts. These DGCs are not homogeneous in size, varying from a few kb to more than 1.2 Mb. Sequences localized between two divergent transcription units are called strand-switch regions, and the nucleotide sequences in these regions do not display a particular consensus sequence for any RNA polymerase known promoter but show a composition bias with increased AT content in relation to other chromosomal regions (28).

Transcription by RNA polymerase II (RNA pol II) in these organisms is unique. After the completion of chromosome 1 (chr1) of *L. major* (16), the study of its transcriptional activity showed that coding strand-specific RNA pol II transcribing activity was initiated in the strand-switch region and proceeded bidirectionally (7). Similarly, on chr3, transcription starts at discrete regions (perhaps not exclusively, but more intensely) and proceeds along the DGC (6). No substantial sequence similarity was found between the regions of transcription initiation identified on LmjF chr1 and chr3. Previous work had shown that RNA pol II initiates transcription at random in *Leishmania*, not requiring a specific promoter (29). Nevertheless, in intact chromosomes, RNA pol II transcription is predominantly, but not exclusively, strand specific and unidirectional (30). As a result, the full complements of DGCs are transcribed at the same rate.

Interestingly, even if a characteristic RNA pol II promoter region does not seem to exist preceding the long DGCs, a typical and discrete pol II promoter has been found and characterized in the intergenic sequences of the tandem array encoding about 120 copies of the spliced leader RNA (SL RNA) gene (31-33). Several conserved genes encoding subunits of basal transcription factors for RNA pol II have been identified in the *Leishmania* genome, but some homologs for several of those have not been found thus far, making the mechanism of RNA transcription in these organisms a theme for further investigation.

The tandem organization of ribosomal RNA (rRNA) genes and the characteristics of RNA polymerase I (RNA pol I) promoters resemble the structure present in other organisms. One large tandem array contains approximately 60 copies of the genes for 24S, 18S, and 5.8S rRNA genes, whereas 5S rRNA genes are dispersed in 11 different loci. RNA pol I promoters are present and exhibit the expected characteristics: they are strong promoters, insensitive to  $\alpha$ -amanitin, poorly conserved in sequence, and regulated by upstream repeated motifs (34-37). RNA pol I is responsible for the transcription of the rRNA units (6) in *Leishmania* and, again unusually, variant surface antigen-coding genes in *T. brucei*, a singular example of transcription of protein-coding genes by RNA pol I. Transcription of tRNAs and small RNAs seems to be driven by RNA pol III promoters (6).

Another atypical feature of the *Leishmania* (and Trypanosomatids in general) genome is the almost complete absence of introns. Very few examples of *cis*-spliced genes have been found thus far in these organisms (19, 38).

Over 40 examples of sequences for which there is strong evidence of horizontal transfer from bacteria were found in the *L. major* genome. Some of them are exclusive of *Leishmania*, whereas many others are shared with *Trypanosoma* species.

## Control of Gene Expression in *Leishmania*

During its life cycle, *Leishmania* alternates between the extracellular promastigote that multiplies in the digestive tract of the insect vector and the intracellular amastigote, a parasite of macrophages in the vertebrate host (<http://www.dpd.cdc.gov/DPDx/HTML/Leishmaniasis.htm>). After colonization and multiplication of procyclic promastigotes in the insect gut and before the inoculation of infective parasites with the next blood meal, promastigotes undergo a developmental differentiation process known as metacyclogenesis (39). Procyclic and metacyclic promastigotes and amastigotes have the ability to differentially express gene products, as exemplified by numerous genes and proteins already characterized (40-48).

Although in most organisms a first and decisive level of gene expression control is given by the transcription rate of a particular sequence, this is clearly not the case in *Leishmania*, as discussed above. To increase transcription, *Leishmania* parasites rely on duplication or amplification of gene sequences, and these organisms reveal a striking genome plasticity, being able to respond to pressure by changing the ploidy locally or in the whole genome (49-52). Consequently, other than amplifying the whole genome, *Leishmania* has to rely on post-transcriptional mechanisms to control gene expression.

One of the consequences of polycistronic transcription is the lack of capping at the 5' end of discrete gene transcripts, making them unstable. Stability of mRNAs is achieved in these organisms by *trans*-splicing. This type of RNA processing was first described in Trypanosomatids and later demonstrated also in other organisms.

The *trans*-splicing machinery is responsible for transferring a capped small RNA, known as spliced-leader (SL), to the 5' end of almost all mRNAs (53, 54). The acceptor site for the SL RNA is generally an AG dinucleotide preceded by a pyrimidine-rich sequence upstream from the open reading frame (ORF) by 100-250 nt. The polypyrimidine tract is also responsible for directing polyadenylation of the upstream mRNA (18, 53, 54).

The rate of splicing and maturation of RNA is, therefore, one of the levels for the control of gene expression in *Leishmania*. Examples of alternative splicing of transcripts have been described in *T. cruzi* and *T. brucei* (55, 56).

On the other hand, the control of RNA degradation has been shown to play a part in the regulation of gene expression. In several cases, the presence of particular motifs in the 3'-untranslated region (3'UTR) has been shown to be essential for regulating the mRNA half-life (43, 57-59). Strong evidence of the requirement for *de novo* protein synthesis to occur indicates that degradation or stabilization of an mRNA takes place through protein interaction with the 3'UTR sequence (60), and accordingly, several genes for proteins with RNA-binding motifs were identified. Regulatory sequences in the 3'UTR can also regulate translation, as has been shown recently for amastin and heat shock proteins 70 and 83 (61-63).

RNA editing represents still another mechanism of gene expression control. Primary transcripts for several mitochondrial genes do not encode functional ORFs, and the explanation for that eluded many scientists until the demonstration by Benne *et al.* (1) that primary transcripts were modified by the addition or deletion of uridines, providing for a functional ORF for several mitochondrial enzymes. The editing phenomena was later shown to be based on small complementary RNA molecules encoded by the kinetoplast DNA minicircles (64, 65). These small RNAs were called guide RNAs (gRNA).

## Experimental Approaches for Gene Function Studies: Reverse and Forward Genetics

*Leishmania* genetic features did not allow studies using classical genetic approaches. Therefore, until the late 1980s, gene function was inferred from similarity to genes from other organisms or expression in heterologous systems. The breakthrough for Trypanosomatids' reverse genetics started with the transient expression of reporter genes in *Leptomonas* spp. (66). Shortly after, appropriate conditions for transient and stable transfections were established for *Leishmania* (67-69).

Obligatory features of vectors for transient and stable transfections were determined (18, 29, 70). In both cases, promoterless circular plasmids gave readily detectable expression in *Leishmania*. The expression is dependent on the presence of the polypyrimidine track and acceptor site (a dinucleotide AG) for *trans*-splicing positioned upstream to a reporter gene, in the case of transient expression, or to a selectable marker in vectors designed for stable transfection. Several vectors, reporter genes, and positive or negative selectable markers are now available (71). Downstream to the gene to be expressed, a 3'UTR from a resident gene is added to ensure the proper polyA tail addition. Positional cloning between the 5' and 3' UTRs, as described above, attains transcription of any given gene. Transient transfection vectors may be designed to contain an endogenous promoter for RNA pol I genes or a heterologous one, such as phage T7 promoter, which drives transcription more efficiently (34, 71, 72).

Data now available on 3'UTR sequence elements that are able to regulate the stability and translation efficiency of RNA (72, 73) may be of help for the future development of vectors designed to achieve defined patterns of developmental expression, which will represent another relevant instrument for reverse genetics.

Long genomic fragments cloned into cosmid vectors have been used for expression of several genes at once, becoming an invaluable tool for functional complementation studies. Cosmid genomic libraries are very useful to rescue phenotypes and unravel the function of novel genes (74, 75). Functional studies carried out with such recombinants have offered evidence that in episomes, transcription happens from both strands, and no weakening attributable to the position of the gene within an insert was observed (76).

For technical reasons, transient transfection has limited applications: it demands high amounts of plasmid DNA for detection of the reporter gene expression, and the attainable levels do not allow subcellular localization of the exogenous protein. Its main use is restricted to searches for sequence elements involved in gene regulation. Therefore, permanently transformed cell lines are the preferred approach for most of the transfection applications. Expression of genes present in the episomes can be enhanced by increasing the concentration of selective drug, which leads to amplification of the plasmid copy number and to higher levels of transcription. This is a very convenient approach because it allows tuning up the level of drug pressure and overexpression of the gene of interest in the episome, making possible a correlation between the gene product and a given phenotype.

Nevertheless, it is also well established that the absence of drug pressure will lead to a loss of circular extrachromosomal molecules, and it is true that after a few serial passages, the original clone is turned into a mixed population of cells, with and without the episome. The stability of the exogenous gene can be achieved by its integration into the genome. To promote integration of DNA into the *Leishmania* genome, a linear fragment with non-cohesive ends must be transfected. The *sine qua non* condition for insertion is the presence of sequence

homology between the transfected fragment and the integration site in the genome (77). The high frequency and efficiency of homologous recombination are used to integrate an exogenous gene to be overexpressed in the rDNA locus. This tactic places the gene under the control of the only known strong promoter, which drives constitutive expression in *Leishmania*. Transcription powered by RNA pol I in such an arrangement not only leads to overexpression of the gene in higher levels than obtained by RNA pol II-driven expression of episomal genes but brings the advantage of stabilization of the overexpressed gene (78).

More importantly, the high frequency and efficiency of homologous recombination made it possible to knock out genes using linear fragments designed to replace a resident gene with a selectable marker (77). Gene replacement happens after a double crossing-over event between homologous sequences flanking the target gene and the selectable marker. Relevant technical features to obtain genome insertion with high efficiency have been explored by different groups (77, 79). There are no sound reports of non-homologous integration in the *Leishmania* genome. The parasite cell offers favorable homologous recombination machinery but, as mentioned above, *Leishmania* is mostly diploid and has no manipulatable sexual cycle, making the knock-out of a given gene more difficult. Two rounds of transfection and two different selectable markers are the minimal requirements to obtain a null mutant for a given locus (80). Targeting occurs with comparable efficiencies at both steps and independently of the order of the selectable markers used in each transfection. Several genes have been knocked out using this two-step strategy since 1990, and it has even been shown that long stretches of genomic sequences (40 kb) can be deleted by homologous replacement (81, 82). This finding shows that gene clusters with several copies can be targeted at once.

Interestingly, an unsuccessful double replacement of a given gene is now taken as indicative of a gene essential for the parasite's survival. Several reports on different targets demonstrate that the parasite uses extreme genetic means to avoid the loss of an essential gene. These reports were also very important to reveal that ploidy in *Leishmania* is anything but strict: alterations of chromosome number either by aneuploidy or tetraploidy are recurrent findings (49, 52). Such a limitation for the study of essential genes has been overcome by alternative approaches in trypanosomes but not in *Leishmania*. Generation of conditional knock-outs using a tetracycline-inducible system to fine-tune the level of expression of an essential gene has been successfully established in *T. brucei* (83), but it is not fully operational in *Leishmania* (84). An efficient inducible system for functional studies of genes that cannot be knocked out would be a remarkably useful tool for reverse genetics in *Leishmania*.

Gene silencing or RNAi technology as a means to knock down genes is another extremely important technical advance for functional genetic studies that is booming in *T. brucei* but missing in *Leishmania*. In African trypanosomes, a stem-loop or a double-stranded RNA can be transfected or directly introduced in the parasite and can be used in conjunction with the *tet* repressor system to knock down any given gene transiently. This approach allows the quick investigation of several genes in parallel and has been widely used and shown to be the best choice for functional genomics in several organisms including *T. brucei*.

## Global Approaches for Functional Analyses of the Parasite Genome: Transcriptome and Proteome

Before the completion of the genome, investigators devoted to functional studies in *Leishmania* had to use "one gene at a time" strategies, applying the available reverse and forward genetics technology. There is now an urgent need for global approaches to functionally tackle the multitude of data available. The completion of the Tri-Tryp sequencing represents a landmark in the study of these human pathogens. The analyses of the genome sequence and comparative genomics has already generated an enormous amount of information. But within the available data lies an enormous number of unanswered questions. For example, of the 8,370 *L. major* predicted protein-coding genes, about 10% seem to be unique, having no orthologs in *Trypanosoma* or other organisms (19, 22) (<http://www.genedb.org/genedb/leish/index.jsp>). The function of these unknown genes, as well as the

understanding of the physiology of these parasites, will depend on new technical developments, some of which are already in use for the study of the *Leishmania* genome as well as other genomes.

Possible approaches to large-scale functional studies involve the characterization of the complete set of RNA molecules (known as the transcriptome), of the complete set of proteins (the proteome), or of the complete set of low molecular weight metabolic intermediates (the metabolome), as well as the generation of genome-wide, single-gene replacements.

Examples of the study of a full genome aimed at identifying and characterizing the whole set of genes are already available. Whole-genome analysis by single-gene replacements has been accomplished in *Saccharomyces cerevisiae* by the Yeast Deletion Program (85, 86). Phenotypes generated by each gene deletion were analyzed by assays of growth fitness under different culture conditions and by visual examination of cell shape and size. Some problems became evident after that initiative: the redundancy hiding effects of related genes, the presence of essential genes (about 14% in *S. cerevisiae*), and the lack of detectable phenotype after deletion of some genes. The refinement of the available tools for phenotype analysis becomes then an obvious necessity when we think about global functional analysis of parasite genes.

Tools for a functional study of the *Leishmania* genome through a similar type of approach were developed. Transposable elements have been adapted to *Leishmania* and could be used to generate random disrupted mutants (87-89). Drawbacks of this approach are the diploidy, restricting the observation of a mutant phenotype to the cases where gene dosage is sufficient to produce a change, and the fact that mutants may not display a “visible” phenotype to the methods of observation feasible at the moment.

Any functional approach will have to take into account several variables, and the developmental stage, physiological state, growth conditions, etc. will influence the set of expressed products in a given cell. The study of a response to a particular stimulus is actually a valuable tool to understand the physiology of the cell.

The study of transcriptomes has been applied to several parasites (90-92) and innumerable cell types in complex organisms. It is a powerful approach and can generate a large amount of data.

Several comparative studies of *Leishmania* transcripts have been undertaken using microarrays of random genomic sequences, ESTs, and oligonucleotides. Even before the completion of the *L. major* genome, microarrays were already in use to evaluate changes among the different developmental stages. The first analyses were based on PCR amplification of inserts of promastigote- or amastigote-derived *L. major* cDNA libraries. Transcripts purified from *L. major* promastigotes or lesion-amastigotes tested on that array detected about 15% of the uniquely spotted sequences as up- or downregulated in amastigotes (93).

Another array built with random genomic sequences estimated to cover about 80% of the genome (14, 88) was probed for differential expression during metacyclogenesis. About 15% of the sequences in the array showed changes in abundance by a factor of 1.5 or more (94).

These early estimates of transcript regulation in *Leishmania* were not in agreement with data obtained on microarray analyses of steady-state RNA in *T. brucei* (95, 96), which estimated that 2% of trypanosome genes showed developmental regulation at the mRNA level.

Akopyants *et al.* (97) showed, with the same sequences used in the random-genomic array mentioned above, that a change in abundance of transcripts was observed in only 1-3% of the spotted sequences. These results were validated through Northern blotting, indicating that microarrays could be used for analysis of the *Leishmania* transcriptome. The discrepancy with the previous studies may be attributed, first, to the source of sequences in some arrays. The EST libraries were probably enriched for highly expressed genes, and that may represent a strong bias in the analysis of the microarray. On the other hand, more stringent criteria used for defining the cut-off for differential expression might explain the differences obtained with the genomic derived microarray.

Holzer *et al.* (98) used whole-genome high-density *L. major* oligonucleotide microarrays to analyze RNA isolated from *L. mexicana* promastigotes, lesion-derived amastigotes, and axenic amastigotes. In this study, several new steps were taken: an oligonucleotide array was built based on the whole set of predicted ORFs in the *L. major* genome and assayed against another *Leishmania* species, showing that the degree of sequence conservation in the translated sequences is high enough to allow cross-species analyses to be performed. Furthermore, a comparison of axenic cultured amastigotes and lesion-derived amastigotes was conducted. Conclusions were in agreement with most of the previously reported data: 3.5% of all genes in the *Leishmania* genome showed differentially regulated, steady-state mRNA levels. Most of the regulated transcripts are promastigote enriched. Very few amastigote-upregulated molecules were detected. Interestingly, axenic amastigotes behaved much more like promastigotes than lesion-derived amastigotes, sharing with the first most of the upregulated sequences. Thus, these authors have confirmed that regulatory control in *Leishmania* is exerted mainly downstream of mRNA concentration. These findings were not unexpected given the lack of transcription control. They also indicated that post-transcriptional regulation at the RNA level is not an exceptionally important player in the control of gene expression in this case.

It must be stressed, at this point, that obtaining a large number of lesion-amastigotes for functional studies becomes a very serious problem in some circumstances. For some *Leishmania* species, tissue-amastigotes are very scarce indeed, and it is hard to envisage methods for obtaining enough cells to purify large amounts of RNA or proteins.

Even if differential gene expression based on abundance of transcripts was more common in *Leishmania*, there is often a poor correlation between the transcriptome and levels of translated proteins as shown in other systems (99, 100). It is then understandable why efforts were turned to the study of the final products of the cell machinery.

The term proteome was used initially in 1995 to describe the total protein complement expressed by a genome (101, 102). The basic study of proteomes depends on the use of two main techniques: two-dimensional electrophoretic separation of proteins and mass spectrometry. Two-dimensional electrophoresis (2-DE) of proteins, initially described by O'Farrell (103) has gained several technical refinements recently, making it more reliable and reproducible. However, technical problems still exist, mainly because of poor sensitivity of the method and lack of reproducibility, and to poor hydrophilicity and extreme *pI*s of some proteins.

Mass spectrometry is used for the identification of all spots detected in a 2D-gel or of a selection based on several criteria (abundance, differences between developmental stages, metabolic labeling, etc.). Two major improvements on the technique made it more amenable to protein analysis: matrix-assisted laser desorption and ionization (MALDI) (104, 105) and electrospray ionization (106, 107).

A peptide-mass fingerprint obtained by MALDI-TOF analysis of trypsin-digested proteins can then be compared with full-translated genome data. Furthermore, peptide sequence tags can be obtained from the same peptide mixture by subjecting it to tandem MS.

Proteome analyses of *Leishmania* species have been applied to study the developmental differentiation. Early reports on the *L. infantum* proteome identified over 2,000 spots (of the approximately 8,300 genes predicted in the genome) in 2-DE. The comparison between promastigote and axenic amastigote patterns revealed about 3% of differentially expressed spots (108). Similar results were observed with *L. donovani* (5%) or *L. mexicana* (7%) axenic amastigotes (109, 110). The number of differentially expressed proteins in these studies is very low, and at this point it is not possible to say whether this is attributable to a very stable expression pattern and a low number of differentially expressed proteins or to technical difficulties. Alternatively, as has been shown for the transcription pattern, axenic amastigotes may not represent a reliable tool to evaluate the physiology of lesion amastigotes.



A high resolution 2-DE separation of the proteome of *L. major* promastigotes (with about 3,700 spots) was obtained and used to study mutant lines displaying drug resistance to methotrexate or overexpressing trypanothione reductase (111). This study allowed confirmation of drug resistance mechanisms and showed trypanothione to be post-translationally modified.

Large-scale identification of protein spots observed by 2-DE has not been accomplished for *Leishmania* proteins thus far but, applied to comparative proteomic analysis of *T. cruzi* differentiation, has uncovered interesting information on the energy sources used in different life stages (112).

One interesting application of proteome analyses will come from the study of isolated cellular fractions or organelles. For example, the *T. brucei* flagellum proteome, with 331 components, has been characterized (113). Several proteins identified in this study were subjected to functional ablation analysis through RNAi, bringing further insights into the flagellum physiology, especially in the bloodstream stage of *T. brucei*.

Methods for cell disruption and subcellular fractionation of *T. brucei* have been described (114) and successfully applied to fractionate *Leishmania* promastigotes (115). Direct identification of proteins in organelles may represent a powerful tool to be applied to functional studies.

Current proteome investigations are focused, as exemplified above, on detecting changes in the abundance of certain proteins as a response to a given status or stimulus. Consistent evaluations of quantitative changes in abundance are not easily obtained with classical proteomic techniques. One approach developed to perform fine quantitative analysis of protein components in a complex mixture is the differential protein labeling using reagents such as isotope-coded affinity tags (ICAT) (100, 116). This technique has been already applied to a subset of *T. cruzi* proteins (117) and has been used to analyze differential protein expression in other infectious agents (118, 119).

Progress on methodologies to address large-scale functional proteomics may greatly facilitate our understanding of these parasites. Techniques to study the interaction proteome, transient or stable post-translational modifications, and subcellular trafficking are beginning to appear in large-scale applications for the study of other organisms and will have to be adapted according to the particularities of these parasites.

## Comparative Genomics of Trypanosomatids: Learning from Inter- and Intrageneric Analyses

The recent availability of the Tri-Tryp genomes is a marker in the knowledge about the biology of these three pathogens. Whereas analysis of a single genome provides remarkable biological insights on any particular organism, comparative analysis of multiple genomes offers considerably more information. It expands our ability to understand the genetic and evolutionary bases of the shared and distinct parasitic modes and lifestyles and better assign putative function to predicted coding sequences.

Comparison of gene content and genome architecture, composition, and organization of protein domains encoded by each of the genomes of *L. major*, *T. cruzi*, and *T. brucei* was very informative. In spite of the marked differences regarding their life cycles and pathogenesis, these parasites share about 6,200 genes, which are distributed in large syntenic, polycistronic gene clusters. Comparison of the predicted protein sequences within each of the three genomes revealed that the number of conserved proteins is higher between the two intracellular parasites *L. major* and *T. cruzi* and lower when comparison is made between one of them and *T. brucei*. Twelve percent of the *Leishmania* proteome is composed of species-specific members against 32 and 26% of *T. cruzi* and *T. brucei*, respectively. Because most of the species-specific proteins are annotated as members of surface antigen families, it is possible to relate the differences in absolute number of unique sequences to strategies of survival and immune evasion of each Trypanosomatid (120).

Despite their evolutionary distance, the genomes of these three organisms are highly syntenic. The comparison of *T. brucei* and *L. major* genomes shows that 68 and 75% of their genes, respectively, remain in the same genomic context; 110 blocks of synteny span 30.7 Mb of the *Leishmania* genome. The strand-switch regions separating DGCs are frequently the spot for synteny breakpoints in *Leishmania* and *T. brucei*. Many of the species-specific genes are found in nonsyntenic internal or subtelomeric regions of the chromosomes from each parasite. Furthermore, in these regions also reside retroelements and structural RNAs, and gene family expansions take place. Despite the reported conservation, comparative analyses indicate the presence of gene divergence, sequence acquisition, and loss (less frequent than gene insertion), or rearrangement within syntenic regions of chromosomes (22).

The *L. major* subtelomeric regions are shorter (<20 kb) than the trypanosome ones, with relatively few repetitive sequences. Nevertheless, accumulating evidence indicates that recombination events take place at these sites and may be involved in genetic divergence and gene function gain (19) (L. Brito and A. K. Cruz, unpublished results).

Data accumulated over the years on comparative studies of particular *Leishmania* genes have already provided some clues about what is to be expected from genomic large-scale comparative analysis. For example, the structure and genomic organization of the major surface protease (MSP) genes in several species of *Leishmania* were characterized (reviewed by Yao *et al.* (121)). Several species have multiple copies of the MSP genes organized in tandem. All MSP genes encode similar amino acid sequences. Differences among the different copies in one species are mainly located in the C-terminal coding regions and their 3'UTRs. The comparison between related copies in different species revealed a high conservation of the coding sequence and lesser degrees of similarity in the untranslated regions. The same was observed in other loci, such as the META cluster (122). Taking those studies into account, we might expect to find single nucleotide changes in the coding regions and differences in the untranslated region that might be related to regulation of expression of these sequences.

On the other hand, searching for sequence conservation on non-coding regions of the genomes of three *Leishmania* species may be a route to find elements of functional relevance for gene expression regulation. Comparative analysis of *L. major*, *L. infantum*, and *L. braziliensis* genomes revealed conservation of overall gene content and genome architecture. Large-scale synteny results of these comparative analyses indicate that there are no synteny breaks among the compared genomes. Despite conservation of genomic context, there are differences (about 200 species-specific genes) that most likely reflect specific adaptations to distinct species-specific selection pressures. Furthermore, species-specific genes identified thus far are not clustered (21).

## The Central Role of Bioinformatics to Put Forward Functional Genomics in *Leishmania*

The recent success of the international scientific community in decoding the genetic blueprint of the entire genome of Tri-Tryps (19, 22) has led to the post-genomics era, where there is a need for an intellectual fusion of biomedicine and information technology.

Bioinformatics plays a crucial role in data manipulation, data curation, and knowledge extraction, thus bridging the gap between disparate information sources for subsequent biological model building, refinement, and validation. The post-genome challenge, particularly in the case of these unicellular human pathogens, is to translate new information about genes, their control pathways, proteins, and their interactions into improved healthcare.

In the field of structural annotation, a term that refers to the identification of putative genes in DNA sequence, the discovery of protein-coding genes has been pursued based on the development of several bioinformatics approaches and methodologies (for reviews, see Fickett (123), Burge and Karlin (124), Claverie (125), and Worthey and Myler (126)). Each one of them exhibits performance trade-offs, increasing sensitivity (ability to

detect true positives) and increasing selectivity (ability to exclude false positives), and as a general rule, algorithms for finding protein-coding genes can be classified into two big categories: similarity based (extrinsic methods) and algorithm based (intrinsic methods) (127). The similarity-based category relies on similarity-based algorithms using localized alignments of query sequences against a set of public domain databases, whereas the algorithm-based category, also termed *ab initio* approaches, relies upon the evaluation of compositional properties of exons, introns, and other features without explicit reference to other sequences.

As a general rule and because the use of multiple lines of evidence lead to the increase of sensitivity and selectivity, gene prediction often uses a combination of approaches, including both intrinsic and extrinsic methods. Particularly regarding *L. major*, a combination of CodonUsage (128), GeneScan (129), TestCode (130), and Glimmer (131), together with the manual inspection of the gene calls by curators, has led to a gene identification rate greater than 95%. It's worth mentioning that ~70% of the genes have no significant similarity to existing genes in sequence databases.

On the other hand, the *in silico* identification of non-protein-coding genes is a less advanced discipline, mostly because of the diversity associated with these sequences. Therefore, efforts have been focused on comparative analyses of completed genomes available to date.

The parasitic Trypanosomatids have been studied extensively at the molecular and biochemical levels, through classical biochemistry and molecular biology. Furthermore, several unusual aspects of biochemical physiology are shared between trypanosomatids but are absent from cells in their mammalian hosts. Some examples include the thiol metabolism, mechanisms to maintain a reducing intracellular redox milieu, and folate metabolism. Nevertheless, even with some potentially good drug targets that have been identified, safe, effective, and affordable medicines for the treatment of leishmaniasis are missing (132). Therefore, exploitation of differences between the genomes of the parasite and its vertebrate host is important and may lead to innovative routes for drug target discovery. There is a current initiative of the UNICEF-UNDP-World Bank-WHO Special Programme for Research and Training in Tropical Disease/World Health Organization (<http://www.who.int/tdr/grants/workplans/gdr.htm>) to bring together the pharmaceutical industry with their combinatorial libraries of lead compounds, medicinal chemistry, and genome data-mining to find new drug targets and the corresponding inhibitory molecules acting selectively against the parasites. Although some are optimistic about the development of a single drug useful against all Trypanosomatids, the profound differences in the ways each parasite interacts with the host means that diverse pharmacological criteria need to be met in developing novel drugs. Thus, a single anti-trypanosomatid “cure-all” is unlikely.

Bioinformatics tools, learned and developed as a function of the Trypanosomatids genome project, are now being applied to comparative genomics of *Leishmania*. Understanding why different *Leishmania* species cause such diverse diseases has been an elusive and attractive target for many parasitologists. Comparative genomics may provide new clues to understanding the virulence mechanisms of various species and may turn the present course of controlling and treating leishmaniasis into a major step forward.

## References

1. Benne R, Van den Burg J, et al. Major transcript of the frameshifted *coxII* gene from trypanosome mitochondria contains four nucleotides that are not encoded in the DNA. *Cell*. 1986;46(6):819–826. PubMed PMID: 3019552.
2. Opperdoes FR, Borst P. Localization of nine glycolytic enzymes in a microbody-like organelle in *Trypanosoma brucei*: the glycosome. *FEBS Lett*. 1977;80(2):360–364. PubMed PMID: 142663.
3. Johnson PJ, Kooter JM, et al. Inactivation of transcription by UV irradiation of *T. brucei* provides evidence for a multicistronic transcription unit including a VSG gene. *Cell*. 1987;51(2):273–281. PubMed PMID: 3664637.

4. Boothroyd JC, Cross GA. Transcripts coding for variant surface glycoproteins of *Trypanosoma brucei* have a short, identical exon at their 5' end. *Gene*. 1982;20(2):281–289. PubMed PMID: 7166234.
5. Walder JA, Eder PS.et al. The 35-nucleotide spliced leader sequence is common to all trypanosome messenger RNA's. *Science*. 1986;233(4763):569–571. PubMed PMID: 3523758.
6. Martinez-Calvillo S, Nguyen D.et al. Transcription initiation and termination on *Leishmania major* chromosome 3. *Eukaryot Cell*. 2004;3(2):506–517. PubMed PMID: 15075279.
7. Martinez-Calvillo S, Yan S.et al. Transcription of *Leishmania major* Friedlin chromosome 1 initiates in both directions within a single region. *Mol Cell*. 2003;11(5):1291–1299. PubMed PMID: 12769852.
8. Wincker P, Ravel C.et al. The *Leishmania* genome comprises 36 chromosomes conserved across widely divergent human pathogenic species. *Nucleic Acids Res*. 1996;24(9):1688–1694. PubMed PMID: 8649987.
9. Koonin EV, Galperin MY. Prokaryotic genomes: the emerging paradigm of genome-based microbiology. *Curr Opin Genet Dev*. 1997;7(6):757–763. PubMed PMID: 9468784.
10. Ravel C, Wincker P.et al. Medium-range restriction maps of five chromosomes of *Leishmania infantum* and localization of size-variable regions. *Genomics*. 1996;35(3):509–516. PubMed PMID: 8812485.
11. Britto C, Ravel C.et al. Conserved linkage groups associated with large-scale chromosomal rearrangements between Old World and New World *Leishmania* genomes. *Gene*. 1998;222(1):107–117. PubMed PMID: 9813266.
12. Ivens AC, Lewis SM.et al. A physical map of the *Leishmania major* Friedlin genome. *Genome Res*. 1998;8(2):135–145. PubMed PMID: 9477341.
13. Ivens AC, Blackwell JM. The *Leishmania* genome comes of Age. *Parasitol Today*. 1999;15(6):225–231. PubMed PMID: 10366828.
14. Akopyants NS, Clifton SW.et al. A survey of the *Leishmania major* Friedlin strain V1 genome by shotgun sequencing: a resource for DNA microarrays and expression profiling. *Mol Biochem Parasitol*. 2001;113(2):337–340. PubMed PMID: 11295190.
15. El-Sayed NM, Donelson JE. A survey of the *Trypanosoma brucei* rhodesiense genome using shotgun sequencing. *Mol Biochem Parasitol*. 1997;84(2):167–178. PubMed PMID: 9084037.
16. Myler PJ, Audleman L.et al. *Leishmania major* Friedlin chromosome 1 has an unusual distribution of protein-coding genes. *Proc Natl Acad Sci U S A*. 1999;96(6):2902–2906. PubMed PMID: 10077609.
17. Aly R, Argaman M.et al. A regulatory role for the 5' and 3' untranslated regions in differential expression of hsp83 in *Leishmania*. *Nucleic Acids Res*. 1994;22(15):2922–2929. PubMed PMID: 8065903.
18. LeBowitz JH, Smith HQ.et al. Coupling of poly(A) site selection and trans-splicing in *Leishmania*. *Genes Dev*. 1993;7(6):996–1007. PubMed PMID: 8504937.
19. Ivens AC, Peacock CS.et al. The genome of the kinetoplastid parasite, *Leishmania major*. *Science*. 2005;309(5733):436–442. PubMed PMID: 16020728.
20. *C. elegans* Sequencing Consortium Genome sequence of the nematode *C. elegans*: a platform for investigating biology. The *C. elegans* Sequencing Consortium. *Science*. 1998;282(5396):2012–2018. PubMed PMID: 9851916.
21. Peacock CS, Seeger K.et al. Comparative genomic analysis of three *Leishmania* species that cause diverse human disease". *Nat Genet*. 2007;39(7):839–847. PubMed PMID: 17572675.
22. El-Sayed NM, Myler PJ.et al. The genome sequence of *Trypanosoma cruzi*, etiologic agent of Chagas disease. *Science*. 2005;309(5733):409–415. PubMed PMID: 16020725.
23. Bringaud F, Ghedin E.et al. Evolution of non-LTR retrotransposons in the trypanosomatid genomes: *Leishmania major* has lost the active elements. *Mol Biochem Parasitol*. 2006;145(2):158–170. PubMed PMID: 16257065.
24. Wickstead B, Ersfeld K.et al. Repetitive elements in genomes of parasitic protozoa. *Microbiol Mol Biol Rev*. 2003;67(3):360–375. PubMed PMID: 12966140.
25. Sunkin SM, Kiser P.et al. The size difference between *Leishmania major* friedlin chromosome one homologues is localized to sub-telomeric repeats at one chromosomal end. *Mol Biochem Parasitol*. 2000;109(1):1–15. PubMed PMID: 10924752.

26. Fu G, Barker DC. Characterisation of Leishmania telomeres reveals unusual telomeric repeats and conserved telomere-associated sequence. *Nucleic Acids Res.* 1998;26(9):2161–2167. PubMed PMID: 9547275.
27. Pedrosa AL, Ruiz JC et al. Characterisation of three chromosomal ends of Leishmania major reveals transcriptional activity across arrays of reiterated and unique sequences. *Mol Biochem Parasitol.* 2001;114(1):71–80. PubMed PMID: 11356515.
28. Tosato V, Ciarloni L et al. Secondary DNA structure analysis of the coding strand switch regions of five Leishmania major Friedlin chromosomes. *Curr Genet.* 2001;40(3):186–194. PubMed PMID: 11727994.
29. Curotto de Lafaille MA, Laban A et al. Gene expression in Leishmania: analysis of essential 5' DNA sequences. *Proc Natl Acad Sci U S A.* 1992;89(7):2703–2707. PubMed PMID: 1557376.
30. Monnerat S, Martinez-Calvillo S et al. Genomic organization and gene expression in a chromosomal region of Leishmania major. *Mol Biochem Parasitol.* 2004;134(2):233–243. PubMed PMID: 15003843.
31. Gilinger G, Bellofatto V. Trypanosome spliced leader RNA genes contain the first identified RNA polymerase II gene promoter in these organisms. *Nucleic Acids Res.* 2001;29(7):1556–1564. PubMed PMID: 11266558.
32. Luo H, Gilinger G et al. Transcription initiation at the TATA-less spliced leader RNA gene promoter requires at least two DNA-binding proteins and a tripartite architecture that includes an initiator element. *J Biol Chem.* 1999;274(45):31947–31954. PubMed PMID: 10542223.
33. Thomas S, Westenberger SJ et al. Intragenomic spliced leader RNA array analysis of kinetoplasts reveals unexpected transcribed region diversity in Trypanosoma cruzi. *Gene.* 2005;352:100–108. PubMed PMID: 15925459.
34. Gay LS, Wilson ME et al. The promoter for the ribosomal RNA genes of Leishmania chagasi. *Mol Biochem Parasitol.* 1996;77(2):193–200. PubMed PMID: 8813665.
35. Martinez-Calvillo S, Sunkin SM et al. Genomic organization and functional characterization of the Leishmania major Friedlin ribosomal RNA gene locus. *Mol Biochem Parasitol.* 2001;116(2):147–157. PubMed PMID: 11522348.
36. Uliana SR, Fischer W et al. Structural and functional characterization of the Leishmania amazonensis ribosomal RNA promoter. *Mol Biochem Parasitol.* 1996;76(1-2):245–255. PubMed PMID: 8920010.
37. Yan S, Lodes MJ et al. Characterization of the Leishmania donovani ribosomal RNA promoter. *Mol Biochem Parasitol.* 1999;103(2):197–210. PubMed PMID: 10551363.
38. Mair G, Shi H et al. A new twist in trypanosome RNA metabolism: cis-splicing of pre-mRNA. *RNA.* 2000;6(2):163–169. PubMed PMID: 10688355.
39. Sacks DL, Perkins PV. Identification of an infective stage of Leishmania promastigotes. *Science.* 1984;223(4643):1417–1419. PubMed PMID: 6701528.
40. Bellatin JA, Murray AS et al. Leishmania mexicana: identification of genes that are preferentially expressed in amastigotes. *Exp Parasitol.* 2002;100(1):44–53. PubMed PMID: 11971653.
41. Boucher N, Wu Y et al. A common mechanism of stage-regulated gene expression in Leishmania mediated by a conserved 3'-untranslated region element. *J Biol Chem.* 2002;277(22):19511–19520. PubMed PMID: 11912202.
42. Brooks DR, Denise H et al. The stage-regulated expression of Leishmania mexicana CPB cysteine proteases is mediated by an intercistronic sequence element. *J Biol Chem.* 2001;276(50):47061–47069. PubMed PMID: 11592967.
43. Charest H, Zhang WW et al. The developmental expression of Leishmania donovani A2 amastigote-specific genes is post-transcriptionally mediated and involves elements located in the 3'-untranslated region. *J Biol Chem.* 1996;271(29):17081–17090. PubMed PMID: 8663340.
44. Flinn HM, Smith DF. Genomic organisation and expression of a differentially-regulated gene family from Leishmania major. *Nucleic Acids Res.* 1992;20(4):755–762. PubMed PMID: 1371863.
45. Kelly BL, Nelson TN et al. Stage-specific expression in Leishmania conferred by 3' untranslated regions of L. major leishmanolysin genes (GP63). *Mol Biochem Parasitol.* 2001;116(1):101–104. PubMed PMID: 11463473.

