Last modified October 24, 2005

Elementary Sequence Analysis

Brian Golding and Dick Morton

Department of Biology
McMaster University
Hamilton, Ontario
L8S 4K1

These notes are in Adobe Acrobat format (they are available upon request in other formats) and they can be obtained from the website http://helix.biology.mcmaster.ca/courses.html. Some of the programs that you will be using in this course and which will be run locally can be found at http://evol.mcmaster.ca/p3S03.html.

The "blue text" should designate links within this document while the "red text" designate links outside of this document. Clicking on the latter should activate your web browser and load the appropriate page into your browser. If these do not work please check your Acrobat reader setup. The web links are accurate to the best of our knowledge but the web changes quickly and we cannot guarantee that they are still accurate. The links designated next to the JAVA logo, , require that JAVA be installed on your computer.

These notes are used in Biology 3S03. The purpose of this course is to introduce students to the basics of bioinformatics and to give them the opportunity to learn to manipulate and analyze DNA/protein sequences. Of necessity only some of the more simple algorithms will be examined.

The course will hopefully cover . . .

- databases of relevance to molecular biology.
- some common network servers/sites that provide access to these databases.
- use of the internet to obtain sequence analysis software and data.
- methods of sequence alignment.
- methods of calculating genetic distance.
- methods of phylogenetic reconstruction.
- codon usage.
- methods for detecting gene coding regions.

The formal part of the course will consist of two approximately one hour lectures each week. Weekly assignments will be be provided to practice and explore the lecture material. In addition there will be an optional tutorial to help students with these assignments or other problems. These assignments will be 40% of your grade and three, in class quizzes will make up the remainder.

We would appreciate any comments, corrections or updates regarding these notes.

Golding@McMaster.CA Morton@McMaster.CA

This document was entirely constructed with pdfLaTeX. Enormous kudos to all those that make this great software free to everyone.

Contents

Chapter 1

Preliminaries

A reminder: if your Acrobat reader is correctly set, the "blue text" should designate links within this document while the "red text" designate links outside of this document. Clicking on the latter should activate your web browser and load the appropriate page into your browser.

1.1 Resources

There are many resources that one can make use of to study bioinformatics and they are becoming increasingly available to the general public. These notes are my attempt at a small contribution toward this growing body of 'on-line' literature, software, data and knowledge.

Please note that bioinformatics is inherently a multi-disciplinary field making use of biological, mathematical, statistical and computer science knowledge. As such any resources available for any of these disciplines will be of use in bioinformatics. The more skilled that you are in any one of these areas the better off you will be. But you should have a basic minimum knowledge from each of these fields to study bioinformatics. There is a growing body of information available that is specific for bioinformatics.

1.1.1 Electronic Resources

You should note that there are many other valuable hypertext sources that are available to you. I would particularly recommend the VSNS (Virtual School of Natural Sciences) Biocomputing course notes from Bielefeld Germany.

- Introduction
- Pairwise Sequence Alignments
- Networking
- Multiple Alignment
- Mathematical Basis of Molecular Phylogenetics
- Genetic Algorithms and Protein Folding

Another VSNS course is Principles of Protein Structure running out of Birbeck College.

- Overview of Protein Synthesis

- Primary Structure
- Protein Geometry
- Overview of Molecular Forces
- Secondary Structure
- Super-Secondary Structure
- Tertiary Structure
- Protein Folds
- Quaternary Structure
- Protein Interactions

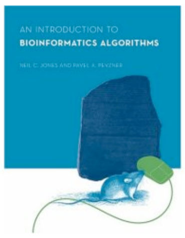
Also excellent is the "lecture notes" site for the Algorithms in Molecular Biology from the Univ. of Washington.

There are many others that you should be able to discover on your own (including one that has garnered the domain name bioinformatics.org), You might want to start your search for others at Pedro's web pages or at the MolBiol toolkit. Each of these are a fabulous resource and often they are "straight from the horse's mouth". You should make frequent use of these resources and others throughout this course (and perhaps you can bring to our attention the ones that you find most valuable that are not listed).

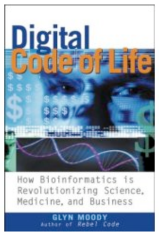
There are also many software packages that provide you with access to a collection of programs that deal with bioinformatics. For example, if you have cash, the famous MatLab software suite provides a toolbox for bioinformatics. For those with less cash, there are interesting projects – Vlinux, Bioknoppix, Vigyaan – that provide you with a bootable CD image. Simply burn the CD (it is free) and then boot from the CD. This provides a free computer system with lots of bioinformatic, biomolecular software at your fingertips (nothing to install, nothing to change on your computer, simply remove CD and reboot when done). There are many other software sources that will be explored in this course (and provided through the links of these notes). For our purposes some of the software that will be discussed below has been provided for you at http://evol.mcmaster.ca/p3S03.html.

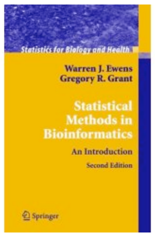
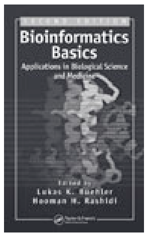
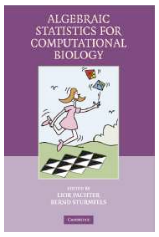
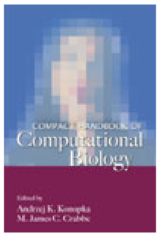
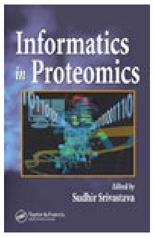
1.1.2 Textbooks

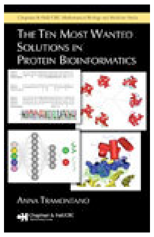
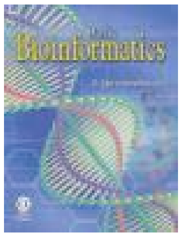
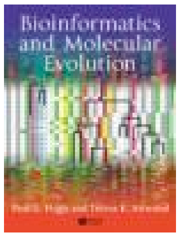
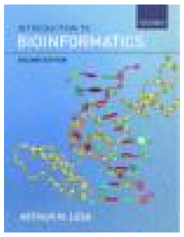
There are now an enormous number of books available that deal with sequence analysis in biology. A selection of just a few that have been published in the last year or so are

- 

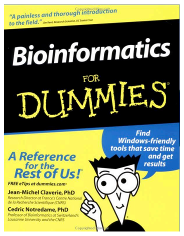
An Introduction to Bioinformatics Algorithms (Computational Molecular Biology) by N.C. Jones and P.A. Pevzner. 2004, Bradford Books

- 
Digital
Code of Life
How Bioinformatics is Revolutionizing Science, Medicine, and Business
GLYN MOODY
Author of Rebel Code
Digital Code of Life : How Bioinformatics is Revolutionizing Science, Medicine, and Business by G. Moody. 2004, John Wiley & Sons

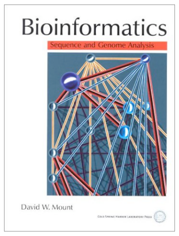
- 
Statistical Methods in Bioinformatics : An Introduction (Statistics for Biology and Health) by W. Ewens and G. Grant. 2005, Springer Verlag
- 
Bioinformatics Basics: Applications in Biological Science and Medicine, 2nd Ed. by H.H. Rashidi and L.K. Buehler. 2005, CRC Press
- 
Algebraic Statistics for Computational Biology, Edited by L. Pachter and B. Sturmfels. 2005, Cambridge Univ. Press
- 
Compact Handbook of Computational Biology bt A.K. Konopka and M.J.C. Crabbe. 2004, CRC Press
-
Bioinformatics for Biologists Rolf Apweiler. 2006 Chapman & Hall
- 
Informatics In Proteomics by Sudhir Srivastava. 2005, CRC Press
-
Pattern Discovery in Bioinformatics Laxmi Parida. 2006, CRC Press

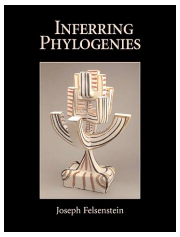
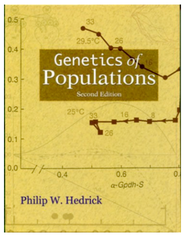
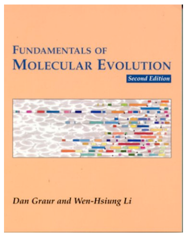
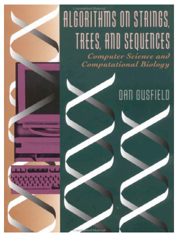
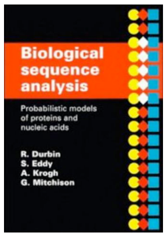
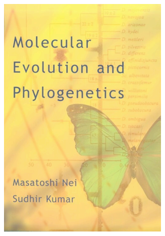
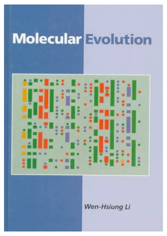
-  The Ten Most Wanted Solutions in Protein Bioinformatics by A. Tramontano. 2005, CRC Press
-  Basic Bioinformatics by S. Ignacimuthu. 2004, Alpha Science Intl
-  Bioinformatics and Molecular Evolution by P. Higgs and T. Attwood. 2004, Blackwell Publishing
-  An Introduction to Bioinformatics, 2nd Ed. by A. Lesk. Oxford Univ. Press

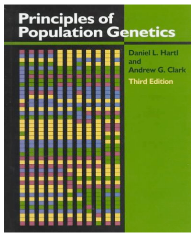
and there is EVEN now a book from the popular "dummy" series

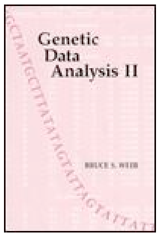
-  Bioinformatics for Dummies by J.-M.Claverie and C.Notredame 2003.

A selection of the more important and recommended texts are

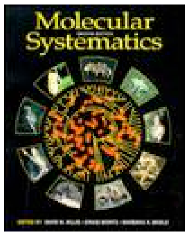
-  Bioinformatics: Sequence and Genome Analysis by D.W.Mount 2001, Cold Spring Harbor Laboratory Press. (*highly recommended*).

- 
  Inferring Phylogenies by J.Felsenstein 2003, Sinauer Associates. (*highly recommended*).
- 
  Genetics of Populations by P.W.Hedrick 2000, Jones and Bartlett.
- 
  Fundamentals of Molecular Evolution by D.Graur and W.H.Li, 1999, Sinauer.
- 
  Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology by D.Gusfield 1997, Cambridge University Press (*highly recommended*).
- 
  Biological Sequence Analysis by R.Durbin, S.Eddy, A.Krogh and G.Mitchison 1998, Cambridge Univ. Press (*highly recommended*).
- 
  Molecular Evolution and Phylogenetics by M.Nei and S.Kumar, 2000 Oxford University Press.
- 
  Molecular Evolution by W.H.Li 1997, Sinauer.

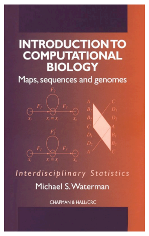
- 
Principles of Population Genetics
Daniel L. Hartl and Andrew G. Clark
Third Edition
Principles of Population Genetics by D.L.Hartl and A.G.Clark 1997, Sinauer (*highly recommended*).

- 

Genetic Data Analysis II by B.Weir 1996, Sinauer (*highly recommended*).

- 

Molecular Systematics by D.Hillis, C.Moritz and B.Mable 1996, Sinauer.

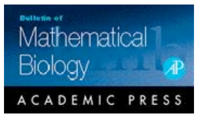
- 
INTRODUCTION TO COMPUTATIONAL BIOLOGY
Maps, sequences and genomes
Interdisciplinary Statistics
Michael S. Waterman
CHAPMAN & HALL/CRC
Introduction to Computational Biology: Maps, Sequences and Genomes by M.Waterman 1995, Chapman and Hall.

In addition to these there are many texts on evolution, on DNA and on proteins that have useful chapters and sections on sequence analysis.

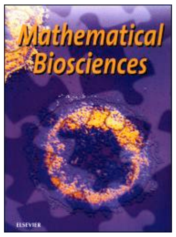
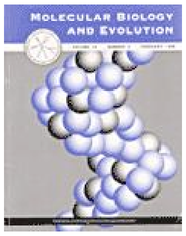
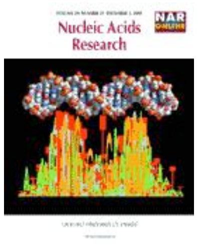
1.1.3 Journal sources

Again there are many periodicals relevant to sequence analysis. Besides the general science journals such as Nature, PNAS, Science, EMBO, . . . there are several which are more specific to molecular evolution, to computers in biology, and to sequence analysis. Some of these journals include . . .

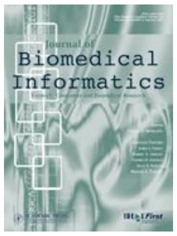
-
Applied Bioinformatics

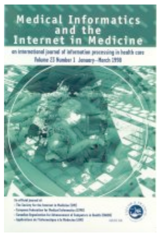
- Bioinformatics (formerly entitled Computer Applications In The Biosciences : CABIOS).
- Briefings in Bioinformatics
- BMC Bioinformatics
- BMC Evolutionary Biology
-  Bulletin of Mathematical Biology
- Genome Biology
- Genomics
- In Silico Biology

- Journal of Computational Biology

- J. Mathematical Biology
- Journal of Molecular Evolution
-  Mathematical Biosciences
-  Molecular Biology and Evolution
- Molecular Phylogenetics and Evolution
-  Nucleic Acids Research
- PLOS Computational Biology
- Systematic Biology

and in the medical sciences there are many (!) more, including

-  Journal of Biomedical Informatics

- Journal of the American Medical Informatics Association
-  Medical Informatics and the Internet in Medicine

for a more complete listing of medically related journals see MedBioWorld.

To find individual papers on specific topics there is the Swiss sequence analysis bibliographic database SeqAnalRef or the more general search engines from N.C.B.I. Entrez for access to papers in Medline and the more recent Google Scholar pages.

1.2 Biological preliminaries

We will assume throughout the remainder that some familiarity with basic biology has been obtained. We do not assume any more knowledge of mathematics than can be obtained at a high school level.

1.2.1 Some notes on terminology

There are some terms that will be used here, that are commonly abused. Unfortunately, I too will use some terms that are not precise so you should be aware of the proper definitions (the following are modified from Futuyma 1986, Evolutionary Biology, Sinauer Assoc.).

Homology Contrary to some statements in other bioinformatic texts, homology and similarity are not the same thing. A trait from two different species or taxa are said to be similar if they have some resemblance of one to another. Homology means a great deal more. Two traits from a different species or taxa are homologous if they are derived (with or without modification) from a common ancestor.

In general when working with sequences, one assumes homology if one finds excessive similarity between the two sequences. However, you should be aware that this is an inference that should be consciously made.

Example: The traditional example is that of the wings of birds and bats. Their wings are similar in that they enable flight, have the same name and have similar aerodynamic constraints but they are not homologous. They are not homologous because the common ancestor of both birds and bats did not have wings, rather wings evolved within each group separately.

Mutations A mutation is an error in the replication of a nucleotide sequence. It may encompass one or many nucleotides and in complicated situations may involve disjoint nucleotides. They can be caused by internal errors of metabolism or by external agents such as radiation.

Substitutions Mutations are not substitutions. Substitutions are differences in two sequences (generally the descendant from the ancestral) caused originally by mutations but which have been acted on by selection.

Example: Because substitutions have been exposed to selection, the frequency of occurrence of individual substitutions and mutations are quite different. In general substitutions at the second position of a codon are (almost always)

Table 1.1: One letter amino acid codes

Alanine	A	Arginine	R	Asparagine	N
Aspartic acid	D	Cysteine	C	Glutamic acid	E
Glutamine	Q	Glycine	G	Histidine	H
Isoleucine	I	Leucine	L	Lysine	K
Methionine	M	Phenylalanine	F	Proline	P
Serine	S	Threonine	T	Tryptophan	W
Tyrosine	Y	Valine	V	Unknown	X

much less frequent than those in the third codon position. This is because a change at the second codon position will alter the amino acid encoded but this is not always the case for changes at the third codon position. By contrast, we expect mutations to occur equally frequently at each of the codon positions.

Replacements The term replacement is suggested to be used when differences between amino acid sequences are observed.

1.2.2 Letter Codes for Sequences

To store a large amount of data on a computer it would be quite inefficient to store the amino acids as "Glutamic acid" or to store ambiguous nucleotides as "A or G". For this reason there are standard codes to represent amino acids and nucleotides. Both of these are one letter codes and can be stored on electronic media with reasonable efficiency.

Amino acids have in the past often been designated by a three letter code. This three letter code is not suitable for electronic media and is now largely obsolete. The standard one letter amino acid codes are shown in Table 1.1. Also commonly in use are B to represent either Aspartic acid or Asparagine and Z to represent either Glutamic acid or Glutamine.

There are also standard one letter codes to represent nucleotides. While most people are familiar with the simple codes of A, C, G, T, and U there are more extensive codes to include ambiguities in the nucleotides. The extended one letter code for nucleotides is given in Table 1.2. The complete generality of this code is seldom used. More common is the use of only part of the extended code

R	A or G
Y	T or C
N	A,T,C or G
X	unknown

Some programs prefer to store RNA codes rather than DNA codes. In general T and U can often be taken as synonyms.

Table 1.2: One letter nucleotide codes.

Based on Nomenclature Committee of the International Union of Biochemistry (NC-IUB). Molecular Biology and Evolution 3:99-108 (1986).

Guanine	G	G
Adenine	A	A
Thymine	T	T
Cytosine	C	C
Purine	G or A	R
Pyrimidine	T or C	Y
Amino	A or C	M
Keto	G or T	K
Strong (3H bonds)	G or C	S
Weak (2H bonds)	A or T	W
Not G	A or C or T	H
Not A	G or T or C	B
Not T	G or C or A	V
Not C	G or A or T	D
Any	G or C or T or A	N
Unknown	?	X

Chapter 2

Genomics

In the last decade there has been a data explosion in the biological sciences. These have been termed the 'omics. The most relevant to this course is genomics. Which I will briefly explore in this section. But beware there are many other that are of relevance to this course and many of the techniques are relevant to all of the 'omics. Other fields that we will not have the time to explore include proteomics, transcriptomics, metabolomics, pharmacogenomics, toxicogenomics and so on. All have the fields have the same characteristic of generating enough data that a simple hands-on approach by a single researcher is not adequate.

2.1 Where the data comes from

The study of genomics is, as the name implies, the study of entire genomes. This includes all elements of the genome – the genes, the proteins, and the non-coding regions of an organism's chromosomes. It entails a study of the structure of these elements, how they work, how they interact and how they evolve.

But genomes are huge. The human genome is over 3 billion nucleotides in total and encodes tens of thousands of genes and perhaps a hundred thousand proteins with, a currently unknown number of interacting components. Genomics is not possible without a high-throughput approach.

Technological advances have made it possible to sequence the entire genome of organisms and to do this in a high-throughput format such that it can be accomplished within a short period of time (becoming more rapid each year with each new advance). This course deals with the basics of the analysis of sequence data but some background on it the origin of the sequence data is required.

2.2 How DNA is sequenced

The first direct attempts to sequence an RNA molecule were by Holley and co-workers in 1965 (R.W. Holley *et al.*, 1965, Science 147:1462-1465). The technique that they used was very labor intensive and it took them approximately one year to determine the 77 nucleotides that make up the alanine transfer RNA of yeast.

Modern methods rely on gel electrophoresis to separate different sized fragments of a larger DNA molecule and the size of the fragments is used to provide the clues about the the linear order of nucleotides. Electrophoresis is the application of an electrical charge to a gel-like substance. Gels can be composed of different materials such as starch, acrylamide, or agarose. Because most molecules have a static electrical charge, when they are placed in an electric field within a gel, they will move according to their charge and also according to their size. Each of the different gel substances have different effects on the movement of these molecules. DNA being an acid, has a slightly negative charge and will therefore migrate toward the positively charged end of a gel. For DNA separation (where size and charge are proportional), acrylamide gels are generally used and these will primarily separate the molecules on the basis of their size. Small molecules (small fragments of DNA) will migrate faster and travel a greater distance through the gels. Larger molecules migrate slower

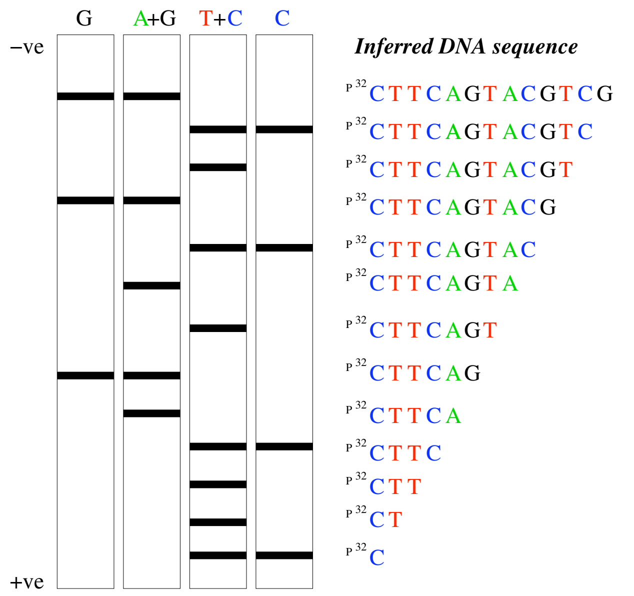


Figure 2.1: The Maxam-Gilbert method of sequencing DNA. The black bars indicate what would be seen in an autoradiogram of the lanes from a sequencing gel. Shown on the right is the inference of the corresponding DNA sequence.

and will not move as far. Electrophoretic methods are sensitive enough to discern the difference in length between DNA molecules that differ by a single nucleotide.

Maxam and Gilbert sequencing makes use of electrophoresis to determine DNA fragment sizes. This method was developed by A.M. Maxam and W. Gilbert in 1977 (A new method for sequencing DNA, Proc.Natl.Acad.Sci. 74:560-564). The first step is to clone a DNA fragment of interest. This is necessary to obtain a large quantity of a specific DNA molecule. The next step makes use of polynucleotide kinase to add radioactively labeled phosphate to the $5'$ end of a cloned DNA molecule (to prevent labeling of both $5'$ ends some further tricks are required that will not be explored here). At this point, the molecules are all labeled with a radioactive probe that can be readily detected by placing the gel next to a large piece of photographic film. The radioactive probe will expose the film at the spot on the film that corresponds to its position in the gel.

But at this point all molecules are still the same length and it would not be possible to differentiate them on a gel. The next step is therefore to divide into four separate aliquots. Into one aliquot, dimethylsulfate will added. This methylates guanine side groups but the reaction is not permitted to go to completion and hence only some guanines will be methylated. When treated with heat the glycosidic bond of the methylated guanines in these molecules will be broken leaving a free sugar in the DNA. Alkali treatment at 90^0C will cleave the DNA at this free sugar. Into a second aliquot, a mild acid is added. This will preferentially cleave the DNA at locations where there is an adenine or guanine present. Into a third aliquot, hydrazine is added and the DNA is preferentially cleaved at points which contain cytosines and thymines. Into the fourth aliquot, hydrazine with high salt (2M NaCl) is added which suppresses the cleavage at thymines.

The end result is that these four reactions now contain DNA fragments broken at G's, at A+G's, at T+C's and at C's. When these fragments are run through a gel the DNA fragments separate by size (migrating different distances through the gel) and can be visualized by an autoradiogram that detects the radioactive phosphorus. From the pattern of bands on the gel the DNA sequence can be readily inferred as shown in Figure 2.1.

The Sanger method of sequencing is far more commonly used today due to its greater simplicity (but the Maxam-Gilbert method has other uses). This method makes use of the replication of DNA template by a polymerase. The method therefore requires a primer for DNA synthesis and then interrupts this process at points corresponding to the linear sequence of nucleotides. The method was developed by F. Sanger & colleagues and makes use of dideoxyribonucleotides which will be incorporated into a replicating molecule at random positions (F.Sanger, S.Nicklen, A.R.Coulson, 1977, Proc. Natl. Acad.Sci. 74:5463). The dideoxynucleotides lack the $3'OH$ on the sugar. The diagram in Figure 2.2 shows a cartoon of the normal process of DNA synthesis. DNA nucleotides are normally $2'$-deoxynucleotides and have an OH group at the $3'$

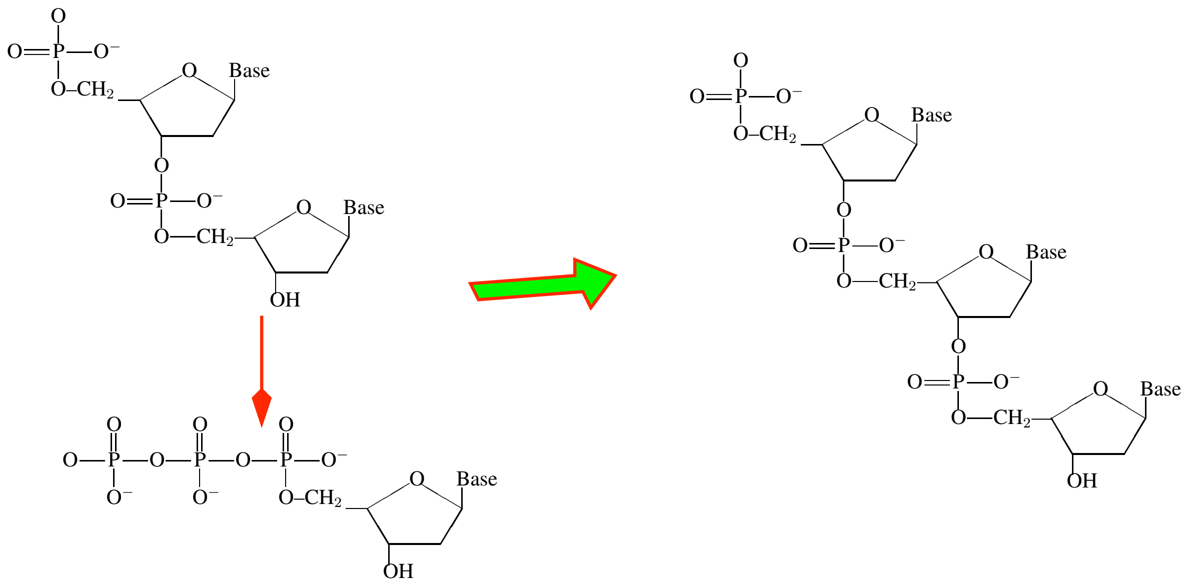


Figure 2.2: The normal process of DNA replication. Only one chain of the sequence is diagrammed (the template strand is not shown). The polymerase catalyzes the addition of nucleotide triphosphate (bottom left) to the growing strand leading to a larger molecule shown on the right.

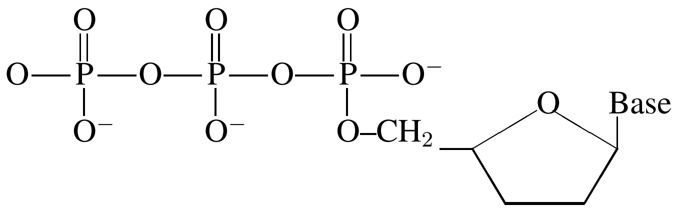


Figure 2.3: A dideoxynucleotide triphosphate. This nucleotide will be incorporated into a growing sequence strand but because it lacks a $3' - OH$, this nucleotide will block further addition.

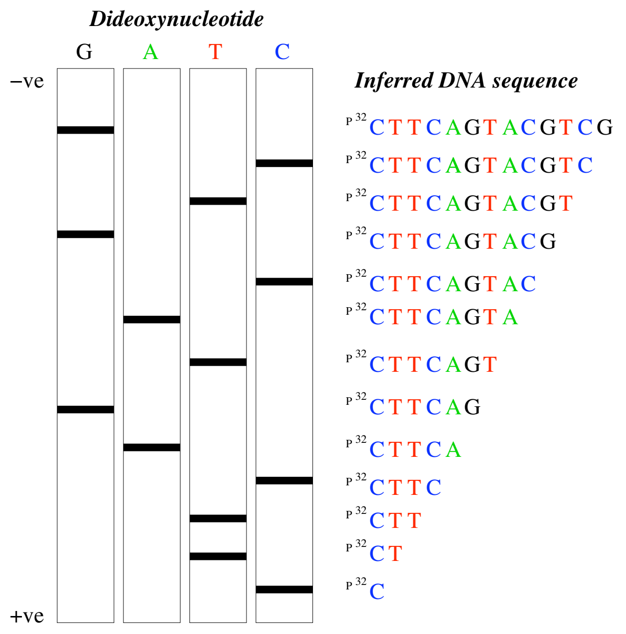


Figure 2.4: The Sanger method of sequencing DNA. The black bars indicate what would be seen in an autoradiogram of the lanes from a sequencing gel. Shown on the right is the inference of the corresponding DNA sequence.

carbon. With the addition of nucleotide triphosphate, a polymerase will catalyze a reaction indicated by the red arrow where the OH is exchanged for bond with the phosphate group of the next nucleotide in order (according to the complementary strand which is not shown in this diagram). Sanger's method makes use of $2'$,$3'$-dideoxynucleotide triphosphates (Figure 2.3) and the $3'$ carbon the point where the next nucleotide attaches via the formation of a phosphate bond ("O - P - O"), the polymerase will stall at the point of addition of the dideoxynucleotide. But even if the polymerase still has proof-reading activity, it will not rapidly excise the dideoxynucleotide because the corresponding bases are correctly hydrogen bonded. Again, four individual reactions containing one of the four dideoxynucleotides can be constructed and the sequence can again be inferred. In this case, the radioactive label can be attached to the primer.

