

Oxford University Bioinformatics Centre



# Introduction to GCG and SeqLab

31 July 2001

Sir William Dunn School of Pathology  
South Parks Road  
Oxford, OX1 3RE

# Contents

1. What is GCG?
2. Some of the functions of GCG programs
3. Ways to access GCG programs
  - 3.1 Running programs via the Command Line
  - 3.2 Running programs via SeqLab
4. Finding help for GCG programs
  - 4.1 Within your account
  - 4.2 On the web
5. Sequence formats
  - 5.1 An example nucleotide sequence in GCG format
  - 5.2 The same file in fasta format
  - 5.3 ReadSeq
6. Naming conventions
7. Starting the tutorial
8. Working with the command line and prompted command line
  - 8.1 Accessing documentation via the command line
  - 8.2 Running programs via the command line
  - 8.3 The prompted command line - Running Map
  - 8.4 The non-prompted command line - Running Map
  - 8.5 Automating the command line
9. Giving files as input for programs
  - 9.1 List files
  - 9.2 Examples of running programs with single vs. multiple sequences as input
  - 9.3 Multiple sequence files in GCG format
  - 9.4 More notes on formatting
10. Sequence retrieval
11. Dealing with graphical results
  - 11.1 Running pepplot
12. Opening SeqLab
13. The SeqLab environment
  - 13.1 List Mode
  - 13.2 Editor Mode
14. Adding sequences to SeqLab
  - 14.1 Add sequences one at a time from files in your account
  - 14.2 . Add sequences directly from databases using sequence ID's.

- 15. Editor mode
  - 15.1 Adding sequences directly from within the Editor mode
  - 15.2 Features Coloring and Residue Coloring
  - 15.3 Editing sequences in SeqLab
  - 15.4 Editing multiple sequences
  - 15.5 Adding Comments to your sequence
  - 15.6 Other options you have when you work with SeqLab

- 16. Running programs
  - 16.1 Getting and Viewing the Results
  - 16.2 What is the How: option asking me?
  - 16.3 Job Manager
  - 16.4 Output Manager
  - 16.5 Dealing with the Results Files in the Output Manager

- 17. Printing from SeqLab

- 18. Exercises
  - 18.1 Mapping programs
    - 18.1.1 Map
    - 18.1.2 MapSort
  - 18.2 Protein Analysis Programs
    - 18.2.1 Motifs
  - 18.3 Multiple Sequence Alignment Programs
    - 18.3.1 Clustalw
    - 18.3.2 Pileup
  - 18.4 After the Multiple Sequence Alignment
    - 18.4.1 Pretty
    - 18.4.2 PrettyBox
  - 18.5 Weights and Multiple Sequence Alignments

- 19. General Comments

Appendix 1 : Running Processes in the Background

Appendix 2: Creating a new sequence in Editor Mode (not recommended)

Appendix 3:RSF file format

Appendix 4: How to convert graphics images

## 1. What is GCG?

GCG, also known as The Wisconsin Package (TM) is a package of computer programs that allows you to manipulate and analyse nucleic acid and protein sequences.

Some of the programs are very useful. However, it is worth remembering that some of the programs are quite old and other, non-GCG programs may be available that are better, faster, or both!

Of particular note are the Staden package of programs which is used to deal with ABI sequence data and sequence projects, and the EMBOSS package of programs, which duplicates and extends many of the capabilities of GCG. Staden is most commonly used via an Xwindows interface, and the EMBOSS programs are available via the command line, or web forms.

There are many other programs available on molbiol. A good source of information about what is available is our web site: <http://www.molbiol.ox.ac.uk>

## 2. Some of the functions of GCG programs

**Comparison.** Compare two or more sequences. Create, edit, display, and analyse multiple sequence alignments.

**Database Searching and Retrieval.** Search nucleic acid or protein sequence databases for sequences similar to your query sequence or sequence pattern.

**DNA/RNA Secondary Structure.** Predict and display optimal and suboptimal DNA or RNA secondary structures.

**Editing and Publication.** Enter sequences from a digitizer or a keyboard and edit them.

**Evolution.** Determine and display evolutionary phylogenies from multiple sequence alignments.

**Fragment Assembly.** Manage and assemble nucleotide sequence fragments in a sequencing project (We highly recommend that you use programs from the Staden package for this.)

**Gene Finding and Pattern Recognition.** Recognise terminators, repeats, protein coding regions, and other consensus patterns.

**Mapping.** Calculate and display restriction digests and simulate RNA fingerprints.

**Primer Prediction.** Predict optimal primers for PCR reactions. (Note: there is a web program called Primer3 that we recommend for this.)

**Protein Analysis.** Identify sequence motifs in protein sequences and make predictions about peptide isolation.

**Translation.** Translate nucleic acids into proteins and backtranslate proteins into nucleic acids.

### 3. Ways to access GCG programs

Before you run GCG programs, you must “initialise the GCG package”. This means that you need to setup the system to run GCG programs. To do this, you just need to type:

**gcg**

on the command line.

You will need to initialise GCG once in each window you work in. This means that if you connect to molbiol through an Xwindows session (Exceed or Exodus), you must type **gcg** in each xterm window you will be running GCG programs in.

**You can access gcg programs in two ways:**

- 1. via the command line,**
- 2. via the SeqLab environment.**

#### 3.1. Running programs via the command line

To run GCG programs on the command line, you type in the program name, and any optional parameters you need. A full list of command line options for each program is available within the help for that program.

##### **Advantages of using command line interface**

- Once you are familiar with it, it is fast and easy to control.
- If you are carrying out a particular analysis repetitively on different sequence files, the program can be included in a script to help automate the process. A script is a name given to a list of computer commands typed in a file. All the commands in that file can be executed by the machine rather than you running each step manually.
- It is easier for OUBC to trace problems you may be having if you use the command line because a history of recent commands you have executed is automatically generated.

##### **Disadvantages of using the command line interface:**

- You have to read the help for programs to know the full range of options available to you
- You have to type in command names, etc.

#### 3.2. Running programs via SeqLab

SeqLab is a graphical interface (i.e. windows-like) to GCG programs and provides a point-and-click multiple sequence editor and annotation tool. Technically speaking, it is an Xwindows interface to the GCG programs.

##### **Advantages of using the SeqLab environment**

- It is somewhat more intuitive than the command line, though you do have to figure out where everything is at first!
- Lots of “point and click”. Very little typing is necessary

- Contains a good multiple sequence alignment editor
- The program options can be set easily
- Help is readily available at the click of a button.

#### **Disadvantages of using the SeqLab environment**

- SeqLab gives you no option for what your output files should be named. You can change the name of the output file afterwards, but you will then have two copies of the output in your account. The only way to delete one is to go back to the command line and delete the files manually!!
- Once you are familiar with the command line, SeqLab can be slow to use in comparison.
- If you have a task that you want to carry out repetitively, SeqLab can be tedious
- There is no history kept of what you have been doing within SeqLab, so it is hard to trace problems.
- The results of all the analyses you run are saved in your account, unless you purposefully delete them. This is also true of command line gcg but because you are working in a separate, graphical window, it is easier to forget that you are accumulating files, wanted or not, in your account. If these files are very large, (e.g. blast search results), you may exceed the quota on your account.

## **4. Finding help for GCG programs?**

You can access the official GCG documentation when you are logged into your account, via the command line, or from within SeqLab. The same help is also available via the web.

### **4.1. Within your account**

#### *i. Command Line*

After you have initialised the GCG package by typing

**gcg**

on the command line, you can obtain help grouped either by program name, by typing

**genhelp**

or help grouped according to function, by typing

**genman**

A full list of command line options for any program can be produced by typing the name of that program followed by **-check**. For example, to get the list of options for the program translate, you could type:

**translate -check**

#### *ii. Through SeqLab*

On all the windows presented to you in the SeqLab environment, there is a help button. When this button is clicked on, a new window appears containing help or help topics.

There is also a Help menu on the main SeqLab window. Choosing the option “On the Wisconsin Package” under Help will bring up a menu of program types such as that seen by typing **genman** on the command line.

#### 4.2. On the web:

*i. Using the search box*

Search for help on a program by entering its name in the search box on our home page: **<http://www.molbiol.ox.ac.uk>**

*ii. Via the GCG manual page*

Click on GCG10 under the word Software on our homepage.

This takes you to a general help page for GCG, from which you can access the official GCG documentation.

The URL to go directly to the GCG manual is:

**<http://www.molbiol.ox.ac.uk/documentation/gcg10docs/gcgmanual.html>**

The URL to go directly to the GCG Help menu is:

**<http://www.molbiol.ox.ac.uk/documentation/gcg10docs/gcghelp.html>**

## 5. Sequence formats

Something that often causes problems for new users is sequence formatting. “Format” refers to the way a sequence looks in a file. For example: are there spaces between some of the characters? How is the title specified so that it is not read as part of the sequence? Etc.

There are many different sequence formats. A few common ones are:

<b>GCG</b>	<b>Genbank</b>	<b>ABI</b>
<b>FASTA</b>	<b>Clustal</b>	
<b>GDE</b>	<b>Staden</b>	

### 5.1 .An example of a nucleotide sequence in GCG format is:

```
small.seq Length: 57 July 8, 1999 12:49 Type: N Check: 9806 ..  
  
1  AGGTTGCTGG CTGGCTGCAG CTAGCTGATC GCTGAATCGA TCGGCATGCT  
51  TGAGCTG
```

## 5.2 The same file in fasta format looks like:

```
>SMALL
AGGTTGCTGGCTGGCTGCAGCTAGCTGATCGCTGAATCGATCGGCATGCTTGAGCTG
```

Many programs require sequences to be in a particular format in order to be able to read and interpret them. Command line GCG programs usually accept only sequences in GCG format.