46. Moore LL, Santrich C.et al. Stage-specific expression of the *Leishmania mexicana* paraflagellar rod protein PFR-2. *Mol Biochem Parasitol.* 1996;80(2):125–135. PubMed PMID: 8892290.
47. Rochette A, McNicoll F.et al. Characterization and developmental gene regulation of a large gene family encoding amastin surface proteins in *Leishmania* spp. *Mol Biochem Parasitol.* 2005;140(2):205–220. PubMed PMID: 15760660.
48. Wang Y, Dimitrov K.et al. Stage-specific activity of the *Leishmania major* CRK3 kinase and functional rescue of a *Schizosaccharomyces pombe* *cdc2* mutant. *Mol Biochem Parasitol.* 1998;96(1-2):139–150. PubMed PMID: 9851613.
49. Cruz AK, Titus R.et al. Plasticity in chromosome number and testing of essential genes in *Leishmania* by targeting. *Proc Natl Acad Sci U S A.* 1993;90(4):1599–1603. PubMed PMID: 8381972.
50. Martinez-Calvillo S, Stuart K.et al. Ploidy changes associated with disruption of two adjacent genes on *Leishmania major* chromosome 1. *Int J Parasitol.* 2005;35(4):419–429. PubMed PMID: 15777918.
51. Mottram JC, Coombs GH. *Leishmania mexicana*: subcellular distribution of enzymes in amastigotes and promastigotes. *Exp Parasitol.* 1985;59(3):265–274. PubMed PMID: 3158538.
52. Uliana SR, Goyal N.et al. *Leishmania*: overexpression and comparative structural analysis of the stage-regulated meta 1 gene. *Exp Parasitol.* 1999;92(3):183–191. PubMed PMID: 10403759.
53. Matthews KR, Tschudi C.et al. A common pyrimidine-rich motif governs trans-splicing and polyadenylation of tubulin polycistronic pre-mRNA in trypanosomes. *Genes Dev.* 1994;8(4):491–501. PubMed PMID: 7907303.
54. Ullu E, Matthews KR.et al. Temporal order of RNA-processing reactions in trypanosomes: rapid trans splicing precedes polyadenylation of newly synthesized tubulin transcripts. *Mol Cell Biol.* 1993;13(1):720–725. PubMed PMID: 8417363.
55. Manning-Cela R, Gonzalez A.et al. Alternative splicing of LYT1 transcripts in *Trypanosoma cruzi*. *Infect Immun.* 2002;70(8):4726–4728. PubMed PMID: 12117992.
56. Vassella E, Braun R.et al. Control of polyadenylation and alternative splicing of transcripts from adjacent genes in a procyclin expression site: a dual role for polypyrimidine tracts in trypanosomes? *Nucleic Acids Res.* 1994;22(8):1359–1364. PubMed PMID: 8190625.
57. Beetham JK, Myung KS.et al. Glycoprotein 46 mRNA abundance is post-transcriptionally regulated during development of *Leishmania chagasi* promastigotes to an infectious form. *J Biol Chem.* 1997;272(28):17360–17366. PubMed PMID: 9211875.
58. Brittingham A, Miller MA.et al. Regulation of GP63 mRNA stability in promastigotes of virulent and attenuated *Leishmania chagasi*. *Mol Biochem Parasitol.* 2001;112(1):51–59. PubMed PMID: 11166386.
59. Quijada L, Soto M.et al. Identification of a putative regulatory element in the 3'-untranslated region that controls expression of HSP70 in *Leishmania infantum*. *Mol Biochem Parasitol.* 2000;110(1):79–91. PubMed PMID: 10989147.
60. Wilson ME, Paetz KE.et al. The effect of ongoing protein synthesis on the steady state levels of Gp63 RNAs in *Leishmania chagasi*. *J Biol Chem.* 1993;268(21):15731–15736. PubMed PMID: 8340397.
61. Folgueira C, Quijada L.et al. The translational efficiencies of the two *Leishmania infantum* HSP70 mRNAs, differing in their 3'-untranslated regions, are affected by shifts in the temperature of growth through different mechanisms. *J Biol Chem.* 2005;280(42):35172–35183. PubMed PMID: 16105831.
62. Larreta R, Soto M.et al. The expression of HSP83 genes in *Leishmania infantum* is affected by temperature and by stage-differentiation and is regulated at the levels of mRNA stability and translation. *BMC Mol Biol.* 2004;5:3. PubMed PMID: 15176985.
63. McNicoll F, Muller M.et al. Distinct 3'-untranslated region elements regulate stage-specific mRNA accumulation and translation in *Leishmania*. *J Biol Chem.* 2005;280(42):35238–35246. PubMed PMID: 16115874.
64. Blum B, Bakalara N.et al. A model for RNA editing in kinetoplastid mitochondria: "guide" RNA molecules transcribed from maxicircle DNA provide the edited information. *Cell.* 1990;60(2):189–198. PubMed PMID: 1688737.

65. Sturm NR, Simpson L. Kinetoplast DNA minicircles encode guide RNAs for editing of cytochrome oxidase subunit III mRNA. *Cell*. 1990;61(5):879–884. PubMed PMID: 1693097.
66. Bellofatto V, Cross GA. Expression of a bacterial gene in a trypanosomatid protozoan. *Science*. 1989;244(4909):1167–1169. PubMed PMID: 2499047.
67. Kapler GM, Coburn CM, et al. Stable transfection of the human parasite *Leishmania major* delineates a 30-kilobase region sufficient for extrachromosomal replication and expression. *Mol Cell Biol*. 1990;10(3):1084–1094. PubMed PMID: 2304458.
68. Laban A, Tobin JF, et al. Stable expression of the bacterial *neor* gene in *Leishmania enriettii*. *Nature*. 1990;343(6258):572–574. PubMed PMID: 2300209.
69. Laban A, Wirth DF. Transfection of *Leishmania enriettii* and expression of chloramphenicol acetyltransferase gene. *Proc Natl Acad Sci U S A*. 1989;86(23):9119–9123. PubMed PMID: 2594753.
70. LeBowitz JH, Coburn CM, et al. Development of a stable *Leishmania* expression vector and application to the study of parasite surface antigen genes. *Proc Natl Acad Sci U S A*. 1990;87(24):9736–9740. PubMed PMID: 2124701.
71. Clayton CE. Genetic manipulation of kinetoplastida. *Parasitol Today*. 1999;15(9):372–378. PubMed PMID: 10461166.
72. Clayton CE. Life without transcriptional control? From fly to man and back again. *EMBO J*. 2002;21(8):1881–1888. PubMed PMID: 11953307.
73. Ghedin E, Charest H, et al. Inducible expression of suicide genes in *Leishmania donovani* amastigotes. *J Biol Chem*. 1998;273(36):22997–23003. PubMed PMID: 9722523.
74. Ryan KA, Dasgupta S, et al. Shuttle cosmid vectors for the trypanosomatid parasite *Leishmania*. *Gene*. 1993;131(1):145–150. PubMed PMID: 8370535.
75. Ryan KA, Garraway LA, et al. Isolation of virulence genes directing surface glycosyl-phosphatidylinositol synthesis by functional complementation of *Leishmania*. *Proc Natl Acad Sci U S A*. 1993;90(18):8609–8613. PubMed PMID: 8378337.
76. Pedrosa AL, Cruz AK. The effect of location and direction of an episomal gene on the restoration of a phenotype by functional complementation in *Leishmania*. *Mol Biochem Parasitol*. 2002;122(2):141–148. PubMed PMID: 12106868.
77. Cruz A, Beverley SM. Gene replacement in parasitic protozoa. *Nature*. 1990;348(6297):171–173. PubMed PMID: 2234081.
78. Misslitz A, Mottram JC, et al. Targeted integration into a rRNA locus results in uniform and high level expression of transgenes in *Leishmania* amastigotes. *Mol Biochem Parasitol*. 2000;107(2):251–261. PubMed PMID: 10779601.
79. Papadopoulou B, Dumas C. Parameters controlling the rate of gene targeting frequency in the protozoan parasite *Leishmania*. *Nucleic Acids Res*. 1997;25(21):4278–4286. PubMed PMID: 9336458.
80. Cruz A, Coburn CM, et al. Double targeted gene replacement for creating null mutants. *Proc Natl Acad Sci U S A*. 1991;88(16):7170–7174. PubMed PMID: 1651496.
81. Curotto de Lafaille MA, Wirth DF. Creation of Null/+ mutants of the alpha-tubulin gene in *Leishmania enriettii* by gene cluster deletion. *J Biol Chem*. 1992;267(33):23839–23846. PubMed PMID: 1429722.
82. McKean PG, Denny PW, et al. Phenotypic changes associated with deletion and overexpression of a stage-regulated gene family in *Leishmania*. *Cell Microbiol*. 2001;3(8):511–523. PubMed PMID: 11488813.
83. Wirtz E, Clayton C. Inducible gene expression in trypanosomes mediated by a prokaryotic repressor. *Science*. 1995;268(5214):1179–1183. PubMed PMID: 7761835.
84. Yan S, Martinez-Calvillo S, et al. A low-background inducible promoter system in *Leishmania donovani*. *Mol Biochem Parasitol*. 2002;119(2):217–223. PubMed PMID: 11814573.
85. Giaever G, Chu AM, et al. Functional profiling of the *Saccharomyces cerevisiae* genome. *Nature*. 2002;418(6896):387–391. PubMed PMID: 12140549.
86. Winzeler EA, Shoemaker DD, et al. Functional characterization of the *S. cerevisiae* genome by gene deletion and parallel analysis. *Science*. 1999;285(5429):901–906. PubMed PMID: 10436161.

87. Augusto MJ, Squina FM.et al. Specificity of modified *Drosophila mariner* transposons in the identification of *Leishmania* genes. *Exp Parasitol*. 2004;108(3-4):109–113. PubMed PMID: 15582507.
88. Beverley SM, Akopyants NS.et al. Putting the *Leishmania* genome to work: functional genomics by transposon trapping and expression profiling. *Philos Trans R Soc Lond B Biol Sci*. 2002;357(1417):47–53. PubMed PMID: 11839181.
89. Robinson KA, Beverley SM. Improvements in transfection efficiency and tests of RNA interference (RNAi) approaches in the protozoan parasite *Leishmania*. *Mol Biochem Parasitol*. 2003;128(2):217–228. PubMed PMID: 12742588.
90. Fitzpatrick JM, Johnston DA.et al. An oligonucleotide microarray for transcriptome analysis of *Schistosoma mansoni* and its application/use to investigate gender-associated gene expression. *Mol Biochem Parasitol*. 2005;141(1):1–13. PubMed PMID: 15811522.
91. Llinas M, Bozdech Z.et al. Comparative whole genome transcriptome analysis of three *Plasmodium falciparum* strains. *Nucleic Acids Res*. 2006;34(4):1166–1173. PubMed PMID: 16493140.
92. Radke JR, Behnke MS.et al. The transcriptome of *Toxoplasma gondii*. *BMC Biol*. 2005;3:26. PubMed PMID: 16324218.
93. Almeida R, Gilmartin BJ.et al. Expression profiling of the *Leishmania* life cycle: cDNA arrays identify developmentally regulated genes present but not annotated in the genome. *Mol Biochem Parasitol*. 2004;136(1):87–100. PubMed PMID: 15138070.
94. Saxena A, Worthey EA.et al. Evaluation of differential gene expression in *Leishmania major* Friedlin procyclics and metacyclics using DNA microarray analysis. *Mol Biochem Parasitol*. 2003;129(1):103–114. PubMed PMID: 12798511.
95. Brems S, Guilbride DL.et al. The transcriptomes of *Trypanosoma brucei* Lister 427 and TREU927 bloodstream and procyclic trypomastigotes. *Mol Biochem Parasitol*. 2005;139(2):163–172. PubMed PMID: 15664651.
96. Diehl S, Diehl F.et al. Analysis of stage-specific gene expression in the bloodstream and the procyclic form of *Trypanosoma brucei* using a genomic DNA-microarray. *Mol Biochem Parasitol*. 2002;123(2):115–123. PubMed PMID: 12270627.
97. Akopyants NS, Matlib RS.et al. Expression profiling using random genomic DNA microarrays identifies differentially expressed genes associated with three major developmental stages of the protozoan parasite *Leishmania major*. *Mol Biochem Parasitol*. 2004;136(1):71–86. PubMed PMID: 15138069.
98. Holzer TR, McMaster WR.et al. Expression profiling by whole-genome interspecies microarray hybridization reveals differential gene expression in procyclic promastigotes, lesion-derived amastigotes, and axenic amastigotes in *Leishmania mexicana*. *Mol Biochem Parasitol*. 2006;146(2):198–218. PubMed PMID: 16430978.
99. Anderson L, Seilhamer J. A comparison of selected mRNA and protein abundances in human liver. *Electrophoresis*. 1997;18(3-4):533–537. PubMed PMID: 9150937.
100. Gygi SP, Aebersold R. Absolute quantitation of 2-D protein spots. *Methods Mol Biol*. 1999;112:417–421. PubMed PMID: 10027266.
101. Wasinger VC, Cordwell SJ.et al. Progress with gene-product mapping of the Mollicutes: *Mycoplasma genitalium*. *Electrophoresis*. 1995;16(7):1090–1094. PubMed PMID: 7498152.
102. Wilkins MR, Sanchez JC.et al. Progress with proteome projects: why all proteins expressed by a genome should be identified and how to do it. *Biotechnol Genet Eng Rev*. 1996;13:19–50. PubMed PMID: 8948108.
103. O'Farrell PH. High resolution two-dimensional electrophoresis of proteins. *J Biol Chem*. 1975;250(10):4007–4021. PubMed PMID: 236308.
104. Hillenkamp F, Karas M.et al. Matrix-assisted laser desorption/ionization mass spectrometry of biopolymers. *Anal Chem*. 1991;63(24):1193A–1203A. PubMed PMID: 1789447.
105. Karas M, Hillenkamp F. Laser desorption ionization of proteins with molecular masses exceeding 10,000 daltons. *Anal Chem*. 1988;60(20):2299–2301. PubMed PMID: 3239801.
106. Fenn JB, Mann M.et al. Electrospray ionization for mass spectrometry of large biomolecules. *Science*. 1989;246(4926):64–71. PubMed PMID: 2675315.



107. Smith RD, Loo JA.et al. New developments in biochemical mass spectrometry: electrospray ionization. *Anal Chem.* 1990;62(9):882–899. PubMed PMID: 2194402.
108. El Fakhry Y, Ouellette M.et al. A proteomic approach to identify developmentally regulated proteins in *Leishmania infantum*. *Proteomics.* 2002;2(8):1007–1017. PubMed PMID: 12203896.
109. Bente M, Harder S.et al. Developmentally induced changes of the proteome in the protozoan parasite *Leishmania donovani*. *Proteomics.* 2003;3(9):1811–1829. PubMed PMID: 12973740.
110. Nugent PG, Karsani SA.et al. Proteomic analysis of *Leishmania mexicana* differentiation. *Mol Biochem Parasitol.* 2004;136(1):51–62. PubMed PMID: 15138067.
111. Drummelsmith J, Brochu V.et al. Proteome mapping of the protozoan parasite *Leishmania* and application to the study of drug targets and resistance mechanisms. *Mol Cell Proteomics.* 2003;2(3):146–155. PubMed PMID: 12644573.
112. Atwood JA, Weatherly DB.et al. The *Trypanosoma cruzi* proteome. *Science.* 2005;309(5733):473–476. PubMed PMID: 16020736.
113. Broadhead R, Dawe HR.et al. Flagellar motility is required for the viability of the bloodstream trypanosome. *Nature.* 2006;440(7081):224–227. PubMed PMID: 16525475.
114. Fairlamb AH, Bowman IB. Cell disruption and subcellular fractionation of *Trypanosoma brucei*. *Trans R Soc Trop Med Hyg.* 1974;68(4):275. PubMed PMID: 4417067.
115. Rodrigues CO, Scott DA.et al. Presence of a vacuolar H<sup>+</sup>-pyrophosphatase in promastigotes of *Leishmania donovani* and its localization to a different compartment from the vacuolar H<sup>+</sup>-ATPase. *Biochem J.* 1999;340(Pt 3):759–766. PubMed PMID: 10359662.
116. Gygi SP, Rist B.et al. Quantitative analysis of complex protein mixtures using isotope-coded affinity tags. *Nat Biotechnol.* 1999;17(10):994–999. PubMed PMID: 10504701.
117. Paba J, Ricart CA.et al. Proteomic analysis of *Trypanosoma cruzi* developmental stages using isotope-coded affinity tag reagents. *J Proteome Res.* 2004;3(3):517–524. PubMed PMID: 15253433.
118. Guina T, Wu M.et al. Proteomic analysis of *Pseudomonas aeruginosa* grown under magnesium limitation. *J Am Soc Mass Spectrom.* 2003;14(7):742–751. PubMed PMID: 12837596.
119. Schmidt F, Donahoe S.et al. Complementary analysis of the *Mycobacterium tuberculosis* proteome by two-dimensional electrophoresis and isotope-coded affinity tag technology. *Mol Cell Proteomics.* 2004;3(1):24–42. PubMed PMID: 14557599.
120. El-Sayed NM, Myler PJ.et al. Comparative genomics of trypanosomatid parasitic protozoa. *Science.* 2005;309(5733):404–409. PubMed PMID: 16020724.
121. Yao C, Donelson JE.et al. The major surface protease (MSP or GP63) of *Leishmania* sp. Biosynthesis, regulation of expression, and function. *Mol Biochem Parasitol.* 2003;132(1):1–16. PubMed PMID: 14563532.
122. Ramos CS, Franco FA.et al. Characterisation of a new *Leishmania* META gene and genomic analysis of the META cluster. *FEMS Microbiol Lett.* 2004;238(1):213–219. PubMed PMID: 15336424.
123. Fickett JW. Finding genes by computer: the state of the art. *Trends Genet.* 1996;12(8):316–320. PubMed PMID: 8783942.
124. Burge C, Karlin S. Prediction of complete gene structures in human genomic DNA. *J Mol Biol.* 1997;268(1):78–94. PubMed PMID: 9149143.
125. Claverie JM. Computational methods for the identification of genes in vertebrate genomic sequences. *Hum Mol Genet.* 1997;6(10):1735–1744. PubMed PMID: 9300666.
126. Worthey EA, Myler PJ. Protozoan genomes: gene identification and annotation. *Int J Parasitol.* 2005;35(5):495–512. PubMed PMID: 15826642.
127. Stein LD. Using Perl to facilitate biological analysis. *Methods Biochem Anal.* 2001;43:413–449. PubMed PMID: 11449734.
128. Bibb MJ, Findlay PR.et al. The relationship between base composition and codon usage in bacterial genes and its use for the simple and reliable identification of protein-coding sequences. *Gene.* 1984;30(1-3):157–166. PubMed PMID: 6096212.

129. Tiwari S, Ramachandran S.et al. Prediction of probable genes by Fourier analysis of genomic sequences. *Comput Appl Biosci.* 1997;13(3):263–270. PubMed PMID: 9183531.
130. Fickett JW. Recognition of protein coding regions in DNA sequences. *Nucleic Acids Res.* 1982;10(17): 5303–5318. PubMed PMID: 7145702.
131. Delcher AL, Harmon D.et al. Improved microbial gene identification with GLIMMER. *Nucleic Acids Res.* 1999;27(23):4636–4641. PubMed PMID: 10556321.
132. Trouiller P, Olliaro P.et al. Drug development for neglected diseases: a deficient market and a public-health policy failure. *Lancet.* 2002;359(9324):2188–2194. PubMed PMID: 12090998.

## Chapter B03. Tuberculosis, Leprosy, and Other Mycobacterioses

Sylvia Cardoso Leão, MD, PhD,<sup>1</sup> Maria Isabel Romano, PhD,<sup>2</sup> and Maria Jesus Garcia, MD, ScB, PhD<sup>3</sup>

Created: January 30, 2006; Updated: September 24, 2007.

This chapter refers to diseases that represent major public health problems, such as tuberculosis, leprosy, and Buruli ulcer, and emerging diseases, caused by other mycobacterial species. A brief account of bacteria responsible for each disease and the respective global situation is followed by a description of finished and ongoing genome projects and the impact of genome projects on mycobacteria research. *Mycobacterium* is one of the most sequenced bacterial genera. This wealth of comparative genome sequence information provides unique opportunities for new insights into the biology of these globally important pathogens to address the scientific imperatives of better drugs, vaccines, and diagnostics for mycobacterial diseases.

### The Global Burden of Tuberculosis, Leprosy, and Other Mycobacterioses

#### Diseases Caused by *Mycobacterium tuberculosis* Complex

*Mycobacterium tuberculosis* complex (MTBC) includes closely related bacteria showing higher than 99% similarity at the DNA level. The group comprises several main members, such as *M. tuberculosis*, responsible for most human tuberculosis cases, *Mycobacterium bovis*, the agent of bovine tuberculosis, which can also infect other animals as well as humans, *Mycobacterium africanum*, a prevalent cause of human tuberculosis on the African continent, and the vole bacillus *Mycobacterium microti* (1) ([Genus Mycobacterium](#)). It also contains the vaccine strain Bacille Calmette-Guérin (BCG). Other recently proposed members of the MTBC are *Mycobacterium caprae*, primarily infecting goats in Spain (2, 3), and *Mycobacterium pinnipedii*, infecting seals in Australia and Argentina (4). A variant human isolate of *M. tuberculosis*, the *canetti* strain, was also described recently and has been associated with the “ancient” tuberculosis lineages (5). Despite this close relationship, they show a large variability in their phenotypic properties, epidemiology, and incidence in human tuberculosis, and for this and other historical reasons, they are still considered different species (6). However, the extent of MTBC speciation is not yet resolved (7).

According to WHO reports, one-third of the world’s population is currently infected with the tuberculosis bacillus, and each year, about two million people die of tuberculosis ([WHO Tuberculosis](#)). Lack of control of tuberculosis is largely attributable to delayed or under diagnosis, non-efficient protection by BCG vaccination, the use of multiple drugs and prolonged treatment regimens, which leads to the existence and spread of multidrug-resistant strains, and association with HIV infection. Other social factors also have strong influence, such as increase in migration and poverty. The evolution of human tuberculosis incidence since 1990 is depicted in the maps in Figure 1 and Figure 2.

Bovine tuberculosis is a chronic zoonotic disease whose etiological agent is *M. bovis*. It constitutes a serious animal health problem, causing economic losses due to decreased meat and milk production and to low exportation of cattle products ([TB in cattle](#)). Although the main host of *M. bovis* is cattle, other animals, including humans, may be affected. A study performed in the main milk production region of Argentina showed

---

**Author Affiliations:** 1 Departamento de Microbiologia, Imunologia e Parasitologia, Universidade Federal de São Paulo, Brazil; Email: sylvia.leao@unifesp.br. 2 Instituto de Biotecnología, Centro de Investigación en Ciencias Veterinarias del CICVyA (CICV), Instituto Nacional de Tecnología Agropecuaria (INTA), Buenos Aires, Argentina; Email: mromano@cnia.inta.gov.ar. 3 Departamento de Medicina Preventiva y Salud Pública y Microbiología, Facultad de Medicina, Universidad Autónoma de Madrid (UAM), Madrid, Spain; Email: mariaj.garcia@uam.es.

that in the period 1984 to 1989, *M. bovis* was responsible for 2.4 to 6.2% of tuberculosis cases in human beings, of which 64% were rural and meat workers (8). The AIDS epidemic has also increased the risk of transmission of *M. bovis* to humans. Nosocomial transmission of bovine tuberculosis produced by multidrug-resistant *M. bovis* strains among HIV-positive individuals was described recently in Spain (9).

## Leprosy: A Public Health Problem

Leprosy remains a public health problem, mainly in developing countries, because in spite of the actions promoted by WHO, prevalence has been reduced, but the number of new cases has remained constant, with more than 690,000 cases reported (WHO; [New Case Detection](#)). The discovery of the leprosy bacillus by Hansen in 1873 constituted the first demonstration of a clear association of a microorganism to human disease. *Mycobacterium leprae*, the etiological agent of leprosy, is still unable to be grown as axenic culture and has an extremely slow doubling time in tissues (approximately 14 days). Large quantities of bacilli could be isolated for biochemical and genetic studies using the nine-banded armadillo as a surrogate host. The complete genome sequence of the armadillo-derived Indian isolate, the TN strain, was performed (10).

Leprosy is considered a low transmissible disease, and the means of transmission are uncertain. However, *M. leprae* infection is thought to spread by the respiratory route, because bacilli could be isolated from nasal swabs of patients. Leprosy develops very slowly in the infected host (2 to 10 years); there are two polar, stable forms based on skin smears (named multibacillary (MB) and paucibacillary (PB) leprosy, respectively) and a complex spectrum of unstable forms ([Classification](#)).

Using comparative genomics, seven strains of the leprosy bacilli from separate geographic origins were recently analyzed, showing a surprisingly stable genome. Authors concluded that cases of leprosy around the world could be attributable to a single clone, providing a general evolutionary scheme for *M. leprae* on the basis of single nucleotide polymorphism (SNP) data (11) (Figure 3). More recently, the computer analysis of substitutions in the pseudogenes, and its comparison with the functional orthologs of closely related genomes, has allowed the reconstruction of the gene content of the common ancestor of *M. leprae* and mycobacteria (12).

## Other Mycobacterioses: Neglected Diseases

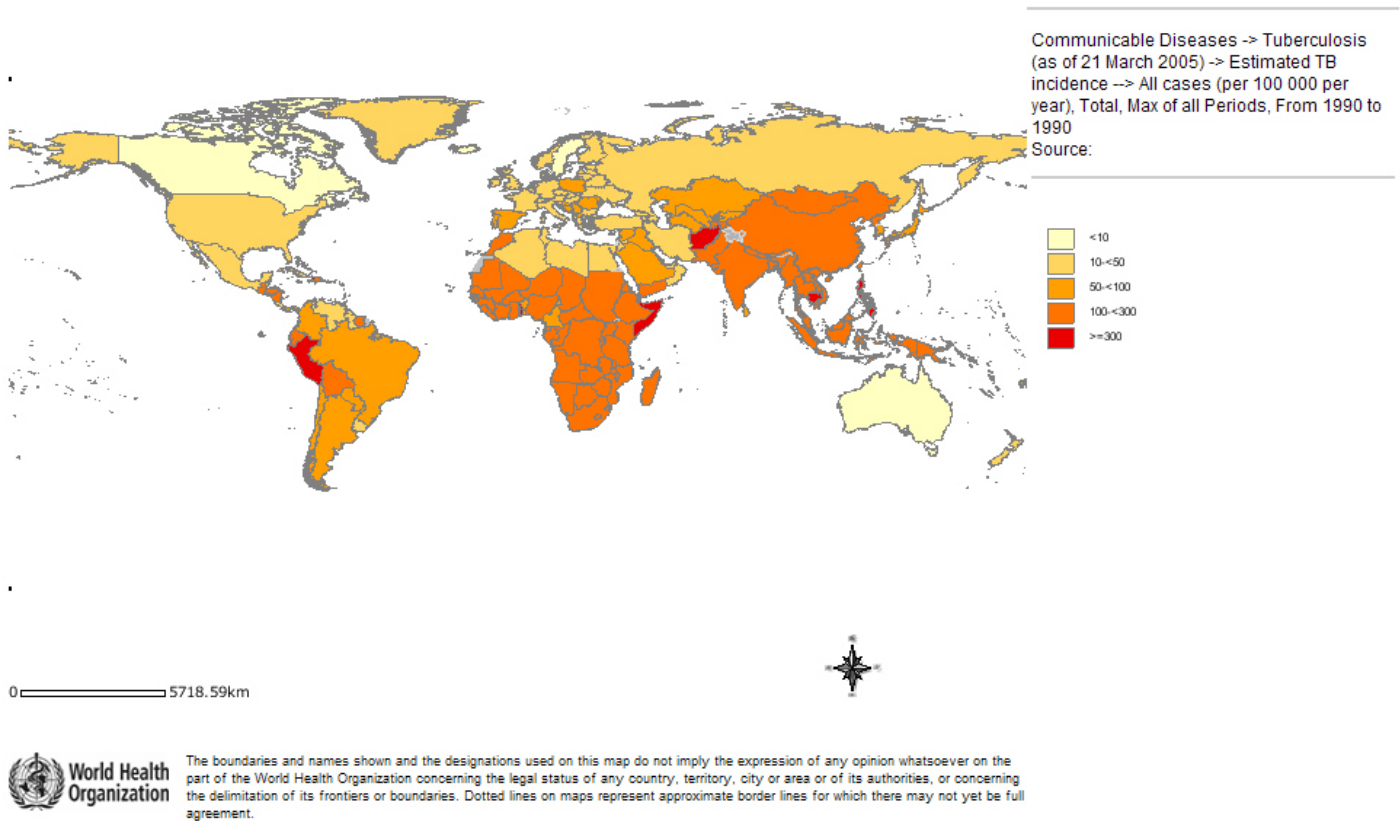
### Buruli Ulcer

*Mycobacterium ulcerans* is an emerging human pathogen responsible for Buruli ulcer, a necrotizing skin disease found most commonly in West Africa (13), but outbreaks have also been reported in the Americas, Australia, and Asia ([WHO Buruli Ulcer](#)). The first detailed clinical description of ulcers caused by *M. ulcerans* has been attributed to Dr. Albert Cook, a missionary physician who worked in Uganda in the late 1800s ([Buruli ulcer](#)). Since the 1980s, Buruli ulcer has emerged as a serious public health problem in an increasing number of countries. In terms of numbers of cases, it is probably the third most common mycobacterial disease in humans, after tuberculosis and leprosy (Figure 4).

Knowledge of *M. ulcerans* at the molecular level has come only in the last 5 years, primarily from research focused on developing tools for either rapid diagnosis or molecular epidemiological investigation. *M. ulcerans* has an unexpectedly close genetic relationship with *Mycobacterium marinum*. Analysis of the 16S rRNA gene of both species revealed greater than 99.8% sequence identity. Biochemically, *M. ulcerans* has also been found to resemble *M. marinum*. However, there are also substantial phenotypic differences and dramatic differences in the pathologies and treatment of the diseases caused by each of these organisms. Also at a genetic level, multi-copy insertion sequences, IS2404 and IS2606, are present in the genome of *M. ulcerans* and absent from *M. marinum* (14).

This close relationship has been exploited in comparative genetic analysis by multi-locus sequencing (15). The results of this analysis suggest that *M. ulcerans* has recently diverged from *M. marinum* by the acquisition and

Communicable Diseases -> Tuberculosis (as of 21 March 2005) -> Estimated TB incidence --> All cases (per 100 000 per year), Total, Max of all Periods, From 1990 to 1990



**Figure 1.** TB incidence, all forms (per 100,000 population per year), By Country, Total, 1990. From: WHO|Global TB database ([http://www.who.int/tb/country/global\\_tb\\_database/en/index1.html](http://www.who.int/tb/country/global_tb_database/en/index1.html)).

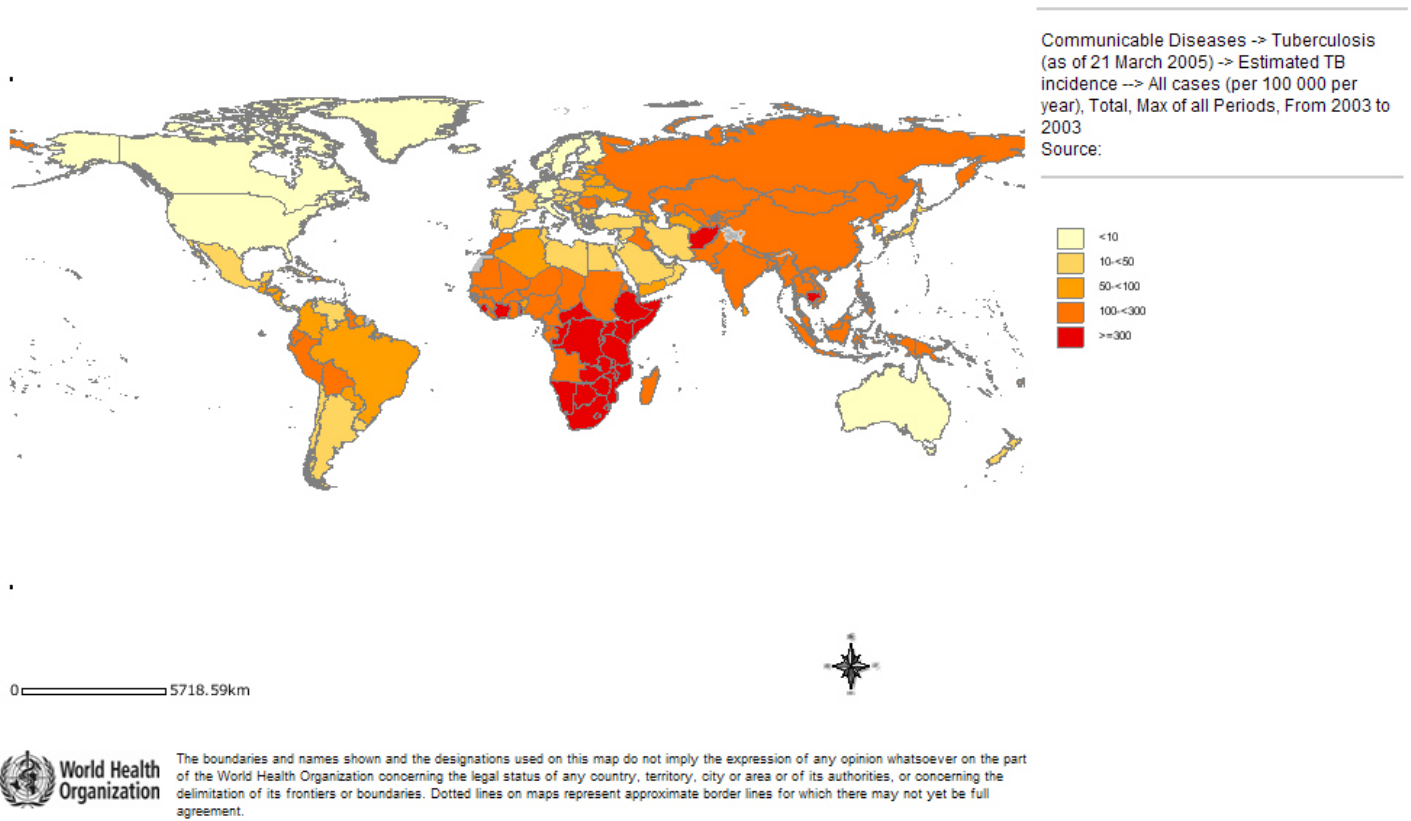
concomitant loss of DNA driven mainly by the activity of mobile DNA elements. *M. ulcerans*, but not *M. marinum*, expresses a plasmid-encoded immunomodulatory macrolide toxin, mycolactone, which plays an important role in virulence and pathology (16). Recently, it was suggested that the acquisition of this plasmid and the subsequent ability to produce mycolactone appears to be a key issue in the evolution of *M. ulcerans* from a common *M. marinum* progenitor (17)

## Johne's Disease

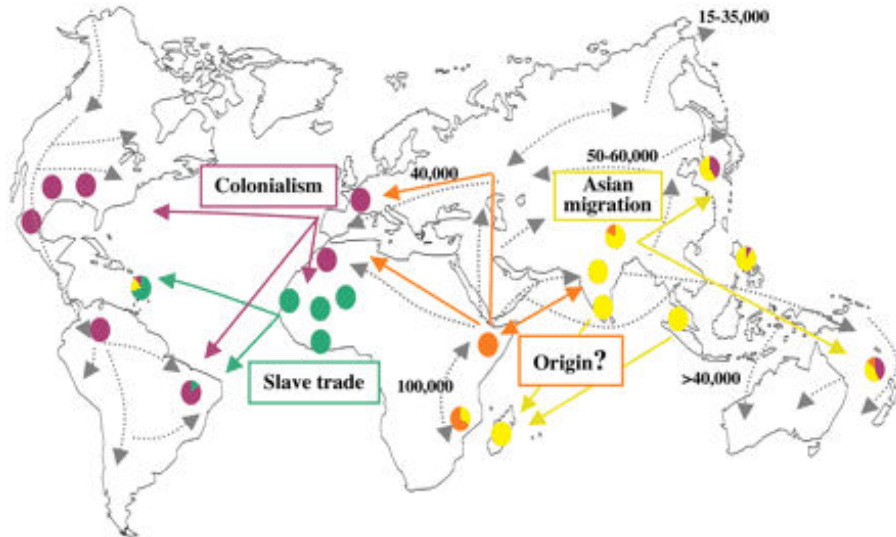
Johne's disease or paratuberculosis is a chronic disease produced by *Mycobacterium avium* subsp. *paratuberculosis* (MAP) that causes chronic enteritis in bovine, ovine, and other small ruminants, thus leading to production losses because of deficient food conversion. The disease has also been diagnosed in primates, deer, rabbits, foxes, and camels, as well as in a wide variety of wild animals (Johne's disease, paratuberculosis). It is currently associated with Crohn's disease in humans; therefore, it is considered a zoonotic disease (18).

MAP belongs to the *M. avium* complex (MAC), which comprises a group of opportunistic pathogenic mycobacteria able to infect humans and other animals and is also widely distributed in the environment. The group includes four species: *M. avium*, *Mycobacterium intracellulare*, and the recently described *Mycobacterium chimaera* (19) and *Mycobacterium colombiense* (20). *M. avium* has now been divided into four subspecies: *M. avium* subsp. *paratuberculosis*, *M. avium* subsp. *silvaticum*, *M. avium* subsp. *avium*, and *M. avium* subsp. *hominisuis* (21). The subspecies *paratuberculosis* presents higher than 95% sequence similarity with *M. avium* subsp. *avium* but is unique in the fact that the majority of its isolates require the iron-chelating agent mycobactin for growth.

Communicable Diseases -> Tuberculosis (as of 21 March 2005) -> Estimated TB incidence --> All cases (per 100 000 per year), Total, Max of all Periods, From 2003 to 2003



**Figure 2.** TB incidence, all forms (per 100,000 population per year), By Country, Total, 2004. From: WHO|Global TB database ([http://www.who.int/tb/country/global\\_tb\\_database/en/index1.html](http://www.who.int/tb/country/global_tb_database/en/index1.html)).



**Figure 3.** From Leprae Monot, Science 308:1040-42, 2005.



**Figure 4.** Buruli ulcer: global situation.

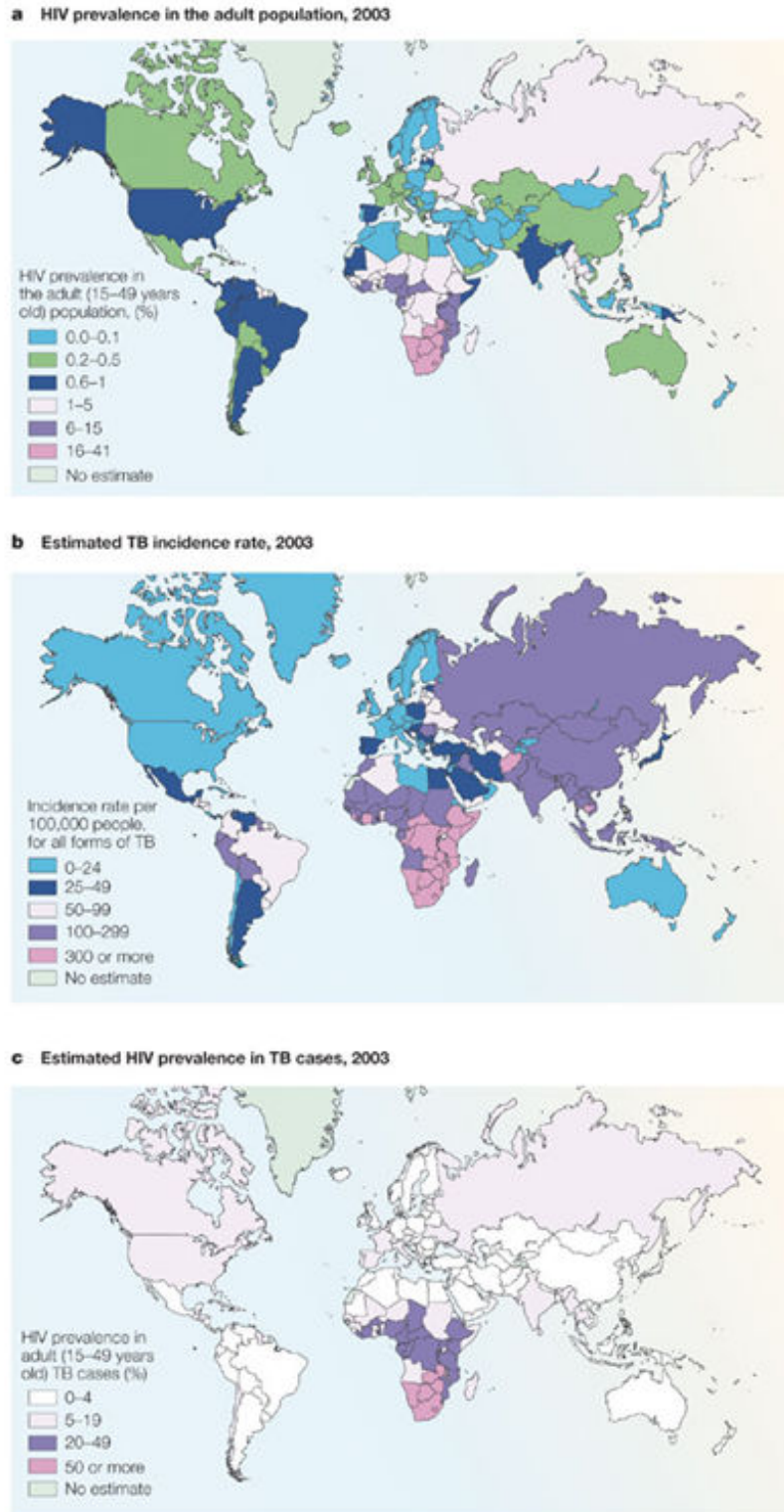
## Diseases Caused by Rapidly Growing Mycobacteria

Infections caused by rapidly growing mycobacteria (RGM) are not reportable diseases, and consequently their incidence and prevalence are largely unknown. Nevertheless, an increasing number of outbreaks and clusters of infections caused by RGM have been reported worldwide in the last years (22–27). RGM species are widely distributed in the environment and have been isolated from potable water, wastewater, soil, and inanimate surfaces (28).