The Sanger method therefore creates a collection of DNA fragments that are blocked at random points by these dideoxynucleotides. Like the Maxam-Gilbert method it too has four reactions mixtures that are each run in a different lane of a gel. The method originally required fairly large volumes and the dangerous use of radioactive labels. Cloning DNA fragments to generate sufficient raw material of a single DNA molecule was difficult. Reading the resulting autoradiograms became a tiresome task that many a graduate student has complained about.

More recent improvements have overcome many of these problems. First the chemistry has become more standardized and reaction volumes have become smaller. The PCR (polymerase chain reaction) was able, in most cases, to replace any requirement for cloning by generating large quantities of a template. Instead of a radioactive probe attached to primers, fluorescent probes are used. Using four different fluorescent colours, you can combine the reactions into a single lane on a gel. You can shrink the size of the lane to a capillary. Then as the DNA fragments migrate within the electrophoretic field, the fluorescent probes can be excited by a laser and their emitted light can be detected and automatically measured by a photometer. The intensity is measured as the run proceeds and is automatically stored into a computer. An example of a sequence chromatograph is shown in Figure 2.5-2.7 (this chromatograph comes from the bacterium *Sinorhizobium meliloti*). The resulting bases can be inferred by a computer program and automatically analyzed.

2.3 The reality of sequencing includes errors

As with most human endeavors the process of DNA sequencing is not 100% accurate. The beginning of a sequence run (or trace) is usually too poor to permit inference of the DNA sequence. Also as the mixture of DNA fragments is run for an

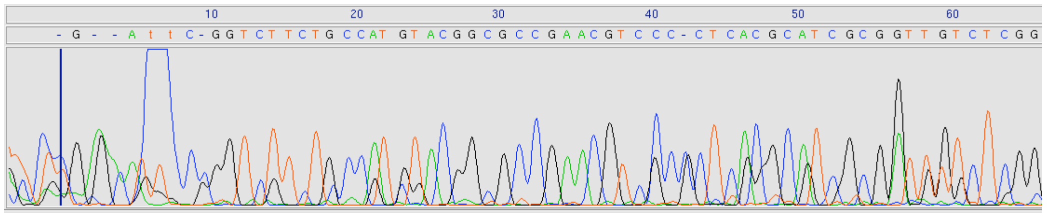


Figure 2.5: Example of the beginning of a trace

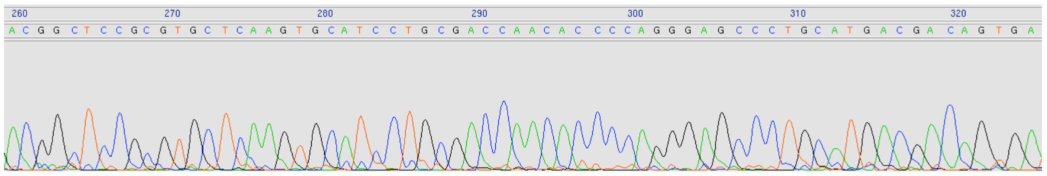


Figure 2.6: Example of the middle of a trace

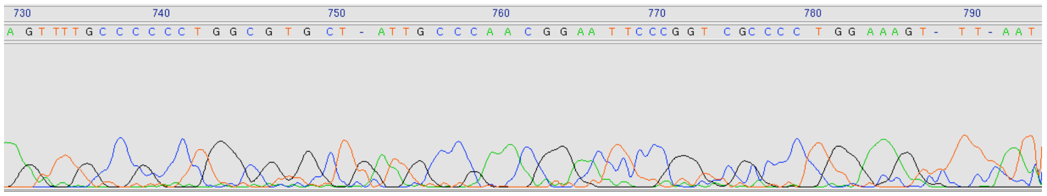


Figure 2.7: Example near the useful end of a trace

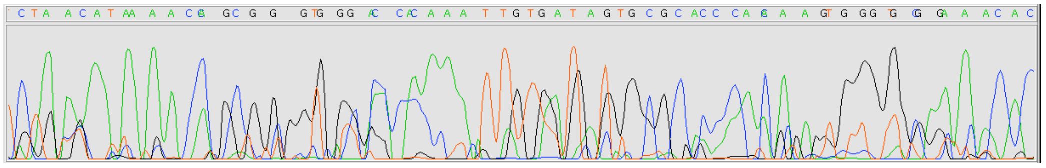


Figure 2.8: Example of a poor trace

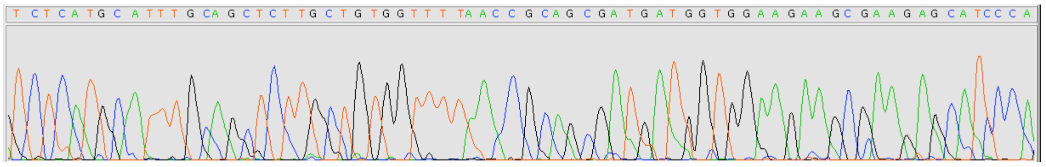


Figure 2.9: Example of a better quality trace

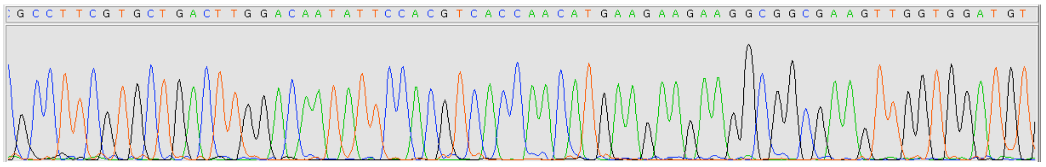


Figure 2.10: Example of a good trace

extended period of time, the electrophoretic resolution of the fragments becomes poor and identical fragments will migrate to different distances in the gel. This causes the trace for each nucleotide to spread out and become broader. This itself is not a problem but as the height of the chromatograph peaks shrink and as their overlaps become more extensive, the ability to determine which nucleotide is followed by which becomes more difficult.

In addition, a poor trace can result from many different factors. For example, if there is a repetitive region being sequenced, the polymerase might stutter as it goes through the region. Alternatively there might be more than one template being sequenced. In either case, the trace will contain more than one sequence superimposed and it will be impossible to correctly call the sequence (but under good conditions, base substitution polymorphisms can be detected).

Compression is a common phenomena in DNA sequencing. This occurs when two (or more) guanine nucleotides appear in the sequence in a row, these bases will stack together and appear much closer electrophoretically than would a mixture of other nucleotides. Since base calling makes use of the separation between peaks it can be fooled into calling a single base present with a wide peak rather than two bases present each with peaks pushed together. For all of these reasons it usually necessary to sequence the same segment of DNA from the opposite direction to ensure that the nucleotides have been correctly determined.

2.4 From sequence to genome

At this point many projects will end and move on the next step of analysis. All that was of concern was a particular gene's sequence and this has been obtained and (hopefully) confirmed with multiple reads. Other projects however, are interested in obtaining the complete genomic sequence. Sometimes this is a matter of economics. It is cheaper to sequence the entire organism in a single laboratory and make the data available to everyone, than to have thousands of laboratories each sequencing individual pieces. At other times it is a matter of intrinsic interest to determine the genomic sequence. There are features of biology and evolution that were not apparent without this information such as the evolution of gene rearrangements.

In a single sequence run using the above methods (there are others with different advantages) you can reliably detect less

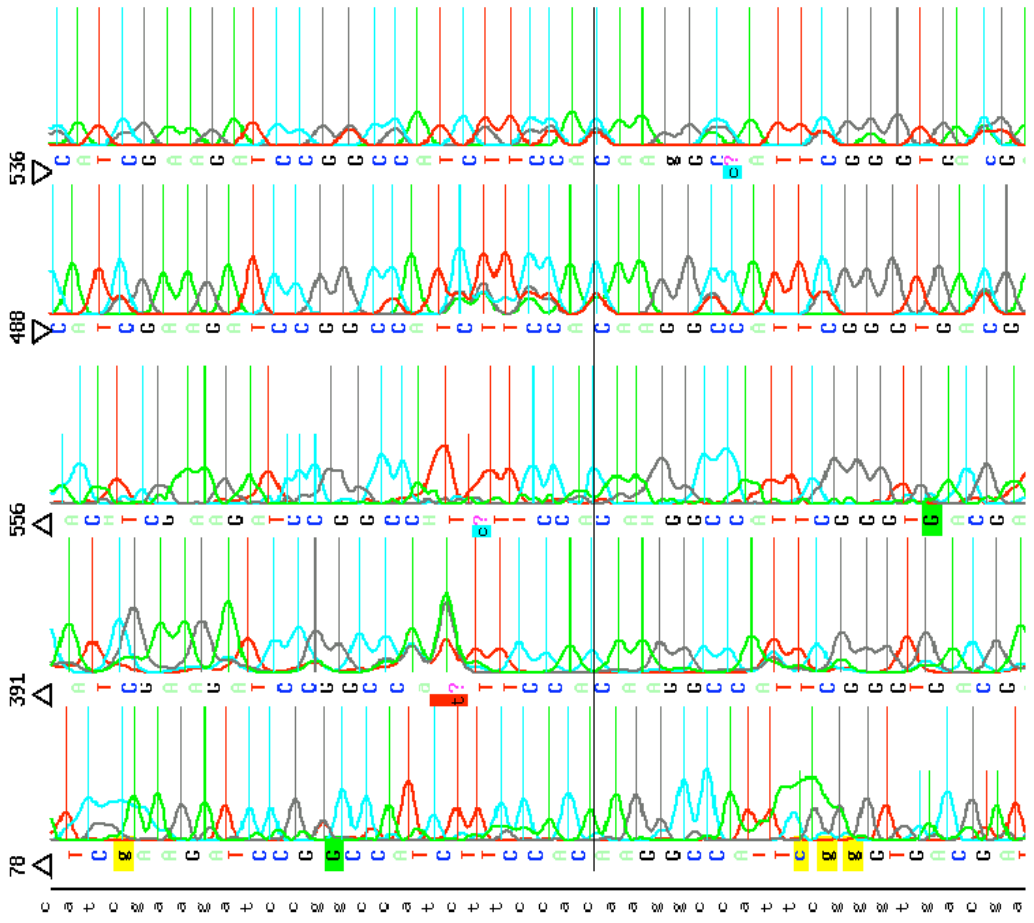


Figure 2.11: An example of multiple traces from the same sequence

than 1,000 nucleotides. This is a very small number when you consider that even a bacterial cell will contain a chromosome of over 4,000,000 nucleotides. Further, remember that to ensure accuracy, the molecules must be sequenced several times and in both directions (or since DNA synthesis occurs in only one direction more accurately both strands of the original molecule must be sequenced; 5′ and 3′). In many genome sequencing projects, an average base is covered often 7× to 10×. So even the tiny bacteria has grown to 28,000,000 – 40,000,000 nucleotides. In order to move beyond a single gene sequence to genome sequences the entire process must be automated as far as possible.

Currently *Shotgun* sequencing has become the most popular method to sequence a genome. This involves collecting completely random sequences from the organism (hopefully a truly random collection rather than a biased collection). These are all cloned into a plasmid (vector) and sequenced with a standard primer that reads from the plasmid sequence into the cloned sequence. Unlike a directed sequence project this method requires more work but more of it can be automated and done in a rather blind fashion. The method requires a computer to put together the individual reads into a coherent collection. This process will be illustrated with the next few figures. Note that the following figures are derived from the ACEMBLY suite of software that accompany the ACeDB suite. Other suites of software are popular including the Staden package, PHRED/PHRAP package and so on. More about these later. Each of the figures in this section come from a project to determine the genomic sequence of the bacterium *Sinorhizobium meliloti* (in particular the chromosome pEXO from this organism).

Multiple reads from a single region of a sequence are shown in Figure 2.11. This is a collection of sequence reads in both directions and you will note that the traces do not agree in what the sequence should be. For example the bottom trace

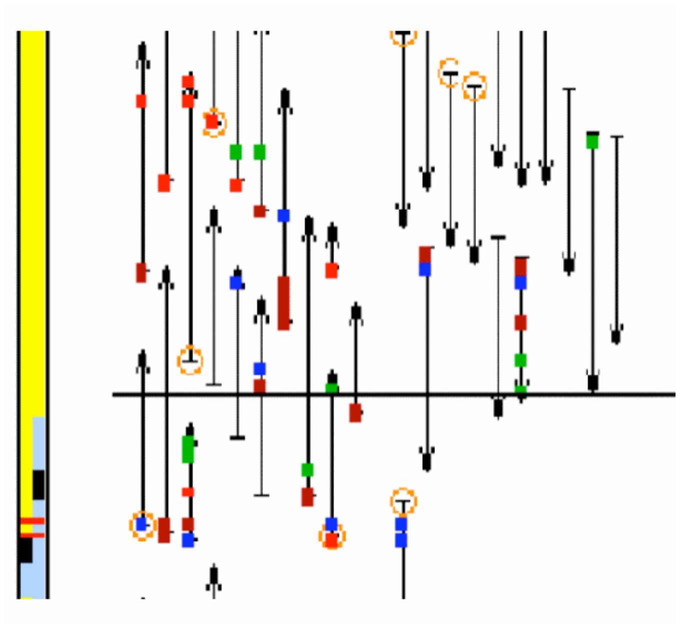


Figure 2.12: An example of overlapping traces

does not infer that a "G" should be in the beginning of the sequence "TCGAA" and hence this is highlighted by the yellow background. The overall sequence has to decided by an evaluation of each of the different reads. The degree of reliability of each trace is taken into account (is this the end of a trace?, is it poor quality?) as well as the relative intensities from each of the four fluorescent probes.

All of the reads from each of the sequences are put together in this way to create a consensus sequence. There may be a large number of these reads for any one region of a genomic sequence (a diagrammatic example of overlapping reads is shown in Figure 2.12). Here the coloured boxes indicate disagreements between the individual reads and the consensus, while the circles at the ends of the reads indicate that vector sequence has been trimmed from the end of the reads. The bar to the left indicates the progress of the sequencing for this region of the consensus. The yellow strip on each side of the bar indicates good coverage in both forward and reverse directions. The blue colour indicates limited coverage in one direction and the black colour indicates that there is no sequence in that direction. The red strips in the bar indicate an unresolved disagreement between the reads for a particular base. Note that although there are many coloured boxes on the individual reads indicating disagreements between the reads, these are generally resolved by multiple reads and result in only a few red bars.

As the sequences for the genome accumulate, a consensus among individual reads is found by computer. This consensus grows in size as new reads are made and as they overlap in their sequence. It is a time consuming process to take each read and determine if and how it might overlap with the other reads. Intelligent algorithms have been developed to carry out this process.

As the reads are put together, the consensus sequence will grow in length. These growing chunks of sequence are called "contigs" (contiguous regions of sequence). An example of contigs are shown in Figure 2.13. The individual reads are shown on the right of the figure. The blue arrows show a contiguous overlapping consensus sequence, with the largest region at the top moving down to smaller regions and with singleton reads at the bottom. Previous contigs joined together in this analysis are shown by the black arrows to the left of the blue contigs.

One would hope that with enough reads the contigs will be joined into a single sequence that would represent the entire chromosome. However, at some point, there are diminishing returns and it is more efficient to target a particular gap between contigs to join them together. This can be done by taking the sequence at the end of a contig and making sequencing primers that would extend beyond the limit of the contigs. Sometimes other more devious measures have to be applied to fill these gaps. Sometimes they simply cannot be filled. This is the case for many eukaryotic sequences. The

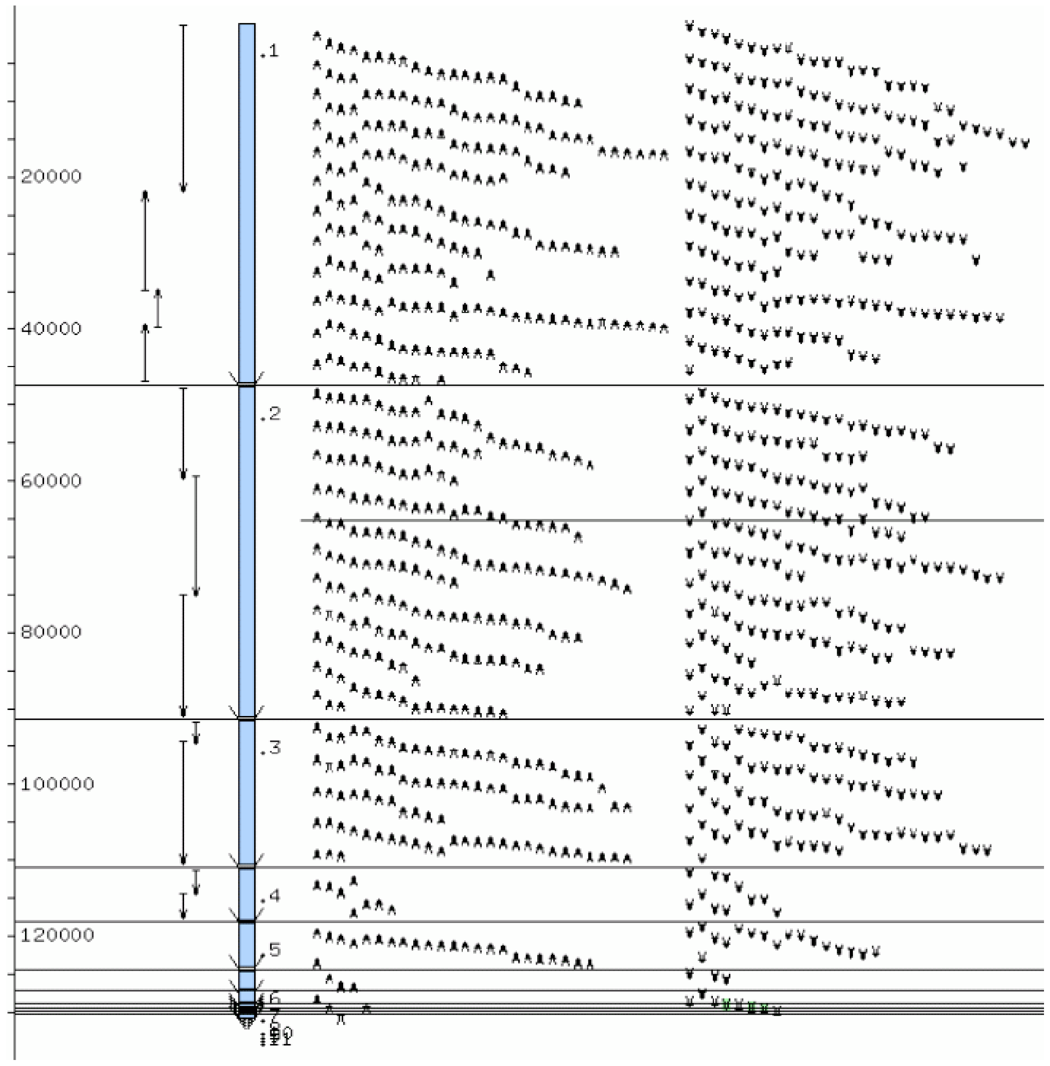


Figure 2.13: Contigs Example

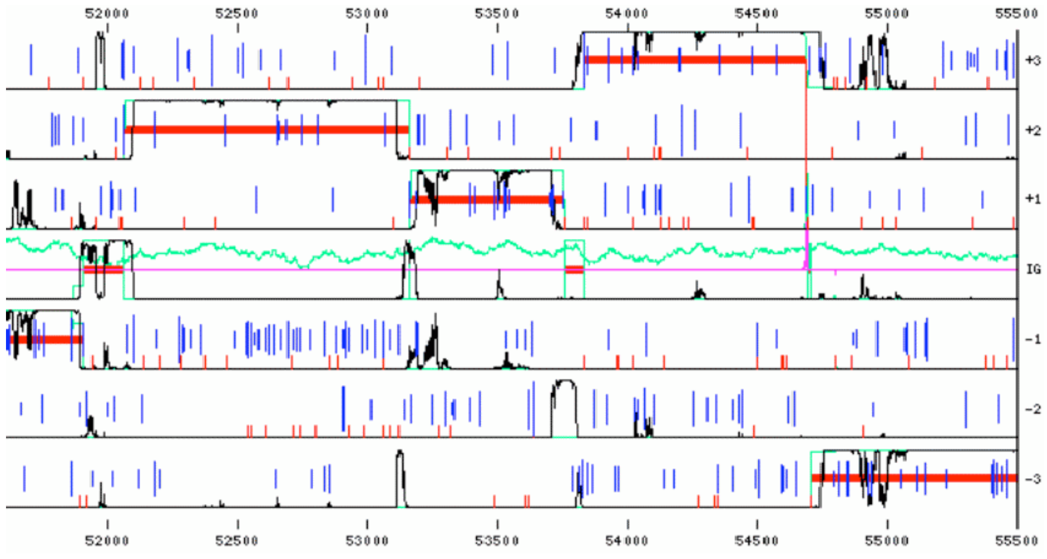


Figure 2.14: Potential coding regions must be found — in this case using a Hidden Markov Chain method called FrameD

centromere of many eukaryotes consists of short sequences repeated up to a million times. There is no reason to sequence through these (ignoring the difficulties of actually doing so) and hence they are intentionally left as gaps in the sequence.

The next step in most genomic sequencing projects is to figure out (at least in a preliminary sense) what the sequence does. That is, where are the genes, where are structural features such as repeats, signal sequences and so on. In prokaryotes this is comparatively easy since their genes are contiguous along the sequence and are without internal gaps. In eukaryotes, the genes are interrupted by the presence of introns and the individual exons of genes may be separated by long distances. Even with prokaryotes however, there are no flags sitting in the DNA stating that this is a gene. Some of the methods of annotation will be discussed in greater detail in Chapter 11.

Briefly, to identify a gene you require open reading frames of sufficient length to be a reasonable gene or exon ("reasonable" having a very loose definition). You can make use of similarities to other existing genes. You require a ribosome binding site at the beginning of the genes in prokaryotes. Further more, sequences within genes have particular patterns that can be searched (more on this in Chapter 11). The analysis in Figure 2.14 makes use of all of these characteristics to search for the presence of genes. There are three rows at the top (the three frames possible in the $5'$ direction), a middle bar showing other features, and three rows at the bottom (the three possible frames in the $3'$ direction). The vertical axis indicates the chances that a gene is encoded in any one region. The blue and red vertical bars are for start/stop codons. The horizontal red bars indicate the genes "called" by this particular method.

Predicting the presence of a gene is difficult. Generally multiple methods are used, many of which are tailored specifically for the species being considered. Finally, humans will carefully double check all of the computer predictions and create an annotation of potential genes for the genome. A diagram of such an annotation is shown in Figure 2.15. This is a segment of the annotation for the genome sequence of the bacteria *Sinorhizobium meliloti*. This is presented in a typical fashion. The boxes are meant to represent genes in the $5'$ direction on top of the line or the $3'$ direction below the line. The colours of the boxes represent different types of genes with many of the boxes hypothetical, unknown, or unique (again very typical of many genomes ... we don't know what most of the genes do).

2.5 The future of sequencing?

There are many companies that are trying to develop methods to sequence DNA more rapidly and with less cost. Much of the progress on the latter has been achieved via minaturization. But to accomplish the former, novel methods to sequence DNA are being explored.

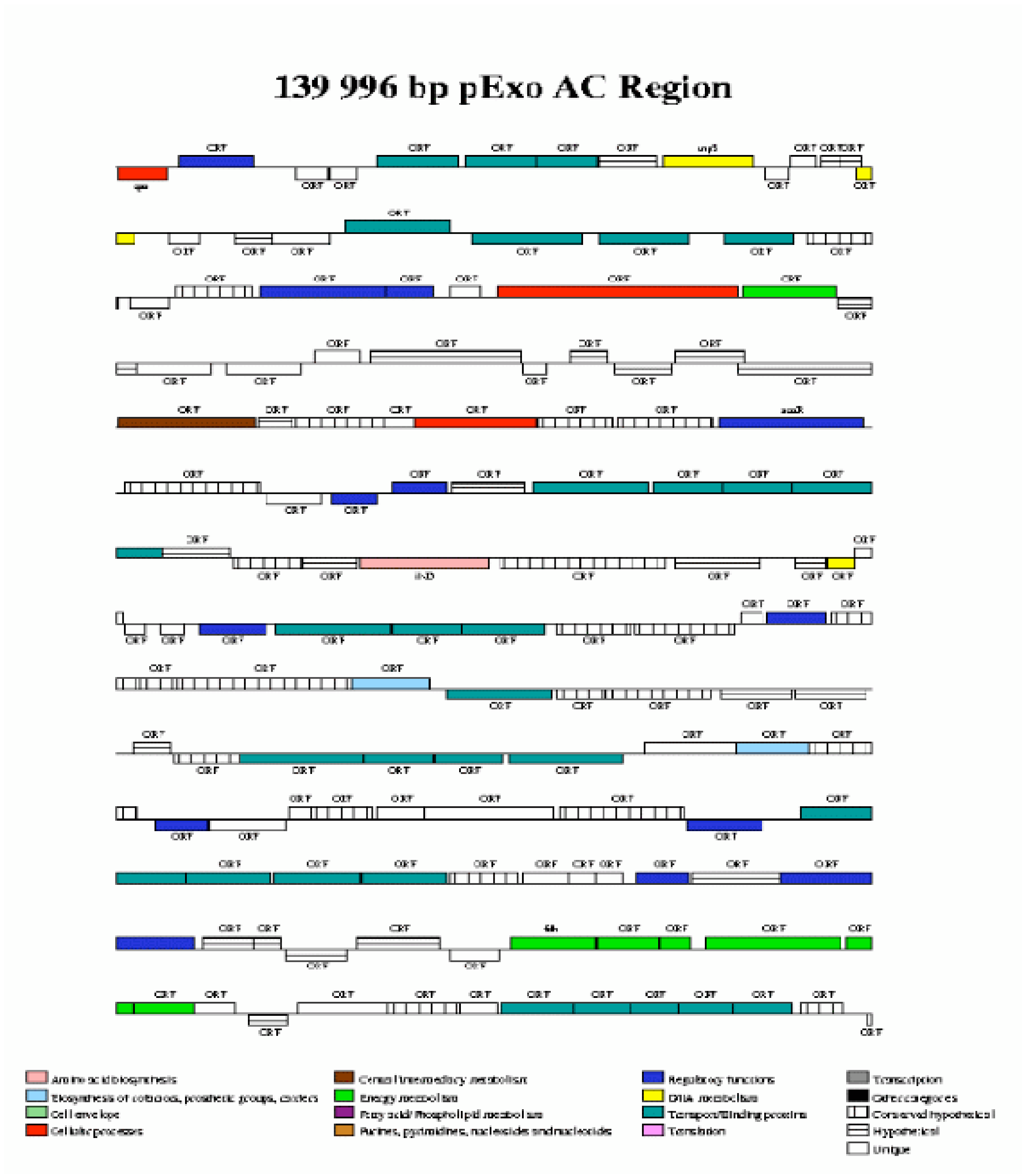

139 996 bp pExo AC Region
ORF
Amino acid biosynthesis
Biosynthesis of cofactors, prosthetic groups, carriers
Cell envelope
Cellular processes
Central intermediary metabolism
Energy metabolism
Fatty acid/ Phospholipid metabolism
Purines, pyrimidines, nucleosides and nucleotides
Regulatory functions
DNA metabolism
Transport/Binding proteins
Translation
Transcription
Other categories
Conserved hypothetical
Hypothetical
Unique

Figure 2.15: An example of the annotation for a fragment of a genome

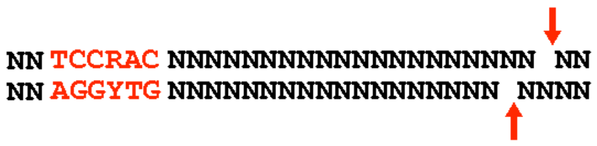

NN TCCRAC NNNNNNNNNNNNNNNNNNNN NN
NN AGGYTG NNNNNNNNNNNNNNNNNN NNNN

Figure 2.16: The Mme1 restriction enzyme creates staggered cuts at a distance from the recognition site

Resequencing methods are and have been developed. These methods are particularly important in the generation of SNP data. SNP stands for Single Nucleotide Polymorphism (see the SNP consortium and the SNP fact sheet) and are differences between individuals (polymorphisms) that can be used to map genes, to analyze human risk factors for disease development, and, if disease mutations are known, to predict the occurrence of genetic diseases. The concept behind resequencing is that once the sequence is known, it is possible to use this knowledge to aid in the determination of new sequences. This is of particular use in mutational analysis. An individual at risk can be rapidly, cheaply screened for mutations that cause a particular disease. This is usually done by constructing oligonucleotides that will cover most of the likely changes. Then hybridization of the patient's DNA to these oligos is quantified. Different methods make use of a gain of hybridization signal to an oligo containing sequences known to cause the disease. Other methods make use of the loss of a hybridization signal to perfect match oligos. The tricks here involve construction of a large number of oligos and the subsequent scanning of the degree of hybridization to each oligo. The ultimate goal of this methodology would be to create a "universal array" that contains all possible oligonucleotides. Although quantifying the presence of all possible oligos does not permit the determination of a new genome sequence, it can be used to determine the sequence of a variant of a known sequence. Theoretically at least, Pe'er *et al.* 2002 PNAS 99:15492 have shown that an array consisting of just 8'mers is sufficient ($8^4 = 4096$) to resequence targets of more than 2kb (as will be seen below, an array this size is easily achieved).

