EMBOSS course

http://ebiokit/tutorials/EMBOSS_course/So,whatisEMBOSS02.html

So, what is EMBOSS?

EMBOSS is a free Open Source software analysis package specially developed for the needs of the molecular biology (e.g. EMBnet) user community. The software automatically copes with data in a variety of formats and even allows transparent retrieval of sequence data from the web. Also, as extensive libraries are provided with the package, it is a platform to allow other scientists to develop and release software in true open source spirit. EMBOSS also integrates a range of currently available packages and tools for sequence analysis into a seamless whole. EMBOSS breaks the historical trend towards commercial software packages.

The **EMBOSS** suite:

- Provides a comprehensive set of sequence analysis programs (approximately 150)
- Integrates other publicly available packages
- Encourages the use of EMBOSS in sequence analysis training.
- Encourages developers elsewhere to use the EMBOSS libraries.
- Supports all common Unix platforms including Linux, Digital Unix, Irix and Solaris.
- Within EMBOSS you will find over 150 programs (applications).

These are just some of the areas covered:

- Sequence alignment
- Rapid database searching with sequence patterns
- Protein motif identification, including domain analysis
- EST analysis
- Nucleotide sequence pattern analysis, for example to identify CpG islands.
- Simple and species-specific repeat identification
- Codon usage analysis for small genomes
- Rapid identification of sequence patterns in large scale sequence sets.
- Presentation tools for publication
- And much more.

More information about EMBOSS can be found here:
http://emboss.sourceforge.net/

Exercise: wossname

Here we will introduce you to the EMBOSS utility **wossname** will produce a list of all the various EMBOSS applications. All programs in the EMBOSS suite have help information click on the (read the manual) link!

Exercise:
On the commandline of the terminal type in **wossname** followed by PROTEIN

```
127:~ebiokit$ wossname PROTEIN
Finds programs by keywords in their one-line documentation
SEARCH FOR 'PROTEIN'
antigenic         Finds antigenic sites in proteins
backtranambig     Back translate a protein sequence to ambiguous codons
backtranseq       Back translate a protein sequence
charge            Protein charge plot
checktrans        Reports STOP codons and ORF statistics of a protein
compseq           Count composition of dimer/trimer/etc words in a sequence
digest            Protein proteolytic enzyme or reagent cleavage digest
emowse            Protein identification by mass spectrometry
epestfind         Finds PEST motifs as potential proteolytic cleavage sites
fproml            Protein phylogeny by maximum likelihood
fpromlk           Protein phylogeny by maximum likelihood
fprotdist         Protein distance algorithm
fprotpars         Protein parsimony algorithm
freak             Residue/base frequency table or plot
fuzzpro           Protein pattern search
fuzztran          Protein pattern search after translation
garnier           Predicts protein secondary structure
helixturnhelix    Report nucleic acid binding motifs
hmoment           Hydrophobic moment calculation
iep               Calculates the isoelectric point of a protein
makeprotseq       Creates random protein sequences
msbar             Mutate sequence beyond all recognition
mwcontam          Shows molwts that match across a set of files
mwfilter          Filter noisy molwts from mass spec output
octanol           Displays protein hydropathy
oddcomp           Find protein sequence regions with a biased composition
patmatdb          Search a protein sequence with a motif
patmatmotifs      Search a PROSITE motif database with a protein sequence
pepcoil           Predicts coiled coil regions
pepinfo           Plots simple amino acid properties in parallel
pepnet            Displays proteins as a helical net
pepstats          Protein statistics
pepwheel          Shows protein sequences as helices
pepwindow         Displays protein hydropathy
pepwindowall      Displays protein hydropathy of a set of sequences
preg              Regular expression search of a protein sequence
profit            Scan a sequence or database with a matrix or profile
prophecy          Creates matrices/profiles from multiple alignments
prophet           Gapped alignment for profiles
pscan             Scans proteins using PRINTS
psiphi            Phi and psi torsion angles from protein coordinates
```

```
shuffleseq      Shuffles a set of sequences maintaining composition
sigcleave       Reports protein signal cleavage sites
tcode           Fickett TESTCODE statistic to identify protein-coding DNA
tmap            Displays membrane spanning regions
tranalign       Align nucleic coding regions given the aligned proteins
```

This set of commands will cause **wossname** to write out the list of programs that are associated with proteins.

To produce a list of all the current EMBOSS programs, start up **wossname** again but instead of specifying a keyword, just type **wossname**. A list of programs will scroll onto your screen, divided up into groups according to their functions. Scroll up and down to see them all.

To read the manual for each programme listed type, **tfm name_of_programme**

```
127:~ebiokit$ tfm pepinfo
```
This will give you information on the function of the programme and how to use it.

We'll see some more later. Let's move on to some sequence analysis...

Working with sequences, seqret

Seqret reads in a sequence, and writes it out, in the format that you need. It is probably the most commonly used EMBOSS program. You do not need any format converter, use **seqret**!

Exercise: seqret

Download the file L0770.embl to your directory and convert it to a fasta format file using the EMBOSS command **seqret**. Inspect the downloaded file using a text editor.