Mycobacterial taxonomy is in constant evolution, and genotypic studies led to recent splitting and redefinition of species defined previously by phenotype-based methods (29). The rationale of genotypic taxonomy consists in the identification of highly conserved genomic regions harboring hypervariable sequences that show species-specific deletions, insertions, or replacements of single nucleotides. The 16S rRNA gene was, for many years, the primary target of molecular taxonomic studies, combined with analysis of unique mycobacterial cell wall lipids, the mycolic acids, and led to classification of RGM in three main groups: the *Mycobacterium chelonae-abscessus* group, the *M. fortuitum* group, and the *M. smegmatis* group.

RGM species are closely related to each other, and the indication that bacterial strains belong to the same species if they have fewer than 5- to 15-base differences in their 16S rRNA gene sequences is not applicable to RGM (30). Therefore, genes other than 16S rRNA have recently been included to assess phylogenetic relationships and to help clarify the taxonomy of RGM (31–34). Phylogenetic trees based upon individual sequences of five genes—16S rRNA, *recA*, *rpoB*, *sodA*, and *hsp65*—of 19 RGM species were compared with trees based on the combined dataset of the five genes and combined datasets of *recA* and *rpoB*. Six phylogenetic groups were recognized: the *M. chelonae-abscessus* group, *M. mucogenicum* group, *M. fortuitum* group, *M. mageritense* group, *M. wolinsky* group, and *M. smegmatis* group (35).

Information derived from the ribosomal RNA has been applied for further recognition of species within RGM. Mycobacteria have a minimum number of ribosomal operons, one or two copies per genome. Typical RGM have two copies. It was demonstrated that the chromosomal location of the two *rrn* operons in RGM is conserved within each species, as indicated by the Restriction Fragment Length Polymorphism (16S-RFLP)-derived patterns (36). Since the first description in 1994 (37), this approach has been demonstrated to complement the DNA-DNA hybridization data in the delimitation of closely related RGM species (38, 39).



Copyright © 2005 Nature Publishing Group  
 Nature Reviews | Immunology

**Figure 5.** From Nunn et al. (42).

The near completion of the genomes of *M. smegmatis*, *M. chelonae*, and *M. abscessus* will certainly further improve taxonomy studies and our understanding of these emerging pathogens.



## Mycobacteria and AIDS

Soon after the beginning of AIDS pandemic, it was evident that mycobacterial infections, not only tuberculosis but also opportunistic infections, were a major cause of morbidity and mortality among patients living with HIV/AIDS worldwide. MAC members are responsible for most opportunistic bacterial infections in late stages of the disease, and other non-tuberculous mycobacteria can also be involved. The panorama dramatically changed in countries where highly active antiretroviral treatment (HAART) became widely available, with the beneficial effect of restoration of pathogen-specific immune responses. Thus, the incidence of mycobacterial infections, including tuberculosis, decreased in treated cohorts living in high- and low-income countries (40).

MAC bacilli are acquired from the environment (person-to-person transmission had not been demonstrated), enter the host by the gastrointestinal mucosa, and cause disseminated disease with fatal consequences in patients with AIDS. Epidemiological evidence indicates that most AIDS patients with disseminated MAC infections have recently acquired these organisms, suggesting that infection was not due to previously viable MAC bacilli that could reactivate; therefore, MAC bacteria seem not to be able to establish latent infection, compared with that of *M. tuberculosis* (41).

Association of HIV infection and tuberculosis is the major adult infectious killer in the developing world, and approximately 10% of all global tuberculosis cases are attributable to HIV. About 13 million people are infected with both causative organisms. The biggest impact has been described in the sub-Saharan Africa: tuberculosis notifications have tripled since the mid-1980s, and death rates are fourth times higher compared with countries with good tuberculosis-control programs. Probably because of its association with HIV, Africa is the only continent where tuberculosis incidence is rising, causing a global increase of about 1% per year (42) (Figure 5).

Molecular epidemiology studies on MTBC strains isolated from AIDS patients showed that both re-infection and new infection occur in these patients. It has been speculated that HIV/AIDS patients could be an ecological niche for *M. tuberculosis* low virulent clones without the selective pressure provided by the immunocompetent patient (43). As a consequence of the increase in the number of tuberculosis cases due to HIV, the performance of tuberculosis-control programs has deteriorated. The lack of human resources is the major bottleneck for implementation of services in developing countries. Global development efforts have been established until 2015 within the [Millennium development goals](#) by the United Nations with prominent features addressed against Tuberculosis and HIV/AIDS. The achievements will depend on progress in Africa.

## Mycobacterial Genomes

Several mycobacterial genomes have been completed or are near completion. The updated list of mycobacterial genome projects can be viewed at [Mycobacterial genome-sequencing projects](#) and [GOLD: Genomes OnLine Database Homepage](#)

### *Mycobacterium tuberculosis* and *M. canettii*

The [The Wellcome Trust Sanger Institute](#) sequenced *M. tuberculosis* strain H37Rv using cosmids and BAC clones supplied by researchers at the [Institut Pasteur](#). This strain, isolated in 1905, retained full virulence in animal models and is widely used in tuberculosis research. The sequence of 4,411,529 base pairs (bp) has a G + C content of 65.6%. The genome is rich in repetitive DNA, particularly insertion sequences, new multigene families, and duplicated housekeeping genes. The annotated genome published in 1998 (44) identified 3,974 genes. Annotation was reviewed 4 years later (45) with identification of 82 additional genes. The annotated sequence is deposited in public databases with accession number [AL123456](#). Information about genes and predicted proteins can be obtained at [TubercuList Web Server](#).

A second *M. tuberculosis* strain was sequenced at [The Institute for Genomic Research](#) (TIGR). This virulent isolate, CDC1551, was involved in a cluster of tuberculosis cases in the USA and was highly transmissible to

humans. Comparison of the two genome sequences revealed large-sequence and single-nucleotide polymorphisms (46).

The *M. tuberculosis* strain 210, from the Beijing family, widely distributed in the USA (47), is now complete at TIGR. <http://www.tigr.org/tdb/mdb/mdbinprogress.html>The Broad Institute is engaged in sequencing additional *M. tuberculosis* strains: A1, C, Ekat-4, F11, Haarlen, KZN 605 (XDR), KZN 1435 (MDR), KZN 4207 (DS), Peruvian 1, Peruvian 2, W-148, and *M. tuberculosis* spp. Recent achievements of these genome projects are: F11: Finished as a single supercontig (scaffold) with high quality; C: The current assembly contains 160 sequence contigs in 4 supercontigs (scaffolds); and Haarlem: The current assembly contains 65 sequence contigs in 8 supercontig (scaffold). The entire genomes can be [downloaded](#)

*M. canetti* is a member of the tubercle bacilli, with untypical smooth colony morphology, that has been isolated from tuberculosis patients in East Africa. Shotgun sequencing of strain CIP 140010059 will begin soon at the [Sanger Institute](#)

### ***Mycobacterium bovis*, BCG, *M. africanum*, and *M. microti***

The complete sequence of *M. bovis* strain AF2122/97, a fully virulent UK isolate from a cow showing caseous lesions in lung and bronchomediastinal lymph nodes, was obtained by the [The Wellcome Trust Sanger Institute](#) in collaboration with [Institut Pasteur](#) and is available at the [BoviList Web Server](#). Annotation was managed using [ARTEMIS](#) with single-nucleotide polymorphism (SNP) identification performed by using the EMBOSS package. The sequence and annotation have been deposited in the EMBL database under accession number [BX248333](#). The sequence and analysis have now been published (48).

The genome sequence of *M. bovis* AF2122/97 is >99.95% identical to that of *M. tuberculosis*, but deletion of genetic information has led to a reduced genome size (4,345,492 bp), which is about 66 kb smaller than the one from *M. tuberculosis* H37Rv. Cell wall components and secreted proteins show the greatest variation.

As an extension to the *M. bovis* genome project, the [The Wellcome Trust Sanger Institute](#), in collaboration with the [Institut Pasteur](#), determined the genome sequence of *M. bovis* BCG-Pasteur 1173P2 using clones from the [ordered BAC library](#) and the physical map as tools. A 4×-coverage shotgun sequence of *Mycobacterium bovis* BCG Pasteur was obtained. Comparison of the BCG genome sequence with those of virulent tubercle bacilli highlighted the genetic rearrangements that led to attenuation of this live vaccine strain (49). A second strain of *M. bovis* BCG, Moreau RDJ, the Brazilian vaccine strain of *Mycobacterium bovis* that is very immunogenic with few side effects, is being sequenced using random shotgun by researchers from FIOCRUZ and [Fundação Atauilho de Paiva](#), in Brazil. This strain will be used for comparative genomics.

The [Sanger Institute](#) is sequencing the genome of *M. africanum* strain GM041182. Shotgun sequencing is complete. A database of reads is available for searching at the [Blast Server](#), or for download from the [FTP](#) site.

The Sanger Institute is sequencing the genome of *M. microti* strain OV254 (OV stands for Orkney Vole), in collaboration with the Pasteur Institute ([Unité de Génétique Moléculaire Bactérienne](#)). A whole genome screen based on BAC maps was also undertaken (50).

### ***Mycobacterium leprae***

The sequencing project of the complete genome of a strain of *M. leprae*, originally isolated in Tamil Nadu (TN), is finished ([Mycobacterium leprae genome project](#)). The complete sequence is 3,268,203 bp in length with a G+C content of 57.8% and was generated from a combination of cosmid and whole-genome shotgun sequencing. There are 1,604 protein-coding genes and 1,116 pseudogenes (10). Both the sequence and annotation have been deposited in the public databases with the accession number [AL450380](#). The project was undertaken as a collaboration between the Sanger Institute and the Pasteur Institute ([Leproma Web Server](#)).

Analysis of the leprosy genome indicates that it had suffered important genomic downsizing, which appears to have resulted from extensive recombination events between dispersed repetitive sequences. As a consequence, it had lost essential metabolic routes, including most of the microaerophilic and anaerobic respiratory chains, providing explanation for its intracellular parasitic lifestyle (10). In this respect, the *M. leprae* genome is very different from the other mycobacterial genomes analyzed thus far.

In a recent paper, the use of bioinformatics and comparative genomics methods allowed the identification of putative leprosy-specific antigenic proteins suitable for use for diagnostic purposes (51).

### ***Mycobacterium marinum* and *M. ulcerans***

The genome project of *M. marinum*, M strain, is now finished. It is 6,636,827 bp in length, with an average G+C content of 65.73%. The project has been accomplished by the Sanger Institute in collaboration with research centers from the USA, France, and Australia ([Mycobacterium marinum genome project](#)).

Institut Pasteur is sequencing the 4.4-Mb genome of an epidemic strain of *M. ulcerans* ([Unité de Génétique Moléculaire Bactérienne](#)). The [BuruList Web Server](#) is currently operating only as a BLAST server because the genome is still in a preliminary stage of assembly. In parallel with the genome project, pulsed field gel electrophoresis experiments and sequence analysis have led to the identification of a large plasmid that is required for the production of mycolactone (52).

The [Clemson University Genomics Institute](#) has also started a sequencing project of *M. ulcerans*.

### ***Mycobacterium avium* subsp. *avium* and *M. avium* subsp. *paratuberculosis***

The genome sequence of *M. avium* subsp. *avium* (strain 104) is in progress at [TIGR](#). The genome has 4.70 million base pairs (Mb). The sequence is complete, and all gaps are closed. This strain is a common representative of the species complex, isolated from an AIDS patient.

The [sequence of MAP](#) (strain K-10) has been recently completed at Minnesota University and consists of a circular chromosome of 4.83 Mb and contains about 4,350 open reading frames (53). This strain is a virulent bovine clinical isolate, isolated in Wisconsin in the mid-1970s. A random shotgun approach was adopted to sequence the genome of MAP K-10.

### **Rapidly Growing Mycobacteria**

The sequence of *M. smegmatis* strain mc<sup>2</sup>155 is in progress at [TIGR](#). New sequences generated for gap closure have been assembled and added to the data release. This strain is a highly efficient plasmid transformation mutant, widely used in mycobacterial research (54).

[Genoscope](#) is undertaking sequencing of type strains *M. abscessus* ATCC 19977, isolated from a knee abscess in 1953, and *M. chelonae* ATCC 35752, isolated from turtle in 1921, which were already compared by subtractive hybridization by the same group.

Mycobacterial strains involved in degradation of environmentally hazardous chemicals (55, 56) are being sequenced by [DOE Joint Genome Institute](#).

## **Impact of Genome Projects on Mycobacterial Research**

The recent determination of genomic nucleotide sequences of *M. tuberculosis* H37Rv and CDC 1551, *M. bovis*, *M. leprae*, and MAP makes *Mycobacterium* one of the most sequenced bacterial genera (57). This wealth of

comparative genome sequence information provides unique opportunities for new insights into the biology of these globally important pathogens.

## Microarrays

Concurrent emergence of new functional genomic technologies such as microarrays, in addition to recently developed genetic recombination tools for mycobacteria, creates new opportunities to exploit genome information to address the scientific imperatives of better drugs, vaccines, and diagnostics for mycobacterial diseases. Tuberculosis microarrays have mainly been used (31) to compare genomes within the *M. tuberculosis* complex (35), simultaneously monitor the relative expression of every gene in the genome by comparing mRNA levels, and (16) for analysis of SNPs for genotyping and drug resistance detection using high-density oligonucleotides array. Whole genome microarrays for *M. tuberculosis* are available through the [Pathogen Functional Genomics Resource Center at TIGR](#) and [St. George's Hospital Medical School](#).

Tuberculosis is a complex disease influenced by the genotypes of both host and pathogen as well as the immunological status of the host; then microarrays containing genes from the host can also be used to study host–*M. tuberculosis* interactions. Experimental mouse infection with *M. tuberculosis* (*in vivo* experiments) (58) or mouse macrophage infection models (*in vitro* experiments) (59) represent an approach to study *M. tuberculosis* pathogenesis. [Affymetrix](#) supplies a gene chip with murine genes for these experiments.

The microarray methodology has been extensively applied to analyze one of the most intriguing features that characterize the infection caused by the tubercle bacilli: the dormant stage. Transcriptomic analyses have been performed in both *in vivo* and *in vitro* models of dormancy, and a *dormancy regulon* was identified in the conditions tested. This regulon is under the control of the two-component system *dosR/S* and encompasses up to 48 different genes; unfortunately, nearly one-third of the genes putatively related to dormancy in *M. tuberculosis* are of unknown function, indicating the scarce information available concerning this infective situation (60–63).

A collaborative research between VLA of Weybridge, UK and the Argentinean laboratory was done using a DNA microarray containing non-redundant CDS from the two sequenced *M. tuberculosis* strains, CDC1551 and H37Rv, and from *M. bovis* AF2122/97 to evaluate genetic variability between *M. tuberculosis* complex strains (64, 65). In *M. microti*, a new deletion was identified that removes five genes that code for ESAT-6 family antigens and PE-PPE proteins. This region, called MiD4, was also found to be deleted from *M. pinnipedii*, supporting the previous idea of a common evolutionary lineage for both species (65). To extend the repertoire of these deletion markers, a whole genome microarray analysis of the recently defined *M. pinnipedii* species was done (64). Two deletions that are exclusive to *M. pinnipedii* were found: PiD1 that removes the orthologs of the *M. tuberculosis* genes *Rv3530c* and *Rv3531c*, and PiD2 that encompasses genes *Rv1977* and *Rv1978* (64).

## New Typing Methods

The classical molecular epidemiology that was applied in the analysis of the mycobacterial diseases was based on Restriction Fragment Polymorphism Analysis (RFLP) patterns attributable to the chromosomal distribution of specific insertion sequences. Comparison of the patterns derived from the IS6110–RFLP analysis has been an essential tool to track the transmission and distribution of *M. tuberculosis* isolates worldwide, helping in the control of the disease. Data derived from completion of genome sequencing allowed the development of new, quicker, and also highly discriminative PCR-based typing methods, such as MIRU–VNTR and Spoligotyping, the latter only applicable to MTBC thus far.

### MIRU–VNTR

Special tandem repeat sequences such as the Variable Number Tandem Repeat (VNTR) loci, called Mycobacteria Interspersed Repetitive Units (MIRU), have been described in *M. tuberculosis* ([MIRU Inventory](#)), *M. bovis* ([Mycobacterium bovis molecular typing database](#)), *M. leprae* and *M. avium* complex (66–75). The

properties of MIRU that distinguish them from other tandem repeat sequences are that they have regulatory elements, such as initiation and stop codons and consensus to bacterial ribosome-binding site 5'-TGA GGA GGA GC-3'. Most MIRU overlap the termination and initiation codons of their flanking genes (74). Polymorphism at these tandem repeat loci can occur as a result of either nucleotide sequence changes among individual repeat units or variation in the number of repeat units.

## Spoligotyping

In human tuberculosis, molecular typing has advanced by the use of the insertion sequence IS6110 (76), which is repeated many times in the *M. tuberculosis* genome, producing a genotypic heterogeneity of isolates. In *M. bovis*, IS6110 is less useful because the genome of most strains contains only one or very few IS6110 copies (77). The insertion element IS6110 is frequently found in a unique locus of the *M. tuberculosis* complex genome called the direct repeat (DR) region. A more recent technique, called spoligotyping (78), is highly efficient for typing MTBC, including *M. bovis*. With this method, DNA polymorphisms in the genomic DR locus of MTBC isolates are visualized. This locus contains multiple, well-conserved 36-bp DRs interspersed with nonrepetitive 34- to 41-bp DNA spacer sequences. Spoligotyping involves the amplification of the whole DR region, followed by hybridization of the amplified DNA to a set of spacer oligonucleotides, covalently linked to a membrane (*Mycobacterium bovis* spoligotype database).

Considering that spoligotyping is a rapid and easy-to-apply technique, it is important to improve its capacity for differentiation. New spacers of the DR region were analyzed in Argentinian isolates of *M. bovis*. These isolates were discriminated more accurately by the novel probe spoligotyping than by traditional spoligotyping (79).

Two databases, SPOLDB4 and MIRU-VNTRplus, containing MIRU-VNTR and spoligotype patterns, are now available for online comparison and analysis of patterns obtained in the laboratory.

## Comparative Genomics

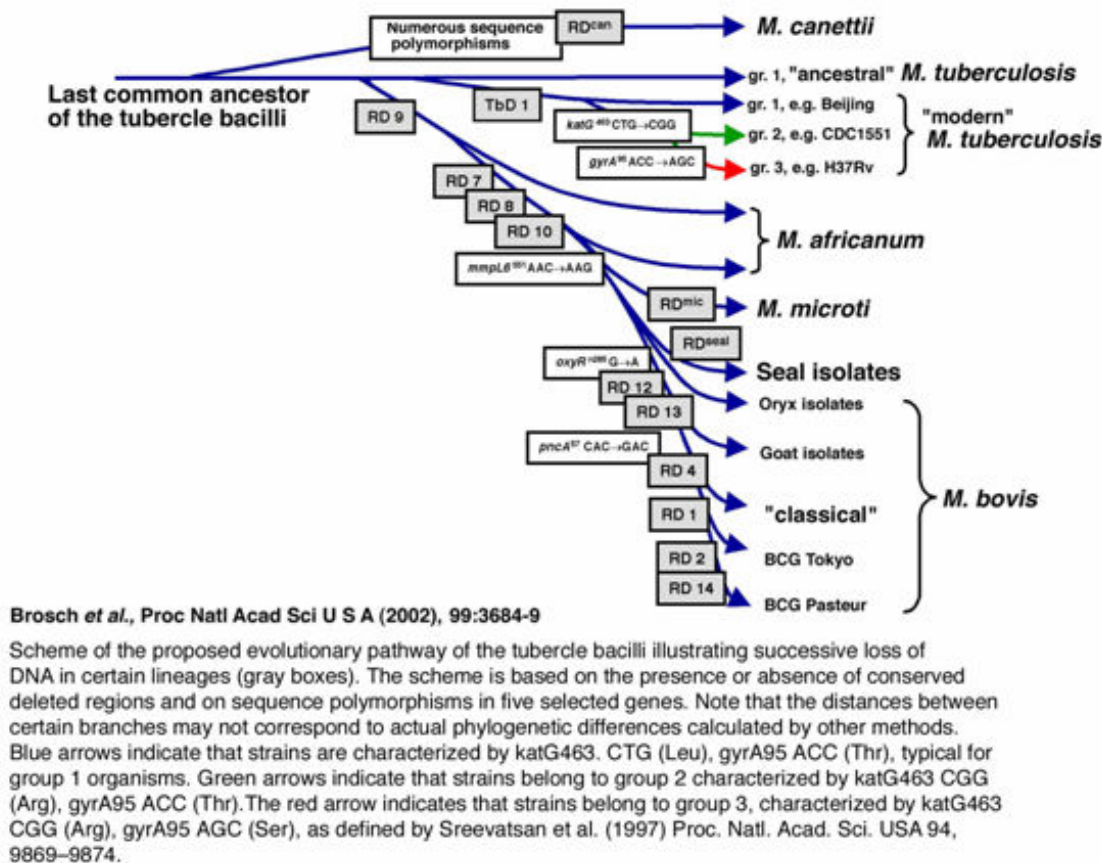
Comparison of mycobacterial genomes can be performed by a set of different approaches. One of them, the array technology, easily identify deletion events but cannot readily detect insertions or duplications. Other powerful method includes whole-genome sequence comparison that covers from SNP to gene rearrangements (80) and allows detection of large sequence polymorphisms (LSP), considered the major contributor to genetic diversity within members of the genus. Several of these methodologies have been used to compare members of MTBC, and more rarely were applied to other members of the genus.

### Comparative Genomics between Members of MTBC

Whole genome comparisons of *M. tuberculosis* H37Rv and *M. bovis* BCG Pasteur were undertaken using two different approaches. Using BAC-arrays and direct comparison of canonical BACs from ordered libraries, several polymorphic genomic regions were uncovered (48). By performing comparative hybridization experiments on a DNA microarray comprising 4896 spots, representing 3902 ORFs of *M. tuberculosis* H37Rv, 16 regions of difference (RD1-16) were shown to be absent from some *M. bovis* BCG relative to *M. tuberculosis* H37Rv (81). In parallel, five regions deleted in H37Rv were described (RvD)1-5 (82) compared with genomes of other members of the complex. The combined findings of these studies can be found at [UnitéGénétique Moléculaire Bactérienne](#).

Recently, a *M. tuberculosis*-specific deletion (TbD1) was identified. This deletion is characterized by the absence of a 2,153-bp fragment truncating genes *mmpS6* and *mmpL6* in “modern” strains of *M. tuberculosis*; however, this region is present in all other members of the *M. tuberculosis* complex (6).

In another study (83), *M. tuberculosis* H37Rv and *M. bovis* AF2122/97 genomic sequences were compared using blastn and MSPcrunch (84) software and visualized with the [Artemis Comparison Tool](#) (ACT) version 4.0 for Apple. The criteria used to select loci for further studies were that they were not IS6110 insertions, PE-PPE



**Figure 6.** From Brosch et al. (82).

family genes, single-nucleotide polymorphisms, or previously published regions of difference. Four polymorphisms were found: 1) a deletion of *Rv3479* specific to *M. bovis*; 2) that the *rpfA* gene is shortened to various extents in *M. bovis*; 3) an insertion in *Rv0648* in *M. bovis*; and 4) a duplication of *lppA/B* in *M. bovis*. Interestingly, *lppA* is also duplicated in *M. tuberculosis* CDC 1551.

Comparative genomics has given also some clues to understand the evolution of members of the complex. Thus, the analysis of the distribution of variable regions indicates that, contrary to the initial proposed scheme, *M. tuberculosis* did not descend from *M. bovis*; instead, both bacteria descended from a single common ancestor named “ancient” TB, which appears to be closely related to *M. canetti*, a member of the complex isolated from humans described recently. Both *M. bovis* and “modern” *M. tuberculosis* evolved from that common ancestor, as well as all other components (82) (Figure 6).

Identification of the different members of MTBC in the laboratory is difficult using standard methodologies; however, their correct differentiation has important influence in the treatment to be chosen and the consequent management of the patients. The development of comparative genomics has allowed the description of PCR-based schemes for such purposes. On the basis of variable distribution of insertion and deletions in the genomes of the several members, some flow charts were described for *M. tuberculosis* complex differentiation (80, 85).

## Comparative Genomics between Members of MAC

Some few studies have been performed mainly focused on the detection of specific targets for identification of MAP, by comparison of its genome with that of other related members of MAC. The annotated sequence of *M. avium* subsp. *avium* strain 104 (provided by TIGR) was used initially to assemble a whole-genome DNA

microarray representing the predicted coding sequences. Seventy-base-pair-long oligonucleotide probes were designed and synthesized for 4,158 of 4,480 predicted open reading frames (ORFs). Each probe was printed in duplicate onto microarray slides to permit genomic DNA comparison of *M. avium* subsp. *avium* strain 104 and the following strains: 1) *M. avium* subsp. *paratuberculosis* K10 (cow strain); 2) *M. avium* subsp. *paratuberculosis* LN20 (sheep strain); and 3) *M. avium* subsp. *silvaticum* 49884 (ATCC strain). Microarray comparison revealed 14 LSP regions that distinguish a single strain of *M. avium* subsp. *avium* (MAA) from other strains of MAP (86). These LSP regions encompass 572 genes, more than 700 Kb, that represents 13.5% of the MAA genome. Remarkably higher diversity was demonstrated in this complex compared with the genomic variability described among *M. tuberculosis* complex isolates (87). This could be in agreement with consideration of MAP and MAA as different subspecies within the species *M. avium*.

As part of this ongoing work to characterize *M. avium* complex organisms, sequenced genome of *M. avium* subsp. *avium* strain 104 (TIGR) and *M. avium* subsp. *paratuberculosis* strain K10 (GenBank accession number NC\_002944) were compared using Artemis software for visualization of the *M. avium* subsp. *paratuberculosis* strain K10 genome and annotation of *M. avium* subsp. *avium* strain 104. Genome sequence comparison, via visual inspection in Artemis, revealed two types of LSPs: those present in the former but missing in the latter (LSP<sup>A</sup>s), and those only present in the latter (LSP<sup>P</sup>s). Three LSP<sup>A</sup>s and 17 LSP<sup>P</sup>s were found. The distribution of these LSPs were examined across a panel of *M. avium* complex isolates, revealing 1 LSP<sup>A</sup> absent in all MAP isolates tested; then this absence was described as 100% specific of the MAP genome, and 7 LSP<sup>P</sup>s genomic regions present only in the analyzed MAP strains, and described as highly specific of MAP isolates; other 10 LSP<sup>P</sup> regions were not specific of the MAP genome (86).

The analysis of a larger number of MAP isolates showed that these isolates have a high degree of genetic conservation, with no differences with the reference strain K10. Deletions detected by comparison with other members of the MAC are, as expected, associated with the presence of mobile genetic elements (88).

## Comparative Genomics as a Tool for the Analysis of Mycobacterial Speciation

The pool of different mycobacterial genome projects currently in progress opens new possibilities for comparison between members of different species in the genus, a task still to be undertaken. LSPs, based on insertion/deletions and rearrangements of the genomes, are being considered the main mechanism that occurs during the evolution of the bacterial genomes. These changes are considered important in the separation of isolates in different genomic species and could putatively be studied by determining a gene's location and its organization in the chromosome.

Recently, a new website ([Microbesonline](#)) has been described and organized for quick comparison of bacterial genomes (89). Only five mycobacterial genomes are included in that site thus far, three of them members of MTBC, plus MAP and *M. leprae*.

Using the facilities offered at that website ([MicrobesOnline Comparative Genome Browser](#)), a multispecies comparison of the location and gene organization of the available mycobacterial genomes was performed. An expected result was found when the *M. leprae* genome was analyzed; this genome showed major differences from all other mycobacterial genomes, differences that were scattered along the chromosome. As an exception, only short regions showed conserved gene organization in the *M. leprae* genome when compared with the equivalent region in other mycobacterial genomes. For example, the region corresponding to *recN* (ML1360 and Rv1696) shows gene conservation in about a 10-Kb length among all five compared mycobacterial genomes; and the region corresponding to *dnaA* (ML0001 and Rv0001) is also conserved along 12 Kb in all five genomes.

The genome of *M. tuberculosis* H37Rv was considered as reference for organization of the contained ORFs (from Rv0001 to Rv3924) and was theoretically divided into eight regions (containing approximately 500 ORFs each), taking the following ORFs as references: Rv0001 (*dnaA*); Rv0500 (*proC*); Rv1001 (*arcA*); Rv1522 (*mmpL12*); Rv2031 (*acr/hspX*); Rv2500 (*fadE19*); Rv3065 (*emr*); and Rv3504 (*fadE26*). Approximately 50 Kb were checked

upstream and downstream from each of the previous genes. A genome multispecies browser was applied to each of the selected regions, and as a result, the eight genomic regions could be divided into three groups.

Group I regions were conserved in members of the MTBC as well as in MAP. This includes Rv0001, where genes related to chromosome replication and partitioning are located. Group II regions were conserved in members of the MTBC but were different than the corresponding region in MAP. This includes Rv0500 (this region encloses the *icl* gene, which has been related to dormancy in *M. tuberculosis*) and Rv3065 (this region includes components of the cluster *nrd* involved in DNA replication). Group III variable regions were conserved in all four mycobacteria, including members of the MTBC. This includes four of the remaining regions, such as Rv1522 (the *mmpL12* gene thought to be involved in the fatty acid transport), Rv2031 (gene coding for the stress protein  $\alpha$ -crystallin), Rv2500 (belonging to a family of genes associated with fatty acid degradation and whose components have more or less scattered distribution along the chromosome), and Rv3504 (several genes coding for members of the PE-PGRS family of proteins, which have been related to the pathogenesis of the genus, are located within this region). A curious result was that the region defined by the gene Rv1001 divided two genomic zones: one belongs to Group II, located in the corresponding downstream region, and the other to Group III, located in the corresponding upstream region.

The previous data are deeply biased because of the small number of species compared (actually only three clearly different species); however, as more mycobacterial genomes were included in this site, a similar approach can be used to help in the detection of genomic regions that putatively would participate in mycobacterial speciation.

## References

1. Euzéby JP. 2004, posting date. List of Bacterial names with Standing in Nomenclature. Société de Bactériologie Systématique et Vétérinaire [online.]
2. Aranaz A, Cousins D, Mateos A, Dominguez L. Elevation of *Mycobacterium tuberculosis* subsp. *caprae* Aranaz et al. 1999 to species rank as *Mycobacterium caprae* comb. nov., sp. nov. *Int J Syst Evol Microbiol.* 2003;53:1785–1789. PubMed PMID: 14657105.
3. Niemann S, Richter E, Rusch-Gerdes S. Biochemical and genetic evidence for the transfer of *Mycobacterium tuberculosis* subsp. *caprae* Aranaz et al. 1999 to the species *Mycobacterium bovis* Karlson and Lessel 1970 (approved lists 1980) as *Mycobacterium bovis* subsp. *caprae* comb. nov. *Int J Syst Evol Microbiol.* 2002;52:433–436. PubMed PMID: 11931153.
4. Cousins DV, Bastida R, Cataldi A, Quse V, Redrobe S, Dow S, Duignan P, Murray A, Dupont C, Ahmed N, Collins DM, Butler WR, Dawson D, Rodriguez D, Loureiro J, Romano MI, Alito A, Zumarraga M, Bernardelli A. Tuberculosis in seals caused by a novel member of the *Mycobacterium tuberculosis* complex: *Mycobacterium pinnipedii* sp. nov. *Int J Syst Evol Microbiol.* 2003;53:1305–1314. PubMed PMID: 13130011.
5. Gutierrez MC, Brisse S, Brosch R, Fabre M, Omais B, Marmiesse M, Supply P, Vincent V. Ancient origin and gene mosaicism of the progenitor of *Mycobacterium tuberculosis*. *PLoS Pathog.* 2005;1:e5. PubMed PMID: 16201017.
6. Brosch R, Behr MA. Comparative Genomics and evolution of *Mycobacterium bovis* BCG. In: Cole ST, et al, editors. *Tuberculosis and Tubercle bacilli.* ASM press; 2005. pp. 155–164.
7. Mostowy S, Behr MA. The origin and evolution of *Mycobacterium tuberculosis*. *Clin Chest Med.* 2005;26:207–216. PubMed PMID: 15837106.
8. Latini MS, Latini OA, Lopez ML, Cecconi JO. Tuberculosis bovina en seres humanos. *Rev Argent Torax.* 1990;51:13–16.
9. Samper S, Martin C, Pinedo A, Rivero A, Blazquez J, Baquero F, van Soolingen D, van Embden J. Transmission between HIV-infected patients of multidrug-resistant tuberculosis caused by *Mycobacterium bovis*. *AIDS.* 1997;11:1237–1242. PubMed PMID: 9256941.
10. Cole ST, Eiglmeier K, Parkhill J, James KD, Thomson NR, Wheeler PR, Honore N, Garnier T, Churcher C, Harris D, Mungall K, Basham D, Brown D, Chillingworth T, Connor R, Davies RM, Devlin K, Duthoy S,



- Feltwell T, Fraser A, Hamlin N, Holroyd S, Hornsby T, Jagels K, Lacroix C, Maclean J, Moule S, Murphy L, Oliver K, Quail MA, Rajandream MA, Rutherford KM, Rutter S, Seeger K, Simon S, Simmonds M, Skelton J, Squares R, Squares S, Stevens K, Taylor K, Whitehead S, Woodward JR, Barrell BG. Massive gene decay in the leprosy bacillus. *Nature*. 2001;409:1007–1011. PubMed PMID: 11234002.
11. Monot M, Honore N, Garnier T, Araoz R, Coppee JY, Lacroix C, Sow S, Spencer JS, Truman RW, Williams DL, Gelber R, Virmond M, Flageul B, Cho SN, Ji B, Paniz-Mondolfi A, Convit J, Young S, Fine PE, Rasolofy V, Brennan PJ, Cole ST. On the origin of leprosy. *Science*. 2005;308:1040–1042. PubMed PMID: 15894530.
  12. Gomez-Valero L, Rocha ECP, Latorre A, Silva FJ. Reconstructing the ancestor of *Mycobacterium leprae*: the dynamics of gene loss and genome reduction. *Genome Res*. 2007;17:1178–1185. PubMed PMID: 17623808.
  13. Debacker M, Aguiar J, Steunou C, Zinsou C, Meyers WM, Guedenon A, Scott JT, Dramaix M, Portaels F. *Mycobacterium ulcerans* disease (Buruli ulcer) in rural hospital, Southern Benin, 1997–2001. *Emerg Infect Dis*. 2004;10:1391–1398. PubMed PMID: 15496239.
  14. Stinear T, Ross BC, Davies JK, Marino L, Robins-Browne RM, Oppedisano F, Sievers A, Johnson PD. Identification and characterization of IS2404 and IS2606: two distinct repeated sequences for detection of *Mycobacterium ulcerans* by PCR. *J Clin Microbiol*. 1999;37:1018–1023. PubMed PMID: 10074520.
  15. Stinear TP, Jenkin GA, Johnson PD, Davies JK. Comparative genetic analysis of *Mycobacterium ulcerans* and *Mycobacterium marinum* reveals evidence of recent divergence. *J Bacteriol*. 2000;182:6322–6330. PubMed PMID: 11053375.
  16. Adusumilli S, Mve-Obiang A, Sparer T, Meyers W, Hayman J, Small PL. *Mycobacterium ulcerans* toxic macrolide, mycolactone modulates the host immune response and cellular location of *M. ulcerans* in vitro and in vivo. *Cell Microbiol*. 2005;7:1295–1304. PubMed PMID: 16098217.
  17. Yip MJ, Porter JL, Fyfe JAM, Lavender CJ, Portaels F, Rhodes M, Kator H, Colorni A, Jenkin GA, Stinear T. Evolution of *Mycobacterium ulcerans* and other mycolactone-producing mycobacteria from a common *Mycobacterium marinum* progenitor. *J Bacteriol*. 2007;189:2021–2029. PubMed PMID: 17172337.
  18. Hermon-Taylor J, Bull T. Crohn's disease caused by *Mycobacterium avium* subspecies paratuberculosis: a public health tragedy whose resolution is long overdue. *J Med Microbiol*. 2002;51:3–6. PubMed PMID: 11800469.
  19. Tortoli E, Rindi L, Garcia MJ, Chiaradonna P, Dei R, Garzelli C, Kroppenstedt RM, Lari N, Mattei R, Mariottini A, Mazzarelli G, Murcia MI, Nanetti A, Piccoli P, Scarparo C. Proposal to elevate the genetic variant MAC-A, included in the *Mycobacterium avium* complex, to species rank as *Mycobacterium chimaera* sp. nov. *Int J Syst Evol Microbiol*. 2004;54:1277–1285. PubMed PMID: 15280303.
  20. Murcia MI, Tortoli E, Menendez MC, Palenque E, Garcia MJ. *Mycobacterium colombiense* sp. nov., a novel member of the *Mycobacterium avium* complex and description of MAC-X as a new ITS genetic variant. *Int J Syst Evol Microbiol*. 2006;56:2049–2054. PubMed PMID: 16957098.
  21. Mijs W, de Haas P, Rossau R, Van der Laan T, Rigouts L, Portaels F, van Soolingen D. Molecular evidence to support a proposal to reserve the designation *Mycobacterium avium* subsp. *avium* for bird-type isolates and 'M. avium subsp. *hominissuis*' for the human/porcine type of *M. avium*. *Int J Syst Evol Microbiol*. 2002;52:1505–1518. PubMed PMID: 12361252.
  22. Freitas D, Alvarenga L, Sampaio J, Mannis M, Sato E, Sousa L, Vieira L, Yu MC, Martins MC, Hoffling-Lima A, Belfort R. An outbreak of *Mycobacterium chelonae* infection after LASIK. *Ophthalmology*. 2003;110:276–285. PubMed PMID: 12578767.
  23. Sampaio JLM, Chimara E, Ferrazoli L, Telles MAS, Del Guercio VMF, Jericó ZVN, Miyashiro K, Fortaleza CMCB, Padoveze MC, Leao SC. Use of four molecular methods for typing of *Mycobacterium fortuitum* group strains causing post augmentation mammoplasty infections. *Clin Microbiol Infect*. 2006;12:142–149. PubMed PMID: 16441452.
  24. Tiwari TS, Ray B, Jost KC, Rathod MK, Zhang Y, Brown-Elliott BA, Hendricks K, Wallace RJ. Forty years of disinfectant failure: outbreak of postinjection *Mycobacterium abscessus* infection caused by contamination of benzalkonium chloride. *Clin Infect Dis*. 2003;36:954–962. PubMed PMID: 12684906.
  25. Wallace RJ, Brown BA, Griffith DE. Nosocomial outbreaks/pseudo-outbreaks caused by nontuberculous mycobacteria. *Annu Rev Microbiol*. 1998;52:453–490. PubMed PMID: 9891805.