Still other methods of resequencing being explored make use of primer extension reactions to perfect match oligos. These oligos are then arrayed on a surface (e.g. see section 2.6.1) and sequencing is performed on this surface. The dideoxyribonucloeside triphosphates are added each labelled with a different fluorescent dye and then fluorescent microscopy is used to assign the identity of target nucleotides extended from the $3'$ end of oligo (Pastinen *et al.* 1997 Genome Res 7:606).

Another method being explored is to make use of the developments in mass spectrometry. Matrix-assisted laser desorption ionization time-of-flight mass spectrometry (MALDI-TOF MS) combined with methods to ionize macromolecules using electrospray ionization. Normally creating ions of macromolecules has been difficult but advances in laser technology and ionization methods have made this possible for fragments of DNA. The advantage of a mass spectrometry method is that it is highly repeatable and consistently accurate. This is particularly useful with DNA fragments that are difficult to sequence through gel electrophoresis and in fact can be used to sequence RNA molecules (for a review see Edwards *et al.* 2005 Mutation Research 573:3). This method also has the ability to resequence small genomes and is useful in clinical applications (Tost and Gut 2005 Clin Biochem 38:335).

To resequence large genomes a method has been developed by Shendure *et al*. 2005 Science DOI: 10.1126/science.1117389 that can (in principle) handle an entire bacterial genome. Their method begins by size selecting randomly sheared 1kb fragments from the genome. These are ligated to a universal linker under conditions that will result in both ends of the 1kb fragments being ligated to the ends of the linker (creating circular molecules).

The linker contains a Mme1 restriction site at each end. Mme1 is a restriction site that recognizes the sequence $5'$-TCCRAC-$3'$ and then creates a staggered cut 20 bases in the $3'$ direction on the $5'$-$3'$ (upper) strand and 18 bases away in the $3'$ direction on the $3'$-$5'$ (lower) strand (see Figure 2.16). Cutting the circular construct with this enzyme creates a molecule that contains the linker with 18 bp of genomic sequence at each end. Universal amplification/sequencing primers are then added to each end. Hence, this results in $2 \times$ 18bp of genomic DNA flanked and separated by universal primers that are used for amplification/sequencing. These two pairs of 18bp are approximately 1 kb apart in the original genome.

These primers are used to amplify this construct. The construct is attached to a 1μm-bead (to learn about bead technologies see www.dynalbiotech.com or see the company's brochure for a quick introduction on surface activated dynabeads). The amplification is done using ePCR – "e" standing for emulsion PCR. Emulsion PCR is standard PCR but done in an oil-water emulsion such that each bead is likely to occupy a single water droplet. All amplified fragments will then attach to the bead, resulting in a bead that has many copies of a single fragment.

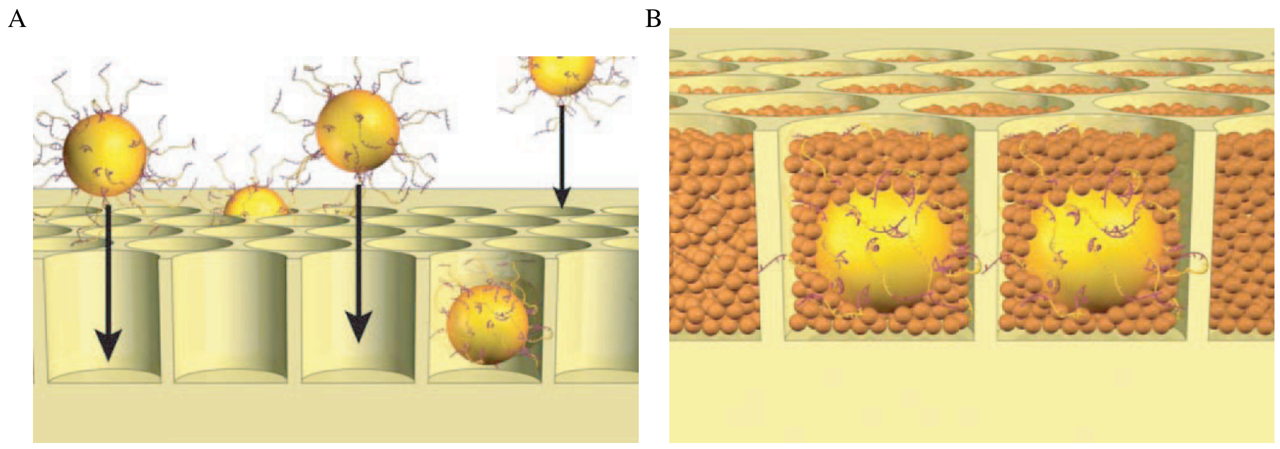


Figure 2.17: To accomplish pyrosequencing templates are attached to beads in individual wells (A) and surrounded by smaller beads with attached enzymes (B; these figures are from Margulies *et al.* 2005 Nature doi:10.1038)

They then use an odd method of determining the sequence in these short fragments. They wish to avoid the cost of acrylimide sequencing. Instead they use oligo's that have specific fluorescent bases at a different positions (for details see their paper). Using these they can determine the sequence of the first 6 bp and the last 7 bp of each of the two 13-mers. A computer then puts these small fragments onto an already known genome.

As a demonstration of this technology they resequenced *E. coli* for SNP's in an evolved strain. They collected 30 Mb after 60 hours of instrument time (2.4 days). This technique is good for resequencing of bacteria. It will need to be enhanced to permit eukaryotic resequencing because the 1 kb distance is not long enough and the sequence determined is too short to correctly place some repeated elements.

The most exciting method to sequence DNA *de novo* has been developed and patented by the company 454 Life Science Corporation. This method is described in the article Margulies *et al.* 2005 Nature doi:10.1038. They make use of a method that can detect the released pyrophosphate when a nucleotide triphosphate is added to a growing chain (Figure 2.2). They use the enzyme sulfurylase to catalyze the PPi to ATP. The concentration of ATP is then sensed making use of the firefly's luciferase enzyme. The amount of light produced is measured by a sensitive CCD (charge-coupled device) camera and should be in direct relation to the amount of PPi released and hence of the the ATP concentration.

The next trick that they use is to amplify individual fragments from a genome. They do this by randomly shearing the genome into fragments. Fragments are then covalently ligated to a four nucleotide marker/primer fragment. Each fragment is then bound to a single bead by ensuring an excess bead concentration. Then a PCR reaction to amplify random fragments using the ligated primers is performed but again it is an ePCR done in an oil/reaction-mixture emulsion such that each bead will uniquely occupy a single droplet. The result is that only one fragment is amplified per droplet and all the amplified copies become attached to a single bead.

The beads are placed in a matrix containing wells that can each hold only a single 28-μm bead (Figure 2.17A). The matrix is 60mm $\times$ 60mm (a square approximately equal to the size of the small side of a credit card) and should contain approximately 1.6 million wells. Smaller beads are added that carry immobilized enzymes required for sequencing and required for the generation of fluorescence (Figure 2.17B).

In successive waves the matrix/slide is washed with a solution of a single nucleotide triphosphate, then a wash solution, followed by the next nucleotide triphosphate and so on. During each wash the fluorescence of the well is measured and sent to a computer. The computer quantitates the level of fluorescence and calls the number of nucleotides of that particular type added in this well. By quickly washing the matrix/slide and measuring the addition of the next nucleotide triphosphate, the technique can carry out shotgun sequencing of an entire genome.

In the Marguilies *et al.* 2005 Nature doi:10.1038 article, the authors demonstrate the technique by resequencing the genome of *Mycoplasma genitalium*. Their run through the instrument took 243 minutes for 42 cycles of reads/washes. The total read lengths after these 42 cycles were on average 108bp long (multiple bases can be added per cycle; e.g. if there are three

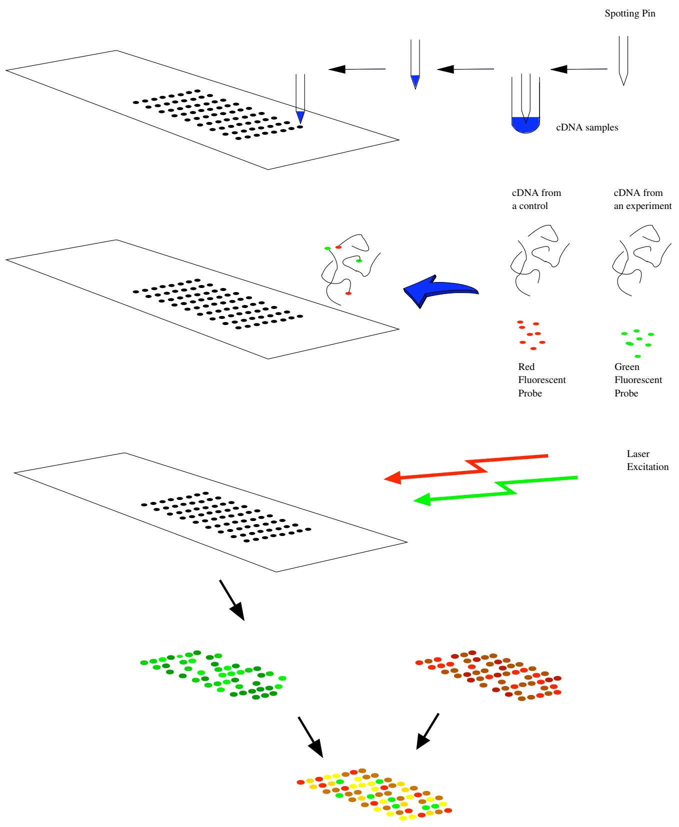


Figure 2.18: An illustration of how a cDNA microarray is created

A's in a row in the template). This run generated over 47 million good quality bases read. Thus it took just four hours to sequence the entire genome (neglecting gap closure). Indeed the authors state that they repeated the whole process eight times yielding a 320-fold coverage of the genome.

There are problems with each of these methods but they are in early stages and each method can be improved. Together they hold the promise that in a few years/decades you will go to your doctor's office, they will take a pin-prick of blood and your complete genetic profile will be determined within hours.

2.6 Other kinds of biological data

While this course is concerned with "Elementary Sequence Analysis" there are several other kinds of biological data that provide a challenge to the quantitatively minded biologist. These include the following three examples but are by no means limited to them. Indeed automation and large scale production in all aspects of biology are becoming more and more common. For example, LIMS, which stands for "Laboratory Information and Management Systems", are a response to this general trend in biological laboratories. Soon most biologists will be all too familiar with bar-code readers as the standard way to track their samples and to retrieve and to store all of the relevant information about those samples directly entered into computers for storage and analysis.

2.6.1 Microarrays

A microarray is the placement of tens of thousands (sometimes hundreds of thousands) of molecular samples into a small array. The goal of most microarray experiments is to analyze gene expression levels. This generally involves only the

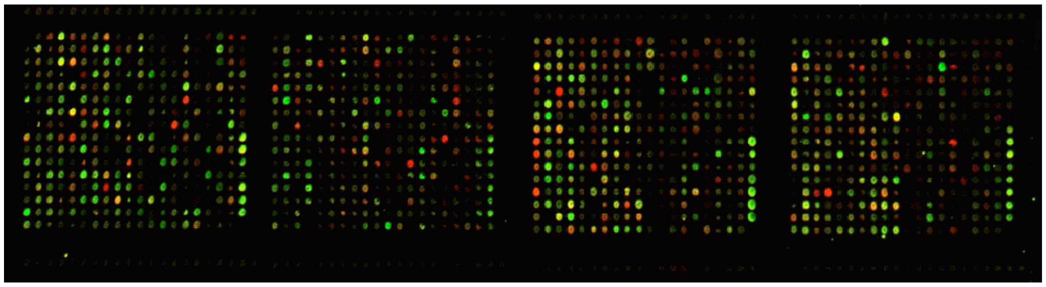


Figure 2.19: Microarray Example

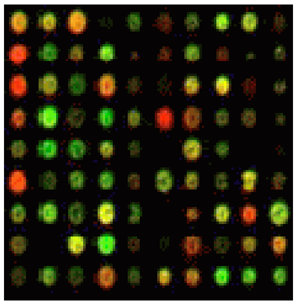


Figure 2.20: An enlarged frame from the previous figure

transcriptome level and hence any discrepancies causes by differential translation of transcripts is not taken into account. There are several types of microarray and the variety grows each year so I can only illustrate one example here. I will illustrate here only the glass slide microarray with affixed cDNA sequences. The concept here is to use a silica coated slide (the same size as a standard laboratory glass slide; 25 mm by 76 mm). With this single slide it is possible to monitor the expression of hundreds of thousands of genes.

The first, and often most difficult step, is to collect samples of genes from an organism. Total genomic sequence projects provide this information but not usually suitable sequences individual to each gene. Another way to get the sequences is by collecting the mRNA from an organism (lets say humans) and to use a reverse transcriptase (an enzyme similar to a polymerase but instead of copying DNA to DNA, it copies RNA to DNA) that will translate the mRNA into a DNA copy (cDNA). Each of these cDNA's are then cloned and amplified to make thousands of copies of each one (usually via a PCR reaction).

These DNA copies of individual genes are then spotted onto a glass slide that is coated with a polylysine solution (or an aldehyde coating, or a variety of other coatings) to which the DNA will adhere. The spotting is accomplished using microscopic pins (or microliter ink jets (from the technology developed around commercial ink-jet printers)) that are precisely positioned (robotically) on the slide. At the same time the identity of individual spots is recorded along with its coordinates on the slide.

The next step is collect mRNA from a tissue of interest and in which the gene expression levels will be measured. Using the microarray technology, absolute expression levels are difficult to determine but relative expression is more practical. So a typical example would be to take mRNA from two different tissues, say a normal somatic human skin cell and cells from a cancerous tumor. A fluorescent probe is attached to each cDNA constructed from the mRNAs. A different fluorescent probe is attached to each sample of cDNAs. The cDNAs are hybridized to the microarray slide in such a way that homologous cDNA molecules will attach and bind to the corresponding spot on the slide (see Figure 2.18).

The fluorescence of each spot is measured by exposing the slide to two different lasers. Each laser emits a specific light frequency that will excite only the corresponding fluorescent probe causing it to emit photons which can be captured by a photometer.

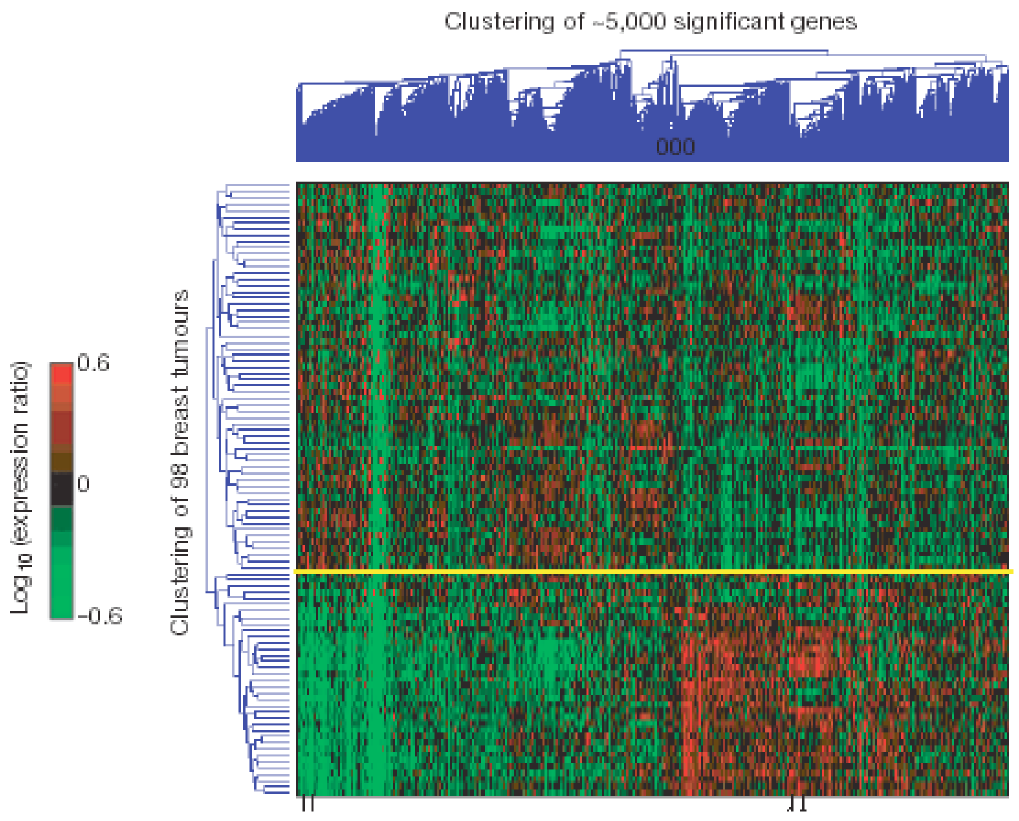


Figure 2.21: An example of genes clustered on the basis of their gene expression patterns (from van 't Veer *et al*. 2002 Nature 415:530)

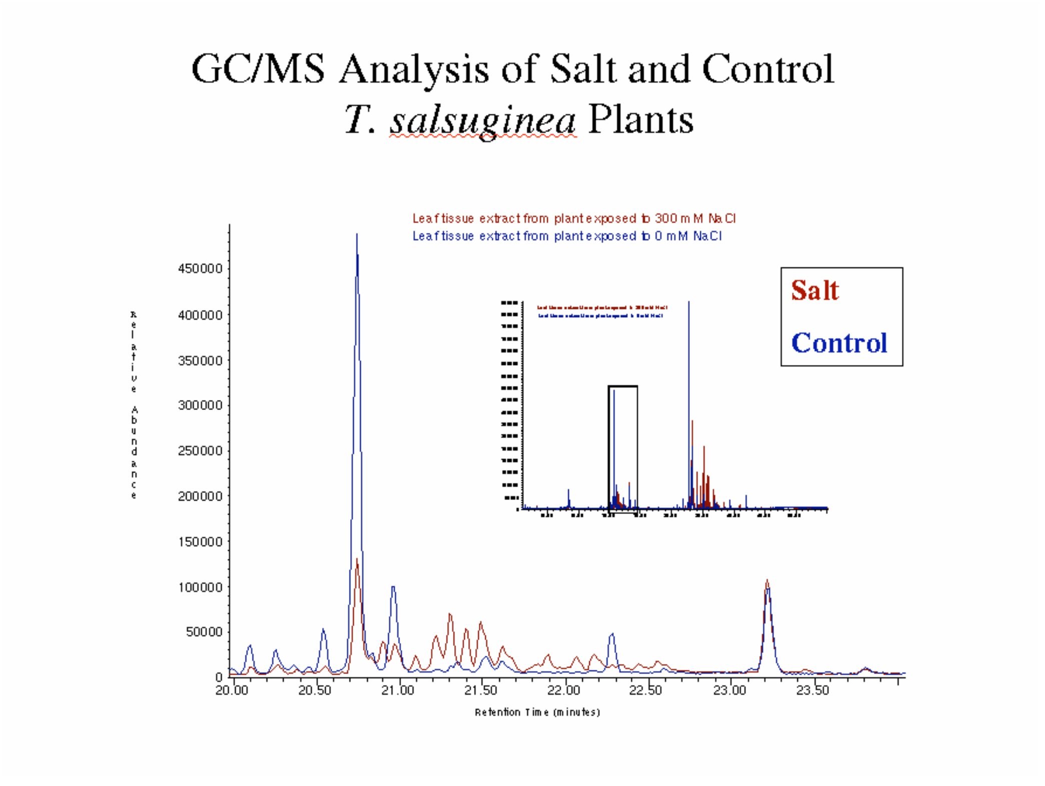


Figure 2.22: Example of a GCMS experiment (chromatograph courtesy of E. Weretilnyk)

By comparing the relative amounts of each fluorescent probe (red and green colours are traditionally used to visualize the fluorescence) a measure of the gene expression levels for each gene can be obtained. In our example and in the combination, a yellow spot indicates genes that are expressed in equal concentration in cancer and normal tissues. A red spot indicates a gene that has been turned on in a cancer cell and a green spot indicates a gene that has been turned off in a cancer cell. The total absence of a spot indicates that it is turned off in both tissues (and doesn't attract non-specific binding) (see Figure 2.19 - 2.20). To aid the analysis of so many genes, a common statistical practice is to cluster genes with similar patterns of expression in a hierarchical fashion (Eisen *et al.* 1998 PNAS 95:14863-14868). Then the scientist can immediately discover which genes are coordinately expressed. An example of such of clustering is shown in Figure 2.21. Here the authors have clustered across the top of the microarray about 5000 genes and on the left, have clustered 98 breast cancer tumors. This indicates that different breast cancer tumors have different collections of genes up/down-regulated. The authors van 't Veer *et al.* (2002 Nature 415:530) were able to show that these tumors responded differently to chemotherapy. This difference is undoubtedly due to these differential gene expressions.

In the end, with a single afternoon's work (after a perhaps more substantial preparation time), a single laboratory could generate more than half a million data points relating to the expression of hundreds of thousands of genes under different conditions. This creates a large analysis task for the bioinformatician.

This short note does not scratch the surface of this new and exciting technology. There are many types of surfaces to coat the slides. Many types of ways to spot the cDNAs. Many ways to select the DNA to be spotted. Many ways to hybridize and label the cDNA. Many ways to analyze the resulting data. On top of all that, as stated earlier, there are many other kinds of DNA microarrays, of protein microarrays, of microbeads, and so on. Their utility will be limited most by our imagination and by the bioinformatician's (e.g. your) ability to analyze the results.

2.6.2 Mass spectrometry methods

Mass spectrometry is another area that has seen rapid advances and which provides a massive amount of information very quickly. A basic mass spectrometer will take a sample of some chemical/compound and then ionize the chemical creating a gaseous spray that is injected into a vacuum chamber which separates the ions on the basis of charge. A detector will then measure how long it takes the ions (time of flight) to reach them – a quantity dependent of their mass and charge. Hence, mass spectrometry is a technique designed to measure the mass-charge ratio of a compound. It can do this incredibly accurately with mass accuracy measurements less than 1 part per million.

This basic technique has several variants. MALDI-TOF is a technique that uses matrix-assisted laser desorption ionization (-time of flight) to create the initial ionization. These units have become relatively inexpensive and their descendants will become common biology laboratory instruments. MS-MS is a tandem mass spectrometer which adds another chamber in front of the detector with nitrogen or argon gas to collide with the ions and break them into constituent pieces. LC-MS combines liquid chromatography with mass spectrometry while GC-MS combines gas chromatography. All to yield further discriminant power. The latest technique is FT-ICR MS (Fourier transform - ion cyclotron resonance). This makes use of fast Fourier transform to build the spectra from an analysis of the complete sample (rather than by combining individual curves) and uses an induced resonance frequency of the ions to aid detection. This instrument is capable of measuring the mass/charge of both large (proteins) and small (metabolite) compounds without extensive preliminary sample preparation.

While these methods are generally used to identify compounds it is also possible to use them to identify protein modifications and even to sequence a protein. This is done by purifying a protein, adding it to these instruments and they will break it into constituent pieces. Since many intermediates of these fragments will also be present and since the Mass/charge of the amino acids is known, a computer can quickly come up with a protein sequence, that when fragmented, would yield the observed bands.

Using these techniques it is possible to determine the presence of any collection of proteins within a complex cellular tissue (this requires knowledge of the pieces first – i.e. a genome sequence). And hence to gather a snapshot over time of a cell's entire proteome. Through multiple, time course analyses it is possible to determine the changing concentrations of all these proteins. And it is possible to examine the metabolomics of a cell – the concentration of smaller constituent molecules such as sugars and alcohols. An example of such a mass spec is shown in Figure 2.22. Here the metabolites from the leaves from a plant grown under high salt conditions are compared to the metabolites from the leaves of a plant grown under normal salt conditions. There are obvious differences that can be utilized to determine how the plant has responded to this salt stress.

Again, there are a large number of peaks in such chromatograms, representing a large number of compounds, influenced/controlled by proteins and genes. Fertile ground for a bioinformatician.

2.6.3 Textual information

Another obvious source of large quantities of information is the scientific literature itself. It has been many decades since I felt that I was doing enough reading (and then only due to my naïveté). Each day all of the scientists of the world are committing their knowledge and their experiments to publishers. Statistics show that many of these papers will not be read by other that one of the original authors and more will be read by just a few experts in the field.

So how can this wealth of information be used to create knowledge? One way is through a process known as data mining. The concept is treat the textual information that has been published in the same way that one would treat a large numerical data set – as an entity that can dissected and analyzed.

A simple example is shown in Figure 2.23. This is just one analysis that can be done. Here the authors (R. Feldman *et al.*) have collected 30,000 abstracts from biomedical articles that have published. The abstracts are freely available to anyone (more on accessing these later). They have queried how often a disease and a gene are mentioned together in these abstracts. By "together", the authors have chosen this to mean "in the same sentence". This yields a diagram of how often different diseases are mentioned together with specific genes. Many of the relationships in this diagram are expected but, of course, if known relationships did not show up, one would question the results. It is the unusual relations that warrant

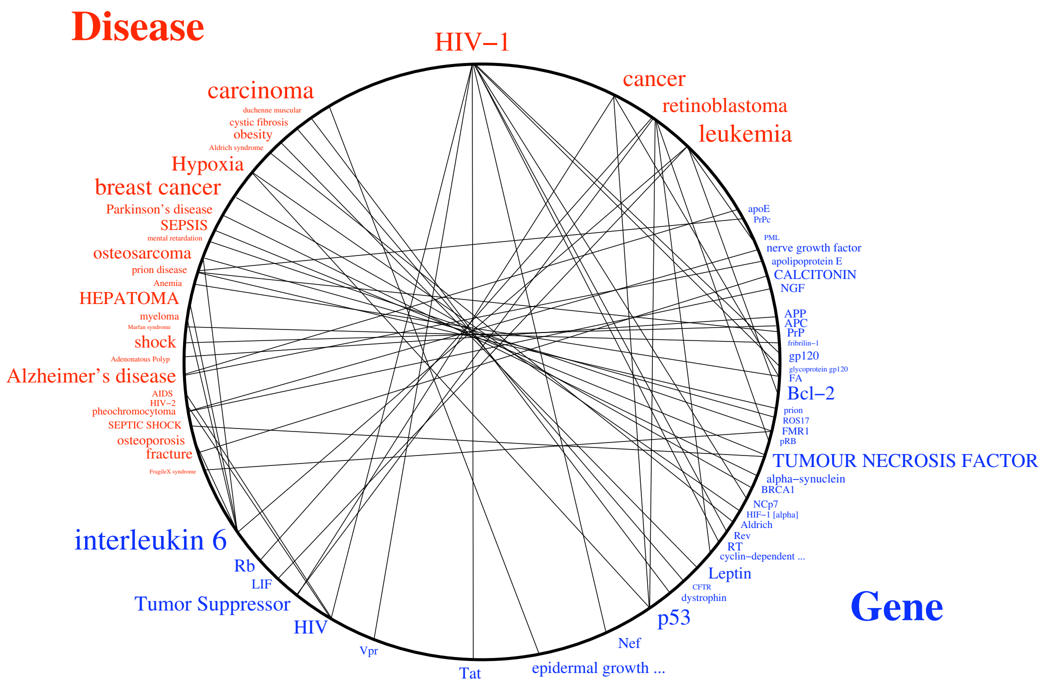


Figure 2.23: Example of literature mining (modified from R. Feldman *et al.* Biosilico 1:69-80 2003)

further exploration.