### Note

You can directly **Import** (not Add) sequences into the SeqLab editor that are in FASTA, GenBank flatfile, ABI, SCF or GDE file formats. When you save imported files in SeqLab, they are converted to GCG's rsf file format. See Appendix 3.

When working on the command line, your sequences must be in GCG format in order to be used with GCG analysis programs.

## 5.3 ReadSeq

There are a number of programs available to convert sequences from one format to another. The most versatile is a program called **readseq**. We will touch on changing sequence formats in this tutorial, but you can read more about readseq, and sequence formats in general, at:

<http://www.molbiol.ox.ac.uk/help/sequenceformats.htm>

## 6. Naming conventions

GCG programs add default suffixes to the names of results files according to what kind of data the file contains, and what program created the file.

By default, some of the common endings for GCG-formatted files are:

.seq	file containing a single nucleotide sequence
.pep	file containing a single peptide sequence
.msf	multiple sequence file (nucleotide or peptide)
.tfa	a sequence (or sequences) in fasta format
.sdn	a sequence in staden format
.rsf	a sequences in rich sequence format

The suffixes GCG, (and other programs) attach to the end of filenames are just conventions, however, they are *useful* conventions. You may be tempted to start changing them, but unless you really need to, it can often make your life more difficult in the future. For example, since GCG, by default, will give a nucleotide sequence the name

**somename.seq**

you will always know in the future, (unless you start changing things) that files with .seq in your account are likely to be in GCG format.

We will look at fasta formatting in a moment, but the default ending for this is .tfa. In the future you will know that sequences with this ending are likely to be in fasta format.

This may seem trivial now, but as you acquire more and more data, it helps to have a system where you don't have to look inside each sequence file to find out what format the data is in.

## 7. Starting the tutorial

It has been assumed that you have some familiarity with basic Unix and your account (to the level you might attain during the "Introduction to Bioinformatics at Oxford" course). If there is any terminology used in this practical that you do not understand, please ask a demonstrator.

The first part of this tutorial will access GCG documentation, and run GCG programs via the command line. The second half will introduce you to the Xwindows version of GCG, SeqLab.

First, create a directory to work in for the duration of this course. Type:

```
mkdir gcg_course
```

Then move into this directory. Type:

```
cd gcg_course
```

```
cp ~trainer/gcg_course/* .
```

Whether you work on the command line, or in SeqLab, you will first need to initialise GCG. Type:

```
gcg
```

## 8. Working with the Command Line and Prompted Command Line

### 8.1 Accessing documentation via the command line

On the command line, type

```
genhelp
```

Scroll through the options listed using the down arrow key and/or the space bar, until you see the word FindPatterns.

**Highlight FindPatterns and press the return key.**

You are now presented with help for the program FindPatterns. You can either scroll down the page using the down arrow key and/or the space bar, or you can highlight any of the titles and press the return key. These are links to the relevant section of the documentation. E.g. Function will take you to an explanation of the function of the program.

When you have read all you want to, type: **q**

When you are prompted, confirm that you want to quit by typing **y**

Now type: **genman**

You can see that unlike genhelp, which lists the names of programs alphabetically, **genman organises programs by their function**. This can be useful if you know what you want to do, but do not know what programs are available.

**Highlight and click on Gene Finding and Pattern Recognition.**

**Highlight FindPatterns**

As you can see, the documentation presented to you is exactly the same as that through genhelp. If you access the documentation through the web, you'll find it is the same again, however, you may find the web pages easier to browse through.

## 8.2 Running programs via the command line

Note: The following section is also covered in the **Introduction to Bioinformatics at Oxford** course. If you have just done this course and feel comfortable with the information, you can just read through this section and move on. **If you have not done the Introductory course this term, or are not comfortable working on the command line, please work through this section.**

Most programs can be run purely by typing everything the program needs to know on the command line and pressing the return key. Usually this resembles the way you give Unix commands: you enter the command followed by flags or arguments specifying how you want the program to run. Some programs will prompt you for the information they need if you choose not to enter it on the command line.

If a program will prompt you for the information it needs, why bother to type everything on the command line? Two reasons:

1) Many programs can be tailored to run according to your needs, but when a program prompts you for input, it usually only prompts you for information that is absolutely necessary in order to run. There may be many useful options that you miss out on using if you only answer the prompts.

2) If you have repetitive tasks to carry out, (e.g. say you have 100 sequences to analyse in a particular way), the easiest (i.e. fastest) method to set this up involves using the command line.

To illustrate these points, we will be using the GCG program **map**. This program looks for restriction sites in nucleotide sequences and is also useful for looking for open reading frames and translations in any, or all 6, frames. (All GCG programs accept sequences in gcg format.)

### 8.3 The Prompted Command Line - Running Map

When you run map (or any GCG program on the command line) you can provide as much, or as little information on the command line as you like. If you supply all the required information, the program will run without prompting you for anything further. If you miss out any vital information, you will be prompted for it. If you miss out any information that is not vital, but affected the way you wanted the program to run, you will not be prompted for it; the program will run, but not as you wanted.

*Please note:* Not all command line programs will prompt for information, and even those that do may not be as polite as GCG. If you forget a vital piece of information in other programs, they may tell you what you have done wrong, or they may just not run and give no reason. If this happens to you, the first thing you should ask is: did I forget anything, or type anything wrong, on the command line?

We will start by running map as simply as possible. Type

```
map
```

You will see the following prompt appear:

```
Map maps a DNA sequence and displays both strands of the mapped
sequence with restriction enzyme cut points above the sequence
and protein translations below.  Map can also create a peptide
map of an amino acid
sequence.
```

```
(Linear) MAP of what sequence ?
```

**Type the name of the sequence: testseq1.seq**

Next you are prompted for what nucleotide you wish to start at. By default the mapping will start at the first, but you can choose anywhere within your sequence to start. For now, just go with the default: **press the return key.**

The next prompt asks up to what nucleotide number you wish to run the mapping on. For now, let's go with the default again, which will be the last residue of your sequence: **press the return key.**

Now you should be seeing something like this:

```
(Linear) MAP of what sequence ?  testseq1.seq
```

```
      Begin (* 1 *) ?
      End  (*   430 *) ?
```

```
Select the enzymes:  Type nothing or (*) to get all enzymes.
Type (?) for help on which enzymes are available and how to
select them.
```

```
Enzyme(* * *):
```

```
Try typing ? to view all the enzymes you could choose. Keep
pressing return until you see a prompt like:
```

```
597 possible enzymes in (**).           Enzyme:
```

Now you could enter particular enzyme names if you so wished. For now, just press return, which chooses to map using all the restriction enzymes you saw listed.

The next prompt looks like:

```
What protein translations do you want:
```

```
a) frame 1    b) frame 2    c) frame 3  
d) frame 4    e) frame 5    f) frame 6
```

```
t)hree forward frames    s)ix frames    o)pen frames only  
n)o protein translation    q)uit
```

```
Please select (capitalize for 3-letter) (* t *):
```

As well as mapping where restriction enzymes cut, the program map can also translate your nucleotide sequence in as many frames as you like. Later in the practical, you may want to try this exercise again using different options at this point. The open reading frames option will only translations of minimum length of 20 amino acids without a stop or nonsense codon. (This length can be changed on the command line.)

For now, just choose the default (translation in 3 forward frames): **press the return key.**

You are now prompted to give a name for a file that will store the results of your analysis.

```
What should I call the output file (* testseq1.map *) ?
```

Recall the discussion about naming conventions above. For now, choose the default: **press the return key.**

When your analysis has run you will find the results in a file called **testseq1.map.**

Use the command less to look at this file: type

```
less testseq1.map
```

You should see your sequence, a three frame translation of it, and sites marked for where restriction enzymes cut the sequence. Keep pressing the space bar until you near the bottom of the file. Here you should find two lists: one of enzymes that did cut your sequence, and one of enzymes that did not.

*That's it. You've run your first command line program!*

## 8.4 The Non-Prompted Command line - running Map

Now we're going to run map again, only this time, you'll give a bit of information on the command line itself, and then choose the defaults for everything you are prompted for.

Run the program map again, but this time, type the following on the command line:

```
map -begin=1 -menu=t testseq2.seq
```

What this means is: "Run the program map, using the sequence testseq2.seq, begin at residue 1 and translate the sequence in the 3 forward frames".

If you cannot see why the above command means this, please ask a demonstrator.

You will notice that you are now only prompted for the information you have not already given the program. In this case, what nucleotide to stop the analysis at, what enzymes to check, and what to call the output. Choose the defaults for all of these: **press the return key for each.**

You can supply all the information needed by the program on the command line. If you do this, the program just runs and prompts you for nothing.

For example, to run the program the same as above, you can type:

```
map -begin=1 -end=430 -out=testseq2.map -menu=t -enzymes=* testseq2.seq
```

In fact, the GCG programs all include an option **-def** that means: "Run this program using the default answers to all the prompts".

So to run map and automatically agree to all the default settings of Map, (i.e. no prompting), you can type:

```
map -def testseq2.seq
```

You can "mix and match" default and non-default values by specifying the non-default values on the command line and including the **-def** flag. This tells the program to take into account the options you have specified, but to use the defaults for everything else.

For example, **type:**

```
map -begin=150 -out=anothertry.map -def testseq2.seq
```

**Now look at anothertry.map. Type:**

```
less anothertry.map
```

It is the same as testseq2.map except that the analysis did not include the first 150 nucleotides. Note that we did not choose to take the default output name (we specified **-out=anothertry.map**) The default given would have been testseq2.map and this would have over-ridden our previous analysis of testseq2.seq.

To look more closely at the command line options for the program Map, type the following:

**map -check**

This gives you a listing of all the command line options available for this program.

**Please note:** the -check flag is specific to programs in the GCG package. Other programs may use **-help**, others still **-h** to bring up this type of information. Others still may have no equivalent....you need to check the documentation for a program to determine the correct syntax.

