A Brief Introduction to Multiple Sequence Analysis through GCG’s SeqLab

The SeqLab Graphical User Interface (GUI) is a ‘front-end’ to the Wisconsin Sequence Analysis Package. It provides an intuitive alternative to command line by allowing menu-driven access to most of GCG’s almost 150 different programs and is a great way to develop, refine, and analyze multiple sequence alignments. So what’s so great about a multiple sequence alignment? They are:

• very useful in the development of PCR primers and hybridization probes;
• great for producing annotated, publication quality, graphics and illustrations;
• invaluable in structure/function studies through homology inference;
• essential for building “Profiles” for remote homology similarity searching; and
• required for molecular evolutionary phylogenetic inference programs such as those from PAUP* (Phylogenetic Analysis Using Parsimony [and other methods]) and PHYLIP (PHYLogeny Inference Package).

This introductory tutorial will illustrate many of SeqLab’s multitude of features, just the ‘tip-of-the-iceberg,’ hopefully whetting your appetite enough to encourage further exploration.

July 25, 2005

A GCG® Wisconsin Package™ SeqLab® tutorial for the Woods Hole Marine Biological Laboratory’s Workshop on Molecular Evolution.

Author and Instructor: Steven M. Thompson
Introduction

The power and sensitivity of sequence based computational methods dramatically increases with the addition of more data. As in pair-wise comparisons, those areas most resistant to change are functionally the most important to the molecule. However, with increased dataset size, the patterns of conservation become evermore clear. But how does one work with more than just two sequences at a time? You could painstakingly manually align all your sequences using some type of editor, and many people do just that, but some type of an automated solution is desirable, at least as a starting point to manual alignment. However, solving the dynamic programming algorithm for more than just two sequences rapidly becomes intractable as computational needs increase with the exponent of the dataset size (complexity=\[\text{sequence length}\]num\text{ber of sequences}). Mathematically this is an N-dimensional matrix, quite complex indeed. One program, MSA (version 2.0, 1995), does attempt to globally solve this equation, however, the algorithm’s complexity precludes its use in most situations.

Several heuristics have been employed over the years to simplify the complexity of the problem. One way to still globally solve the equation, and yet reduce its complexity is to restrict the search space to only the most conserved ‘local’ portions of all the sequences involved. This approach is used by the program PIMA (version 1.4, 1995). However, the most commonly used approach to the problem is known as the pairwise, progressive dynamic programming solution. This variation of the dynamic programming algorithm generates a global alignment, but restricts its search space at any one time to the local neighborhood of the full length of just two sequences. The pairwise, progressive solution is implemented in several programs including Des Higgins’ ClustalW (1994) and the GCG program PileUp. Both programs insert gaps to align the full length of a sequence set to produce a multiple sequence alignment.

Given a particular sequence of interest, one can use any text search tool, such as GCG’s LookUp, or NCBI’s Entrez, or other tools on the World Wide Web, to find that entry’s name in a sequence database. After the entry has been identified a natural next step is to use some type of similarity matching program, such as FastA and/or BLAST to help prepare a list of sequences to be aligned. One of the more difficult aspects of multiple sequence alignment is knowing what sequences you should attempt it with. Any list of sequences from any program will need to be restricted to only those sequences that actually should be aligned. Beware the ‘apples and oranges’ problem. Make sure that the group of sequences that you align are in fact related, that they actually belong to the same gene family, and that the alignment is meaningful. An alignment is a statement of homology — be sure that it makes sense. Either make paralogous (i.e. evolution via gene duplication) comparisons to ascertain gene phylogenies, or orthologous (within one ancestral loci) comparisons to estimate organismal phylogenies; try not to mix them up without complete data representation. Confusion and misleading interpretations can result otherwise. Also be wary of trying to align genomic sequences with cDNA when working with DNA; the introns will cause all sorts of headaches. Similarly, don’t align mature and precursor proteins from the same organism and loci. It doesn’t make evolutionary sense, as one is not evolved from the other, rather one is the other. These are all easy mistakes to make; try your best to avoid them.
As in pairwise alignment and sequence database searching, all of this stuff is much easier with protein sequences versus nucleotides. Twenty symbols are just much easier to align than only four; the signal to noise ratio is so much better, and amino acids have the concept of similarity. If you are forced to align nucleotides, the whole process becomes much more difficult. Therefore, as it is in database searching, translate nucleotide sequences to their protein counterparts, if you are dealing with coding sequences, before performing further analyses, including multiple sequence alignment. If one is required to align nucleotides because the region does not code for a protein, then automated methods may be able to help as a starting point, but they are certainly not guaranteed to come up with a biologically correct alignment. The resulting alignment will probably have to be extensively edited, if it works at all. Nucleotides are that much more difficult to align.

Profiles are a tremendously powerful approach. Originally described by Gribskov (1987), later refinements have added more statistical rigor (see e.g. Eddy’s Hidden Markov Model Profiles [1996 and 1998]). The strategy involves preparing and refining a multiple sequence alignment of significantly similar sequences, or domains within sequences, and then generating a ‘profile’ from that alignment. Databases can then be searched with the profile. Profile searching is tremendously powerful. It can provide the most sensitive, albeit extremely computationally intensive, database similarity search possible. Often profile analysis can show features not obvious to individual members. A distinct advantage is further manipulations and database searches using the profile algorithms consider evolutionary issues. Gaps are penalized more heavily in conserved areas than they are in variable regions and the more highly conserved a residue is, the more important it becomes. Furthermore, any generated consensus sequences are not based merely on the positional frequency of particular residues but rather utilize the evolutionary conservation of substitutions based on the amino acid substitution matrix specified, by default the BLOSUM62 table (Henikoff and Henikoff, 1992). Therefore, the resultant consensus residues are the most evolutionarily conserved, rather than just statistically the most frequent. This can mean much more to us than an ordinary consensus and is especially appropriate in the design of hybridization and PCR probes for unknown sequences where data is available in related species.

We can visualize these areas of an alignment that profile searching puts the most emphasis on. They are the most conserved areas of an alignment, and thus structurally and functionally the most important. Realize that in addition to the primary sequence conservation seen in these regions, structure and function is also conserved. We will use SeqLab’s built in color functions and the GCG program PlotSimilarity to help visualize these crucial regions within our alignment. PlotSimilarity can be used to ascertain alignment quality by showing which portions of an alignment are conserved, by indicating the overall average similarity, and by noting the changes in these estimates as an alignment is adjusted. Furthermore, PlotSimilarity is a very helpful assistant in probe design by allowing you to visualize the most important, conserved regions of an alignment. It is invaluable for designing phylogenetic specific probes as it clearly localizes areas of high conservation and variability in an alignment. Depending on the dataset that you analyze, any level of phylogenetic specificity can be achieved. Pick areas of high variability in the overall dataset that correspond to areas of high conservation in phylogenetic category subset datasets to differentiate between universal and specific potential probe sequences. One can then use various primer discovery programs such as the GCG program Prime to further localize and test potential probes for common PCR conditions and problems.
Finally, we can use multiple sequence alignments to infer phylogeny. A multiple sequence alignment is itself a hypothesis about evolutionary history. Based on the explicit assertion of homologous positions in an alignment several algorithms available can estimate the most reasonable evolutionary tree for that alignment. Therefore, devote considerable time and energy toward developing the most satisfying multiple sequence alignment possible. Quality alignments mean everything for obtaining meaningful results from phylogenetic inference algorithms. All of the molecular sequence phylogenetic inference programs make the validity of your input alignment their first and most critical assumption. Be sure that the alignment makes biological sense. Use all available information and understanding to insure that your alignment is as good as it can be. Make sure that known enzymatic, regulatory, and structural elements all align, for the results of your inference are absolutely dependent upon your alignment. To help assure the reliability of any alignment always use comparative approaches. Look for conserved structural and functional sites to help guide your judgment. In ribosomal RNA alignments researchers have successfully used the conservation of covarying sites to assist in this process. That is, as one base in a stem structure changes the corresponding Watson-Crick paired base will change in a corresponding manner. This process has been used extensively by the Ribosomal Database Project formerly at the University of Illinois, Urbana Campus, but now housed at the Center for Microbial Ecology at Michigan State University to help guide the construction of their rRNA alignments and structures (http://rdp.cme.msu.edu/index.jsp). Use everything available to insure that you have prepared a satisfying alignment. Remember the old adage: “garbage in — garbage out!”