It will look like this:

```
ID   L07770; SV 1; linear; mRNA; STD; VRT; 1684 BP.
XX
AC   L07770;
XX
DT   12-DEC-1992 (Rel. 34, Created)
DT   17-APR-2005 (Rel. 83, Last updated, Version 8)
XX
DE   Xenopus laevis rhodopsin mRNA, complete cds.
XX
KW   G protein-coupled receptor; phototransduction protein; retinal protein;
KW   rhodopsin; transmembrane protein.
XX
OS   Xenopus laevis (African clawed frog)
OC   Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Amphibia;
OC   Batrachia; Anura; Mesobatrachia; Pipoidea; Pipidae; Xenopodinae; Xenopus;
OC   Xenopus.
XX
RN   [1]
RP   1-1684
RA   Knox B.E., Scalzetti L.C., Batni S., Wang J.Q.;
RT   "Molecular cloning of the abundant rhodopsin and transducin from Xenopus
RT   laevis";
RL   Unpublished.
XX
RN   [2]
RP   1-1684
RX   DOI; 10.1074/jbc.271.6.3179.
RX   PUBMED; 8621718.
RA   Batni S., Scalzetti L., Moody S.A., Knox B.E.;
```

```
RT   "Characterization of the Xenopus rhodopsin gene";
RL   J. Biol. Chem. 271(6):3179-3186(1996).
XX
SQ   Sequence 1684 BP; 426 A; 431 C; 339 G; 488 T; 0 other;
     ggtagaacag cttcagttgg gatcacaggc ttctagggat cctttgggca aaaaagaaac        60
     acagaaggca ttctttctat acaagaaagg actttataga gctgctacca tgaacggaac       120
     agaaggtcca aatttttatg tccccatgtc caacaaaact ggggtggtac gaagcccatt       180
     cgattaccct cagtattact tagcagagcc atggcaatat tcagcactgg ctgcttacat       240
     gttcctgctc atcctgcttg ggttaccaat caacttcatg accttgtttg ttaccatcca       300
     gcacaagaaa ctcagaacac ccctaaacta catcctgctg aacctggtat ttgccaatca       360
     cttcatggtc ctgtgtgggt tcacggtgac aatgtacacc tcaatgcacg gctacttcat       420
     ctttggccaa actggttgct acattgaagg cttctttgct acacttggtg gtgaagtggc       480
     cctctggtca ctggtagtat tggccgttga aagatatatg gtggtctgca agcccatggc       540
     caacttccga ttcggggaga accatgctat tatgggtgta gccttcacat ggatcatggc       600
     tttgtcttgt gctgctcctc ctctcttcgg atggtccaga tacatcccag agggaatgca       660
     atgctcatgc ggagtagact actacacact gaagcctgag gtcaacaatg aatcctttgt       720
     tatctacatg ttcattgtcc acttcaccat tcccctgatt gtcatcttct tctgctatgg       780
     tcgcctgctc tgcactgtca aagaggctgc agcccagcaa caggaatctg ctaccaccca       840
     gaaggctgag aaagaggtca ccagaatggt tgttatcatg gtcgttttct tcctgatctg       900
     ttgggtgccc tatgcctatg tggcattcta catcttcacc caccagggct ctaactttgg       960
     cccagtcttc atgaccgtcc cagctttctt tgccaagagc tctgctatct acaatcctgt      1020
     catctacatt gtcttgaaca aacagttccg taactgcttg atcaccaccc tgtgctgtgg      1080
     aaagaatcca ttcggtgatg aagatggctc ctctgcagcc acctccaaga cagaagcttc      1140
     ttctgtctct tccagccagg tgtctcctgc ataagagctt caccagggct gtctcagggt      1200
     ccgctgcctc acacaattcc catcacttaa gccctgtcta cttgttgcga aggcaaagaa      1260
     ttccacagtt ttaatattta cccccattct gcccaacctt ggacactgta agagctgacc      1320
     ccattactgc tgggaaggcc caagctttgt tgcattctga tgtgatcctt tcagcagaaa      1380
     atgggtggat tcaatgaatt tcaccaaggc tgtacataac aataacatta gtctgaaggc      1440
     acctcccacc cagagaatgc aacacttatt tatctctgtc ttttcttgac atattgatgc      1500
     tgcttctatt catggtcact aacaaaaagt cccattttac aatgcaactg aaagtaatgt      1560
     atttttgtaa tataataaca tatttcatgc aatctcctct gcttattggc aaggtctgat      1620
     atagtgagga tagacagcca gacccttgc attaaaatcc tgtattaaaa atttctttgc       1680
     aagt                                                                  1684
//
```

On the commandline type the following:

```
$ seqret L07770.embl -osformat fasta
Reads and writes (returns) sequences
output sequence(s) [l07770.fasta]:
```

Look at the resulting file.

You should get the same sequence l07770.fasta:
Look at the result:

```
>L07770   L07770.1   Xenopus   laevis   rhodopsin   mRNA,   complete   cds.
ggtagaacagcttcagttgggatcacaggcttctagggatcctttgggcaaaaaagaaac
acagaaggcattctttctatacaagaaaggactttatagagctgctaccatgaacggaac
agaaggtccaaatttttatgtccccatgtccaacaaaactggggtggtacgaagcccatt
cgattaccctcagtattacttagcagagccatggcaatattcagcactggctgcttacat
gttcctgctcatcctgcttgggttaccaatcaacttcatgaccttgtttgttaccatcca
gcacaagaaactcagaacacccctaaactacatcctgctgaacctggtatttgccaatca
cttcatggtcctgtgtgggttcacggtgacaatgtacacctcaatgcacggctacttcat
ctttggccaaactggttgctacattgaaggcttctttgctacacttggtggtgaagtggc
cctctggtcactggtagtattggccgttgaaagatatatggtggtctgcaagcccatggc
caacttccgattcggggagaaccatgctattatgggtgtagccttcacatggatcatggc
tttgtcttgtgctgctcctcctctcttcggatggtccagatacatcccagagggaatgca
atgctcatgcggagtagactactacacactgaagcctgaggtcaacaatgaatcctttgt
tatctacatgttcattgtccacttcaccattcccctgattgtcatcttcttctgctatgg
tcgcctgctctgcactgtcaaagaggctgcagcccagcaacaggaatctgctaccaccca
gaaggctgagaaagaggtcaccagaatggttgttatcatggtcgttttcttcctgatctg
ttgggtgccctatgcctatgtggcattctacatcttcacccaccagggctctaactttgg
cccagtcttcatgaccgtcccagctttctttgccaagagctctgctatctacaatcctgt
catctacattgtcttgaacaaacagttccgtaactgcttgatcaccaccctgtgctgtgg......
```

Pairwise sequence alignment

This chapter is about sequence similarity. Let us start with a warning: there is no unique, precise, or universally applicable notion of similarity. An alignment is an arrangement of two sequences, which shows where the two sequences are similar, and where they differ. An optimal alignment, of course, is one that exhibits the most similarities, and the least differences. Broadly, there are three categories of methods for sequence comparison.

•Segment methods compare all overlapping segments of a predetermined length (e.g., 10 amino acids) from one sequence to all segments from the other. This is the approach used in dotplots.
•Optimal global alignment methods allow the best overall score for the comparison of the two sequences to be obtained, including a consideration of gaps.
•Optimal local alignment algorithms seek to identify the best local similarities between two sequences but, unlike segment methods, include explicit consideration of gaps.