If you look at the list presented to you, you will see that there are a lot of options you were not prompted for before. A few examples are:

-CIRcular	treats the sequence as circular
-MISmatch=1	finds restriction sites with one or fewer mismatches
-SILent	finds translationally silent potential restriction sites
-MINSitelen=6	selects enzymes with 6 (or more) bases in recognition site
-MINCuts=2	shows only enzymes that cut at least 2 times
-MAXCuts=2	shows only enzymes that cut no more than 2 times
-ONCe	shows only enzymes that cut once

To choose any of these, or the other options not prompted for, you need to specify them on the command line. We will do this for one sequence, and then we will look at an efficient method for running a particular analysis on many sequences.

Let's make a restriction map of testseq3.seq, looking for enzymes that have recognition sites of at least six bases and that cut a minimum of twice. For anything that we want the default value for, we can just leave it out of the command line, but include the -def flag. **So type:**

**map -mins=6 -minc=2 -def testseq3.seq**

You should now have a file called testseq3.map in your directory. (List the contents of the directory and see. To do this, type **ls** on the command line.)

*As you can see, running the pure command line is somewhat faster than the prompted command line once you know what you want to do.*

## 8.5 Automating the command line

The command line truly comes into its own when you need to run an analysis over and over again. For example, what if I had 100 sequences I wanted to map exactly the same way? The prompted command line would become tedious very quickly. And the full command line would too, although it would be faster.

### **The solution involves something called a foreach loop.**

What this does is to take a list of, in this case, sequences, and say: ‘**Foreach** thing in this list, do the following:’

So, for a mapping analysis, you might want to do something like: ‘Foreach sequence in my list, run the program map to look for the enzymes that that have recognition sites of at least six bases and that cut a minimum of twice.’

In this way, mapping 100 sequences would only take a bit more time than mapping one sequence, and would require practically no extra effort on your part.

Since the general idea is to get the computer to read a list, and run an analysis on each item in that list, you need to generate a list of the sequences you want analysed. We will go through one example here.

#### **Note**

If you have no prior experience with Unix, you may find it a challenge to set up your own foreach loops the first time. Please contact us at [genmail@molbiol.ox.ac.uk](mailto:genmail@molbiol.ox.ac.uk) if you are trying to do this and we will help you.

Type the following on the command line:

```
ls testseq*.seq
```

You should see 5 sequences listed:

```
testseq1.seq testseq2.seq testseq3.seq testseq4.seq testseq5.seq
```

Let’s assume we wish to map these five sequences. We now know that the command

```
ls testseq*.seq
```

will list our sequences of interest. We will use this to help us get the computer to understand that we wish to run our analysis on each of these sequences.

The first step is to give the command:

```
foreach i ( ‘ls testseq*.seq’ )
```

Here there are several things to note:

- ◆ we have used the command **‘foreach’**
- ◆ the “i” means “each thing”
- ◆ as you saw above, the command **ls testseq\*.seq** lists the sequences we want to analyse. The quotation marks around the **ls testseq\*.seq** command are *backquotes* and the computer understands this to mean “take the results of the command inside these backquotes”.

(The backquote key is usually at the top left hand side of the keyboard.)

- ◆ The brackets are important: the computer needs them to understand what you want

So the overall effect of that one line is: **‘foreach thing in the list that can be generated using the command ls testseq\*.seq, do the following:’**

If you have typed the foreach line in, you will now be seeing a prompt like:

```
jbloggs@enterprise [demo] foreach i ( ‘ls testseq*.seq‘ )  
foreach?
```

After the foreach?, we need to tell the computer what it is it needs to do for each item in the list. So **type**:

```
map -in=$i -mins=6 -max=2 -def
```

and **press the return key**.

You will now see another foreach? Prompt. **Type**

```
end
```

to let the computer know that you have told it all it needs to do for each item in the list.

Now **type**:

```
ls -l *.map
```

You should see that you have run the mapping analysis on all 5 sequences.

As you can see, running this analysis on 100, or 1000 sequences, would be relatively painless if you did it using a foreach loop.

## 9. Giving files as input for programs

Some programs only accept a single sequence as input. However, others may be able to accept multiple sequences, and others still may require multiple sequences to work on, (for example, for multiple sequence alignment.)

How can you give a program more than one sequence to work on?

This depends on the program involved, but with GCG programs, there are two main methods:

1. you can give the names of the sequences you wish analysed in a list file, (a file of filenames)
2. you can enter a single file with many sequences within it - in GCG this can be an MSF file (multiple sequence file) or an RSF file (rich sequence format).

### 9.1 List files

A list file is essentially a file of filenames. They can be given to some GCG programs as input, but you have to warn the GCG program that what you are giving it is a list file.

To do this, you add the prefix `@` to your filename. We will look at this using the command **tofasta** in the next section.

First, we need to create a list file. This is just a list of the names of sequences you want to analyse.

Use the pico editor to create a list of sequences that we will convert into fasta format. To do this, type

```
pico
```

Then enter the following:

```
cd4_cerae.pep  
cd4_erypa.pep  
cd4_human.pep  
swissprot:cd4_mouse  
swissprot:cd4_rabbit  
swissprot:cd4_sheep
```

When you have inserted these six lines, **quit from the editor by pressing ctrl-x.**

**Save the file with the name *cd4.list***

This list file is a mixture of files that are in your `gcg_course` directory (in your account), and files in a protein database, Swissprot. If you list the files currently in your `gcg_course` directory, by typing

```
ls
```

You should see the files `cd4_cerae.pep`, `cd4_erypa.pep`, and `cd4_human.pep` there.

GCG will interpret the line **swissprot:cd4\_rabit** as meaning: “find the sequence with the ID cd4\_rabit in the swissprot sequence database, retrieve it, and use it in this analysis”. (cd4\_rabit is not a typo. This sequence really has been entered into the sequence databases with this name.)

GCG will interpret any entry written as **database:sequenceID** in the same way as described above. The database must be one that GCG know about. A list of these databases can be found at: [http://www.molbiol.ox.ac.uk/databases/gcg\\_farms.htm](http://www.molbiol.ox.ac.uk/databases/gcg_farms.htm)

Information about creating list files can also be found at:  
<http://www.molbiol.ox.ac.uk/help/unix/listfile.htm>

If you wanted to use files in any other directory, you will need to tell the machine where to look for them. To do this, you need to specify the *full path* of your files. For example:

```
/usr/users/user1/jbloggs/gcg_course/cd4_cerae.pep  
/usr/users/user1/jbloggs/gcg_course/cd4_erypa.pep
```

and so on. The **pwd** command can be very useful if you are not sure what the full path to your file is.

If you do not understand the concept of paths, please refer to the Introductory notes available on the web at: <http://www.molbiol.ox.ac.uk/tutorials/index.html>

## 9.2 Examples of running programs with single vs. multiple sequences as input

To illustrate these concepts, we will run the program **tofasta** to change a sequence's format from GCG to fasta. We will first run this on a single sequence, and then on a group of sequences.

### 10.2.1 Converting a single sequence:

First, look at the gcg-formatted sequence testseq1.seq. Type:

```
less testseq1.seq
```

Note what it looks like and **type q** to exit the **less** program.

Now type:

```
tofasta testseq1.seq
```

Accept the defaults when you are prompted by pressing the return key. You should see something like the following on your screen when you are done:

```
ToFastA converts GCG sequence(s) into FastA format.
```

```
Begin (* 1 *) ?  
End (* 430 *) ?  
Reverse (* No *) ?
```

What should I call the output file (\* testseq1.tfa \*) ?

```
TESTSEQ1 430 characters.
```

430 symbols written into "testseq1.tfa".

Look at the resulting file testseq1.tfa by typing:

```
less testseq1.tfa
```

Note that the header information has been almost entirely removed, and is now a single line starting with a >. It probably looks something like:

```
>testseq1
```

There can be as much information on the line with the > as you want. The word immediately following the > is often interpreted as the sequence name.

As you can see, the fasta format is very simple. The minimal requirements are the > at the start of a single header line, and the sequence (either protein or nucleotide) in a continuous block starting on the next line.

*Many non-GCG programs expect sequence in fasta format. You should read the documentation of a program to find out what format it expects sequence in, but if you cannot find that information, fasta format is a good first guess.*

Fasta format files created using tofasta have the ending **.tfa** by default.

### 10.2.2 Converting a group of sequences:

*tofasta* can also be used to build a fasta multiple sequence file using a listfile of filenames as input. This command can be very useful if you need to enter a group of fasta-formatted sequences into a program. Examples of programs that use such input are **dotter** and **clustalw**.

Type:

```
tofasta @cd4.list
```

Notice that you need to write an @ symbol directly before your list name (no spaces!) to make GCG understand you are giving it a list of sequence names.

Accept defaults by pressing the return key - accept all defaults except for output name - call it **cd4.tfa** instead

This should produce an output file called cd4.tfa. Look at it by typing:

```
less cd4.tfa
```

Notice that a multiple sequence fasta format file is made up of a series of single sequences, one after another.

### 9.3 Multiple Sequence Files in GCG format

List files are one way to input multiple sequences into a program. However, some GCG programs require a multiple sequence file. In contrast to a listfile, this type of file contains the sequences themselves. The file `cd4.tfa` is a multiple sequence file, however it is in fasta, not GCG, format.

The multiple sequence files you may require as input for GCG programs are usually produced in the course of earlier steps of your analysis.

For example, the output of the GCG multiple sequence alignment program **pileup** is **msf** format, a GCG multiple sequence format. Accordingly, the alignment program `pileup` gives its output files the default suffix **.msf**.

*Please note: We generally recommend the program `clustalw` for multiple sequence alignments, not `pileup`!*

To use an `msf` file as input to another GCG program, you need to warn the program that you are about to input a number of sequences, and you need to specify which sequences you want analysed. To warn GCG programs that there are multiple sequences in your `msf` file, you need to include `{ }` at the end of the filename. Inside these brackets, you can include the names of the sequences you want included in the analysis. If you want all the sequences included, you can leave the braces empty, or you can put the `*` wild card in: `{*}`.