One of the biggest problems in this field is that of sequence format. Each suite of programs requires a different sequence format. GCG sequence format exists both as single and Multiple Sequence Format (MSF) and SeqLab has its own format called Rich Sequence Format (RSF) that contains both sequence data and reference and feature annotation. PAUP* has a required format called the NEXUS file and PHYLIP has its own unique input data format requirements. Several different programs are available to allow us to convert formats back and forth between the required standards, but it all can get quite confusing. One program, ReadSeq by Don Gilbert at Indiana University, allows for the back and forth conversion between several different formats. The PAUP* interfaces in the GCG system, PAUPSearch and PAUPDisplay, automatically generate their required NEXUS format directly from the GCG formatted files, so this is not nearly as much of a hassle. Alignment gaps are another problem. Different program suites may use different symbols to represent them. Furthermore, not all gaps in sequences should be interpreted as deletions. Interior gaps are probably okay to represent this way, as regardless of whether a deletion, insertion or a duplication event created the gap, logically they will be treated the same by the algorithms. These are called indels. However, end gaps should not be represented as indels because a lack of information beyond the length of a given sequence may not be due to a deletion or insertion event. It may have nothing to do with the particular stretch being analyzed at all. It may just not have been sequenced! These gaps are just placeholders for the sequence. Therefore, it is safest to manually edit an alignment to change leading and trailing gap symbols to ‘unknown.’ This will assure that the programs do not make incorrect assumptions about your sequences.

I reiterate, the most important factor in inferring reliable phylogenies is the accuracy of the multiple sequence alignment. The interpretation of your results is utterly dependent on the quality of your input. In fact, many experts advice against using any parts of the sequence data that are at all questionable. Only analyze those
portions that assuredly do align. If any portions of the alignment are in doubt, throw them out. This usually
means trimming down the alignment’s terminal ends and may require internal trimming as well. SeqLab
makes this process much easier than previous means. Another possibility is to exclude portions with
SeqLab’s Mask option. This allows the user to differentially weight different parts of their alignment to reflect
their confidence in it. It can be a handy trick with some data sets, especially those with both highly conserved
and highly variable regions.

**SeqLab exercise**

I will illustrate the techniques in this exercise with a dataset containing a protein with representatives in all the
branches of cellular life. I use a broad representation across all cellular life while still keeping within the
practical limits of this evening’s computer lab session. In the exercise you will use the GCG GUI SeqLab and
the program PileUp to prepare and refine an alignment of this protein. You will also use the programs
PlotSimilarity, Motifs, and HelicalWheel to analyze it. And finally, you will learn about some of the tools and
tricks available for producing output appropriate as input to phylogenetic inference software.

The Elongation Factors are a vital protein family crucial to protein biosynthesis. They are ubiquitous to all of
cellular life and, in concert with the ribosome, they must have been one of the very earliest enzymatic
factories to evolve. Three distinct subtypes of elongation factors all work together to help perform the vital
function of protein biosynthesis. In [Eu|Bacteria and Eukaryota nuclear genomes they have the following
names (the nomenclature in Archaea has not been completely worked out and is often contradictory):

<table>
<thead>
<tr>
<th>Eukaryota</th>
<th>[Eu]Bacteria</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>EF-1α</td>
<td>EF-Tu</td>
<td>Binds GTP and an aminoacyl-tRNA; delivers the latter to the A site of ribosomes.</td>
</tr>
<tr>
<td>EF-1β</td>
<td>EF-Ts</td>
<td>Interacts with EF-1α/Tu to displace GDP and thus allows the regeneration of GTP-EF-1α/Tu</td>
</tr>
<tr>
<td>EF-2</td>
<td>EF-G</td>
<td>Binds GTP and peptidyl-tRNA and translocates the latter from the A site to the P site.</td>
</tr>
</tbody>
</table>

The Elongation Factor subunit 1-Alpha (EF-1α) in Eukaryota and most Archaea (called Elongation Factor Tu
in [Eu]Bacteria [and Euk and Arch' plastids]) has guanine nucleotide, ribosome, and aminoacyl-tRNA binding
sites, and is essential to the universal process of protein biosynthesis, promoting the GTP-dependent binding
of aminoacyl-tRNA to the A-site of the intact ribosome. The hydrolysis of GTP to GDP mediates a
conformational change in a specific region of the molecule. This region is conserved in both EF-1α/Tu and
EF-2/G and seems to be typical of GTP-dependent proteins which bind non-initiator tRNAs to the ribosome.

In *E. coli* EF-Tu is encoded by a duplicated loci, *tufA* and *tufB* located about 15 minutes apart on the
chromosome at positions 74.92 and 90.02 (ECDC). In humans at least twenty loci on seven different
chromosomes demonstrate homology to the gene. However, only two of them are potentially active; the
remainder appear to be retropseudogenes (Madsen, et al., 1990). It is encoded in both the nucleus and
mitochondria and chloroplast genomes in eukaryotes and is a globular, cytoplasmic enzyme in all life forms.

The three-dimensional structure of Elongation Factor 1α/Tu has been solved in more than fifteen cases.
Partial and complete *E. coli* structures have been resolved and deposited in the Protein Data Bank (1EFM,
1ETU, 1DG1, 1EFU, and 1EFC), the complete *Thermus aquaticus* and *Thermus thermophilus* structures have
been determined (1TTT, 1EFT, and 1AIP), and even cow EF-1α has had its structure determined (1D2E).
Most of the structures show the protein in complex with its nucleotide ligand, some show the ternary complex. The *Thermus aquaticus* structure is shown below as drawn by NCBI's Cn3D molecular visualization tool:

Notice that half of the protein has well defined alpha helices and the rest is rather unordered coils partly defined by beta strands. GTP fits right down in amongst all the helices in the pocket. The *Thermus aquaticus* structure has six well-defined helices that occur from residue 24 through 38, 86 through 98, 114 through 126, 144 through 161, 175 through 184, and 194 through 207. There are also two short helices at residues 47 to 51 and 54 to 59. The guanine nucleotide binding site involves residues 18 to 25, residues 81 to 85, and residues 136 to 139. Residue 8 is associated with aminoacyl-tRNA binding.

Because of strong evolutionary pressure resulting in very slow divergence and because of its ubiquity, EF-1α is an appropriate gene on which to estimate early life phylogenies and with which to ask early branching order questions in ‘deep’ eukaryotic evolution. In fact, a series of papers in the early-90’s, notably those by Iwabe, et al. (1989), Rivera and Lake (1992), and Hasegawa, et al. (1993) all base ‘universal’ trees of life on this gene. Iwabe, et al. used the trick of aligning the EF-1α gene parologue EF-1β to their EF-1α dataset to root the tree.

I) **Log onto your UNIX-based host account using X Windows.**

I use **bold type** in this tutorial for those commands and keystrokes that you are to type in at your console or for buttons that you are to click in SeqLab. I also use **bold type** for section headings. Screen traces are shown in a “typewriter” style **Courier font** and “///////////” indicates abridged data. The greater-than symbol, “>” indicates the system prompt and should not be typed as a part of commands. **Really important statements may be underlined.**
The Wisconsin Package only runs on server computers running the UNIX operating system, but it can be accessed from any networked terminal. SeqLab requires X Windows graphics for its display. This can be supplied through genuine X Windowing on a UNIX (including Linux) workstation or through X server emulation on desktop personal computers running operating systems other than UNIX. Microsoft Windows/Intel machines are often set up with either Xwin32 or eXceed to provide this function; pre-OS X Macintoshes are often loaded with either MacX or eXodus software, OS X Macs can run true X11 windowing since they are actually UNIX machines. Apple distributes an X11 package for these machines and another implementation called XDarwin is also available.

Each participant in the session should use a different UNIX account. Login with the account and password supplied to you at the beginning of the workshop. Use the appropriate connection commands on the workstation that you are sitting at to launch X and connect to the UNIX host computer that runs GCG at this site. An X-style terminal window should appear on the desktop after a few moments, if it doesn’t, launch one with the appropriate command. Get an instructor’s assistance for this step if you are unsure of yourself. There are too many variations in method for them all to be described here. I am also available for individualized personal help in your own laboratories back home, if you are having difficulties connecting with and using the GCG server there; just contact me at stevet@bio.fsu.edu. A couple of ‘X’ tips should be mentioned at this point though. Rather than holding mouse buttons down, to activate items, just click on them; and buttons are turned on when they are pushed in and shaded. Also, do not close windows with the X-server software’s close icon in the upper right- or left-hand window corner, rather, always use GCG’s “Close” or “Cancel” or “OK” button, usually at the bottom of the window.

2) Multiple Sequence Alignment — Introducing GCG’s SeqLab.

First let’s look at the list file that I have prepared for your use. Use the GCG command “fetch” to pull the file into your account from the GCG public databases (this is not usual GCG practice):

> fetch EF1a-Tu.pep.list

Check out the list file format. Following the more command:

> more EF1a-Tu.pep.list

This is a list of 25 representative Elongation Factor 1 Alpha (Tu in[Eu]Bacteria) protein sequences. This list spans all of cellular life and attempts to collect sequences from a broad phylogenetic spectrum available in Swiss-Prot. ..