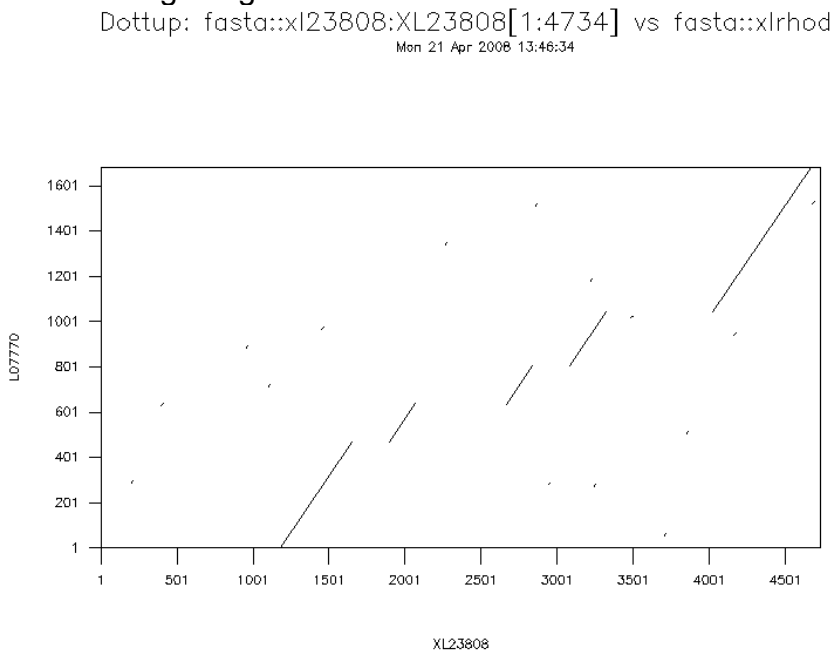
Dotplots

The most intuitive representation of the comparison between two sequences uses dot-plots. One sequence is represented on each axis and significant matching regions are distributed along diagonals in the matrix.

Exercise: Making a dotplot

Program: dottup
Displays a wordmatch dotplot of two sequences
Parameters:
`Input sequence:` xl23808
`Second sequence:` xlrhodop
(Pay attention that the second letter after x is the letter l and not number 1 (one)
`Word size [4]: 10`
`Graph type [x11]:` (Will send the image to your screen. Alternatively type in `png` or `gif` to produce a png image to load into a word processing document)

The resulting image should look like this:



The diagonal lines represent areas where the two sequences align well. You

can see that there are five clear diagonals. You will remember that we are aligning genomic and cDNA - these five diagonals represent the five exons of the gene! To confirm this, we need to do a more rigorous analysis. For this we will perform a pairwise alignment.

The basic idea behind the sequence alignment programs is to align the two sequences in such a way as to produce the highest score - a scoring matrix is used to add points to the score for each match and subtract them for each mismatch. The matrices used for nucleic acid alignments tend to involve fairly simple match/mismatch scoring schemes, while the matrices commonly used for scoring protein alignments are more complex, with scores designed to reflect similarity between the different amino acids rather than simply scoring identities. Over time various mutations occur in sequences; the scoring matrices attempt to cope with mutations, but insertions and deletions require some extra parameters to allow the introduction of gaps in the alignment. There are penalties both for the creation of gaps and for the extension of existing ones; the default gap parameters given in alignment programs have been found to be empirically correct with test sequences but you should experiment with different gap penalties.

Global alignment

A global alignment is one that compares the two sequences over their entire lengths, and is appropriate for comparing sequences that are expected to share similarity over the whole length. The alignment maximises regions of similarity and minimises gaps using the scoring matrices and gap parameters provided to the program. The EMBOSS program needle is an implementation of the Needleman-Wunsch [3] algorithm for global alignment; the computation is rigorous and needle can be time consuming to run if the sequences are long.

Exercise: needle
Program: needle
(Needleman-Wunsch global alignment).

```
Input sequence: xlrhodop
Second sequence: xl23808
Gap opening penalty [10.0]:
Gap extension penalty [0.5]:
```

Result:

```
########################################
# Program: needle
# Rundate: Mon 21 Apr 2008 14:04:19
# Commandline: needle
#    -asequence xlrhodop
#    -sbegin1 1
#    -send1 1684
#    -bsequence xl23808
#    -gapopen 10.0
#    -gapextend 0.5
#    -brief
#    -aformat srspair
#    -auto
# Align_format: srspair
# Report_file: .needle.08.04.21:14.04.19/l07770.needle
########################################

#=======================================
#
# Aligned_sequences: 2
# 1: L07770
# 2: XL23808
# Matrix: EDNAFULL
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 4734
# Identity:    1683/4734 (35.6%)
# Similarity:  1683/4734 (35.6%)
# Gaps:        3050/4734 (64.4%)
# Score: 7471.0
#
#
#=======================================

XL23808         1101 aatcctttgttcgtgacgctgggggttgcaagcttactccaggtgggact   1150

L07770             1 ------------------------------ggtagaacagcttcagttgg     20
                                                   .||||||||||||||||||
XL23808         1151 ttaaaaggacgaggggacagtgggtcatactgtagaacagcttcagttgg   1200

L07770            21 gatcacaggcttctagggatcctttgggcaaaaaagaaacacagaaggca     70
                     ||||||||||||||||||||||||||||||||||||||||||||||||||
XL23808         1201 gatcacaggcttctagggatcctttgggcaaaaaagaaacacagaaggca   1250
```

We've only shown part of the output, as it is very long. You should look at the whole output and note that there are five aligned regions that represent the five exons as predicted from the previous dotplot.

Local alignment

As we mentioned above, global sequence alignment algorithms align sequences over their entire lengths. You do need to think about whether that type of alignment makes sense for your sequences. For our example, where we expect each exon to be represented in the sequences and in the same order, it has worked well - however, how well do you think this approach would work with, for example, multidomain proteins that share one domain but not others, or sequences where there have been regions of duplication? A second comparison method, local alignment, searches for regions of local similarity and need not include the entire length of the sequences. Local alignment methods are very useful for scanning databases or when you do

not know that the sequences are similar over their entire lengths. The EMBOSS program water is a rigorous implementation of the Smith Waterman algorithm for local alignments [4].

Exercise: water
Program: water
Smith-Waterman local alignment.
Input sequence: xlrhodop
Second sequence: xl23808
Gap opening penalty [10.0]:
Gap extension penalty [0.5]:

Result:

```
########################################
# Program: water
# Rundate: Mon 21 Apr 2008 14:12:32
# Commandline: water
#    -asequence xl23808
#    -sbegin1 1
#    -send1 4734
#    -bsequence xlrhodop
#    -gapopen 10.0
#    -gapextend 0.5
#    -brief
#    -aformat srspair
#    -auto
# Align_format: srspair
# Report_file: .water.08.04.21:14.12.31/xl23808.water
########################################

#=======================================
#
# Aligned_sequences: 2
# 1: XL23808
# 2: L07770
# Matrix: EDNAFULL
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 3487
# Identity:    1683/3487 (48.3%)
# Similarity:  1683/3487 (48.3%)
# Gaps:        1804/3487 (51.7%)
# Score: 7475.0
#
#
#=======================================

XL23808         1182 gtagaacagcttcagttgggatcacaggcttctagggatcctttgggcaa   1231
                     ||||||||||||||||||||||||||||||||||||||||||||||||||
L07770             2 gtagaacagcttcagttgggatcacaggcttctagggatcctttgggcaa     51

XL23808         1232 aaaagaaacacagaaggcattctttctatacaagaaaggactttatagag   1281
                     ||||||||||||||||||||||||||||||||||||||||||||||||||
L07770            52 aaaagaaacacagaaggcattctttctatacaagaaaggactttatagag    101
```

Scroll down the entire output and again, note that five exons have been found. In these cases we have not had to adjust the gap parameters from the defaults used in these programs. You should be aware that you might need to do so with your own sequences.