26. Winthrop KL, Abrams M, Yakrus M, Schwartz I, Ely J, Gillies D, Vugia DJ. An outbreak of mycobacterial furunculosis associated with footbaths at a nail salon. *N Engl J Med.* 2002;346:1366–1371. PubMed PMID: 11986410.
27. Campos-Herrero MI, Garcia D, Figuerola A, Suarez P, Campo C, Garcia MJ. Bacteremia caused by the novel species *Mycobacterium canariasense*. *Eur J Clin Microbiol Infect Dis.* 2006;25:58–60. PubMed PMID: 16391913.
28. Primm TP, Lucero CA, Falkinham JO. Health impacts of environmental mycobacteria. *Clin Microbiol Rev.* 2004;17:98–106. PubMed PMID: 14726457.
29. Tortoli E. Impact of genotypic studies on mycobacterial taxonomy: the new mycobacteria of the 1990s. *Clin Microbiol Rev.* 2003;16:319–354. PubMed PMID: 12692101.
30. Fox GE, Wisotzkey JD, Jurtshuk P. How close is close: 16S rRNA sequence identity may not be sufficient to guarantee species identity. *Int J Syst Bacteriol.* 1992;42:166–170. PubMed PMID: 1371061.
31. Adekambi T, Colson P, Drancourt M. *rpoB*-based identification of nonpigmented and late-pigmenting rapidly growing mycobacteria. *J Clin Microbiol.* 2003;41:5699–5708. PubMed PMID: 14662964.
32. Blackwood KS, He C, Gunton J, Turenne CY, Wolfe J, Kabani AM. Evaluation of *recA* sequences for identification of *Mycobacterium* species. *J Clin Microbiol.* 2000;38:2846–2852. PubMed PMID: 10921937.
33. Domenech P, Jimenez MS, Menendez MC, Bull TJ, Samper S, Manrique A, Garcia MJ. *Mycobacterium mageritense* sp. nov. *Int J Syst Bacteriol.* 1997;47:535–540. PubMed PMID: 9103645.
34. McNabb A, Eisler D, Adie K, Amos M, Rodrigues M, Stephens G, Black WA, Isaac-Renton J. Assessment of partial sequencing of the 65-kilodalton heat shock protein gene (*hsp65*) for routine identification of *Mycobacterium* species isolated from clinical sources. *J Clin Microbiol.* 2004;42:3000–3011. PubMed PMID: 15243051.
35. Adekambi T, Drancourt M. Dissection of phylogenetic relationships among 19 rapidly growing *Mycobacterium* species by 16S rRNA, *hsp65*, *sodA*, *recA* and *rpoB* gene sequencing. *Int J Syst Evol Microbiol.* 2004;54:2095–2105. PubMed PMID: 15545441.
36. Menendez MC, Garcia MJ, Navarro MC, Gonzalez-y-Merchand JA, Rivera-Gutierrez S, Garcia-Sanchez L, Cox RA. Characterization of an rRNA operon (*rrnB*) of *Mycobacterium fortuitum* and other mycobacterial species: implications for the classification of mycobacteria. *J Bacteriol.* 2002;184:1078–1088. PubMed PMID: 11807068.
37. Domenech P, Menendez MC, Garcia MJ. Restriction fragment length polymorphisms of 16S rRNA genes in the differentiation of fast-growing mycobacterial species. *FEMS Microbiol Lett.* 1994;116:19–24. PubMed PMID: 7907567.
38. Brown BA, Springer B, Steingrube VA, Wilson RW, Pfyffer GE, Garcia MJ, Menendez MC, Rodriguez-Salgado B, Jost KC, Chiu SH, Onyi GO, Bottger EC, Wallace RJ. *Mycobacterium wolinskyi* sp. nov. and *Mycobacterium goodii* sp. nov., two new rapidly growing species related to *Mycobacterium smegmatis* and associated with human wound infections: a cooperative study from the International Working Group on Mycobacterial Taxonomy. *Int J Syst Bacteriol.* 1999;49(Pt 4):1493–1511. PubMed PMID: 10555330.
39. Jimenez MS, Campos-Herrero MI, Garcia D, Luquin M, Herrera L, Garcia MJ. *Mycobacterium canariasense* sp. nov. *Int J Syst Evol Microbiol.* 2004;54:1729–1734. PubMed PMID: 15388736.
40. Lawn SD, Bekker LG, Miller RF. Immune reconstitution disease associated with mycobacterial infections in HIV-infected individuals receiving antiretrovirals. *Lancet Infect Dis.* 2005;5:361–373. PubMed PMID: 15919622.
41. Horsburgh CR. The pathophysiology of disseminated *Mycobacterium avium* complex disease in AIDS. *J Infect Dis.* 1999;179Suppl 3S461–S465. PubMed PMID: 10099120.
42. Nunn P, Williams B, Floyd K, Dye C, Elzinga G, Raviglione M. Tuberculosis control in the era of HIV. *Nat Rev Immunol.* 2005;5:819–826. PubMed PMID: 16200083.
43. Ahmed N, Hasnain SE. Genomics of *Mycobacterium tuberculosis*: old threats & new trends. *Indian J Med Res.* 2004;120:207–212. PubMed PMID: 15520478.
44. Cole ST, Brosch R, Parkhill J, Garnier T, Churcher C, Harris D, Gordon SV, Eiglmeier K, Gas S, Barry CE, Tekaia F, Badcock K, Basham D, Brown D, Chillingworth T, Connor R, Davies R, Devlin K, Feltwell T,

- Gentles S, Hamlin N, Holroyd S, Hornsby T, Jagels K, Krogh A, McLean J, Moule S, Murphy L, Oliver K, Osborne J, Quail MA, Rajandream MA, Rogers J, Rutter S, Seeger K, Skelton J, Squares R, Squares S, Sulston JE, Taylor K, Whitehead S, Barrell BG. Deciphering the biology of *Mycobacterium tuberculosis* from the complete genome sequence. *Nature*. 1998;393:537–544. PubMed PMID: 9634230.
45. Camus JC, Pryor MJ, Medigue C, Cole ST. Re-annotation of the genome sequence of *Mycobacterium tuberculosis* H37Rv. *Microbiology*. 2002;148:2967–2973. PubMed PMID: 12368430.
  46. Fleischmann RD, Alland D, Eisen JA, Carpenter L, White O, Peterson J, DeBoy R, Dodson R, Gwinn M, Haft D, Hickey E, Kolonay JF, Nelson WC, Umayam LA, Ermolaeva M, Salzberg SL, Delcher A, Utterback T, Weidman J, Khouri H, Gill J, Mikula A, Bishai W, Jacobs Jr WR, Venter JC, Fraser CM. Whole-genome comparison of *Mycobacterium tuberculosis* clinical and laboratory strains. *J Bacteriol*. 2002;184:5479–5490. PubMed PMID: 12218036.
  47. Beggs ML, Eisenach KD, Cave MD. Mapping of IS6110 insertion sites in two epidemic strains of *Mycobacterium tuberculosis*. *J Clin Microbiol*. 2000;38:2923–2928. PubMed PMID: 10921952.
  48. Garnier T, Eiglmeier K, Camus JC, Medina N, Mansoor H, Pryor M, Duthoy S, Grondin S, Lacroix C, Monsempe C, Simon S, Harris B, Atkin R, Doggett J, Mayes R, Keating L, Wheeler PR, Parkhill J, Barrell BG, Cole ST, Gordon SV, Hewinson RG. The complete genome sequence of *Mycobacterium bovis*. *Proc Natl Acad Sci U S A*. 2003;100:7877–7882. PubMed PMID: 12788972.
  49. Gordon SV, Brosch R, Billault A, Garnier T, Eiglmeier K, Cole ST. Identification of variable regions in the genomes of tubercle bacilli using bacterial artificial chromosome arrays. *Mol Microbiol*. 1999;32:643–655. PubMed PMID: 10320585.
  50. Brodin P, Eiglmeier K, Marmiesse M, Billault A, Garnier T, Niemann S, Cole ST, Brosch R. Bacterial artificial chromosome-based comparative genomic analysis identifies *Mycobacterium microti* as a natural ESAT-6 deletion mutant. *Infect Immun*. 2002;70:5568–5578. PubMed PMID: 12228284.
  51. Araoz R, Honore N, Cho S, Kim JP, Cho SN, Monot M, Demangel C, Brennan PJ, Cole ST. Antigen discovery: a postgenomic approach to leprosy diagnosis. *Infect Immun*. 2006;74:175–182. PubMed PMID: 16368971.
  52. Stinear TP, Mve-Obiang A, Small PL, Frigui W, Pryor MJ, Brosch R, Jenkin GA, Johnson PD, Davies JK, Lee RE, Adusumilli S, Garnier T, Haydock SF, Leadlay PF, Cole ST. Giant plasmid-encoded polyketide synthases produce the macrolide toxin of *Mycobacterium ulcerans*. *Proc Natl Acad Sci U S A*. 2004;101:1345–1349. PubMed PMID: 14736915.
  53. Li L, Bannantine JP, Zhang Q, Amonsin A, May BJ, Alt D, Banerji N, Kanjilal S, Kapur V. The complete genome sequence of *Mycobacterium avium* subspecies paratuberculosis. *Proc Natl Acad Sci U S A*. 2005;102:12344–12349. PubMed PMID: 16116077.
  54. Snapper SB, Melton RE, Mustafa S, Kieser T, Jacobs WR. Isolation and characterization of efficient plasmid transformation mutants of *Mycobacterium smegmatis*. *Mol Microbiol*. 1990;4:1911–1919. PubMed PMID: 2082148.
  55. Khan AA, Kim SJ, Paine DD, Cerniglia CE. Classification of a polycyclic aromatic hydrocarbon-metabolizing bacterium, *Mycobacterium* sp. strain PYR-1, as *Mycobacterium vanbaalenii* sp. nov. *Int J Syst Evol Microbiol*. 2002;52:1997–2002. PubMed PMID: 12508859.
  56. Miller CD, Hall K, Liang YN, Nieman K, Sorensen D, Issa B, Anderson AJ, Sims RC. Isolation and characterization of polycyclic aromatic hydrocarbon-degrading *Mycobacterium* isolates from soil. *Microb Ecol*. 2004;48:230–238. PubMed PMID: 15107954.
  57. Brosch R, Pym AS, Gordon SV, Cole ST. The evolution of mycobacterial pathogenicity: clues from comparative genomics. *Trends Microbiol*. 2001;9:452–458. PubMed PMID: 11553458.
  58. Lopez B, Aguilar D, Orozco H, Burger M, Espitia C, Ritacco V, Barrera L, Kremer K, Hernandez-Pando R, Huygen K, van Soolingen D. A marked difference in pathogenesis and immune response induced by different *Mycobacterium tuberculosis* genotypes. *Clin Exp Immunol*. 2003;133:30–37. PubMed PMID: 12823275.

59. Schnappinger D, Ehrt S, Voskuil MI, Liu Y, Mangan JA, Monahan IM, Dolganov G, Efron B, Butcher PD, Nathan C, Schoolnik GK. Transcriptional Adaptation of *Mycobacterium tuberculosis* within Macrophages: Insights into the Phagosomal Environment. *J Exp Med*. 2003;198:693–704. PubMed PMID: 12953091.
60. Bacon J, James BW, Wernisch L, Williams A, Morley KA, Hatch GJ, Mangan JA, Hinds J, Stoker NG, Butcher PD, Marsh PD. The influence of reduced oxygen availability on pathogenicity and gene expression in *Mycobacterium tuberculosis*. *Tuberculosis (Edinb)*. 2004;84:205–217. PubMed PMID: 15207490.
61. Kendall SL, Rison SC, Movahedzadeh F, Frita R, Stoker NG. What do microarrays really tell us about *M. tuberculosis*? *Trends Microbiol*. 2004;12:537–544. PubMed PMID: 15539113.
62. Muttucumaru DG, Roberts G, Hinds J, Stabler RA, Parish T. Gene expression profile of *Mycobacterium tuberculosis* in a non-replicating state. *Tuberculosis (Edinb)*. 2004;84:239–246. PubMed PMID: 15207493.
63. Voskuil MI, Visconti KC, Schoolnik GK. *Mycobacterium tuberculosis* gene expression during adaptation to stationary phase and low-oxygen dormancy. *Tuberculosis (Edinb)*. 2004;84:218–227. PubMed PMID: 15207491.
64. Bigi F, Garcia-Pelayo MC, Nunez-Garcia J, Peralta A, Caimi KC, Golby P, Hinds J, Cataldi A, Gordon SV, Romano MI. Identification of genetic markers for *Mycobacterium pinnipedii* through genome analysis. *FEMS Microbiol Lett*. 2005;248:147–152. PubMed PMID: 15979818.
65. Garcia-Pelayo MC, Caimi KC, Inwald JK, Hinds J, Bigi F, Romano MI, van Soolingen D, Hewinson RG, Cataldi A, Gordon SV. Microarray analysis of *Mycobacterium microti* reveals deletion of genes encoding PE-PPE proteins and ESAT-6 family antigens. *Tuberculosis (Edinb)*. 2004;84:159–166. PubMed PMID: 15207485.
66. Bull TJ, Sidi-Boumedine K, McMinn EJ, Stevenson K, Pickup R, Hermon-Taylor J. *Mycobacterial* interspersed repetitive units (MIRU) differentiate *Mycobacterium avium* subspecies paratuberculosis from other species of the *Mycobacterium avium* complex. *Mol Cell Probes*. 2003;17:157–164. PubMed PMID: 12944117.
67. Chanchaem W, Palittapongarnpim P. A variable number of tandem repeats result in polymorphic alpha - isopropylmalate synthase in *Mycobacterium tuberculosis*. *Tuberculosis (Edinb)*. 2002;82:1–6. PubMed PMID: 11914056.
68. Frothingham R, Meeker-O'Connell WA. Genetic diversity in the *Mycobacterium tuberculosis* complex based on variable numbers of tandem DNA repeats. *Microbiology*. 1998;144(Pt 5):1189–1196. PubMed PMID: 9611793.
69. Mazars E, Lesjean S, Banuls AL, Gilbert M, Vincent V, Gicquel B, Tibayrenc M, Locht C, Supply P. High-resolution minisatellite-based typing as a portable approach to global analysis of *Mycobacterium tuberculosis* molecular epidemiology. *Proc Natl Acad Sci U S A*. 2001;98:1901–1906. PubMed PMID: 11172048.
70. Romano MI, Amadio A, Bigi F, Klepp L, Etchehoury I, Llana MN, Morsella C, Paolicchi F, Pavlik I, Bartos M, Leao SC, Cataldi A. Further analysis of VNTR and MIRU in the genome of *Mycobacterium avium* complex, and application to molecular epidemiology of isolates from South America. *Vet Microbiol*. 2005;110:221–237. PubMed PMID: 16171956.
71. Roring S, Scott A, Brittain D, Walker I, Hewinson G, Neill S, Skuce R. Development of variable-number tandem repeat typing of *Mycobacterium bovis*: comparison of results with those obtained by using existing exact tandem repeats and spoligotyping. *J Clin Microbiol*. 2002;40:2126–2133. PubMed PMID: 12037076.
72. Savine E, Warren RM, van der Spuy GD, Beyers N, van Helden PD, Locht C, Supply P. Stability of variable-number tandem repeats of mycobacterial interspersed repetitive units from 12 loci in serial isolates of *Mycobacterium tuberculosis*. *J Clin Microbiol*. 2002;40:4561–4566. PubMed PMID: 12454152.
73. Sola C, Filliol I, Legrand E, Lesjean S, Locht C, Supply P, Rastogi N. Genotyping of the *Mycobacterium tuberculosis* complex using MIRUs: association with VNTR and spoligotyping for molecular epidemiology and evolutionary genetics. *Infect Genet Evol*. 2003;3:125–133. PubMed PMID: 12809807.
74. Supply P, Magdalena J, Himpens S, Locht C. Identification of novel intergenic repetitive units in a mycobacterial two-component system operon. *Mol Microbiol*. 1997;26:991–1003. PubMed PMID: 9426136.

75. Supply P, Mazars E, Lesjean S, Vincent V, Gicquel B, Locht C. Variable human minisatellite-like regions in the *Mycobacterium tuberculosis* genome. *Mol Microbiol.* 2000;36:762–771. PubMed PMID: 10844663.
76. van Embden JD, Cave MD, Crawford JT, Dale JW, Eisenach KD, Gicquel B, Hermans P, Martin C, McAdam R, Shinnick TM. Strain identification of *Mycobacterium tuberculosis* by DNA fingerprinting: recommendations for a standardized methodology. *J Clin Microbiol.* 1993;31:406–409. PubMed PMID: 8381814.
77. Romano MI, Alito A, Fisanotti JC, Bigi F, Kantor I, Cicuta ME, Cataldi A. Comparison of different genetic markers for molecular epidemiology of bovine tuberculosis. *Vet Microbiol.* 1996;50:59–71. PubMed PMID: 8810008.
78. Kamerbeek J, Schouls L, Kolk A, van Agterveld M, van Soolingen D, Kuijper S, Bunschoten A, Molhuizen H, Shaw R, Goyal M, van Embden J. Simultaneous detection and strain differentiation of *Mycobacterium tuberculosis* for diagnosis and epidemiology. *J Clin Microbiol.* 1997;35:907–914. PubMed PMID: 9157152.
79. Caimi K, Romano MI, Alito A, Zumarraga M, Bigi F, Cataldi A. Sequence analysis of the direct repeat region in *Mycobacterium bovis*. *J Clin Microbiol.* 2001;39:1067–1072. PubMed PMID: 11230428.
80. Cole ST. Comparative and functional genomics of the *Mycobacterium tuberculosis* complex. *Microbiology.* 2002;148:2919–2928. PubMed PMID: 12368425.
81. Behr MA, Wilson MA, Gill WP, Salamon H, Schoolnik GK, Rane S, Small PM. Comparative genomics of BCG vaccines by whole-genome DNA microarray. *Science.* 1999;284:1520–1523. PubMed PMID: 10348738.
82. Brosch R, Gordon SV, Marmiesse M, Brodin P, Buchrieser C, Eiglmeier K, Garnier T, Gutierrez C, Hewinson G, Kremer K, Parsons LM, Pym AS, Samper S, van Soolingen D, Cole ST. A new evolutionary scenario for the *Mycobacterium tuberculosis* complex. *Proc Natl Acad Sci U S A.* 2002;99:3684–3689. PubMed PMID: 11891304.
83. Amadio A, Romano MI, Bigi F, Etchechoury I, Kubica T, Niemann S, Cataldi A, Caimi K. Identification and characterization of genomic variations between *Mycobacterium bovis* and *M. tuberculosis* H37Rv. *J Clin Microbiol.* 2005;43:2481–2484. PubMed PMID: 15872289.
84. Sonnhammer EL, Durbin R. A workbench for large-scale sequence homology analysis. *Comput Appl Biosci.* 1994;10:301–307. PubMed PMID: 7922687.
85. Parsons LM, Brosch R, Cole ST, Somoskovi A, Loder A, Bretzel G, Van Soolingen D, Hale YM, Salfinger M. Rapid and simple approach for identification of *Mycobacterium tuberculosis* complex isolates by PCR-based genomic deletion analysis. *J Clin Microbiol.* 2002;40:2339–2345. PubMed PMID: 12089245.
86. Semret M, Alexander DC, Turenne CY, de Haas P, Overduin P, van Soolingen D, Cousins D, Behr MA. Genomic polymorphisms for *Mycobacterium avium* subsp. *paratuberculosis* diagnostics. *J Clin Microbiol.* 2005;43:3704–3712. PubMed PMID: 16081899.
87. Semret M, Zhai G, Mostowy S, Cleto C, Alexander D, Cangelosi G, Cousins D, Collins DM, van Soolingen D, Behr MA. Extensive genomic polymorphism within *Mycobacterium avium*. *J Bacteriol.* 2004;186:6332–6334. PubMed PMID: 15342607.
88. Paustian ML, Kapur V, Bannantine JP. Comparative genomic hybridizations reveal genetic regions within the *Mycobacterium avium* complex that are divergent from *Mycobacterium avium* subsp. *paratuberculosis* isolates. *J Bacteriol.* 2005;187:2406–2415. PubMed PMID: 15774884.
89. Alm EJ, Huang KH, Price MN, Koche RP, Keller K, Dubchak IL, Arkin AP. The MicrobesOnline Web site for comparative genomics. *Genome Res.* 2005;15:1015–1022. PubMed PMID: 15998914.



## Chapter B04. Transgenic Vectors: *Anopheles* and *Aedes*

Márcia Aparecida Sperança, PhD,<sup>1</sup> Paulo Eduardo Martins Ribolla,<sup>2</sup> and Margareth Lara Capurro<sup>3</sup>

Created: December 6, 2006; Updated: May 11, 2007.

### Vector Control Is Still the Most Effective Way to Manage Arthropod-borne Infectious Diseases

Worldwide, a variety of diseases caused by viruses, protozoans, and helminths are transmitted to humans by mosquitoes and are responsible for more than 2 million deaths per year. Malaria, transmitted by anopheline mosquitoes, leishmaniasis by flebotomines, trypanosomiasis by triatomines, and dengue and yellow fever, transmitted mainly by *Aedes aegypti*, are examples of these arthropod-borne diseases that impose major economic and social burdens on affected populations.

Historically, two methods have been used to control human and veterinary arthropod-borne diseases: development and use of drugs, and vector control and vaccine only in the case of yellow fever. Despite many efforts, effective vaccines against malaria, dengue, leishmaniasis, and similar types of diseases are not available. Recently, pathogens are becoming resistant to drugs, and elimination of the vector has been problematic because of the demise of public health programs for vector control and the development of pesticide resistance by vector populations.

Methods to control malaria and dengue are still focused on mosquito control. This approach was responsible for malaria elimination from a variety of countries, and today, the use of insecticide-impregnated bed nets in Africa is proving to be an important strategy to decrease malaria transmission (1). In addition, insecticides have been the strategy of choice to control *Ae. aegypti* proliferation.

However, socioeconomic issues including dramatic population growth, increased urbanization without planning, lack of sanitation, deforestation, rapid dissemination of pathogens and vectors by airplane travelers, and occurrence of drug-resistant parasites have been contributing to the difficulties in controlling these important vector-borne diseases. In recent years, some arthropod-transmitted pathogens that were previously controlled to some extent are reappearing and have also been found in new geographic areas. The morbidity and mortality associated with these diseases are overwhelming: billions of people are now at risk for infection.

Thus, the increasing public health importance of these pathogens, the failure of conventional approaches to control them, and the lack of alternatives in the face of the vectors' pesticide resistance and drug-resistant pathogens demonstrate the need for novel and efficacious control strategies for these diseases.

One alternative pathway to combat malaria and dengue transmission was proposed about 15 years ago. This relatively new approach consists of genetic manipulation of mosquito vectors to impair its ability to be infected by and to transmit *Plasmodium* or Dengue virus. The steps necessary to obtain a transgenic mosquito and the ethical aspects involved in its release in nature will be discussed in this chapter.

## Transgenic Mosquitoes as an Alternative Solution to Overcome Vector Control Problems

More than 40 years ago, Knippling (2, 3) had an idea to use genetically modified mosquitoes to combat the disease. Knippling (2, 3) proposed the field release of sterile males to decrease arthropod-borne pathogen transmission or insect population. This strategy, named Sterile Insect Technique (SIT), showed positive results with the elimination of the New World screwworm, *Cochliomyia hominivorax* from the South in the USA, from Mexico, and from Central America (4). Sterile males are produced by irradiation and released in the field to compete with wild ones. Females that have been mated with irradiated mosquitoes will not produce descendents. This methodology was tested with *Anopheles* in the laboratory and showed promising results (5), but field experiments seemed to be less efficient (6). Limitations included the separation of male insects by physical methods, which are not easy for most vector species, including anopheline mosquitoes (7); poor mating competitiveness of the sterile male compared with the wild insect (6); occurrence of emerging species of the malarial vector *Anopheles*, with different ecological preferences and significant prezygotic reproductive isolation (8).

The approach to generate a transgenic mosquito unable to transmit pathogens emerges as an alternative strategy because SIT has some limitations, as discussed above. There are several methodological and ethical aspects that have to be discussed regarding this strategy. How to generate efficient mosquitoes that are completely unable to transmit certain pathogens and how to spread and control this phenotype in natural populations will be discussed here. At the end of the chapter, the ethical aspects of this issue will also be discussed.

### Transgenesis Tools (Transposons, Viruses, etc.)

To produce a transgenic vector refractory to a pathogen, several technological features are necessary: identification of efficient transformation and driver transfection systems, finding of suitable reporter genes, characterization of efficient tissue- and developmental-specific promoters, discovery of appropriate anti-parasite effector genes, and development of efficient microinjection techniques.

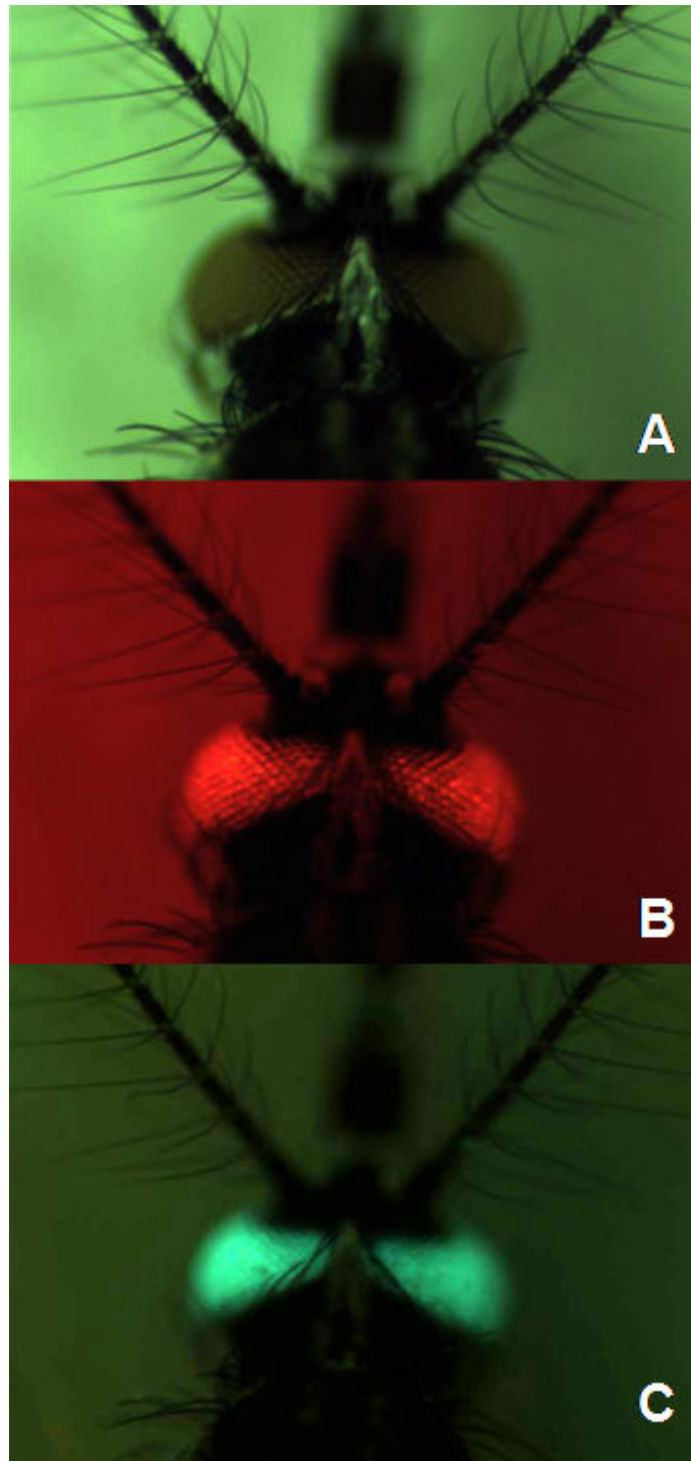
The first question that emerges when we think of transgenic mosquitoes is an efficient DNA vector to introduce a foreign gene into the mosquito. In the last decades, different stable and transient vector transformation systems have been developed, such as transposable elements and endosymbionts. Characteristics and application of each of these tools will be discussed in more detail.

Virus-transfection systems, which enable transient expression of transgenes, have been used when relatively large numbers of molecules need to be screened to characterize effector anti-parasitic molecules and to analyze whether a gene is involved in decreasing vector competence (9).

To produce a transgenic mosquito, another problem that has to be solved is how to introduce heterologous genes into the mosquito germ line. Until now, the most efficient method to stably transform a mosquito is the injection of the transgene construct into early embryos, a technique imported from *Drosophila* and that has low efficiency and is time consuming.

Another requirement is to identify the transformed insects. Identification of a mosquito expressing a transgene was performed with the transient Sindbis virus (SIN) system, through the use of the Green Fluorescent Protein (GFP) from *Aequorea victoria* as a marker (10). Earlier attempts to construct a stable transgenic mosquito were set up with insecticide and antibiotic resistance genes as markers (11-13). This initial strategy was based on previous research performed in bacteria, where a restrictive selection enabled only transformed organisms to survive. After that, the *cinnabar* gene encoding the kynurenine hydroxylase of *Drosophila* was found to complement a white-eyed mutation in an old laboratory strain of *Ae. aegypti* and has been used largely as a marker (14). The problem with this marker is the difficulty of obtaining a mutant mosquito and also that the





**Figure 1.** Natural and fluorescent images of the head of a female *Ae. aegypti* expressing dual marker genes. In this experiment, the mosquito was transformed with two different fluorescent genes under control of an eye-specific promoter. A, incident light; B, DsRed filter; C, EGFP filter. (This figure was the kind contribution of Antony A. James, PhD.)

phenotype can be observed only in adult stages of the transformed mosquitoes. Using GFP, fluorescence of transformed individuals is observed even in the early larval stage, saving the time and effort of rearing the nontransgenic insects to adults. GFP constructs driven by the polyubiquitin (15) and PAX eye-specific promoters (16) were first used on the *Ae. aegypti* germ line transformation (17) and, subsequently, have been used to transform the germ line of other species of mosquitoes (18, 19) (Figure 1).

Once a transgenic mosquito is obtained, the transgene needs to efficiently spread in the natural population. It is unlikely that reliance on Mendelian inheritance alone could accomplish this, because it would require an enormous number of mosquitoes. Thus, drive mechanisms—such as mobile DNA elements or other shuttles of genetic material that are spread in a non-Mendelian manner (e.g., symbiotic bacteria)—have been the tools of choice to produce transgenic mosquitoes (20, 21).

## Virus

Virus infection of the mosquito started with experiments on culture cells. Positive infection results were first obtained with SIN in a variety of eukaryotic cells (22) and rapidly became a tool to express different proteins in eukaryotic models. SIN from genus *Alphavirus* and the Togaviridae family has a positive-sense RNA genome with 11,703 (nt) in length and nonstructural and structural proteins encoded from two different promoters. The nonstructural proteins, including RNA-dependent RNA polymerase (RDRP), are encoded by and translated from the 5' two-thirds of the genome. SIN structural proteins are encoded in the 3' one-third of the virus RNA and are translated from a subgenomic mRNA transcribed intracellularly by the RDRP from an internal promoter (23). The SIN natural cycle involves birds and *Culex* mosquitoes (24), being also able to infect *Aedes* and *Anopheles* mosquitoes and a broad range of vertebrate animal cells (25). Use of SIN as a vector was possible after a complete cDNA copy of the genome was produced by Rice *et al.* (26). This cDNA can yield infectious viral RNA by *in vitro* transcription. Two types of expression vector derived from the cDNA SIN clone have been developed: the double-subgenomic SIN (dsSIN) expression vector TE/3'J and the SIN replicon (9). TE/3'J has a duplicated internal promoter from which transgenes can be expressed, and the SIN replicon had its structural protein genes deleted to be replaced by up to 6,000 bp of heterologous genetic material. Infective virions are produced directly by injecting *in vitro*-transcribed RNA of TE/3'J into competent eukaryotic cells, whereas for producing the SIN replicon, it is necessary to co-infect cells with a helper virus expressing the viral structural genes.

Thus, insect transformation by SIN expression systems emerged as an alternative tool that has been used in the laboratory to test genes that impaired pathogen transmission. Most of the experiments in mosquitoes have been done with *Aedes aegypti* and aimed to demonstrate the efficacy of effector molecules, such as antisense RNAs, for their ability to block dengue, yellow fever, and LaCross virus transmission (27, 28). The same strategy has been used to characterize single-chain recombinant antibodies as target molecules to block malaria transmission using the experimental model of avian malaria *Plasmodium gallinaceum* infecting *Ae. aegypti* mosquitoes (29). SIN expression systems have been also used to silence expression of endogenous genes for rapid characterization of vector molecules *in vivo* (30). Thus, SIN expression systems are contributing to the better understanding of the biology of important vectors of human disease and to characterize important target molecules potentially capable to block pathogen transmission.

## Transposons

Transposons are mobile genetic elements found in animals as well as in plants, existing in many families and superfamilies according to its genomic organization. The transposons used to genetically engineer the DNA elements of insects are inserted into chromosomes by a transposase enzyme encoded in the transposon itself. Others transposons replicate by reverse transcriptase, such as retroviruses, but those elements are not used in current genetic modifications of mosquitoes.

Transposable elements (TEs), or mobile genetic elements, are integral components of the eukaryotic genomes. Because they have the ability to replicate and spread in the genome primarily as "selfish" genetic units, TEs tend to constitute significant proportions of the genome. For example, at least 45% of the human genome is TE-derived sequences (31). Recent evidence suggests that the "selfish" property may have enabled TEs to provide the genome with potent agents to generate tremendous genetic and genomic plasticity (32). For example, TEs may

have reshaped the human genome by ectopic rearrangements, by creating new genes, and by modifying and shuffling existing genes (33).

The idea of using TEs to genetically transform mosquitoes came from the modification of the naturally occurring transposable P-element from *Drosophila melanogaster* in genetic and developmental studies of these insects. Initially, the P-element was tested in *Ae. aegypti*, but integration frequency was extremely low and did not result from the action of the P-element transposase (13). After that, other DNA transposable elements were searched and tested in mosquitoes, and initially, two elements were able to stably transform embryos of *Ae. aegypti*: *Hermes* from the *hAT* family of transposons derived from the housefly *Musca domestica* (34), and the mariner element MOSI from *Drosophila mauritiana* (35). In these studies, the transposable elements were linked to the reporter *cinnabar* gene to complement the white eye color of *Ae. aegypti*. The mechanism of movement of these elements is referred to as cut-and-paste through its inverted repeat-specific sequence (36). However, sometimes co-integration of plasmid sequences flanking the element's inverted repeat was observed by mechanisms that are not well understood (37).

Two additional transposition elements have been used to produce transgenic mosquitoes: *piggyback* isolated initially from *Trichoplusia ni*, and *Minos* from *Drosophila hydei*. *piggyBac* vectors transform efficiently *Ae. aegypti* (19), *Anopheles stephensi* (38), *Anopheles albimanus* (39), and *Anopheles gambiae* (40). The *Minos* element has also been used to transform several insects including the mosquito *A. stephensi* (41), and its transposition rate is similar to that found for other elements. An advantage for *Minos* is that non-canonical integration events have not been found in mosquitoes as reported for *mariner*, *Hermes*, and *piggyBac* (37). Additional research is needed to understand the behavior of these elements to verify the viability of using *Minos* transgenic insects in fieldwork.

## Endosymbionts

An alternative potential mechanism to introduce an effector gene into a vector population involves the use of the intracellular symbiotic bacterium *Wolbachia*, which infects a variety of insect tissues (42). An interesting characteristic of *Wolbachia* is its cytoplasmic incompatibility. Only when a male infected with the bacteria mates with an infected female are the eggs viable, and the bacteria are transovarially transmitted to the next generation. This effect leads to increased reproductive success and consequently to a fast dissemination of a transgene, because genetically modified *Wolbachia* could be used to transduce mosquitoes with parasite transmission-blocking genes (43). This interesting methodology is still not available for anopheline mosquitoes because of the lack of symbiotic bacteria that could be potentially used in these insects. Also, it is not yet possible to genetically transform *Wolbachia* (44).

## Effector Molecules and Promoters

A remarkable number of effector molecules that are capable of blocking pathogen transmission in mosquitoes have been investigated. For example, several mechanisms have been carried out for interfering with different developing stages of malaria parasites in mosquitoes (45). In an attempt to target sporozoites, two groups of peptides have been tested against *Plasmodium* ssp. showing interference on transmission. The first group corresponds to competitor peptides that bind to salivary gland receptors blocking sporozoite invasion (46, 47). The second group consists of peptides, which might have parasitocidal effects in mosquitoes such as cecropins, magainins, and defensins, or hybrid peptides such as the cecropin-like peptides and scorpine (48, 49). Most of these natural peptides, at high concentrations, are capable of reducing the levels of sporogonic stages of malaria parasites. Arrighi and collaborators (50), using natural and synthetic peptides, showed that three short novel hybrid peptides can interfere with oocyst development. The authors also demonstrated that hybrid peptides consisting of random coils and turns were particularly active against the sporogonic stages of *P. berghei* and *P. yoelli nigeriensis*.

In addition, using the Sindbis expression system, several heterologous proteins and antisense RNAs with anti-parasite properties have been investigated in *Ae. aegypti* (9, 51). Single-chain antibody fragments (ScFv), composed of fused heavy-chain and light-chain variable regions of monoclonal antibodies expressed as the product of a single gene, have been used to block malaria parasite development. For example, a ScFv-specific to the malarial circumsporozoite protein (CSP) reduced by 99% the number of *P. gallinaceum* sporozoites in salivary glands of *Ae. aegypti* (29). Furthermore, Olson and collaborators (27) in 1996 showed inhibition of Dengue 2 (DEN2) in mosquitoes *Ae. aegypti* transiently expressing an antisense RNA corresponding to a fragment of 567 bases of the pre-membrane protein from DEN2. Using the same strategy and the use of different antisense RNAs, the same research group demonstrated the inhibition of DEN replication in mosquito and mosquito cells by a mechanism similar to the post-transcriptional silencing or RNA interference (RNAi) (52-54). Likewise, other potential antipathogen molecules (55) can be considered such as phospholipase PLA2, the venom of a bee, which reduced the number of *P. berghei* oocysts in *A. stephensi* (56), and serine proteases and phenoloxidases that are critical enzymes in the malaria melanotic encapsulation pathway.