SwissProt:EF10_XENLA
SwissProt:EF11_DRome
SwissProt:EF11_HUMAN
SwissProt:EF1A_ARATH
SwissProt:EF1A_DICDI
SwissProt:EF1A_ENTHI
SwissProt:EF1A_EUGGR
SwissProt:EF1A_GIALA
SwissProt:EF1A_ONCVO
SwissProt:EF1A_PLAFK
OK, now for something completely different. The SeqLab GUI, based on Steve Smith’s (1994) GDE (the Genetic Data Environment) makes running the Wisconsin Package much more intuitive by providing a common editing interface from which GCG programs can be launched. Launch GCG’s GUI by typing “seqlab &” (without the quotes). The ampersand, “&,” is not necessary but it allows you to retain control of the initial terminal window by running SeqLab in the background, thereby not tying up the terminal window. This way you can switch back and forth between the terminal and SeqLab windows. After a moment two more windows will open; click in the smaller “Welcome” one and check “OK.” This will put you in the main SeqLab window where all analyses may be performed. All menus that I refer to from this point on in SeqLab will be within the SeqLab display, not anywhere else on the monitor — those are menus that talk to your workstation, not the UNIX host. Before beginning the analyses, go to the “Options” menu and select “Preferences . . .” The defaults are usually fine; I just want to point out some helpful settings.

First notice that there are three different “Preferences” settings that can be changed: “General”, “Output,” and “Fonts”; start with “General.” The “Working Dir . . .” setting will be the directory from which SeqLab was initially launched. This is where all SeqLab’s working files will be stored; it can be changed if desired, however, for now leave it as is. Be sure that the “Start SeqLab in:” choice has “Main List” selected and that “Close the window” is selected under the “After I push the ‘Run’ button.” choice. Next select the “Output Preferences.” Make sure that “Automatically display new output” is turned on (pushed in and shaded!). Leave the other choices alone. Take a look at the “Fonts” menu next. We will leave all these choices as is, but if you are dealing with large alignments and/or are using a small monitor, then changing to a smaller Editor font point size may be desirable, to allow you to see more of the alignment at once. Click “OK” to accept any changes. Next under the “Options” menu, take a look at “Graphics Devices . . .”. The site’s standard SetPlot menu should be displayed, if it’s set up; press the “Cancel” button to close the window. Now the SeqLab interface is ready to be utilized.

Be sure the “Mode:” “Main List” choice is selected and then go to the “File” menu. Pick “Add sequences from” and select “Sequence Files.” (Only GCG format compatible sequences or list files are accessible through this route. Use SeqLab’s Editor “Import” function to directly load GenBank format sequences or ABI style trace files without the need to reformat.) This will produce an “Add Sequences” window from which you can select sequences to add to your working.list. The “Filter” box is very important here! By default files are
filtered such that only those that end with the extension ".seq" are displayed. This won't do us any good as the sequences that we want to add are in the list file that you fetched into your account at the very beginning. Therefore, delete the ".seq" extension in the "Filter" box (including the period); be sure to leave the "*" wild card. Press the "Filter" button to display all of the files in your working directory. Select the file entitled "EF1a-Tu.pep.list" from the "Files" box, and then check the "Add" and then the "Close" buttons at the bottom of the window to put the file in your working list. It will appear in the SeqLab "Main List" window. Be sure it is selected and now switch to "Mode:" "Editor" to load the sequences into the SeqLab editor. Notice that all of the sequences now appear in the editor window with color-coded amino acids. The nine color groups are based on a clustering of the BLOSUM62 matrix (Henikoff and Henikoff, 1992) and roughly categorize the amino acids based on their physical properties. Expand the window full-screen. You should be able to see all of your sequences now. The display will look something like this:

![Image of SeqLab editor interface]

Explore the editor interface for a while. Nearly all GCG programs are accessible through the "Functions" menu including the powerful similarity search tools FastA and BLAST and the profile suites mentioned in the introduction. (Do not run any similarity searches at this point.) The scroll bar at the bottom allows you to move through the sequences linearly. You can select any sequence or position by 'capturing' them with the mouse. The "pos:" and "col:" indicators show you where the cursor is located in any particular sequence and the overall dataset respectively. The "1:1" scroll bar near the upper right-hand corner allows you to 'zoom' in or out on the sequences; move it to 2:1 and beyond and notice the difference in the display. Go to the "Display:" box and change it from "Residue Coloring" to "Feature Coloring." The colors are now based on the information from the database Feature Table for each entry. Change the "Display:" to "Graphic Features:" now the features are represented using the same colors as before but in a 'cartoon' fashion. Use
the mouse to move your cursor to one of the colored areas; quickly double-click it (or use the “Features” selection under the “Windows” menu). This will produce a new window that describes the features located at the cursor. Click on one of the features to get more information on it and to select it in its entirety. All the features are fully editable through the “Edit” check box in this panel and new features can be added with several desired shapes and colors through the “Add” check box. The display will look something like my example below:

Close the "Sequence Features" window and return your display to "1:1."

3) Structural analysis and annotation.

While on the topic of feature annotation, let’s briefly explore a protein structural analysis program available within GCG. As most of you realize, structural prediction is fraught with difficulties. However, using comparative multiple sequence approaches is by far the most reliable strategy. In my opinion, the best predictor of secondary structure around, PredictProtein, available on the Web at http://www.embl-heidelberg.de/predictprotein/predictprotein.html, offered by the Protein Design Group at the European Molecular Biology Laboratory, Heidelberg, Germany, uses multiple sequence alignment profile techniques along with neural net technology. A multiple sequence alignment is performed by a weighted dynamic programming method (MaxHom, Schneider, 1991) and a secondary structure prediction is produced by the profile network method (PHD). PHD is rated at an expected 70.2% average accuracy for the three states
helix, strand, and loop (Rost and Sander, 1993 and 1994). In fact, even three-dimensional modeling without crystal coordinates is possible. This is “homology modeling.” It will often lead to remarkably accurate representations if the similarity is great enough between your protein and one in which the structure has been solved through experimental means. Automated homology modeling is even available through the Web at Amos Bairoch’s Expasy server in Switzerland (http://www.expasy.ch/swissmod/SWISS-MODEL.html).

In the sample dataset that we are using we have a perfect example of being able to use structural inference. The *E. coli* sequence has been crystallized and, therefore, is completely annotated with three-dimensional structural features. Be sure that your display is still set to either feature coloring or graphic features and then select the entirety of one of the larger helices by double clicking it and then selecting its name in the “Sequence Features” window; close the window afterwards. Now go to the “Functions,” “Protein Analysis,” “Helical Wheel . . .” menu to launch the HelicalWheel program on that particular helix. Press “Run” in the program window and the standard “wheel” type representation of the helix will appear momentarily, as shown below:

![Helical Wheel Image](image)

The most amphiphilic (sometimes known as amphipathic because of their strong tendency to be highly antigenic for T-cells) helix (i.e. the helix with the highest hydrophobic moment, in other words, the helix with the greatest partitioning of hydrophobic and hydrophilic residues from one face to the other) that I could find, was the one from residue 143 through 160. What do you think? Several other structural programs are available under the “Protein Analysis” menu, but I encourage you to proceed with the tutorial at this point. If
you do run some of these other GCG “Protein Analysis” programs in the future, especially PepPlot and PeptideStructure/PlotStructure, be forewarned that they use very old and unreliable secondary structure prediction algorithms. They are NOT to be trusted as secondary structure predictors. However, they do plot several other very worthwhile and reliable attributes such as hydrophobicity and hydrophobic moments.

One of the more important things to realize is many of these types of algorithms are based on soluble, globular proteins. Using the same parameters with all types of proteins is not at all appropriate. Therefore, when dealing with other types of proteins, especially membrane-associated or membrane-spanning proteins, you should alter the default parameters appropriately. The simplest parameter to change is often the window size. It should be set approximately to the size of the feature being analyzed (e.g., use a window size of about 21 when trying to find membrane-spanning alpha helices).

Many, many features have been described and catalogued in protein sequences over the years. Most of these have recognizable consensus patterns that allow you to screen an unknown sequence for their occurrence. In many cases this can be a tremendous aid in ascertaining the function of an unknown peptide sequence. This database of catalogued consensus patterns is called PROSITE. The GCG program Motifs performs a search against this database. Motifs searches for recognized structural, regulatory and enzymatic consensus sequences in the PROSITE Dictionary of Protein Sites and Patterns (Bairoch, 1992). The program can tolerate mismatches with a mismatch option and it displays an abstract with selected references for each motif signature found.

Select the *E. coli* sequence entry name and then go back to the “Functions,” “Protein Analysis” menu. Pick “Motifs . . .” to launch the program. A “Which selection” window may pop up asking if you want to use the “selected sequences” or “selected region”; choose “selected sequences” to run the program on the full length of the *E. coli* EF-Tu protein. The “Motifs” program window will then display; press “Run” to screen the sequence without any options. Carefully look over the text file that is displayed. Notice the sites that have been characterized in this sequence and the extensive bibliography associated with them:

```plaintext
MOTIFS from: /users/thompson/.seqlab-mendel/input_20.rsf{EFTU_ECOLI_1}
Mismatches: 0                May 3, 1999 20:18 ..
```

```plaintext
input_20.rsf{EFTU_ECOLI_1}  Check: 1399  Length: 393   P02990 escherichia coli
 . elongation factor tu (ef-tu) (p-43). 11/97
```

```
Atp_Gtp_A          (A,G)x4GK(S,T)
                   (G)x{4}GK(T)
18: NVGTI      GHVDHGTK     TLTAA

*****************************************************************************
* ATP/GTP-binding site motif A (P-loop) *
*****************************************************************************
```

From sequence comparisons and crystallographic data analysis it has been shown [1,2,3,4,5,6] that an appreciable proportion of proteins that bind ATP or GTP
share a number of more or less conserved sequence motifs. The best conserved of these motifs is a glycine-rich region, which typically forms a flexible loop between a beta-strand and an alpha-helix. This loop interacts with one of the phosphate groups of the nucleotide. This sequence motif is generally referred to as the 'A' consensus sequence [1] or the 'P-loop' [5].