EMBOSS contains other pairwise alignment programs - stretcher and matcher are global and local alignment programs respectively that are less rigorous than needle and water and therefore run more quickly; they may be useful for

database searching.

Supermatcher is designed for local alignments of very large sequences and is even less rigorous in its implementation. The documentation pages for all these programs can be found at www.emboss.org

Protein analysis

This chapter will introduce you to a few of the EMBOSS applications that can be used to analyze protein sequences. Obviously, the pairwise sequence comparison methods illustrated in the previous chapter with nucleic acid sequences can also be used with protein sequences.

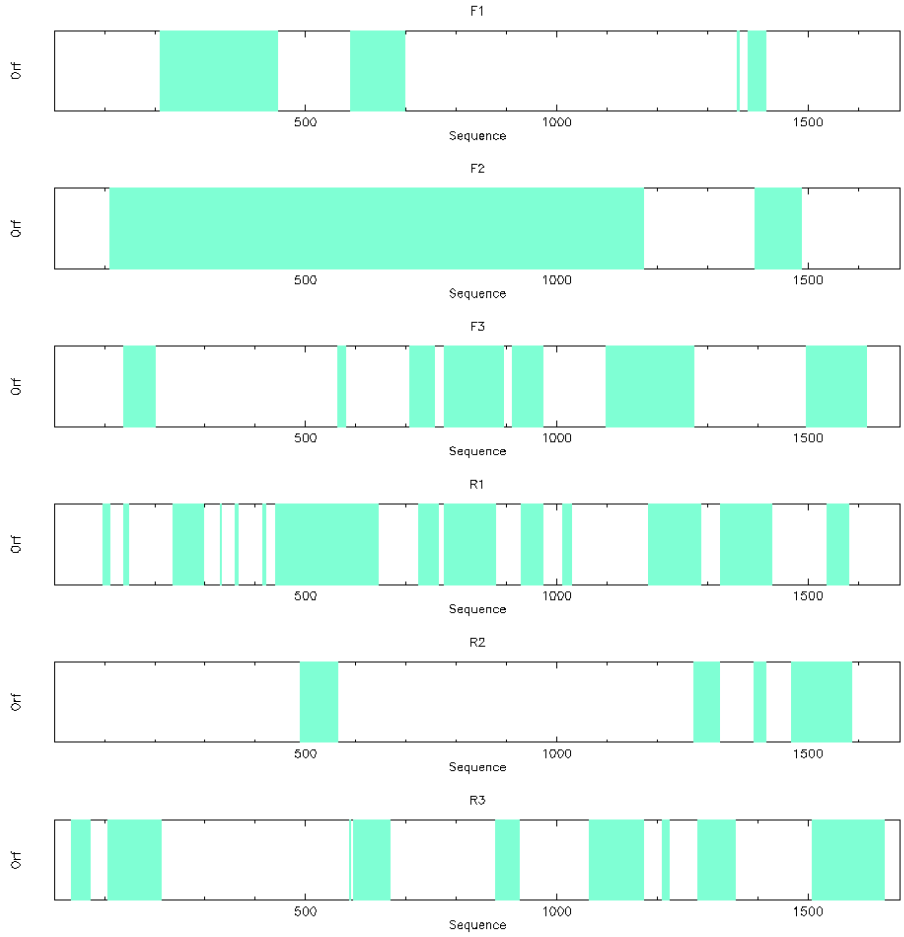
•Identifying the ORF
•Exercise: plotorf
•Exercise: getorf
•Translating the sequence
•Exercise: transeq
•Secondary structure prediction
•pepinfo
•Exercise: pepinfo
•Predicting transmembrane regions
•Exercise: tmap

Identifying the ORF

In this section we'll show you some simple EMBOSS applications for translating your cDNA sequence into protein. You should be aware that gene structure prediction is a tough problem, and recognising exon/intron boundaries in genomic sequence is not easy; for now, rather than deal with that aspect of prediction, we'll use the cDNA sequence in our practical. First, we need to identify our open reading frame. We can get a rapid visual overview of the distribution of ORFs in the six frames of our sequence using the EMBOSS program plotorf.

Exercise: plotorf
Program: plotorf
Plot potential open reading frames Input sequence: xlrhodop
Graph type [png]:

You will see a graphical output that shows the potential open reading frames (ORF) in all six frames:



Exercise: getorf

Program: getorf -opt
Finds and extracts open reading frames (ORFs)
Input sequence: xlrhodop
Make the choices:
Begin: 100
End: 2000 (this we know from plotorf)
Minimum nucleotide size of ORF to report: 0
Maximum nucleotide size of ORF to report: 1000000
Type of output: Nucleic sequence between START and STOP codons
Run getorf

Notice that you can specify the organism whose codon usage table is most appropriate for your sequence, and you can also choose the type of information that is reported to you. In our case, we are simply interested in the positions of the start and stop codons for this sequence. Plotorf is just a graphical representation of the textual information produced by getorf. Since we asked for all ORFs above a minimum size to be reported, getorf is telling

us about a number of potential ORFs. We know from plotorf that our ORF will be in the region 0 to 1200, so scroll through the output file, xlrhodop.orf, until you identify this. What are the actual start and end positions?

