



Advanced GCG

Running GCG from the command line

The aim of this class is to run a typical GCG programs using the command line. Since GCG programs all work in the same way, you should be able to run almost any GCG program after this session. A rudimentary knowledge of UNIX is assumed. Refer to the ACHS-305 "Introduction to UNIX" handout.

Login to your account

GCG operates from the command line of your UNIX account. Login to your UNIX account. If your default is to run **Umenu**, choose "Go to UNIX"

Start GCG

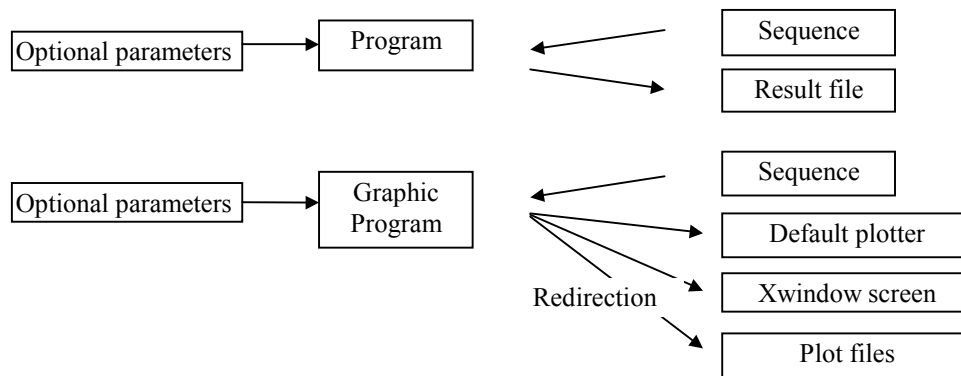
To start GCG type `gcg`

Running a typical program

To see a list of all the available GCG programs, type `genhelp`

To see a list of all available programs, grouped by function, type `genmanual`

All GCG programs work basically the same. The program will operate on a file, normally a sequence file, and write the results to another file. The results are not written to the screen. To see the output of a program, you need to view the output file, normally the UNIX command "more filename".



Importing a text sequence

To fetch a sequence to work on for this tutorial type
`fetch enigma`

To reformat a text file to GCG format type
`reformat enigma`

Hint: To see the content of the file type `more enigma`

Note that your original text file will be overwritten by the new sequence file. To retain your original text file, redirect the output of the reformat command to another file, for example "enigma.seq" by typing `reformat enigma -out=enigma.seq`

Hint: If your text file includes a heading or other text, separate the heading from the text with a ". ." symbol. You can insert the ". ." symbol using your favorite text editor, such as jove. All text after the symbol will be treated as part of the sequence. Spaces, returns and numbers in the sequence will be ignored.

Fetching a sequence from the Genbank database

To fetch the X01832 sequence from the database type

```
fetch X01832
```

The file X01832.gb_ro will be placed in your directory. (gb_ro refers to the rodent subdivision of GenBank where the sequence is located.)

Hint: Rename the file to a more sensible file name with the UNIX command :
`mv X01832.gb_ro mouse_IgM.seq`

If the GenBank file contains a translated peptide sequence, a second file called X01832.gp_ro will be placed in your directory. (gp is the GenPept database that contains translated protein sequences from GenBank.). *Not all GenBank sequences contain translated protein sequences and entries in GenPept.*

Mapping restriction sites

To map restriction sites in the sequence “enigma” type

```
map enigma
```

The program will prompt for the range of the sequence to analyze.

```
Begin (* 1 *) ?          Press <return> to accept the default of 1, or enter your own value.
```

```
End (* 631 *) ?         Press <return> to accept the default of 631, or enter your own value.
```

```
Enzyme (* * *) :       <return> will select all enzymes  
?                       provide information on the available enzymes  
EcoRI                   will select a particular enzyme Eco R1 (Note the format!)
```

```
n                       To select no translation type
```

The program will prompt for the name of the output file. Press <return> without typing a name, to use the default file name “enigma.map.” When the program is finished, you will return to your prompt.