There are numerous ATP- or GTP-binding proteins in which the P-loop is found. We list below a number of protein families for which the relevance of the presence of such motif has been noted:

- ATP synthase alpha and beta subunits (see <PDOC00137>).
- Myosin heavy chains.
- Kinesin heavy chains and kinesin-like proteins (see <PDOC00343>).
- Dynamins and dynamin-like proteins (see <PDOC00362>).
- Guanylate kinase (see <PDOC00670>).
- Thymidylate kinase.
- Thymidylate kinase.
- Shikimate kinase (see <PDOC00868>).
- Nitrogenase iron protein family (nifH/frxC) (see <PDOC00580>).
- ATP-binding proteins involved in 'active transport' (ABC transporters) [7] (see <PDOC00185>).
- DNA and RNA helicases [8,9,10].
- GTP-binding elongation factors (EF-Tu, EF-1alpha, EF-G, EF-2, etc.).
- Ras family of GTP-binding proteins (Ras, Rho, Rab, Ral, Ypt1, SEC4, etc.).
- Nuclear protein ran (see <PDOC00859>).
- ADP-ribosylation factors family (see <PDOC00781>).
- Bacterial dnaA protein (see <PDOC00771>).
- Bacterial recA protein (see <PDOC00131>).
- Bacterial recF protein (see <PDOC00539>).
- Guanine nucleotide-binding proteins alpha subunits (Gi, Gs, Gt, G0, etc.).
- DNA mismatch repair proteins mutS family (See <PDOC00388>).
- Bacterial type II secretion system protein E (see <PDOC00567>).

Not all ATP- or GTP-binding proteins are picked-up by this motif. A number of proteins escape detection because the structure of their ATP-binding site is completely different from that of the P-loop. Examples of such proteins are the E1-E2 ATPases or the glycolytic kinases. In other ATP- or GTP-binding proteins the flexible loop exists in a slightly different form; this is the case for tubulins or protein kinases. A special mention must be reserved for adenylate kinase, in which there is a single deviation from the P-loop pattern: in the last position Gly is found instead of Ser or Thr.

-Consensus pattern: [AG]-x(4)-G-K-[ST]

- Other sequence(s) detected in SWISS-PROT: in addition to the proteins listed above, the 'A' motif is also found in a number of other proteins. Most of these proteins probably bind a nucleotide, but others are definitively not ATP- or GTP-binding (as for example chymotrypsin, or human ferritin light chain).

-Expert(s) to contact by email: Koonin E.V.
  koonin@ncbi.nlm.nih.gov

-Last update: November 1997 / Text revised.

  EMBO J. 1:945–951(1982).
Efactor_Gtp

\text{K)(S,T)(I,V)x2(G,S,T,A,C,K,R,N,Q} \\
\text{D(N)x3E(K)x(R)(G)(I)(T)(I)x2(S)} \\
\text{50: AFDQI DNAPEEXARGINTS HVEYD} \]

* GTP-binding elongation factors signature *

Elongation factors [1,2] are proteins catalyzing the elongation of peptide chains in protein biosynthesis. In both prokaryotes and eukaryotes, there are three distinct types of elongation factors, as described in the following table:

<table>
<thead>
<tr>
<th>Eukaryotes</th>
<th>Prokaryotes</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>EF-1alpha</td>
<td>EF-Tu</td>
<td>Binds GTP and an aminoacyl-tRNA; delivers the latter to the A site of ribosomes.</td>
</tr>
<tr>
<td>EF-1beta</td>
<td>EF-Ts</td>
<td>Interacts with EF-1a/EF-Tu to displace GDP and thus allows the regeneration of GTP-EF-1a.</td>
</tr>
<tr>
<td>EF-2</td>
<td>EF-G</td>
<td>Binds GTP and peptidyl-tRNA and translocates the latter from the A site to the P site.</td>
</tr>
</tbody>
</table>

The GTP-binding elongation factor family also includes the following proteins:

- Eukaryotic peptide chain release factor GTP-binding subunits [3]. These proteins interact with release factors that bind to ribosomes that have encountered a stop codon at their decoding site and help them to induce release of the nascent polypeptide. The yeast protein was known as SUP2 (and also as SUP35, SUF12 or GST1) and the human homolog as GST1-Hs.
- Prokaryotic peptide chain release factor 3 (RF-3) (gene prfC). RF-3 is a class-II RF, a GTP-binding protein that interacts with class I RFs (see <PDOC00607>) and enhance their activity [4].
- Prokaryotic GTP-binding protein lepA and its homolog in yeast (gene GUF1) and in Caenorhabditis elegans (ZK1236.1).
- Yeast HBS1 [5].
- Rat statin S1 [6], a protein of unknown function which is highly similar to EF-1alpha.
- Prokaryotic selenocysteine-specific elongation factor selB [7], which seems to replace EF-Tu for the insertion of selenocysteine directed by the UGA codon.
- The tetracycline resistance proteins tetM/tetO [8,9] from various bacteria such as Campylobacter jejuni, Enterococcus faecalis, Streptococcus mutans and Ureaplasma urealyticum. Tetracycline binds to the prokaryotic ribosomal 30S subunit and inhibits binding of aminoacyl-tRNAs. These proteins abolish the inhibitory effect of tetracycline on protein synthesis.
- Rhizobium nodulation protein nodQ [10].

In EF-1-alpha, a specific region has been shown [12] to be involved in a conformational change mediated by the hydrolysis of GTP to GDP. This region is conserved in both EF-1alpha/EF-Tu as well as EF-2/EF-G and thus seems typical for GTP-dependent proteins which bind non-initiator tRNAs to the ribosome. The pattern we developed for this family of proteins include that conserved region.

Consensus pattern: D-[KRSTGANQFYW]-x(3)-E-[KRAQ]-x-[RKQD]-[GC]-[IVHK]-[ST]-[IV]-x(2)-[GSTACKRNQ]

Sequences known to belong to this class detected by the pattern: ALL, except for 11 sequences.
Other sequence(s) detected in SWISS-PROT: NONE.
Last update: November 1997 / Text revised.

4) Performing the alignment — the PileUp program

Click the “Close” box and return your display to “1:1” and “Residue Coloring.” Take a look at each of the members of the list. Quickly double click on various entries’ names to see the database reference descriptions for them (or click on the “INFO” button). (This is the same information that you can get with the GCG command “typedata -ref” at the command line.) Next select all of the entries in the list through the “Edit” menu “Select All” command (or by dragging the mouse through all the entries or by shift-clicking the bottom and top entry [select nonadjacent entries with Cntrl-clicks, except on Linux use Cntrl-Rtclick]). Once all of your sequences are selected, go to the “Functions” menu and select “Multiple comparison.” Click on “PileUp...” to align the entries. A new window will be produced with the parameters for running PileUp. Be sure that the “How:” box says “Background Job.” For this first pass accept all of the program defaults by merely pressing the “Run” button and the window will go away. The program will first compare every sequence with every other one. This is the pairwise nature of the program, and then it will progressively merge them into an alignment in the order of determined similarity, from most to least. The window will close and then, after a few moments, depending on the complexity of the alignment and the load on the server, new output windows will automatically display. The top window will be the Multiple Sequence Format (MSF) output from your PileUp run. Notice the BLOSUM62 matrix and gap introduction and extension penalties used by default. In most cases these work just fine though they can be changed if desired (the BLOSUM30 matrix can be very helpful for aligning quite divergent sequences). Scroll through your alignment to check it out and then “Close” the window afterwards. An abridged output file from my example follows below. Notice the interleaved character of the sequences, yet they all have unique identities, addressable by using their MSF filename together with their own name in braces, {name}:

```plaintext
!!AA_MULTIPLE_ALIGNMENT 1.0
PileUp of: @/users/thompson/.seqlab-mendel/pileup_1.list

Symbol comparison table: GenRunData:blosum62.cmp  CompCheck: 6430

GapWeight: 12
GapLengthWeight: 4

pileup_1.msf  MSF: 483  Type: P  April 29, 1999 17:22  Check: 9074 ..

Name: eftu_anani       Len:   483  Check: 7317  Weight:  1.00
Name: eftu_theaq       Len:   483  Check: 2028  Weight:  1.00
Name: eftu_ecoli       Len:   483  Check: 8483  Weight:  1.00
Name: eftu_myctu       Len:   483  Check: 7189  Weight:  1.00
Name: eftu_myctu       Len:   483  Check: 9514  Weight:  1.00
Name: eftu_theaq       Len:   483  Check:  426  Weight:  1.00
Name: eftu_chltr       Len:   483  Check:  545  Weight:  1.00
Name: ef10_xenla       Len:   483  Check: 7491  Weight:  1.00
Name: ef11_human       Len:   483  Check: 7663  Weight:  1.00
Name: ef11_drome       Len:   483  Check: 3601  Weight:  1.00
Name: efla_oncvo       Len:   483  Check: 9453  Weight:  1.00
Name: efla_yeast       Len:   483  Check: 3241  Weight:  1.00
Name: efla_arath       Len:   483  Check: 4009  Weight:  1.00
Name: efla_wheat       Len:   483  Check: 5710  Weight:  1.00
Name: efla_wheat       Len:   483  Check: 2391  Weight:  1.00
```