Obviously this is yet another rich source of information. There are too many other technological advances to discuss them all here. Most of them however, require a scientist with a bent toward quantification and analysis.

Chapter 3

Unix Basics

My purpose in this chapter is to quickly introduce a novice to computers to such a level that they can perform useful work. For this course 'work' is being defined as an ability to enter, to search out, but mostly to manipulate and to analyze sequences and produce information that is biologically relevant. Because of this goal, some of the methods suggested in this chapter (and elsewhere) may not be the shortest or best way to do something. They will however, often be the simplest way, but since the aim is to try and find the lowest common denominator, not even this can be guaranteed.

3.1 UNIX Operating Systems

In the past I have covered three different types of computer operating systems but this time we will reduce it to just one; UNIX systems. Much of the genomic work at major laboratories is done using UNIX systems. The presence of this section is not meant as a recommendation for UNIX. Indeed it has a steep learning curve and if you want a quick answer to one question it is certainly not worth the time. But UNIX is well suited to handling diverse and unanticipated jobs.

Unlike APPLE or WINDOWS computer systems, the UNIX operating system is *in your face* and the philosophy of using the computer is quite different. In WINDOWS the operating system (from a user's standpoint) is merely a platform upon which software runs (and often the operating system gets in the way). In UNIX the operating system is designed to give you tools and make it "easy" for you to design your own tools to do any desired job. This means that you must learn something about the operating system and the many tools that are available. In addition knowing how to use just one tool is seldom sufficient to accomplish complicated tasks. Here I can provide only a brief introduction to the most important concepts and a few commands to get you started.

If you have a graphical interface to the computer, you can interact with a UNIX computer in much the same way as APPLE or WINDOWS. In UNIX this interface is built upon a system called 'X' and hence you will note many pieces of software that come in multiple versions, one with an 'X' attached. For example 'maple' and 'xmaple' are text only and graphical interface versions of the same program. This graphical interface would have most of the same capabilities and, indeed, have a 'look-n-feel' that is very similar to APPLE or WINDOWS machines. In my opinion, however, by learning with the graphical interface you lose much of the power of UNIX because, again, in that case the operating system interface does it all for you. You don't learn how to do it yourself and when you want to do something more than what the interface offers — how would one do this? Is it even possible? The answer is, of course, yes and that is why I will have you painfully suffer through a command line interface.

When you find my ramblings insufficient you can find more more information in any of a thousand books on UNIX. One I can recommend for beginners is "UNIX for the Impatient" by P.W. Abrhams and B.R. Larson, 1992/1997 (Addison Wesley). There are also many introductory packages available 'on line'. You might wish to explore CERN's UNIX help for beginners or their UNIX users guide, UNIX Survival, or UNIX Resources.

UNIX based workstations are built by several companies including IBM, Digital Equipment Corporation (now COMPAQ; shortly to be Hewlett Packard), Hewlett Packard, Sun Microsystems, Silicon Graphics, and others. In addition there are free versions of UNIX available for many processors including INTEL's Pentium processors and for Mac's. Each has a slightly different flavour of UNIX but the minor subset of commands that we will entertain here are constant across platforms. UNIX is an old operating system (approximately 1969) with many capabilities. Along with these capabilities are often complexities.

3.1.1 Logging on/off

When ever attempting access to a UNIX computer the user will be prompted with a request for their user identification (userID) and their password. UNIX has, from the start, been a multiuser computer system and it uses the userID to keep individual users separate and the passwords to provide a first level of security. The prompt to sign on will usually be either a request for USERID:, LOGIN:, USERNAME:, and so on. Your password will not be echoed back to the screen as you type it for obvious security reasons. Passwords should be more than 6 characters long, should include some numbers or symbols, and should never be a word that is found in a search-able dictionary or database.

Upon successful access, the computer may display what kind of computer you are on, may display when and where you last logged in from, and usually will check if you have any new mail. It will then present you with a prompt and await commands. The prompt is customizable and can include such things as the computer's name, command numbers, date, and so on.

If the computer has difficulty recognizing what type of computer you are typing from, you may need to type

```
setenv TERM vt100
      tset
```

These commands will declare the terminal type to be a 'vt100' terminal (a basic generic terminal type).

To exit the computer simply type `exit` from the prompt. Again this is customizable to be anything you desire.

3.1.2 UNIX File System

File Structure:

Files are organized hierarchically. At the base of the file system is a root directory simply referenced by '/'. Underneath this file will be other files that can be of several different types. Under each file that is identified as a "directory", other subdirectories are possible. Different levels of directories, subdirectories, sub-subdirectories, etc. are separated by a '/'. The organization of files on a UNIX system is somewhat standardized but each vendor will do it with their own unique variations. On a typical LINUX machine the top directory contains the files ...

`boot`	
`etc`	
`lib`	
`misc`	
`net`	
`proc`	
`sbin`	
`usr`	- under this subdirectory system-wide programs unique to each installation are normally placed
`bin`	
`dev`	
`home`	- where your personal files will be located
`lost+found`	
`mnt`	- traditionally where filesystems for diskettes, cdroms, etc. will be placed
`opt`	
`root`	
`tmp`	- files needed momentarily by the system or programs
`var`	

All of these files are subdirectories (often called folders on APPLE and PC machines). Most of these subdirectories contain files that are used by the operating system and most of these are files that you should never have to be concerned with. The user directories (where you can put your files) are generally placed under the subdirectory `/home`. So your home directory and where you will automatically be placed when you enter the machine will be `/home/yourid`. This is the standard location for all LINUX computers. On other machines it may be somewhere else. For example, on a Silicon Graphics computer home directories are usually `/usr/people/yourid`.

The other important directory that you should know about is `/usr/local`. This is the location where many files unique to a particular machine are normally placed. For example, in this location my machine has special files to do sequence analysis, phylogenetic analysis, etc. that are useful to every user. So rather than locating them in just one user's files, i.e. `/home/yourid`, they are stored in a central location that all can access — `/usr/local`. You may want to look there to see some of the programs installed on your local machine. The binaries (the actual executable files) for many of these programs are often stored in `/usr/local/bin`.

File Names:

Information is stored in separate files each with unique identification names. Names and extensions are arbitrary and have no reasonable limit to length or characters. The extension may be used to indicate the type of information contained in the file. Although not a requirement, a fortran file will generally have an '.f' extension, a pascal file will have a '.p' extension, and so on. Any character is permitted and even the period '.' is just another character in most file names and is not treated in any special way. Hence a filename such as "test.dat.obj" is quite acceptable. You can have a blank space as part of a filename as in "test dat obj" but this becomes quite confusing for some programs (and people) and hence is not a recommended practice.

The full name of a file will be something like `/usr/local/test.f` Note that a forward slash is used to designate different subdirectories. Here `usr` is a primary (root or top) level subdirectory, `local` is a subdirectory within this and `test.f` would be a file in the `local` subdirectory. There are not different version numbers of a file and if a backup copy is desired it must have a different name.

UNIX operating systems have always been case sensitive. The file `/usr/local/Test.f` is therefore a different file from that listed above. The difference is that a 'T' and a 't' are treated as entirely different characters. This applies not only to file names but also to commands. In general, the default will be lower case. However, if some book or example indicates that one or more letter is upper case, then this case must be copied exactly to achieve the desired result.

File Types:

There are several types of files. The most common types are textual, binary, directory, and symbolically linked files. The latter are files that simply point to other files. There are other types but these are the ones most commonly encountered.

Paths:

Individual files can be specified in several different ways. An absolute name can be given such as `/usr/bin/ls`. Or a relative address could be used such as `../../ls` which means to go up two subdirectories levels and find a file/program called `ls`. If you simply types `ls` the computer will search for a file/program called `ls` in the current directory and if not found, will search through a specified set of directories (you can change this list).

Your Files:

You can create files in your own home directory, but you (usually) cannot create files in another persons home directory. This is dependent on the extent of the privileges you have been given. A 'long' listing of a file's characteristics would look something like . . .

```
-rwxr-xr-x    1 brian    user         133 Sep 13  1996 fnd
-rw-r--r--    1 brian    user        2564 Dec 23 15:04 fumarate.pdb
drwxr-xr-x    4 brian    user        2560 Nov 26 13:55 gde96
-rw-r--r--    1 brian    user       13181 Sep 15  1996 genodb.html
drwxr-xr-x    2 brian    user         512 Nov 12 14:14 genome.sites
-rw-------    1 brian    user        3797 Jul 15  1996 hummingbird.tech
```

The first letter (here either `d` or `-`) indicates the type of file (here subdirectories or textual/binary), the next three letters give the permissions that the file's owner has to read, write or execute that file (in the case of a directory, execute is the ability to enter or view the subdirectory). The next three letters give the read, write execute permissions of anyone in the same user group, and the next three letters given the permissions of anyone on the computer. So in this example only the owner is given permission to change these files, but anybody can read/enter all files/directories except for the last one. The next number gives the number of links this file has. This is followed by the owner of these files, `brian` and the group to which this user belongs, `user`. The size of the files, the date of creation and the name of the file is shown.

Historically UNIX has been a very open system and hence the default permissions are such that any user can usually read the files of all other users. In the past this has created some security problems but these particular problems are comparatively easy to fix. You have the ability to change this default behaviour or to change the permissions of any file.

3.1.3 Commands

Commands have a general structure that consists of the command name, followed by arguments. If the argument begins with a '-' it is called a flag and is used to modify the behaviour of the command. Most commonly used commands are placed in subdirectories that are searched by default and hence their complete location does not have to be specified. A typical program is run by typing

```
cmd -flag argument
```

The name of the program is `cmd` and must be somewhere that the system normally looks for program names or alternatively the path must be specified. The flags (which may or may not be present) will alter the behaviour of the program while the arguments (which may or may not be present) might include files of information to be read from or written to by the program. A few of the most basic, general commands follow.

ls - list files:
This command will give a listing of files in the current directory or in the directory supplied as an argument. To find particular files an asterisk acts as a wild card. A

```
ls a*.f
```

will list all files that begin with `'a'` and end with `'.f'`. A command such as

```
ls -l
```

will give a 'long' listing (such as was shown above),

```
ls -t
```

will sort the listing according to date and so on. There are many other flags and the flags can be combined to achieve many different responses from the same command.

mkdir/rmdir - make/remove directories:
These commands will create/delete a subdirectory with a name supplied as an argument. Only empty subdirectories can be deleted by the default command.

pwd - show present working directory:
This will print out your current location in the hierarchical file system.

cd - change directory:
Change to a subdirectory given as an argument. If no argument is given it will change to your 'home' directory. A command

```
cd ..
```

will move up one directory. A tilde is a useful character to identify home directories. The command

```
cd ~brian
```

will take you to `brian`'s home directory while the command

```
more ~/filename
```

will view the file `filename` in your own home directory.

mv/rm - move/remove files:

These commands can be used to move the location of a file. E.g.

```
mv fnd ..
```

will delete the file `fnd` from the current directory and place it one level higher. This command can also be used to rename files -

```
mv fnd dnf
```

will delete `fnd` and create `dnf`. The command

```
rm fnd
```

will simply delete the file. Be very careful with the use of asterisks and the `rm` command,

```
rm *
```

will delete all files!

cp - copy files:

This command will copy files. It requires two arguments and the first named file is copied to the second file. If the second file is a directory the file is copied to the named directory with a same filename. E.g.

```
cp fnd dnf
```

will create the file `dnf` as a copy of `fnd` but

```
cp fnd gde96/
```

will create a second copy of `fnd` in the subdirectory `gde96`.

cat - concatenate files:

In its simplest form this command will print the contents of a file to the screen. If no argument is given it will wait to accept any input that is typed from the keyboard (terminated by a `<ctrl>d`) and then print this out to the screen. This command is particularly useful when combined with redirection (see below).

more - print a file one screenful at a time:

This command will view the contents of a file supplied as an argument. The screen can be stepped through a file by typing the space bar. A single line is advanced by an 'enter' key or n lines by typing the number n followed by the space bar. For simple text files, the 'b' key will move backwards a screen.

lpr - print:

The `lpr` command will send a file (argument) to a printer. In general most UNIX machines are set up for postscript but individual printers can be set to accept other types of input. Indeed many modern printers will switch 'on the fly' to match the input it is receiving. Postscript is a graphics language that describes the structure of a figures (e.g. a circle of width x) rather than individual points (e.g. actual bit mapped points). In this way it is independent of the resolution of any viewer, simply providing instructions on how to display a figure at the viewer's maximum resolution. You can recognize if a file is postscript by either the filename extension (`*.ps`, `*.eps` or rarely, `*.epsi`) or by its contents. A postscript file will usually begin with something like

```
%!PS-Adobe-2.0
%%Creator: WiX PSCRIPT
%%Title: ramermap1.cdr FROM CorelDRAW!
statusdict begin 0 setjobtimeout end
statusdict begin statusdict /jobname (ramermap1.cdr FROM CorelDRAW!) put end
{}stopped pop
{statusdict /lettertray get exec
       ....
```

If the file is not in postscript and the default printer on your system is an old printer and expects postscript then you must translate the file first. A common (and free) program to do this is `a2ps`. This program takes a file (supplied as an argument) and changes it to postscript, and then automatically pipes it to `lpr`.

For both `lpr` and `a2ps` (and many other programs) the -P*printername* flag will direct the output to the particular printer chosen. For example, the command

```
a2ps -Pps filename
```

will translate the file `filename` to postscript and pipe the output to the printer named `ps`.

3.1.4 Help

All of the above commands have many other abilities. To find out about these abilities there are manual pages stored on the computer. Typing

```
man cat
```

will generate a manual page that describes this command and then passes this page to the `more` viewer.

If you have a graphic interface you can also run `xman` which has more capabilities. There is a move afoot to replace the man programs with a similar but more advanced program called `info` (but I still prefer the old system). You will also often find files under the directory `/usr/doc` or `/usr/share/doc` (along with the directory `/usr/share/doc/HOWTO` which is particularly useful for beginners).

In addition you can search an index of manual pages with `man -k word`. All manual pages that are considered relevant to `word` will be listed. If you want to know more about the `man` command, type

```
man man
```

(of course).

3.1.5 Redirection

To UNIX, the keyboard and the screen are just different types of input/output streams. If desired you can redefine these. For example, you can redirect output of a command such that it is not put onto the screen but rather put into a file. For example the command

```
ls -aF > myfiles
```

will run the command

```
ls -aF
```

Input to the command ends at this point and the part

```
> myfiles
```

instructs the computer to put the output of the command into a new file to be called `myfiles`. So

```
cat fnd > dnf
```

is equivalent (well, . . . close enough) to

```
cp fnd dnf
```

In general,

>	will "send output to . . . "
>>	will "append output onto the end of . . . "
<	will "take input from . . . "

You can also specify a "pipe" symbolized by '`|`' which will take the output of one command and use it as input for another command. It is kind of like a

```
> <
```

command. For example you could enter the command

```
ls /usr/lib | more
```

This will take the file listing of subdirectory `/usr/lib` and give that information to the `more` command.

3.1.6 Shells

Shells are a command interpreter that will be run on all UNIX computers. You can think of the shell as a layer of program through which all of your commands are passed before being processed. Again there are many different shells. The popular ones are the 'sh' Bourne shell, 'bash' Bourne again shell, 'ksh' Korn shell, 'csh' C shell, and the 'tcsh' shell. The latter is the default shell run on the computer you will be using (though the bash shell is generally recommended).

The `tcsh` has several nice capabilities (many shared by the other shells). One of these is filename completion. If you type the beginning of a filename and then type 'tab', the computer will finish this filename. If the request is ambiguous the computer will finish the filename as far as possible and then beep. Typing `<ctrl>d` will display matching filenames up to that point.

This shell also keeps a numbered history of your last commands. You can step through them using the 'up'/'down' arrow keys. The commands can then be repeated or edited. For example, if you type

```
ls
 cat fnd
```

and then type the 'up' arrow twice. This will return you to the `ls` command. Alternately, you could rerun the command by typing just

```
!l
```

An exclamation mark followed by a string will repeat the last command beginning with that string. An exclamation mark followed by a number will rerun that numbered command. The command `history` will give a numbered list of your past commands.

This shell also permits the creation of aliases. Aliases can be set up as follows, type

```
alias dir `ls -aF`
```

Thereafter, typing `dir` will run

```
ls -aF
```

(The ‘a’ flag shows hidden files and the ‘F’ flag adds a ‘/’ to end of a directory, a ‘*’ to the end of a binary, a ‘@’ to the end of a link. Be careful as these are not actually part of the file name). Aliases can be bypassed by preceding them with a backslash. Thus

```
\dir
```

will return `dir` to its original definition and ignore the alias (in our case `dir` is not a defined command and you will get an error message).

3.1.7 Special ‘hidden’ files

Files that begin with a period are called hidden files and are not shown by a default `ls` command. You can ‘see’ them with an

```
ls -a
```

command.

Two of these files are `.cshrc` and `.login`. These files are read (and the commands inside executed) every time you start up a csh (or tcsh) shell and every time you login to get onto the computer. The file `.cshrc` contains many aliases and you can edit this file and add your own. Your default path to search for files and commands can also be defined in this file.

Many programs may define a `.xxxrc` file. They use this file to read and store variables that will be used in the programme.

3.1.8 Background Processes

UNIX users are notoriously impatient. If the computer is taking too long to finish a job, it can be put into ”the background”. This means that the computer will work on this process and at the same time, present to you another prompt for your next command. The way to put a job into the background is to add a ’&’ at the end of the line. A job number will be supplied to you and then the computer will start working on that process. Any output from this command will be sent to the screen or a file as appropriate but it cannot accept interactive input in this state. Thus the command

```
ls &
```

will run `ls` in the background and present you with another prompt. (But hopefully on my machines the `ls` should be done before you get a chance to type in anything else).

Another way to do this, particularly if interactive commands are required only at the beginning of a program, is to type `<ctrl>z` when the process has started its work. This will suspend the job (the job is not killed but nor is it active). To restart the job type “fg” (mnemonic foreground). To put the job into the background type “bg” (mnemonic background).

To check on the jobs that you have running use a `ps` command. This will list the processes that the computer is currently working on. To cancel a job use

```
kill pid
```

where "pid" is the number associated with the job according to the `ps` command. Alternatively

```
kill %n
```

where 'n' is the number of the job given to you when you typed '&'. Finally, to kill a program that is currently executing (assuming it will still accept input from the keyboard), enter `<ctrl>c`.

3.1.9 Utilities

UNIX has many standard utilities that are very useful but I can only talk about a few here. Perhaps the most used is the search utility that will find text in a file/files. There are a family of "grep" commands that perform these searches. The command

```
grep -i Frank /usr/*/address.bok
```

This command will search all subdirectories under `/usr` that have a file named `address.bok`. It searches inside these files for the text `Frank`. The flag `-i` causes the search to be done in a case insensitive fashion. As an example, to see only the process that the machine is running for you rather than all processes, type

```
ps | grep yourid
```

Depending on what you wish to do, there are also `egrep` and `fgrep` variants of this utility. Some other commonly used utilities are `sort, cut, paste, diff,` and `tr`. For information on these see the `man` pages.

3.1.10 Scripts-Languages

Most UNIX systems come with a variety of scripting languages. The simplest of these is usually the scripting language of the shell itself. Beginning a file with the line

```
#!/bin/csh
```

and changing its permissions to be executable (`chmod`), will execute each of the commands following in the file according to that shell (the experts recommend scripting in the `sh` shell rather than `csh`). If the line is missing, the current shell will be used. Any command could be put in these files.

More capabilities are offered by the `sed` script editor and more still by the `awk` programming language. Most systems will also include more extensive programming languages. Since UNIX is built upon C, almost all machines include the C programming language (and more recently, the C++ programming language). Languages with growing popularity are `java`, `perl` and `phython`. Some computers may include fortran (`f77`), pascal (`pc`), `tcl/tk` and others.

For each of these you can obtain limited information from the `man` pages. But to actually learn how to use them, you should find some books or examine instructional web pages (indeed the first prize offered for web-based courses went to a site teaching C++ at MIT.

With respect to bioinformatics and sequence analysis there are two very important resources of which you should be aware. There are libraries of subroutines and objects (bits of computer language code) that you can incorporate into your own programs. These libraries are publically and freely available for all to use. An extremely useful collection of code, the `perl` library can be found at www.bioperl.org. `Java` libraries can be found at www.biojava.org, the Phylogenetic Analysis Library (PAL) can be found at www.cebl.auckland.ac.nz/pal-project/; an effort being led by Dr. A. Drummond and Dr. K. Strimmer, and a collection of algorithm libraries for bioinformatics designed to be fast and efficient can be found at bioinformatics.org/ALiBio.

3.1.11 Editors

There are many editors available both for free and commercially. If you have used `pine`, you have used the `pico` editor. The most common and ubiquitous editor is `EMACS`. `EMACS` is available on most UNIX computers but, in the past, it has been rather picky about the terminals it will talk with.

An older, more basic and works from anything editor is called `vi`. This editor was designed to work without the aid of a mouse and to permit easy mapping of keyboards to accommodate multiple hardware manufacturers.

This editor is invoked by typing

```
vi filename
```

Again, if the file does not exist then it will be created by this command. There are two modes to the basic editor – command mode and insert mode. In the former mode, everything typed from the keyboard is treated as a command while in the latter mode, everything typed from the keyboard is added to the file. To change from insert mode to command mode use the 'escape' key. There are several ways to change from command mode to insert mode. To insert text after the cursor hit the 'i' key while in command mode. To append text to the end of a line use 'a' while in command mode. Use the 'x' key to delete characters under the cursor. In command mode 'dd' will delete entire lines, so do not idly type keys while in command mode. The arrow keys can be used to move around (albeit slowly). This editor has many commands and many capabilities (see Table 3.1 or see a book on UNIX for more). To exit the editor, move to command mode by hitting 'escape' and then type 'ZZ' (note upper case!). Your work is automatically saved but a backup of the state of the old file is not generally made.

On my computer `vi` is again aliased and in reality typing `vi` will invoke `vim` instead. This is a modernized version of `vi` which is actively being developed (2002). This project has included mouse support if you have proper terminal definitions for mouse standards (e.g. an `xterm` interface). It also has a graphic interface started by `gv` (another alias, actually `gvim`). This update includes many features, the best being easy customization and simple programming abilities. To find out more check out the vim web site, type

```
:help topic
```

inside the editor (note the preceding colon), or examine the documentation files (locally in the subdirectory

```
/usr/local/doc/vim/vimguide_booklet.pdf
```

and

```
/usr/share/doc/vim-common-6.3/doc.
```

3.2 A few internet facts

There are many things on the internet that are useful and a great deal that is not useful (to anyone!). The pessimists would claim that the best days of the internet are behind it. A few years ago, high speed networks existed between universities and the public did not have ready access to these networks and even most university students did not avail themselves. This meant an open highway upon which scientists could transmit their material at rapid rates (the equivalent of sitting on the proverbial German autobahn in a hot car with no other cars – or cops in sight). Today, even little children are aware of the net and can use it to play games (the equivalent of sitting in that same hot sports car, stuck in gridlock, in New York traffic at rush hour with accidents in all directions). On the other hand, the optimists would claim that now far more material and databases exist to be used and would never have been created if people couldn't access them.

Note that messages are propagated throughout the subnet that you are attached to. Hence your connection may slow or speed up as your colleagues at work transfer large files or head off to coffee breaks. The type of wires that your local internet is composed of will have the major determinant of the speed you actually achieve. In terms of security, it also means that your files are transferred and are accessible to anyone on the same network – software programs called "sniffers" are

Table 3.1: A few vi Editor commands

ESC	return to command mode.		
i	change to insert mode.	a	append to the right of cursor.
:w	save file	ZZ	save the file and exit.
:q	quit	:q!	abort edit.
w	move right a word.	b	move left a word.
H	move to top of screen	M	move to middle of screen
L	move to bottom of screen	`^F`	scroll one screen forward
`^B`	scroll one screen backward		
cw	change word	cc	change line
˜	change case	C	change rest of line
s	substitute character under cursor		
u	undo last command.	U	undo all changes to line
x	delete character	dw	delete word
dd	delete line	:5,10d	delete lines 5-10
:set nu	show line numbers.	:set nonu	hide line numbers.
11yy	copy 11 lines into buffer	11dd	delete 11 lines into buffer
p/P	put buffer below/above line	:1,2 co 3	copy lines 1,2 to after line 3
:1,2 m 6	move lines 1,2 to after line 6		
G	go to last line.	11G	go to line 11.
/str	search for str.	?str	search backwards for str
n	find next occurrence		
:g/search/s//replace/g	find and replace		
:g/search/s//replace/gc	same but consult		
v	locally defined to reformat paragraphs (it is mapped to !}fmt)		

available to access these files. Hence do not assume that your sensitive letters/data cannot be obtained by others unless you use security software.

3.3 News Groups

3.3.1 USENET

USENET is an old electronic bulletin board. It was started in the late 1970's between two UNIX computers in North Carolina. It quickly spread to most UNIX machines across the country and then ported to other computers. The access and programs that provide access are generally free (once you have an internet connection). However, the USENET feed can require a great deal of disk space to hold recent traffic and may require maintenance.

For access, either use web connections or telnet to a computer that provides this service, e.g. McMail and logon, then type "trn" at a McMail prompt. The version on McMail is a "threaded read news" program. The "threaded" part means that it will sort the messages/posts such that replies, further queries on the same subject and so on, will be listed as a sub-heading from the original message. This creates a hierarchical listing of the messages that is much more useful than a loose collection of the messages. The "trn" program will lead you through most of the things required (simply answer yes, no (quit) or default (space) to most prompts).

To get a listing of all news groups available on USENET type "l" (lower case el) at the prompt. This will list all unsubscribed news groups (add an "l search" to find "search" news-groups). I have never been able to sit through the entire listing of these groups (and more are added every day). To subscribe to one of these news groups type "g news_group_name" (g for get). For help on the program "trn" type "h" from the trn prompt.

3.3.2 BIONET

BIONET is an electronic bulletin board originally from Intelligenetics Inc. BIONET was formerly the National Computer Resource for Molecular Biology, run by Intelligenetics on behalf of NIH. BIONET later merged with SEQNET in the U.K. and Biotech at UMDC to form the international BIOSCI network which became a collaborative effort between IntelliGenetics, SERC labs in Daresbury, U.K., the Biomedical Center at the University of Uppsala, Sweden and the University College at Dublin.

The BIONET is part of the USENET. Hence, you can gain access to all of the BIONET groups through the USENET. However BIONET is also available as a e-mail mailing list (I believe that this is how it was originally set up) or through the web. To subscribe to BIONET via e-mail send mail to "BIOSCI@net.bio.net".

The BIONET includes many discussion groups on diverse topics. It includes things such as the pre-publication release of the Table Of Contents from many journals. Generally the Table of Contents are available several months before the journal will be actually printed. There are discussion groups for RAPDs, for evolution, for immunology, for virology, for forestry, ... take your pick.

By subscribing to this group you will find out the meaning of junk electronic mail.

3.3.3 IRC

For those that require instant gratification there are chat groups and instant messenger services. These will provide messages transmitted to all individuals in the group virtually as they are typed.

IRC (Internet Relay Chat) provides a way of communicating in real time with people connected to other computers running IRC servers. The server relays information to and from other servers (computers) on the same net. The IRC consists of various separate networks (or "nets") of servers/computers that allow users to connect to IRC. The largest nets are EFnet (the original IRC net, often having more than 32,000 people at once), Undernet, IRCnet, DALnet, and NewNet.

Generally, the user (such as you) runs a program (called a "client") to connect to a server on one of the IRC nets. Once connected to an IRC server on an IRC network, you will usually join one or more "channels" and can converse with others there in real time. On EFnet, there often are more than 12,000 channels, each devoted to a different topic. Conversations may be public (where everyone in a channel can see what you type) or private (messages between only two people, who may or may not be on the same channel).

Several UNIX clients include `Jabber` (an XML based client with services for pagers, phone networks, and web-based services), `GnomeICU` (based on the Gnome desktop), and `Xchat`.

3.4 Some older communication means

3.4.1 gopher

An old protocol. Section deleted.