Result:

```
>L07770_7 [110 - 1171] Xenopus laevis rhodopsin mRNA, complete cds.
atgaacggaacagaaggtccaaatttttatgtccccatgtccaacaaaactggggtggta
cgaagcccattcgattaccctcagtattacttagcagagccatggcaatattcagcactg
gctgcttacatgttcctgctcatcctgcttgggttaccaatcaacttcatgaccttgttt
gttaccatccagcacaagaaactcagaacacccctaaactacatcctgctgaacctggta
tttgccaatcacttcatggtcctgtgtgggttcacggtgacaatgtacacctcaatgcac
ggctacttcatctttggccaaactggttgctacattgaaggcttctttgctacacttggt
ggtgaagtggccctctggtcactggtagtattggccgttgaaagatatatggtggtctgc
aagcccatggccaacttccgattcggggagaaccatgctattatgggtgtagccttcaca
tggatcatggctttgtcttgtgctgctcctcctctcttcggatggtccagatacatccca
gagggaatgcaatgctcatgcggagtagactactacacactgaagcctgaggtcaacaat
gaatcctttgttatctacatgttcattgtccacttcaccattcccctgattgtcatcttc
ttctgctatggtcgcctgctctgcactgtcaaagaggctgcagcccagcaacaggaatct
gctaccacccagaaggctgagaaagaggtcaccagaatggttgttatcatggtcgttttc
ttcctgatctgttgggtgccctatgcctatgtggcattctacatcttcacccaccagggc
tctaactttggcccagtcttcatgaccgtcccagctttctttgccaagagctctgctatc
tacaatcctgtcatctacattgtcttgaacaaacagttccgtaactgcttgatcaccacc
ctgtgctgtggaaagaatccattcggtgatgaagatggctcctctgcagccacctccaag
acagaagcttcttctgtctcttccagccaggtgtctcctgca
```

Translating the sequence

From the previous exercise you should have found that the region to be translated is from 110 to 1171 in our cDNA sequence.
Now we can use transeq to translate that region and use the translated peptide for some further analyses.
Let's practice with the program transeq. Specify the subregion of your sequence; in this case we will ask transeq to translate only the part of xlrhodop that we have identified as the coding region.

Exercise: transeq
transeq xlrhodop
Begin: 110
End: 1171
run transeq
Translate nucleic acid sequences

Result:

```
>L07770_1 Xenopus laevis rhodopsin mRNA, complete cds.
MNGTEGPNFYVPMSNKTGVVRSPFDYPQYYLAEPWQYSALAAYMFLLILLGLPINFMTLF
VTIQHKKLRTPLNYILLNLVFANHFMVLCGFTVTMYTSMHGYFIFGQTGCYIEGFFATLG
GEVALWSLVVLAVERYMVVCKPMANFRFGENHAIMGVAFTWIMALSCAAPPLFGWSRYIP
EGMQCSCGVDYYTLKPEVNNESFVIYMFIVHFTIPLIVIFFCYGRLLCTVKEAAAQQQES
ATTQKAEKEVTRMVVIMVVFFLICWVPYAYVAFYIFTHQGSNFGPVFMTVPAFFAKSSAI
YNPVIYIVLNKQFRNCLITTLCCGKNPFGDEDGSSAATSKTEASSVSSSQVSPA
```

Save as x1rhodop.pep

Secondary structure prediction

The question of how DNA sequence determines specific protein structure has been a constant source of fascination and speculation since the problem was identified. It remains an extremely difficult area; generally referred to as the ``folding problem'', it is one of the major outstanding questions in molecular biology. Many attempts have been made to predict the tertiary structure of a protein from its sequence. These fall into two distinct approaches:

•One approach is to set up a realistic mechanical model of the protein chain and simulate the folding process.
•Other approaches are empirical as they proceed by inference from known tertiary structures.

The approach to structure prediction based on mechanical models has the innate (possibly fatal) attraction that, in theory, it requires no prior knowledge of protein tertiary structure. If successful it could be applied uniformly to all sequences. By contrast, all methods based on inference from known structures are inherently limited in their applicability. They will only be appropriate for predicting structures similar to those, which were used in the inference process. Fortunately there are often biophysical or biochemical clues that help make this decision and these are often integrated in the methods for structure prediction.
Currently the best way to achieve reasonable secondary structure predictions is to run a variety of prediction algorithms over your sequence and determine a consensus among the results. There are various web servers that will do these multiple analyses for you.
As yet, coverage of secondary structure prediction within EMBOSS is limited. More algorithms will be added to enable the consensus approach described above. We'll take a look now at some of the predictions you can currently perform using EMBOSS.

•pepinfo
•Exercise: pepinfo
•Predicting transmembrane regions
•Exercise: tmap

Pepinfo

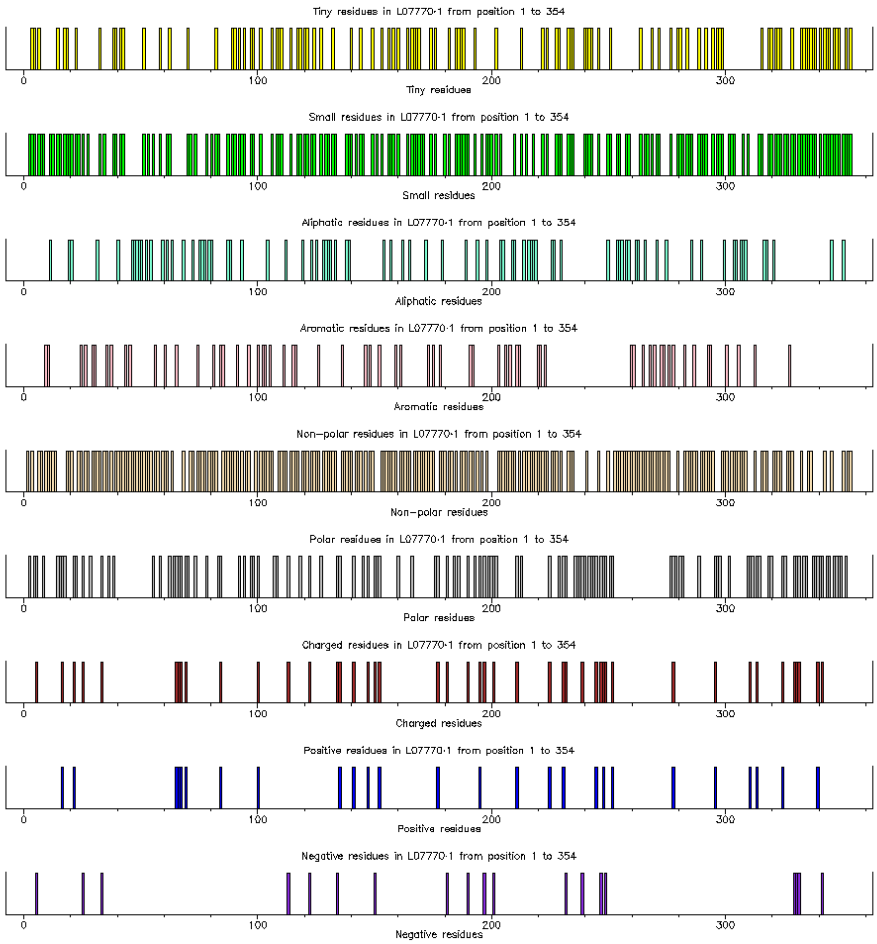
Pepinfo produces information on amino acid properties (size, polarity, aromaticity, charge etc). Hydrophobicity profiles are also available and are useful for locating turns, potential antigenic peptides and transmembrane helices. Various algorithms are employed including the Kyte and Doolittle hydropathy measure - this curve is the average of a residue-specific hydrophobicity index over a window of nine residues. When the line is in the