An abridged output file from my example follows below. Notice the interleaved character of the sequences, yet they all have unique identities, addressable by using their MSF filename together with their own name in braces, {name}:
//

1 eftu_anani MARAKFERTK PHANIGTIGH VDHGKTTLTA AITTVLAKAG .MA.KARAYA
eftu_theaq ~AKGEFIRTK PHNVNGTIGH VDHGKTTLTA ALTYVAAAAEN PNV.EVKDYG
eftu_ecoli ~SKEKFERTK PHNVNGTIGH VDHGKTTLTA AITTVLAKTY G..GAARAFD
eftu_myctu MARAKFQRTK PHNVNGTIGH VDHGKTTLTA AITTVLAKAR PDLNETKAFD
eftu_mygca MAKERFRSRK PHNVNGTIGH VDHGKTTLTA AITCVLASK.. AGTSEAKKYD
eftu_thema MAKEKFRRTK PHNVNGTIGH VDHGKTTLTA AITYKLSLK ..VLAQYIPYD
eftu_chltr ~SKETFQRNK PHNIGTIGH VDHGKTTLTA AITTVLAKTY G..GAARAFD
ef10_xenla ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef11_human ~~~~MGKEK HINIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef11_drome ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_oncvo ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_yeast ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_arath ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_wheat ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_euggr ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_enthi ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_tetpy ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_dicdi ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_plafk ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_giala ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_theac ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_metva ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_halma ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_pyrwo ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE
ef1a_sulso ~~~~MGKEK TKHLIVVIGH VDSGKSTTTG HLIYKCGGID KRTIEKFEKE

51
eftu_anani D.............. IDAAPE ERARGITINT AHVEYETGNR HYAHVDCEPH
eftu_theaq D.............. IDAAPE ERARGITINT AHVEYETGNR HYAHVDCEPH
eftu_ecoli Q............... IDAAPE ERARGITINT AHVEYETGNR HYAHVDCEPH
eftu_myctu Q............... IDAAPE ERARGITINT AHVEYETGNR HYAHVDCEPH
eftu_mygca E............... IDAAPE ERARGITINT AHVEYETGNR HYAHVDCEPH
eftu_thema Q............... IDAAPE ERARGITINT AHVEYETGNR HYAHVDCEPH
eftu_chltr S............... IDNTPE ERARGITINT AHVEYETGNR HYAHVDCEPH
ef10_xenla AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef11_human AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef11_drome AQEMGKGSFK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_oncvo AQEMGKGSFK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_yeast AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_arath AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_wheat AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_euggr AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_enthi AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_tetpy AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_dicdi AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_plafk AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_giala AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_theac AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_metva AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_halma AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_pyrwo AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH
ef1a_sulso AAEKGKSFHK YAWVLKDLKA ERERGTDI ALWKFETSKY YFIIIADPGH

//
Notice the listing of sequence names near the top of the file. This listing contains an important number called the checksum. All GCG sequence programs utilize this number as a unique sequence corruption identifier.
There is a checksum line for the whole alignment as well as individual checksum lines for each member of the alignment. If any two of the checksum numbers are the same, then those sequences are identical. If they are, an editor can be used to place an exclamation point, “!” at the start of the checksum line in which the duplicate sequence occurs. Exclamation points are interpreted by GCG as remark delineators; therefore, the duplicate sequence will be ignored in subsequent programs. Or the sequence could be “CUT” from the alignment with the SeqLab Editor. Another important number on the individual checksum lines should be pointed out. The “Weight” designation determines how much importance each sequence contributes to a profile made of the alignment. Sometimes it is worthwhile to adjust these values so that the contribution of a collection of very similar sequences does not overwhelm the signal from a few more divergent sequences. In the SeqLab interface the “Sequence Info . . .” window can be used to accomplish this. However, we will not be bothering with it here.

After scrolling through your alignment and then “Close”ing its window, the next window visible will be the “SeqLab Output Manager.” This is a very important window and will contain all of the output from your current SeqLab session. Files may be displayed, printed, saved in other locations with other names, and deleted from this window. **We need to use an extremely important function at this point:** press the “Add to Editor” button and specify “Overwrite old with new” in the next window when prompted, to take your MSF output and merge it with the RSF (Rich Sequence Format: the alignment as well as all reference feature information) file in the open editor. This will keep all feature information intact, inserting gaps as needed, renumbering all reference locations. “Close” the “Output Manager” after loading your new alignment. The next window will contain PileUp’s cluster dendrogram; in my example’s case, the following:
This dendrogram of the similarity clustering relationships between the sequences is automatically created when you run PileUp. It shows the clustering process used to create the alignment. The length of the vertical lines is proportional to the difference in similarity between the sequences. This is not an evolutionary or phylogenetic tree and it should not be presented as one. (Although, if the rates of evolution for each lineage are exactly the same, which is seldom the case in nature, it could be the same as one.) No substitution models for multiple hits or methods for correction of unequal rates of divergence are used in its construction. It merely indicates the relative similarity of the sequences. However, the dendrogram can assist in determining sequence weighting factors to even out each sequences’ contribution to a profile.

If desired, you can directly print from this window to a PostScript file by picking “Print...” Be sure that the “Output Device:” chosen is “[Encapsulated] PostScript File.” You can rename the output file to anything that you may want in this window; click “Proceed” to create the EPSF output in your current directory. To actually print this file you may need to ftp it to a local machine attached to a PostScript savvy printer unless you have direct access to a UNIX sytem printer and it is PostScript compatible. (All Macintosh compatible laser printers run PostScript by default. Carefully check any laser printer connected to a Wintel system to be sure that it is PostScript compatible.) “Close” the dendrogram window to return to the editor. Notice that your residues now align by color. My editor display looks like the following after loading the MSF file:

5) Visualizing conservation in multiple sequence alignments.

The most conserved portions of an alignment are those most resistant to evolutionary change, often due to some type of structural constraint. We can use the GCG graphics program PlotSimilarity to visualize these most conserved portions of a multiple sequence alignment. This is also a very nice way to see those areas of
an alignment that may need improving by pointing out the most variable regions. The program draws a graph of the running average similarity along a group of aligned sequences (or of a profile with the -Profile option).

Be sure that all of the sequence names are selected and then go back to the “Functions” menu and under the “Multiple comparison” section choose “PlotSimilarity . . .”. We need to change some of the program defaults there so choose “Options . . .”. Check “Save SeqLab colormask to” and “Scale the plot between:” the “minimum and maximum values calculated from the alignment.” (The first option’s output file will be used in the next step and the second specification launches the program’s command line -Expand option which blows up the plot, scaling it between the maximum and minimum similarity values observed so that the entire graph is used rather than just the portion of the Y axis that your alignment happens to occupy.) The Y-axis of the resulting plot will use the similarity values from whichever symbol comparison matrix was used to create your alignment or you can specify an alternative. The default matrix, BLOSUM62, begins its identity value at 4 and ranges up to 11; mismatches go as low as -4. “Close” the window; notice that the “Command Line:” box now reflects your updated options. Click the “Run” box to launch the program. The output will quickly return. “Close” the plotsimilarity.cmask display and the “Output Manager” and then take a look at the similarity plot. My example follows below:

![Plot of similarity](image.png)

Make a PostScript file of this plot too, if desired. You can directly print from this window to a PostScript file by picking “Print . . .”. Just as before, be sure that the “Output Device:” chosen is “[Encapsulated] PostScript
File.” You can rename the output file to anything that you may want in this window; click “Proceed” to create the EPSF output file in your current directory and then “Close” the window. Regardless of whether you print this plot or not, take notes of where the similarity significantly falls off within and at the beginning and end of the alignment; in my example above, a region around 220, 300, and about the last 25 residues or so. Now go to the “File” menu and click on “Open Color Mask Files.” This will produce another window from which you should select your new “plotsimilarity.cmask” file; click on “Add” and “Close” the window. This will produce a gray scale overlay on your sequences that describes their regional similarity where the darker the gray is corresponds to higher similarity values. My sample alignment, using a zoom factor of 8 to 1, looks like the following. Notice the strong conservation peak centered around residue 100 in the alignment shown on the following page, one of EF-1α/Tu’s GTP binding regions:

![Alignment with similarity overlay]

6) Improving alignments within SeqLab.