Type `more enigma.map` to see the results of the analysis. (To print this file to your default printer, type `lpr enigma.map`.)

The real power of the GCG programs are hidden in the command line options. To see which options are available type

```
genhelp map            To see the available optional parameters type  
opt                   Hint: typing genhelp map opt<return> is faster.
```

If you want to limit your search to restriction enzymes that have 6-base recognition sites, type

```
map enigma -six
```

Many more options can be set, e.g.

- `-miss=1` to introduce one base mismatches, useful for point mutations.
- `-app` provides information on the restriction enzymes used, including their cutting sites and suppliers.
- `-d` accepts all the defaults of the program.
- `-out=` redirect the output to another file name.

Example: `map enigma -six -once -out=sixbase.map -d`

Other programs that map restriction enzymes:

- `mapsort` arranges restriction fragments on order of size.
- `mapplot` prints a graphic representation of the cutting sites.

Searching Databases

Searching databases for text

To search the databases for a particular author or enzyme name type
lookup

Choose a library or accept the default (All libraries) by pressing
<return>

The following screen will be displayed:

```
Complete the query form below:

      All text:
      Definition:
      Author:
      Keyword:
      Sequence name:
      Accession number:
      Organism:
      Reference:
      Title:
      Feature:
      On or after (dd-mmm-yy):
      Shortest sequence length:

      On or before (dd-mmm-yy):
      Longest sequence length:

      Inter-field operator:  AND          Form of output list:  Whole Entries

Press <Ctrl>D to continue.
```

Use the following guidelines when you write LookUp queries:

All queries are case insensitive.

- &** specifies AND. A & B means find all entries that contain both A and B
- |** specifies OR. A | B means find all entries that contain either A or B.
- !** specifies BUT-NOT. A ! B means find all entries that contain A but not B.
- ()** Parentheses group expressions to evaluate first. For example, Smithies & (Slightom | Blechl) searches for sequences containing Slightom or Blechl. Then, out of those sequences it searches for those which also contain Smithies.
- ?** any single character. The value s?ith includes Smith, Slith, Sjith, etc., but not Sith.
- *** anything or nothing. The value *smith* includes Smith, Hocsmith, Smithies, etc.
- #** literal query Adding a pound sign (#) to the value, for instance pseudo#. will only find those entries where the word pseudo occurs by itself.

Use single words, or words connected with “&,” “|” or “!” to fill in the blanks and press <Ctrl>d

An example of a list file called lookup.list:

```
LOOKUP in: swissprot,pir,embl,genbank,em_tags,gb_tags of: "[SQ-ALL: histone* &
h1*]"

492 entries  January 19, 1996 15:22 ..

SWISSPROT:B4_XENLA ! ID: 730d0001
! DE B4 PROTEIN (HISTONE H1-LIKE PROTEIN) .
SWISSPROT:H101_CHICK ! ID: 303a0001
! DE HISTONE H1.01.
SWISSPROT:H103_CHICK ! ID: 313a0001
! DE HISTONE H1.03.
SWISSPROT:H10_HUMAN ! ID: 333a0001
! DE HISTONE H1' (H1.0) . //etc.
```

To fetch a sequence e.g. B4_XENLA type

```
fetch SWISSPROT:B4_XENLA
```

To fetch all the sequences in the lookup.list file, type

```
fetch @lookup.list
```

Searching databases for sequence similarity

To run BLAST type

```
blast enigma
```

Select option “8) nr” and accept all the other defaults.

To see the results of our search type **more enigma.blastn**

FASTA is more sensitive, but slower program and may take several hours to complete. It is therefore best to submit a search as a batch job.

To submit a fasta search as a batch job type

```
fasta enigma -bat
```

Accept the defaults.

The job will be automatically submitted and the file containing the results will appear in your directory.