3.4.2 Modem communication

An old way to communicate between computers is to use a phone line. Does anyone use a raw communication means anymore? To make use of the phone lines a modem and some software to drive the modem are required. This method of communication is rather slow but most people have access to a telephone. One of the oldest programs, a common and a free program is known as KERMIT. KERMIT is very versatile and a cheap solution. This program will open communications and permit simple data transfer between your PC and another computer. There is also a program termed ‘ppp’ that is usually distributed for free with most UNIX computers and will create what is called a ‘point to point protocol’ for communications over telephone lines. For further information on these see the `man` pages on any UNIX based computer.

3.4.3 telnet

This is a program that permits communications across the internet. Many internet service providers will feature `telnet`. In order to connect to a remote computer using this program simply type

```
telnet remotehost
```

where `remotehost` is the name of the remote computer (or it’s numbered identification code). If access is permitted to the remote computer, you will be presented with a prompt for your userID and then for your password. From a `telnet` prompt you can also enter

```
open remotehost
```

or

```
quit
```

These are the only two commands that a beginner really needs.

Should you use `telnet`? **No!** Remember I stated that internet traffic flows freely through the entire subnet – it is not solely directed to a single machine and inaccessible to all others. This means that others on the same subnet can read your internet traffic. It also means that they can read the password you transmitted to the remote host. Since there are now many unscrupulous people on the internet (unlike the original days when all was open) this is a serious security hole.

As a result of this my computers do not permit `telnet` access and if there is a machine that you are using that does permit it, beware that you are using something that is basically a public resource and don’t send that machine any passwords or any sensitive data or letters to your lovers.

3.4.4 ftp

The `telnet` program does not permit file transfer. If you need this capability you should use another program such as `ftp`. This stands for file transfer protocol. This is one of the protocols normally subsumed in a web browser (with user anonymous). Again this program is invoked by

```
ftp remotehost
```

The commands that a beginner should know about are

- `open remotehost` – opens a connection to another machine
- `user yourID` – permits access (along with appropriate password).
- `user anonymous` – free access (it is customary to supply your e-mail address as password).
- `cd/ls` – the same commands as explained above to change directory and to list files.
- `send/put` – transfer local files to the remote computer.
- `get` – transfer remote files to local computer.
- `mput/mget` – multiple put/get which permit the use of wildcards, such as an asterisk, to transfer many files at once.
- `binary` – set file transfer mode to binary rather than the default ASCII. This is necessary to transfer many files and will cause fewer problems if you always enter `binary` mode.
- `quit` – exit program.

Should you use `ftp`? **No!** Same reason, passwords and other information are send across the internet and can be snooped. Anonymous ftp is fine (basically what a web browser will do) because no password is transmitted (though the information transferred can still be read).

3.4.5 ssh

The programs `ssh`, `scp` and `sftp` are programs that you **should** use in preference to `telnet` and `ftp`. These programs are replacements for `rsh` and `rcp`, where the more logical ‘r’ stood for remote (hence to open a shell on a remote computer – rsh, or to do a remote copy – rcp). The ‘s’ stands for secure and the difference between `rsh` and `ssh` is that the information is encrypted before it is sent across the internet (including encryption of any transmitted username and password) and then de-encrypted ‘on the fly’ at the remote computer location. The encryption is different each time a different connection is made and is difficult (nothing is impossible) to crack. To use `ssh` simply type

```
ssh remotehost
```

and you are off. You might see some information about the nature of the encryption, about exchange of keys and so on. For `scp` the commands are the same as `cp` except that you can use a ‘:’ to separate file names from machine names. For example,

```
scp george@california.edu:stuff/filename1 frank@newyork.edu:filename2
```

will copy a file named `filename1` in subdirectory `stuff` under the home directory (default) of user `george` on a machine in California to user `frank` on a machine in New York even if you are a third user sitting on a machine in Canada (of course you will have to have passwords to all three accounts). The commands for `sftp` are the same as those for `ftp`. See the `man` pages for further information on these protocols.

There are other programs that can be saved for later.

3.4.6 Mail

I have been stressing in this course the utility of a command line interface to UNIX. There are of course, therefore, character based interfaces to e-mail. The two most popular character based interfaces are `pine` and `mutt`. Each are easy to use and more importantly you can, with ease, include them in programs that you write (so for example, mail to Frank the results of the program and send the ancillary data generated to Susan; or send of formated emails to specific people based on the program results). These programs have some simple properties that make them very easy to use. As just one example to send a mail message with hundreds of attachments (and annoy your friends) simply enter the command

```
pine user@remote.machine -attachlist directory/*
```

This command will send an email to a particular user on a remote machine and will attach to this email all of the files in the directory `'directory'`.

3.5 Web Search Engines

Most browsers come with specific search engines and there are a variety of search engines available from specific sites. These search engines do not begin a search of the internet as the request comes in. Instead most use electronic robots that roam the internet to constantly search sites on the web and then catalog the results in a database. It is the database that searched by your request.

In general, these requests miss a large number of sites and most search engines will give thousands of hits to even simple and very specific searches. Of course most of these are all false positives. Indeed, one report indicated that a search for a phrase from an obscure poem yielded 29 million hits from the Go.com search engine.

The two best search engines that were reported in a review from August 2000 were Google (now used by Yahoo) and FAST (AllTheWeb.com, Lycos – now partnered with FAST). The advantages of these engines changes over time as they are constantly improved or commercialized. It is worthwhile to search out the hints, tips, secrets, advanced searches. A regular search will rarely find the answer you are looking for in a rapid and efficient manner. Even the best advanced features however, are a long way from the power of "regular expression" searches.

3.6 Servers

3.6.1 List Servers

A listserver is another kind of message system. In this case, a computer program takes care of everything. The program has the ability to send and receive mail and will act upon some specific requests. Any mail sent to the listserver is then passed on to everyone else that it has on its list. The two major commands for a listserver are "subscribe listname your_name" and "unsubscribe listname". Both of these should be mailed to"listserv@xxxxxxxxx". To access some of the things stored by the listserver try ...

"review listname"

"index listname"

"send listname filename"

"help".

To actually post something to the entire group send mail to "listname@xxxxxxxxxxx" where the x's indicate the address of the host machine.

Again there are many more commands available. The number of listservers is very large. As only a flavour, there is a server for statistics programs - send "SEND INDEX" to

statlib@temper.stat.cmu.edu

If this general realm of statistics is not your favourite cup of tea, you can concentrate on just classification and multivariate data analysis. Send a message to the CLASS-L listserver at SUNY or see the special subdirectory of the above file server.

3.6.2 Mail Servers

Mail/file servers are computers that are setup to automatically receive electronic mail and respond to a specific series of commands that may be contained within these files. Almost all file servers will respond to the command

```
help
```

and this is usually a good place to begin. This will usually describe the commands that are available and will describe how they should be used with this particular computer. Unlike listservers, the commands for each file server may be different on every machine. The major commands that most machines will recognize are help, get, and dir. But many (such as the EMBL server below) will not recognize any more commands. Since these are machines responding to a mail message, the input must be strictly formatted. A blank "subject" line is usually a good idea since this may or may not be interpreted as an extra command.

While many sites have moved over to a web-based form of interaction, there are still many sites that still make use of the file server interface. Many sites that have a web-based will also support the dual interface. As an example, we will consider in subsequent sections the *query* mail server that provides mail access to the stored repository of known DNA sequences.

Chapter 4

Databases

4.1 Introduction

Molecular biology has undergone amazing advances in the last twenty years. We can now sequence DNA and proteins in most any laboratory in the country. Indeed it is sometimes even given to undergraduate students as an laboratory exercise. Most universities don't do this but this is because of the use of radioisotopes rather than the difficulty of the technique. The ability to rapidly and easily sequence DNA has also lead to a shift in the way that science is now done. It has become easier to simply sequence the gene (and more preliminary information can be often be gained this way) than to carry out a sophisticated and well thought out experiment.

These advances have lead to the establishment of genome projects. These are specific projects to set up laboratories to sequence DNA in an efficient way rather than having each laboratory do the sequencing *in house*. The largest of these projects was the human genome project to which the United States government alone committed $3,000,000,000.00 (that is three billion!). Other governments of the world are also supporting their own projects. Additionally, more organisms than just humans are being examined and are making fast progress.

Computer technology has also undergone an amazing advance in the last twenty years. It is now unusual for scientists not to have a computer on their desk. Additionally, though somewhat contrary to popular use, these computers are not simply fancy typewriters. They have many capabilities beyond word processing and can deal with a large amount of information. They are also capable of doing analyses that are beyond the computational ability of any scientist.

One of the other major advances in computer technology has been in connectivity. Most computers can now be connected to a network that permits access to other computers all over the world. This means that when you switch on a computer, you are not simply turning on a typewriter, but rather, opening a window through which you can pass to any where in the world. This permits anyone with a computer to access databases of all kinds - if they know how. The purpose of this section is to provide you an entry point to this knowledge.

None of the genome projects, nor most of the other projects that create databases, would have been funded if their research was kept private. Indeed an openness about research results has been a long standing principle that has guided science. It is oft quoted that "the experiment is not finished until it has been published". Publication has been the traditional method of permitting worldwide access to research results. However, the retrieval of this information can be a labour intensive practice that requires great skill. Here, I am mainly referring to simple factual data rather than experiments that require interpretation. To accumulate this factual data and to make use of it is often difficult. With computer databases, however, this data is as accessible to you as it was to the expert that compiled it. You can bring the data directly to your desktop in its entirety, cut/paste the pieces you want, and analyze it according to your fancy. An article by W. Gilbert in NATURE suggests that this combination of advances will lead to a shift in the way science will be done in the future.

Towards a paradigm shift in biology.
W. Gilbert NATURE 349:99 1991.

The steady conversion of new techniques into purchasable kits and the accumulation of nucleotide sequence data in the electronic data banks leads one practitioner to cry, "Molecular biology is dead - Long live molecular biology!".

There is a malaise in biology. The growing excitement about the genome project is marred by a worry that something is wrong - a tension in the minds of many biologist reflected in the frequent declaration that sequencing is boring, and yet everyone is sequencing. What can be happening? Our paradigm is changing.

Molecular biology, from which has sprung the attitude that the best approach is to identify a relevant region of DNA, a gene, and then to clone and sequence it before proceeding, is now the underpinning of all biological science. Biology has been transformed by the ability to make genes and then the gene products to order. Developmental biology now looks first for a gene to specify a form in the embryo. Cellular biology looks to the gene to specify a structural element. And medicine looks to genes to yield the body's proteins or to trace causes for illnesses. Evolutionary questions - from the origin of life to the speciation of birds - are all traced by patterns on DNA molecules. Ecology characterizes natural populations by amplifying their DNA. The social habits of lions, the wanderings of turtles and the migrations of human populations leave patterns on their DNA. Legal issues of life or death can turn on DNA fingerprints.

And now the genome project contemplates working out the complete DNA pattern and listing every one of the genes that characterize all of the model species that biologist study - ourselves even included.

At the same time, all of these experimental processes - cloning, amplifying and sequencing DNA - have become cook-book techniques. One looks up a recipe in the Maniatis book, or sometimes simply buys a kit and follows the instructions in the inserted instructional leaflet. Scientists write letters bemoaning the fact that students no longer understand how their experiments really work. What has been the point of their education?

The questions of science always lie in what is not yet known. Although our techniques determine what questions we can study, they are not themselves the goal. The march of science devises ever newer and more powerful techniques. Widely used techniques begin as breakthroughs in a single laboratory, move to being used by many researchers, then by technicians, then to being taught in undergraduate courses and then to being supplied as purchased services - or, in their turn, superseded.

Fifteen years ago, nobody could work out DNA sequences, today every molecular scientists does so and, five years from now, it will all be purchased from an outside supplier. Just this happened with restriction enzymes. In 1970, each of my graduate students had to make restriction enzymes in order to work with DNA molecules; by 1976 the enzymes were all purchased and today no graduate student knows how to make them. Once one had to synthesize triphosphates to do experiments; still earlier, of course, one blew one's own glassware.

Yet in the current paradigm, the attack on the problems of biology is viewed as being solely experimental. The 'correct' approach is to identify a gene by some direct experimental procedure - determined by some property of its product or otherwise related to its phenotype - to clone it, to sequence it, to make its product and to continue to work experimentally so as to seek an understanding of its function.

The new paradigm, now emerging, is that all the 'genes' will be known (in the sense of being resident in databases available electronically), and that the starting point of a biological investigation will be theoretical. An individual scientist will begin with a theoretical conjecture, only then turning to experiment to follow or test that hypothesis. The actual biology will continue to be done as "small science" - depending on individual insight and inspiration to produce new knowledge - but the reagents that the scientist uses will include a knowledge of the primary sequence of the organism, together with a list of all previous deductions from that sequence.

How quickly will this happen? It is happening today: the databases now contain enough information to affect the interpretations of almost every sequence. If a new sequence has no match in the databases as they are, a week later a still new sequence will match it. For 15 years, the DNA databases have grown by 60 per cent a year, a factor of ten every five years. The human genome project will continue and accelerate this rate of increase. Thus I expect that sequence data for all of the model organisms and half of the total knowledge of the human organism will be available in five to seven years, and all of it by the end of the decade.

To use this flood of knowledge, which will pour across the computer networks of the world, biologists not only must become computer- literate, but also change their approach to the problem of understanding life.

The next tenfold increase in the amount of information in the databases will divide the world into haves and have-nots, unless each of us connects to that information and learns how to sift through it for the parts we need. This is not more difficult than knowing how to access the scientific literature as it is at present, for even that skill involves more than a traditional reading of the printed page, but today involves a search by computer.

We must hook our individual computers into the worldwide network that gives us access to daily changes in the database and also makes immediate our communications with each other. The programs that display and analyze the material for us must be improved - and we must learn how to use them more effectively. Like the purchased kits, they will make our life easier, but also like the kits we must understand enough of how they work to use them effectively.

The view that the genome project is breaking the rice bowl of the individual biologist confuses the pattern of experiments done today with the essential questions of the science. Many of those who complain about the genome project are really manifesting fears of technological unemployment. Their hard-won PhDs seem suddenly to be valueless because they think of themselves as being trained to a single marketable skill, for a particular way of doing experiments. But this is not the meaning of their education. Their doctorates should be testimonials that they had solved a novel problem, and in so doing had learned the general ability to find whatever new or old techniques were needed; a skill that transcends any particular problem.

There is now a new concept of public data. Everyone that desires access can retrieve this data. This includes not only scientists and medical practitioners but also private companies and even members of the general public. The data is also raising a large number of ethical problems that have not been considered.

These advances have combined to create a new field of science. This is called bioinformatics (along with its relative – medical informatics). It is, basically, a mixture of computer science, mathematics, and biology. It combines aspects from all three fields to study the methods and the problems associated with the task of bringing information to a researcher, sorting this mass of information in a meaningful way, and then analyzing it.

Our concern in this section will be focused on the databases of relevance to molecular biology. However, you should be aware that this is but the tip of the iceberg – there are many databases of many natures. There are other biological databases such as one on RAPD data, some of a biochemical nature, such as one on enzyme kinetics, some of a more general nature, and some just plain weird.

The major databases for molecular biology are centered around the molecular sequence databases. The genome projects supplying these databases promise to yield the greatest mass of data that biology has ever seen. The human genome alone covers 3 billion nucleotides. In February of 2001, the completion of the human genome draft sequence was jointly announced by the private company Celera Genomics and the publically funded Human Genome Project. This represents an enormous accomplishment and will probably represent the biggest achievement since the discovery of the structure of DNA.

Figure 4.1: The completion of a first draft of the human genome was announced in February of 2001

But the human genome is only the beginning. There are many other eukaryotes whose genome has been sequenced and even more in the pipeline. Currently in the public domain there are many more whole eukaryotic genome shotgun component projects nearly completion including the fish *Fugu*, the puffer fish, the nematode *C. briggsae* and the mosquito *Aedes* (to compliment *Anopheles*), the fruit fly *Drosophila pseudoobscura*, cow, potato, and maize with more to come.

This mass of data also presents many problems – how do you store all of this information, how do you access it, and move it? The rate of accumulation of sequence data is exponentially growing. This has been partly due to the fact that the technology to carry out DNA sequencing has rapidly advanced. Today, almost the entire job can be carried out by robots – from an input of tissue, the robots can automatically extract the DNA, amplify regions of interest, and prepare sequence cocktails. These are then loaded onto the gels of automatic sequencing machines. These machines will run the gels, a laser will scan the gels and calculate the DNA sequence. Finally, the sequence is automatically entered into a computer and the computer will automatically assemble the fragments and may also do some preliminary analysis. In addition, the number of laboratories that routinely sequence DNA has also increased.

The result of this increased activity is shown in Figure 4.2. There are over 104 billion nucleotides in the database. Most of this data is annotated but in 2004 EMBL has included in its data release nucleotides of mostly unannotated whole genome shotgun data. Over 52 billion of these nucleotides come from these raw data sources. The current official EMBL release 83 yields over 94 billion nucleotides and is the data that is plotted Figure 4.2. You can predict that (if the rate of growth remains constant and continues), by the year 2010 (only a few years away) there will be more than 700 billion (that is, it should in less than five years, increase by more than 7 fold in comparison to its current amount!!). Obviously an exponential growth cannot continue (physical laws prevent this). However, I have stated this every year since I first taught this course in 1990 and, still, every year I have been proven wrong (comparable figures from other years are 1993, 1995, 1998, 1999, 2000, 2001, 2002, 2003, 2004). (How can one be sure that it is exponential? Well a simple way is to see if the data is a straight line when plotted on a log scale. Take a look at a log plot and see if you think that these data are linear.) Regardless, all of this mass of data is open for analysis and is a rich research field.

There are three major nucleotide sequence databases. These are EMBL (European Molecular Biology Laboratory), NCBI (the U.S. National Center for Biotechnology Information) and DDBJ (the DNA Data Bank of Japan). Each of these databases attempt to collect all of the known nucleic acid (DNA/RNA) sequences. The sequences were collected from published sources and most journals now require submission of the sequences to a database before publication is permitted. Many sequences are directly deposited into the databases and will not be published in any other form. In addition to the sequences, the databases also contain many other useful bits of data, including (but not limited to) organism, tissue, function, and bibliographic information.

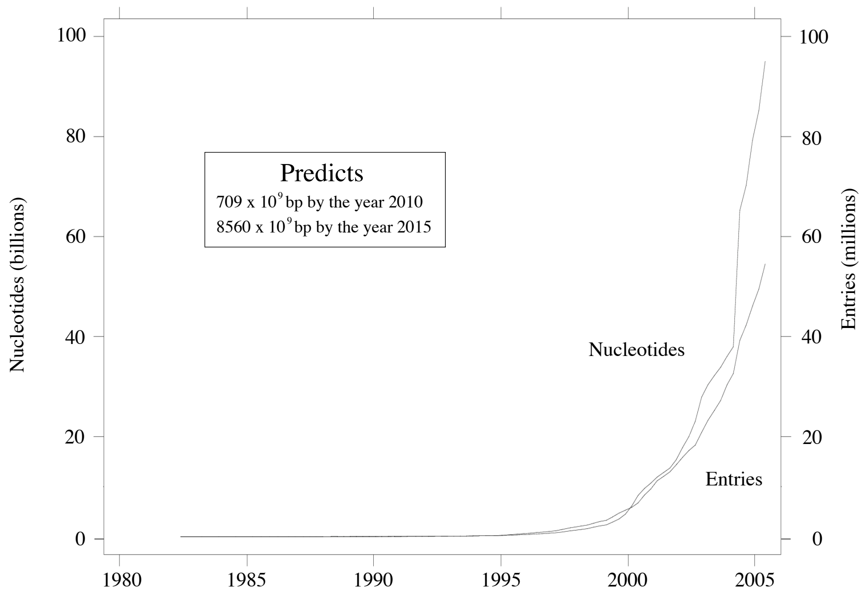


Figure 4.2: The growth of the EMBL database

All three of these organizations are in electronic contact with each other and exchange sequence information daily. Hence, you need not worry that one database might not have the sequence of interest but a search of some other database would have it (at least in theory, anyway).

The following sections are intended to give you a flavour of the database contents.

4.2 N.C.B.I.

The easiest way to explain what is contained in the database is to examine an actual entry from the data base. This is shown below. This example contains the nucleotide sequence of the first exon of the human lung adenocarcinoma (PR310) c-K-ras oncogene.

Note that the actual sequence information provided at the end of the entry may be, as in this case, only a small fraction of the total data entry. NCBI organizes its entries onto several lines each of which begin with a special header. The first header and that which always begins the entry is the LOCUS name. This provides a identifying code word (in uppercase) to be associated with this entry. It also gives the length of the sequence and the date the sequence was entered or last modified.

Example Entry #1: GenBank/NCBI for HUMCKRASA

```
LOCUS       HUMCKRASA     450 bp ss-mRNA            PRI       15-SEP-1990
DEFINITION  Human PR310 c-K-ras protein mRNA, 5' end.
ACCESSION   M35504
KEYWORDS    c-K-ras protein; c-myc oncogene.
SOURCE      Human (patient PR310) lung carcinoma, cDNA to mRNA.
  ORGANISM  Homo sapiens
            Eukaryota; Animalia; Chordata; Vertebrata; Mammalia; Theria;
            Eutheria; Primates; Haplorhini; Catarrhini; Hominidae.
REFERENCE   1  (bases 1 to 450)
  AUTHORS   Yamamoto,F., Nakano,H., Neville,C. and Perucho,M.
  TITLE     Structure and mechanisms of activation of c-K-ras oncogenes in
            human lung cancer
```

```
   JOURNAL   Prog. Med. Virol. 32, 101-114 (1985)
   STANDARD  full automatic
 FEATURES             Location/Qualifiers
      CDS           1..>450
                  /note="PR310 c-K-ras oncogene"
                  /codon_start=1
                  /translation="MTEYKLVVVGAGGVGKSALTIQLIDNHFVDEYDPTIEDSYRKQV
                  VIDGETCLLDILDTAGHEEYSAMRDQYMRTGEGFLCVFAINNTKSFEDIHHYREQIKR
                  VKDSEDVPMVLVGNKCDLPSRTVDTKQAQDLARSYGIPFIQTSAKTRQ"
    source        1..450
                  /organism="Homo sapiens"
   pept        1  >   450     PR310 c-K-ras oncogene
 BASE COUNT      155 a     71 c    106 g    118 t
 ORIGIN
         1 atgactgaat ataaacttgt ggtagttgga gctggtggcg taggcaagag tgccttgacg
        61 atacagctaa ttgacaatca ttttgtggac gaatatgatc caacaataga ggattcctac
       121 aggaagcaag tagtaattga tggagaaacc tgtctcttgg atattctcga cacagcaggt
       181 catgaggagt acagtgcaat gagggaccag tacatgagga ctggggaggg ctttctttgt
       241 gtatttgcca taaataatac taaatcattt gaagatattc accattatag agaacaaatt
       301 aaaagagtta aggactctga agatgtacct atggtcctag taggaaataa atgtgatttg
       361 ccttctagaa cagtagacac aaaacaggct caggacttag caagaagtta tggaattcct
       421 tttattcaaa catcagcaaa gacaagacag
 //
```

The next line contains a short DEFINITION of the sequence that is contained in the entry. An ACCESSION number is a unique identifying sequence for this data entry. Note that only the accession number will necessarily be constant across nucleotide databases (NCBI, EMBL, DDBJ). The accession numbers are unique among these three nucleotide databases but are not necessarily unique between other databases (e.g. between protein and nucleotide databases). LOCUS names are variable and can be changed. The LOCUS names are often changed as nomenclature is changed or as sequences are merged into larger entries. The ACCESSION number and the LOCUS names are two character strings that can be easily used to access and retrieve sequence entries from NCBI.

Following these, come KEYWORDS that identify the particular entry. The SOURCE line describes how the sequence was cloned/sequenced and the ORGANISM line describes the species/construct from which the sequence originates. Following this, is a description of REFERENCES that deal with this entry. Note that this would include only the original papers describing the sequencing and not any other subsequent papers that might analyze the sequence. Multiple references will be given when different labs have sequenced the same DNA or when different publications describe different parts of the sequence. Throughout the NCBI database, numbers in square brackets indicate items in the REFERENCE list. The STANDARD describes any checks on the accuracy of the sequence.

The FEATURES section will describe things such as coding sequence start/stop, leader sequence start/stop, presence of signal sequences, locations of exons/introns, repeats, polymorphisms, and so on. There may also be comments in this section that can be useful. It may describe the some of the interesting facts that may go along with this sequence. This might include why it was sequenced, how it relates to other sequences in the database, some unusual features of the sequence, etc.

The BASE COUNT line gives the proportions of each nucleotide in the sequence. The ORIGIN line gives details of where the sequence starts relative to restriction sites (or other location markers) that aided the cloning.

Finally the sequence follows in lower case, in groups of 10 and with the number of the first nucleotide given on the left.

The above example entry is a particularly short sequence. This is not the norm for NCBI entries. Most entries contain a longer sequence as shown in the example below.