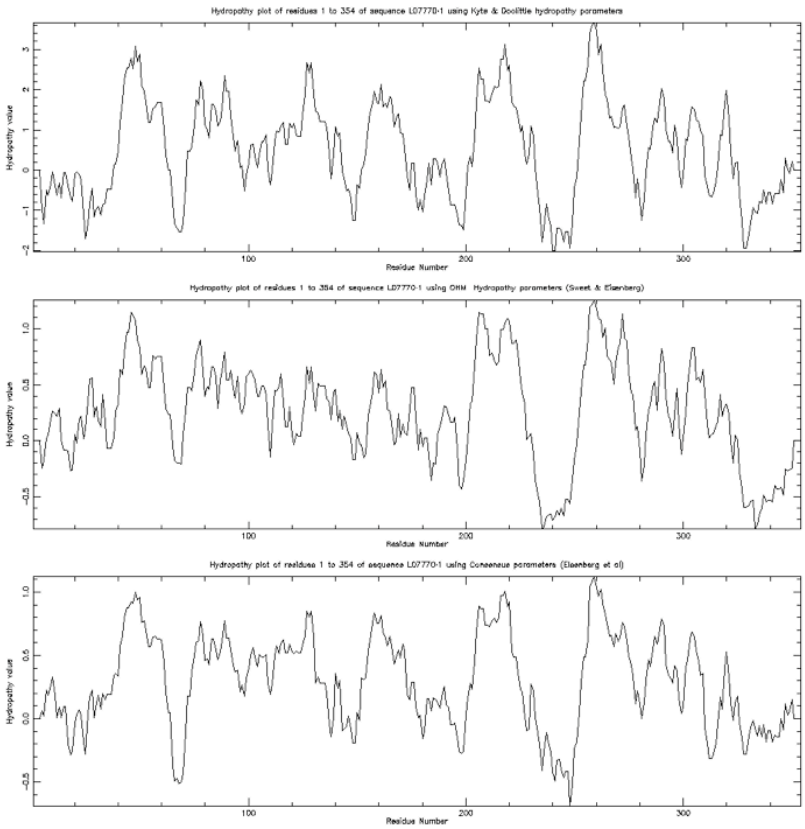
upper half of the frame, it indicates a hydrophobic region, and when it is in the lower half, a hydrophilic region.

Exercise: pepinfo
Program: pepinfo
choose xlrhodop.pep

Remember that you produced this in a previous exercise.
Plots simple amino acid properties in parallel
Graph type [png]:

You will see pictures that look like this:

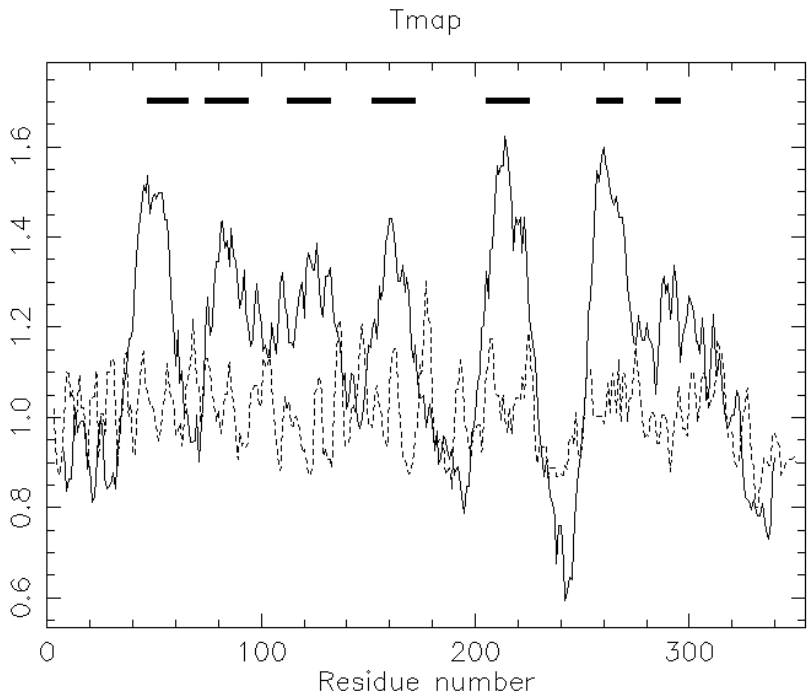




Predicting transmembrane regions

The results from the pepinfo hydropathy plot showed seven highly hydrophobic regions within xlrhodop.pep. Could these be transmembrane domains? We can use the EMBOSS program tmap to investigate this possibility:

Exercise: tmap
Program: tmap
Displays membrane spanning regions
Sequences file to be read in: xlrhodop.pep
Graph type [png]:
You will see a window that looks like this:



The bars across the top represent areas where transmembrane segments are predicted. Taken in combination with the results from pepinfo, we can see that there may be seven transmembrane helices in this protein.
There are various other programs you can use to analyze your peptide sequence - to find out what is available, try rerunning wossname as we did in the first chapter.

Pattern matching

In a number of cases, the active site of a protein can be recognized by a specific ``fingerprint'' or ``template'', a fairly small set of residues that are unique to a family of proteins. An example is the sequence GXGXXG (where G=glycine and X=any amino acid) which defines a GTP binding site. Searching for a (rather loose) predefined string of characters in a sequence is called Pattern Matching.

The EMBOSS program patmatmotifs looks for sequence motifs by searching with a pattern search algorithm through the given protein sequence for the patterns defined in the PROSITE database, compiled by Dr. Amos Bairoch at the University of Geneva. PROSITE is a database of protein families and domains, based on the observation that, while there are a huge number of different proteins, most of them can be grouped, on the basis of similarities in their sequences, into a limited number of families. Proteins or protein domains belonging to a particular family generally share functional attributes and are derived from a common ancestor.