Expression of a transgene in the mosquito also requires the use of an appropriate promoter. The ideal promoter should enable the expression of a foreign gene in the tissue of interest and at a proper time. According to the pathogen life cycle inside the arthropod vector, the foreign gene would be expressed in salivary glands, hemolymph, midgut, Malpighian tubules, or thoracic musculature. For example, salivary gland promoters for the *maltase-like I* (Mali) (57) and *Apyrase* (Apy) (58) genes have been able to direct expression of recombinant firefly luciferase in Hermes-transformed *Ae. aegypti* in the tissues and at the time of expression of the corresponding endogenous genes (59). However, expression of the transgene driven by these promoters was too weak to be used in the construction of transgenic mosquitoes. *Ae. aegypti* vitellogenin (60) and carboxypeptidase (61) promoters have been used to drive strong, blood-inducible expression of the antibacterial peptide defensin in the hemolymph of *Ae. aegypti* and anti-malarial peptides in *A. stephensi*.

As will be described below, research taking advantage of the recent developments in genomics and bioinformatics promises to aid in the identification of promoters that express the transgene at the developmental site of the pathogen and with a minimal cost to the mosquito reproduction.

## Bioinformatics as a Tool for the Transgenic Vector Construction

To develop novel, arthropod-borne disease control strategies, different tools of molecular biology and genomics are being applied to investigate vector-parasite interactions. Characterizations of new mosquito promoter sequences specifically regulated in different tissues and at the proper time are of particular interest to the construction of transgenic mosquitoes refractory to pathogen transmission. Furthermore, discovery of parasite molecules involved in its infection and development inside the mosquito vector can also give new insights to the production of anti-parasite transgenes. Both types of information have been only recently obtained by different research groups working on diverse vector-parasites models through the use of genomics and microarray chip technologies analyzed by sophisticated bioinformatic tools.

Recently, the complete genome of *Anopheles gambiae*, the most important human malaria vector, has been sequenced by an international consortium (62). In parallel, the first mosquito cDNA microarray was constructed with four thousand cDNA Expressed Sequence Tags (ESTs) sequenced from a library prepared from a hemocyte-like cultured cell line (63, 64). These arrays were used to detect genes that are up-regulated in the mosquito during infection with parasites and bacteria (64). These authors showed specific expression of the mosquito genes encoding isocitrate dehydrogenase, dsRNA binding RNase 3 and a mitochondrial phosphate carrier, when RNA source was obtained from mosquitoes infected with malaria but not bacteria. Genomic analysis of these transcripts can in turn contribute to the characterization of promoter sequences that are activated only in the presence of malaria parasites, offering a tool to produce transgenic mosquitoes specifically refractory to malaria transmission. Using the same cDNA microarray, Kumar *et al.* (65) verified that the most prominent cluster of

gene expression differences between parasite-susceptible and -refractory mosquitoes was related to genes coding proteins involved in the reactive oxygen species production. These results, associated with morphological and physiological approaches, enabled the authors to implicate reactive oxygen species as one important factor contributing to the *Plasmodium* melanotic encapsulation phenotype of a malaria-resistant mosquito strain. Thus, these results also offer other possibilities to construct pathogen-refractory transgenic mosquitoes by using transgenes coding proteins involved in the production of reactive oxygen species. Obviously, this all-new strategy has to be largely studied, because for example, in this last case, reactive oxygen species can be very harmful to the mosquito.

Moreover, functional genomic studies have enabled identification of diverse *An. gambiae* promoters specifically expressed in different mosquito tissues and at different times in response to several biological conditions, including blood feeding and infection by *Plasmodium*. Vlachou *et al.* (66) have shown that 7% of the *An. gambiae* midgut transcripts are differentially expressed during *P. berghei* oocyst invasion. The expression profile of *An. gambiae* females was assessed by microarray analysis in response to blood-feeding, resulting in the identification of three groups of developmentally expressed genes (early, middle, and late). These were classified according to physiological responses: blood feeding, blood digestion, peritrophic matrix formation, egg development, and immunity (67).

Genomics and post-genomics studies on *An. gambiae* mosquitoes have taken advantage of the availability of new EST sequences and EST cDNA libraries constructed from pooled mosquito developmental stages (NAP1) and adult heads (NAH). Furthermore, a new cDNA microarray platform with 20,000 ESTs was built. To better analyze the results of gene expression using these libraries and microarrays, a bioinformatic tool named AnoEST has been developed (68).

The genome of the mosquito, *Aedes aegypti*, the major vector of dengue viruses, is currently being sequenced (69) (transcriptome analysis). Using microarray technology, *Ae. aegypti* midgut differential expression profile in response to blood feeding over a time course of 72 h has also been performed (70) and is available at the website <http://www.etox.ucr.edu/gill/home.htm>. More recently, the characterization of transcripts expressed in the fat body of *Ae. aegypti* at 24 hours post-blood meal was published. These data also provide a basic tool for understanding the processes occurring in the fat body and could identify putative new genes whose promoters can be used to specifically express transgenes in this organ of *Ae. aegypti*(71).

Simultaneous microarray-based transcription analysis of the murine malaria model *An. stephensi* and *P. berghei* correspond to an interesting and elegant approach to characterize molecules involved in the parasite–arthropod vector interactions. By this methodology, numerous novel genes were identified during malaria midgut invasion, oocyst differentiation, and ookinete development from both parasite and mosquito origin (72, 73). One novel molecule identified with this methodology corresponded to the gene encoding a Plasmodium surface protein with a von Willebrand factor A-like adhesive domain named WARP that is expressed only in ookinetes and early oocysts. It was shown that the anti-WARP polyclonal antibody strongly inhibits (70-92%) *Plasmodium* development in the mosquito, making it a candidate antigen for transmission-blocking vaccines or for construction of refractory mosquitoes expressing recombinant antibodies raised against this protein. The same approach enabled identification of transcription profiles of both organisms presenting temporal correlation between processes such as *Plasmodium* invasion of the midgut epithelium, *Anopheles* immune responses to *Plasmodium* infection, and apoptosis and expulsion of invaded midgut cells from the epithelium (74). All of these available data can be used to help in the construction of a malaria-refractory mosquito. Similar transcriptome experiments envisioning identification of specifically expressing transcripts from *Ae. aegypti* in response to dengue infection are in progress (personal communication of Margareth Lara Capurro).

Bioinformatic analyses of the *An. gambiae* genome have also enabled identification of sex-specific genes that can be used to construct transgenic lines expressing sex-controlled genes, for example, to be used in programs based on the sterile insect technique as discussed above (75).

In addition to genomics, gene expression, and DNA microarray technology, pathogen arthropod vectors have been investigated by proteomics, because many types of information cannot be obtained from genes alone. For example, exon–intron structure is very difficult to predict by bioinformatics; mRNA is subject to post-transcriptional regulation as alternative splicing and polyadenylation or can be regulated at the protein translation level. Proteins can also be regulated by proteolysis and compartmentalization. Thus, with proteomic approaches, several biochemical and physiological regulation features can be identified, such as signal peptides, which can be used as tools in the construction of pathogen-refractory mosquitoes. Using mass spectrometry as strategy to characterize peptides from *An. gambiae*, it was possible to correct and confirm genome annotations, and also, novel proteins were discovered (76). Additional experiments seek the identification of differentially expressed proteins by mass spectrometry of hemolymph from *Ae. aegypti* infected and not infected by the malaria parasite *P. gallinaceum* (personal communication of Margareth Lara Capurro).

As considered in this section, genomics, microarray techniques, proteomics, and bioinformatic analysis have been of extreme importance to characterize new promoter and parasite target molecules to be used in the construction of pathogen-refractory mosquitoes. However, much research still has to be done to test the available information to produce a successful parasite-refractory transgenic mosquito.

## From Laboratory to Field Work

Once a pathogen-refractory transgenic mosquito is obtained in the laboratory, the next step is its introduction to the environment to substitute the specific pathogen-susceptible vector population. To reach this objective, several aspects have to be carefully investigated in both the wild target population and in the transgenic population.

Recent research on the fitness of transgenic mosquitoes, including measurements of longevity and fertility and use of population cages, has been performed. Most of these studies were reviewed by Riehle *et al.* (77) and showed a reduced fitness load of transgenic mosquitoes when compared with the wild population. Only one transgene, the peptide SM1 expressed in *A. stephensi*, did not impose a detectable fitness load. The principal difference of this transgene was a specific promoter that restricted the foreign gene expression to posterior midgut cells for only a few hours after a blood meal, and the protein was secreted from the cells, which is what probably minimized the fitness decrease. Catteruccia *et al.* (78) also reported a reduced fitness of four different transgenic mosquito lines expressing fluorescent reporter proteins from an actin promoter, compared with wild-type mosquitoes. Irvin *et al.* (79) examined the impact of transgenesis on the fitness of *Ae. aegypti* transformed with enhanced GFP gene and two transposase genes derived from the *Hermes* and *MOS1* TEs. The authors found that demographic parameters were significantly diminished in transgenic mosquitoes relative to the untransformed laboratory strain.

Reduced fitness can occur because of different aspects of the manipulated vector: inbreeding depression, toxicity of a foreign protein expressed in abundance (80); random integration of transposable elements altering important vector genes; and the presence of a transposition repressor in the vector population, which after several generations could inhibit transposition gradually. This last aspect is of practical importance because in such cases the transgene(s) can be driven through a population only once (77).

Inbreeding depression is characteristic of the transgenic laboratory mosquito strains because each transgenic line arises from a single fertilized zygote containing transgenic gametes, and this characteristic can also reduce mosquito fitness (81).

Ribeiro and Kidwell (82) and Boete and Koella (83) developed a theoretical model suggesting that absolute absence of fitness load may not be essential for introducing genes into the wild population. The same authors also suggested that any released mosquito would need to be nearly 100% refractory to have any impact on malaria transmission. Thus, to obtain such a high blocking capability, it will be necessary to construct a

transgenic mosquito with multiple refractory genes that may probably incur greater fitness costs to the mosquito. However, there is considerable lack of experimental data to corroborate or disprove these models.

The target wild population also has to be analyzed. Locations where releases of transgenic mosquitoes are to occur also need to be well studied so that the transgenic mosquito population can be introduced without disruption of the surrounding natural environment. Thus, research has to be done to define the populations in the wild that will be the targets of genetic intervention, including size, ecological features, breeding structure, migration characteristics, number and distribution of genetically distinct vector populations, etc. It is important to stress that in malaria transmission, the ecology of the vectors are very complex in most locales. For example, the main African vectors, *A. gambiae* and *A. funestus*, are complexes of seven and nine (at least) subspecies, respectively, with different behaviors and ecology. Furthermore, *A. gambiae sensu stricto* consists of at least two sibling species with complex relations, revealing another level of complexity that is likely to be common within anophelines.

Mosquito mating behavior is another task to be investigated because in the past, the low success of genetically controlled arthropod vectors through the SIT was because of the low mating competence of released sterile males (6).

Another important aspect to be investigated is the biology of the pathogens to be eliminated, principally with respect to the genomic plasticity of malaria parasites and viruses such as dengue, of which today there are four serotypes. Thus, molecular epidemiology studies should be performed on the pathogen population also, principally if the effector gene is based on parasite gene targets. These problems can in part be solved by transforming mosquitoes with multiple effector molecules, but even so, it is important to stress the potential escape from the block and the development of resistant parasites.

A number of other aspects of the eventual release of genetically modified arthropod vectors of human pathogens must be considered. These include potential environmental hazards, public health risks, and public perceptions. Some governmental organizations developed standards and protocols to guide manipulation of important biological resources involved in human diseases. The National Institutes of Health and Centers for Disease Control have developed standards for the classification of risk and conditions for work with human disease agents. The United States Department of Agriculture Animal and Plant Health Inspection Service (USDA-APHIS) has established guidelines and an approval process for the release of transgenic arthropod plant pests. Although none of these directly addresses the release of genetically modified arthropod vectors of human disease, they establish precedents for risk assessment and regulation (84-86). Individual and public health risks must be assessed in both laboratory and pilot field studies, and accurate information must be provided to the public at the earliest possible time (87).

## Future Perspectives

A number of technological breakthroughs such as the identification of efficient TEs, the finding of suitable transformation markers, the characterization of efficient tissue-specific promoters, and the discovery of anti-pathogen effector gene candidates have been extensively explored by genomics, microarray techniques, proteomics, and bioinformatics analysis. Now, scientific efforts have to be made to test all new molecules characterized, including the development of an efficient mosquito embryo microinjection technique to enable several tests in a short period of time. Effector transgenes that confer full resistance against *Plasmodium* or dengue virus have not yet been developed. Difficulties in obtaining such molecules are hampered by the high genomic plasticity of parasites, which enables them to escape from human and arthropod immune systems. The biology of an ideal effector trait must be fully understood before it is put in use, because unintended phenotypes leading to irreversible alterations of the mosquito biology and potentially harmful ecological effects could be disastrous. Thus, several tasks have to be solved before we can expect to release transgenic insects to the environment to block human diseases.

## References

1. Hawley WA, Phillips-Howard PA, Ter Kuile FO, Terlouw DJ, Vulule JM, Ombok M, Nahlen BL, Gimnig JE, Kariuki SK, Kolczak MS, Hightower AW. Community-wide effects of permethrin-treated bed nets on child mortality and malaria morbidity in Western Kenya. *Am J Trop Med Hyg.* 2003;68Suppl. 4121–127. PubMed PMID: 12749495.
2. Knipling EF, Laven H, Craig GB, Pal R, Kitzmiller JB, Smith CN, Brown AW. Genetic control of insects of public health importance. *Bull World Health Organ.* 1968;38(3):421–438. PubMed PMID: 5302334.
3. Knipling EF. Sterile-male method of population control. *Science.* 1959;130: 902–904. PubMed PMID: 14410136.
4. Wyss JH. Screwworm eradication in the Americas. *Ann N Y Acad Sci.* 2000;916:186–193. PubMed PMID: 11193620.
5. Andreasen MH, Curtis CF. Optimal life stage for radiation sterilization of *Anopheles* males and their fitness for release. *Med Vet Entomol.* 2005;19:238–244. PubMed PMID: 16134971.
6. Benedict MQ, Robinson AS. The first releases of transgenic mosquitoes: an argument for the sterile insect technique. *Trends Parasitol.* 2003;19(8):349–355. PubMed PMID: 12901936.
7. Alphey L, Andreasen M. Dominant lethality and insect population control. *Mol Biochem Parasitol.* 2002;121:173–178. PubMed PMID: 12034450.
8. Stump AD, Shoener JA, Costantini C, Sagnon N, Besansky NJ. Sex-linked differentiation between incipient species of *Anopheles gambiae*. *Genetics.* 2005;169:1509–1519. PubMed PMID: 15654109.
9. Handler AM, James AA. *Insect transgenesis: methods and applications.* Boca Raton (FL): CRC Press LLC. 2000.
10. Higgs S, Traul D, Davis BS, Kamrud KI, Wilcox CL, Beaty BJ. Green fluorescent protein expressed in living mosquitoes--without the requirement of transformation. *Biotechniques.* 1996;21(4):660–664. PubMed PMID: 8891217.
11. McGrane V, Carlson JO, Miller BR, Beaty BJ. Microinjection of DNA into *Aedes triseriatus* ova and detection of integration. *Am J Trop Med Hyg.* 1988;39:502–510. PubMed PMID: 3195697.
12. Miller LH, Sakai RK, Romans P, Gwadz RW, Kantoff P, Coon HG. Stable integration and expression of a bacterial gene in the mosquito *Anopheles gambiae*. *Science.* 1987;237:779–781. PubMed PMID: 3039658.
13. Morris AC, Eggleston P, Crampton JM. Genetic transformation of the mosquito *Aedes aegypti* by microinjection of DNA. *Med Vet Entomol.* 1989;3:1–7. PubMed PMID: 2519641.
14. Cornel AJ, Benedict MQ, Rafferty CS, Howells AF, Collins FG. Transient expression of the *Drosophila melanogaster cinnabar* gene rescues eye color in the white eye (WE) strain of *Aedes aegypti*. *Insect Biochem Mol Biol.* 1997;27:993–997. PubMed PMID: 9569641.
15. Handler AM, Harrel RA. Germline transformation of *Drosophila melanogaster* with the piggyback transposon vector. *Insect Mol Biol.* 1999;8:449–457. PubMed PMID: 10634970.
16. Horn C, Jaunich B, Wimmer EA. Highly sensitive, fluorescent transformation marker for *Drosophila* transgenesis. *Dev Genes Evol.* 2000;210:623–629. PubMed PMID: 11151299.
17. Pinkerton AC, Michel K, O'Brochta DA, Atkinson PW. Green fluorescent protein as a genetic marker in transgenic *Aedes aegypti*. *Insect Mol Biol.* 2000;9(1):1–10. PubMed PMID: 10672065.
18. Ito J, Ghosh A, Moreira LA, Wimmer EA, Jacobs-Lorena M. Transgenic anopheline mosquitoes impaired in transmission of malaria parasite. *Nature.* 2002;417:452–455. PubMed PMID: 12024215.
19. Kokoza V, Ahmed A, Wimmer EA, Raikhel AS. Efficient transformation of the yellow fever mosquito *Aedes aegypti* using piggyback transposable element vector pBac(3xP3-EGFP afm). *Insect Biochem Mol Biol.* 2001;31:1137–1143. PubMed PMID: 11583926.
20. Curtis CF, Sinkins SP. Wolbachia as a possible means of driving genes into populations. *Parasitology.* 1998;116Suppl.S111–S115. PubMed PMID: 9695115.
21. Kidwell MG, Ribeiro JM. Can transposable elements be used to drive disease refractoriness genes into vector populations? *Parasitol Today.* 1992;8:325–329. PubMed PMID: 15463527.



22. Xiong C, Levis R, Shen P, Schlesinger S, Rice CM, Huang HV. Sindbis virus: an efficient, broad host range vector for gene expression in animal cells. *Science*. 1989;243(4895):1188–1191. PubMed PMID: 2922607.
23. Strauss JH, Strauss EG. The alphaviruses: gene expression, replication and evolution. *Microbiol Rev*. 1994;58:491–562. PubMed PMID: 7968923.
24. Taylor RM, Hurlbut HS, Work TH, Klingston JR, Frothingham TE. Sindbis virus: a newly recognized arthropod-transmitted virus. *Am J Trop Med Hyg*. 1955;4:844. PubMed PMID: 13259009.
25. Hurlbut H, Thomas J. The experimental host range of the arthropod-borne animal viruses in arthropods. *Virology*. 1960;12:391. PubMed PMID: 13716941.
26. Rice CM, Levis R, Strauss JH, Huang HV. Production of infectious RNA transcripts from Sindbis virus cDNA clones: mapping of lethal mutations, rescue of a temperature-sensitive marker, and in vitro mutagenesis to generate defined mutants. *J Virol*. 1987;61:3809–3819. PubMed PMID: 3479621.
27. Olson KE, Higgs S, Gaines PJ, Powers AM, Davis BS, Kamrud KI, Carlson JO, Blair CD, Beaty BJ. Genetically engineered resistance to dengue-2 virus transmission in mosquitoes. *Science*. 1996;272:884–886. PubMed PMID: 8629025.
28. Powers AM, Kamrud KI, Olson KE, Higgs S, Carlson JO, Beaty BJ. Molecularly engineered resistance to California serogroup virus replication in mosquito cells and mosquitoes. *Proc Natl Acad Sci U S A*. 1996;93:4187–4191. PubMed PMID: 8633038.
29. de Lara Capurro ML, Coleman J, Beerntsen BT, Myles KM, Olson KE, Rocha E, Krettli AU, James AA. Virus-expressed, recombinant single-chain antibody blocks sporozoite infection of salivary glands in *Plasmodium gallinaceum*-infected *Aedes aegypti*. *Am J Trop Med Hyg*. 2000;62(4):427–433. PubMed PMID: 11220756.
30. Johnson BW, Olson KE, Allen-Miura T, Rayms-Keller A, Carlson JO, Coates CJ, Jasinskiene N, James AA, Beaty BJ, Higgs S. Inhibition of luciferase expression in transgenic *Aedes aegypti* mosquitoes by Sindbis virus expression of antisense luciferase RNA. *Proc Natl Acad Sci U S A*. 1999;96(23):13399–13403. PubMed PMID: 10557332.
31. Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, Devon K, Dewar K, Doyle M, FitzHugh W, Funke R, Gage D, Harris K, Heaford A, Howland J, Kann L, Lehoczky J, LeVine R, McEwan P, McKernan K, Meldrim J, Mesirov JP, Miranda C, Morris W, Naylor J, Raymond C, Rosetti M, Santos R, Sheridan A, Sougnez C, Stange-Thomann N, Stojanovic N, Subramanian A, Wyman D, Rogers J, Sulston J, Ainscough R, Beck S, Bentley D, Burton J, Clee C, Carter N, Coulson A, Deadman R, Deloukas P, Dunham A, Dunham I, Durbin R, French L, Grafham D, Gregory S, Hubbard T, Humphray S, Hunt A, Jones M, Lloyd C, McMurray A, Matthews L, Mercer S, Milne S, Mullikin JC, Mungall A, Plumb R, Ross M, Shownkeen R, Sims S, Waterston RH, Wilson RK, Hillier LW, McPherson JD, Marra MA, Mardis ER, Fulton LA, Chinwalla AT, Pepin KH, Gish WR, Chissoe SL, Wendl MC, Delehaunty KD, Miner TL, Delehaunty A, Kramer JB, Cook LL, Fulton RS, Johnson DL, Minx PJ, Clifton SW, Hawkins T, Branscomb E, Predki P, Richardson P, Wenning S, Slezak T, Doggett N, Cheng JF, Olsen A, Lucas S, Elkin C, Uberbacher E, Frazier M, Gibbs RA, Muzny DM, Scherer SE, Bouck JB, Sodergren EJ, Worley KC, Rives CM, Gorrell JH, Metzker ML, Naylor SL, Kucherlapati RS, Nelson DL, Weinstock GM, Sakaki Y, Fujiyama A, Hattori M, Yada T, Toyoda A, Itoh T, Kawagoe C, Watanabe H, Totoki Y, Taylor T, Weissenbach J, Heilig R, Saurin W, Artiguenave F, Brottier P, Bruls T, Pelletier E, Robert C, Wincker P, Smith DR, Doucette-Stamm L, Rubenfield M, Weinstock K, Lee HM, Dubois J, Rosenthal A, Platzer M, Nyakatura G, Taudien S, Rump A, Yang H, Yu J, Wang J, Huang G, Gu J, Hood L, Rowen L, Madan A, Qin S, Davis RW, Federspiel NA, Abola AP, Proctor MJ, Myers RM, Schmutz J, Dickson M, Grimwood J, Cox DR, Olson MV, Kaul R, Raymond C, Shimizu N, Kawasaki K, Minoshima S, Evans GA, Athanasiou M, Schultz R, Roe BA, Chen F, Pan H, Ramser J, Lehrach H, Reinhardt R, McCombie WR, de la Bastide M, Dedhia N, Blöcker H, Hornischer K, Nordsiek G, Agarwala R, Aravind L, Bailey JA, Bateman A, Batzoglu S, Birney E, Bork P, Brown DG, Burge CB, Cerutti L, Chen HC, Church D, Clamp M, Copley RR, Doerks T, Eddy SR, Eichler EE, Furey TS, Galagan J, Gilbert JG, Harmon C, Hayashizaki Y, Haussler D, Hermjakob H, Hokamp K, Jang W, Johnson LS, Jones TA, Kasif S, Kasprzyk A, Kennedy S, Kent WJ, Kitts P, Koonin EV, Korfi I, Kulp D, Lancet D, Lowe TM, McLysaght A, Mikkelsen T, Moran JV, Mulder N, Pollara VJ, Ponting CP, Schuler G, Schultz J, Slater G, Smit AF, Stupka E, Szustakowski

- J, Thierry-Mieg D, Thierry-Mieg J, Wagner L, Wallis J, Wheeler R, Williams A, Wolf YI, Wolfe KH, Yang SP, Yeh RF, Collins F, Guyer MS, Peterson J, Felsenfeld A, Wetterstrand KA, Patrinos A, Morgan MJ, de Jong P, Catanese JJ, Osoegawa K, Shizuya H, Choi S, Chen YJ. International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature*. 2001;409:860–921. PubMed PMID: 11237011.
32. Kidwell MG, Lisch DR. Transposable elements and host genome evolution. *Trends Ecol Evol*. 2000;15(3): 95–99. PubMed PMID: 10675923.
  33. Hedges DJ, Batzer MA. From the margins of the genome: mobile elements shape primate evolution. *Bioessays*. 2005;27(8):785–794. PubMed PMID: 16015599.
  34. Jasinskiene N, Coates CJ, Benedict MQ, Cornel AJ, Rafferty CS, James AA, Collins FH. Stable transformation of the yellow fever mosquito, *Aedes aegypti*, with the Hermes element from the housefly. *Proc Natl Acad Sci U S A*. 1998;95:3743–3747. PubMed PMID: 9520437.
  35. Coates CJ, Jasinskiene N, Miyashiro L, James AA. Mariner transposition and transformation of the yellow fever mosquito, *Aedes aegypti*. *Proc Natl Acad Sci U S A*. 1998;95:3748–3751. PubMed PMID: 9520438.
  36. Wilson R, Orsetti J, Klocko AD, Aluvihare C, Peckham E, Atkinson PW, Lehane MJ, O'Brochta DA. Post-integration behavior of a Mos1 mariner gene vector in *Aedes aegypti*. *Insect Biochem Mol Biol*. 2003;33:853–863. PubMed PMID: 12915177.
  37. O'Brochta DA, Sethuraman N, Wilson R, Hice RH, Pinkerton AC, Levesque CS, Bideshi DK, Jasinskiene N, Coates CJ, James AA, Lehane MJ, Atkinson PW. Gene vector and transposable element behavior in mosquitoes. *J Exp Biol*. 2003;206:3823–3834. PubMed PMID: 14506218.
  38. Nolan T, Bower M, Brown AE, Crisanti A, Catteruccia F. piggyBac-mediated germline transformation of the malaria mosquito *Anopheles stephensi* using the red fluorescent protein dsRED as a selectable marker. *J Biol Chem*. 2002;277:8759–8762. PubMed PMID: 11805082.
  39. Perera OP, Harrell RA, Handler AM. Germ-line transformation of the South American malaria vector, *Anopheles albimanus*, with a piggyBac/EGFP transposon vector is routine and highly efficient. *Insect Mol Biol*. 2002;11:291–297. PubMed PMID: 12144693.
  40. Grossman GL, Rafferty CS, Clayton JR, Stevens TK, Mukabayire O, Benedict MQ. Germline transformation of the malaria vector, *Anopheles gambiae*, with the piggybac transposable element. *Insect Mol Biol*. 2001;10:597–604. PubMed PMID: 11903629.
  41. Catteruccia F, Nolan T, Blass C, Muller HM, Crisanti A, Kafatos FC, Loukeris TG. Toward *Anopheles* transformation: *Minos* element activity in anopheline cells and embryos. *Proc Natl Acad Sci U S A*. 2000;97:2157–2162. PubMed PMID: 10681436.
  42. Beard CB, Durvasula RV, Richards FF. Bacterial symbiosis in arthropods and the control of disease transmission. *Emerg Infect Dis*. 1998;4:581–591. PubMed PMID: 9866734.
  43. Sinkins SP, Curtis CF, O'Neill SL. The potential application of inherited symbiont systems to pest control. In: O'Neill SL, Hoffman A, Werren J, editors. *Influential passengers*. Oxford (UK): Oxford University Press; 1997. p. 155-175.
  44. Christophides GK. Transgenic mosquitoes and malaria transmission. *Cell Microbiol*. 2005;7(3):325–333. PubMed PMID: 15679836.
  45. Nirmala X, James AA. Engineering Plasmodium-refractory phenotypes in mosquitoes. *Trends Parasitol*. 2003;19(9):384–387. PubMed PMID: 12957511.
  46. Ghosh AK, Ribolla PE, Jacobs-Lorena M. Targeting Plasmodium ligands on mosquito salivary glands and midgut with a phage display peptide library. *Proc Natl Acad Sci U S A*. 2001;98(23):13278–13281. PubMed PMID: 11687659.
  47. Myung JM, Marshall P, Sinnis P. The Plasmodium circumsporozoite protein is involved in mosquito salivary gland invasion by sporozoites. *Mol Biochem Parasitol*. 2004;133:53–59. PubMed PMID: 14668012.
  48. Gwadz RW, Kaslow D, Lee JY, Maloy WL, Zasloff M, Miller LH. Effects of magainins and cecropins on the sporogonic development of malaria parasites in mosquitoes. *Infect Immun*. 1989;57(9):2628–2633. PubMed PMID: 2759705.

49. Shahabuddin M, Fields I, Bulet P, Hoffmann JA, Miller LH. Plasmodium gallinaceum: differential killing of some mosquito stages of the parasite by insect defensin. *Exp Parasitol*. 1998;89:103–112. PubMed PMID: 9603495.
50. Arrighi RB, Nakamura C, Miyake J, Hurd H, Burgess JG. Design and activity of antimicrobial peptides against sporogonic-stage parasites causing murine malaras. *Antimicrob Agents Chemother*. 2002;46(7): 2104–2110. PubMed PMID: 12069961.
51. Sanchez-Vargas I, Travanty EA, Keene KM, Franz AW, Beaty BJ, Blair CD, Olson KE. RNA interference, arthropod-borne viruses, and mosquitoes. *Virus Res*. 2004;102(1):65–74. PubMed PMID: 15068882.
52. Adelman ZN, Blair CD, Carlson JO, Beaty BJ, Olson KE. Sindbis virus-induced silencing of dengue viruses in mosquitoes. *Insect Mol Biol*. 2001;10(3):265–73. PubMed PMID: 11437918.
53. Franz AW, Sanchez-Vargas I, Adelman ZN, Blair CD, Beaty BJ, James AA, Olson KE. Engineering RNA interference-based resistance to dengue virus type 2 in genetically modified *Aedes aegypti*. *Proc Natl Acad Sci U S A*. 2006;103(11):4198–4203. PubMed PMID: 16537508.
54. Travanty EA, Adelman ZN, Franz AWE, Keene KM, Beaty BJ, Blair CD, James AA, Olson KE. Using RNA interference to develop dengue virus resistance in genetically modified *Aedes aegypti*. *Insect Biochem Mol Biol*. 2004;34:607–613. PubMed PMID: 15242701.
55. Beerntsen BT, James AA, Christensen BM. Genetics of mosquito vector competence. *Microbiol Mol Biol Rev*. 2000;64(1):115–137. PubMed PMID: 10704476.
56. Moreira LA, Ito J, Ghosh A, Devenport M, Zieler H, Abraham EG. Bee venom phospholipase inhibits malaria parasite development in transgenic mosquitoes. *J Biol Chem*. 2002;277:40839–40843. PubMed PMID: 12167627.
57. James AA, Blackmer K, Racioppi JV. A salivary gland-specific maltase-like gene of the vector mosquito, *Aedes aegypti*. *Gene*. 1989;75:73–83. PubMed PMID: 2470653.
58. Smartt CT, Kim AP, Grossman GL, James AA. The apyrase gene of the vector mosquito, *Aedes aegypti*, is expressed specifically in the adult female salivary glands. *Exp Parasitol*. 1995;81:239–248. PubMed PMID: 7498420.
59. Coates CJ, Jasinskiene N, Pott G, James AA. Promoter directed expression of recombinant fire-fly luciferase in the salivary glands of Hermes-transformed *Aedes aegypti*. *Gene*. 1999;226:317–325. PubMed PMID: 9931506.
60. Kokoza V, Ahmed A, Cho WL, Jasinskiene N, James AA, Raikhel A. Engineering blood meal-activated systemic immunity in the yellow fever mosquito *Aedes aegypti*. *Proc Natl Acad Sci U S A*. 2000;97:9144–9149. PubMed PMID: 10908672.
61. Moreira LA, Edwards MJ, Adhami F, Jasinskiene N, James AA, Jacobs-Lorena M. Robust gut-specific gene expression in transgenic *Aedes aegypti* mosquitoes. *Proc Natl Acad Sci U S A*. 2000;97:10895–10898. PubMed PMID: 11005862.
62. Holt RA, Subramantan GM, Halpern A, Sutton GG, Charlab R, Nusskern DR, Wincker P, Clark AG, Ribeiro JM, Wides R, et al. The genome sequence of the malaria mosquito *Anopheles gambiae*. *Science*. 2002;298:129–149. PubMed PMID: 12364791.
63. Dimopoulos G, Casavant TL, Chang S, Scheetz T, Roberts C, Donohue M, Schultz J, Benes V, Bork P, Ansorge W. *Anopheles gambiae* pilot gene discovery project identification of mosquito innate immunity genes from expressed sequence tags generated from immune-competent cell lines. *Proc Natl Acad Sci U S A*. 2000;97:6619–6624. PubMed PMID: 10841561.
64. Dimopoulos G, Christophides GK, Meister S, Schultz J, White KP, Barillas-Mury C, Kafatos FC. Genome expression analysis of *Anopheles gambiae*: responses to injury, bacterial challenge, and malaria infection. *Proc Natl Acad Sci U S A*. 2002;99:8814–8819. PubMed PMID: 12077297.
65. Kumar S, Christophides GK, Cantera R, Charles B, Han YS, Meister S, Dimopoulos G, Kafatos FC, Barillas-Mury C. The role of reactive oxygen species on *Plasmodium* melanotic encapsulation in *Anopheles gambiae*. *Proc Natl Acad Sci U S A*. 2003;100:14139–14144. PubMed PMID: 14623973.

66. Vlachou D, Schlegelmilch T, Christophides GK, Kafatos FC. Functional genomic analysis of midgut epithelial responses in *Anopheles* during *Plasmodium* invasion. *Curr Biol*. 2005;15(13):1185–1195. PubMed PMID: 16005290.
67. Dana AN, Hong YS, Kern MK, Hillenmeyer ME, Harker BW, Lobo NF, Hogan JR, Romans P, Collins FH. Gene expression patterns associated with blood-feeding in the malaria mosquito *Anopheles gambiae*. *BMC Genomics*. 2005;6:5–29. PubMed PMID: 15651988.
68. Kriventseva EV, Koutsos AC, Blass C, Kafatos FC, Christophides GK, Zdobnov EM. AnoEST: toward *A. gambiae* functional genomics. *Genome Res*. 2005;15:893–899. PubMed PMID: 15899967.
69. Severson DW, Knudson DL, Soares MB, Loftus BJ. *Aedes aegypti* genomics. *Insect Biochem Mol Biol*. 2004;34:715–721. PubMed PMID: 15242713.
70. Sanders HR, Evans AM, Ross LS, Gill SS. Blood meal induces global changes in midgut gene expression in the disease vector, *Aedes aegypti*. *Insect Biochem Mol Biol*. 2003;33:1105–1122. PubMed PMID: 14563362.
71. Feitosa FM, Calvo E, Merino EF, Durham AM, James AA, de Bianchi AG, Marinotti O, Capurro ML. A transcriptome analysis of the *Aedes aegypti* vitellogenic fat body. *J Insect Sci*. 2006;6:1–26.
72. Abraham EG, Islam S, Srinivasan P, Ghosh AK, Valenzuela JG, Ribeiro JMC, Kafatos FC, Kimopoulos G, Jacobs-Lorena M. Analysis of the *Plasmodium* and *Anopheles* transcriptional repertoire during Ookinete development and midgut invasion. *J Biol Chem*. 2004;279(7):5573–5580. PubMed PMID: 14627712.
73. Srinivasan P, Abraham EG, Ghosh AK, Valenzuela J, Ribeiro JMC, Dimopoulos G, Kafatos FC, Adams JH, Fujioka H, Jacobs-Lorena M. Analysis of the *Plasmodium* and *Anopheles* transcriptomes during oocyst differentiation. *J Biol Chem*. 2004;279(7):5581–5587. PubMed PMID: 14627711.
74. Xu X, Dong Y, Abraham EG, Kocan A, Srinivasan P, Ghosh AK, Sinden RE, Ribeiro JMC, Jacobs-Lorena M, Kafatos FC, Dimopoulos G. Transcriptome analysis of *Anopheles stephensi-Plasmodium berghei* interactions. *Mol Biochem Parasitol*. 2005;142(1):76–87. PubMed PMID: 15907562.
75. Scali C, Catteruccia F, Qiuxiang L, Crisanti A. Identification of sex-specific transcripts of the *Anopheles gambiae* doublesex gene. *J Exp Biol*. 2005;208(19):3701–3709. PubMed PMID: 16169947.
76. Kalume DE, Peri S, Reddy R, Zhong J, Okulate M, Kumar N, Pandey A. Genome annotation of *Anopheles gambiae* using mass spectrometry-derived data. *BMC Genomics*. 2005;6:128–138. PubMed PMID: 16171517.
77. Riehle MA, Srinivasan P, Moreira CK, Jacobs-Lorena M. Towards genetic manipulation of wild mosquito populations to combat malaria: advances and challenges. *J Exp Biol*. 2003;206:3809–3816. PubMed PMID: 14506216.
78. Catteruccia F, Godfray HC, Crisanti A. Impact of genetic manipulation on the fitness of *Anopheles stephensi* mosquitoes. *Science*. 2003;299(5610):1225–1227. PubMed PMID: 12595691.
79. Irvin N, Hoddle MS, O'Brochta DA, Carey B, Atkinson PW. Assessing fitness costs for transgenic *Aedes aegypti* expressing the GFP marker and transposase genes. *Proc Natl Acad Sci U S A*. 2004;101(3):891–896. PubMed PMID: 14711992.
80. Liu HS, Jan MS, Chou CK, Chen PH, De NJ. Is green fluorescent protein toxic to the living cells? *Biochem Biophys Res Commun*. 1999;260:712–717. PubMed PMID: 10403831.
81. Taylor C, Toure YT, Carnahan J, Norris DE, Dolo G, Traore SF, Edillo FE, Lanzaro GC. Gene flow among populations of the malaria vector, *Anopheles gambiae*, in Mali, West Africa. *Genetics*. 2001;157(2):743–750. PubMed PMID: 11156993.
82. Ribeiro JM, Kidwell MG. Transposable elements as population drive mechanisms: specification of critical parameter values. *J Med Entomol*. 1994;31(1):10–16. PubMed PMID: 8158612.
83. Boete C, Koella JC. Evolutionary ideas about genetically manipulated mosquitoes and malaria control. *Trends Parasitol*. 2003;19(1):32–38. PubMed PMID: 12488224.
84. Hollander AK. Environmental impacts of genetically engineered microbial and viral biocontrol agents. In: Maramorosch K, editor. *Biootechnology for biological control of pests and vectors*. Boca Raton (FL): CRC Press; 1991. p. 251–266.

85. Hoy MA. Impact of risk analyses on pest-management programs employing transgenic arthropods. *Parasitol Today*. 1995;11:229–232.
86. Simonsen L, Levin BR. Evaluating the risk of releasing genetically engineered organisms. *Trends Ecol Evol*. 1998;2:s27.
87. Asner M. Public relations: the scientist and the public, the government, and the media. In: Purchase HG, Mackenzie, *Agricultural biotechnology. Introduction to field testing*. Washington (DC): Office of Agricultural Biotechnology, USDA; 1990. p. 35.



## Chapter B05. Proteomics Studies in *Trypanosoma cruzi*

Ernesto S. Nakayasu, BSc<sup>1</sup> and Igor C. Almeida, PhD<sup>2</sup>

Created: October 12, 2006; Updated: September 12, 2007.

In this chapter, we will briefly discuss some of the most relevant advances in Bioinformatics tools for proteomic analysis and their application for the study of the *Trypanosoma cruzi* proteome. Furthermore, we will discuss the expression and functional proteomic analyses in *T. cruzi*, how these combined approaches may help us to improve our understanding of protein functions, and discover new molecular targets for the development of novel therapies against this parasite.