Example Entry #2: GenBank/NCBI - GORHBBPG

```
LOCUS       GORHBBPG     7055 bp    DNA             PRI       13-JUL-1993
DEFINITION  Gorilla beta-globin and eta-globin pseudogenes and an Alu repeat.
ACCESSION   K02543 M18037
NID         g177056
KEYWORDS    Alu repeat; globin; hemoglobin; pseudogene.
SOURCE      Lowland gorilla (Gorilla gorilla gorilla; SF-4) blood DNA, clone
            Ggo lambda-1059-1.1 [1].
  ORGANISM  Gorilla gorilla
            Eukaryota; Animalia; Chordata; Vertebrata; Mammalia; Theria;
            Eutheria; Primates; Haplorhini; Catarrhini; Pongidae.
```

```
REFERENCE   1  (bases 1613 to 3763)
  AUTHORS   Chang,L.Y. and Slightom,J.L.
  TITLE     Isolation and nucleotide sequence analysis of the beta-type globin
            pseudogene from human, gorilla and chimpanzee
  JOURNAL   J. Mol. Biol. 180, 767-784 (1984)
  MEDLINE   85134894
REFERENCE   2  (bases 1613 to 3763)
  AUTHORS   Koop,B.F., Goodman,M., Xu,P., Chan,K. and Slightom,J.L.
  TITLE     Primate eta-globin DNA sequences and man's place among the great
            apes
  JOURNAL   Nature 319, 234-238 (1986)
  MEDLINE   86118664
REFERENCE   3  (bases 1 to 7055)
  AUTHORS   Miyamoto,M.M., Slightom,J.L. and Goodman,M.
  TITLE     Phylogenetic relations of humans and African apes from DNA
            sequences in the pseudo-eta-globin region
  JOURNAL   Science 238, 369-373 (1987)
  MEDLINE   88018021
COMMENT     [3]  revises [1],[2].
            Computer-readable sequence for [3] kindly provided by M.M.Miyamoto,
            6-FEB-1988.

            NCBI gi: 177056
FEATURES             Location/Qualifiers
     source          1..7055
                     /organism="Gorilla gorilla"
                     /isolate="SF-4"
                     /sub_species="gorilla"
                     /sequenced_mol="DNA"
                     /tissue_type="blood"
     repeat_region   1067..1391
                     /note="Alu repeat"
     repeat_region   1814..1852
                     /note="direct degenerate repeat copy A"
     repeat_region   1853..1889
                     /note="direct degenerate repeat copy B"
     mRNA            1935..3667
                     /note="pseudo-beta-globim mRNA"
     CDS             join(1988..2078,2200..2422,3274..3400)
                     /gene="pseudo-beta-globin"
                     /pseudo
                     /codon_start=1
     exon            2200..2422
                     /gene="pseudo-beta-globin"
                     /pseudo
                     /number=2
BASE COUNT     2215 a   1315 c   1439 g   2086 t
ORIGIN      1 bp upstream of EcoRI site.
        1 gaattcctgg ttggctgatg gaagatgggg caactgttca ctggtatgca gggttttaga
       61 tgtatgtacc taaggatatg aggtatggca atgaacagaa attcttttgg gaatgagttt
      121 tagggccatt aaaggacatg acctgaagtt tcctctcagg ccagtcccca caactcaata
      181 taaatgtgtt tcctgcatac agtcaaagtt gccacttctt tttcttcata tcatcgatct
      241 ctgctcttaa agataatctt ggttttgcct caaactgttt gtcactacaa actttcccca
      301 tgttcctaag taaaacagat aactgcctct caactatatc aagtagacta aaatattgtg
      361 tctctaatat cagaaattca gctttaatat attgggttta actctttgaa atttagagta
      421 tccttgaaat acacatgggg gtgatttcct aaactttatt tcttgtaagg atttatctca
      481 ggggtaacac acaaaccagc atcctgaacc tctaagtatg aggacagtaa gccttaagaa
      541 tataaaataa actgttattc tctctgccgg tgcaagtgcg ccctgtctat tcctgaaatt
      601 gctcgtttga gacgcatgag acgtgcagca catgagacac gtgcagcagc ctgtggaata
      661 ttgtcagtga agaatgtctc tgcctgatta gatataaaga caagttaaac acagcattag
      721 actatagctc aagcctgtgc cagacacaaa tgacctaatg cccagcatgg gccatggaat
      781 ctcctatcct cttgcttgaa cagagcagca cacttctccc ccaacactat tagatgttct
      841 ggcataattt tgtagatatg taggatttga catggactat tgttcaatga ttcagaggaa
      901 atctcctttg ttcagataag tacactgact actaaatgga ttaaaaaaca cagtaataaa
      961 acccagtttt ccccttattt ccctagtttg tttcttattc tgctttcttc caaattgatg
     1021 ctggatagag gtgtttattt ctattctaaa aagtgatgaa attggccggg cgcggtggct
     1081 cacacctgta atcccagcac tttgggaggc tgaggtgggc ggatcacgag gtcaggagat
     1141 caagaccatc ctggctaaca tggtgaaacc ccatctctac taaaaataca aaaaaattag
     1201 ccagagacgg tggcgggtgc ctgtagtccc agctactcgg gaggctgagg caggagaatg
     1261 gtgtgaacct gggaggcaga gcttgcagtg agcagagatc gtgccactgc acactccagc
     1321 ctgggtgaca aagcaagact ccatctcaaa aaaaaaaaaa aaaagaaaga aagaaagaaa
     1381 gaaaaataaa ggtgatgaaa ttgtgtattc aatgtagtct caagagaatt gaaaaccaag
     1441 aaaggctgtg gcttcttcca cataaaacct ggatgaataa caggataaca cgtcgttaca
     1501 ttgtcacaac tcctgatcca ggaattgatg gctaagatat tcgtaattct tatccttttc
     1561 agttgtaact tattcctatt tgtcagcatt caggttatta gcggccgctg gcgaagtcct
     1621 tgagaaataa actgcacact ggacggtggg gatagcgtag gaaaatggag gggaaggaag
     1681 taaagttcca aattaagcct gaacagcaaa gttcccctga gaaggccacc tggattctat
     1741 cagaaactcg aatgtccatc ttgcaaaact tccttgccca aaccccaccc ctggagtcac
     1801 aacccaccct tgaccaatag attcatttca ctaagagaag caaagggctg gtcaatggat
     1861 tcatttcact gggagaggca aagggctggg ggccagagag gagaagtaaa aagccacaca
     1921 tgaagcagca atgcaggcat gtttctggct catctgtgat caccaggaaa ctcccagatc
     1981 tgacactgta gtgcatttca ctgctgacaa gaaggctgct gccaccagcc tgtgaagcaa
     2041 ggttaaggtg agaaggctgg aggtgagatt ctgggcaggt aggtactgga agccggggca
     2101 aggtgcagaa aggcagaaag tgtttctgaa agagggatta gcccattgtc ttacatagtc
     2161 tgactttgca cctgctctgt gattatgact atcccacagt ctcctggttg tctacccatg
     2221 gacctagagg tactttgaaa gttttggata tctgggctct gactgtgcaa taatgggcaa
     2281 ccccaaagtc aaggcacatg gcaagaaggt gctgatctcc ttcggaaaag ctgttatgct
     2341 cacggatgac ctcaaaggca cctttgctac gctgagtgac ctgcactgta acaagctgca
     2401 cgtggaccct gagaacttcc tggtgagtac taagtacact cacactttct tctttaccct
     2461 tagatatttg cactatgggc acttttgaaa gcagaggtgg ctttctcttg tgttatgagt
     2521 cagctgtggg atataatatt tcagcagtgg gattttgaga gttatgttgc tgtaaataac
     2581 ataactaaaa tttggtagag caaggactac gaataatgga aggccactta ccatttgata
     2641 gctctgaaaa acacatctta taaaaaattc tggccaaaat caaactgagt gtttttggat
     2701 gagggaacag aagttgagat agagaaaata acatctttcc tttggtcagc gaaattttct
     2761 ataaaaatta atagtcactt ttcttcatag tcctggaggt tagaaaaaga tcaactgaac
     2821 aaagtagtgg gaagctgtta aaaagaggat tgtttccctc ctaatgatga tggtatactt
     2881 ttgtacgcat ggtacaggat tctttgttat gagtgtttgg gaaaattgta tgtatgtatg
     2941 tatgtgatga ctggggactt atcctatcca ttactgttcc ttgaagtact attatcctac
     3001 tttttaaaag gacgaagtct ctaaaaaaaa tgaaacaatt aatcacaata tgctggggta
     3061 gtgagttggc atagcaagta agagaaggat aggacacaat gggaggtgca gggctgccag
     3121 tcatattgaa gctgatatct agcccataat ggtgagagtt gctcaaactc tggtcaaaaa
     3181 ggatgtaagt gttatatcta tttactgcaa gtccagcttg aggccttcta ttcactatgt
     3241 accattttct ttttatcttc actccctccc cagctcttag gcaacgtgat attgattgtt
     3301 ttggcaaccc acttcagcga ggagtttacc ctacagatac aggcttcctg gcagtaacta
     3361 acaaatgctg tggttaatgc tgtagcccac aagaccactg agtcccctgt ccactatgtt
     3421 tgtacctact ggtccactat gtttgtacct atgtccccaa atctcatctc ctttagatgg
     3481 gggaggttgg ggagaagagc agtatcctgc ctgctgattc agttcctgca tgataaaaat
     3541 aaaataaaga aatatgctct ctaagaaata tcattgtatt ctttttctgt ctttatattt
     3601 taccctgatt cagccaaaag gacgcaccat ttctgatgga aatgagaatg ttggagaatg
     3661 ggagcttaag gacagagaag atactttctt gcaatcctgc aagaaaagag agaacttgtg
     3721 ggttgattta gtggggtagt tactcctagg aaggggaaat cgtctctaga ataagacaat
     3781 gtctttacag aaagggaggt caatggaggt actctttgga gatgtaagag gattgttggt
     3841 agtgtgtaga ggtatgttag gactcaaatt agaagttctg tataggctat tatttgtatg
     3901 aactcaggat acagctcatt tggtgactgc agttcacttc tacttatttt gaacaacata
     3961 tttttttatga cttataatga agtggggatg gggcttccta gagaccaatc aagggccaaa
     4021 ccttgaactt tctcttaacg tcttcaatgg tattaataga gaattatctc taaggcatgt
     4081 gaactggctg tcttggtttt catctgtact tcatctgcta cctctgtgac ctgaaacata
     4141 tttataattc cattaagctg tacatatgat agatttatca tatttatttt ccttaaagga
     4201 tttttgtaag aacgaattga attgatacct gtaaagtctt tatcacacta cccgataaat
     4261 aataaatctc tttgttcagc tctctgtttc tataaatatg tacaagtttt attgttttta
     4321 gtggtagtga ttttattctc tttctatata tatatataca catatgtgtg cattcataaa
     4381 tatatacaat ttttatgaat aaaaaattat tagcaatcaa tattgaaaac cactgatttt
     4441 tgtttatgtg agtaaacagc agattaaaag gctgagattt aggaaacagc acgttaagtc
     4501 aagttgatag aggagaatat ggacatttaa aagaggcagg atgatataaa attagggaaa
     4561 ctggatgcag agaccagatg aagtaagaaa aatagctatt gttttgagca aaaatcactg
     4621 aagtttctcg catatgagag tgacataata aatagggaaa tgtagaaaat tgattcacat
     4681 gtatatatat atatagaact gattagacaa agtctaactt gggtatagtc agaggagctt
     4741 gctgtaatta tattgaggtg atggataaag aactgaagtt gatggaaaca atgaagttaa
     4801 gaaaaaaaat cgagtaagag accactgtgg cagtgattgc acagaactgg aaaacactgt
     4861 gaaacagaga gtcagagatg acagctaaaa tccctgcctg tgaatgaaaa gaaggaaatt
     4921 tattgacaga acagcaaatg cctacaagcc ccctgtttgg atctggcaat gaacatagtc
     4981 attctgtggc aatcacttca aactcctgta cccaagaccc ttaggaagta tgtagcaccc
     5041 tcaaacctaa aacctcaaag aaagaggttt tagaagatat aatatccttt cttctccagt
     5101 ttcattaatc cccaagcctc tttctcaaag tatttcctct atgtgtccac cccaaagagc
     5161 tcacctcacc atatctcttg agtgggagca catagatagg cggtgctacc atctgacagc
     5221 ttctgaaatt cctttgtcat atttttgagt ccccactaat aacccacaaa gcagaataaa
     5281 taacagttgc tcatgtacaa taatcactca actgctgtct tgtagcatac attaattaag
     5341 cacattcttt gaataattac tgtgtccaaa caatcacact ttaaaatctc acacttatgc
     5401 tatcccttgc ccttctgaat gtcactctgt attttaaatt aagagaggag ggttgaattt
     5461 cctgtgttac ttattgttca tttctcgatg aggagttttc acattcacct ttactggaaa
     5521 acacataagc acacatctta caggaaaaat ataccaaact gacatgtagc atgaatgctt
     5581 gtgcatgtag tcatataaaa tcttgtagca atgtaaacat tctctgatat acacatacag
     5641 atgtgtctat atgtctacac aatttcttat gctccatgaa caaacattcc atgcacacat
     5701 aagaacacac actgttacag atgcatactt gagtgcattg acaaaattac cccagtcaat
     5761 ctagagaatt tggatttctg catttgactc tgttagcttt gtgcatgctg ttcatttact
     5821 ctgggtgatg tctttccctc attttgccct gtctatcttg tactcatact ttaagtccta
     5881 acttatatgc tatctcaatt aagaagctat tttttttttaa attttaactg ggcttaaagc
     5941 cctgtctata aactctgcta caattatggg ctctttctta taatatttag tgtttttcct
     6001 actaatgtac ttaatctgct cattgtatct tcctaccact aaattttaac ctcatttatg
     6061 gtagagacat tgtcttgtaa actcttattt ccctagtatt tggagatgaa aaaaaaagat
     6121 taaattatcc aaaattagat ctctcttttc tacattatga gtattacact atccatagag
     6181 aagtttgttt gagacctaaa ctgaggaacc tttggttcta aaatgactat gtgatatctt
     6241 agtatttgta ggtcatgagg ttccttcctc tgcctctata aggctgttcc ctcaatctcc
     6301 cttgctatag tttgattagt caacaagcat gtgtcatgca tttattcaca tcagaatttt
     6361 catacactaa taagacatag tatcagaagt cagtttatta gttatatcag ttagggtcca
     6421 tcaaggaaag gacaaaccat tatcagttac tcaacctaga attaaataca gctcttaata
     6481 gttaattatc cttgtattgg aagagctaaa atatcaaata aaggacagtg cagaaatcta
     6541 gatgttagta acatcagaaa acctcttccg ccattaggcc tagaagggca gaaggagaaa
     6601 atgtttatac caccagagtc cagaaccaga gaccataacc agaggtccac tggattcagt
     6661 gagctagtgg gggctccttg gagagagcca gaactaggtg tctaatgggg gtatcaaaat
     6721 atcagccata aaaaaccata aaaaagactg tctgctgtag gagatctgtt cagagagaga
     6781 gagagacaag aaataatctt gcttatgctt tccctcagcc agtgtttacc actgcagaat
     6841 gtacatgcaa ctgaaagggt gaggaaacct gggaaatgtc agttcctcaa atacagagaa
     6901 cactgaggga aggatgagaa ataaatttga aagcagacat gaatggtaat tgacagaagg
     6961 aaactaggat gtgtccagta aatgaataat tacagtgtgc agtgattatt gcaatgatta
     7021 atgtattgat aagataatat gaaaacacag aattc
//
```

Nor is this a particularly large entry. Many of the entries now originate from complete genomes (or exceptionally long contigs that when joined together represent a complete genome) that could be many millions of nucleotides long. As of August 2005, there are more than twenty eight eukaryotic organisms completely sequenced (the yeast *Saccharomyces cerevisiae* and nine other fungi, six single celled protists, the nematodes *Caenorhabditis elegans* (and *briggsae*), the plants *Arabidopsis thaliana* and *Oryza sativa*, the fruit fly *Drosophila melanogaster* (and *simulans*), the mosquito *Anopheles gambiae*, the fish *Danio rerio* and *Tetraodon nigroviridis*, the chicken *Gallus gallus*, mouse *Mus musculus*, the rat *Ratuus norvegicus*, man's best friend *Canis familiaris*, the chimpanzee *Pan troglodytes* and *Homo sapiens*). I suggest "more than" simply because the human additions to database entries can't keep up with the automated sequencing machines. Many more will appear in the next few years. The prokaryotic genomes are moving more rapidly and as of August 2005, there have been at least 22 archaeal and 221 bacterial genomes completely sequenced (the first being *Haemophilus influenzae* in 1995 coming in at 1,830,140 bp.) and there are many more nearing completion.

In the past, full genome sequences were reported as major milestones of achievement in journals such as SCIENCE and NATURE. In the last 1.5 months of 1997, the journal NATURE published five issues. Of these, three issues reported the completion of three different bacterial genomes (*Bacillus subtilis, Archaeoglobus fulgidus*, and *Borrelia burgdorferi*). Today, many genomes are still reported in high profile journals but others are published in more specialized journals and some are simply published online. The rate of genome completion is rapidly speeding up and as many more genomes rapidly near completion, the news worthiness of a single genome tends to diminish. But their utility increases with each one determined.

In addition to the complete genomes, there is a long list of viruses and organelles (e.g. see also the OGMP - organelle genome megasequencing program) that have been completely sequenced including the CMV DNA virus (300,000 bp), the Epstein-Barr virus genome (172,282 bp), the AIDS virus (9,737 bp), human mitochondria (16,569 bp), human leukaemia virus type I (9,032 bp), lambda (48,502 bp), PhiX174 (5,386 bp) and many more.

For more information about NCBI go to their web site.

4.3 E.M.B.L.

The same entry as in Example #1 above is also present at EMBL (as they all should be). It's format is shown in Example #3.

Example Entry #3: EMBL - HSCKRA01

```
ID   HSCKRA01   standard; RNA; HUM; 450 BP.
XX
AC   M35504;
XX
SV   M35504.1
XX
DT   26-NOV-1990 (Rel. 25, Created)
DT   04-MAR-2000 (Rel. 63, Last updated, Version 3)
XX
DE   Human PR310 c-K-ras protein mRNA, 5' end.
XX
KW   c-K-ras oncogene; c-myc proto-oncogene.
XX
OS   Homo sapiens (human)
```

```
OC   Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia;
OC   Eutheria; Primates; Catarrhini; Hominidae; Homo.
XX
RN   [1]
RP   1-450
RX   MEDLINE; 85271309.
RA   Yamamoto F., Nakano H., Neville C., Perucho M.;
RT   "Structure and mechanisms of activation of c-K-ras oncogenes in human lung
RT   cancer";
RL   Prog. Med. Virol. 32:101-114(1985).
XX
DR   GOA; Q14014; Q14014.
DR   SPTREMBL; Q14014; Q14014.
XX
FH   Key             Location/Qualifiers
FH
FT   source          1..450
FT                   /db_xref="taxon:9606"
FT                   /organism="Homo sapiens"
FT   CDS             1..>450
FT                   /codon_start=1
FT                   /db_xref="GOA:Q14014"
FT                   /db_xref="SPTREMBL:Q14014"
FT                   /note="PR310 c-K-ras oncogene"
FT                   /protein_id="AAA35689.1"
FT                   /translation="MTEYKLVVVGAGGVGKSALTIQLIDNHFVDEYDPTIEDSYRKQVV
FT                   IDGETCLLDILDTAGHEEYSAMRDQYMRTGEGFLCVFAINNTKSFEDIHHYREQIKRVK
FT                   DSEDVPMVLVGNKCDLPSRTVDTKQAQDLARSYGIPFIQTSAKTRQ"
XX
SQ   Sequence 450 BP; 155 A; 71 C; 106 G; 118 T; 0 other;
     atgactgaat ataaacttgt ggtagttgga gctggtggcg taggcaagag tgccttgacg        60
     atacagctaa ttgacaatca ttttgtggac gaatatgatc caacaataga ggattcctac       120
     aggaagcaag tagtaattga tggagaaacc tgtctcttgg atattctcga cacagcaggt       180
     catgaggagt acagtgcaat gagggaccag tacatgagga ctggggaggg ctttctttgt       240
     gtatttgcca taaataatac taaatcattt gaagatattc accattatag agaacaaatt       300
     aaaagagtta aggactctga agatgtacct atggtcctag taggaaataa atgtgatttg       360
     ccttctagaa cagtagacac aaaacaggct caggacttag caagaagtta tggaattcct       420
     tttattcaaa catcagcaaa gacaagacag                                        450
//
```

Note that the entry contains the same information but in a slightly different form. In this case, the data is more structured with defined prefixes at the beginning of every line. This difference can be useful if you wish to write your own code to analyze some features of this data.

The EMBL databases have moved from Heidelberg, Germany to Hinxton Hall (Cambridge, England). But many of the people doing protein analysis still exist at Heidelberg. For more information about the EMBL database check out their web site or send e-mail to `netserv@ebi.ac.uk`.

4.4 D.D.B.J.

For the particular entry chosen in Example #1, the DDBJ format is essentially equivalent (there are minor differences). It is ...

Example Entry #4: DDBJ - HUMCKRASA

```
LOCUS       HUMCKRASA     450 bp ss-mRNA            PRI       15-SEP-1990
DEFINITION  Human PR310 c-K-ras protein mRNA, 5' end.
ACCESSION   M35504
KEYWORDS    c-K-ras protein; c-myc oncogene.
SOURCE      Human (patient PR310) lung carcinoma, cDNA to mRNA.
  ORGANISM  Homo sapiens
            Eukaryota; Animalia; Metazoa; Chordata; Vertebrata; Mammalia;
            Theria; Eutheria; Primates; Haplorhini; Catarrhini; Hominidae.
REFERENCE   1  (bases 1 to 450)
```

```
   AUTHORS   Yamamoto,F., Nakano,H., Neville,C. and Perucho,M.
   TITLE     Structure and mechanisms of activation of c-K-ras oncogenes in
             human lung cancer
   JOURNAL   Prog. Med. Virol. 32, 101-114 (1985)
   STANDARD  simple staff_entry
 FEATURES             Location/Qualifiers
      CDS             1..>450
                      /note="PR310 c-K-ras oncogene"
                      /codon_start=1
 BASE COUNT      155 a     71 c    106 g    118 t
 ORIGIN
         1 atgactgaat ataaacttgt ggtagttgga gctggtggcg taggcaagag tgccttgacg
        61 atacagctaa ttgacaatca ttttgtggac gaatatgatc caacaataga ggattcctac
       121 aggaagcaag tagtaattga tggagaaacc tgtctcttgg atattctcga cacagcaggt
       181 catgaggagt acagtgcaat gagggaccag tacatgagga ctggggaggg ctttctttgt
       241 gtatttgcca taaataatac taaatcattt gaagatattc accattatag agaacaaatt
       301 aaaagagtta aggactctga agatgtacct atggtcctag taggaaataa atgtgatttg
       361 ccttctagaa cagtagacac aaaacaggct caggacttag caagaagtta tggaattcct
       421 tttattcaaa catcagcaaa gacaagacag
 //
```

The DDBJ began in 1986 and is operated from grants from the Japanese Ministry of Education, Science and Culture. For more information go to their web site.

4.5 SwissProt

These are the three major nucleotide databases, but there are also a large number of protein sequence databases. Again many of these databases are very large. For example, release 47.6 of SWISS- PROT (Aug 2 2005) contains 188,752 annotated entries containing a total of 68,301,856 amino acid residues. There are more than 473 entries having proteins larger than 2500 residues including the absolutely massive human nebulin protein of 6669 amino acids and the human nesprin 1 protein of 8797 amino acids. The entries in this database are similar to the nucleotide databases of EMBL. Two examples are shown below.

Example Entry #5: SWISS-PROT - ACYO_HUMAN

```
ID   ACYO_HUMAN     STANDARD;      PRT;    98 AA.
AC   P07311;
DT   01-APR-1988 (REL. 07, CREATED)
DT   01-APR-1988 (REL. 07, LAST SEQUENCE UPDATE)
DT   01-NOV-1995 (REL. 32, LAST ANNOTATION UPDATE)
DE   ACYLPHOSPHATASE, ORGAN-COMMON TYPE ISOZYME (EC 3.6.1.7)
DE   (ACYLPHOSPHATE PHOSPHOHYDROLASE) (ACYLPHOSPHATASE, ERYTHROCYTE
DE   ISOZYME).
OS   HOMO SAPIENS (HUMAN).
OC   EUKARYOTA; METAZOA; CHORDATA; VERTEBRATA; TETRAPODA; MAMMALIA;
OC   EUTHERIA; PRIMATES.
RN   [1]
RP   SEQUENCE.
RX   MEDLINE; 87101109.
RA   LIGURI G., CAMICI G., MANAO G., CAPPUGI G., NASSI P., MODESTI A.,
RA   RAMPONI G.;
RL   BIOCHEMISTRY 25:8089-8094(1986).
CC   -!- FUNCTION: ITS PHYSIOLOGICAL ROLE IS NOT YET CLEAR.
CC   -!- CATALYTIC ACTIVITY: AN ACYLPHOSPHATE + H(2)O = A FATTY ACID ANION
CC       + ORTHOPHOSPHATE.
CC   -!- TISSUE SPECIFICITY: ORGAN-COMMON TYPE ISOZYME IS FOUND IN MANY
CC       DIFFERENT TISSUES.
CC   -!- SIMILARITY: HIGH, WITH ORGAN-COMMON TYPE ACYLPHOSPHATASES. LESS
CC       WITH MUSCLE TYPE ACYLPHOSPHATASES.
DR   PIR; A25587; QPHUE.
DR   HSSP; P00818; 1APS.
DR   PROSITE; PS00150; ACYLPHOSPHATASE_1.
DR   PROSITE; PS00151; ACYLPHOSPHATASE_2.
KW   HYDROLASE; ACETYLATION; MULTIGENE FAMILY.
FT   MOD_RES       1      1       ACETYLATION.
SQ   SEQUENCE   98 AA;  11130 MW;  51080 CN;
     AEGNTLISVD YEIFGKVQGV FFRKHTQAEG KKLGLVGWVQ NTDRGTVQGQ LQGPISKVRH        60
     MQEWLETRGS PKSHIDKANF NNEKVILKLD YSDFQIVK                                98
//
```

Example Entry #6: SWISS-PROT - CD25_YEAST

```
ID   CC25_YEAST     STANDARD;      PRT;  1589 AA.
AC   P04821;
DT   13-AUG-1987 (REL. 05, CREATED)
DT   01-JAN-1988 (REL. 06, LAST SEQUENCE UPDATE)
DT   01-NOV-1995 (REL. 32, LAST ANNOTATION UPDATE)
DE   CELL DIVISION CONTROL PROTEIN 25.
GN   CDC25 OR CTN1.
OS   SACCHAROMYCES CEREVISIAE (BAKER'S YEAST).
OC   EUKARYOTA; FUNGI; ASCOMYCOTINA; HEMIASCOMYCETES.
RN   [1]
RP   SEQUENCE FROM N.A.
RX   MEDLINE; 87131091.
RA   BROEK D., TODA T., MICHAELI T., LEVIN L., BIRCHMEIER C., ZOLLER M.,
RA   POWERS S., WIGLER M.;
RL   CELL 48:789-799(1987).
RN   [2]
RP   SEQUENCE FROM N.A.
RX   MEDLINE; 86220116.
RA   CAMONIS J.H., KALEKINE M., GONDRE B., GARREAU H., BOY-MARCOTTE E.,
RA   JACQUET M.;
RL   EMBO J. 5:375-380(1986).
RN   [3]
RP   DOMAINS.
RX   MEDLINE; 89181526.
RA   MUNDER T., MINK M., KUNTZEL H.;
RL   MOL. GEN. GENET. 214:271-277(1988).
RN   [4]
RP   FUNCTION.
RX   MEDLINE; 91203884.
RA   JONES S., VIGNAIS M.L., BROACH J.R.;
RL   MOL. CELL. BIOL. 11:2641-2646(1991).
CC   -!- FUNCTION: PROMOTES THE EXCHANGE OF RAS-BOUND GDP BY GTP. THIS
CC       PROTEIN POSITIVELY CONTROLS THE LEVEL OF CELLULAR CAMP AT START,
CC       THE STAGE AT WHICH THE YEAST CELL DIVISION CYCLE IS TRIGGERED.
CC   -!- SIMILARITY: CONTAINS A COPY OF THE SH3 DOMAIN.
CC   -!- SIMILARITY: TO OTHER GUANINE-NUCLEOTIDE RELEASING FACTORS OF THE
CC       CDC25 FAMILY.
DR   EMBL; X03579; X03579.
DR   EMBL; M15458; M15458.
DR   PIR; A26596; RGBYC5.
DR   HSSP; P00519; 1ABL.
DR   LISTA; SC00152; CDC25.
DR   SGD; L0000263; CDC25.
DR   PROSITE; PS00720; GDS_CDC25.
DR   PROSITE; PS50002; SH3.
KW   GUANINE-NUCLEOTIDE RELEASING FACTOR; CELL DIVISION; CELL CYCLE;
KW   MITOSIS; TRANSMEMBRANE; SH3 DOMAIN.
FT   TRANSMEM   1452   1473       POTENTIAL.
FT   DOMAIN       58    128       SH3.
FT   CONFLICT    497    497       I -> Y (IN REF. 2).
FT   CONFLICT    954    963       PVGHHEPFKN -> LSVIMNLSR (IN REF. 2).
SQ   SEQUENCE   1589 AA;  179091 MW;  13488958 CN;
     MSDTNTSIPN TSSAREAGNA SQTPSISSSS NTSTTTNTES SSASLSSSPS TSELTSIRPI        60
     GIVVAAYDFN YPIKKDSSSQ LLSVQQGETI YILNKNSSGW WDGLVIDDSN GKVNRGWFPQ       120
     NFGRPLRDSH LRKHSHPMKK YSSSKSSRRS SLNSLGNSAY LHVPRNPSKS RRGSSTLSAS       180
     LSNAHNAETS SGHNNTVSMN NSPFSAPNDA SHITPQSSNF NSNASLSQDM TKSADGSSEM       240
     NTNAIMNNNE TNLQTSGEKA GPPLVAEETI KILPLEEIEM IINGIRSNIA STWSPIPLIT       300
     KTSDYKLVYY NKDLDIYCSE LPLISNSIME SDDICDSEPK FPPNDHLVNL YTRDLRKNAN       360
     IEDSSTRSKQ SESEQNRSSL LMEKQDSKET DGNNNSINDD DNNNENNKNE FNEAGPSSLN       420
     SLSAPDLTQN IQSRVVAPSR SSILAKSDIF YHYSRDIKLW TELQDLTVYY TKTAHKMFLK       480
     ENRLNFTKYF DLISDSIVFT QLGCRLMQHE IKAKSCSKEI KKIFKGLISS LSRISINSHL       540
     YFDSAFHRKK MDTMNDKDND NQENNCSRTE GDDGKIEVDS VHDLVSVPLS GKRNVSTSTT       600
     DTLTPMRSSF STVNENDMEN FSVLGPRNSV NSVVTPRTSI QNSTLEDFSP SNKNFKSAKS       660
     IYEMVDVEFS KFLRHVQLLY FVLQSSVFSD DNTLPQLLPR FFKGSFSGGS WTNPFSTFIT       720
     DEFGNATKNK AVTSNEVTAS SSKNSSISRI PPKMADAIAS ASGYSANSET NSQIDLKASS       780
     AASGSVFTPF NRPSHNRTFS RARVSKRKKK YPLTVDTLNT MKKKSSQIFE KLNNATGEHL       840
     KIISKPKSRI RNLEINSSTY EQINQNVLLL EILENLDLSI FINLKNLIKT PSILLDLESE       900
     EFLVHAMSSV SSVLTEFFDI KQAFHDIVIR LIMTTQQTTL DDPYLFSSMR SNFPVGHHEP       960
     FKNISNTPLV KGPFHKKNEQ LALSLFHVLV SQDVEFNNLE FLNNSDDFKD ACEKYVEISN      1020
     LACIIVDQLI EERENLLNYA ARMMKNNLTA ELLKGEQEKW FDIYSEDYSD DDSENDEAII      1080
     DDELGSEDYI ERKAANIEKN LPWFLTSDYE TSLVYDSRGK IRGGTKEALI EHLTSHELVD      1140
     AAFNVTMLIT FRSILTTREF FYALIYRYNL YPPEGLSYDD YNIWIEKKSN PIKCRVVNIM      1200
     RTFLTQYWTR NYYEPGIPLI LNFAKMVVSE KIPGAEDLLQ KINEKLINEN EKEPVDPKQQ      1260
     DSVSAVVQTT KRDNKSPIHM SSSSLPSSAS SAFFRLKKLK LLDIDPYTYA TQLTVLEHDL      1320
     YLRITMFECL DRAWGTKYCN MGGSPNITKF IANANTLTNF VSHTIVKQAD VKTRSKLTQY      1380
     FVTVAQHCKE LNNFSSMTAI VSALYSSPIY RLKKTWDLVS TESKDLLKNL NNLMDSKRNF      1440
     VKYRELLRSV TDVACVPFFG VYLSDLTFTF VGNPDFLHNS TNIINFSKRT KIANIVEEII      1500
     SFKRFHYKLK RLDDIQTVIE ASLENVPHIE KQYQLSLQVE PRSGNTKGST HASSASGTKT      1560
     AKFLSEFTDD KNGNFLKLGK KKPPSRLFR                                        1589
//
```

Again, various features are on individual lines – the identification line (ID) giving a locus name, the accession number line (AC), the date of entry (DT), a description of the entry (DE), a line specifying the organism (OS), the organism's phylogenetic classification (OC), lines describing the reference number, author and location (RN, RA, RL), the comment lines (CC), a database reference line (DR) to cross reference the entry to other database entries, the keyword line (KW), the feature tables (FT) and the sequence header (SQ) giving length in aa, molecular weight and a checking number defined in A. Bairoch, J. Biochem. 203: 527 (1983).