**If this is not done the program you are attempting to run will complain that it can't read a sequence.**

You have a sample multiple sequence alignment file in your `gcg_course` directory called **pileup.msfc**. Take a look at how the sequences appear in this file. **Type**:

```
less pileup.msfc
```

Now run **tofasta** on these sequences. **Type**:

```
tofasta pileup.msfc{*}
```

When you are prompted for an output name, **type**:

```
pileup.tfa
```

Take a look at this file. Type:

```
less pileup.tfa
```

Fasta format files can also be changed to GCG format using the command **fromfasta**. This command can be used to reformat single sequences, or a file containing multiple

sequences. To read more about this command, look in `genhelp`, `genmanual`, or on the web pages.

#### 9.4 More notes on formatting

The Staden package produces, and expects files in its own staden format, which contains a single sequence in a block, with no header lines at all. By convention, Staden files have the ending `.sdn`. Staden format files only ever contain a single sequence. Staden format files are converted to GCG format using **fromstaden** and in the opposite direction using **tostaden**.

Finally, there is another GCG reformatting command: **reformat**. Reformat reformats sequence files into GCG format but **BE WARNED** - it gives no parameter prompts and may not do what you expect!! It will reformat anything, even if it is not sequence data at all, and will not report any errors to you. *We do not recommend reformat.*

We recommend that you try using the non-GCG program `readseq` for most reformatting jobs. More information about `readseq` is available at: <http://www.molbiol.ox.ac.uk/help/sequenceformats.htm>

## 10. Sequence Retrieval

There are two related GCG command-line programs that retrieve sequences from biological databases: **fetch** and **typedata**. Both programs take the sequence accession number or the database name of a sequence and retrieve that database entry. The only difference between them is that **fetch** makes a copy of the sequence in your directory, while **typedata** copies the database information to the screen rather than to a file. Both programs use the same optional parameters. We will concern ourselves mostly with **fetch**.

Try using **fetch** to retrieve the sequence of rat CD4 from the Swissprot protein database. Type:

```
fetch
```

Fetch copies GCG sequences or data files from the GCG database into your directory or displays them on your terminal screen.

```
FETCH what sequence(s)?
```

```
Type    cd4_rat
```

You should then see the output name on screen:

```
cd4_rat.swissprot
```

Notice that the sequence is stored in your directory as a file called `cd4_rat.swissprot`. Have a look at the retrieved sequence file. Type:

```
less cd4_rat.swissprot
```

Notice that the file is separated into two main parts, the actual sequence itself; in the lower portion of the file, and a header section of sequence-related information in the top half. The header section contains information describing the sequence, including the species from which it was taken, the date the sequence was created, and last modified, the references of any related papers, the position and identity of any known structural features and often other information as well.

Fetch does have an annoying quirk that it is worth knowing about though:

Read through the `cd4_rat.swissprot` file. In it there is a reference to the corresponding EMBL nucleotide entry. The accession number for this is `m15768`.

**Try and fetch this nucleotide sequence in the same way you just fetched the swissprot entry.**

**Take a look at the resulting file using the command *less*.**

Is this what you expected to see? Remember, you were trying to fetch the corresponding nucleotide entry for the Swissprot protein sequence `cd4_rat`.

Why did you end up fetching a peptide file? **Fetch** looks for an accession number in the peptide databases first. If it finds the entry in a peptide database, it returns it to you, and does not look any further.

To force **fetch** to get the nucleotide entry, you need to specify the name of the database within which fetch should search for the entry. In this case it is `Embl`. Try typing:

```
fetch embl:m15768
```

If you do this, you should get a file called: **m15768.emrod**

Fetch has some options that modify the way the program works.

To see what options are available, type:

```
fetch -check
```

For instance:

```
fetch -outfile=newfilename
```

tells the program **fetch** to save whatever sequence you later specify into a new file called `newfilename`.

```
fetch -doclines=6
```

tells **fetch** to only copy the first 6 non-blank lines of the header section of the file.

If you merely want to have a look at a particular sequence in the database, and don't want to save a copy of it to a file, try:

```
typedata filename | less
```

Here you are piping the information through the command `less`, which allows you to view the sequence entry one screen at a time.

Once we have retrieved a sequence and saved it in a directory, we can study it further.

## 11. Dealing with graphical results

As you have seen, you can view text output, you can use the `more` command, or the `less` command .

**more filename**

**less filename**

In order to view any graphical output of a GCG program, you will need to run the **setplot** command, (once in each session) to tell the system what sort of terminal you have.

### 11.1 Running pepplot

To illustrate this, we will run the program `pepplot`, which produces graphical results.

You must setup your session to deal with graphics before you run the program. Type:

**setplot**

Today you are need a **Color X windows Graphics Window**. If you are not running Xwindows (`exceed` or `exodus`) on your local machine, you will need to choose another option. When the appropriate option is highlighted, press the return key.

A new window should appear entitled `GCG_graphics`.

**Do not close the graphics window!** It must be left open in order to work. If you close it, you need to open a new one before you generate graphical results again.

If you want to make sure you have chosen the appropriate graphics type, you can check. Type:

**plottest**

on the command line in your xterm window. You should see a diagram appear in the window.

In your xterm window, type:

**pepplot cd4\_rat.swissprot**

You can refer to the documentation to find out what `pepplot` does.

Accept all the defaults by pressing the return key after each prompt.

The results should appear in the GCG\_graphics window.

***Saving results produced this way***

Some programs produce savable graphical results by default (e.g. prettybox, which you will look at later.) Peplot does not. To save such a figure, the easiest thing to do is redefine where you want your graphics sent. That is, close your currently GCG\_Graphics window, and type

**setplot**

on the command line. This time, choose the option:

**CPSX    Hardcopy - Plot to colour postscript file plot.ps (copy)**

**Now run peplot again.**

This causes the results to be written to a file called **plot.ps**, which is a colour postscript file. This file can be **viewed** using the program ghostview

**ghostview plot.ps &**

Or **printed** on a postscript printer by typing

**psprint plot.ps**

Should you create postscript files like this, we **HIGHLY RECOMMEND** that you change the name as soon as you can. Otherwise you may find that you are overwriting files you wished to keep with new results.

[Ghostview is discussed further later in the practical.]

Although you may have the general gist of working with command line GCG programs by now, you may want to carry out the following example. (If you are running short of time, move onto the next section: **13. Working in SeqLab**)

**Translating a nucleotide sequence to a peptide sequence**

The program **translate** translates a single nucleotide sequence into a peptide sequence using the reading frame specified by the user. To run this program on the sequence testseq1.seq, type:

**translate testseq1.seq**

You can choose to translate in any forward frame by choosing which nucleotide to start your translation with. When prompted in this example, type 3.

Translate translates nucleotide sequences into peptide sequences.

```
Begin (* 1 *) ? 3
End (* 430 *) ?
Reverse (* No *) ?
```

Range begins AATGA and ends AAAGA. Is this correct (\* Yes \*)  
?

That is done, now would you like to:

- A) Add another exon from this sequence
- B) Add another exon from a new sequence
  
- C) Translate and then add more genes from this sequence
- D) Translate and then add more genes from a new sequence
  
- W) Translate assembly and write everything into a file

**Please choose one (\* W \*):**

**Next it will say:**

What should I call the output file (\* testseq1.pep \*) ?

Look at the resulting peptide sequence. Type:

**less testseq1.pep.**

Try running translate on the reverse strand by using the prompted option reverse.

If you have time, look at some of the options available for the program translate by referring to the documentation, or typing

**translate -check**

## 12. Opening SeqLab

To start SeqLab, you have two choices:

1) if you have initialised gcg once already in your window, (and if you are working through this practical, you will have), you type:

**seqlab &**

(for an explanation of the &, see Appendix 1)

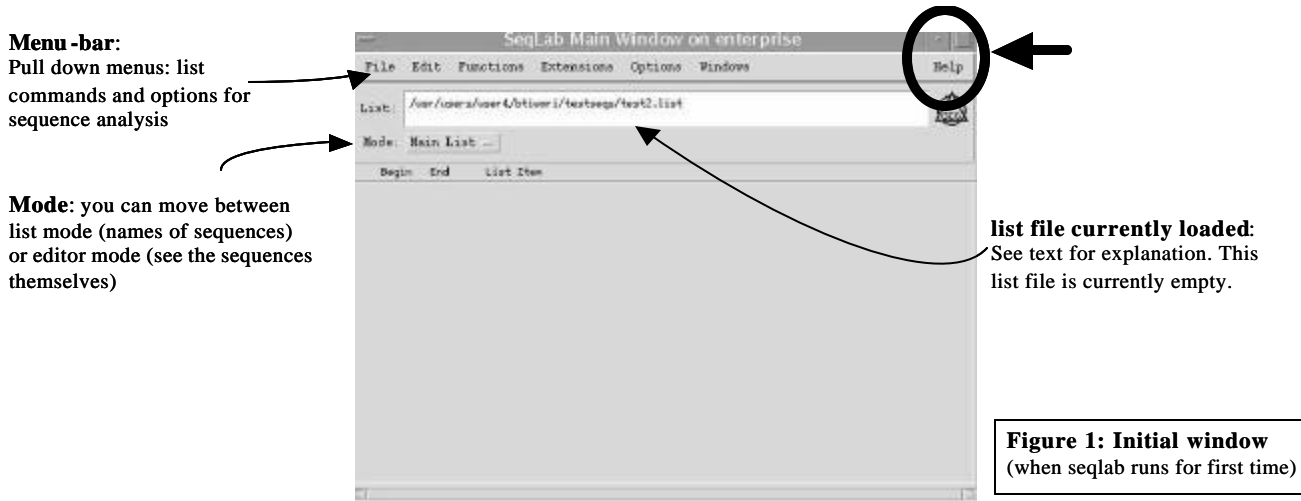
2) If you have not initialised gcg yet, you could type:

**xgcg &**

This command runs the gcg initialisation and the opening of SeqLab in a single step.