Note: BLAST connects directly to NCBI and can search the current updates of the database. All the other GCG programs use our local database. It may therefore be possible to find sequences with BLAST that are not available with Fetch or Fasta. Check the GCG banner for the current status of the databases.

Note: The expressed sequence tags division of GenBank contains expressed, but unannotated sequences. To search GenBank including the est sequences, specify **genbankplus:*** or **gbp:***.

The following commands are available:

| Editor | Command Line |
|---|--|
| <p>G, A, T, . . . - insert a sequence character <Delete> - delete a sequence character (<Backspace> on some systems) /TAACG<Return> - find the next occurrence of TAACG (last pattern entered is the default) <Ctrl>H - move to start of the sequence <Ctrl>E - move to end of the sequence [n]<Right-arrow> - go ahead n characters (It may be the arrow keys on the numerical keypad.) [n]<Left-arrow> - go back n characters <Up-arrow> - go up to check sequence <Down-arrow> - go down to original sequence 'markcharacter - go to marked position 37<Return> - go to position 37 (any positive integer) < - go back 50 characters > - go ahead 50 characters <Ctrl>R - redraw the screen <Ctrl>D - enter command mode</p> | <p>[] indicates optional parameters. n position of the nucleotide s start position f finish position</p> <p>[n] EDit seqname -get a new sequence file to edit Include [seqname] - insert another sequence [at position n] prompts for range and strand) s,f Delete - delete a range of bases [s] Check [/Blind] - check (or proofread) a range of bases [beginning at s] 37 - go to base 37 REDraw - redraw the screen [n] COMment comment - insert a comment [at position n] [n] COMment - enter comment editing mode [at position n] If you do not supply a position number, the default will be the last position of the cursor in the sequence. [n] HEAding - edit documentary heading [at line n] OVERstrike - enter overstrike mode INSert - enter insert mode [n] Mark markcharacter - mark the sequence [at position n] PERFect - require finds to be perfect matches PROtein - set sequence type to PROTEIN NUCleotide - set sequence type to NUCLEOTIDE [s,f] Write [seqname] - SAVE AS, write [a part of] the sequence to a file Help - show commands in screen and command modes [s,f] EXit [seqname] - SAVE [a part of] the sequence AND QUIT Quit - QUIT the editor WITHOUT SAVING the sequence</p> |

The command mode allows you to save or import sequences and edit the header. To enter the command mode type **<Ctrl>D**

To move back to the sequence line type **<return>**

To edit the header, type at the command mode type **head<return>**

Move around with the arrow keys and make any changes you want. To return to the command mode type **<control>D**

To export the portion of the sequence between position 550 and 600, type
550,600 write fragment.seq<return> (only the “w” of “write” is really necessary)
 The file “fragment . seq” will be placed in your directory.

To include (or insert) a sequence at a specific position in the sequence go to screen mode and move the cursor to position 572. Type **<control>d** to move to the command mode. Type
include fragment.seq<return> (Only the “i” of “include” is necessary.)
 If you did not want to move the cursor type **572 i fragment . seq**

You will be prompted for the range and the strand of the sequence you would like to include.

Comments keep track of a modifications and can be added by typing at the command line **comment<return>**
 (Only the “c” of “comment” is really necessary.) The program will default to the last position of the cursor.

To **save** (or write) the current sequence, type at the command line **w<return>**

To **save as** a new file name type **w filename<return>**

To **exit** the editor **and save** the sequence, type **ex** or **exit<return>**

To **quit** the editor type at the command line **q** or **quit<return>** This will **NOT save** the sequence.

Working with multiple sequences

Fetch five sequences as follows:

```
fetch gb:m60331 -out=m60331.seq
fetch gb:x71334 -out=x71334.seq
fetch gb:x71335 -out=x71335.seq
fetch gb:x71336 -out=x71336.seq
fetch gb:x71337 -out=x71337.seq
```

Multi-sequence alignment

The program “pileup” is a powerful variation of gap. PileUp creates a multiple sequence alignment from a group of related sequences using progressive, pairwise alignments. It can also plot a tree showing the clustering relationships used to create the alignment.