The beauty of this representation is you can now select only those regions of low similarity and try to improve their alignment automatically. This is possible because of PileUp’s -InSitu option. Be sure that all of your sequences are selected and then zoom back in your alignment to 1:1 so that you can see individual residues
and then scroll to the end. It’s best to start at the carboxy termini in this process so that the positions of the low similarity regions do not become skewed as you proceed through the procedure. Now select a region of low similarity, either by using the mouse or by using the “Edit” “Select Range” function (determine the positions by placing your cursor at the beginning and end of the range to be selected and noting the column number). Once all of your sequences and the region that you wish to improve are selected, go to the “Functions” menu and again select “Multiple comparison.” Click on “PileUp . . .” to realign all of the sequences within that region. (The “Windows” menu also contains a listing of all of the programs that you have used in the current session; you can launch any of them from there as well.) You will be asked whether you want to use the “Selected sequences” or “Selected region;” it is very important to specify “Selected region.” This will produce a new window with the parameters for running PileUp. Next, be sure to click on “Options . . .” to change the way that PileUp will perform the alignment. In the “Options” window check the gap creation and extension boxes and change their respective values to much less than the default. Changing them to 3 and 1 respectively works well for me in this step. Most importantly, check “Realign a portion of an existing alignment;” this calls up the command line -InSitu option. Otherwise only that portion of your alignment selected will be retained in the output. Furthermore, we really don’t need another similarity dendrogram, so uncheck the “Plot dendrogram” box. “Close” the window and notice the new options in the PileUp “Command Line:” “Run” the program to improve your alignment. The window will go away and your results will return very quickly since you are only realigning a portion of the alignment; new output windows will automatically display. The top window will be the MSF output from your PileUp run. Notice the BLOSUM62 matrix used by default (others can be specified in the options menu) and the lowered gap introduction and extension penalties of 3 and 1 respectively. Scroll through your alignment to check it out and then “Close” the window. The next window will be the “Output Manager.” Just like before, click on “Add to Editor” and then specify “Overwrite old with new” in the new “Reloading Same Sequences” window to merge the new alignment with the old one and retain all feature information. This feature information may help guide your alignment efforts in subsequent steps. “Close” the “Output Manager” window after loading your new alignment.

Your alignment should now be a bit better within the specified region. Repeat this process in all areas of low similarity, again, working from the carboxy termini toward the amino end. Notice that all of the options that you last specified are retained by the program so you don’t need to reapply them. You can also save these run parameters so that they will come up in subsequent sessions by clicking on the “Save Settings” box in any of the program run windows. You may want to go to the “File” menu periodically to save your work using the “Save as . . .” function in case of a computer or network problem. It’s also probably a good idea to reperform the PlotSimilarity and color mask procedure after going through the entire alignment to see how things have improved after you’ve finished the various InSitu PileUps. If you discover an area that you can not improve through this automated procedure, then it is time to either manually ‘correct’ it or ‘throw it away.’ Again, note those ‘problem’ areas and then switch back to “Residue Coloring.” This will ease manual alignment by allowing your eyes to work with columns of color.

Other things that can help manual alignment are “GROUP”ing and “Protections.” The “GROUP” function allows you to manipulate ‘families’ of sequences as a whole — any change in one will be propagated
throughout them all. To “GROUP” sequences, select those that you want to behave collectively and then click on the “GROUP” icon right above your alignment. You can have as many groups as you want. The space bar will introduce a gap into the sequence and the delete key will take a gap away. However, you can not delete a sequence residue without changing that sequence’s (or the entire alignment’s) “Protections.” Click on the padlock icon to produce a “Protections” window. Notice that the default protection allows you to modify “Gap Characters” and “Reversals” only. Check “All other characters” to allow you to “Cut” regions out of your alignment and/or delete individual residues and then click “OK” to close the window. A very powerful manual alignment function can be thought of as the ‘abacus’ function. To take advantage of this function select the region that you want to slide and then press the shift key as you move the region with the right or left arrow key. You can slide residues greater distances by prefacing the command keystrokes with the number of spaces that you want them to slide.

Make subjective decisions regarding your alignment. Is it good enough; do things line up the way that they should? If, after all else, you decide that you just can’t align some region, or even an entire sequence, then perhaps get rid of it with the “Cut” function. Cutting out an entire sequence may leave some columns of gaps in your alignment. If this is the case, then reselect all of your sequences and go to the “Edit” menu and select “Remove Gaps . . .” “Columns of gaps.” Another alternative is the mask function that I will describe below. Notice the extreme amino and carboxy ends of the alignment. Amino and carboxy termini seldom align properly and are often jagged and uncertain. This is common in multiple sequence alignments and subsequent analyses should probably not include these regions. If loading sequences from a FastA or BLAST run, allowing SeqLab to trim the ends automatically based on beginning and ending constraints considerably improves this situation. Overall, things to look for include strongly conserved residues such as tryptophans, cysteines, and histidines, important structural amino acids such as prolines, tyrosines and phenylanines, and the conserved isoleucine, leucine, valine triumvirate; make sure they all align. After you have finished tweaking, evaluating, and readjusting your alignment to make it as ‘satisfying’ as possible, change back to “Feature Coloring” “Display.” Those features that are annotated should now align perfectly.

This is another way to assure that your alignment is as biologically ‘correct’ as possible. Everything you do from this point on, and especially later if you use alignments to ascertain molecular evolution, is absolutely dependent on the quality of the alignment! You need a very clean, unambiguous alignment that you can have a very high confidence in — truly a biologically meaningful alignment. Each column of symbols must actually contain homologous characters.

Sometimes you may want to align DNA sequences along with their corresponding proteins (the “Group” function is very helpful for this) in order to perform phylogenetic analyses on the DNA rather than on the proteins. This is especially important when dealing with datasets that are quite similar since the proteins may not reflect many differences hidden in the DNA. Furthermore, many people prefer to run phylogenetic analyses on DNA rather than protein regardless of how similar they are — the multiple substitution models are much more robust for DNA.

The logic to this paired protein and DNA alignment approach is as follows:
1) The easy case where you can align the DNA directly. If the DNA sequences are directly alignable because they are quite similar, then merely create your DNA alignment. Next use the “Edit” menu “Translate” function and the “align translations” option to create aligned corresponding protein sequences. Select the region to translate based on the CDS reference in each DNA sequence’s annotation. Be careful of CDS entries that do not begin at position 1—the GenBank CDS feature annotation “/codon_start=” identifies which position the translation begins within the first codon listed. You may also have to trim sequences down to just the relevant gene, especially if they’re genomic. You’ll have to change their protections with the padlock icon if this is the case. Group each protein to its corresponding DNA sequence so that subsequent manipulations will keep them together.

2) The way more difficult case where you need to use the protein sequences to create the alignment because the DNA is not directly alignable. In this case you need to load the protein sequences first, create their alignment, and then load their corresponding DNA sequences. You can find the DNA sequence accession codes in the annotation of the protein sequence entries. Next translate the unaligned DNA sequences into new protein sequences with the Edit-Translate function using the “align translations” option and Group these to their corresponding DNA sequences, just as above. However, this time the DNA along with their translated sequences are not aligned as a set, just the other protein set is aligned. Also, Group all of the aligned protein dataset together, separately from the DNA/aligned translation set. Now comes the manual part; painstakingly rearrange your display to place the DNA, its aligned translation, and the original aligned protein sequence side-by-side and then manually slide one set to match the other. Use the “CUT” and “PASTE” buttons to move the sequences around. When pasting realize that the “Sequence clipboard” contains complete sequence entries, whereas the “Text clipboard” only contains sequence data, amino acid residues or DNA bases as the case may be. The translated sequence entries can be “CUT” away after they’re aligned to the rest of the set. Merge the newly aligned sequences into the existing alignment Group as you go and then start on the next one. It sounds difficult, but since you’re matching up two identical protein sequences, the DNA translation and the original aligned protein, it’s really not too bad. The Group function keeps everything together the way it should be so that you don’t lose your original alignment as you space residues apart to match them up to their respective codons. Some codons may become spaced apart in this process and will have to be adjusted afterwards. As usual, save your work often.

Many other alignment editors are available for cleaning up multiple sequence alignments. However, I think that you will find SeqLab most satisfying, and only using a GCG compatible editor assures that the format will not be corrupted. If you do make any changes to a GCG sequence data file with a non-GCG compatible editor, you must reformat the alignment afterwards. However, reformating MSF (or RSF with the -RSF option) files requires a couple of tricks. If this step is not done exactly correct, you will get very strange results. If you do need to do this for any reason, you must use the appropriate Reformat option (-MSF or -RSF) and you must specify all the sequences within the file, i.e. “{*},” for example:

```bash
> reformat -msf your_favorite.msf{*}
```
Here you will not need to perform this step, unless for some perverse reason you decided to edit your alignment with a non-GCG compliant editor such as pico; however, it may prove necessary in other situations. After reformatting, the new MSF or RSF file will follow GCG convention, with updated format, numbering, and checksums.