Exercise: patmatmotifs
Program: patmatmotifs
Search a motif database with a protein sequence
Input sequence: xlrhodop.pep

Result:

```
########################################
# Program: patmatmotifs
# Rundate: Mon 21 Apr 2008 15:48:30
# Commandline: patmatmotifs
#    -sequence xirhodop.pep
#    -sbegin1 1
#    -send1 354
#    -nofull
#    -prune
#    -rformat dbmotif
#    -auto
# Report_format: dbmotif
# Report_file: .patmatmotifs.08.04.21:15.48.29/l07770_1.patmatmotifs
########################################

#=======================================
#
# Sequence: L07770_1     from: 1   to: 354
# HitCount: 2
#
# Full: No
# Prune: Yes
# Data_file: /usr/ebiotools/share/EMBOSS/data/PROSITE/prosite.lines
#
#=======================================

Length = 17
Start = position 123 of sequence
End = position 139 of sequence

Motif = G_PROTEIN_RECEP_F1_1

TLGGEVALWSLVVLAVERYMVVCKPMA
   |             |
  123           139

Length = 17
Start = position 290 of sequence
End = position 306 of sequence

Motif = OPSIN

PVFMTVPAFFAKSSAIYNPVIYIVLNK
   |             |
  290           306


#---------------------------------------
#---------------------------------------
```

In our case we already know that our sequence is a rhodopsin. However, if you had an unknown sequence, we hope you can see that identifying motifs might provide you with information to help you plan further experiments.

Protein fingerprints

PRINTS is a database that defines functional protein families, identifying each domain by a number of short, particularly well conserved sequences. A full

match to one of these "fingerprints" will match all the relevant short sequences in the correct order. A partial match is recorded if some are missing or if they occur in an incorrect order. The PRINTS database can be searched using the pscan program which is available within EMBOSS.

Exercise: pscan
Program: pscan
Scans proteins using PRINTS
Input sequence: xlrhodop.pep
Minimum number of elements per fingerprint [2]:
Maximum number of elements per fingerprint [20]:

Result:

```
CLASS 1
Fingerprints with all elements in order

Fingerprint GPCRRHODOPSN Elements 7
    Accession number PR00237
    Rhodopsin-like GPCR superfamily signature
  Element 1 Threshold 54% Score 61%
             Start position 39 Length 25
  Element 2 Threshold 49% Score 49%
             Start position 72 Length 22
  Element 3 Threshold 48% Score 55%
             Start position 117 Length 23
  Element 4 Threshold 50% Score 69%
             Start position 152 Length 22
  Element 5 Threshold 51% Score 82%
             Start position 204 Length 24
  Element 6 Threshold 42% Score 72%
             Start position 250 Length 25
  Element 7 Threshold 46% Score 68%
             Start position 288 Length 27


CLASS 2
All elements match but not all in the correct order

Fingerprint RHODOPSIN Elements 6
    Accession number PR00579
    Rhodopsin signature
  Element 1 Threshold 80% Score 100%
             Start position 3 Length 19
  Element 2 Threshold 76% Score 94%
             Start position 22 Length 17
  Element 3 Threshold 53% Score 90%
             Start position 85 Length 17
  Element 4 Threshold 71% Score 100%
             Start position 191 Length 17
  Element 5 Threshold 56% Score 97%
             Start position 271 Length 19
  Element 6 Threshold 81% Score 95%
             Start position 319 Length 14

CLASS 3 Not all elements match but those that do are in order

CLASS 4 Remaining partial matches
```

Multiple Sequence Analysis

The simultaneous alignment of many nucleotide or amino acid sequences is

now an essential tool in molecular biology. Multiple alignments are used to find diagnostic patterns to characterize protein families; to detect or demonstrate homology between new sequences and existing families of sequences; to help predict the secondary and tertiary structures of the new sequences; to suggest oligonucleotide primers for PCR; and as an essential prelude to molecular evolutionary analysis.

One of the most popular programs for performing multiple sequence alignments is clustalw ([1]). EMBOSS has an interface to clustal called emma clustal (and thus emma) creates a multiple sequence alignment from a group of related sequences using progressive pairwise alignments. It can also produce a dendogram showing the clustering relationships used to create the alignment.

The dendogram shows the order of the pairwise alignments of sequences and clusters of sequences that together generate the final alignment, but it is not an evolutionary tree, although the length of the branches is related to the relative distance of the sequences. clustal finds global optimal alignments. The alignment procedure begins with the pairwise alignment of the two most similar sequences, producing a cluster of two aligned sequences. This cluster can then be aligned to the next most related sequence or cluster of aligned sequences.Two clusters of sequences can be aligned by a simple extension of the pairwise alignment of two individual sequences. The final alignment is achieved by a series of progressive, pairwise alignments that include increasingly dissimilar sequences and clusters, until all sequences have been included in the final pairwise alignment. When gaps are inserted into a sequence to produce an alignment, they are inserted at the same position in all the sequences of the cluster. Each pairwise alignment uses the method of Needleman and Wunsch extended for use with clusters of aligned sequences.

pscan has told us that our sequence belongs to the rhodopsin family. This is a very large family of sequences - for example, you can see the Pfam entry for rhodopsin by doing a keyword search at
http://www.sanger.ac.uk/Software/Pfam
We will now use some further members of the family from SwissProt and produce a multiple alignment; we'll then use this multiple alignment to produce a profile of this group of sequences and use that to align them all to our original sequence.

•Exercise: emma
•Exercise: prettyplot

Exercise: emma

Exercise: emma
Program: emma
Multiple alignment program - interface to ClustalW program
Input sequence: OPS2.fasta
Parameters:
type=protein
output=fasta
matrix=blosum
gapopen=10.000
gapext=5.000
gapdist=8
hgapresidues=GPSNDQEKR
maxdiv=30..

We have aligned OPS2_* sequences from two fruit fly species; two crab species, locust and scallop. Let's see what emma made of them. Open file output with a text editor.