In addition to these protein databases, there are databases devoted to particular families of proteins and to particular organisms. In addition there are protein databases constructed from translations of the nucleotide databases – NCBI's is called GenPept and EMBL's is termed TrEMBL (release 30.6 of TrEMBL (Aug 2 2005) translates the EMBL nucleotide database to yield 1,942,311 entries of 624,047,190 amino acids). The SwissProt database is still released on CD-ROM if you wish to have everything on hand at once. But again, the best access for the SwissProt database is through their web site (or better yet the ExPASy (Expert Protein Analysis System) web site).

Another protein sequence database of interest is PIR (Protein Information Resource) sponsored by NBRF (National Biomedical Research Foundation) at Georgetown University. This database of protein sequences is completely cross-referenced to known nucleic acid sequences, has data on x-ray crystallography and active site determination, and is fully annotated. The last release of this database was on Dec 2004 as it is now integrated into UniProt.

In an effort to combine the information in these disparate protein databases, the UniProt database was constructed. It joins the information contained in Swiss-Prot, TrEMBL, and PIR. UniProt (Universal Protein Resource) claims to be the world's most comprehensive catalog of information on proteins. It is divided into three parts: UniProt Knowledgebase (UniProtKB) is the central access point for extensive curated protein information, including function, classification, and cross-reference. The UniProt Reference Clusters (UniRef) databases combine closely related sequences into a single record. The UniProt Archive (UniParc) is meant to be a comprehensive repository for all protein sequences. The August 2 2005 UniProt release 5.6 contains 2,131,063 entries (188,752 entries from Swiss-Prot and 1,942,311 entries from TrEMBL).

4.6 Organization of the entries

The entries can also be grouped according to their organismal affiliation. To give you an example of how this data is distributed by organism consider the data in Table 4.1 (this table excludes whole genome shotgun data). Several groups require further explanation. The large Unannotated group contains sequence entries that have not yet had things like the FEATURE table or other niceties added to the entry. The Synthetic group contains things such as plasmids, Yacs, etc that have been constructed by researchers. The ESTs (**e**xpressed **s**equence **t**ags) are short sequences derived from cDNAs. That is messenger RNA is reverse transcribed to cDNA, and the cDNA is partially sequenced from the the poly-A addition site. The STSs (**s**equence **t**agged **s**ites) and the GSSs (**g**enome **s**urvey **s**equences) are short sequences that can serve to map various regions. The HTC stands for **h**igh **t**hroughput **c**DNA sequences. The HTG stands for **h**igh **t**hroughput **g**enomic sequences. These are contig sequences greater than 2kb that have not yet been fully released from large scale sequencing laboratories. The HTG0 stands for **h**igh **t**hroughput **g**enomic sequences but with one-to-feww pass reads of a single clone. The sequences will normally be checked, assembled and annotated by these groups before official release. Beware these sequences may therefore contain more than their share of sequencing errors.

These data can be compared to previous years - the fourth column gives the corresponding EMBL data from 1995, and the fifth column above gives GenBank/Intelligenetics data from release 63 in June of 1990. The fastest growing groups (excluding the entries at the bottom) were plants, invertebrates and human/primates. The slowest growing groups were fungi and other mammalian sequences.

Table 4.1: Sequence by Organism (millions of nucleotides)

Species	EMBL rel 83 Aug 05	EMBL rel 63 Jun 00	EMBL rel 44 Aug 95	IG rel 63 Jun 90
Human	4158	825	-	-
Primate	-	-	41	2
Mus musculus	2819	-	-	-
Other Rodent	116	85	29	2
Other Mammalian	396	23	8	2
Other Vertebrate	1102	26	9	2
Invertebrate	717	326	36	2
Plant	1519	204	17	3
Organelle	323	56	11	2
Fungi	251	72	23	-
Prokaryotes	1033	189	43	2
Viral	268	82	28	2
Bacteriophage	16	4	2	1
Unannotated	2	2	5	0
Synthetic	31	9	5	1
ESTs	14544	1641	100	-
STSs	492	51	6	-
GSSs	7155	838	-	-
HTG	11490	3756	-	-
HTG0	510	-	-	-
HTC	422	-	-	-
Patents	1450	-	-	-
Environmental Samples	112	-	-	-
Unclassified	2	-	-	-

Table 4.2: The Most Sequenced Organisms (DDBJ:release 62 Jun 2005 vs. GB:Rel. 104 1997)

	Species	Nucleotides	Nucleotides	Species
1	*Homo sapiens*	11213851753	551650819	*Homo sapiens*
2	*Mus musculus*	6505474259	133404600	*Mus musculus*
3	*Rattus norvegicus*	5667052014	113659949	*Caenorhabditis elegans*
4	*Danio rerio*	2093785791	46566672	*Arabidopsis thaliana*
5	*Zea mays*	1624627840	29198918	*Drosophila melanogaster*
6	*Bos taurus*	1146316826	28567944	*Saccharomyces cerevisiae*
7	*Oryza sativa*	1080286305	17345841	*Escherichia coli*
8	*Xenopus tropicalis*	834465420	13826989	*Rattus norvegicus*
9	*Canis familiaris*	800858021	9715721	*Bacillus subtilis*
10	*Drosophila melanogaster*	778765821	9198129	Human immunodeficiency virus type 1
11	*Arabidopsis thaliana*	648678442	8932632	*Oryza sativa*
12	*Gallus gallus*	648678442	7522232	*Fugu rubripes*
13	*Pan troglodytes*	501117501	6398746	*Schizosaccharomyces pombe*
14	*Sorghum bicolor*	462496838	5099500	*Gallus gallus*
15	*Ciona intestinalis*	418888609	4509620	*Mycobacterium tuberculosis*
16	*Brassica oleracea*	404289611	4350356	*Toxoplasma gondii*
17	*Medicago truncatula*	384785347	4312388	*Brugia malayi*
18	*Sus scrofa*	384556170	4164811	*Bos taurus*
19	*Macaca mulatta*	378222624	3829215	*Synechocystis* sp.
20	*Triticum aestivum*	336241101	3160564	*Xenopus laevis*

The top twenty organisms (excluding chloroplast and mitochondrial sequences) as of June 2005, are shown in Table 4.2. Note that humans have already been done several times over. Also note that bacterial sequences can no longer make it among the top twenty. Even with multiple sequencing of their genomes, they do not contain the quantity of sequence that is present in eukaryotes.

Besides the data itself, most of the databases also maintain various index files. These include indices of authors, journals, organism, etc. all cross-referenced by accession number or locus name. Again these can be very helpful to analyze the entries of the databases.

4.7 Other Major Databases

There are many other databases of note and not all can be covered here. One of database of importance, though at first glance might be considered boring, is the GO database. This project is, in my mind, somewhat like the huge project that eventually became the Oxford Dictionary of the English Language. This is not simply a dictionary but rather the definitive reference for a language that constantly changes and evolves. It is not just a dictionary but includes the meaning, history, pronunciation, usage, quotations, variants, etymology and so on of any word both present and past. The GO database is (will become) the dictionary of gene nomenclature. The example from their website states "if you were searching for new targets for antibiotics, you might want to find all the gene products that are involved in bacterial protein synthesis, and that have significantly different sequences or structures from those in humans. But if one database describes these molecules as being involved in 'translation', whereas another uses the phrase 'protein synthesis', it will be difficult for you - and even harder for a computer - to find functionally equivalent terms". The Gene Ontology (GO) project is a collaborative effort to address the need for consistent descriptions of gene products in different databases. This is quickly becoming a standard that must be used in the annotations of new genomes.

The PDB (Protein Data Bank), sponsored by Rutgers University, contains the 3-D atomic coordinates from x-ray diffraction or NMR studies. The database also contains secondary and other structural features such as bond connectivity data. The individual database entries are usually directly suitable for entry into 3-D rendering programs.

The PROSITE database is very useful. It lists the distinct structural motifs in proteins. This includes amino acid post-translational modifications, topogenic sequences, domains of specific biological function (e.g. DNA binding domains), enzyme active sites and signature patterns that are specific to a family or group of proteins. For example, it lists the Kringle domain signature as (Y,F)-C-R-N-P-D; a triple-looped, disulfide cross-linked region found mostly in serine proteases and plasma proteins. For more information on this database, send e-mail to the EMBL databases.

Some databases are built on filtered and 'value-added' derivatives of these basic sequence databases. For example the COG database holds aligned clusters of orthologous proteins. There are proteins collected from 43 completed genomes and then compared "all-against-all" to yield 3307 clusters of related proteins (as of August 2002). One of the tools associated with this database is the `COGnitor` program, which will assign query proteins to pre-existing clusters (and hence usefully identify its functional category).

Besides these sorts of databases, there are also databases containing different types of information. The GDB (Genome Database) contains mapping information of the human genome project. It contains information on the location of genes, DNA segments, expressed sequence tags (EST's), clinical phenotypes, polymorphisms and alleles, probes, CEPH reference family data markers, etc. As part of this database, Victor McKusick has made available a computer readable form of his **Mendelian Inheritance in Man** (OMIM) This lists clinical disorders or traits in man, gene names, clinical observations, inheritance patterns, allelic variations, chromosome locations, linkages and so on. The GDB and OMIM and most of the other molecular biology databases are all cross-linked. These databases are maintained by the Welch medical library at Johns Hopkins.

Similar to the GDB, is the database for the mouse genome MGI: Mouse Genome Informatics. This again contains mapping information of much the same sort as the human database. However, it also includes homologies for mice, humans and 23 other species. Thus, if you are interested in a gene on chromosome 11 in mice, you can find out where it has been located in some other species (the references to the papers showing this, what other genes are similarly located and so on).

Pick an organelle and again you will find specialized databases. For just the mitochondria try the comprehensive MITOP web site, or MitoDat a database of Mendelian Inheritance and the Mitochondrion db (mitochondrial nuclear genes) or MITOMAP a database for the human mitochondrial genome. If a whole organelle is too large for your tastes, how about picking something smaller like a database for just part of the mitochondria, the hypervariable control region at HvrBase or how about the weird on wonderful inteins. Don't know what inteins are? Check out that link.

Recently there have been more projects to establish databases for the deposition of microarray gene expression data. The NCBI version of this data is housed in Gene Expression Omnibus (GEO). GEO is a gene expression and hybridization array data repository, as well as an online resource for the retrieval of gene expression data from any organism or artificial source. The EBI microarray ArrayExpress Database is a similar database to store and permit the query of microarray experiments.

There are many more databases. I have only given you a taste of some of the major databases. In addition to each of these major databases, there are databases on each of the organisms that have major genomics projects

- E. coli,
- Yeast,
- Arabidopsis,
- Mouse,
- Cattle,
- Drosophila,
- Caenorhabditis elegans,
- Fungi,
- Maize,
- Rice,
- Grasses - Gramene,
- Mycobacterium,
- HIV,

and so on. So pick your favorite organism and do a search for it – there will be a web site devoted to it's genome (so long as it is not too unusual).

There are many other databases on diverse aspects such as the

- CEPH-Genethon human physical map data and Genethon database provide a connection between the physical map of the human genome and the genetic/sequence data;
- a human cDNA database;
- a human expressed gene anatomy database EGAD;
- genome sequencing centers such as Baylor College of Medicine and the Sanger Center maintain their own databases on projects they are working on;
- the National Biomedical Research Foundation's NRL3D database derived from the 3D protein structures at the Brookhaven National Laboratories;
- an immunogenetics database IMGT;
- the Database of Expressed Sequence Tags dbEST;
- the Database of Sequence Tagged Sites dbSTS;
- the Eukaryotic Promoter Database EPD;
- a database of 3D-diagrams of proteins;
- BMRB (BioMagResBank) a database of NMR Spectroscopy data;
- CCDC (Cambridge Crystallographic Data Centre);
- HIC-Up (Hetero-compound Information Centre Uppsala); a database of small molecules commonly found associated with larger molecules when their 3D-structure is determined
- HIV Protease Database;
- Klotho: Biochemical Compounds Declarative Database;
- Library of Protein Family Cores;
- NDB (Nucleic Acid Database);
- Prolysis: A Protease and Protease Inhibitor Web Server;

- Protein Kinase Resource;
- Protein Motions Database;
- RELIBase;
- SCOP: Structural Classification of Proteins;
- CATH Protein Structure Classification;
- Enzyme Structures Database;
- PDBsum, a database of the known 3D structures of proteins and nucleic acids;
- the PredictProtein server which can generate multiple sequence alignments, predict secondary structure, predict residue solvent accessibility, predict transmembrane helices, predict topology of transmembrane proteins, etc.

This list goes on and on (and increases each month). Choose what you are interested in ... chances are others are interested as well and have built a database.

4.8 Remote Database Entry retrieval

4.8.1 Entrez

The premier method that should be mentioned is probably the one method that you will use more than any other. This is a NCBI project termed ENTREZ. This program can search across databases or natively through

- PubMed: biomedical literature citations and abstracts
- PubMed Central: free, full text journal articles
- Books: online books
- OMIM: online Mendelian Inheritance in Man
- Site Search: NCBI web and FTP sites
- Nucleotide: sequence database (GenBank)
- Protein: sequence database
- Genome: whole genome sequences
- Structure: three-dimensional macromolecular structures
- Taxonomy: organisms in GenBank
- SNP: single nucleotide polymorphism
- Gene: gene-centered information
- HomoloGene: eukaryotic homology groups
- PubChem Compound: small molecule chemical structures
- PubChem Substance: chemical substances screened for bioactivity
- Genome Project: genome project information
- UniGene: gene-oriented clusters of transcript sequences
- CDD: conserved protein domain database
- 3D Domains: domains from Entrez Structure
- UniSTS: markers and mapping data
- PopSet: population study data sets

Table 4.3: The ENTREZ search fields

		Available for Database ...				
Field	Short term	Nucleotide	Protein	Genome	Structure	PopSet
Accession	ACCN	Yes	Yes	Yes	Yes	Yes
All Fields	ALL	Yes	Yes	Yes	Yes	Yes
Author Name	AUTH	Yes	Yes	Yes	Yes	Yes
EC/RN Number	ECNO	Yes	Yes	Yes	Yes	Yes
Feature Key	FKEY	Yes	No	Yes	No	Yes
Filter	FILT	Yes	Yes	Yes	Yes	Yes
Gene Name	GENE	Yes	Yes	Yes	No	Yes
Issue	ISS	Yes	Yes	Yes	Yes	Yes
Journal Name	JOUR	Yes	Yes	Yes	Yes	Yes
Keyword	KYWD	Yes	Yes	Yes	No	Yes
Modification Date	MDAT	Yes	Yes	Yes	Yes	Yes
Molecular Weight	MOLWT	No	Yes	No	No	No
Organism	ORGN	Yes	Yes	Yes	Yes	Yes
Page Number	PAGE	Yes	Yes	Yes	Yes	Yes
Primary Accession	PACC	Yes	Yes	Yes	No	Yes
Properties	PROP	Yes	Yes	Yes	No	Yes
Protein Name	PROT	Yes	Yes	Yes	No	Yes
Publication Date	PDAT	Yes	Yes	Yes	Yes	Yes
SeqID String	SQID	Yes	Yes	Yes	No	Yes
Sequence Length	SLEN	Yes	Yes	Yes	No	No
Substance Name	SUBS	Yes	Yes	No	Yes	No
Text Word	WORD	Yes	Yes	Yes	Yes	Yes
Title Word	TITL	Yes	Yes	Yes	No	No
Volume	VOL	Yes	Yes	Yes	Yes	Yes

- GEO Profiles: expression and molecular abundance profiles
- GEO DataSets: experimental sets of GEO data
- Cancer Chromosomes: cytogenetic databases
- PubChem BioAssay: bioactivity screens of chemical substances
- GENSAT: gene expression atlas of mouse central nervous system
- Journals: detailed information about the journals indexed in PubMed and other Entrez databases
- NLM Catalog: catalog of books, journals, and audiovisuals in the NLM collections
- MeSH: detailed information about NLM's controlled vocabulary

One of the unique features of ENTREZ is that it was the first molecular biology database to incorporate links from one type of data (e.g. nucleotide) to the others (e.g. to proteins via translations, to MEDLINE entries via their MeSH numbers (NLM's Medical Subject Headings)). In addition, it incorporates an algorithm that identifies *related* entries in the databases. By *related*, we might mean genes in the same multigene family, or articles written about genes that have the same function, other proteins that function in the same biochemical pathway. Hence besides requesting the sequence for something, you can also find "everything else *like* this one". Thus, you can request MEDLINE abstracts of papers that are on similar or related topics (without any prior knowledge of their existence). Besides these "soft-links" via MeSH numbers there are also hard links encoded in the database that relate the abstract of the paper that reported the sequence or the protein entry of the nucleotide sequence.

When using the ENTREZ program, as with any web search engnine you must learn how to limit your search and to perform (as far as possible) formated queries. In the web based form of the ENTREZ program this is done through the "Limits"

Table 4.4: Some ENTREZ PubMed search fields

Field	Short term	Field	Short term
Affiliation	AD	All Fields	ALL
Author	AU	Corporate Author	CN
EC/RN Number	RN	Entrez Date	EDAT
Filter	FILTER	First Author	1AU
Full Author Name	FAU	Grant Name	GR
Issue	IP	Investigator	IR
Journal Title	TA	Language	LA
MeSH Date	MHDA	MeSH Major Topic	MAJR
MeSH Subheadings	SH	MeSH Terms	MH
NLM Unique ID	JID	Other Term	OT
Pagination	PG	Personal Name as Subject	PS
Pharmacological Action	PA	Place of Publication	PL
Publication Date	DP	Publication Type	PT
Publisher Identifier	AID	Secondary Source ID	SI
Subset	SB	Substance Name	NM
Text Words	TW	Title	TI
Title/Abstract	TIAB	Unique Identifiers	UID
Volume	VI		

tab and the "Preview/Index" tab located just below the query entry box. These tabs permit the search to be restricted to individual organisms, to particular features and so on. They permit previous queries to be combined with logical operators such as AND, OR, NOT. The "neighbor" tab will be found among one of the many menu items that make this program a very powerful search engine. These are best explored through actual use.

The search fields that are available are shown in Table 4.3. These fields can be entered directly into the query search as "(adh OR mdh) AND Drosophila [ORGN] AND 1000:5000 [SLEN]" for example. The square brackets limit the previous term to the designated field – in this case search for the word Drosophila only in the organism field (ORGN). But the adh and mdh terms are searched in all fields by default. The range operator ':' is permissible with the ACCN, MOLWT, and SLEN fields. The boolean terms are AND, OR, NOT — they must be in upper case and can be combined with brackets ('(','\)') to clarify meaning.

The search fields for Entrez PubMed are slightly different from these and are shown in Table 4.4. The boolean operators ("AND", "OR", etc.) are the same and all of these can be combined to yield a highly structured query. The documentation for PubMed can be found at http://www.ncbi.nlm.nih.gov/entrez/query/static/help/pmhelp.html.

The ENTREZ program is also released as a standalone program free of charge and you can download them to your computer from the NCBI ftp site.

4.8.2 NCBI retrieve

Discontinued Apr 2002: a great loss of convienence!

There is also an e-mail retrieval service at NCBI that is at times more convenient when you want to retreive large numbers of entries and when you are not interested in an interactive display (e.g. for remote access via a comptuer program). Mail your inquiry off to query@ncbi.nlm.nih.gov with the following example format for your query.

```
DATALIB gb
TITLES
MAXDOCS 30
BEGIN
BOVPRL
J02459 [ACC]
```

The DATALIB must be gb or genbank (GenBank), gbu or gbupdate (only updates since the last release), gbonly (official release only), emb or embl (EMBL), emblu or emblupdate (only updates since the last release), emblonly (full release only),sp or swiss or swissprot (Swiss-Prot), spu or swissprotupdate (updates only), pir (PIR database), omim (OMIM), vector (vector sequences), gp or genpept (translated GenBank), gpu or gpuupdate (updates only), kabatnuc (immunological nucleotide sequences), kabatpro (immunological protein sequences), and, though not stated in the official documentation, MEDLINE also works.

TITLES will display only the title of the matching record. MAXDOCS/MAXLINES restrict the volume of returns. Only DATALIB and BEGIN are mandatory. The above will search for records with LOCUS titles "BOVPRL" or accession number J02459. (NOTE: to put an underscore in the search, enclose the locus name in double quotes).

This retrieval service permits boolean searches. A logical **OR** is the implied default - as above, BOVPRL **or** J02459. But a logical **AND** and a logical **NOT** can be added to the query. Hence, "BOVPRL AND J02459" will retrieve records with both BOVPRL and J02459 in the record. The queries can be constructed with parenthesis to group items and with asterisks to match anything. For example, "(lysine OR glutamine) NOT vitelloge*". The field restrictor [ACC] restricts J02459 to be located in the accession number field. The field restrictors (the three letter codes) for the major databases are:

```
#        GENBANK and GBUPDATE Field Descriptions
DEFINITION [DEF]LOCUS [LOC] ACCESSION NO. [ACC]
KEYWORDS [KEY] SEGMENT [SEG] SOURCE [SRC]
REFERENCE [REF] COMMENT [COM] FEATURES [FEA]
ORIGIN [ORG]

#        EMBL and EMBLUPDATE Field Descriptions
DEFINITION [DEF] ID [LOC] ACCESSION [ACC]
KEYWORDS [KEY] DATES [DAT] SOURCE [SRC]
CROSS-REF [DXR] REFERENCE [REF] COMMENT [COM]
FEATURES [FEA]

#        SWISS-PROT Field Descriptions
DEFINITION [DEF] ID [LOC]  ACCESSION [ACC]
KEYWORDS [KEY]  DATES [DAT]  GENE NAME[GEN]
SOURCE [SRC]  ORGANISM CLASSIFICATION [CLS]
ORGANELLE [ORG]  REFERENCE [REF]  COMMENT [COM]
FEATURES [FEA]  CROSS REFERENCE [DCR]  SEQUENCE DATA [BAS]

#        PIR Protein Data Base (NBRF)
DEFINITION [DEF] ALT-NAME [ALT]  SUMMARY [SUM]
DATE [DAT]  SUPERFAMILY [SUP]  ACCESSION NO [ACC]
HOST [HST]  KEYWORDS [KEY]
SOURCE [SRC]  GENETICS [GEN]  INCLUDES [INC]
REFERENCE [REF]  COMMENT [COM]  FEATURES [FEA]

#        Online Mendelian Inheritance in Man (OMIM)
MIM NUMBER [NO] TITLE [TI] MINI-MIN [MN]
TEXT [TX] ALLELIC VARIANTS [AV] SEE ALSO REFERENCES [SA]
REFERENCES [RF] CLINICAL SYNOPSES [CS] CREATION DATE [CD]
EDIT HISTORY [ED]

#        Brookhaven Protein Data Bank (PDB)
DEFINITION [DEF] HEADER [HDR] ACCESSION [ACC]
DATE [DAT] SOURCE [SRC] AUTHOR [AUT]
SUPERSEDE [SPR] REFERENCE [REF] COMMENTS [COM]
FOOTNOTE [FTN] HETEROGENS [HET]
```

4.8.3 EMBL get

EMBL sequences can also be obtained either via the emblfetch program. In addition to this simple search form, there is also a more extensive and powerful internet form that permits many databases to be searched at once. This is termed the SRS (sequence retrieval system) which has been developed by LION Bioscience and released free of charge for academic use. The particular feature of SRS is its ability to link seamlessly between multiple life science databases and to integrate this data.

Data can also be obtained via an e-mail message send to the databases at netserv@ebi.ac.uk. The subject line of an e-mail message should be blank. In the interior of the message put the following:

```
get nuc:pip03xx
get nuc:x03392
get prot:kap_yeast
```

This will get the sequence with accession numbers pip03xx and x03392 from the nucleotide databases and the protein sequence with locus name kap_yeast from the protein database.

4.8.4 Others

There are, again, many other database access tools. For example, there is the DBGET system. This is run out of Japan (the Supercomputer Laboratory (SCL) in Kyoto and the Human Genome Center (HGC) in Tokyo). Once again, this search engine can find relevant data from several databases, including:

```
DNA: GenBank and EMBL
       GenBank: nucleic acid sequence database
       EMBL: nucleic acid sequence database
Protein: SWISS-PROT, PIR, PRF and PDBSTR
       SWISS-PROT: protein sequence database
       PIR: protein sequence database
       PRF: protein sequence database
       PDBSTR: Re-organized Protein Data Bank
KEGG Pathway Database
       PATHWAY: KEGG Pathway Database
       GENES: KEGG Genes Database
       BRITE: Biomolecular Relations in Information Transmission and Expression
       LIGAND: Ligand chemical database for enzyme reactions
PMD: Protein Mutant Database
PDB: Protein Structure Database
AAindex: Amino Acid index database
LITDB: PRF protein/peptide literature database
OMIM: Online Mendelian Inheritance in Man
Medline: Literature database
EPD: Eukaryotic promoter database
TRANSFAC: Transcription factor database
MotifDic: Dictionary of protein sites and patterns
```

Each of the databases can have individual access tools that can provide more specialized access. For example, the PDB database supports viewing of protein structures via VRML (virtual reality modelling language), Rasmol (a freely available program for displaying molecules in three dimensions), FirstGlance and Protein Explorer (two other programs that require a commercial product), and via still photographs. There is also a special browser for the SWISS 3DIMAGE database and so on.

For each database, look for a specialized browser to access the data making use of the peculiarities of the data stored. If you are not sure what database to be searching, try the very capable search launcher from the Baylor College of Medicine, Houston.

4.9 Reliability

The data within the databases may not always be what it pretends to be. This venture is a human one and humans make mistakes. Indeed, the venture relies on the contributions of many people and they all have different standards of accuracy. One of the most common errors in the early days was the presence of vector sequence in the midst of some other sequence. Today this is not such a large problem since most entries are now automatically screened against known vectors and the error can be caught before the sequence makes it into the databases.

Smaller errors in sequences are also common. The human APRT gene sequence was determined and entered into the data base by one laboratory. A few months later, another laboratory published a paper with a sequence that differed from the previous work by 13 nucleotides and 60 insertions/deletions over 3 kb. It is impossible to tell how much of this may be due to polymorphism and how much may be due to actual sequencing error. Because this kind of duplication is not done for every sequence it is impossible to say that this is typical or atypical of the sequencing done. However, as a counter example, a check of the yeast genome revealed only a couple of differences over many megabases (H. Bussey - personal communication). Unfortunately, many errors are not easily corrected. Current policy for most of the databases is that the people running them are responsible for the database en masse while the actual data is the business of the researchers. On a more positive note, you will also find many entries that were created long ago and yet, last modified very recently to incorporate the latest information.

Another problem that has shown up is trivial data entries. The following entry was noticed by Reinhard Doelz.