Two windows pop up. The top window gives some information about the software. Click on the OK button to make it disappear.

You now should have a window that looks like that in Figure 1.



Click on the help button and look at your options. The most useful are:

**On the Interface:** instructions on how to work in the SeqLab interface

**On Help:** How to use the online help

**On the Wisconsin Package:** detailed descriptions of the programs in GCG.

Take some time now, or later in the tutorial, to look more closely at the help available.

Click on each of the drop down menus and see what options they present.

Below these menus you see **List** followed by the name of the list file you are currently working with. This is discussed further below.

In figure 1, the name of the list file will be something like:

```
/usr/users/user1/jbloggs/gcg_course/working.list
```

Note: SeqLab records a file called .seqlab-history in your home directory. This file records the details of your SeqLab session when you exit and uses this information to set up SeqLab the next time you run it.

This means that if you want to start using SeqLab from within a different directory than gcg\_course, you should start it up from within that new directory, choose Preferences from under the Options menu, and enter your new Working Dir, (i.e. the directory you are running SeqLab from.)

### 13. The SeqLab environment

SeqLab has two “modes”: **list mode** and **editor mode**.

In List mode, you see the list of sequences you are currently working on. To this list, you can add sequences from your own files, or from the databases. Information displayed to you in this mode includes the name of a sequence, its location, whether it is nucleotide or protein, etc, etc.

In the Editor mode you can see your sequences, create new sequences, or edit sequences. In addition, you can create or edit an alignment, and work with sequence features. See figure 3.

#### 13.1. List Mode

The first time you start up SeqLab, it opens in List mode and presents a default list called working.list. After this first session, SeqLab will open in List mode, but will

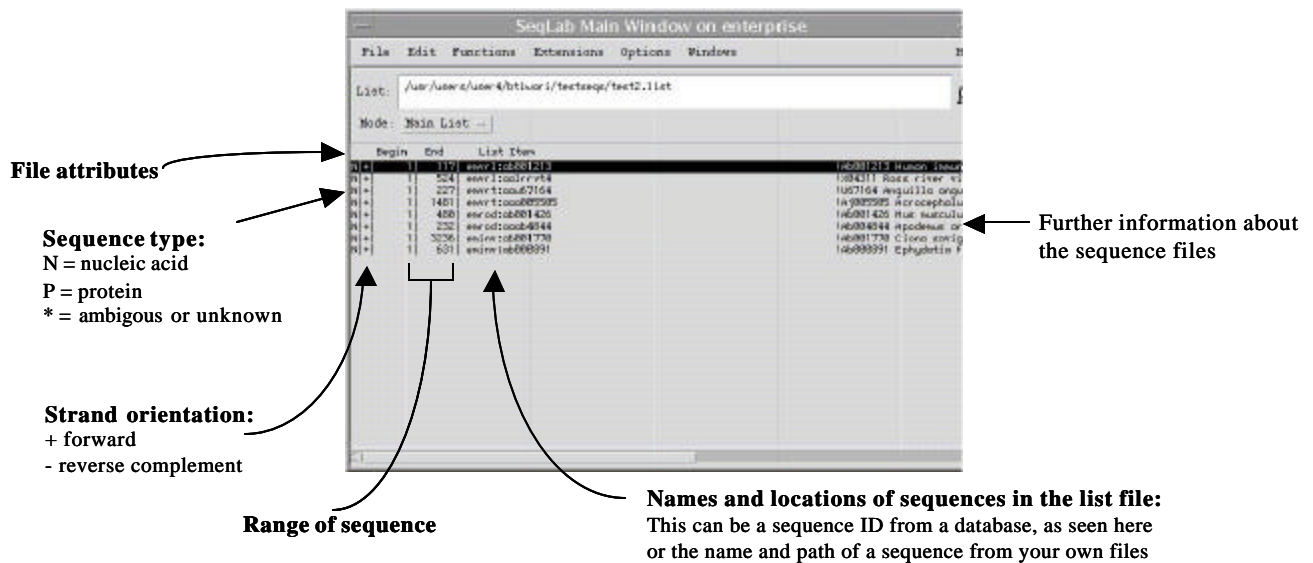


Figure 2: Main List Mode

open the listfile you were working on in your last SeqLab session.

After you have added the sequences you want, you can change the name of the list from “working.list” to something else. You do this by using the “Save As” option under the File menu. It is advisable to leave the suffix “.list” on whatever name you choose so that you can see right away that it is a list file.

List files are very useful when you need to manage multiple projects. Each project can have its own list of sequences.

*Recall: Lists include the names and locations of the sequences you want to work with.*

You can now:

1. Start a new list by going to the File menu and clicking on New List, or
2. Open a list file you have already been working with by going to the File menu and clicking on Open List.

List files can contain names of single sequences from databases or your own files, or can contain files that contain multiple sequences, such as the output from a multiple sequence alignment program. Remember, only GCG format files are recognised by GCG programs.

### 13.2. Editor Mode

In the Editor mode, you can see your sequences, create new sequences, or edit sequences. In addition, you can create or edit an alignment and work with sequence features. See figure 3. These options are discussed further later in the tutorial



Figure 3: Editor Mode

## 14. Adding Sequences to Seqlab

Make sure you are in Main List Mode

You are now ready to add sequences to your list. You can do this two ways:

1. Add sequences one at a time from files in your account.
2. Add sequences directly from databases using sequence ID's.

### 14.1 Add sequences one at a time from files in your account

Select the Add Sequences From option under the File pulldown menu. You then get a further choice of Databases or Sequence Files.

#### Choose Sequence Files.

A new window called **Add Sequences** appears. See figure 4 below

#### A Note about Filters

At the top of this new window is the filter. Files with names which match the filenames and location you provide in the filter box will be listed in the right hand box under the word Files.

For example, in figure 4, the filter is set to:

**`/usr/users/user1/jbloggs/testdir/*.seq`**

Using this as the filter, the only documents listed in the Files box will be ones inside the directory called testdir, and which have a filename that starts with anything, but end with the characters .seq

(If you do not understand the use of \* in the filter statement above, please refer to the General Knowledge section near the end of this tutorial.)

Any directories within the testdir directory are listed in the Directories box.

You can change the filter to look in any directory within your account, and to list any filenames you like. To list all filenames within a certain directory, enter \* after the directory name. e.g.

`/usr/users/user1/jbloggs/gcg_course/*`

To choose a particular sequence to add to the list, click on the sequence name and then click on the **Add** button

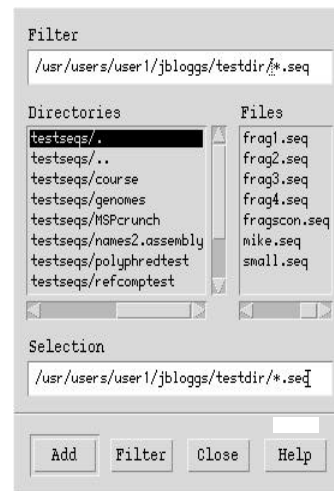


Figure 4: Choosing sequences from files.

If you have any sequences in gcg format in your account, you could enter them into the list now.

If you know the sequence name and where it is located, you can just type this directly into the box called Selection.

For example, add the sequences:

```
cd4_cerae.pep
cd4_human.pep
```

## 14.2 . Add sequences directly from databases using sequence ID's.

To do this, **go to the Add Sequences From option under File .**

You get a further choice: Databases or Sequence Files.

### **Choose Databases.**

You get a choice of a number of databases to choose from.

### **Click on the database of your choice.**

Notice the box at the bottom of this window called **Database Specification.**

In here you can specify the database you want to search.

You can create a temporary list of sequences by clicking on the Show Matching Entries button on the right hand side of the window. Any sequence matching the pattern in the Database Specification box will be listed in the right hand panel. Double clicking on any sequence in that panel adds it to your Main List.

If you know the sequence ID, you can write it directly in the database specification box.

### **Add the following sequences to your list.**

```
emmam:epcd4
emmam:mfd348
emmam:mnd346
embl:e12615
swissprot:cd4_macmu
swissprot:cd4_rabit
```

### ***Looking at a sequence:***

- You can view any sequence entry shown in the Entries section of the window by highlighting it and then clicking on **View Sequence**.
- If you move the cursor over a particular sequence name in your Main List, and double click, the sequence itself appear in a separate window.

## 15. Editor mode

To view a sequence in Editor Mode, you need to highlight it in the Main List.

**Click on a sequence name to highlight it.**

To highlight more than one sequence name found *consecutively* in the list, you can press the Shift key down, and while it is depressed, press the left mouse button, moving the cursor over the names of sequences you are interested in.

To highlight *non-consecutive* sequences in the list, do the same as above, but use the Control key instead of the Shift key.

**Highlight the sequences you are interested in**

(Try to pick at least one peptide sequence and one nucleotide sequence.)

To get into editor mode, click on the button next to the word **Mode** and drag the cursor over the word **Editor**.

**Move into editor mode.**

You should see something like Figure 3.

### 15.1 Adding sequences directly from within the Editor mode

The process is the same as when adding sequences into the main list, except that when you go the **Add Sequences From** under the **File** menu, you are given the extra option, **Main List**.

**There is a catch to adding sequences directly into the Editor:**

you must expressly save them using the **Save As** option under the File menu. If you do not do this, the sequence is lost as soon as you go back to List mode. Using "Save As" saves the file into your account AND enters it into the main list.

Try adding the sequence:

**emmam:cacd42**

directly into the Editor window. **Now change back to List mode**. Is the sequence there?