Result:

```
>OPS2_DROME
MERSHLPETPFDLAHSGPRFQAQSSGNGSVLDNVLPDMAHLVNPYWSRFAPMDPMMSKIL
GLFTLAIMIISCCGNGVVVYIFGGTKSLRTPANLLVLNLAFSDFCMMASQSPVMIINFYY
-ETWVLGPLWCDIYAGCGSLFGCVSIWSMCMIAFDRYNVIVKGINGTPMTIKTSIMKILF
IWMMAVFWTVMPLIG-WSAYVPEGNLTACSIDYMTRMWNPRSYLITYSLFVYYTPLFLIC
YSYWFIIAAVAAHEKAMREQAKKMNVKSLRSSED-CDKSAEGKLAKVALTTISLWFMAWT
PYLVICYFGLFKIDG-LTPLTTIWGATFAKTSAVYNPIVYGISHPKYRIVLKEKCPMCVF
GNTDEPKPDAPASDTETTSEADSKA----------------------------------
----------------------------
>OPS2_DROPS
MERSLLPEPPLAMALLGPRFEAQTGGNRSVLDNVLPDMAPLVNPHWSRFAPMDPTMSKIL
GLFTLVILIISCCGNGVVVYIFGGTKSLRTPANLLVLNLAFSDFCMMASQSPVMIINFYY
-ETWVLGPLWCDIYAACGSLFGCVSIWSMCMIAFDRYNVIVKGINGTPMTIKTSIMKIAF
IWMMAVFWTIMPLIG-WSSYVPEGNLTACSIDYMTRQWNPRSYLITYSLFVYYTPLFMIC
YSYWFIIATVAAHEKAMRDQAKKMNVKSLRSSED-CDKSAENKLAKVALTTISLWFMAWT
PYLIICYFGLFKIDG-LTPLTTIWGATFAKTSAVYNPIVYGISHPNDRLVLKEKCPMCVC
GTTDEPKPDAPPSDTETTSEAESKD----------------------------------
----------------------------
>OPS2_LIMPO
----------MANQLSYSSLGWPYQPNASVVDTMPKEMLYMIHEHWYAFPPMNPLWYSIL
GVAMIILGIICVLGNGMVIYLMMTTKSLRTPTNLLVVNLAFSDFCMMAFMMPTMASNCFA
-ETWILGPFMCEVYGMAGSLFGCASIWSMVMITLDRYNVIVRGMAAAPLTHKKATLLLLF
VWIWSGGWTILPFFG-WSRYVPEGNLTSCTVDYLTKDWSSASYVIIYGLAVYFLPLITMI
YCYFFIVHAVAEHEKQLREQAKKMNVASLRANADQQKQSAECRLAKVAMMTVGLWFMAWT
PYLIIAWAGVFSSGTRLTPLATIWGSVFAKANSCYNPIVYGISHPRYKAALYQRFPSLAC
GSGESGSDVKSEASATMTMEEKPKSPEA--------------------------------
----------------------------
>OPS2_HEMSA
---MTNATGPQMAYYGAASMDFGYPEGVSIVDFVRPEIKPYVHQHWYNYPPVNPMWHYLL
GVIYLFLGTVSIFGNGLVIYLFNKSAALRTPANILVVNLALSDLIMLTTNVPFFTYNCFS
GGVWMFSPQYCEIYACLGAITGVCSIWLLCMISFDRYNIICNGFNGPKLTTGKAVVFALI
SWVIAIGCALPPFFG-WGNYILEGILDSCSYDYLTQDFNTFSYNIFIFVFDYFLPAAIIV
FSYVFIVKAIFAHEAAMRAQAKKMNVSTLRSNEA-DAQRAEIRIAKTALVNVSLWFICWT
PYALISLKGVMGDTSGITPLVSTLPALLAKSCSCYNPFVYAISHPKYRLAITQHLPWFCV
HETETKSNDDSQSNSTVAQDKA----------------------------------------
----------------------------
>OPS2_SCHGR
-----MVNTTDFYPVPAAMAYESSVGLPLLGWNVPTEHLDLVHPHWRSFQVPNKYWHFGL
AFVYFMLMCMSSLGNGIVLWIYATTKSIRTPSNMFIVNLALFDVLMLLEMPMLVVSSLFY
-QRPVGWELGCDIYAALGSVAGIGSAINNAAIAFDRYRTISCPIDG-RLTQGQVLALIAG
TWVWTLPFTLMPLLRIWSRFTAEGFLTTCSFDYLTDDEDTKVFVGCIFAWSYAFPLCLIC
CFYYRLIGAVREHEKMLRDQAKKMNVKSLQSNADTEAQSAEIRIAKVALTIFFLFLCSWT
PYAVVAMIGAFGNRAALTPLSTMIPAVTAKIVSCIDPWVYAINHPRFRAEVQKRMKWLHL
GEDARSSKSDTSSTATDRTVGNVSASA----------------------------------
```

```
--------------------------
>OPS2_PATYE
-------------------------------------MPFPLNRTDTALVISPSEFRII
GIFISICCIIGVLGNLLIIIVFAKRRSVRRPINFFVLNLAVSDLIVALLGYPMTAASAFS
-NRWIFDNIGCKIYAFLCFNSGVISIMTHAALSFCRYIIICQYGYRKKITQTTVLRTLFS
IWSFAMFWTLSPLFG-WSSYVIEVVPVSCSVNWYGHGLGDVSYTISVIVAVYVFPLSIIV
FSYGMILQEKVCKD------SRKNGIRAQQRYTPRFIQDIEQRVTFISFLMMAAFMVAWT
PYAIMSALAIGSFNV--ENSFAALPTLFAKASCAYNPFIYAFTNANFRDTVVEIMAPWTT
RRVGVSTLPWPQVTYYPRRRTSAVNTTDIEFPDDNIFIVNSSVNGPTVKREKIVQRNPIN
VRLGIKIEPRDSRAATENTFTADFSVI
```

The sequences are very similar, but there are some differences - note the gaps that have been inserted. Also note that since this is a global alignment algorithm, gaps have been inserted to make all the sequences the same length.
Differences in alignment can be very difficult to see in this format. The program Prettyplot can enhance visualisation of your results, by aligning the sequences on top of one another. Save the results as OSP2.aln.

Exercise: prettyplot

Open OPS2.aln in prettyplot!

Exercise: prettyplot
Program: prettyplot
Displays aligned sequences, with coloring and boxing
Input sequence set: OPS2.aln
Graph type [png]:
You will obtain two png images. Identical residues are shown in red, and similar residues in green. This type of display can give you a first impression of regions of conservation.

Alternatively the output from multiple alignment can be opened in Jalview.

Bibliography

1
D.G. Higgins J.D. Thompson and T.J. Gibson.
CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position specific gap penalties and weight matrix choice. Nucleic Acids Research., 22:4673-4680, 1994.

2
A.D. McClachlan M. Gribskov and D. Eisenberg.
Profile analysis - detection of distantly related proteins. .
Proc. Natl. Acad. Sci. USA, 84:4355-4358, 1987.

3
S.B. Needlman and C.D. Wunsch.
A general method applicable to the search for similarities in the amino acid sequence of two proteins. .
J. Mol. Biol., 48:443-453, 1970.

4
T.F. Smith and M.S. Waterman.
Identification of common molecular subsequences.
J. Mol. Biol., 147:195-197, 1981.

About this tutorial ...

Introduction to Sequence Analysis using wEMBOSS
This documents were originally prepared for the command line EMBOSS.
The first version of this document was generated using the LaTeX2HTML translator Version 98.1p1 release (March 2nd, 1998)

The command line arguments were: latex2html emboss_tutorial.tex.
The translation was initiated by EMBnet on 2003-09-10
Changed for wEMBOSS September 2007 by Erik Bongcam-Rudloff using iWeb on MacOSX.
Latest version 19 September 2007.