### Bioinformatics in Proteomic Analysis

In the beginning of the 1990s, peptide sequencing by mass spectrometry (MS) was a time-consuming process. Each peptide was individually selected and fragmented by collision-induced dissociation, and the resulting tandem MS (or MS/MS) data were then analyzed manually by a procedure known as *de novo* sequencing. With the advance of automatic data recording by most mass spectrometers, a large amount of data could be now collected. The interpretation of MS spectra, however, was still a problem to be solved. In 1993, Henzel *et al.* (1) developed a database search algorithm to interpret peptide-mass fingerprinting (PMF) data. With this advance, complex mixtures of proteins could be separated by two-dimensional gel-electrophoresis (2-DE), the spots digested with trypsin and analyzed by Matrix-Assisted Laser Desorption/Ionization-Time of Flight-Mass Spectrometry (MALDI-TOF-MS), which accelerated tremendously the process of protein identification by MS. On the other hand, the combination of reverse phase liquid chromatography in tandem with electrospray-mass spectrometry (LC-MS) made it possible to collect MS/MS spectra from thousands of peptides from a complex protein mixture in a single run.

In addition, rapid advances in bioinformatics resulted in the development of several algorithms that could automatically process and interpret the MS and MS/MS data. In 1994, Eng and colleagues (2) developed the first database-search algorithm (Sequest) to analyze electrospray-ionization (ESI) MS/MS spectra. Since then, several other database-search algorithms, such as Mascot (3) and Phenyx (4, 5), have been developed. Essentially, all of these algorithms scan the database(s) for peptides with molecular masses identical to the experimentally measured peptides. Then, each peptide candidate is fragmented *in silico* and compared with the experimental ESI-MS/MS spectrum. Nonetheless, each algorithm has a different way to calculate the score to rank peptide candidates, which results in the identification of different peptides (6). Thus, the use of more than one database search algorithm can result in a better coverage of the proteomic analysis (6, 7). For instance, recently, three database search algorithms (i.e., Mascot, Phenyx, and Sequest) were applied for the proteomic analysis of vesicles secreted by infective trypomastigote forms of *T. cruzi*, resulting in a 50% increase in the number of identified proteins (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data).

The combination of LC-MS/MS analysis with database search of spectra made it possible to analyze thousands of peptides from very complex samples, such as tissue extracts or whole-cell lysates. Currently, with the continuous release of an enormous amount of proteomic data from different sources, one of the main concerns is the accuracy and reliability of the protein sequencing information being generated (8). Yet, many researchers in the field set their own threshold parameters to validate their data, without further performing any methodical statistical analysis. The first statistical analysis to validate a proteomic dataset was done in 2003 by Peng *et al.* (9),

who performed an LC-MS/MS analysis of whole yeast extract. All collected data were analyzed with the Sequest algorithm against the yeast database and the reverse version of the same database. The estimation of false-positive rate (FPR) was based on the number of positive identifications in the reverse database. More recently, Weatherly *et al.* (10) developed a tool, named PROVALT, to assemble database search results from several Mascot analyses and simultaneously estimate the FPR. PROVALT was also applied to assemble and validate *T. cruzi* proteomic data (11). PROVALT uses the reverse database approach to estimate the FPR. However, there is another approach to estimate the FPR using a chimeric database. In this approach, sequences from the database of the organism being studied (e.g., TcruziDB) are inserted in a larger database with randomly generated sequences. The estimation of the FPR is based on the number of positive identifications for randomly generated sequences (12, 13). This type of approach was recently applied to validate the data obtained from the proteomic analysis of vesicles secreted by *T. cruzi* trypomastigotes (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data).

Another concern in proteomic data is the number of redundant hits. This is a special concern in *T. cruzi* proteomic analysis because of the existence of multigene families, such as *trans*-sialidase (TS)/gp85, mucins, MASP, and gp63. There are several members from these multigene families that share exactly the same peptide(s). If one or more of these peptides are identified by proteomic analysis, there is no way to determine which one(s) actually is the expressed protein sequence. However, when the identified peptides are grouped into proteins, some “diagnostic peptides” (not shared by other sequences) can be found.

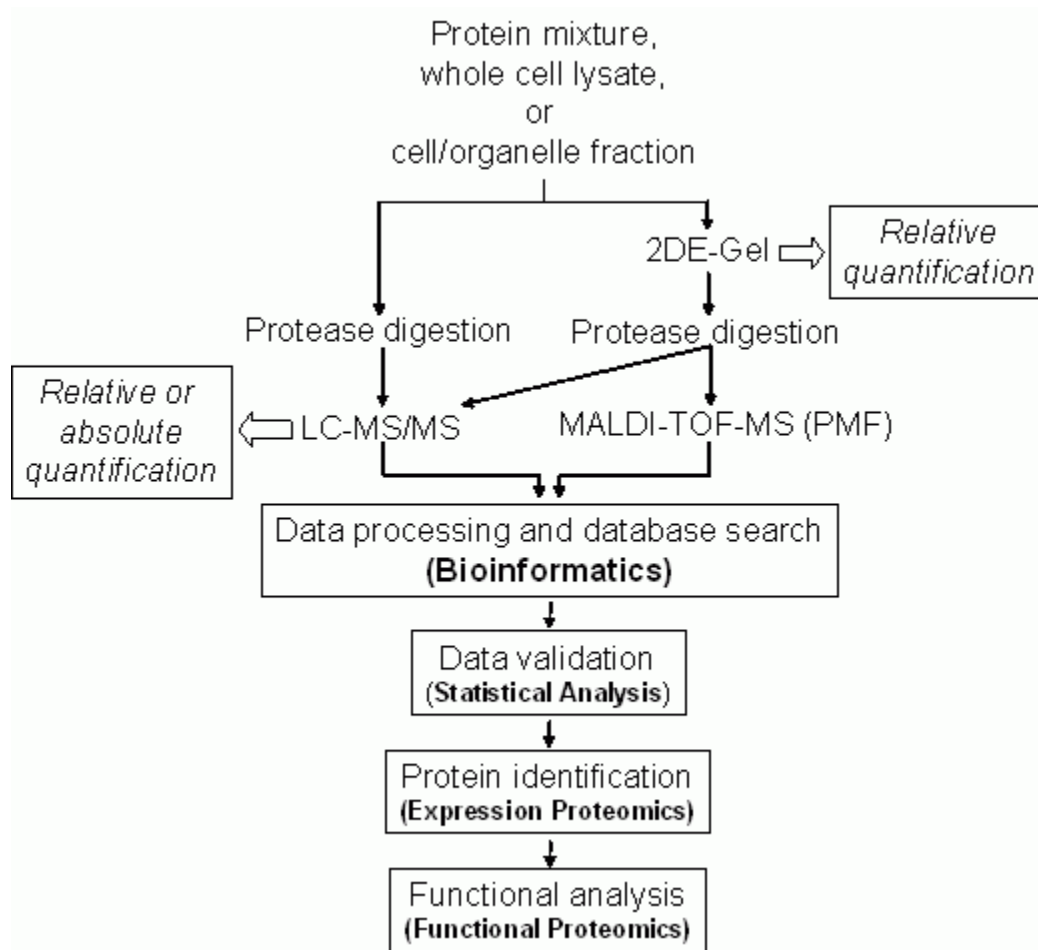
PROVALT uses a different approach regarding the redundant hits. This software validates all these sequences as a group of proteins (10, 11). There are two concerns about validating these sequences as protein groups: i) overestimation of number of identified proteins; and ii) artificial decrease of FPR, by increase of the number of “true-positive” identifications. On the other hand, the problem of validating only the most probable sequence is that the number of proteins can be underestimated. Thus, it is the role of researchers to decide whether it is better to over- or underestimate the number of identified proteins.

Although many researchers are interested in only identifying proteins, several of them are also interested in quantifying the expression levels of those proteins (reviewed by Ong and Mann (14)). There are two types of quantification for proteomic analysis: relative and absolute. Relative quantification can be performed by 2-DE, or labeling samples with different isotopes and analyzing them by LC-MS (see below). Absolute quantification is based on the addition of internal standard(s) into the samples for LC-MS analysis. Then, the abundance of each peptide is normalized by the internal standard(s) and compared with a curve of external standards. Bioinformatics tools are, therefore, essential for the quantification of large proteomic datasets. For instance, Xpress (15) and ASAPratios (16) are tools developed to relatively quantify samples labeled with different isotopes. The automation of absolute quantification of proteins by MS is a little more complicated, because it requires both an internal and an external standard curve. However, recently, Ishihama *et al.* (17) found a correlation between the number of identified peptides from the same protein and the absolute amount of this protein. This process does not require any internal or external standard and is based on the concept that the most abundant proteins have a better coverage in proteomic analysis by LC-MS (18). In recent years, quantitative proteomics has gained great significance in the comparative analysis of cells from different life cycle stages, physiological states, and drug treatments. A current overview of proteomic analysis is schematized in Figure 1.

## Expression Proteomics in *T. cruzi*

The main goal of the expression proteomics approach, as the term implies, is to provide a snapshot of the proteins expressed by either whole cell, organelle, or cell fraction at different times and culture conditions. The rationale is that by determining which, when, and where proteins are expressed, we will advance our knowledge about protein localization and potential function. Moreover, in the case of parasites, expression proteomic





**Figure 1.** Overview of the two major approaches used for proteomic analysis. PMF, peptide-mass fingerprinting.

analysis may provide new insights about host–parasite interplay and potential molecular targets for the development of novel chemo- and immunotherapeutic interventions (19, 20).

The first *Trypanosoma cruzi* proteomic data were published in 2004 by Paba *et al.* (21). In this paper, the authors performed a comparative proteomic analysis by 2-DE and PMF of three different parasite life cycle stages, namely epimastigote, mammalian cell-derived trypomastigote, and extracellular amastigote. The latter was obtained by *in vitro* differentiation of mammalian cell-derived trypomastigotes. As expected, several spots showed to be specific for each parasitic stage, whereas others were shared by all three stages. After imaging analysis, 80 spots from 2-DE gels with whole trypomastigote extract were submitted to PMF by MALDI-TOF-MS, and 19 different proteins were identified, including heat-shock proteins (HSP 60, HSP 70, and HSP 90), elongation factors, metabolic proteins (enolase, pyruvate kinase, and 2,3-biphosphoglycerate mutase), and structural proteins (KMP 11, tubulin, and paraflagellar rod components). Concurrently, Parodi-Talice *et al.* (22) analyzed 70 spots from 2-DE gels with whole epimastigote extract by PMF, resulting in the identification of 45 of them. These proteins included HSP 60 and HSP 70, metabolic proteins (dehydrogenases, 3-phosphoglycerate kinase, cystathione  $\beta$ -synthase, trypanredoxin peroxidase, and peroxiredoxin), structural proteins ( $\alpha$ - and  $\beta$ -tubulin), and proteins related to cell cycle and growth (CCHC zinc finger protein, RNA binding protein RGGm, nascent polypeptide-associated complex  $\alpha$ , and retrotransposon hot spot protein).

Isotope-coded affinity tagging (ICAT) is another approach widely used to identify and relatively quantify proteins from different cell stages or stimuli (23). In this approach, certain amino acid residues from proteins from one cell stage (or stimulus, treatment, or control) are labeled with biotin containing no deuterium atom,

whereas proteins from the other cell stage (or stimulus or treatment) are labeled with biotin containing eight deuterium atoms. The two differentially labeled samples are mixed, and the protein mixture is digested with proteases. Labeled peptides are purified by affinity chromatography using avidin or streptavidin column, and the released peptides are analyzed by liquid chromatography coupled to tandem mass spectrometry (LC-MS/MS). In the MS, peptides from one stage (or stimulus or treatment) are detected with a shift of 8 Da when compared with peptides from the other cell stage (or stimulus, treatment, or control). This information combined with tandem MS data provides the identification and relative quantification of proteins in a single LC-MS/MS analysis (14, 24). Paba *et al.* (25) applied the ICAT approach to analyze proteins from trypomastigote and amastigote stages, resulting in the identification and relative quantification of 41 proteins from *T. cruzi*. These proteins include heat shock proteins (HSP 60, HSP 70, and HSP 83), elongation factor, metabolic proteins (enolase, pyruvate dehydrogenase, and ATPase), and structural proteins (tubulin, paraflagellar rod components, and a protein precursor homologous to oocyst wall protein of *Cryptosporidium parvum*).

The comparative analyses between trypomastigotes and amastigotes, and trypomastigotes and epimastigotes showed a relatively low number of differentially expressed proteins (21, 25). Paraflagellar rod proteins were shown to be more expressed in trypomastigotes than in epimastigotes and amastigotes (trypomastigotes > epimastigotes > amastigotes). Also, enolase, elongation factors, heat shock proteins, pyruvate kinase, Rab 6, and vacuolar ATP synthase were shown to be more expressed in trypomastigotes than in epimastigotes (21). Using the ICAT approach, Paba *et al.* (25) found that the Ran/TC4 homolog and RNA1 polyprotein were more abundant in trypomastigotes than in amastigotes, whereas poly zinc-finger protein, cruzipain, and a transmembrane glycoprotein homolog of the bovine ephemeral fever virus were shown to be more expressed in amastigotes than in trypomastigotes. It is worth pointing out, however, that these two studies, despite their unquestionable importance, failed to identify and quantify major surface glycosylphosphatidylinositol (GPI)-anchored glycoproteins, such as mucins and *trans*-sialidase (TS)/gp85 superfamily members (26, 27), expression of which changes according to the parasite developmental stage. Nevertheless, it is understandable why these glycoproteins were not detected in the proteomic studies described above. In the case of mucins, for instance, they are heavily *O*-glycosylated and highly variable in their peptide sequence (28, 29), and thus are unlikely to be detected by conventional proteomic approaches (29).

More recently, a broader proteomic analysis of all four developmental stages of *T. cruzi* has identified 223 proteins from the TS/gp85 superfamily, clearly denoting the profusion of these surface glycoproteins (11). However, it is difficult to determine exactly how many sequences from each group of TS/gp85 were actually identified, because sequences from the same group have been annotated under different names in various databases, such as GenBank ([www.ncbi.nih.gov/Genbank/](http://www.ncbi.nih.gov/Genbank/)), TcruziDB ([www.tcruzidb.org](http://www.tcruzidb.org)), and GeneDB ([www.genedb.org](http://www.genedb.org)). Another problem is related to the redundancy of multigene family sequences, such as the TS/gp85 glycoproteins. The approach applied by Atwood *et al.* (11) validating redundant proteins as protein groups made it difficult to discriminate which ones were actually expressed by the parasite. It is worth pointing out, however, that members from all gp85/TS groups were identified in the study by Atwood and colleagues (11). Interestingly, protein hits such as gp82 was found only in the metacyclic trypomastigote stage, in agreement with previous data (30). More recently, the proteomic analysis of vesicles shed by trypomastigote forms of *T. cruzi* (TcVes) resulted in the identification of several TS/gp85 superfamily members, including SA85, Tc85, gp90, trypomastigote ligand, TS, FL-160, and CRP (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data). To avoid redundancy, for the annotation of the proteomic data obtained from these vesicles, only the most probable sequences were accepted. In the case that the same peptide(s) was shared by more than one sequence, only the sequence that appeared first in the database was validated.

Another important superfamily of surface proteins is the major surface protease (MSP), also known as GP63 or gp63. gp63 is a GPI-anchored glycoprotein of 63 kDa, with metalloprotease activity (31). Despite some recent controversies, *Leishmania* GP63 members seem to play important roles during infection, such as: i) regulation of

the complement-mediated lysis by converting active C3b into the inactive form C3bi; ii) opsonization and facilitation of the phagocytosis process; and iii) helping the parasite to survive inside host cells (31). In *T. cruzi*, gp63 also seems to play a key role in the infection process, because parasite treatment with anti-gp63 antibodies resulted in decreased host–cell invasion (32). The analysis of the *T. cruzi* genome showed the presence of 425 members of the gp63 family, including 251 pseudogenes (33). The functional genes encode for proteins of approximately 550 amino acids with potential *N*-glycosylation and GPI-anchoring sites. Treatment of trypomastigotes with phosphatidylinositol (PI)-specific phospholipase C (PI-PLC) resulted in the release of gp63 from the parasite surface, strongly indicating the presence of a GPI-anchor in these proteins (32). The proteomic analysis of all four stages of *T. cruzi* led to the identification of several gp63 hits (11). In addition, recent proteomic analysis of vesicles shed by cell-derived trypomastigotes showed the presence of at least three members of gp63 family, two of them also found by Atwood *et al.* (11) (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data).

Genomic analysis of *T. cruzi* revealed two new surface protein superfamilies, the mucin-associated surface proteins (MASP) (1377 members, 433 pseudogenes) and mucin-like proteins (123 members) (33). Both gene families display signal peptide for GPI-anchoring and several potential *O*- and *N*-linked glycosylation sites. Proteomic analysis of all four parasite stages showed the presence of MASP products, but not mucin-like ones (11). Akin to mucins (29), both superfamilies are potentially highly glycosylated and, therefore, difficult to be identified by conventional proteomic analysis. Table 1 summarizes the findings of the expression proteomic studies described above.

Broad expression proteomic analysis of whole parasite extracts, nevertheless, has serious technical limitations, mainly because of the constraints in resolution and sensitivity of available protein/peptide separation and spectrometric methods. For instance, the 2-DE approach suffers serious limitation in resolving proteins with high molecular mass, hydrophobic proteins, or proteins with extensive post-translational modifications (e.g., glycosylation, phosphorylation) (34). Also, recovery of peptides from the polyacrylamide gel after digestion with the specific protease is usually very low (<1%). On the other hand, despite being much more sensitive than 2-DE-based approaches, liquid chromatography (LC)-based separation techniques still have some limitations in resolving peptide isoforms or modified peptides (e.g., glycopeptides). Furthermore, even very low amounts (picomoles) of the peptide mixture applied to the LC instrument still overwhelm the mass spectrometer mass-analyzer(s) and detector.

In addition, broad expression proteomic approach provides such a complex set of information that it is usually very difficult to interpret and put in a more meaningful perspective regarding the parasite biology and pathogenicity. For instance, the comprehensive survey of proteins expressed by the whole cell does not add any particular information about the specific subcellular localization of the identified proteins, especially of those annotated as hypothetical proteins or proteins with unknown homology. To understand the function of these proteins without performing any additional functional proteomic analysis, it would be extremely useful to acquire information about their subcellular localization. The rationale behind this idea is that the subcellular localization of proteins is intimately related to their function.

Therefore, we believe that a focused expression, proteomic approach can be much more informative than a broad proteomic analysis. In this regard, our major efforts have been focused at the comprehensive expression proteomic analysis of parasite-enriched organelles, compartments, or subcellular fractions. For instance, more recently, a detailed proteomic analysis of vesicles secreted by mammalian cell-derived trypomastigote stage of *T. cruzi* (TcVes) has been carried out (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data). These vesicles were described years ago by Goncalves *et al.* (35), but their biological role and biogenesis thus far remain largely unknown. More recently, it has been shown that these trypomastigote-derived vesicles strongly trigger proinflammatory response in murine macrophages via a Toll-like receptor 2 (TLR2)-mediated pathway (A.C.T. Torrecilhas, I.C. Almeida *et al.*,

unpublished data). In addition, it has been demonstrated that these vesicles considerably enhance parasite infectivity *in vitro* by engaging TLR2. Therefore, we believe that a comprehensive expression proteomic analysis of TcVes will certainly increase our understanding of their origin and biological significance. For carrying out these analyses, TcVes were fractionated by gel-filtration chromatography following ultracentrifugation. Enriched TcVes were then obtained by affinity chromatography with immobilized anti- $\alpha$ -Gal antibodies, purified from chronic Chagasic patients (28). For protein identification, enriched TcVes were digested using three different proteolysis protocols to increase peptide identification coverage: a) trypsin in the presence of urea; b) double digestion with trypsin and Glu-C endoproteinase; and c) trypsin in the presence of 60% methanol. Released peptides were then analyzed by LC-MS/MS (36). Using this approach, 110 proteins were identified (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data), with a false-positive rate of approximately 5% at the protein level (~0-4% false-positive rate at the peptide level), according to statistical models for accurate peptide identification (12, 13). About one-half of these proteins are members of the TS/gp85 superfamily (27). TS are enzymes that transfer sialic acid from host cells or host molecules to parasite glycoconjugates, the mucin-like glycoproteins or GPI-mucins (26, 27, 37). Sialic acid residues on trypomastigote GPI-mucins (tGPI-mucins) confer protection against lytic anti- $\alpha$ -Gal antibodies produced by chronic Chagasic patients (38) and play an important role in host-cell invasion (39). Other members of the TS/gp85 superfamily do not have enzymatic activity, but they are equally relevant because they are involved in host immunomodulation (40), cell attachment (41), and regulation of host complement system (42).

Another important class of identified proteins in the TcVes is the gp63 family. gp63 is the major surface protease of *Leishmania spp.*, and it seems to regulate host phagocytosis and the complement system (31). In addition, two proteins with high sequence similarity to *T. cruzi* mucin-like glycoproteins were identified. These glycoconjugates are the major surface macromolecules (29) and the principal target of lytic anti- $\alpha$ -Gal antibodies from chronic Chagasic patients (28). Most sequences of TS/gp85-superfamily members, gp63, and mucin-like proteins found in TcVes have a putative GPI-attachment signal peptide. It remains to be determined, however, whether the expressed proteins are indeed GPI-anchored. This is relevant for understanding the proinflammatory activity of TcVes. GPI-anchors from trypomastigote mucin-like glycoproteins (tGPIs) are known to induce a strong proinflammatory response in macrophages (43) via the activation of TLR2 (44).

Additionally, in this focused expression proteomic study of TcVes, several proteins were usually found in the exosomal proteome (45, 46), such as the structural proteins tubulin, kinesin and dynein, heat shock proteins (HSP70 and HSP90), metabolic proteins, and elongation factors (E.S. Nakayasu, A.C.T. Torrecilhas, F.C. Gozzo, L.L. Nohara, D.J. Lamont, M.J.M. Alves, M.A.J. Ferguson, and I.C. Almeida, unpublished data). Moreover, some proteins related to exosomal/vacuolar biogenesis (47, 48), such as heat shock protein 85 (HSP85), elongation factor 1-alpha, glyceraldehyde 3-phosphate dehydrogenase (GAPDH), vacuolar ATP synthase subunit B, and tetratricopeptide repeat (TPR) protein, have also been identified in TcVes. Exosomes are membrane vesicles released into the extracellular milieu by a variety of mammalian cells, including immune cells (e.g., B- and T-cells, dendritic cells (DCs)) (49). Some of their biological functions include: a) antigen presentation by antigen-presenting cells (APCs); b) transfer of MHC class I and II-peptide complexes between different DCs; c) anti-tumoral response by T-cell stimulation; and d) membrane exchange between cells. Exosomes act like a protein cargo to several immune cells, transporting signaling molecules, receptors, and antigen-presenting molecules (e.g., major histocompatibility complex (MHC)) and co-receptors. One of the most remarkable roles of exosomes is to instruct naïve immune cells with peptide-loaded MHCs class I and II. Most of these roles were determined by *in vitro* studies; therefore, their *in vivo* functions are still mostly unclear. It remains, however, to be established if these vesicles are in fact released *in vivo* by immune cells. The same question applies to the case of *T. cruzi* shed vesicles.

**Table 1.**  
***T. cruzi* proteins identified by proteomic analysis in different studies.**

Protein group	References							
	Paba <i>et al.</i> (21), Proteomics 2004	Parodi-Talice <i>et al.</i> (22), Int J Parasitol 2004	Paba <i>et al.</i> (25), J Proteome Res 2004	Atwood <i>et al.</i> (11), Science 2005				
	Trypo	Epi	Trypo/Ama	Ama	Trypo	Meta	Epi	Total
RHS	0	1	0	263	33	354	275	396
TS/gp85 superfamily	0	0	0	78	120	40	8	209
Ribosomal	0	0	1	130	125	88	149	154
Proteases (not including proteasome)	0	1	1	52	32	76	61	92
Proteasome/Ubiquitin	0	1	0	42	35	55	34	61
Heat shock/Chaperonins	6	10	7	46	38	49	50	54
Cytoskeleton	1	3	4	14	15	26	20	34
Histones	0	0	1	29	27	28	29	29
Elongation factors	2	0	1	27	20	28	26	28
Protein kinases	0	0	0	17	11	16	8	22
MASP	0	0	0	0	11	0	1	12
Protein phosphatases	0	9	0	2	5	8	1	9
Flagellum	2	0	2	4	7	8	7	8
Mucins	0	0	0	0	0	0	0	0
Hypothetical	0	1	0	459	405	733	500	1010
Others	8	19	24	413	311	555	404	666
<b>Total proteins</b>	<b>19</b>	<b>45</b>	<b>41</b>	<b>1576</b>	<b>1194</b>	<b>2064</b>	<b>1573</b>	<b>2784</b>

## Functional Proteomics in *T. cruzi*

It is unquestionable that the few expression proteomic studies carried out thus far in *T. cruzi* have somehow improved our understanding of the role of certain parasite proteins, especially of those located in specific parasite cell fractions (i.e., TcVes). This is valid particularly in the case of the identified proteins that are already annotated in databases as having putative (homologous) function(s). One of the major difficulties, however, is to determine the function of proteins that are annotated as hypothetical or with unknown function or homology. To validate the biological and pathophysiological significance of any parasite protein found by expression proteomics, therefore, a systematic functional proteomics approach needs to be carried out. Gene transfection, knock-out, and RNA interference (RNAi) approaches are of particular relevance to assess the biological roles of proteins identified by descriptive, expression proteomic analysis.

A very elegant example of parasite functional proteomics was given recently by Broadhead *et al.* (50) through the comprehensive proteomic study of the flagellum of *T. brucei*. Following the identification of flagellar proteins by expression proteomic analysis, these authors carried out an RNAi-based survey of selected proteins and established that the flagellar function was vital for the bloodstream forms of *T. brucei*. This conclusion was based in the observation that RNAi-mediated disruption of the expression of key flagellar proteins led to a swift and noticeable collapse of cytokinesis in the bloodstream but not in insect-derived procyclic forms. The authors

proposed that the trypanosome flagellar proteins are targets for the development of new therapeutic approaches against *T. brucei*.

This kind of functional proteomic approach would be extremely enlightening if carried out in *T. cruzi*. Nevertheless, the RNAi approach seems not to work in *T. cruzi*, because the parasite is deficient in one or more of the constituents required for the RNAi pathway (51). Obviously, an effective alternative approach for RNAi would be the knock out of selected proteins identified by expression proteomics. Disappointingly, because of the technical limitations, this technique has not been used in *T. cruzi* as a molecular genetic tool to assess function of proteins identified by descriptive proteomic analysis.

To date, there is just one single functional proteomic study carried out in *T. cruzi*. In this study, Buscaglia *et al.* (29), using both expression and functional proteomic approaches, demonstrated that different members of the *T. cruzi* mucin (TcMUC) family, particularly those belonging to the TcMUC II group, which comprises 582 genes and 159 pseudogenes (33), are concomitantly expressed by cell-derived trypomastigotes. Moreover, the authors used several biochemical and immunological approaches to show that the TcMUC II group of genes codes for the majority of mucins of the infective trypomastigote stage, and these molecules are the major glycoproteins covering the parasite surface. *T. cruzi* mucins display a variable, non-repetitive, highly *O*-glycosylated central domain, followed by a short conserved C-terminus and a GPI anchor. By heterologous expression (in *E. coli*) of some of the TcMUC II members identified by expression proteomic analysis, it was also demonstrated that the short and conserved C-terminal peptide of TcMUC II mucins, but not their extensive and variable central domain, elicited strong antibody responses in patients with Chagas' disease and experimentally infected animals. In conclusion, this study suggested that the highly variable central domain found in all members of the TcMUC II group could be used as a decoy by the parasite to avoid the host immune defense.

In summary, as discussed briefly in this chapter, proteomic analysis of *T. cruzi*, particularly through focused expression or functional proteomic approach, may provide new insights about the localization, and biological and/or pathophysiological role(s) of key parasite proteins. Eventually, these newly identified proteins could be explored as molecular targets for the development of novel therapeutic approaches to treat or prevent Chagas' disease, which still severely affects millions of individuals throughout Latin America.

## Acknowledgments

I.C.A. is supported by Grants R01AI070655 and 5G12RR008124 from the National Institute of Allergy and Infectious Diseases and the National Center for Research Resources, respectively, both components of the National Institutes of Health. The content of this study is solely the responsibility of the authors and does not necessarily represent the official views of the National Institute of Allergy and Infectious Diseases, the National Center for Research Resources, or the National Institutes of Health. E.S.N. is supported by a fellowship from the Graduate School, University of Texas at El Paso, and NIH Grant R01AI070655 (to I.C.A.) . We thank Tiago Sobreira for critical reading of the manuscript.

## References

1. Henzel WJ, Billeci TM, Stults JT, Wong SC, Grimley C, Watanabe C. Identifying proteins from two-dimensional gels by molecular mass searching of peptide fragments in protein sequence databases. *Proc Natl Acad Sci U S A.* 1993;90:5011–5015. PubMed PMID: 8506346.
2. Eng JK, McCormack AL, Yates JR. An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J Am Soc Mass Spectrom.* 1994;5:976–989.
3. Perkins DN, Pappin DJ, Creasy DM, Cottrell JS. Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis.* 1999;20:3551–3567. PubMed PMID: 10612281.

4. Colinge J, Masselot A, Cusin I, Mahe E, Niknejad A, Argoud-Puy G, Reffas S, Bederr N, Gleizes A, Rey PA, Bougueleret L. High-performance peptide identification by tandem mass spectrometry allows reliable automatic data processing in proteomics. *Proteomics*. 2004;4:1977–1984. PubMed PMID: 15221758.
5. Colinge J, Masselot A, Giron M, Dessingy T, Magnin J. OLAV: towards high-throughput tandem mass spectrometry data identification. *Proteomics*. 2003;3:1454–1463. PubMed PMID: 12923771.
6. Kapp EA, Schutz F, Connolly LM, Chakel JA, Meza JE, Miller CA, Fenyo D, Eng JK, Adkins JN, Omenn GS, Simpson RJ. An evaluation, comparison, and accurate benchmarking of several publicly available MS/MS search algorithms: sensitivity and specificity analysis. *Proteomics*. 2005;5:3475–3490. PubMed PMID: 16047398.
7. Heller M, Ye M, Michel PE, Morier P, Stalder D, Junger MA, Aebersold R, Reymond F, Rossier JS. Added value for tandem mass spectrometry shotgun proteomics data validation through isoelectric focusing of peptides. *J Proteome Res*. 2005;4:2273–2282. PubMed PMID: 16335976.
8. Kuster B, Schirle M, Mallick P, Aebersold R. Scoring proteomes with proteotypic peptide probes. *Nat Rev Mol Cell Biol*. 2005;6:577–583. PubMed PMID: 15957003.
9. Peng J, Elias JE, Thoreen CC, Licklider LJ, Gygi SP. Evaluation of multidimensional chromatography coupled with tandem mass spectrometry (LC/LC-MS/MS) for large-scale protein analysis: the yeast proteome. *J Proteome Res*. 2003;2:43–50. PubMed PMID: 12643542.
10. Weatherly DB, Astwood JA, Minning TA, Cavola C, Tarleton RL, Orlando R. A Heuristic method for assigning a false-discovery rate for protein identifications from Mascot database search results. *Mol Cell Proteomics*. 2005;4:762–772. PubMed PMID: 15703444.
11. Atwood JA, Weatherly DB, Minning TA, Bundy B, Cavola C, Opperdoes FR, Orlando R, Tarleton RL. The *Trypanosoma cruzi* proteome. *Science*. 2005;309:473–476. PubMed PMID: 16020736.
12. Keller A, Nesvizhskii AI, Kolker E, Aebersold R. Empirical statistical model to estimate the accuracy of peptide identifications made by MS/MS and database search. *Anal Chem*. 2002;74:5383–5392. PubMed PMID: 12403597.
13. Nesvizhskii AI, Keller A, Kolker E, Aebersold R. A statistical model for identifying proteins by tandem mass spectrometry. *Anal Chem*. 2003;75:4646–4658. PubMed PMID: 14632076.
14. Ong SE, Mann M. Mass spectrometry-based proteomics turns quantitative. *Nat Chem Biol*. 2005;1:252–262. PubMed PMID: 16408053.
15. Han DK, Eng J, Zhou H, Aebersold R. Quantitative profiling of differentiation-induced microsomal proteins using isotope-coded affinity tags and mass spectrometry. *Nat Biotechnol*. 2001;19:946–951. PubMed PMID: 11581660.
16. Li XJ, Zhang H, Ranish JA, Aebersold R. Automated statistical analysis of protein abundance ratios from data generated by stable-isotope dilution and tandem mass spectrometry. *Anal Chem*. 2003;75:6648–6657. PubMed PMID: 14640741.
17. Ishihama Y, Oda Y, Tabata T, Sato T, Nagasu T, Rappsilber J, Mann M. Exponentially modified protein abundance index (emPAI) for estimation of absolute protein amount in proteomics by the number of sequenced peptides per protein. *Mol Cell Proteomics*. 2005;4:1265–1272. PubMed PMID: 15958392.
18. Liu H, Sadygov RG, Yates JR. A model for random sampling and estimation of relative protein abundance in shotgun proteomics. *Anal Chem*. 2004;76:4193–4201. PubMed PMID: 15253663.
19. Barrett J, Jefferies JR, Brophy PM. Parasite proteomics. *Parasitol Today*. 2000;16:400–403. PubMed PMID: 10951601.
20. Biron DG, Moura H, Marche L, Hughes AL, Thomas F. Towards a new conceptual approach to "parasitoproteomics". *Trends Parasitol*. 2005;21:162–168. PubMed PMID: 15780837.
21. Paba J, Santana JM, Teixeira AR, Fontes W, Sousa MV, Ricart CA. Proteomic analysis of the human pathogen *Trypanosoma cruzi*. *Proteomics*. 2004;4:1052–1059. PubMed PMID: 15048986.
22. Parodi-Talice A, Duran R, Arrambide N, Prieto V, Pineyro MD, Pritsch O, Cayota A, Cervenansky C, Robello C. Proteome analysis of the causative agent of Chagas disease: *Trypanosoma cruzi*. *Int J Parasitol*. 2004;34:881–886. PubMed PMID: 15217726.

23. Gygi SP, Rist B, Gerber SA, Turecek F, Gelb MH, Aebersold R. Quantitative analysis of complex protein mixtures using isotope-coded affinity tags. *Nat Biotechnol.* 1999;17:994–999. PubMed PMID: 10504701.
24. Gygi SP, Rist B, Griffin TJ, Eng J, Aebersold R. Proteome analysis of low-abundance proteins using multidimensional chromatography and isotope-coded affinity tags. *J Proteome Res.* 2002;1:47–54. PubMed PMID: 12643526.
25. Paba J, Ricart CA, Fontes W, Santana JM, Teixeira AR, Marchese J, Williamson B, Hunt T, Karger BL, Sousa MV. Proteomic analysis of *Trypanosoma cruzi* developmental stages using isotope-coded affinity tag reagents. *J Proteome Res.* 2004;3:517–524. PubMed PMID: 15253433.
26. Buscaglia CA, Campo VA, Frasch AC, Di Noia JM. *Trypanosoma cruzi* surface mucins: host-dependent coat diversity. *Nat Rev Microbiol.* 2006;4:229–236. PubMed PMID: 16489349.
27. Frasch AC. Functional diversity in the trans-sialidase and mucin families in *Trypanosoma cruzi*. *Parasitol Today.* 2000;16:282–286. PubMed PMID: 10858646.
28. Almeida IC, Ferguson MA, Schenkman S, Travassos LR. Lytic anti-alpha-galactosyl antibodies from patients with chronic Chagas' disease recognize novel O-linked oligosaccharides on mucin-like glycosyl-phosphatidylinositol-anchored glycoproteins of *Trypanosoma cruzi*. *Biochem J.* 1994;304(Pt 3):793–802. PubMed PMID: 7818483.
29. Buscaglia CA, Campo VA, Di Noia JM, Torrecilhas AC, De Marchi CR, Ferguson MA, Frasch AC, Almeida IC. The surface coat of the mammal-dwelling infective trypomastigote stage of *Trypanosoma cruzi* is formed by highly diverse immunogenic mucins. *J Biol Chem.* 2004;279:15860–15869. PubMed PMID: 14749325.
30. Neira I, Silva FA, Cortez M, Yoshida N. Involvement of *Trypanosoma cruzi* metacyclic trypomastigote surface molecule gp82 in adhesion to gastric mucin and invasion of epithelial cells. *Infect Immun.* 2003;71:557–561. PubMed PMID: 12496211.
31. Yao C, Donelson JE, Wilson ME. The major surface protease (MSP or GP63) of *Leishmania* sp. Biosynthesis, regulation of expression, and function. *Mol Biochem Parasitol.* 2003;132:1–16. PubMed PMID: 14563532.
32. Cuevas IC, Cazzulo JJ, Sanchez DO. gp63 homologues in *Trypanosoma cruzi*: surface antigens with metalloprotease activity and a possible role in host cell infection. *Infect Immun.* 2003;71:5739–5749. PubMed PMID: 14500495.
33. El-Sayed NM, Myler PJ, Bartholomeu DC, Nilsson D, Aggarwal G, Tran AN, Ghedin E, Wortley EA, Delcher AL, Blandin G, Westenberger SJ, Caler E, Cerqueira GC, Branche C, Haas B, Anupama A, Arner E, Aslund L, Attipoe P, Bontempi E, Bringaud F, Burton P, Cadag E, Campbell DA, Carrington M, Crabtree J, Darban H, da Silveira JF, de Jong P, Edwards K, Englund PT, Fazelina G, Feldblyum T, Ferella M, Frasch AC, Gull K, Horn D, Hou L, Huang Y, Kindlund E, Klingbeil M, Kluge S, Koo H, Lacerda D, Levin MJ, Lorenzi H, Louie T, Machado CR, McCulloch R, McKenna A, Mizuno Y, Mottram JC, Nelson S, Ochaya S, Osoegawa K, Pai G, Parsons M, Pentony M, Pettersson U, Pop M, Ramirez JL, Rinta J, Robertson L, Salzberg SL, Sanchez DO, Seyler A, Sharma R, Shetty J, Simpson AJ, Sisk E, Tammi MT, Tarleton R, Teixeira S, Van Aken S, Vogt C, Ward PN, Wickstead B, Wortman J, White O, Fraser CM, Stuart KD, Andersson B. The genome sequence of *Trypanosoma cruzi*, etiologic agent of Chagas disease. *Science.* 2005;309:409–415. PubMed PMID: 16020725.
34. Gygi SP, Corthals GL, Zhang Y, Rochon Y, Aebersold R. Evaluation of two-dimensional gel electrophoresis-based proteome analysis technology. *Proc Natl Acad Sci U S A.* 2000;97:9390–9395. PubMed PMID: 10920198.
35. Goncalves MF, Umezawa ES, Katzin AM, de Souza W, Alves MJ, Zingales B, Colli W. *Trypanosoma cruzi*: shedding of surface antigens as membrane vesicles. *Exp Parasitol.* 1991;72:43–53. PubMed PMID: 1993464.
36. Yates JR. Mass spectral analysis in proteomics. *Annu Rev Biophys Biomol Struct.* 2004;33:297–316. PubMed PMID: 15139815.
37. Schenkman S, Eichinger D, Pereira ME, Nussenzweig V. Structural and functional properties of *Trypanosoma* trans-sialidase. *Annu Rev Microbiol.* 1994;48:499–523. PubMed PMID: 7826016.
38. Pereira-Chioccola VL, Acosta-Serrano A, Correia de Almeida I, Ferguson MA, Souto-Padron T, Rodrigues MM, Travassos LR, Schenkman S. Mucin-like molecules form a negatively charged coat that protects



- Trypanosoma cruzi* trypomastigotes from killing by human anti-alpha-galactosyl antibodies. *J Cell Sci.* 2000;113(Pt 7):1299–1307. PubMed PMID: 10704380.
39. Schenkman S, Robbins ES, Nussenzweig V. Attachment of *Trypanosoma cruzi* to mammalian cells requires parasite energy, and invasion can be independent of the target cell cytoskeleton. *Infect Immun.* 1991;59:645–654. PubMed PMID: 1987081.
  40. Todeschini AR, Girard MF, Wieruszkeski JM, Nunes MP, DosReis GA, Mendonca-Previato L, Previato JO. trans-Sialidase from *Trypanosoma cruzi* binds host T-lymphocytes in a lectin manner. *J Biol Chem.* 2002;277:45962–45968. PubMed PMID: 12237289.
  41. Magdesian MH, Giordano R, Ulrich H, Juliano MA, Juliano L, Schumacher RI, Colli W, Alves MJ. Infection by *Trypanosoma cruzi*. Identification of a parasite ligand and its host cell receptor. *J Biol Chem.* 2001;276:19382–19389. PubMed PMID: 11278913.
  42. Norris KA. Ligand-binding renders the 160 kDa *Trypanosoma cruzi* complement regulatory protein susceptible to proteolytic cleavage. *Microb Pathog.* 1996;21:235–248. PubMed PMID: 8905613.
  43. Almeida IC, Camargo MM, Procopio DO, Silva LS, Mehlert A, Travassos LR, Gazzinelli RT, Ferguson MA. Highly purified glycosylphosphatidylinositols from *Trypanosoma cruzi* are potent proinflammatory agents. *EMBO J.* 2000;19:1476–1485. PubMed PMID: 10747016.
  44. Campos MA, Almeida IC, Takeuchi O, Akira S, Valente EP, Procopio DO, Travassos LR, Smith JA, Golenbock DT, Gazzinelli RT. Activation of Toll-like receptor-2 by glycosylphosphatidylinositol anchors from a protozoan parasite. *J Immunol.* 2001;167:416–423. PubMed PMID: 11418678.
  45. Mears R, Craven RA, Hanrahan S, Totty N, Upton C, Young SL, Patel P, Selby PJ, Banks RE. Proteomic analysis of melanoma-derived exosomes by two-dimensional polyacrylamide gel electrophoresis and mass spectrometry. *Proteomics.* 2004;4:4019–4031. PubMed PMID: 15478216.
  46. Wubbolts R, Leckie RS, Veenhuizen PT, Schwarzmann G, Mobius W, Hoernschemeyer J, Slot JW, Geuze HJ, Stoorvogel W. Proteomic and biochemical analyses of human B cell-derived exosomes. Potential implications for their function and multivesicular body formation. *J Biol Chem.* 2003;278:10963–10972. PubMed PMID: 12519789.
  47. Fevrier B, Raposo G. Exosomes: endosomal-derived vesicles shipping extracellular messages. *Curr Opin Cell Biol.* 2004;16:415–421. PubMed PMID: 15261674.
  48. Takegawa K, Iwaki T, Fujita Y, Morita T, Hosomi A, Tanaka N. Vesicle-mediated protein transport pathways to the vacuole in *Schizosaccharomyces pombe*. *Cell Struct Funct.* 2003;28:399–417. PubMed PMID: 14745133.
  49. Thery C, Zitvogel L, Amigorena S. Exosomes: composition, biogenesis and function. *Nat Rev Immunol.* 2002;2:569–579. PubMed PMID: 12154376.
  50. Broadhead R, Dawe HR, Farr H, Griffiths S, Hart SR, Portman N, Shaw MK, Ginger ML, Gaskell SJ, McKean PG, Gull K. Flagellar motility is required for the viability of the bloodstream trypanosome. *Nature.* 2006;440:224–227. PubMed PMID: 16525475.
  51. DaRocha WD, Otsu K, Teixeira SM, Donelson JE. Tests of cytoplasmic RNA interference (RNAi) and construction of a tetracycline-inducible T7 promoter system in *Trypanosoma cruzi*. *Mol Biochem Parasitol.* 2004;133:175–186. PubMed PMID: 14698430.