**Now go back to Editor mode**. What happened?

**Add this sequence into the Editor window again. Now use the Save As option to save it. Go back to List Mode. Is it there now?**

Go back to your xterm window and type:

**ls**

Is the sequence in your account?

## 15.2 Features Coloring and Residue Coloring

Beside the word **Display** in the Editor mode window, you can choose from a menu of options for what basis you wish to see your sequences coloured on. Two of the most popular options are **Residue Coloring**, where each base or amino acid is given a different colour, and **Features Coloring**, where defined features are given a particular colour.

**Place swissprot:cd4\_macmu in the Editor mode screen.**

**Highlight the sequence name and change the color option to Features Coloring.**

For more information on this, please see Appendix 3, which discusses rsf format files.

With the sequence name still highlighted in Editor mode, you can click on the info button (see figure 3), to bring up the annotation associated with that sequence. As you scroll down, you should find the features section of the annotation (the lines beginning with FT).

**Click on the info button**

**Close this information window by clicking on OK or Cancel.**

**Now place your cursor over the sequence itself, which should still be coloured according to its features.**

**Double click on the sequence.**

You should be presented with a new window. Next to the word **Show** in this window, you can choose between **Features at Cursor**, and **All features** in current sequence.

**Try both these options.**

**Now close the Sequence Features Window.**

**Change the Display: option back to Residue Coloring**

## 15.3 Editing sequences in SeqLab

One of the most useful aspects of SeqLab is the editing you can carry out when you are in Editor mode. We will cover a few of the basics here.

By default, you are allowed to add gaps and reverse/complement a sequence.

By altering the permissions on a sequence, you can allow editing of residues marked N or X, or allow the editing of defined amino acid or nucleotide residues. You can also remove permission to do any of these things which reduces the danger of doing things like leaning on your keyboard and changing your sequence by accident!

**Highlight a sequence name in the Editor Mode**

**Try adding gaps to the sequence. Try removing or overwriting residues of**

**the sequence.**

**Click on the button marked Protect (with a small icon of a lock on it).**

**Look at the options. If there are any you do not understand, click on the**

**Help button**

**Place a check mark in the boxes for N or X and All other characters.**

**Try adding removing or overwriting residues of the sequence now.**

#### **15.4 Editing multiple sequences**

Editing can be particularly important when you are attempting to create a good multiple sequence alignment. The programs may produce approximately what you want, but you may need to hand edit to get it right.

In general, there should be no need to edit the actual sequence residues in a multiple sequence alignment. You should only need to add, remove or reposition gaps. *The individual sequences should be correct before you start aligning them!*

We will illustrate editing multiple sequences with a couple of unaligned sequences. The process is the same for aligned sequences.

**Go back to the Main List**

**Highlight swissprot:cd4\_macmu and cd4\_human.pep**

**Enter Editor Mode**

**Add some gaps to one sequence.**

**Remove the gaps.**

**Now highlight both sequence names, and choose the option Group from under the Edit menu.**

**Now add some gaps to one sequence. What happens?**

If you had an alignment made up of several groups of sequences, you can group the ones that you need to move together.

**Choose to go back to the Main List mode**

A box offering you the chance to save your files, (usually in rsf format, see Appendix 3) will appear. If you do not save the files now, they will be lost.

**For the purposes of this exercise, just click on *Don't Save*.**

## 15.5 Adding Comments to your sequence

You can add comments under a sequence, or sequences when you are in Editor mode.

To do this, you essentially ask SeqLab to create a new sequence, but instead of sequence data, you enter comments. If you then group the sequence of interest, with the line containing the comments, and choose to save them before you exit Editor Mode, you will effectively have underwritten your sequence with a series of comments or labels.

**Highlight cd4\_human.pep in the Main List**

**Enter Editor Mode**

**Choose New Sequence from under the File menu**

**Choose Text for the type of sequence**

**In the line labeled NewText, you can type what you like, where you like. Try entering some comments at about position 50, and at position 320**

**Now group cd4\_human.pep and NewText.**

**Enter some gaps into cd4\_human.pep**

**Choose to go back to Main List Mode.**

**Save your sequences under a name of your choice when prompted.**

**Now highlight this in the Main List and re-enter the Editor Mode.**

**You should see you sequence and comments in the Editor window.**

If you wish to change the name of NewText to something meaningful (a good idea!), highlight NewText in the Editor window, and click on the Info button. In the box marked Name, you can change the name to whatever you like. If you click on OK, you should see your new name in the Editor window.

## 15.6 Other options you have when you work with SeqLab

1.) You can open a pre-prepared list file of sequences. This can be a list file you have saved from a previous SeqLab session, or you can make a list file using a text editor. (It's easy!) Details on how to create a list file were given previously in the practical. You could also refer to the following web page:

***<http://www.molbiol.ox.ac.uk/help/unix/listfile.htm>***

2.) Under the Edit file you will see the options **Remove from List** and **Delete File from Disk**. If you choose to remove a file from the list, ***you are only removing it from the list!*** The file itself will still be in your account. If you want to delete the file entirely, you need to choose and Delete File from Disk.

3.) Enter a sequence manually. This is done in the editor mode and instructions are given in Appendix 2 as this will not be addressed directly during this tutorial and is **not recommended!**

## 16. Running Programs

Programs in SeqLab are grouped by function.

**Look at the list under the Functions menu.**

The general outline of how to run a program in SeqLab is always the same:

- 1) select the sequences you wish to analyse, either in the Main List, or in the Editor Mode. To indicate that you want to carry out an analysis on a given sequence or sequences, the names of these sequences **MUST** be highlighted.
- 2) Select the program you want to run by choosing from the menus under the Functions or Extensions menu.
- 3) A new window then appears and you must fill in information about how you wish the program to run. The exact options offered depend on the program you are running. In all cases you will need to specify which queue to send your job to, if any. (See below for details.)
- 4) Click on the **Run** button to run the program
- 5) View the results as described below.

### 16.1 Getting and Viewing the Results

The output of a program will not appear in the SeqLab window you are working in! You have to refer to two other windows: the **Job Manager** and the **Output Manager**, which can be accessed by choosing the option(s) under the Windows menu.

**Make sure you are in Main List mode**

**Highlight swissprot:cd4\_macmu and swissprot:cd4\_cerae**

**Now change to the Editor mode.**

**Highlight both sequence names**

**Go to the functions menu and choose Pairwise Comparison, and from the submenu choose Bestfit.**

**In the new window that appears, click on the Options button and decide**

**whether you want, or need, to change anything.** If there are options that you do not understand, you should refer to the documentation.

**Click the close button in the Options window, and click on the Help button in the Bestfit window to bring up the documentation.**

**Double click on the subject Parameter Reference to find out what any options you do not understand mean.**

**Click on the Close button, and then open up the Options window again.**

**Select and change any options you may want to try.**

**When you have set the options you want, click on the Close button in this window.**

In the main Bestfit window, there is a box under the words **Command Line** that shows you what you would have had to have written on the command line to get the same options to apply. This is a handy way to learn the command line syntax!

**When you have chosen all the options you want, click on the Run button ONCE.**

You will not see anything happening, but your job is running, or has been sent to the queue you specified and will be processed shortly.

***Do not continue to click on the Run button!!!***

## **16.2 What is the How: option asking me?**

A very important button to take note of is the one at the bottom next to the word How. By default, many of the programs run through SeqLab are sent to the batch queues.

Batch queues allow the machine to decide when it has time to run jobs. Different queues have different priorities; big jobs should be sent to queues that give you a lot of time, but are run with low priority on the machine.

The choices available when you click on the How button refer to how much computer time you will be allowed for your job to run and what priority it is run at.

The general rule is that the longer you think the job will take, the longer you should be willing to wait for the results. This allows people with very small jobs to get their results quickly, while the machine can store the bigger jobs and run them when it deems it has time/energy.

If you have a very long job to run, and you decide to be selfish and run it with the “**Normal**” choice, it may run fine...but it will get cut off dead if it takes too much time! So it is in your best interest to choose the “Long”, “Tonight” or even “Forever” choices if you think your job will take a very long time to run. The jobs we are carrying out today are not very big, so we will stay with the Normal option.

More information about Batch Queues is available at:

**<http://www.molbiol.ox.ac.uk/help/batchqueue.htm>**

***Please note that the “Normal” queue as it is named under SeqLab, is equivalent to the “Batch” queue in other documentation.***

### 16.3 Job Manager

If you **open the Job Manager window** now, you should see information about the Bestfit job that you just submitted. Click on the help button to find information about what is possible using the Job Manager.

### 16.4 Output Manager

You can **access the results** of any analysis you run during your SeqLab session through the Output Manager (see below). You can get to the Output Manager either from within the Job Manager, or from under the windows menu in your main SeqLab window.

### 16.5 Dealing with the Results Files in the Output Manager

Results from analyses can be considered according to whether they can be used as input to other programs or not.

For example, the output of the program Bestfit returns the top scoring local alignment between two sequences. These results cannot be used as input to another program, although you may wish to save them to view later.

However, you might run a multiple sequence alignment using Pileup or Clustalw, and want to submit the resulting alignment to another program, for example: Pretty or PrettyBox.

If you wish to use the output of one analysis program as input to another within the SeqLab environment, *you must add the results from the first program into the Main List.*

**To do this, highlight the name of the output file in the output manager, and then click on Add to Main List.**

Helpfully, if the results of a particular analysis cannot be used as input to another GCG program in SeqLab, the buttons labeled Add to Main List and Add to Editor will be “greyed out” and you will be unable to click on them.

SeqLab does not allow you to choose a name for your results file before a program runs. You can, however, change the name of the output file to something meaningful afterwards. The fastest way to do this is to click on the “Save As” button in the Output Manager, and save the file under the name you want.