A simple multi-sequence alignment

You can do an alignment directly by typing

```
pileup *.seq
```

The result is written to an “msf” (multiple sequence format) file (For more information on the msf file format see later in this document.)

The advantage of this procedure is that is simple. However, it does not allow you to specify parts of the sequences or the order in which they appear. All sequences in your directory that satisfies the wild card will also be included. This can become a problem in a big project. A better way to specify the sequences is through the use of a names file. In its simplest form, a names file is a list of file names.

Create a list file manually

You can use the text editor of your choice to create a names file. In this exercise we use jove (See the “Jove, short descriptions” handout for more information.).

To create a names file called “monkey.fil” type

```
jove monkey.fil
```

Enter the names, one per line as follows

```
m60331.seq begin:470 end:1250
x71334.seq
x71335.seq
x71336.seq
x71337.seq
```

When you are finished save the file by typing **<control>x s**
Exit the program with **<control>x<control>c**

Running pileup with a list file

You can run pileup now by typing

```
pileup @monkey.fil
```

Choose all the defaults.

Hint: If you already have the sequences in your directory, you can quickly create a list file containing all the sequences in your directory. If all the sequence files end in .seq, type:

```
ls *.seq > myseqs.list.
```

Note: On some systems, such as PCs, pressing **<control>s** by accident will freeze the screen. Restore the screen with **<control>q**

The **@** symbol specifies that sample.fil is a names list. If the **@** symbol is omitted, the program will regard sample.fil as a sequence file and produce the error message “no sequence in monkey.fil”

The resulting file should look as follows:

```

PileUp of: @monkey.fil

Symbol comparison table: GenRunData:pileupdna.cmp  CompCheck: 6876

          GapWeight: 5.000
          GapLengthWeight: .300

monkey.msf  MSF: 782  Type: N  February 10, 1995 14:29  Check: 5893 ..

Name: x71334      Len:   782  Check: 2890  Weight:  1.00
Name: x71336      Len:   782  Check: 1594  Weight:  1.00
Name: x71337      Len:   782  Check: 5493  Weight:  1.00
Name: x71335      Len:   782  Check: 4085  Weight:  1.00
Name: m60331      Len:   782  Check: 1831  Weight:  1.00

//

          1                                     50
x71334  ..... ..GGGCCAC TGCAGCCTCA  GCCCAGGAGC  CACCAGATCT
x71336  ..... ..GGGCCAC TGCAGCCTCA  GCCCAGGAGC  CACCAGATCT
x71337  ..... ..GGGCCAC TGCAGCCTCA  GCCCAGGAGC  CACCAGATCT
x71335  TCCCCTACCC CAGGGCCCAC TGCAGCCTCA  GCCCAGGAGC  CACCGGATCT
m60331  .GATGCAGGC CACCTGGCAT  GGTGTGTGAG  GTCCAGCCCC  TTTGCCCTCA

          51                                     100
x71334  CCCAGCACCA TGGTCCGATA  CCGCGTGAGG  AGCCCAGGCG  AACCCCTCGCA
x71336  CCCAGCACCA TGGTCCGATG  CCGCGTGAGG  AGCCCAGGCG  AACGCTCGCA
x71337  .....A TGGTCCGATA  CTGTGTGAGG  AGCCTGAGCG  AACGCTCGCA
x71335  CCCAACACTA TGGTCCGATA  CCACGTGAGG  AGCCCAAGCG  AACGCCACCA
m60331  CAATGACCAA  CGGCCCCCTG  GCATCTATAA  CAGGCCGAG  AGCTGGCCCC