## Chapter B06. Expressed Sequence Tags (ESTs) and Gene Discovery: *Schistosoma mansoni*

Ricardo DeMarco<sup>1</sup> and Sergio Verjovski-Almeida<sup>2</sup>

Created: October 12, 2006; Updated: May 11, 2007.

This chapter describes the current efforts in gene discovery in *Schistosoma mansoni* and the importance of bioinformatics analysis to handle the large body of data that has been generated in the last few years. We also illustrate how the use of several common tools can help to obtain interesting results and how these results lead to further hypotheses regarding the molecular mechanisms underlying the biology of such a complex parasite, which can be the object of future experimentation in the field.

### Estimation of Transcriptome Complexity Using EST Information

One of the first challenges arising from conducting a large-scale analysis of the genome or transcriptome of any organism is to obtain a general description of its gene complement. This is of interest because it is generally accepted that the number of biochemical events in a cell can increase in direct proportion to the number of genes. However, the relationship between gene number and number of different cell types present in a multicellular organism is not well understood (1). Knowledge accumulated thus far permitted only a rough correlation between gene number and organism complexity. The comparative study of transcriptomes from different tissues or life stages from an organism permits an inference of spatial or temporal regulation of gene expression, which may hold some additional clues to the relation between complexity and genetic content. A better understanding of how complexity arises in biological systems should only be reached by comparing a number of most diverse organisms and by the combined input of information describing the gene content plus regulation of gene expression. Additionally, the study of such issues in parasites also raises questions regarding the influence of the parasitic lifestyle in the evolution of the gene content of the organism.

The recent effort to produce approximately 160,000 Expressed Sequence Tags (ESTs) throughout six different stages of *Schistosoma mansoni* life cycle (2) had the aim to complement previous efforts in gene discovery of *S. mansoni* that have produced approximately 16,000 ESTs and a few hundreds of full-length sequences with an estimated coverage of 15 to 20% of the *S. mansoni* transcriptome (3). The new dataset had obtained an estimated sampling of approximately 92% of *S. mansoni* genes, therefore permitting a general description of the *S. mansoni* transcriptome (2). The production of such a large body of data creates a challenge to manage in such a fashion as to generate a compact, accessible, and comprehensible output that allows the exploration of the resulting information with little amount of additional work.

The information obtained from EST sequence is intrinsically fragmentary, because the sources of information are discrete transcripts. As a result of the limitations of cloning and sequencing techniques, only a fraction of the transcript is represented in the EST, making the data even more fragmentary. Moreover, the distribution of messages in the transcriptome is biased, with a few messages represented by thousands of transcript molecules in each cell and hundreds of messages represented by few transcript molecules, which creates redundancy of information collected for abundant messages and scarcity of information for rare messages.

The method of choice to reduce the redundancy of data and to consolidate the transcriptome information is through *in silico* sequence assembly, using DNA sequence assembling programs. Currently, two assemblies of *S. mansoni* EST sequences are publicly available, one resulting from assembly of approximately 125,000 valid EST reads produced by the *Schistosoma mansoni* EST Genome Project (2) using the CAP3 program (4), and the

second is the *S. mansoni* Gene Index produced by TIGR, which uses a combination of alignment and assembly programs (5) to assemble approximately 138,000 ESTs and 400 expressed transcripts of *S. mansoni* (release 5.0 from November 2003; databases can be accessed at <http://bioinfo.iq.usp.br/schisto/> and <http://www.tigr.org/tdb/tgi/index.shtml>, respectively). Despite the different strategies adopted, both approaches produced approximately 30,000 unique cluster sequences (including both singlets and contigs), which represent a condensed form of the information generated by the *Schistosoma* EST sequencing projects. These two databases represent valuable repositories of information for the study of the molecular biology of the parasite, and examples of their application will be shown in the next sections of this chapter.

Although these unique sequences represent a reduced representative set of the information available, the number of unique sequences cannot be interpreted as the number of genes of an organism for two reasons. First, the assembly programs are prone to errors in their assemblies because of bad quality sequences, existence of chimerical clones, and alternative splicing of genes. Second, the fragmentary nature of EST sequencing may generate unique sequences from distinct regions of the same transcripts with no overlapping. Thus, the same transcript will be represented by two distinct unique sequences, and only the acquisition of additional sequence data would be able to join the fragments. In fact, it is expected that existing *S. mansoni* transcriptome assemblies have, on average, more than one unique sequence representing a gene; otherwise, *S. mansoni* would have approximately twice as many genes as well-described invertebrate organisms such as *Drosophila melanogaster* (13,601 genes) (6) and *Caenorhabditis elegans* (18,891 genes) (7).

Using the available EST information, two different bioinformatics approaches were used to estimate the number of genes in *S. mansoni* (2). The first approach was based on the concept that one can map the amino acid translation of coding unique sequences from distinct parts of the same *S. mansoni* gene onto the sequence of an ortholog protein from another organism using the BLASTx program (8) and then assume that all unique *S. mansoni* sequences mapped onto the same orthologous protein would actually represent only one *Schistosoma* gene. For this purpose, a reduced protein database was produced to permit the mapping of *S. mansoni* unique sequences to a non-redundant dataset. To build this reduced protein dataset, we started by collecting all proteins of the complete Swiss-Prot database whose sequences were the first hit of any *S. mansoni* EST cluster, in a BLASTX search that included as queries all ~30,000 *S. mansoni* EST clusters obtained from the assembly of the approximately 125,000 valid EST reads produced by the *Schistosoma mansoni* EST Genome Project (2). To further reduce this protein dataset, all collected protein sequences were compared against themselves using the BLASTp program, grouped by their hits, and only the longest member was selected as the prototype of each group in the database (the other members were discarded).

A second round of BLASTx search, again using all *S. mansoni* EST clusters as queries, now using the reduced protein sequences database as subjects showed that on average 1.9 unique *S. mansoni* EST cluster sequences did map to each subject protein. Assuming that the redundancy of EST clusters from *S. mansoni* genes with orthologs in Swiss-Prot is similar to the redundancy of genes without orthologs, one can extend this calculation into the portion of *S. mansoni* clusters that did not have matches in the reduced protein database. Thus, a factor of 1.9 can be applied to reduce the initial 30,000 clusters down to 15,789 clusters representative of the gene complement, which corrects for redundancy arising from separate non-overlapping portions of coding unique sequences.

However, this approach does not correct for redundant sequence fragments that eventually map to non-coding regions because there is no conservation of non-translated regions between organisms. To address this matter, an additional correction factor was considered to account for the existence of partial gene fragments corresponding to non-coding portions of EST cluster sequences. For this correction, we considered that the average distribution of EST clones representing the coding and non-coding regions of a gene was similar for all *S. mansoni* genes, irrespective of the fact that their full-length sequences was determined previous to our high-throughput EST sequencing project or not. A sample of 120 previously described full-length *S. mansoni* genes

was used as a template, onto which the existing unique EST cluster sequences were mapped. This permitted us to determine the proportion of clusters mapping to coding and non-coding portions of the messages. The ratio of clusters mapping to coding regions/total clusters was 0.8, indicating that most of the EST clones are located in the coding region, although a considerable sampling of 20% of non-coding region exists. A reducing factor of 0.8 was applied to the estimated 15,789 gene complement, thus giving a total of 12,631 estimated *S. mansoni* genes.

Another factor that has to be considered is based on the fact that the coverage of *S. mansoni* transcriptome by EST sequencing was partial, and there may be a considerable fraction of genes that have not been sampled. To estimate the fractional sampling of *S. mansoni* transcriptome, the ESTs were mapped to previously known *S. mansoni* full-length genes deposited in public databases. Because the full-length sequences deposited in GenBank were obtained by individual studies, it was assumed that it represented a mixture of genes with different abundances throughout the *S. mansoni* life cycle and could be considered as an unbiased sampling of the *S. mansoni* transcriptome. Using that model, it was estimated that 92% of the transcriptome was covered by EST sequencing based on the percentage of full-length genes displaying at least one EST mapped to its message.

Using all of the above data, it was possible to estimate the number of *S. mansoni* genes using the formula:

$$\text{Gene complement} = (\text{size of assembled dataset} \times f) / (c \times p)$$

f = Estimated fraction of unique cluster sequences containing coding regions

c = Estimated number of unique cluster sequences per gene p = Estimated coverage of *S. mansoni* transcriptome

The result of this calculation was an estimated 14,205 genes composing the *S. mansoni* transcriptome. This result is somewhat similar to the gene complement of other organisms with comparable complexity, such as *D. melanogaster* and *C. elegans*.

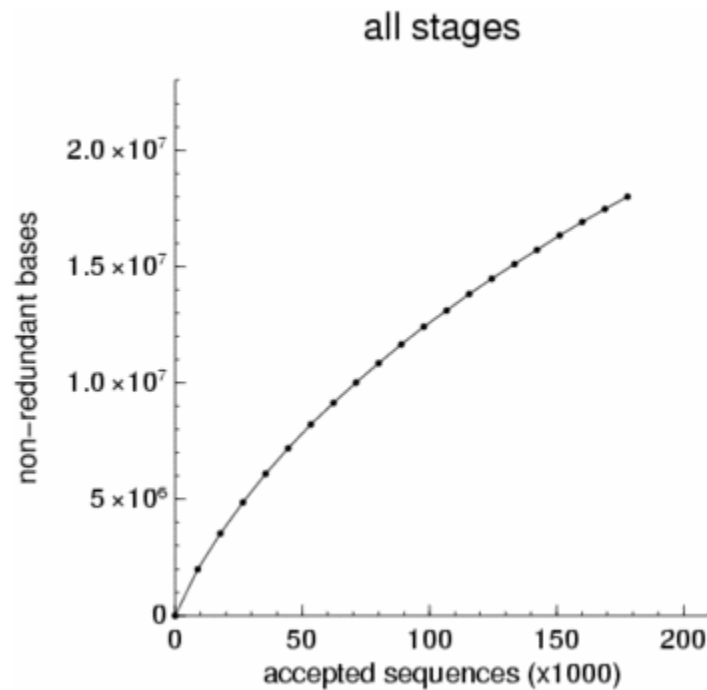
An independent approach that had been used to estimate the *S. mansoni* gene complement was based on the number of non-redundant bases accumulated as a function of the number of acquired EST sequences and showed a distinct pattern of saturation (Figure 1). The data obtained were extrapolated using a hyperbolic function, and it was possible to estimate that  $28.8 \pm 0.58$  million bases exist in the *S. mansoni* transcriptome.

To use this number to estimate the total number of genes, the average length of *S. mansoni* messages was estimated by averaging over all publicly available, full-length *S. mansoni* genes and all known *C. elegans* and *D. melanogaster* genes. Dividing the total number of non-redundant bases by the average length of *S. mansoni* genes, an estimated 13,960 genes was obtained. This number is very similar to that obtained with the method described previously.

An improved estimation of *S. mansoni* gene content will be obtained with the completion of the *S. mansoni* genome sequencing, which is currently under way (9). The combined use of *S. mansoni* transcriptome and genome information associated with algorithms of gene prediction will permit the deduction of new undescribed genes and the mapping of existing gene fragments.

## Annotation of Assembled Sequences and Its Use for Discovery of New Vaccine Candidates

*S. mansoni* is a complex, blood-dwelling human parasite, infecting millions of people in the developing world. The design of an effective vaccine is desirable because it would ideally lead to worldwide programs to prevent infection. In theory, protective vaccines should be more effective than drugs that require constant readministration to counteract morbidity (10). Recently, vaccination trials organized by the World Health Organization using six of the most promising antigens have been disappointing, particularly because none of the antigens achieved a protection of at least 40% against a challenge infection in animal models (11). Although research on these antigens has not been abandoned, the current situation is uncertain about the prospect for



**Figure 1.** Number of non-redundant bases as a function of the number of *S. mansoni* EST reads. To determine the total number of non-redundant bases acquired in the project, all accepted sequences were compared against each other with MegaBLAST. Any base appearing in multiple sequences in an overlapping segment was counted only once. The graph above reaches a plateau when the transcriptome of a cell or organism under study is completely covered by ESTs, using an ideally diverse cDNA library. Alternatively, it reaches a plateau when the particular libraries that are being sequenced become exhausted.

### SmAE GO browser

Enter GO number(s) or description keyword(s) to be searched, comma separated

Query

Expand  level(s) below the selection. Hide GOs without any contig assignment

#### Gene\_Ontology (GO:<go\_id>) (<n1> <n2>)

<n1> is the number of distinct consensus sequences [assembled sequences, SmAE sequences] that have the go\_id as ancestor in the GO directed acyclic graph.  
<n2> is the total number of SmAE sequences below go\_id in the GO directed acyclic graph. If a SmAE sequence is associated with another go\_id is a descendant of this go\_id, that sequence is counted twice.

#### Gene\_Ontology (GO:0003673) (8001\_29818)

##### molecular\_function (GO:0003674) (6614\_14573)

##### enzyme (GO:0003824) (3192\_7270)

##### hydrolase; EC: 3.-.-.- (GO:0016787) (1358\_2610)

##### hydrolase, acting on acid anhydrides; EC: 3.6.-.- (GO:0016817) (578\_704)

##### hydrolase, acting on acid anhydrides, in phosphorus-containing anhydrides; EC: 3.6.1.- (GO:0016818) (578\_699)

##### ATPase; synonym: ATP phosphohydrolase (GO:0016887) (462\_535)

##### ATP-binding cassette (ABC) transporter; EC: 3.6.3.- (GO:0004009) (115\_117)

##### ABC-type uptake permease; TC: 3.A.1.-.- (GO:0015406) (46\_46)

##### molybdate porter; EC: 3.6.3.2; TC: 3.A.1.8.1 (GO:0015412) (1\_1)

##### carbohydrate uptake transporter; TC: 3.A.1.1.-; synonym: sugar transporter (GO:0015608) (1\_1)

##### cation-transporting ATPase (GO:0019829) (43\_43)

##### hydrolase, acting on acid anhydrides, catalyzing transmembrane movement of substances; EC: 3.6.3.- (GO:0016820) (174\_192)

##### ATP-binding cassette (ABC) transporter; EC: 3.6.3.- (GO:0004009) (115\_117)

##### ABC-type uptake permease; TC: 3.A.1.-.- (GO:0015406) (46\_46)

##### molybdate porter; EC: 3.6.3.2; TC: 3.A.1.8.1 (GO:0015412) (1\_1)

##### carbohydrate uptake transporter; TC: 3.A.1.1.-; synonym: sugar transporter (GO:0015608) (1\_1)

##### cation-transporting ATPase (GO:0019829) (43\_43)

##### transporter (GO:0005215) (866\_1242)

##### carrier; synonym: carrier type transporter (GO:0005386) (356\_465)

##### primary active transporter (GO:0015399) (254\_348)

##### P-P-bond-hydrolysis-driven transporter; TC: 3.A.-.-.- (GO:0015405) (194\_248)

##### ATP-binding cassette (ABC) transporter; EC: 3.6.3.- (GO:0004009) (115\_117)

**Figure 2.** Screenshot of a *S. mansoni* GO browser (<http://bioinfo.iq.usp.br/schisto/>).

producing a vaccine against schistosomiasis (11). The recent release of extensive transcriptome data created an opportunity for the proposal of new vaccine candidates (2, 12, 13).

The set of predicted genes obtained from large-scale sequencing of simple pathogen organisms has been used to propose vaccine candidates. The rationale of these studies was to select potentially surface-exposed or exported proteins to use in additional trials (14, 15). Although a similar approach could be used for proposing new vaccine candidates for complex pathogens such as *S. mansoni*, it would not be as straightforward as for simpler pathogens because of the existence of several tissues and life stages. Therefore, a detailed and accurate description of each *S. mansoni* gene certainly is crucial for a rational proposal of vaccine candidates.

The first step to organize the EST data is the assembly of sequences to reduce redundancy of information, as discussed in the previous section. After that, it is necessary to add information to each assembled sequence to provide a description of the protein product of the gene. A simple way to provide a first description of the gene is through the use of the BLASTx program to compare the assembled sequence in its six possible translation frames to existing protein databases. The use of BLASTx to provide a first annotation of sequences is appealing because it does not require the deduction of the Open Reading Frame, being all six frames compared. In addition, the algorithm permits comparison against large databases (such as the GenBank non-redundant database) using a reasonable amount of computational resource. The two existing public assemblies for *S. mansoni* (<http://bioinfo.iq.usp.br/schisto/> and <http://www.tigr.org/tdb/e2k1/sma1/>) use the first BLASTx protein hit to provide a brief description of the assembled gene fragment. These annotations serve as descriptors for the sequence, permitting location of a gene of interest through the use of keyword searches.

Additional information must be added to each sequence to permit a better description of the gene product. For example, prediction of conserved domains, through search algorithm tools of existing databases such as SMART (16) and Pfam (17). These algorithms use protein sequences as input, and therefore it is necessary to use programs to predict the correct Open Reading frame. The *S. mansoni* EST genome project used the ESTscan program (18) to provide a putative translation of the gene fragment (if any).

Another interesting feature that has been added to both repositories of assembled sequences is the annotation with the use of Gene Ontology (GO) terms that use a controlled vocabulary to describe gene products in terms of their associated biological process, cellular components, or molecular function (19). Manual attribution of GO terms to a large repository of sequences is not an easy task, because it requires a large amount of work and expertise from annotators of different areas to perform a correct assignment for components of the most diverse pathways. Instead, what was done for *S. mansoni* databases was to annotate the gene fragments by automated processing using the BLASTx program to compare the existing gene fragments to the sequences of proteins annotated previously for the GO classification. The GO annotation of the best hit with an e-value lower than  $e^{-10}$  (or hits with more than 75% identity and 50% coverage in the case of TIGR gene index) was then transferred to the gene fragment annotation.

The use of a controlled GO vocabulary and the associated structures linking several GOs provide an interesting environment to perform searches for gene fragments that code for proteins of particular interest. However, because no curation has been performed, a significant portion of assignments may present errors, and therefore results must be interpreted with caution. Implementation of tools that permit browsing through the structure of ontologies provides useful ways to explore the different aspects of the parasite biology (Figure 2).

Description of secreted or surface-exposed proteins of *S. mansoni* should be of value, because these make good vaccine candidates due to their exposure to the host immune system. *In silico* predictors of cellular location use the protein signal sequences coded by the 5' portion of genes. However, the existing *S. mansoni* EST information is not particularly useful because most of the genes represented lack the 5' portion corresponding to the amino-terminal regions of the proteins. Because of the difficulties in using such an approach, a search for vaccine candidate genes was based mainly on the GO categorization and orthologs of secreted toxins, and surface

proteins were identified (2). Further experimentation is still necessary to confirm the external location of some of these candidates and to test the potential of these candidates as antigens.

Recently, a set of 20 *S. mansoni* clones was isolated by signal sequence trap, indicating that they may be exposed or secreted (20). Identification of products encoded by some of these clones was possible by comparison using the BLAST program with the existing database of ESTs, which permitted the extension of the original clone sequence and provided a better description of the coding region for the putative secreted/exposed protein.

Proteomic analysis of tegument extracts aiming to detect potential surface proteins has been done recently. The existing EST assemblies were used to match the resulting peptide fragments to the corresponding gene using BLAST searches (21, 22). The intrinsic fragmentary nature of proteomic data, which consist only of short stretches of peptides derived from analysis of mass spectra, calls for a dataset of transcripts to permit comparisons that link the fragmentary peptide data to a cDNA message by the use of bioinformatic tools (23). The existence of annotated databases, with defined, non-redundant gene fragments, certainly facilitates a fast and direct identification of detected proteins and corresponding genes.

## Annotation of Assembled Sequences and Its Use for Proposing New Drug Targets

Praziquantel, a heterocyclic pyrazino-isoquinolone, is the drug of choice for treatment of schistosomiasis and has been used in disease-control programs in several developing countries (24). The drug treatment, however, does not prevent re-infection of individuals in endemic areas, and emergence of drug-resistant strains has been documented (25). This fact raised concern in the scientific community and has placed the development of novel drugs as a necessary task to provide a substitute for praziquantel in the eventual emergence of resistant worms. Additional drugs would permit alternating drug regimens in mass treatments to decrease the chance of drug resistance (26).

Although the exact mechanism of action of praziquantel remains unknown (27), it has been described that the mode of action of praziquantel involves a disruption of  $\text{Ca}^{2+}$  homeostasis (reviewed by Greenberg (28)). Recently, two calcium channel beta subunits have been described in schistosomes (29, 30). *In silico* phylogenetic analysis comparison of these subunits with other organisms' beta subunits have shown that whereas one of them is a conventional invertebrate beta subunit, the other displays a unique structure that is not present in any other phyla (28). One of the features of these unique schistosoma proteins is a substitution in two conserved serines of the protein kinase C phosphorylation site in the beta interaction domain (BID) (30). Coexpression of these schistosome beta subunits with mammalian or cnidarian alpha 1 subunits results in a decrease in current amplitude, in opposition to the expected effect of beta subunit interaction. Moreover, coexpression of schistosome beta subunit with these otherwise praziquantel-insensitive calcium channels confers sensitivity to praziquantel, indicating that these beta subunits may be important molecular targets of praziquantel action (29). Additionally, it has been shown that association with rat calcium channel beta subunit mutated in the two conserved serines in the BID region can confer praziquantel sensitivity to a mammalian A1 subunit (30). This example shows the potential of an initial *in silico* approach to detect differences between parasite and host proteins that may influence in drugs sensibility, leading to more rational approach to drug improvement.

The large repository of genes produced by large-scale EST sequencing has been used to propose novel targets for drugs. One of the strategies used to assign novel drug targets was to search for proteins that have been shown to be targets of existing antihelminthics (2). Additionally, it was possible to propose novel drug targets based on a systematic search in the *S. mansoni* database for PFAM domains described to be present only in invertebrates. Using this approach, it was possible to point out innexins, which are proteins involved in the formation of gap junctions between cells, being exposed in parasite cells in an accessible location for eventual novel drugs (2). It is



expected that further studies of the novel genes of *S. mansoni* will point out novel drug targets, and additional biochemical studies will determine the feasibility of drug development for these targets.

## Use of Expression Data for Providing New Clues to Parasite Biology

*S. mansoni* has a complex life cycle that includes two, free-living stages (Cercaria and Miracidium) and two phases within hosts, one being the intermediate snail host, where asexual reproduction occurs, and the definitive human host, in which the worms differentiate according to their sex and mate to reproduce and are capable of surviving several years, despite the host defenses. This complex parasite displays a bilateral body plan, sexual dimorphism, and multiple organs. Considering the different environments to which the parasites are exposed, it is expected that the gene expression profile of the parasite should vary significantly during its life cycle, reflecting the dynamics of the change in the parasite's physiology.

An interesting aspect of large-scale study of ESTs generated from several tissues or developmental stages is that gene abundance is roughly correlated to representation in an EST database from an organism (31). In that way, it is possible to obtain information relative to abundance of different messages by computing the relative abundance of ESTs produced for each message in a given library of a specific life cycle stage. However, it is necessary to interpret such data with care, because it is known that efficiency of cloning varies from message to message. Moreover, use of normalization techniques in the construction of several cDNA libraries distorts the representation of each message in relation to its natural abundance.

Most of the *S. mansoni* EST data was derived from libraries produced in the *S. mansoni* EST Genome Project by using two different normalization techniques, the ORESTES approach, which relies on the low-stringency RT-PCR amplification of several targets using arbitrary primers (32), or reassociation kinetics based normalized libraries (33). Therefore, the use of *S. mansoni* EST data to estimate relative abundance of messages would need a more sophisticated approach than direct counting. In the *S. mansoni* EST Genome Project, approximately 300 different primers were used, one at a time, in the low-stringency RT-PCR reactions to construct the so-called ORESTES mini-libraries. By using the same primer and RT-PCR conditions but mRNA from different life cycle stages, one should obtain similar cDNA profiles, influenced only by the relative abundance of mRNA targets, which determines the extent of annealing of the same set of messages to the given arbitrary primer. It would be possible to use a specific subset of sequences derived from a given primer to obtain information related to relative abundance of genes. Indeed, the differential counting of sequence frequencies was analyzed with a randomization statistical method (34) that was applied to a subset of ORESTES sequences that were known to be derived from each of six primers, each one being used in the RT-PCR to sample, in parallel, different mRNAs from several stages of the parasite's life cycle. These analyses detected 82 transcripts with differential expression among stages (99.8% confidence) (2). This approach was validated experimentally using semi-quantitative RT-PCR (2).

Discovery of genes that display significant variation along the parasite's life cycle may point out processes that control the parasite's development or regulate adaptation of the parasite to the environment, which includes potential mechanisms of modulation of the host immune system. As described above, it is possible to obtain a rough correlation between the number of ESTs and gene expression for some genes of *S. mansoni* through the use of statistical approaches. In spite of the existing level of inaccuracy in this approach, it is an interesting method to point out some candidates for further investigation, rather than to provide a definitive list of stage-specific genes.

A more powerful approach to address this issue is the use of microarrays to generate information of differential expression. The production of ESTs and their organization in databases are a necessary step to permit the design of probes for the microarray platform. The design of microarrays requires the use of bioinformatics tools

because each platform contains thousands of probes, each representing, desirably, one gene. A rational selection of probes to be present in the platform relies on the annotation, thus permitting a selection of genes of interest. Besides, it is preferable to use assembled EST databases rather than individual EST reads to select probes to avoid excessive and unbalanced selection of redundant probes representing the same gene. Furthermore, the analysis of microarray data also relies on bioinformatics tools to permit data extraction from thousands of spots in a slide image, normalization of data, and statistical analysis to calculate the significance of differential data.

Recently, microarray experiments in *S. mansoni* have been published comparing the differential gene expression of male and females (35, 36). However, either a limited number of amplified clones were used (36), or oligonucleotide probes were designed using a limited dataset of ~16,000 ESTs that existed prior to the extensive sequencing of the *S. mansoni* transcriptome (35).

The current technology permits the construction of microarray slides containing tens of thousands of oligonucleotide probes, which outnumber the expected *S. mansoni* gene complement. It is expected that accumulation of novel sequence data, from *S. mansoni* transcriptome and genome, together with new bioinformatics analysis, will refine the current dataset of unique sequences, reducing the number of fragmented gene sequences, and increasing the number of genes represented in the database. That opens up the possibility for the construction of a whole transcriptome microarray platform that will permit comprehensive studies of the parasite's differential gene expression.

In addition, microarray experiments open the possibility to study the influence of diverse conditions on large-scale gene expression, such as exposure of the parasite to host signaling molecules, drugs, and stress environmental factors.

## **Schistosoma mansoni ATP-diphosphohydrolase: A Practical Example**

ATP-diphosphohydrolases are enzymes that hydrolyze a variety of nucleoside tri- and diphosphates with the release of orthophosphate, the activity of which is dependent on the presence of divalent cations such as calcium and magnesium. They have been shown to be involved in the regulation of inflammatory responses (37) and thromboregulation (38).

An ecto-ATP-diphosphohydrolase (ecto-apyrase) activity on the surface of *S. mansoni* was first described at the beginning of the 1990s in a work that measured both the ATPase and ADPase activities of the enzyme in a fraction of schistosoma tegument. That work proposed that ATP-diphosphohydrolase could regulate the concentration of purine nucleotides around the parasites and hence enable them to escape the host homeostasis by preventing ADP-induced platelet activation (39). Later, it was shown by Western blot that *S. mansoni* tegument extracts display a 63-kDa protein with immunological cross-reactivity with anti-potato apyrase antibodies and anti-*Toxoplasma gondii* nucleoside triphosphate hydrolase antibodies. Immunolocalization using these cross-reactive antibodies suggested that the detected protein should be located on the surface of the parasite (40). Cross-reactivity suggested that the *S. mansoni* protein responsible for the Ecto ATP-diphosphohydrolase activity should possess conserved epitopes with ortholog apyrases from other organisms. In fact, sequencing of three peptide fragments from potato apyrase or ATP-diphosphohydrolase, followed by BLASTx sequence searches in GenBank, permitted the identification of four full-length genes from four different organisms that coded for proteins constituting a new protein family (40), thus far undetected. The ATP-diphosphohydrolase family, also called the GDA1/CD39 nucleoside phosphatase family, displays five conserved sequence motifs (Pfam PF01150; InterPro IPR000407) as identified by multiple sequence alignments (40, 41).

Although these early studies provided an extensive biochemical characterization of the ecto-ATP-diphosphohydrolase activity of the parasite, neither the *S. mansoni* protein sequence nor the gene sequence that codes the protein responsible for this activity was described. Description of the gene and protein sequence was

necessary to permit further detailed studies of characterization of the protein activity and its role in *S. mansoni*. Because the *S. mansoni* ecto-ATP-diphosphohydrolase should contain the conserved amino acid motifs that are characteristic of the GDA1/CD39 family, several approaches using heterologous library screening probes or degenerate primers in connection with RT-PCR were used in the laboratory of the present authors to clone the gene. Unfortunately, the amino acid sequences of the conserved motifs are such that the resulting degenerate primers are extremely rich in G-C stretches, and all directed cloning strategies were without any success.

The large-scale sequencing of *S. mansoni* ESTs provided a new opportunity to obtain a fragment of the gene coding for an ATP-diphosphohydrolase and lead to new approaches to clone the full-length message for this protein. In fact, a BLASTx search of the *S. mansoni* EST Genome Project database detected a single EST coding for a protein similar to a human ATP-diphosphohydrolase (37% identity and 52% similarity over 124 amino acids). This sequence was used to design primers for rapid amplification of cDNA ends (RACE) to obtain the full-length sequence for the gene, which was named *S. mansoni* ATP-diphosphohydrolase-1 (ATPDase1). Cloning of the full-length sequence for the *S. mansoni* ATPDase1 permitted the heterologous expression of the hydrophilic portion of the protein. Using this recombinant protein as antigen, antibodies were produced and permitted the confirmation of the localization of this protein on the *S. mansoni* tegument membrane, exposed to the external surface (42).

Later proteomic studies of the parasite's tegument confirmed the localization of *S. mansoni* ATPDase1 in the membrane fraction of the tegument of adult worms (21, 22). These results were possible because the assembled EST database and the GenBank database contained the *S. mansoni* ATPDase1 partial and full-length gene sequences and permitted the identification of matching peptide fragments. In fact, this protein is one of the few detected in experiments of biotinylation of *S. mansoni* adult worms' surface, which were aimed at identifying exposed proteins (43). ATPDase1 is a potential vaccine candidate because of its permanent contact with the host immune system and its possible role as part of the parasite's escape mechanism from host defenses.

A search for the current information on this gene in public databases shows that the assembled sequence corresponding to ATPDase1 in the *S. mansoni* genome project (SmAE 608449.1) is composed of two reads and spans 407 bp of the gene. Nonetheless, it is annotated similarly to ecto-ATP-diphosphohydrolase II [*Homo sapiens*], the first hit with an e-value of  $3e-13$  in a BLASTx search. A PFAM domain of the GDA1/CD39 (nucleoside phosphatase) family was detected with an e-value of  $3.6e-17$ . Four GO terms were assigned: integral membrane protein, magnesium binding, apyrase, and hydrolase. In this particular case, all of the annotations are correct, and it would be easy to identify this protein either by a direct keyword search or a search for GO terms related with ATP-diphosphohydrolase. Annotation with PSORT for this fragment contains wrong predictions, of a cytoplasmatic protein with no transmembranes, because only a central fragment of the protein with no transmembrane segments is analyzed; the partial cDNA fragment coding for the central hydrophilic portion of the protein is clearly not an ideal subject for this kind of PSORT analysis.

The entry corresponding to ATPDase1 in the TIGR gene index (TC 11432) has the complete sequence of the gene, because the full-length deposited sequence was incorporated to the assembly along with 5 ESTs. It is annotated as similar to ectonucleoside triphosphate diphosphohydrolase 1 (NTPDase1) [*Bos taurus*], according to the results of a BLASTx search, which gives a partial coverage of the protein (12%). It is interesting to note that three ESTs for this gene were produced prior to the large-scale *S. mansoni* transcriptome sequencing effort; however, they all belong to the 3' UTR of the gene, thus precluding their prior identification as fragments of this gene. Only the two fragments produced during the large-scale sequencing effort are within the coding region and permitted identification of the gene family by similarity. On the basis of this example, both EST assembly databases seem to provide a good annotation for their sequences and to be a useful tool in gene discovery.

This exemplifies how the large-scale production of ESTs and the implementation of a comprehensive and annotated database may help in the discovery of new genes of interest. In this particular example, the external

location of the ATP-diphosphohydrolase makes it a potentially interesting protein to understand host–parasite relationships and a candidate for additional vaccine trials.

## Conclusions

Bioinformatics tools are essential to deal with the massive amount of EST data that have been generated for *S. mansoni*. Compilation of assembled sequence data as open and annotated databases enable access by the scientific community to the transcriptome content of the parasite, permitting further studies for characterization of novel genes. Large-scale applications, such as microarray and proteomic experiments, are now possible as a result of the access to such essential data that is needed for the design of experiments and interpretation of results. The first results derived from the access to such information are interesting, and it is expected that in the following years new concepts regarding the biology of *S. mansoni* and control of the disease should emerge from the efforts currently under way.