***There is a big catch when you do this though!***

SeqLab does indeed **copy** your results to the new filename you have, and in the Output Manager it now looks as if you have one copy of your results, (with the new name). However, in fact, there are now two copies of the results in your account, one with the old name, and one with the new name. You need to manually delete the second copy in your account to get rid of it!!

Removing the original copy should be done on the command line, using the **rm** command. You will find yourself accumulating a large number of unwanted files otherwise.

More information about removing files manually can be found at:

[http://www.molbiol.ox.ac.uk/help/unix\\_help.html](http://www.molbiol.ox.ac.uk/help/unix_help.html)

## 17. Printing from SeqLab

You can directly print text files from within SeqLab. (We discuss how to deal with graphical output later in the practical.)

There is a catch though! By default, SeqLab may not be configured correctly.

When you click on any button in SeqLab that gives you the option to print, a window labeled Print Text will appear.

If the Output format is ASCII, you need to ensure that the ASCII print command is **a2ps** *not* **lpr**.

(ASCII format refers to plain text.)

The other option is to print in postscript format. Postscript is the format that graphical results from GCG programs are saved in. This is quite different than text format, and to use this you must have a printer capable of understanding postscript and you can leave the Postscript print command as **lpr**.

*That's it! That's the basics of SeqLab. Below we have included a set of exercises that you should be able to carry out using SeqLab.*

## 18. Exercises

Remember, there is help available on almost every window presented to you in SeqLab. If you aren't sure about something, or want to know more about it, try clicking on the help button!

If you have any questions or problems, please call over a demonstrator.

### 18.1 Mapping Programs

#### 18.1.1 Map

You have already used Map on the command line earlier in the practical. It does a restriction mapping of a DNA sequence and displays both strands of the mapped sequence with restriction enzyme cut points above the sequence. Protein translations can be produced below the sequence if desired.

Map can also create a peptide map of an amino acid sequence.

**Highlight a nucleotide sequence in your list file.**

**Under the Functions menu, find Mapping, and choose Map.**

**You are presented with a choice of enzymes you wish to cut your sequence with,**

(by clicking on the **Enzymes** button), or you can click on the box beside “**No enzymes**”.

**In the “Display Protein Translation Frames” box, check the boxes beside the frames you wish to see translated in your output.**

**Click on the “options” box.**

**Read through the options available to you. Click on boxes that you are interested in.**

**During this practical, try running the program translating in 3 frames and then in 6 frames.**

Compare the output when you check the box next to open frames only in the “Display Protein Translation Frames” box. You are encouraged to try the various options available to you using Map on any of the nucleotide sequences available.

### **18.1.2 MapSort**

MapSort finds the co-ordinates of the restriction enzyme cuts in a DNA sequence and sorts the fragments of the resulting digest by size.

MapSort can sort the fragments from single or multiple enzyme digests. The output from MapSort can also be used to create plasmid maps.

We will try running MapSort on the sequence **embl:e12615** from the sequences you have in your main list.

**Highlight embl:e12615 in your main list.**

**Under the Functions menu, find Mapping, and choose MapSort. Just like in Map, you get a choice of enzymes to cut with.**

**Click on the options box. Notice the options under “Output Format”**

**Run this program at least 3 times to see the different output you get when you:**

-click beside “output results with options indicated below:” (note: look at the options below as well!)

-click beside “output results as a tick-style label file for PLASMID MAP”

-click beside “output results as a block-style label file for PLASMID MAP” Protein analysis

Notice that MapSort will produce the figures using PlasmidMap automatically if you check the appropriate output options. If you ran this on the command line, you would have to run MasSort with the appropriate options, run setplot to get the graphics set up properly, and then run PlasmidMap on the output from MapSort in order to view this figure! In this case, running the program through SeqLab is easier.

## 18.2 Protein Analysis Programs

Under the Functions Menu, you will see “Protein Analysis”. You are welcome to try any of the programs you like. We have included an example of how to run the Motif program below. Remember, if you don’t know what a program does, click on it and then click on the Help button in the program window, or perhaps look up the help for that program on our web pages.

### 18.2.1 Motifs

Motifs looks for sequence motifs by searching through proteins for the patterns defined in the PROSITE Dictionary of Protein Sites and Patterns (<http://www.expasy.ch/prosite/>).

**Highlight swissprot:cd4\_cerae in your Main List**

**Go to the Functions menu, and under the Protein analysis option, click on Motifs**

**Click on the Patterns button.**

**Choose the patterns you wish to search for (or Add All). Then click on Close.**

**Click on the options button. Choose any options you are interested in.**

**Return to the Motifs window and click on the Run button.**

## 18.3 Multiple Sequence Alignment Programs

There are two multiple sequence alignment programs available through SeqLab. One is the GCG program **Pileup**, and the other is a non-GCG program called **Clustalw**. We will look at both, but between the two, we recommend Clustalw!

### 18.3.1 Clustalw

Clustalw is not a GCG program, but has been set up so it is available through the SeqLab interface. It is available under the Extensions menu.

Clustalw is also available on the command line, and there is a very good windows-like version of this program available (outside of gcg) by typing:

**clustalx &**

on the command line. Please give clustalx a try if you have the time.

*To run clustalw through SeqLab:*

**While in Main List mode, highlight the 4 cd4 peptide sequences.**

**Go to the Extensions Menu and click on ClustalW.**

**Choose any options you want and run the program. As usual, your results will be retrievable through the Output Manager.**

**Change the name of the output file (which will probably be something like clustalw.msf) to something meaningful.**

Because we are going to use these results as input to another program, we need to add the results file, when it appears, to the Main List.

**To do this, highlight the name of the results file in the Output Manager, and click on the button Add to Main List.**

#### **Notes about using ClustalW through SeqLab:**

Although some options are presented to you in this version of ClustalW, there are other, very useful options, that you cannot access by running it through SeqLab. Examples are:

*1) the ability to choose the output format of your sequences*

This is very useful if you need to use your multiple sequence alignment file as input to other non-GCG programs. Through SeqLab, you can only produce GCG format files, (e.g. RSF or MSF). Through command line clustalw, or the graphical clustalx, you can produce clustal, gcg, phylip, GDE or NBRF/PIR formats.

*2) the ability to add a single sequence to an alignment of other sequences.* This is known as profile alignment in clustal, and is not available through the SeqLab version of the program.

*3) the phylogenetics functions available through clustal normally, are not available through the SeqLab version.*

### **18.3.2 Pileup**

**Within Main List mode, highlight the 4 cd4 peptide sequences.**

**Got to the Functions menu, and under the Multiple Comparison option, click on Pileup.**

**Click on Run.**

**Add the results file to your Main List.**

#### **Notes about using Pileup through SeqLab:**

Under options, there is a choice to “**Plot dendrogram of clustering relationships**”. If this box is checked, you will be presented with something that looks like a tree and is called something like pileup\_##.figure.

**THIS FIGURE IS NOT A PHYLOGENETIC TREE!!! Please Close this figure! It means nothing!**

If you want to create phylogenetic trees, please consider using **clustal** (outside of SeqLab), programs of the PHYLIP package, PAUP, PAML, PUZZLE or fastDNAML. You will need to read the documentation for these programs before running them!

We suggest you leave the “Plot dendrogram of clustering relationships” box unchecked, (or uncheck it if it is already checked), so that this figure is never generated.

## 18.4 After the Multiple Sequence Alignment

There are a number of programs you can run to make your multiple sequence alignment look like you want it to look. Common examples are: Pretty, PrettyBox, and PrettyPlot.

Pretty and PrettyBox are GCG Programs.

PrettyPlot is not a GCG program: it is an EMBOSS program available on the command line or via a web-form. If you want to know more about PrettyPlot, please ask a demonstrator.

We will use your multiple sequence alignments as input to Pretty and PrettyBox.

### 18.4.1 Pretty

**Highlight the output file from your Pileup multiple sequence alignment in your main list.**

**Go to the Functions menu, and under the Multiple Comparison option, click on Pretty.**

**Click on the Options button.**

The window you now see gives you choices that will determine how your multiple sequence alignment will be presented to you. For example, do you want a consensus sequence or not, do you want positions that are the same as the consensus in upper case letters while those disagreeing are in lower case, etc.

If you want to know more about how Pretty creates the consensus sequence, click on the help button in the main Pretty window, and double click on the “Calculating and Displaying a Consensus” option.

**Run Pretty a couple of times with different options to get a feel for the kind of output you can create.**

### 18.4.2 PrettyBox

PrettyBox is a popular program for shading characters in an alignment according to how they compare to a consensus sequence.

Running PrettyBox on the command line is quite straightforward, but there are a couple of tricks to running it through SeqLab.

One of the tricks being that although you can run this program through SeqLab, you must go back your xterm window in order to view the results!

**Highlight the output file from your clustalw multiple sequence alignment in your main list.**

**Go to the Functions menu, and under the Multiple Comparison option, click on PrettyBox.**

**Click on the Options box and choose the options you want.**

It would be helpful for later examples if you choose to show a consensus sequence.

**Now run the program**

PrettyBox produces a file in what is known as “postscript” format. This is a computer language, and the results you see in the Output Manager are just postscript code. You must view this file through a program able to understand this code to generate the diagram

First you must save your PrettyBox results into your account.

**Click on the Save As button in the Output Manager.**

The usual suffix for a postscript file is **.ps**. It is probably best to give your file a name that ends in **.ps** so that in the future, you would know that this file is in postscript format.

**Give your file a reasonable name.**

Below I have assumed the file is called myprettybox.ps.

Ghostview is a program that allows you to look at postscript files. Go back to your xterm and, on the command line, type:

```
ghostview myprettybox.ps &
```

If you wanted to print a postscript file, you can use the **psprint** command. E.g.

```
psprint myprettybox.ps
```

Note: Postscript is not fun to work with if you are trying to display your file in other graphics programs like Adobe or PowerPoint! For information on how to change a file from postscript format to other graphics formats (e.g. tiff, jpeg, etc.) see Appendix 4.