```

Maintaining the order or sequences

To maintain the order in which the sequences appear in the .msf file the same as in the list file include the -nosort option in the command line by typing

```
pileup @monkey.fil -nosort
```

The program functions exactly as before, i.e. the sequences are aligned pairwise starting at the closest related sequences, **only the order in which the sequences are displayed is changed.**

Programs that produce list files as output

Instead of typing in a list file, you can use the output of a number of programs as list files. If the complete database address of a sequence is included, GCG will automatically look that sequence up in the database when it is needed by the program. Therefore, the sequence does not have to be in your own directory.

| Program name | Optional parameter |
|---------------|--------------------|
| Assemble | -LISTfile |
| Corrupt | -LISTfile |
| FastA | -NOALIGN |
| FindPatterns | -NAMES |
| FromEMBL | -LISTfile |
| FromFastA | -LISTfile |
| FromGenBank | -LISTfile |
| FromIG | -LISTfile |
| FromPIR | -LISTfile |
| Lineup | |
| Motifs | -NAMES |
| Names | |
| Pretty | -UGLy |
| ProfileSearch | |
| Reformat | -LISTfile |
| Sample | -LISTfile |
| Simplify | -LISTfile |
| StringSearch | |
| Lookup | |
| TFastA | -noalign |
| Translate | -LISTfile |
| Wordsearch | |

Producing a list file with Lookup

To search the database with lookup type **lookup<return>**

Select the database

Fill in the search form. In this case “protamine” was typed in the **All text** field.

Write te results to a file, “lookup.list”

```
LOOKUP in: genbank of: "[SQ-ALL: protamine*]"
147 entries January 30, 1996 10:36 ..
GB_BA:ECOTGY1 ! ID: c51e0005
! DEFINITION E.coli tyrT locus containing two Tyr-tRNA-1 genes.
GB_IN:AGPROT ! ID: 2d640005
! DEFINITION Boll Weevil mRNA for protamine.
GB_IN:CFDC2G ! ID: 536e0005
! DEFINITION C.fasciculata cdc2 gene homologue.
GB_IN:DMMST35BA ! ID: 87750005
! DEFINITION D.melanogaster mRNA for protamine (mst35Ba).
GB_IN:DMMST35BB ! ID: 88750005
! DEFINITION D.melanogaster mRNA for protamine (mst35Bb).
GB_IN:DMMSTBAGE ! ID: 96750005
! DEFINITION D.melanogaster gene for protamine (mst35Bb).
```

Note: Since all these sequences have a complete database address, they do not have to be in your local directory. The program will automatically find them in the database.

You can now run pileup using this listfile as follows:

```
pileup @lookup.fil
```

Producing a list file with FastA

To produce a list file using fasta type

```
fasta -noalign
```

Respond with the sequence name **x71334.seq**

You can now edit the **x71334.fasta** output file and use it as a list file for pileup.

Local data files

Many programs use data files to provide the information it needs to execute. For example map uses the file “enzyme.dat” that contains information on restriction enzyme recognition sites. The program will first look in the local directory for the file. If the file is not there, it will look in the default GCG directory. A datafile may also be specified with the **-dat=filename** option. Alternative datafiles are located in the **/gcg/gcgcore/data/moredata** directory. Data files may be copied to your own directory with the fetch command and modified to suit your own purposes. For more about the data files, look in **genhelp** under the program name at the “Local Data files” option.