7) Masking and export format issues.

Consensus methods are another powerful way to visualize similarity within an alignment besides GCG’s PlotSimilarity program. The SeqLab “Edit” menu allows you to easily create several types of consensus. In addition to standard consensus sequences using various similarity schemes, SeqLab also allows you to create consensus “Masks” that screen specified areas of your alignment from further analyses by specifying 0 or 1 weights for each column. Masks can be created manually also through the “New Sequences” menu and can have values all the way through 9. Masking can be very helpful for phylogenetic analysis by excluding those less reliable columns in your alignment where you are not confident in the positional homology. At this point be sure all of your sequences are selected and then create a Mask style sequence consensus of them by going to the “Edit” “Consensus . . .” menu and specifying “Consensus type:” “Mask Sequence.” The default mode is to create an identity consensus at the 2/3'rd plurality level (“Percent required for majority”) with a threshold of 5 (“Minimum score that represents a match”); however, these are a very high values for phylogenetic analysis and would likely not leave much phylogenetically informative data. Therefore, experiment with different lower plurality and threshold values as well as different scoring comparison matrices to see the difference that it can make in the appearance of your alignment. Be sure that “Shade based on similarity to consensus” is checked to generate a color mask overlay on the display to help in the visualization process. (If making a normal sequence consensus rather than a weight mask, you can generate a gray intermediate similarity color as well as the black and white representation. This is a nice way to prepare alignment figures for publication.) The following screen illustrates my example using the BLOSUM62 matrix, a plurality of 15%, and a threshold cutoff value of 3:
Few areas are excluded by the Mask in this alignment because of the large similarity of this group of sequences. This is as it should be for excluding many more columns in this particular alignment would likely just leave all identical sequences and it would be impossible to ascertain how they are related. In fact, as described above, when dealing with sequences this similar, it may be best to align the DNA sequences along with their corresponding proteins and then perform the phylogenetic analyses on the DNA rather than on the proteins. Just like most computational molecular biology techniques, one is always balancing signal against noise — and it can be quite the balancing act! Too much noise or too little signal both degrade the analysis to the point of nonsense.

Once a Mask has been created in SeqLab any of the programs available through the “Functions” menu will use that Mask, if the Mask is selected along with the desired sequences, to weight the columns of the alignment data matrix appropriately. This only occurs through the “Functions” menu.

When you’ve found a combination that you like, you may wish to go to the “File” “Print . . .” command and change “Output Format:” to “PostScript” in order to prepare a PostScript file of your SeqLab display. Play around with the other parameters as you like — notice that as you change the font size the number of pages to be printed varies. In the “Print Alignment” menu specify “Destination . . . File” and give it an appropriate filename and then click “OK.” This should result in a PostScript file of the alignment using the displayed coloring and the specified parameters to be created in the directory where you launched SeqLab. This file can then be transferred to another machine for color PostScript printing, or for importing into PostScript savvy
programs for further manipulation, or it can be printed to a black and white laser printer that will simulate the
colors with gray tones. Unfortunately the format of this ‘raw’ PostScript file is different enough from a
standard Encapsulated PostScript file that you may have to use a different print queue in many instances.
Discuss these matters with your system administrator. It may require some variation of the following type of command:

\[ \text{lpr -PPostScript_queue seqlab_alignment.ps} \]

Return to the “SeqLab Main Window” and go to the “File” “Export” menu; click “Format” in the new window
and notice that “MSF,” “GenBank,” and “GDE2.2” are all available for saving a copy of your RSF file in some
alternative formats. At this point do not export any of these formats and “Cancel” the window. Be sure to
realize that using this export route does not use the Mask data to include or exclude columns from your
alignment. Since we want to take advantage of the Mask data for any subsequent phylogenetic analyses, we
will export our alignment using another method. Therefore, after being sure that all of your sequences as well
as your Mask are selected, go to the “Functions” menu, where all choices will be affected by the Mask, and
choose “Importing/Exporting” “ToFastA . . .” No options are required here; just press “Run” to convert your
alignment into FastA format. We will use FastA as a good intermediate format on the way to PHYLIP’s
required format. The new file will be displayed by SeqLab. The first part of it is shown below:

\[ \text{>EFTU_ECOLI In situ PileUp of: @/export/home/gcg/stevet/.seqlab-snap/pileup_16.list} \]
\[ \text{SKEKFERTKPHVNVGTIGVHDHGKTLT-------AAIT---TV--LA-KTYGGA---ARA} \]
\[ \text{F-DOIDNEEKARGINTSHVEYDPTRHAYVDPCFHDYVKNMIGAAQMDAILV} \]
\[ \text{VAATDG---P-------QTREHLLLQRGVQVPIYIFLKNCDMV---D--DEEELVEMEV} \]
\[ \text{RELSQYDFGDDD-----TPIV---RGSAKALE-----DAEAWEA--KIE-----LAGFLDS} \]
\[ \text{Y-IPEPERAIKPFLLPIEDVFSISGRGVTGVRGERRIYKVEEVEI---VGTIQKST-C} \]
\[ \text{TGVMFRLDLDEGAVENVVLRGIKREEIERQVL-----AKPTIKHITKEFESEVYL} \]
\[ \text{SKDEGGRRHTP---FFKGYRP--------------QF---YFRRTD--VTG-TILPEG--VEMVMP-} \]
\[ \text{GDNIVVTLI---HPIAMDDD-------------GRFAIREGRTVGAVVAKV------------------------} \]
\[ \text{-LS} \]
\[ \text{>EFTU_MYCTU In situ PileUp of: @/export/home/gcg/stevet/.seqlab-snap/pileup_16.list} \]
\[ \text{MAKAKFQRTKPHVNHTIGVHDHGKTLT-------AAIT---KV--LHDKFPLKE--TK} \]
\[ \text{AF-DOIDNAEERQGITINIAHVROYTDKRHYAHDAPFHDYIAKNMIGHAAQMDAILV} \]
\[ \text{VVAATDG---P-------QTREHLLRARQGVPIYILVALNKADAV---D--DEEELLVEME} \]

Notice that it excludes those positions that were masked with zero and that it now follows all FastA format
conventions including the automatic conversion of all GCG style gap periods and tildes to the more universal
gap dash representation. This step, therefore, circumvents the common ‘dot to dash’ problem often
encountered in sequence format conversion. “Close” the ToFastA output window. You may want to use the
“Output Manager” to save the file under a name that makes more sense to you through the “Save As . . .”
menu. We can next use ReadSeq to convert this FastA format file to PHYLIP compatible format.

Temporarily switch to your terminal window behind SeqLab to run Don Gilbert’s program ReadSeq that can
be used to change your FastA format file into something acceptable for PHYLIP use. A limitation of ReadSeq
is it does not allow you to only choose a portion of an alignment, nor does it automatically convert dots and
tildes to hyphens. However, since we have taken care of these points while in SeqLab, it’ll work just fine for us
ReadSeq runs a bit backward from what most people are used to. Begin the program by typing `readseq`. It first prompts you for an appropriate output file name, not an input file. Do not make a mistake in this step by giving the name of your input file first. If you do, you will overwrite the input file while running the program and then when it tries to read it there will be nothing left to read! Next choose “12” off of the ReadSeq menu for the current PHYLIP format and then designate the input sequence file name. (Do not use the GCG `{*}` designator if converting MSF files; this is not a GCG program.) Finally, after the program has read all of the input sequences, specify “All” the sequences by typing the word “all.” When the program again asks for an input sequence, press return, and let it do its thing. A sample screen trace is shown below; as usual, responses are shown in bold:

```bash
> readseq
readSeq (1Feb93), multi-format molbio sequence reader.

Name of output file (?=help, defaults to display):
EF1a-Tu.phy
  1. IG/Stanford           10. Olsen (in-only)
  2. GenBank/GB            11. Phylip3.2
  3. NBRF                  12. Phylip
  4. EMBL                  13. Plain/Raw
  5. GCG                   14. PIR/CODATA
  6. DNAStrider            15. MSF
  7. Fitch                 16. ASN.1
  8. Pearson/Fasta         17. PAUP/NEXUS
  9. Zuker (in-only)       18. Pretty (out-only)

Choose an output format (name or #):
12

Name an input sequence or --option:
EF1a-Tu.tfa

Sequences in EF1a-Tu.tfa (format is 8. Pearson/Fasta)
1) EF1U_ECOLI_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
2) EF1U_MYCTU_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
3) EF1U_THEAQ_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
4) EF1U_ANAN1_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
5) EF1U_THEMA_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
6) EF1U_CHLTR_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
7) EF1U_MYCGA_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
8) EF1A_ARATH_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
9) EF1A_WHEAT_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
10) EF1A_EUGGR_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
11) EF1X_XENLA_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
12) EF11_HUMAN_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
13) EF11_DROME_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
14) EF1A_ONCVO_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
15) EF1A_ENTHI_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
16) EF1A_PLAFF_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
17) EF1A_TETPY_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
18) EF1A_YEAST_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
19) EF1A_DICDI_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
20) EF1A_GTALA_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
21) EF1A_PYRWO_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
22) EF1A_THEAC_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
23) EF1A_SULSO_1 PileUp of: @/mole98/thompson/.seqlab-mole1/pileup_1.list
```
Choose a sequence (# or All):

all

This format requires equal length sequences.
Sequence truncated or padded to fit.
This format requires equal length sequences.
Sequence truncated or padded to fit.
This format requires equal length sequences.
Sequence truncated or padded to fit.
This format requires equal length sequences.
Sequence truncated or padded to fit.
This format requires equal length sequences.
Sequence truncated or padded to fit.