## **18.5 Weights and Multiple Sequence Alignments**

*Before we deal with the issue of weights, make sure you have saved the results of your multiple sequence alignments into your account. To do this you must highlight the name of the results files in the Output Manager, and click on Save As.*

While we could look at the following issue within SeqLab, it is easier to deal with outside of the SeqLab environment.

Go back to your xterm.

You now have two alignment files, one produced by clustalw, and one produced by pileup. Both are in GCG format.

Lets look at them using the unix command **less**. For example purposes, I will write the commands using the filename **pileup.msf** for the pileup output, and **clustalw.msf** for the clustal output.

Go back to your xterm and type

**less pileup.msf**

Near the top of this file is some information about the conditions you used when you ran pileup. For example, what the gap penalties were. As is characteristic for GCG files, there are two dots .. signifying that data is to follow. It is the information after the two dots that is “seen” by GCG programs.

This is followed by a list of information about the sequences in the alignment. The rightmost column gives a weight value of 1.00. You can alter these weights with a text editor if you want to make one, or some sequences, more “important” when Pretty or PrettyBox is calculating a consensus sequence.

For example, if you wanted cd4\_macmu to *be* the consensus sequence, and have all the other sequences compared to it, then you could change the weight next to cd4\_macmu from 1.00 to, say, 4.00. If you did this, every residue in the cd4\_macmu sequence would get 4 “votes” for the consensus, while the other sequences, with weights of 1.00, would get just one vote each. Thus, in this alignment of 4 sequences, the residues of the cd4\_macmu sequence would outscore the rest and be used as the consensus sequence.

**Quit out of less (press q on the keyboard).**

We will edit the .msf file (produced by pileup) using a text editor called nedit.

On the command line, type

**nedit pileup.msf &**

**Now change the weight of cd4\_macmu to 4.00**

**Save and close this file.** Now return to the main SeqLab window.

**Highlight pileup.msf and go to the Functions menu,** and under the Multiple Comparison option, **click on Pretty.**

**Run the Pretty program and look at the results.**

Can you see the difference in the consensus sequence and how the other sequences have been represented?

Another way to edit sequence weights: If you just want to change the weight of a single sequence within a multiple sequence alignment, you can edit them directly in SeqLab.

**Highlight the rsf or msf file in Main List mode**

**Enter Editor mode**

**Double click on the sequence which you want to change the weight of.**

A window containing the information about that sequence appears. You can change the value in the weight box within this window.

### **What about Clustal and PrettyBox?**

The clustal output you have created is in GCG format, and thus can be used as input for Pretty or PrettyBox. There is something you have to do first though! Go back to your xterm window and, if your file was called clustal.msf, type:

```
less clustal.msf
```

Notice the weights! Clustalw calculates weights differently than Pileup.

Due to the way that Clustalw does the alignment, it calculates weights differently than Pileup. This means that the consensus sequence produced by default by Clustalw and Pileup may be different.

Please note that older versions of Clustal produced sequence weights below the value of 1.0. If you use older versions of clustal than those running on Molbiol, you may need to hand edit the weights if you want to use the results in programs like Pretty or PrettyBox.

## **19. General Comments**

Hopefully, at the end of this practical, you will have some knowledge of the type of programs available through GCG and how to access them via the command line and through SeqLab.

We have tried to pinpoint some of the common problems that people experience when they first start using GCG (and unix!) Feel free to experiment with as many of the GCG programs as you like and ask about anything that doesn't seem to work as you think it should!

We can always be reached by email at: **genmail@molbio.ox.ac.uk**

## Appendix 1

### Running processes in the background

When you are running a program, you can have it running in the foreground or the background.

If you run a process in the foreground, you can interact with it. If you run it in the background, you can interact with other processes at the same time.

By running SeqLab in the background, you can still interact with your xterm window, i.e. you can still type in it, carry out commands, etc. In the case of programs running in their own window (for example, SeqLab), running it in the background allows you access both to this program and your original xterm window.

The easiest way to put a process in the background is to type an **&** (ampersand) after the program name. For example:

**seqlab &**

If you forget the **&**, or you decide later that you want to run a program in the background, you can put it into the background after it is already running.

To do this, you:

- 1) suspend the program by typing **Ctrl-z** and then
- 2) type **bg** on the command line. This starts the program running again, but in the background. To bring a program back to the foreground, type **fg** on the command line.

## Appendix 2

### Creating a new sequence in editor mode (Not Recommended!!)

A number of the important options used in created or editing sequences are shown in Figure 3.

Make sure the Display option reads **Residue Coloring**. Click on the **Keyboard Option** Menu and **choose Insert**.

Keyboard Options:

- |            |  |
|------------|--|
| Insert     | Allows you to insert or delete characters to the left of the cursor position   |
| Check      | Allows you to re-enter a sequence overtop of the first sequence. If you type in a character that does not match the original sequence, the screen will flash, and a message appears in the bottom of the window telling you which character did not match. |
| Overstrike | Allows you to type over a sequence and replace characters entered previously.  |

**Go to the File menu, and choose New Sequence.**

You will be given options for what type of sequence you are about to enter. Choose one.

Place the cursor where you wish to begin entering the sequence, and then type in your sequence.

You can cut, copy or paste regions of your sequence by highlighting the region or sequence of interest and using the cut, copy and paste buttons (see Figure 3).

Note on cases: The programs in the GCG package do not differentiate between upper and lower case letters within sequences. However, some other programs do! It is safest to enter all sequences in upper case letters.

### Sequence information

To change the name of your sequence, or add information regarding its orientation, form, type, etc, click on the info button (see Figure 3) and fill in the appropriate boxes.

To protect your sequence against particular types of modifications, click on the Protect button (see Figure 3) and choose the options you think are appropriate.

### Saving your sequence

If the sequence you have added is part of a project, you will want to add it to the sequences in your project list as well as save the sequence.

To do this,

**go to the File menu, and click on Save As.**

By default, your sequence will be saved in Rich Sequence Format (rsf) and will be added to the Main List. RSF format means that any features or comments you have added to the sequence will not be lost.

## Appendix 3

### RSF file format

rsf (Rich Sequence Format) files contain one or more sequences that may or may not be related. In addition to the sequence data, each sequence can be annotated with descriptive sequence information such as:

- Creator/author of the sequence
- Sequence weight
- Creation date
- One-line description of the sequence
- Offset, or the number of leading gaps in a sequence that is part of an alignment or fragment assembly project
- Known sequence features

rsf files are more richly annotated than list files or msf files, which do not save sequence features information as part of the file

The features annotation allows you to graphically view and align sequences based on features as well as run programs on sequence regions selected by feature.

### The official documentation on rsf files is available at:

[http://www.molbiol.ox.ac.uk/documentation/gcg10docs/using\\_sequences.html#using3](http://www.molbiol.ox.ac.uk/documentation/gcg10docs/using_sequences.html#using3)

### Specifying RSF Files

Just like msf files, you have to designate which sequences you want to view from within an rsf file.

Single Sequence: To specify a single sequence within an rsf file, type the name of the rsf file and extension followed by the name of a sequence in curly brackets. For example

```
cd4pileup.rsf{cd4_human}
```

Multiple Sequences: To specify a subset of sequences or all sequences within an rsf file, type the name of the rsf file and extension followed by a file specification and/or asterisk (\*) wildcard in curly brackets. For example, `opsin.rsf{opsg*}` specifies all sequences in `opsin.rsf` beginning with “opsg”; `opsin.rsf{*human*}` specifies all sequences in `opsin.rsf` where “human” is part of the sequence name; and `opsin.rsf{*}` specifies every sequence in `opsin.rsf`.

### Features in RSF Files

For an example and explanation of the features information within an rsf file, see the web page:

[http://www.molbiol.ox.ac.uk/documentation/gcg10docs/figure/using\\_sequences\\_6.gif](http://www.molbiol.ox.ac.uk/documentation/gcg10docs/figure/using_sequences_6.gif)

The colours, shapes, and fill patterns depicted in SeqLab’s Editor are defined in a resource file called **feature.cols**. To customise these attributes, you need to copy `feature.cols` to your current directory and then edit it in the text editor of your choice. To copy `feature.cols` to your directory, type:

```
fetch feature.cols
```

This is very useful as you can define specific colours for particular features relevant to your own sequences.

## Appendix 4

### How to convert graphics images

Some bioinformatics programs produce graphical output in postscript format. Often you will want your results in some other graphical output format.

There is a useful, and easy to use set of programs that are part of a package called ImageMagick. The two programs you might find more useful are convert and display.

To convert files from postscript to any other format, you just need to type the

following:

```
convert myfile.ps myfile.xxx
```

where “myfile” is the name of your file. The program recognises .ps as postscript, and whatever you write instead of xxx as the image type you wish to convert to. For example,

```
convert myfile.ps myfile.jpeg    will convert your file to the jpeg format
```

```
convert myfile.ps myfile.tiff   will convert your file to the tiff format
```

The program display will display most image formats available except postscript (recall: use ghostview for postscript!). For example, to see a jpeg image, just type

```
display myfile.jpeg
```

The image should then appear in its own window. If you click anywhere on the image, a menu should appear. If you go to the File menu, and choose Save, you get a new window in which you can specify where in your account you wish to save the file. If you click on the Format button at the bottom of the page, you will see a list of the formats you can save your image as.

### Note on jpeg format

Jpeg files take up much less “space” than many other image formats. However, this means that if you have a very detailed image, you may lose some of the detail. So, in the case of jpegs, you are given a choice as to the level of quality you want. The range offered is 50% to 100%. The lower the quality, the less space the image takes. So, for an alignment, 50 to 75% may be fine, whereas for a photograph, you would have to decide what % quality you needed, and indeed, whether jpeg was the right format for you.

When you convert from postscript to jpeg format using the convert command, the jpeg file is automatically saved at 75% quality.