| | Data file | Program where used | Function |
|---|--------------------|--|--|
| | enzyme.dat | Map, Mapplot, Mapsort | List of available restriction enzymes |
| | enz_sources.txt | Not used by any program | A list of commercial suppliers of various enzymes |
| | enz_refs.txt | Not used by any program | List of citations of restriction enzymes |
| | proenzyme.dat | PeptideMap, Mapsort, Mapplot | Peptide cleaving enzymes and reagents |
| | proenzall.dat | PepditeSort | A more complete list than proenzyme.dat |
| * | all_proteases.dat | PeptideMap, Mapsort, Mapplot, PepidteSort | A more recent and complete set of proteases. |
| * | tfsites.dat | Map, Mapplot, Mapsort, Findptterns | Eukaryotic transcription factor database. |
| * | ecohigh.cod | BackTranslate, CodonPreference, Frames | Codon frequency tables |
| * | human_high.cod | | |
| * | maize_high.cod | | |
| * | yeast_high.cod | | |
| * | celegans_high.cod | | |
| * | celegans_lox.cod | | |
| * | translate.txt | BackTranslate, CodonFrequency, CodonPreference, Diverge, Frames, Map, Mapplot, Mapsort, Outline, PepData, Publish, Reformat, Translate | Used to define relationships between codons and amino acids. |
| * | tranciliate.txt | | |
| * | transmitodros.txt | | |
| * | transmitomam.txt | | |
| * | transmitoyeast.txt | | |
| | prosite.patterns | Motifs | Dictionary of protein sites and patterns. Use -fre option for frequent sites. |
| | prosite.seqcat | Not used by any program | An alphabetical listing of protein sites and patterns. |
| | comparedna.cmp | Compare | DNA comparison tables. |
| | nswgapdna.cmp | Gap | These tables are all identical. |

| | | | |
|---|---|--|---|
| | pileupdna.cmp plotsimdna.cmp prettydna.cmp repeatdna.cmp | Pileup PlotSimilarity Pretty Repeat | |
| | swgapdna.cmp | Bestfit | DNA comparison table with lower scores (0.9), for ambiguous matches |
| | fastadna.cmp | FastA | DNA comparison table with lower scores for ambiguous matches |
| | segdna.cmp | ProfileMake, Segments | DNA comparison table with weighted, lower scores for ambiguous matches |
| * | seggapdna.cmp | Segments, ProfileGap, ProfileSegments | Alternative matrix. |
| | stemloop.cmp | StemLoop | Scoring matrix for stemloop. |
| * | randomdna.cmp | For any DNA comparison program. | Ambiguous codons rated at 0.3 to extend local alignments in programs such as BestFit. |
| | swgappep.cmp comparepep.cmp blosum50.cmp nwsgappep.cmp swgappep.cmp blosum62.cmp plotsimpep.cmp prettypep.cmp profilepep.cmp repeatpep.cmp segpep.cmp fastapep.cmp | BestFit Compare FastA Gap GapShow PileUp PlotSimilarity Pretty ProfileMake Repeat Segments TFastA | Peptide scoring matrices. |
| * | blosum100.cmp | For all peptide comparisons | Alternative comparison tables |
| * | blosum30.cmp blosum35.cmp blosum40.cmp blosum45.cmp blosum55.cmp blosum60.cmp blosum65.cmp blosum70.cmp blosum75.cmp blosum80.cmp blosum85.cmp blosum90.cmp pam120.cmp pam250.cmp structgapppep.cmp | | |
| * | aminoacid.dat | PeptideSort | Amino acid residue properties |
| * | extinctcoef.dat | | Extinction coefficients |
| * | isoelectric.dat | | pK values |
| | isoelectric.dat | IsoElectric | pK values |
| * | pepplot.dat | PepPlot | Values for secondary structure predictions |
| * | ges.dat | | Values for transbilayer helices |
| * | garnier.dat | | Garnier predictions |
| | helicalwheel.dat | HelicalWheel | Values for helical wheel predictions. |

* Used only when specified as an alternative data file.

Fetching and editing the DNA comparison table

You can fetch the DNA comparison table to your home directory and modify the file with a text editor. It is also good practice to rename it, so you can use the default table when needed. To fetch and redirect the file type

```
fetch pileupdna.cmp -out=fixdna.cmp
```

Fetching and editing the Amino acid comparison table

To modify the amino acid comparison table, first copy the table to your working directory. To do this type

```
fetch pileuppep.cmp -out=fixpep.cmp
```

The file called “fixpep.cmp” will be copied to your directory. You can modify this file with you favorite text editor. If you have a file called pileuppep.cmp in your working directory, GCG will read that one first.