## References

1. Carroll SB. Chance and necessity: the evolution of morphological complexity and diversity. *Nature*. 2001;409:1102–1109. PubMed PMID: 11234024.
2. Verjovski-Almeida S, DeMarco R, Martins EA, Guimaraes PE, Ojopi EP, Paquola AC, Piazza JP, Nishiyama MY, Kitajima JP, Adamson RE. et al. Transcriptome analysis of the acoelomate human parasite *Schistosoma mansoni*. *Nat Genet*. 2003;35:148–157. PubMed PMID: 12973350.
3. Franco GR, Valadao AF, Azevedo V, Rabelo EM. The *Schistosoma* gene discovery program: state of the art. *Int J Parasitol*. 2000;30:453–463. PubMed PMID: 10731568.
4. Huang X, Madan A. CAP3: A DNA sequence assembly program. *Genome Res*. 1999;9:868–877. PubMed PMID: 10508846.
5. Lee Y, Tsai J, Sunkara S, Karamycheva S, Pertea G, Sultana R, Antonescu V, Chan A, Cheung F, Quackenbush J. The TIGR Gene Indices: clustering and assembling EST and known genes and integration with eukaryotic genomes. *Nucleic Acids Res*. 2005;33Database issueD71–D74. PubMed PMID: 15608288.
6. Adams MD, Celniker SE, Holt RA, Evans CA, Gocayne JD, Amanatides PG, Scherer SE, Li PW, Hoskins RA, Galle RF. et al. The genome sequence of *Drosophila melanogaster*. *Science*. 2000;287:2185–2195. PubMed PMID: 10731132.
7. The *C. elegans* Sequencing Consortium. Genome sequence of the nematode *C. elegans*: a platform for investigating biology. *Science*. 1998;282:2012–2018. PubMed PMID: 9851916.
8. Altschul SF, Madden TL, Schaffer AA, Zhang JH, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*. 1997;25:3389–3402. PubMed PMID: 9254694.
9. El-Sayed NM, Bartholomeu D, Ivens A, Johnston DA, LoVerde PT. Advances in schistosome genomics. *Trends Parasitol*. 2004;20:154–157. PubMed PMID: 15099549.
10. Hagan P, Doenhoff MJ, Wilson RA, Al-Sherbiny M, Bergquist R. Schistosomiasis vaccines: a response to a devils' advocate's view. *Parasitol Today*. 2000;16:322–323. PubMed PMID: 10900475.
11. Hagan P, Sharaf O. Schistosomiasis vaccines. *Expert Opin Biol Ther*. 2003;3:1271–1278. PubMed PMID: 14640953.
12. Hu W, Brindley PJ, McManus DP, Feng Z, Han ZG. Schistosome transcriptomes: new insights into the parasite and schistosomiasis. *Trends Mol Med*. 2004;10:217–225. PubMed PMID: 15121048.
13. Verjovski-Almeida S, Leite LC, Dias-Neto E, Menck C, Wilson RA. Schistosome transcriptome: insights and perspectives for functional genomics. *Trends Parasitol*. 2004;20:304–308. PubMed PMID: 15193558.
14. Maione D, Margarit I, Rinaudo CD, Massignani V, Mora M, Scarselli M, Tettelin H, Brettoni C, Iacobini ET, Rosini R, D'Agostino N, Miorin L, Buccato S, Mariani M, Galli G, Nogarotto R, Dei VN, Vegni F, Fraser C, Mancuso G, Teti G, Madoff LC, Paoletti LC, Rappuoli R, Kasper DL, Telford JL, Grandi G. Identification of

- a universal group B *Streptococcus* vaccine by multiple genome screen. *Science*. 2005;309:148–150. PubMed PMID: 15994562.
15. Pizza M, Scarlato V, Masignani V, Giuliani MM, Arico B, Comanducci M, Jennings GT, Baldi L, Bartolini E, Capecchi B, Galeotti CL, Luzzi E, Manetti R, Marchetti E, Mora M, Nuti S, Ratti G, Santini L, Savino S, Scarselli M, Storni E, Zuo P, Broecker M, Hundt E, Knapp B, Blair E, Mason T, Tettelin H, Hood DW, Jeffries AC, Saunders NJ, Granoff DM, Venter JC, Moxon ER, Grandi G, Rappuoli R. Identification of vaccine candidates against serogroup B meningococcus by whole-genome sequencing. *Science*. 2000;287:1816–1820. PubMed PMID: 10710308.
  16. Schultz J, Milpetz F, Bork P, Ponting CP. SMART, a simple modular architecture research tool: identification of signaling domains. *Proc Natl Acad Sci U S A*. 1998;95:5857–5864. PubMed PMID: 9600884.
  17. Bateman A, Birney E, Cerruti L, Durbin R, Eddy SR, Griffiths-Jones S, Howe KL, Marshall M, Sonnhammer EL. The Pfam protein families database. *Nucleic Acids Res*. 2002;30:276–280. PubMed PMID: 11752314.
  18. Iseli C, Jongeneel CV, Bucher P. ESTScan: a program for detecting, evaluating, and reconstructing potential coding regions in EST sequences. *Proc Int Conf Intell Syst Mol Biol* 1999; August 06-10, pp. 138–148.
  19. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet*. 2000;25:25–29. PubMed PMID: 10802651.
  20. Pearson MS, McManus DP, Smyth DJ, Lewis FA, Loukas A. In vitro and in silico analysis of signal peptides from the human blood fluke, *Schistosoma mansoni*. *FEMS Immunol Med Microbiol*. 2005;45:201–211. PubMed PMID: 16051070.
  21. Braschi S, Curwen RS, Ashton PD, Verjovski-Almeida S, Wilson RA. The tegument surface membranes of the human blood parasite *Schistosoma mansoni*: a proteomic analysis after differential extraction. *Proteomics*. 2006;6:1471–1482. PubMed PMID: 16447162.
  22. van Balkom BW, van Gestel RA, Brouwers JF, Krijgsveld J, Tielens AG, Heck AJ, van Hellemond JJ. Mass spectrometric analysis of the *Schistosoma mansoni* tegumental sub-proteome. *J Proteome Res*. 2005;4:958–966. PubMed PMID: 15952743.
  23. Ashton PD, Curwen RS, Wilson RA. Linking proteome and genome: how to identify parasite proteins. *Trends Parasitol*. 2001;17:198–202. PubMed PMID: 11282511.
  24. Hagan P, Appleton CC, Coles GC, Kusel JR, Tchuem-Tchuente LA. Schistosomiasis control: keep taking the tablets. *Trends Parasitol*. 2004;20:92–97. PubMed PMID: 14747023.
  25. Ismail M, Botros S, Metwally A, William S, Farghally A, Tao LF, Day TA, Bennett JL. Resistance to praziquantel: direct evidence from *Schistosoma mansoni* isolated from Egyptian villagers. *Am J Trop Med Hyg*. 1999;60:932–935. PubMed PMID: 10403323.
  26. Fenwick A, Savioli L, Engels D, Robert Bergquist N, Todd MH. Drugs for the control of parasitic diseases: current status and development in schistosomiasis. *Trends Parasitol*. 2003;19:509–515. PubMed PMID: 14580962.
  27. Ribeiro-Dos-Santos G, Verjovski-Almeida S, Leite LC. Schistosomiasis--a century searching for chemotherapeutic drugs. *Parasitol Res*. 2006;99:505–521. PubMed PMID: 16636847.
  28. Greenberg RM. Are Ca<sup>2+</sup> channels targets of praziquantel action? *Int J Parasitol*. 2005;35:1–9. PubMed PMID: 15619510.
  29. Kohn AB, Anderson PA, Roberts-Misterly JM, Greenberg RM. Schistosome calcium channel beta subunits. Unusual modulatory effects and potential role in the action of the antischistosomal drug praziquantel. *J Biol Chem*. 2001;276:36873–36876. PubMed PMID: 11500482.
  30. Kohn AB, Roberts-Misterly JM, Anderson PA, Greenberg RM. Creation by mutagenesis of a mammalian Ca(2+) channel beta subunit that confers praziquantel sensitivity to a mammalian Ca(2+) channel. *Int J Parasitol*. 2003;33:1303–1308. PubMed PMID: 14527513.

31. Zhang L, Zhou W, Velculescu VE, Kern SE, Hruban RH, Hamilton SR, Vogelstein B, Kinzler KW. Gene expression profiles in normal and cancer cells. *Science*. 1997;276:1268–1272. PubMed PMID: 9157888.
32. Dias Neto E, Correa RG, Verjovski-Almeida S, Briones MR, Nagai MA, da Silva W, Zago MA, Bordin S, Costa FF, Goldman GH, Carvalho AF, Matsukuma A, Baia GS, Simpson DH, Brunstein A, de Oliveira PS, Bucher P, Jongeneel CV, O'Hare MJ, Soares F, Brentani RR, Reis LF, de Souza SJ, Simpson AJ. Shotgun sequencing of the human transcriptome with ORF expressed sequence tags. *Proc Natl Acad Sci U S A*. 2000;97:3491–3496. PubMed PMID: 10737800.
33. Soares MB, Bonaldo MF, Jelene P, Su L, Lawton L, Efstratiadis A. Construction and characterization of a normalized cDNA library. *Proc Natl Acad Sci U S A*. 1994;91:9228–9232. PubMed PMID: 7937745.
34. Stekel DJ, Git Y, Falciani F. The comparison of gene expression from multiple cDNA libraries. *Genome Res*. 2000;10:2055–2061. PubMed PMID: 11116099.
35. Fitzpatrick JM, Johnston DA, Williams GW, Williams DJ, Freeman TC, Dunne DW, Hoffmann KF. An oligonucleotide microarray for transcriptome analysis of *Schistosoma mansoni* and its application/use to investigate gender-associated gene expression. *Mol Biochem Parasitol*. 2005;141:1–13. PubMed PMID: 15811522.
36. Hoffmann K.F., Johnston D.A., Dunne D.W. Identification of *Schistosoma mansoni* gender-associated gene transcripts by cDNA microarray profiling. *Genome Biol*. 2002;3:RESEARCH0041. PubMed PMID: 12186648.
37. Mizumoto N, Kumamoto T, Robson SC, Sevigny J, Matsue H, Enjyoji K, Takashima A. CD39 is the dominant Langerhans cell-associated ecto-NTPDase: modulatory roles in inflammation and immune responsiveness. *Nat Med*. 2002;8:358–365. PubMed PMID: 11927941.
38. Enjyoji K, Sevigny J, Lin Y, Frenette PS, Christie PD, Esch JS, Imai M, Edelberg JM, Rayburn H, Lech M, Beeler DL, Csizmadia E, Wagner DD, Robson SC, Rosenberg RD. Targeted disruption of cd39/ATP diphosphohydrolase results in disordered hemostasis and thromboregulation. *Nat Med*. 1999;5:1010–1017. PubMed PMID: 10470077.
39. Vasconcelos EG, Nascimento PS, Meirelles MN, Verjovski-Almeida S, Ferreira ST. Characterization and localization of an ATP-diphosphohydrolase on the external surface of the tegument of *Schistosoma mansoni*. *Mol Biochem Parasitol*. 1993;58:205–214. PubMed PMID: 8479445.
40. Vasconcelos EG, Ferreira ST, Carvalho T, deSouza W, Kettlun AM, Mancilla M, Valenzuela MA, Verjovski-Almeida S. Partial purification and immunohistochemical localization of ATP diphosphohydrolase from *Schistosoma mansoni*--Immunological cross reactivities with potato apyrase and *Toxoplasma gondii* nucleoside triphosphate hydrolase. *J Biol Chem*. 1996;271:22139–22145. PubMed PMID: 8703025.
41. Handa M, Guidotti G. Purification and cloning of a soluble ATP-diphosphohydrolase (apyrase) from potato tubers (*Solanum tuberosum*). *Biochem Biophys Res Commun*. 1996;218:916–923. PubMed PMID: 8579614.
42. DeMarco R, Kowaltowski AT, Mortara RA, Verjovski-Almeida S. Molecular characterization and immunolocalization of *Schistosoma mansoni* ATP-diphosphohydrolase. *Biochem Biophys Res Commun*. 2003;307:831–838. PubMed PMID: 12878186.
43. Braschi S, Wilson RA. Proteins exposed at the adult schistosome surface revealed by biotinylation. *Mol Cell Proteomics*. 2006;5:347–356. PubMed PMID: 16269422.

## Chapter B07. Alternative Splicing: Lessons from Cancer

Sandro J. de Souza<sup>1</sup> and Anamaria A. Camargo<sup>2</sup>

Created: October 12, 2006; Updated: September 17, 2007.

### Introduction

The last 15 years have witnessed an impressive development of a discipline that uses computational expertise and resources to approach biological problems. Bioinformatics, or computational biology as we know this discipline nowadays, is one of the pillars of biomedical sciences in the 21st century. Similar to what happened to molecular biology in the 1980s, it is expected that bioinformatics/computational biology will become a widespread discipline. Basically, all research groups will need to have their own computational expertise to explore the vast amount of data available in the public domain and to integrate that extracted knowledge into their own research program.

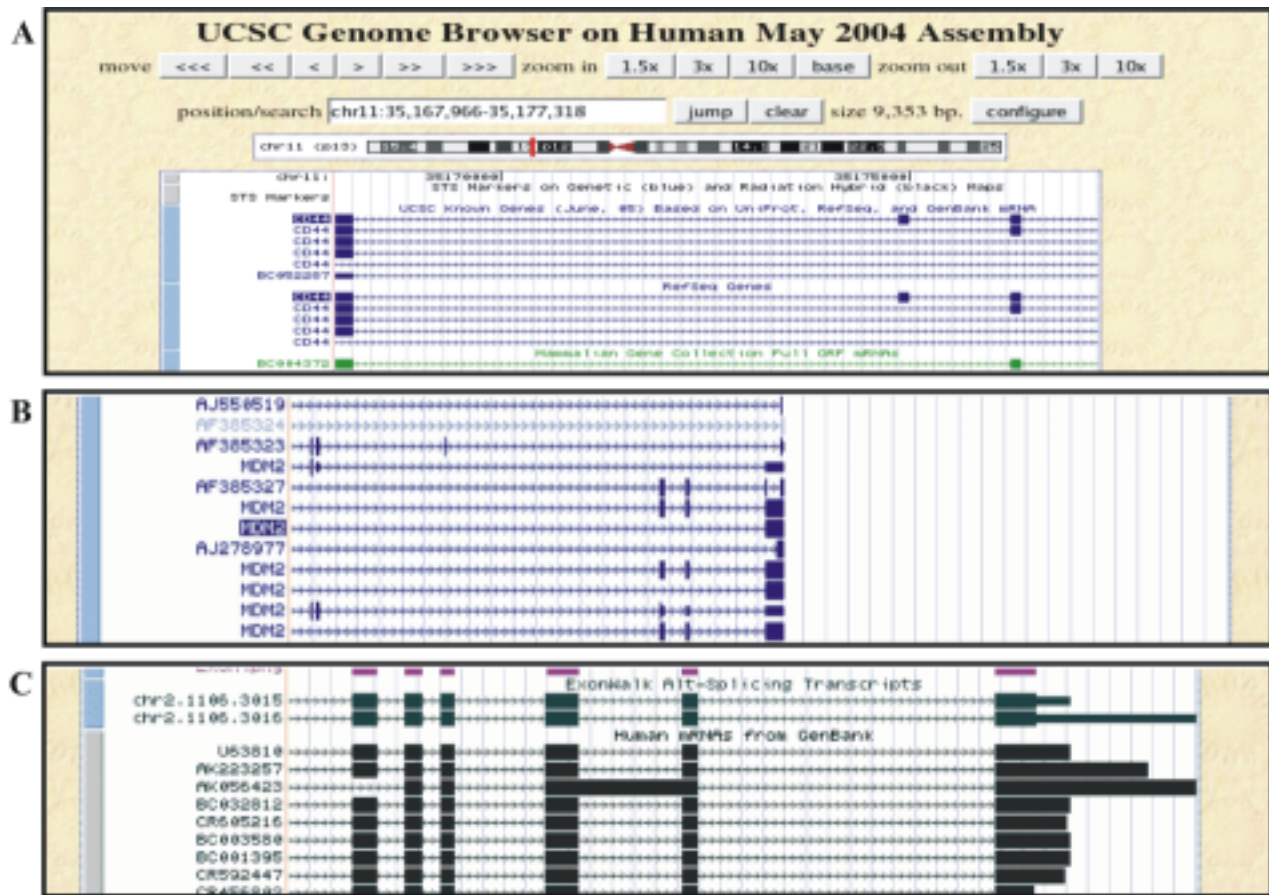
How did this happen? It was simply a consequence of molecular biology. First, the molecular biology revolution allowed the generation of huge amounts of data, illustrated by all the genome projects finished in the last 10 years. From the first moment, biologists needed computers to handle the data. More recently, biologists have used computational tools to make “experiments” and infer biological and functional significance from knowledge derived from *in silico* analyses. Comparative genomics is an example of such rationale.

One of the fields that has progressed more effectively under this new paradigm is transcriptomics. In the early 1990s, the first large-scale approach was used to generate expression data in the form of expressed sequence tags (ESTs) (1). As of June 2007, dbEST contains more than 43 million ESTs from a variety of organisms (dbEST release 062207: <http://www.ncbi.nlm.nih.gov/dbEST>). More recently, SAGE (2), MPSS (3), and microarray (4) technologies have added a more quantitative character to the collection of expression data. The completion of the Human Genome Project provided a scaffold onto which the expression data were mapped. The human genome sequence, now “decorated” with many different types of data, has become a driving resource in the biomedical sciences.

In this review, we will discuss how bioinformatics/computational biology and the genome sequence have been used by us and others to study the phenomenon of alternative splicing. We will give special emphasis to the association between alternative splicing and cancer.

### Alternative Splicing

The splicing of introns in eukaryotic genes is one of the most basic processes within the cell. We are still, however, far from a complete understanding of the regulatory mechanisms acting on splicing. From a chemical standpoint, splicing is very simple. It corresponds to two *trans*-esterification reactions that remove an intervening sequence (the intron) and join the two flanking exons. Biologically, however, splicing is a complex and intricate process. A huge ribonucleoproteic complex, the spliceosome (containing five RNAs and hundreds of proteins), is needed. In spite of the fact that introns are on average 10 times longer than exons, the spliceosome recognizes the correct intron/exon borders with an astonishing precision. This is achieved by a balanced mix of *cis* and *trans* elements. To make things even more complex, elements in *cis* are short and weak. Some of these sequence elements are present in basically all introns: a donor site (|GT), an acceptor site (AG|), a branch site, and a polypyrimidine tract (see Ast (5) and Woodley and Valcarcel (6) for reviews). These elements *per se* are not informative enough to make sure that cells can recognize the correct borders with precision. There



**Figure 1.** UCSC Genome Browser showing the three major types of alternative splicing. A, exon skipping reported for gene *CD44*. B, usage of alternative acceptor site for gene *MDM2*. One of the alternative acceptor sites is reported by sequence AJ278977 (last exon). C, intron retention reported for gene *WDR39*. The retention event is reported by sequence AK056423.

are additional sequence elements, most unknown, that are probably binding sites for RNAs and/or proteins—the *trans* factors. These elements in *trans* are in the spliceosome and bind the elements in *cis* positioning in the radical groups involved in the splicing reaction. Some of these elements can enhance splicing, such as the exonic or intronic splicing enhancers (ESEs or ISEs, respectively). On the other hand, they can also silence splicing and, therefore, are called exonic or intronic splicing silencers (ESSs or ISSs, respectively). Several groups of proteins are now known to bind these elements. SR proteins, for example, bind enhancers preferentially, whereas hnRNPs bind silencers preferentially (for a review, see Matlin *et al.* (7)).

As predicted by Wally Gilbert (8) in his seminal “Why genes in pieces?” paper in the late 1970s, alternative exon/intron borders can be used by cells with profound implications for the encoded protein. In fact, splicing variants have been found since then for a variety of genes in several species. As shown in Figure 1, there are several types of alternative splicing. The most common involves the skipping of one or more exons. Alternative donor and acceptor sites can also be used, and finally, introns can be retained in the mature message.

The availability of a large amount of expressed data in the form of ESTs has allowed large-scale studies on alternative splicing in mammals, especially mouse and human (9, 10). Surprisingly, these analyses showed that alternative splicing is much more frequent than originally estimated. At least one-half of all human genes undergo alternative splicing, and this number is certainly underestimated, because for those genes with a high expression level and consequently more represented in dbEST, the rate is close to 90%. This high frequency of alternative splicing has raised concerns about the biological significance of the splicing variants. Although there is evidence for a functional significance of some splicing variants, some authors have argued that a significant



fraction of all splicing variants is spurious. These variants could be simply the products of leaking in the splicing reaction. In fact, a significant amount of splicing variants occurring in the coding region of genes does not maintain the respective reading frame (11). However, if we take those variants that are present in both mouse and human, most of them conserve the reading frame. Interestingly, some have argued that these spurious variants can have a functional role by down-regulating the normal function of a gene. These authors have termed this process RUST (regulated unproductive splicing and translation) (12, 13). These “unproductive” messages would be degraded by a process known as nonsense-mediated decay (NMD) (14). NMD is an RNA surveillance system that recognizes and targets for destruction those messages presenting a premature stop codon.

Also important in this field is the characterization of the regulation of alternative splicing. Much of our knowledge about the intricate mechanisms regulating constitutive and alternative splicing was achieved through individual laboratories working on specific models. More recently, large-scale approaches have been used extensively in the identification of putative regulatory elements. For example, proteomic analysis has been used in the identification of proteic components of the spliceosome (15). SELEX has been used extensively in the characterization of the RNA-binding specificity of many splicing factors (16). Finally, a combination of experimental and bioinformatics approaches has been used extensively by Chris Burge’s group in the identification of different types of regulatory elements (17-19).

## Computational Approaches to Detect Splicing Variants

Almost the totality of data on splicing variants was deduced by different methods involving sequence comparisons. The idea behind these methodologies is that splicing variants from the same gene share some common sequence and can therefore be grouped together. ESTs are the major source in these studies aimed to catalog the splicing variants. Mironov *et al.* (20), for example, used EST contigs from the TIGR gene index to make an inventory of intron–exon structures in the set of known human genes. With the availability of the human genome sequence, several groups started to make more precise inferences about alternative splicing by simply mapping all cDNAs onto the genome sequence (9, 21). The use of more precise algorithms of alignment, such as Sim4 (22), which pays special attention to the exon/intron border, made these inferences more reliable.

Instead of comparing pairwise alignments, the state-of-art methodology nowadays aligns all cDNAs against the genome, loads the cDNA coordinates into relational databases, and compares the borders among different cDNAs from the same gene. Details on the approach used by our group can be found elsewhere (21, 23, 24). Figure 2 shows two variants from gene *WDR39* extracted directly from our relational database.

The major advantage of this approach is the possibility to perform genome-wide analyses through the development of software that interrogates the relational database. The knowledge extracted can be inserted back in the same database in the form of different tables. The information can also be easily linked to other types of data because most of the related information currently uses the genome sequence as a scaffold.

## Differential Expression of Splicing Variants in Tumors

Alternative splicing has been involved in many different biological phenomena including sex determination in *Drosophila* (25), among others. As expected, alternative splicing seems to be quite important in the pathogenesis of several human diseases. It is believed that around 15% of all human genetic diseases are caused by mutations in sequence elements important for constitutive splicing. One of the first splicing mutations described activates a cryptic acceptor site in the  $\beta$ -globin gene resulting in  $\beta^+$ -thalassemia. The involvement of splicing variants in a specific disease can be more subtle. Let’s take, as an example, the fronto-temporal dementia with Parkinson (FTDP-17), a neurological disease linked to chromosome 17 (26). One of the first candidate genes for this disease was the protein tau, a microtubule-associated protein involved with axonal transport in neurons. Indeed, mutations in tau have been reported to be associated with development of FTDP-17 (27). More recently,

```
mysql> select * from cdna_blocks where seqacc="AK056423";
```

contigid	seqacc	indexid	blocknum	blockid	start_cdna	end_cdna	start_gnm	end_gnm	strand	ident	cluster
CHR2	AK056423	1	0	NULL	1	250	96353852	96354101	+	99	Hs.12109
CHR2	AK056423	1	0	NULL	251	362	96355237	96355348	+	99	Hs.12109
CHR2	AK056423	1	0	NULL	363	451	96355448	96355536	+	100	Hs.12109
CHR2	AK056423	1	0	NULL	452	1350	96356069	96356967	+	99	Hs.12109
CHR2	AK056423	1	0	NULL	1351	3902	96358723	96361274	+	99	Hs.12109

```
5 rows in set (0.00 sec)
```

```
mysql> select * from cdna_blocks where seqacc="BC032812";
```

contigid	seqacc	indexid	blocknum	blockid	start_cdna	end_cdna	start_gnm	end_gnm	strand	ident	cluster
CHR2	BC032812	1	0	NULL	1	238	96353864	96354101	+	100	Hs.12109
CHR2	BC032812	1	0	NULL	239	387	96354933	96355081	+	100	Hs.12109
CHR2	BC032812	1	0	NULL	388	499	96355237	96355348	+	100	Hs.12109
CHR2	BC032812	1	0	NULL	500	588	96355448	96355536	+	100	Hs.12109
CHR2	BC032812	1	0	NULL	589	790	96356069	96356270	+	100	Hs.12109
CHR2	BC032812	1	0	NULL	791	878	96356880	96356967	+	100	Hs.12109
CHR2	BC032812	1	0	NULL	879	1326	96358723	96359170	+	100	Hs.12109

```
7 rows in set (0.00 sec)
```

```
mysql>
```

**Figure 2.** Structure of an in-house database reporting two splicing variants for gene *WDR39*. The first query for sequence AK056423 reports five exons. The second query for sequence BC032812 reports seven exons. Differences are attributable to an inclusion of the second exon in the sequence BC032812 and an intron retention in sequence AK056423 (fourth exon in AK056423 corresponds to the fifth and sixth exons in BC032812).

mutations that alter the ratio of splicing variants, skipping or including exon 10, have also been associated with FTDP-17 (28).

Isolated cases of differential expression of splicing variants in cancer have been reported in the last 10 years (for a review, see Caballero *et al.* (29)). For example, Bcl-x, an apoptosis regulator, has two splicing variants because of alternative donor sites in its exon 2. Only the longer form is differentially expressed in small cell lung carcinoma (30) and breast carcinoma (31). The most known example, however, is CD44, a cell surface glycoprotein. Differential expression of several splicing variants has been observed for a range of different tumors (for a review, see Caballero *et al.* (29)). More recently, Matsushita *et al.* (32) reported that a splicing variant of FIR (FUSE-binding protein-interacting repressor) is unable to repress c-Myc and to drive apoptosis. This splicing variant was only expressed in colorectal cancer cells and was not detected in the adjacent normal cells. The results presented in this report suggest that this variant promotes tumor development. Narla *et al.* (33) reported an association between a germline polymorphism with both an unbalanced expression of a splicing variant of KLF6 and an increased risk for prostate cancer.

The increasing amount of cDNA libraries constructed from both normal and tumor samples allows the development of genome-wide strategies for the identification of tumor-associated splicing variants. Several groups reported genome-wide screening strategies searching for splicing variants differentially expressed in tumors (23, 34-36). Without exception, these authors made use of the huge amount of cDNA data available in the public databases to identify putative variants differentially expressed in tumors. The proportional frequency of cDNAs derived from distinct variants is an indication whether a given variant is differentially expressed in a library or in a pool of libraries. One of the major problems affecting this type of analysis is the identification of genes, not variants, differentially expressed in tumors. This happens because the computational and statistical methods used in the analyses are not sensitive enough to discriminate the expression level of all variants from a given gene. More recently, we tried to overcome this limitation by using SAGE data to discriminate variants differentially expressed from genes differentially expressed (23). Our computational approach identified more

than 1,300 splicing variants putatively associated with cancer. Experimental validation for a subset of these candidates was achieved for both tumor cell lines and patient samples.

All of these reports reinforce the notion that cancer cells reprogram the splicing pattern of their genes. How? One possibility is that splicing factors are differentially expressed in tumors, and this causes a downstream effect on the transcriptome of these cells. We have recently evaluated this possibility through the use of SAGE and microarray data (37). We showed that splicing factors are indeed differentially expressed in tumors.

The discovery of splicing variants associated with cancer represents a promising strategy in the fight against this terrible disease. As already discussed, changes in splicing have been shown to play a functionally significant role in tumorigenesis. They can also serve as targets for the early diagnosis of cancer. Furthermore, cancer-specific splicing variants can present new epitopes recognized by the immune system and may serve as targets for immunotherapy.

## Large-Scale Analysis of Alternative Splicing in Parasites

Parasites are a diverse group of organisms that share the common features of living within and exploiting a host organism. Most parasites undergo complex, multiphase life cycles that can involve extracellular and intracellular stages as well as different hosts. As a result of this complexity and unusual approaches to survival, parasites developed some remarkable adaptations at both the genetic and biochemical level that allow them, for example, to evade host defenses and to facilitate transmission to new hosts.

Molecular studies of parasites, particularly of trypanosomatids and apicomplexans, have revealed novel aspects of gene regulation and expression that could later be applicable to higher eukaryotes. Post-transcriptional modifications, such as *trans*-splicing (38) and RNA editing (39), were first observed and characterized at the molecular level in trypanosomatids. *cis*-Splicing also occurs among parasites, although to a lesser extent than that observed for higher eukaryotes. Interesting examples of alternative splicing in parasites have already been documented. Alternative splicing isoforms of the hypoxanthine-xanthine-guanine phosphoribosyltransferase gene (*HXGPRT*) have been observed in *Toxoplasma gondii*. These isoforms differ in the presence or absence of a 49-amino acid insertion (which is specified by a single, differentially spliced exon) and present different cellular localization, suggesting the existence of functional differences between both isoforms (40). Alternative splicing isoforms of the *PfPK6* gene in *Plasmodium falciparum* have also been observed and shown to be differentially expressed during different asexual erythrocytic stages of this parasite (41).

These initial works suggest that alternative splicing might have an important role in parasites' biology and could regulate important events such as invasion into host cells and evasion of host defense. Genome sequencing projects have been carried or are under way for several relevant human parasites, such as *P. falciparum* and *P. vivax*, *Trichomonas vaginalis*, *Toxoplasma gondii*, *Schistosoma mansoni*, *Leishmania major*, *Trypanosoma cruzi*, and *T. brucei* (<http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=Genome>). Expressed sequences (ESTs and full-length cDNA clones) from different life stages of most of these parasites are also publicly available (<http://www.tigr.org/tdb/tgi/index.shtml>). In this context, large-scale analysis of alternative splicing in parasites, using computational tools similar to the ones we described in this chapter, would be extremely important for a better understanding of parasites' biology and could eventually have important practical implications for preventing, diagnosing, and treating parasitic diseases.

## Final Comments

As discussed by Wang *et al.* (19), the characterization of an "RNA splicing code" will require a detailed catalog of all splicing variants, all splicing factors, all regulatory elements, and the interactions among all of these elements. We expect that bioinformatics/computational biology will continue to provide essential information for the completion of this catalog and finally, a full characterization of alternative splicing.

## References

1. Adams MD, Dubnick M, Kerlavage AR, Moreno R, Kelley JM, Utterback TR, Nagle JW, Fields C, Venter JC. Sequence identification of 2,375 human brain genes. *Nature*. 1992;355:632–634. PubMed PMID: 1538749.
2. Velculescu VE, Zhang L, Vogelstein B, Kinzler KW. Serial analysis of gene expression. *Science*. 1995;270:484–487. PubMed PMID: 7570003.
3. Brenner S, Johnson M, Bridgham J, Golda G, Lloyd DH, Johnson D, Luo S, McCurdy S, Foy M, Ewan M, Roth R, George D, Eletr S, Albrecht G, Vermaas E, Williams SR, Moon K, Burcham T, Pallas M, DuBridge RB, Kirchner J, Fearon K, Mao J, Corcoran K. Gene expression analysis by massively parallel signature sequencing (MPSS) on microbead arrays. *Nat Biotechnol*. 2000;18:630–634. PubMed PMID: 10835600.
4. Fodor SP, Read JL, Pirrung MC, Stryer L, Lu AT, Solas D. Light-directed, spatially addressable parallel chemical synthesis. *Science*. 1991;251:767–773. PubMed PMID: 1990438.
5. Ast G. How did alternative splicing evolve? *Nat Rev Genet*. 2004;5:773–782. PubMed PMID: 15510168.
6. Woodley L, Valcarcel J. Regulation of alternative pre-mRNA splicing. *Brief Funct Genomic Proteomic*. 2002;1:266–277. PubMed PMID: 15239893.
7. Matlin AJ, Clark F, Smith CW. Understanding alternative splicing: towards a cellular code. *Nat Rev Mol Cell Biol*. 2005;6:386–398. PubMed PMID: 15956978.
8. Gilbert W. Why genes in pieces? *Nature*. 1978;271:501. PubMed PMID: 622185.
9. Modrek B, Lee CJ. Alternative splicing in the human, mouse and rat genomes is associated with an increased frequency of exon creation and/or loss. *Nat Genet*. 2003;34:177–180. PubMed PMID: 12730695.
10. Xu Q, Modrek B, Lee C. Genome-wide detection of tissue-specific alternative splicing in the human transcriptome. *Nucleic Acids Res*. 2002;30:3754–3766. PubMed PMID: 12202761.
11. Resch A, Xing Y, Alekseyenko A, Modrek B, Lee C. Evidence for a subpopulation of conserved alternative splicing events under selection pressure for protein reading frame preservation. *Nucleic Acids Res*. 2004;32:1261–1269. PubMed PMID: 14982953.
12. Hillman RT, Green RE, Brenner SE. An unappreciated role for RNA surveillance. *Genome Biol*. 2004;5:R8. PubMed PMID: 14759258.
13. Lewis BP, Green RE, Brenner SE. Evidence for the widespread coupling of alternative splicing and nonsense-mediated mRNA decay in humans. *Proc Natl Acad Sci U S A*. 2003;100:189–192. PubMed PMID: 12502788.
14. Morrison M, Harris KS, Roth MB. smg mutants affect the expression of alternatively spliced SR protein mRNAs in *Caenorhabditis elegans*. *Proc Natl Acad Sci U S A*. 1997;94:9782–9785. PubMed PMID: 9275202.
15. Rappsilber J, Ryder U, Lamond AI, Mann M. Large-scale proteomic analysis of the human spliceosome. *Genome Res*. 2002;12:1231–1245. PubMed PMID: 12176931.
16. Tacke R, Manley JL. The human splicing factors ASF/SF2 and SC35 possess distinct, functionally significant RNA binding specificities. *EMBO J*. 1995;14:3540–3551. PubMed PMID: 7543047.
17. Fairbrother WG, Yeo GW, Yeh R, Goldstein P, Mawson M, Sharp PA, Burge CB. RESCUE-ESE identifies candidate exonic splicing enhancers in vertebrate exons. *Nucleic Acids Res*. 2004;32:W187–90. PubMed PMID: 15215377.
18. Fairbrother WG, Yeh RF, Sharp PA, Burge CB. Predictive identification of exonic splicing enhancers in human genes. *Science*. 2002;297:1007–1013. PubMed PMID: 12114529.
19. Wang Z, Rolish ME, Yeo G, Tung V, Mawson M, Burge CB. Systematic identification and analysis of exonic splicing silencers. *Cell*. 2004;119:831–845. PubMed PMID: 15607979.
20. Mironov AA, Fickett JW, Gelfand MS. Frequent alternative splicing of human genes. *Genome Res*. 1999;9:1288–1293. PubMed PMID: 10613851.
21. Galante PA, Sakabe NJ, Kirschbaum-Slager N, de Souza SJ. Detection and evaluation of intron retention events in the human transcriptome. *RNA*. 2004;10:757–765. PubMed PMID: 15100430.
22. Florea L, Hartzell G, Zhang Z, Rubin GM, Miller W. A computer program for aligning a cDNA sequence with a genomic DNA sequence. *Genome Res*. 1998;8:967–974. PubMed PMID: 9750195.

23. Kirschbaum-Slager N, Parmigiani RB, Camargo AA, de Souza SJ. Identification of human exons overexpressed in tumors through the use of genome and expressed sequence data. *Physiol Genomics*. 2005;21:423–432. PubMed PMID: 15784694.
24. Sakabe NJ, de Souza JE, Galante PA, de Oliveira PS, Passetti F, Brentani H, Osorio EC, Zaiats AC, Leerkes MR, Kitajima JP, Brentani RR, Strausberg RL, Simpson AJ, de Souza SJ. ORESTES are enriched in rare exon usage variants affecting the encoded proteins. *C R Biol*. 2003;326:979–985. PubMed PMID: 14744104.
25. Baker BS. Sex in flies: the splice of life. *Nature*. 1989;340:521–524. PubMed PMID: 2505080.
26. Lynch T, Sano M, Marder KS, Bell KL, Foster NL, Defendini RF, Sima AA, Keohane C, Nygaard TG, Fahn S et al. Clinical characteristics of a family with chromosome 17-linked disinhibition-dementia-parkinsonism-amyotrophy complex. *Neurology*. 1994;44:1878–1884. PubMed PMID: 7936241.
27. Hutton M, Lendon CL, Rizzu P, Baker M, Froelich S, Houlden H, Pickering-Brown S, Chakraverty S, Isaacs A, Grover A, Hackett J, Adamson J, Lincoln S, Dickson D, Davies P, Petersen RC, Stevens M, de Graaff E, Wauters E, van Baren J, Hillebrand M, Joosse M, Kwon JM, Nowotny P, Che LK, Norton J, Morris JC, Reed LA, Trojanowski J, Basun H, Lannfelt L, Neystat M, Fahn S, Dark F, Tannenberg T, Dodd PR, Hayward N, Kwok JB, Schofield PR, Andreadis A, Snowden J, Craufurd D, Neary D, Owen F, Oostra BA, Hardy J, Goate A, van Swieten J, Mann D, Lynch T, Heutink P. Association of missense and 5'-splice-site mutations in tau with the inherited dementia FTDP-17. *Nature*. 1998;393:702–705. PubMed PMID: 9641683.
28. Grover A, DeTure M, Yen SH, Hutton M. Effects on splicing and protein function of three mutations in codon N296 of tau in vitro. *Neurosci Lett*. 2002;323:33–36. PubMed PMID: 11911984.
29. Caballero OL, de Souza SJ, Brentani RR, Simpson AJ. Alternative spliced transcripts as cancer markers. *Dis Markers*. 2001;17:67–75. PubMed PMID: 11673653.
30. Reeve JG, Xiong J, Morgan J, Bleehen NM. Expression of apoptosis-regulatory genes in lung tumor cell lines: relationship to p53 expression and relevance to acquired drug resistance. *Br J Cancer*. 1996;73:1193–1200. PubMed PMID: 8630278.
31. Olopade OI, Adeyanju MO, Safa AR, Hagos F, Mick R, Thompson CB, Recant WM. Overexpression of BCL-x protein in primary breast cancer is associated with high tumor grade and nodal metastases. *Cancer J Sci Am*. 1997;3:230–237. PubMed PMID: 9263629.
32. Matsushita K, Tomonaga T, Shimada H, Shioya A, Higashi M, Matsubara H, Harigaya K, Nomura F, Libutti D, Levens D, Ochiai T. An essential role of alternative splicing of *c-myc* suppressor FUSE-binding protein-interacting repressor in carcinogenesis. *Cancer Res*. 2006;66:1409–1417. PubMed PMID: 16452196.
33. Narla G, DiFeo A, Reeves HL, Schaid DJ, et al. A germline DNA polymorphism enhances alternative splicing of the KLF6 tumor suppressor gene and is associated with increased prostate cancer risk. *Cancer Res*. 2005;65:1213–1222. PubMed PMID: 15735005.
34. Hui L, Zhang X, Wu X, Lin Z, Wang Q, Li Y, Hu G. Identification of alternatively spliced mRNA variants related to cancers by genome-wide ESTs alignment. *Oncogene*. 2004;23:3013–3023. PubMed PMID: 15048092.
35. Wang Z, Lo HS, Yang H, Gere S, Hu Y, Buetow KH, Lee MP. Computational analysis and experimental validation of tumor-associated alternative RNA splicing in human cancer. *Cancer Res*. 2003;63:655–657. PubMed PMID: 12566310.
36. Xu Q, Lee C. Discovery of novel splice forms and functional analysis of cancer-specific alternative splicing in human expressed sequences. *Nucleic Acids Res*. 2003;31:5635–5643. PubMed PMID: 14500827.
37. Kirschbaum-Slager N, Lopes GM, Galante PA, Riggins GJ, de Souza SJ. Splicing factors are differentially expressed in tumors. *Genet Mol Res*. 2004;3:532–544. PubMed PMID: 15688319.
38. Liang XH, Haritan A, Uliel S, Michaeli S. trans and cis splicing in trypanosomatids: mechanism, factors, and regulation. *Eukaryot Cell*. 2003;2(5):830–840. PubMed PMID: 14555465.
39. Stuart K, Panigrahi AK. RNA editing: complexity and complications. *Mol Microbiol*. 2002;45(3):591–596. PubMed PMID: 12139607.
40. Chaudhary K, Donald RG, Nishi M, Carter D, Ullman B, Roos DS. Differential localization of alternatively spliced hypoxanthine-xanthine-guanine phosphoribosyltransferase isoforms in *Toxoplasma gondii*. *J Biol Chem*. 2005;280(23):22053–22059. PubMed PMID: 15814612.

41. Bracchi-Ricard V, Barik S, Delvecchio C, Doerig C, Chakrabarti R, Chakrabarti D. PfPK6, a novel cyclin-dependent kinase/mitogen-activated protein kinase-related protein kinase from *Plasmodium falciparum*. *Biochem J*. 2000;347(Pt 1):255–263. PubMed PMID: 10727426.