This format requires equal length sequences.
Sequence truncated or padded to fit.

This format requires equal length sequences.
Sequence truncated or padded to fit.
This format requires equal length sequences.
Sequence truncated or padded to fit.
This format requires equal length sequences.
Sequence truncated or padded to fit.

Name an input sequence or -option: <rtn>

Never mind if you get the “. . . padded to fit” error message — the program is just doing what it is supposed to do. Do realize, though, that had we not used ReadSeq on the output from ToFastA to convert to PHYLIP, and had rather used a GCG MSF file as input, then an essential change would have to be made before it would be correct for PHYLIP. As mentioned before, periods and tildes will not work to represent indels (gaps); they must all be changed to hyphens (dashes). The following, rather strange, UNIX command works very well for this step from the command line, but you should not need to use it in this exercise:

> tr \~\. \- < infile.phy > outfile.phy

Run more on your new file to see what PHYLIP format looks like:

> more EF1a-Tu.phy

25 483
EF10_XENLA M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF11_HUMAN M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF11_DROME M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_ONCVO M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_ARATH M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_MHAT M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_EUGGR M-----GKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_ENTHI M-----PKEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_TETFY M-----ARGDK VHIINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_DICDI M-----EFFSEK THINIVVIGH VDSGKSTTTG HLIYKCGGID -KRTIEKF
EF1A_PLAFK M-----GKEK THINLVVIGH VDGKSTTGG HIIYKLGID -RRTIEFEK
EF1A_GIALA -------------- ----------- ----STLTH LIYKCGGID- QRTIDYEEKR
EF1A_HALMA ------SDEQ HQLNAIIGHV DHGKSTLGR LLYETGSV-P EHVIEQHKEE
EF1A_METVA M-----AKTK PILNVAFIGH VDAGKSTTVG RLLLDGGAID PQLIV-RLRK
EF1A_THEAC M-----ASQK PHLNLITIGH VDGKSTLGV RLLYEHKGEI- PAHIIEEYEKR
EF1A_PYRWO M--MK--KPKDK PHVNIIVFIGH VDGKSTTIG RLLYDTGNI- PEQIIKKF-E
EF1A_SULSO M------SQK PHLNLIVIGH VDGKSTLGV RLLMDGFID EK--TVKAAEE

TYGGA--ARA F-DQIDNAPE EKARGITINT SHVEYDTPTTR HYAHVDCFGH
KEFPLNE-TK AF--DQIDNAP EERQRGITIN TAHVEYQTDF RHYAHVDAPG
NPN--VE-KD YGD-IDKAP EERQRTGITT HVAHYETKAK HYSVDCFGH
KA-GTSE-AK KY--DEIDAAP EERQKRGITIN TAHVEYATQIN RHYAHVDCFGH
KV--LQ--X YF--DQIDKAP EERQKRGITIN TAHVEYETTEK RHYAHVDCFGH
KA--GMNK--AR AYAD--IADAAP EERQKRGITIN TAHVEYETQNH RHYAHVDCFGH
GDGLAD--FRD YSS-IDNTPE EKARGITINA SHVEYETANR HYAHVDCFGH

Notice that the file begins with two numbers; the first shows the number of sequences in the matrix and the second is the length of the matrix including any gaps and ambiguities. The next section lists the names of the sequences truncated to ten characters, if necessary, along with all the sequences printed in an ‘interleaved’ fashion. Only the first sequence block lists the names, all others just give the sequence data itself.

Return to your SeqLab display to generate a NEXUS file for PAUP* (Swofford, 1989–2005). NEXUS format files are easily and reliably built from GCG alignments with GCG’s interface to the PAUP* package, the paired programs PAUPSearch and PAUPDisplay. These programs provide an easy to use access to a subset of PAUP* within GCG. However, the version of PAUP*, included in the Wisconsin Package version 9.1 through 10.3, either run in native mode or through the PAUPSearch and PAUPDisplay programs, is an old 4.0.0d55 version. And PAUP* is not included at all with the Linux version of GCG because of a license issue! (Although we’ve made the latest update work here for the Course, since Swofford is Course faculty.)

Therefore, for serious phylogenetic analysis, you may want to consider running PAUP* exterior to GCG by getting the latest version, which contains many bugs fixes and enhancements since 4.0.0d55, directly from Sinauer Associates, the publishing company that distributes the software (http://www.sinauer.com/), and installing it on your own computer or having it installed on your local biocomputing server. Regardless, if available, PAUPSearch can be a handy tool for generating NEXUS format files for any version of PAUP*.

Begin the NEXUS conversion process by again being sure all of your alignment sequences, including the weight Mask, are selected and then go to the “Functions” “Evolution” menu. Select “PaupSearch...” to launch the dialogue box. Because we merely want to generate a NEXUS file, we will run PAUPSearch in its fastest mode without actually performing a search. Accept the default “Tree Optimality Criterion” “maximum parsimony” and the “heuristic tree search (fast)” “Method for Obtaining Best Tree(s).” Be sure that the “perform bootstrap replications...” button is not pressed and then launch the “Options” menu by pressing the appropriate button. In the “PaupSearch Options” menu check in the top box to save the PAUPScript file. The PAUPScript output file results from the automatic conversion of your alignment to NEXUS format and contains all the PAUP commands as well as your alignment. (If needed, the PAUP log file keeps track of all that happened during the program run and is a good place to look for any error messages.) You can change or leave the file names as you wish. Uncheck the next box, “Perform the analysis.” This
makes the program do the conversion to generate the NEXUS script but prevents it from performing the
heuristic search for the best tree (equivalent to the command line option –NoRun). Scroll down through the
options menu, leaving the rest of the options in their default settings, but check them out. “Close” the options
menu. Normally PAUPSearch and PAUPDisplay are linked to each other when you run them from the
SeqLab interface. Therefore, since we don’t want to run PAUPDisplay, uncheck the “PaupDisplay . . .”
button in PaupSearch’s main window. Be sure that “How:” “Background Job” is specified on the main
PAUPSearch menu and then press “Run” there. After a moment the results will be displayed.

The PAUPscript output file is very important. It contains the NEXUS format file that was generated by GCG in
order to run PAUP*. Notice that columns of your alignment with zeroes in their Mask are excluded from the
NEXUS alignment. This file can be used to run the latest version of PAUP*, if available, in its native mode by
‘ftping’ it to an appropriate machine. Using a pre OS X capable Macintosh may be desirable in order to take
advantage of PAUP**’s very friendly Macintosh graphical user interface. Since GCG automatically creates this
file for you, correctly encoding all of the required format data, when you run PAUPSearch, there is no need to
hassle with a later conversion of your alignment to NEXUS. As I stated in the introduction, file format
conversion can be the biggest headache of this whole area and here GCG has done all of that work for you.
When using this file as input to native PAUP* you will want to comment out or remove any inappropriate
commands within the command block with a simple text editor. Likewise, this file can be greatly expanded by
encoding any desired commands within its command block.

As stated above, I would recommend running the latest version of PAUP* available, but whatever version you
run, learn how to run the most robust searches possible, before accepting any output as possible
phylogenetic inference.

Now get out of SeqLab by going to the “File” menu and clicking on “Exit.” You will probably be asked if you
want to save your RSF file and any changes in your list. Accept the suggested changes giving appropriate
names and SeqLab will close. This will return you to your terminal window.

9) Conclusions.

Obviously I have only touched the ‘tip of the iceberg’ regarding SeqLab’s full potential. Please refer to the
online GCG documentation on SeqLab available through GenHelp or the Help menus in SeqLab itself to fully
explore its many possibilities. It is an incredibly powerful way to run the Wisconsin Sequence Analysis
Package.

Gunnar von Heijne in his quite readable, but by now somewhat dated, treatise, Sequence Analysis in
Molecular Biology; Treasure Trove or Trivial Pursuit (1987), provides a very appropriate conclusion:

“Think about what you’re doing; use your knowledge of the molecular system involved to
guide both your interpretation of results and your direction of inquiry; use as much information
as possible; and do not blindly accept everything the computer offers you.”

He continues:
“. . . if any lesson is to be drawn . . . it surely is that to be able to make a useful contribution one must first and foremost be a biologist, and only second a theoretician . . . . We have to develop better algorithms, we have to find ways to cope with the massive amounts of data, and above all we have to become better biologists. But that’s all it takes.”

References


Gilbert, D. G. (1990) ReadSeq, public domain software, Biology Department, Indiana University, Bloomington, Indiana, U.S.A.