To use your own comparison table, called “fixpep.cmp” type

```
pileup @yourlisfile.fil -dat=fixpep.cmp
```

Hint: To get more information on data files for any particular command, look under the genhelp, “local data files” option.

Using the Prosite database

The prosite database contains protein sequence motifs, such as phosphorylation and glycosylation sites. To search for these sites on your sequence type

```
motifs filename
```

By default the program will *not* search for frequently occurring sites. To map frequent sites use the **-fre** option.

SEQLAB
MOTIFS are under Functions, Protein Analysis >.
The Patterns... dialog box allows you add or delete patterns, or use selected patterns for your search.

Using the transcription factor database

First, load the transcription factor database to your directory with

```
fetch tfsites.dat
```

Now run map with the “-dat” option

```
map sample.seq -dat=tfsites.dat
```

The tfsites.dat file can also be used by mapplot and mapsort.

SEQLAB
Use Extensions, xterm to open a window for the fetch command
The Functions, mapping, map Enzymes... dialog box allows you add or delete patterns, or use selected patterns for your search. Select tfsites.dat in the Enzyme Data File... dialog box.

Using list files as databases

For programs that are slow, or may produce an excessive number of hits when you search the complete database, it is useful to refine a search. List files are powerful tools to use the output from one program as the input for another. This is particularly useful to narrow searches. For example create a list file with lookup, e.g. select the genbank database and search for protamine in the “All text” field. You can now use the lookup.list output file as a database for fasta. (For more information on creating listfiles, seen ACHS-307.) To search with the query sequence, “dasy.seq” type the following:

```
fasta dasy.seq
```

SEQLAB
Select the list file in the Main List mode and execute the program

reply to the prompt “Search for query in what sequences (* GenEMBL:* *) ?” with
@lookup.list

The program will now execute a fasta search against all the sequences in the list file. This procedure will work for any program that produce a list file and any other program that produces the prompt **Enter the name of the sequence(s)**. Note the plural in the prompt.

Using *.init files to set defaults

In some cases you may want to change the defaults that a program provides. All programs read their defaults from a file called “programname.init” If you provide an identical file name in your local directory, the variables in that file will be used.

The following is an example of an init file for the program prime :

```
These are my parameters for the prime program..
```

```
-DNAconcentration=100  
-SALTconcentration=25  
-TMMINPRimer=65  
-TMMAXPRI=78  
-GCMAXPRODUCT=60  
  
-FONT=3
```

SEQLAB
Settings can be saved and restored by using the Save Settings or Restore in the program dialog box.

You may also use the optional parameter **-init=filename.init** to specify and initialization file.

If you want to search for primers in the enigma.seq sequence, type **prime enigma.seq**
Accept all the defaults, or change them as you want.

The results will be plotted and a *.prime file (**enigma.prime**) will be written.

To specify your own init file, myprime.init, type **prime enigma.seq -init=myprime.init**

Using aliases

An alias is used in UNIX to shorten frequently typed commands. Aliases are set in the file called

```
.kshrc
```

Make sure when you use aliases that they do not conflict with commands you use in a different context. The following is an example of some of the aliases I find useful

```
alias jo="jove"
alias clr="clear"
alias bigfont="xterm -font'-misc-fixed-medium-*-*-*-200-*-*-*-*-*'"
```

Graphics

The graphic output of all GCG programs are printed to the default printer. This will happen automatically and should not require input from the user.

Warning. Attempting to print graphics when a default printer is not set, may hang the program.

There are several optional parameters that control the appearance of the output. These are described in the “Short descriptions of GCG commands” handout. Two very useful options are:

| | |
|----------------------|---|
| -pen=1 | will produce monochrome graphics. |
| -plo=filename | will redirect the output to a file instead of the printer |

All network printers support PostScript and the file produced in this way is a PostScript file. This file can be copied to a printer at a later date to produce the output.

Note: It may be necessary to copy files to the printer as binary files. In DOS the binary switch is “/b” and the command will be
c:\copy filename/b lpt1

Transferring graphics to other programs

Encapsulated PostScript (EPS)

Most programs will import encapsulated postscript (EPS) files. To set GCG to print to an EPS file, type

```
postscript
```

Select **EPSF** or **CEPSF** from the list .

Note: The **-plo=filename.eps** option will override the file setting.

When the program responds with “**To what port is your EPSF connected (* achs_12 *)**” type in the name of the file where you want the data sent.

You can now copy the file to your local micro computer (We recommend “Fetch” on the Mac or “Uva Lan Workplace, RapidFiler” on the PC for this. You may use FTP as well, but remember to set file type to binary.)

Most programs, such as MSword or FreeHand, may not display the actual graphics, but a hatched box instead (This is a function of MSword and Freehand, *not* GCG). If this is the case the graphic will still print properly, but it will not be possible to modify it.

Hewlett Packard Graphics Language (HPGL)

HPGL was designed to drive pen plotters, consequently the output is constructed out of a series of circles and lines. The quality of the text is relatively poor, but this is easy to modify in most graphic packages.

To produce an HPGL file type

```
HPGL
```

```
select HP7470
```

When you run the program that produces the graphics, use the

```
-plo=filename option, e.g.
```

```
maplot -plo=maplot.hpg
```

This file can now be transferred to your local microcomputer and imported into your software.

SEQLAB
The procedure is exactly the same as for EPSF files, except in the Options, Graphic Devices..., Create dialog box choose Language : HPGL, Device: HP7470

Graphics on the Macintosh

A program called GCGFigure is available on the word wide web at
<http://www.gcg.com/pub/mac/>

GCGFigure is a Macintosh program for displaying graphic output from GCG programs. It reads files in GCG's Figure format, displays them, prints them, and converts them to PICT format for use by other Macintosh programs.

SEQLAB

After you have run a graphics program, in the graphics window, choose Print... Select the output device you have specified and in the Port or File: box, fill in the output file name. The file will be placed in your default directory

Alternative sources of Information

World Wide Web Resources

The Uva Molecular Biology provides information on local databases and programs as well as listings of useful international web sites. The address is:

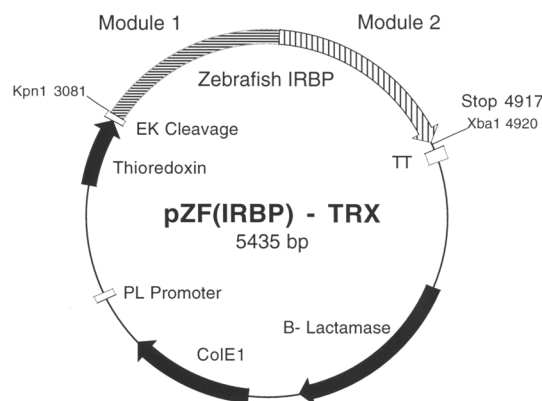
<http://www.med.virginia.edu/achs/molbio/MolBio.html>

Local Programs

We maintain a large number of sequence analysis programs in the `/seqprg/slib/bin/` directory and help files are available in `/seqprg/slib/src`. All local programs will all run after you have typed `gcg`.

Macintosh and PC programs

Macplasmmap and CloneMap are plasmid drawing programs is available in the Data Acquisition laboratory.



Protein structure information

An E-mail service is available where you can send your protein sequences and it will return secondary structure predictions. These predictions are based on known, homologous structures and can be up to 65% accurate if a related structure is known. This is the best accuracy of any method available.

Send questions to Predict-Help@embl-heidelberg.de

Send sequences to PredictProtein@embl-heidelberg.de