Database Silliness

```
LOCUS       A00674          6 bp    DNA             PAT       29-JAN-1993
DEFINITION  Nucleotide sequence 3 from patent number WO8601533
ACCESSION   A00674
KEYWORDS    .
SOURCE      Unknown
  ORGANISM  Unknown
            Unclassified.
REFERENCE   1  (bases 1 to 6)
  AUTHORS
  TITLE     'PRODUCTION OF CHIMERIC ANTIBODIES'
  JOURNAL   Patent: WO 8601533-A 3 13-MAR-1986;
  STANDARD  full automatic
BASE COUNT        3 a      2 c      0 g      1 t
ORIGIN
        1 cactaa
//
```

This is truly an amazing entry. It is fully six nucleotides long, it comes from an unknown source, it comes from an unknown organism and from unknown authors. But it is patented !! What could possibly be the purpose of entering this sequence in the database and even more incredulously, why would one ever patent it? By random chance your DNA must contain this sequence. Since it does and the sequence was patented, be forewarned that you should obtain correct written permission from the patent holders before you replicate it again. More seriously, if you construct oligos for PCR or sequencing, you are probably guilty of patent infringement. Reinhard calculated that this silly hexanucleotide occurs 28340 times within the database and in over 70000 sequences (circa 1993). This entry has now been deleted from the databases but there are other, less extreme entries of doubtful quality.

The take home message from all of this is to look at the data with a critical eye. The actual quantity and type of errors within the databases are not known - some researchers are very careful and can check their sequence data, for others a double check of the sequence data may not be possible. When doing your own research, assume that the sequence may contain some errors and take measures to prevent this from destroying the validity of your conclusions.

Caveat Emptor

Chapter 5

Sequence File Formats

There are many formats that sequence data can be presented in. Each has advantages over the others (e.g. some are small and compact; others contain lots of information) and different programs require different formats as their input. The major databases permit sequences to be stored on your local computer in more than one format and there are programs that will convert one format to another. The most popular of these is a program called `readseq` by D.G. Gilbert (available for UNIX, DOS and APPLE machines).

The `GenBank` and `EMBL` formats have been discussed above. Both the `GenBank` and `EMBL` formats are highly stylized and strictly controlled to conform to consistent standards. Other popular formats are `ASN.1`, `DNAStrider`, `Fitch`, `GCG`, `GDE`, `HENNIG86`, `IG/Stanford`, `MSF`, `NBRF`, `NEXUS`. `PIR/CODATA`, `Pearson/Fasta`, `Phylip - Interleaved`, `Phylip - Sequential`, and `Plain/Raw`, I will not present all here but rather just a smattering.

Most formats will ignore case and this can therefore often be used to add information about the sequences. While the `GenBank` and `EMBL` formats can contain the character '-', they generally do not contian these characters and these formats were not intended to convey the kind of information that includes homologous sites between multiple sequences (the dashes indicate conceptual gaps in the sequences that have been inserted so that homologous parts of the sequence from each species are in the same location).

5.1 Genbank/EMBL

As a quick review, these two formats would be . . .

```
LOCUS       MPU28721     607 bp    DNA             ROD       28-JUN-1995
DEFINITION  Mus pahari adenine phosphoribosyltransferase (APRT) gene, complete
            cds.
ACCESSION   U28721
NID         g881573
KEYWORDS    .
SOURCE      shrew mouse.
  ORGANISM  Mus pahari
            Eukaryotae; mitochondrial eukaryotes; Metazoa; Chordata;
            Vertebrata; Eutheria; Rodentia; Sciurognathi; Myomorpha; Muridae;
            Murinae; Mus.
REFERENCE   1  (bases 1 to 2283)
  AUTHORS   Fieldhouse,D. and Golding,G.B.
  TITLE     Rates of substitution in closely related rodent species
  JOURNAL   Unpublished
REFERENCE   2  (bases 1 to 2283)
  AUTHORS   Fieldhouse,D.
  TITLE     Direct Submission
  JOURNAL   Submitted (07-JUN-1995) Dan Fieldhouse, Biology, McMaster
            University, 1280 Main Street West, Hamilton, ON, L8S 4K1, Canada
FEATURES             Location/Qualifiers
     source          1..2283
                     /organism="Mus pahari"
                     /db_xref="taxon:10093"
     gene            join(46..125,256..362,1509..1642,1847..1925,2044..2186)
```

```
                     /gene="APRT"
     CDS             join(46..125,256..362,1509..1642,1847..1925,2044..2186)
                     /gene="APRT"
                     /EC_number="2.4.2.7"
                     /note="purine salvage enzyme"
                     /codon_start=1
                     /product="adenine phosphoribosyltransferase"
                     /db_xref="PID:g881574"
                     /translation="MSESELKLVARRIRSFPDFPIPGVLFRDISPLLKDPDSFRASIR
                     LLASHLKSTHSGKIDYIAGLDSRGFLFGPSLAQELGVGCVLIRKQGKLPGPTISASYA
                     LEYGKAELEIQKDALEPGQRVVIVDDLLATGGTMFAACDLLHQLRAEVVECVSLVELT
                     SLKGRERLGPIPFFSLLQYD"
BASE COUNT       87 a    228 c    145 g    147 t
ORIGIN
        1 cctgcggata ctcacctcct ccttgtctcc tacaagcacg cggccatgtc cgagtctgag
       61 ttgaaactgg tggcgcggcg catccgcagc ttccccgact tccccatccc gggcgtgctg
      121 ttcaggtgcg gtcacgagcc ggcgaggcgt tggcgccgta ctctcatccc ccggcgcagg
      181 cgcgtgggca gccttgggga tcttgcgggg cctctgcccg gccacacgcg gtcactctcc
      241 tgtccttgtt cccagggata tctcgcccct cttgaaagat ccggactcct tccgagcttc
      301 catccgcctc ctggccagtc acctgaagtc cacgcacagc ggcaagatcg actatatcgc
      361 agggcaaggt ggccttgcta ggccgtactc atcccccacg gtcctatccc ctatcccctt
      421 tcccctcgtg tcacccacag tctaccccac acccatccat tctttcttta acctctgact
      481 cttcctcctt ggtttctcac tgccttggac gcttgttcac cccggatgaa ctccgtaggc
      541 gtctcccttc cctgcttggt accctaaggt gccctcggtg cttgttcgta gagacgaact
      601 ctgctct
//
```

and

```
ID   MP28721    standard; DNA; ROD; 607 BP.
XX
AC   U28721;
XX
NI   g881573
XX
DT   04-JUL-1995 (Rel. 44, Created)
DT   04-JUL-1995 (Rel. 44, Last updated, Version 1)
XX
DE   Mus pahari adenine phosphoribosyltransferase (APRT) gene, complete
DE   cds.
XX
KW   .
XX
OS   Mus pahari (shrew mouse)
OC   Eukaryotae; mitochondrial eukaryotes; Metazoa; Chordata; Vertebrata;
OC   Mammalia; Eutheria; Rodentia; Sciurognathi; Muridae; Murinae; Mus.
XX
RN   [1]
RP   1-2283
RA   Fieldhouse D., Golding G.B.;
RT   "Rates of substitution in closely related rodent species";
RL   Unpublished.
XX
RN   [2]
RP   1-2283
RA   Fieldhouse D.;
RT   ;
RL   Submitted (07-JUN-1995) to the EMBL/GenBank/DDBJ databases.
RL   Dan Fieldhouse, Biology, McMaster University, 1280 Main Street West,
RL   Hamilton, ON, L8S 4K1, Canada
XX
DR   SWISS-PROT; P47956; APT_MUSPA.
XX
CC   NCBI gi: 881573
XX
FH   Key             Location/Qualifiers
FH
FT   source          1. .2283
FT                   /organism="Mus pahari"
FT   CDS             join(46. .125,256. .362,1509. .1642,1847. .1925,2044. .2186)
FT                   /codon_start=1
FT                   /db_xref="PID:g881574"
FT                   /db_xref="SWISS-PROT:P47956"
FT                   /note="purine salvage enzyme; Method: conceptual
FT                   translation supplied by author. NCBI gi: 881574"
FT                   /gene="APRT"
FT                   /EC_number="2.4.2.7"
FT                   /product="adenine phosphoribosyltransferase"
FT                   /translation="MSESELKLVARRIRSFPDFPIPGVLFRDISPLLKDPDSFRASIRL
FT                   LASHLKSTHSGKIDYIAGLDSRGFLFGPSLAQELGVGCVLIRKQGKLPGPTISASYALE
FT                   YGKAELEIQKDALEPGQRVVIVDDLLATGGTMFAACDLLHQLRAEVVECVSLVELTSLK
FT                   GRERLGPIPFFSLLQYD"
XX
SQ   Sequence  607 BP;  87 A; 228 C; 145 G; 147 T; 0 other;
     CCTGCGGATA CTCACCTCCT CCTTGTCTCC TACAAGCACG CGGCCATGTC CGAGTCTGAG        60
     TTGAAACTGG TGGCGCGGCG CATCCGCAGC TTCCCCGACT TCCCCATCCC GGGCGTGCTG       120
     TTCAGGTGCG GTCACGAGCC GGCGAGGCGT TGGCGCCGTA CTCTCATCCC CCGGCGCAGG       180
     CGCGTGGGCA GCCTTGGGGA TCTTGCGGGG CCTCTGCCCG GCCACACGCG GTCACTCTCC       240
     TGTCCTTGTT CCCAGGGATA TCTCGCCCCT CTTGAAAGAT CCGGACTCCT TCCGAGCTTC       300
     CATCCGCCTC CTGGCCAGTC ACCTGAAGTC CACGCACAGC GGCAAGATCG ACTATATCGC       360
     AGGGCAAGGT GGCCTTGCTA GGCCGTACTC ATCCCCCACG GTCCTATCCC CTATCCCCTT       420
     TCCCCTCGTG TCACCCACAG TCTACCCCAC ACCCATCCAT TCTTTCTTTA ACCTCTGACT       480
     CTTCCTCCTT GGTTTCTCAC TGCCTTGGAC GCTTGTTCAC CCCGGATGAA CTCCGTAGGC       540
     GTCTCCCTTC CCTGCTTGGT ACCCTAAGGT GCCCTCGGTG CTTGTTCGTA GAGACGAACT       600
     CTGCTCT                                                                 607
//
```

In the `Genbank` format, sequence information is set aside with key words. The entire entry begins with the keyword `LOCUS` at the beginning of a line and ends with `//`. Different features are set off with different keywords; the sequence information itself with the keyword `ORIGIN`.

The `EMBL` format is similar but with two-letter codes at the beginning of each line to designate different features of the entry (much easier to program). The entire entry begins with the key `ID` at the beginning of a line and ends with `//`.

5.2 FASTA

By far the simplest format is termed the `fasta` (also known as the `Pearson` format). This sequence format contains the minimal amount of information. A `fasta` file will contain just a '>' sign (at the beginning of a line) to indicate the beginning of a new sequence and a word (phrase) to serve as the sequence title. The sequence information itself follows immediately. No other information is stored within a `fasta` file. As an example, I will use a proportion of the *Mus pahari, Mus spicilegus* and *Gerbillus campestris* APRT gene sequences. These sequences would appear as ...

```
>MPU28721       650 bp      1/31/98 14:18:24, 650 bases, F8A0A666 checksum.
------------------------------------CCTGCGGATACTCA
CCTCCTCCTTGTCTCCTACAAGCACGCGGCCATGTCCGAGTCTGAGTTGA
AACTGGTGGCGCGGCGCATCCGCAGCTTCCCCGACTTCCCCATCCCGGGC
GTGCTGTTCAGGTGCGGTCACGAGCCGGCGAGGCGTTGGCGCCGTACTCT
CATCCC-CCGGCGCAGGCGCGTGGGCAGCCTTGGGGATCTTGCGGGGCCT
CTGCCCGGCCACACGCGG-TCACTCTCCTGTCCTTGTTCCCAGGGATATC
TCGCCCCTCTTGAAAGATCCGGACTCCTTCCGAGCTTCCATCCGCCTCCT
GGCCAGTCACCTGAAGTCCACGCACAGCGGCAAGATCGACTATATCGCAG
GGCAAGGTGGCCTTGCTAGGCCGTACTCATCCCCCACGGTCCTATCCCCT
ATCCCCTTTCCCC-TCGTGTCACCCACAGTCTACCCCACACCCATCCATT
CTTTCTTTAACCTCTGACTCTTCCTCCTTGGTTTCTCACTGCCTTGGACG
CTTGTTCACCCCGGATGAACTCCGTAGGCGTCTCCCTTCCCTGCTTGGTA
CCCTAAGG----TGCCCTCGGTGCTTGTTCGTAGAGACGAACTCTGCTCT
>MSU28720       650 bp      1/31/98 14:18:24, 650 bases, 450AB895 checksum.
---------------TCGGATTGACGTGAATTTAGCGTGCTGATACCTA
CCTCCTCCTTGCCTCCTACACGCACGCGGCCATGTCCGAACCTGAGTTGA
AACTGGTGGCGCGGCGCATCCGCAGCTTCCCCGACTTCCCAATCCCGGGC
GTGCTGTTCAGGTGCGGTCACGAGCCGGCGAGGCGTTGGCGCCGTACGCT
CATCCC-CCGGCGCAGGCGCGTAGGCAGCCTCGGGGATCTTGCGGGGCCT
CTGCCCGGCCACACGCGGGTCACTCTCCTGTCCTTGTTCCCAGGGATATC
TCGCCCCTCTTGAAAGACCCGGACTCCTTCCGAGCTTCCATCCGCCTCTT
GGCCAGTCACCTGAAGTCCACGCACAGCGGCAAGATCGACTACATCGCAG
GCGA--GTGGCCTTGCTAGGCCGTGCTCGTCCCCCACGGTCCTAGCCCCT
ATCCCCTTTCCCCCTCGTGTCACCCACAGTCTGCCCCACACCCATCCATT
CTTTCTTCAACCTCTGACACTTCCTCCTTGGTTCCTCACTGCCTTGGACG
CTTGTTCACCCCGGATGAACTATGTAGGAGTCTCCCTTCCCTGCTAGGTA
CCCTAAGGCATCTGCCCTCGGTGCTTGTTCCTAGAGACGAACTCTGCTCT
>GCU28961       650 bp      1/31/98 14:18:24, 650 bases, 606DF2D9 checksum.
CCTCCGCCCTTGTTCCTGGGACAGGCTTGACCCTAGCCAGTTGACACCTC
ACCTCCGCCCTTCCTCT-CACGCACGCGGCCATGGCGGAACCCGAGTTGC
AGCTGGTGGCGCGGCGCATCCGCAGCTTCCCCGACTTCCCCATCCCGGGC
GTGCTGTTCAGGTGCGTCCACGAGCCGCCCAGGCGTTGGCGCTGCGTCCT
CAGCCCTCCGGCGCAGGCGCGTGAGCTGTCTCCGGGATCTTGCGGGGCCT
CCGCCCAGCCATACCCAAGTCACCATCCTG----TGTTCCCAGGGATATC
TCGCCCCTCCTGAAAGACCCGGACTCCTTCCGAGCTTCCATCCGTCTCCT
GGCCAACCATCTGAAGTCCAAGCATGGCGGCAAAATCGACTACATCGCAG
GCGA--GTGTTCTTGCTAGGCCGTGCCCGTTCCC-ACTGTCAGGGCCGCC
```

```
ATCCCGTGTTCCC---------TTTTTCGTGTCACCCACACCCACCCCTC
CTTTCTCTGACACTCCCAAGTTCCCT----GTTCCTCTCTGCCTTGGTCC
CATATTCACCCCGGATGA-CTGCGGAGTCTCCCACCCTCTGACCTCTGCT
CTCAAAGC----------CTGTCCCTAC---TAGAGAGGAACTCTGCTCT
```

Note that although it is a simple format, sequence alignment information (more on this later) can be indicated by the dashes.

5.3 GDE

The GDE format will also contain this kind of alignment information but note that it may have an 'offset' value. This (often annoying) feature permits a compact storage of sequence information at the tails of the sequence. An 'offset' of 36 means to insert 36 '-' in front of the sequence in order to properly line it up with the other sequences. This format can also contain all the information that is present in a GenBank format but does so simply as a 'comment' enclosed in quotation marks and any information may appear within the comment field. The example *Mus pahari, Mus spicilegus* and *Gerbillus campestris* APRT gene sequences in a GDE format would appear as . . .

```
{
name "MPU28721"
type "DNA"
longname Mus pahari
sequence-ID "U28721"
descrip "Mus pahari adenine phosphoribosyltransferase (APRT) gene, complete cds"
creator "Fieldhouse,D. and Golding,G.B."
offset 36
creation-date  1/31/98 14:18:24
direction 1
strandedness 1
comments "
NID         g881573
KEYWORDS    .
SOURCE      shrew mouse.
            Eukaryotae; mitochondrial eukaryotes; Metazoa; Chordata;
            Vertebrata; Eutheria; Rodentia; Sciurognathi; Myomorpha; Muridae;
            Murinae; Mus.
REFERENCE   1  (bases 1 to 2283)
  TITLE     Rates of substitution in closely related rodent species
  JOURNAL   Unpublished
REFERENCE   2  (bases 1 to 2283)
  TITLE     Direct Submission
  JOURNAL   Submitted (07-JUN-1995) Dan Fieldhouse, Biology, McMaster
            University, 1280 Main Street West, Hamilton, ON, L8S 4K1, Canada
FEATURES             Location/Qualifiers
     source          1..2283
                     /organism='Mus pahari'
                     /db_xref='taxon:10093'
     gene            join(46..125,256..362,1509..1642,1847..1925,2044..2186)
                     /gene='APRT'
     CDS             join(46..125,256..362,1509..1642,1847..1925,2044..2186)
                     /gene='APRT'
                     /EC_number='2.4.2.7'
                     /note='purine salvage enzyme'
                     /codon_start=1
                     /product='adenine phosphoribosyltransferase'
                     /db_xref='PID:g881574'
                     /translation='MSESELKLVARRIRSFPDFPIPGVLFRDISPLLKDPDSFRASIR
                     LLASHLKSTHSGKIDYIAGLDSRGFLFGPSLAQELGVGCVLIRKQGKLPGPTISASYA
                     LEYGKAELEIQKDALEPGQRVVIVDDLLATGGTMFAACDLLHQLRAEVVECVSLVELT
                     SLKGRERLGPIPFFSLLQYD'
BASE COUNT      485 a    696 c    590 g    512 t
"
sequence "CCTGCGGATACTCACCTCCTCCTT
GTCTCCTACAAGCACGCGGCCATGTCCGAGTCTGAGTTGAAACTGGTGGCGCGGCGCATC
CGCAGCTTCCCCGACTTCCCCATCCCGGGCGTGCTGTTCAGGTGCGGTCACGAGCCGGCG
AGGCGTTGGCGCCGTACTCTCATCCC-CCGGCGCAGGCGCGTGGGCAGCCTTGGGGATCT
TGCGGGGCCTCTGCCCGGCCACACGCGG-TCACTCTCCTGTCCTTGTTCCCAGGGATATC
TCGCCCCTCTTGAAAGATCCGGACTCCTTCCGAGCTTCCATCCGCCTCCTGGCCAGTCAC
CTGAAGTCCACGCACAGCGGCAAGATCGACTATATCGCAGGGCAAGGTGGCCTTGCTAGG
CCGTACTCATCCCCCACGGTCCTATCCCCTATCCCCTTTCCCC-TCGTGTCACCCACAGT
CTACCCCACACCCATCCATTCTTTCTTTAACCTCTGACTCTTCCTCCTTGGTTTCTCACT
GCCTTGGACGCTTGTTCACCCCGGATGAACTCCGTAGGCGTCTCCCTTCCCTGCTTGGTA
CCCTAAGG----TGCCCTCGGTGCTTGTTCGTAGAGACGAACTCTGCTCT"
}
```

```
{
name "MSU28720"
type "DNA"
longname Mus spicilegus
sequence-ID "U28720"
descrip "Mus spicilegus adenine phosphoribosyltransferase (APRT) gene,"
creator "Fieldhouse,D. and Golding,G.B."
offset 15
creation-date  1/31/98 14:18:24
direction 1
strandedness 1
comments "
NID         g881575
KEYWORDS    .
SOURCE      Steppe mouse.
            Eukaryotae; mitochondrial eukaryotes; Metazoa; Chordata;
            Vertebrata; Eutheria; Rodentia; Sciurognathi; Myomorpha; Muridae;
            Murinae; Mus.
REFERENCE   1  (bases 1 to 2117)
  TITLE     Rates of substitution in closely related rodent species
  JOURNAL   Unpublished
REFERENCE   2  (bases 1 to 2117)
  TITLE     Direct Submission
  JOURNAL   Submitted (07-JUN-1995) Dan Fieldhouse, Biology, McMaster
            University, 1280 Main Street West, Hamilton, ON, L8S 4K1, Canada
FEATURES             Location/Qualifiers
     source          1..2117
                     /organism=‘Mus spicilegus‘
                     /db_xref=‘taxon:10103‘
     gene            join(67..146,278..384,1355..1488,1675..1753,1860..2002)
                     /gene=‘APRT‘
     CDS             join(67..146,278..384,1355..1488,1675..1753,1860..2002)
                     /gene=‘APRT‘
                     /EC_number=‘2.4.2.7‘
                     /note=‘purine salvage enzyme‘
                     /codon_start=1
                     /product=‘adenine phosphoribosyltransferase‘
                     /db_xref=‘PID:g881576‘
                     /translation=‘MSEPELKLVARRIRSFPDFPIPGVLFRDISPLLKDPDSFRASIR
                     LLASHLKSTHSGKIDYIAGLDSRGFLFGPSLAQELGVGCVLIRKQGKLPGPTVSASYS
                     LEYGKAELEIQKDALEPGQRVVIVDDLLATGGTMFAACDLLHQLRAEVVECVSLVELT
                     SLKGRERLGPIPFFSLLQYD‘
BASE COUNT      413 a    652 c    564 g    488 t"
sequence "TCGGGATTGACGTGAATTTAGCGTGCTGATACCTACCTCCTCCTT
GCCTCCTACACGCACGCGGCCATGTCCGAACCTGAGTTGAAACTGGTGGCGCGGCGCATC
CGCAGCTTCCCCGACTTCCCAATCCCGGGCGTGCTGTTCAGGTGCGGTCACGAGCCGGCG
AGGCGTTGGCGCCGTACGCTCATCCC-CCGGCGCAGGCGCGTAGGCAGCCTCGGGGATCT
TGCGGGGCCTCTGCCCGGCCACACGCGGGTCACTCTCCTGTCCTTGTTCCCAGGGATATC
TCGCCCCTCTTGAAAGACCCGGACTCCTTCCGAGCTTCCATCCGCCTCTTGGCCAGTCAC
CTGAAGTCCACGCACAGCGGCAAGATCGACTACATCGCAGGCGA--GTGGCCTTGCTAGG
CCGTGCTCGTCCCCCACGGTCCTAGCCCCTATCCCCTTTCCCCCTCGTGTCACCCACAGT
CTGCCCCACACCCATCCATTCTTTCTTCAACCTCTGACACTTCCTCCTTGGTTCCTCACT
GCCTTGGACGCTTGTTCACCCCGGATGAACTATGTAGGAGTCTCCCTTCCCTGCTAGGTA
CCCTAAGGCATCTGCCCTCGGTGCTTGTTCCTAGAGACGAACTCTGCTCT"
}
{
name "GCU28961"
type "DNA"
longname Gerbillus campestris
sequence-ID "U28961"
descrip "Gerbillus campestris adenine phosphoribosyltransferase (APRT) gene,"
creator "Yazdani,F. and Golding,G.B."
creation-date  1/31/98 14:18:24
direction 1
strandedness 1
comments "
NID         g899456
KEYWORDS    .
SOURCE      Gerbillus campestris.
            Eukaryotae; mitochondrial eukaryotes; Metazoa; Chordata;
            Vertebrata; Eutheria; Rodentia; Sciurognathi; Myomorpha; Muridae;
            Gerbillinae; Gerbillus.
REFERENCE   1  (bases 1 to 2076)
  TITLE     Rates of substitution in closely related rodent species
  JOURNAL   Unpublished
REFERENCE   2  (bases 1 to 2076)
  TITLE     Direct Submission
  JOURNAL   Submitted (12-JUN-1995) Fariborz Yazdani, Biology, McMaster
            University, 1280 Main Street West, Hamilton, Ont L8S 4K1, Canada
FEATURES             Location/Qualifiers
```

```
     source          1..2076
                     /organism='Gerbillus campestris'
                     /db_xref='taxon:41199'
     gene            join(81..160,289..395,1313..1446,1649..1727,1828..1970)
                     /gene='APRT'
     exon            >81..160
                     /gene='APRT'
     CDS             join(81..160,289..395,1313..1446,1649..1727,1828..1970)
                     /gene='APRT'
                     /EC_number='2.4.2.7'
                     /note='purine salvage enzyme'
                     /codon_start=1
                     /product='adenine phosphoribosyltransferase'
                     /db_xref='PID:g899457'
                     /translation='MAEPELQLVARRIRSFPDFPIPGVLFRDISPLLKDPDSFRASIR
                     LLANHLKSKHGGKIDYIAGLDSRGFLFGPSLAQELGLGCVLIRKRGKLPGPTVSASYA
                     LEYGKAELEIQKDALEPGQKVVIVDDLLATGGTMCAACQLLGQLRAEVVECVSLVELT
                     SLKGREKLGPVPFFSLLQYE'
     intron          161..288
                     /gene='APRT'
     exon            289..395
                     /gene='APRT'
     intron          396..1312
                     /gene='APRT'
     exon            1313..1446
                     /gene='APRT'
     intron          1447..1648
                     /gene='APRT'
     exon            1649..1727
                     /gene='APRT'
     intron          1728..1827
                     /gene='APRT'
     exon            1828..>1970
                     /gene='APRT'
BASE COUNT      385 a    666 c    577 g    448 t"
sequence "
CCTCCGCCCTTGTTCCTGGGACAGGCTTGACCCTAGCCAGTTGACACCTCACCTCCGCCC
TTCCTCT-CACGCACGCGGCCATGGCGGAACCCGAGTTGCAGCTGGTGGCGCGGCGCATC
CGCAGCTTCCCCGACTTCCCCATCCCGGGCGTGCTGTTCAGGTGCGTCCACGAGCCGCCC
AGGCGTTGGCGCTGCGTCCTCAGCCCTCCGGCGCAGGCGCGTGAGCTGTCTCCGGGATCT
TGCGGGGCCTCCGCCCAGCCATACCCAAGTCACCATCCTG----TGTTCCCAGGGATATC
TCGCCCCTCCTGAAAGACCCGGACTCCTTCCGAGCTTCCATCCGTCTCCTGGCCAACCAT
CTGAAGTCCAAGCATGGCGGCAAAATCGACTACATCGCAGGCGA--GTGTTCTTGCTAGG
CCGTGCCCGTTCCC-ACTGTCAGGGCCGCCATCCCGTGTTCCC---------TTTTTCGT
GTCACCCACACCCACCCCTCCTTTCTCTGACACTCCCAAGTTCCCT----GTTCCTCTCT
GCCTTGGTCCCATATTCACCCCGGATGA-CTGCGGAGTCTCCCACCCTCTGACCTCTGCT
CTCAAAGC----------CTGTCCCTAC---TAGAGAGGAACTCTGCTCT"
}
```

5.4 NEXUS

The popular `PAUP`, `MacClade` and `Mr.Bayes` programs (and others) use a `NEXUS` format (Maddison, Swofford and Maddison 1997. Syst. Biol., 46, 590-621). The primary feature of this format is its modularity. Files identify themselves with the key phrase ``#NEXUS'' at the beginning of the file. Each block of information begins with ``BEGIN - - -''; and ends with ``END;''. comments can be enclosed within square brackets. For these sequences a simple translation would be

```
#NEXUS

[Name: MPU28721          Len:   650  Check:  643A358]
[Name: MSU28720          Len:   650  Check: FDC8BCDB]
[Name: GCU28961          Len:   650  Check: D8AFF697]

BEGIN TAXA;
    DIMENSIONS NTAX=3;
    TAXLABELS MPU28721 MSU28720 GCU28961;
END;

BEGIN CHARACTERS;
     DIMENSIONS NCHAR=650;
     FORMAT MISSING=? DATATYPE=DNA INTERLEAVE GAP=-;
     MATRIX
MPU28721   -------------------- ----------------CCTG CGGATACTCACCTCCTCCTT GTCTCCTACAAGCACGCGGC CATGTCCGAGTCTGAGTTGA
MSU28720   ---------------TCGGG ATTGACGTGAATTTAGCGTG CTGATACCTACCTCCTCCTT GCCTCCTACACGCACGCGGC CATGTCCGAACCTGAGTTGA
```

```
GCU28961   CCTCCGCCCTTGTTCCTGGG ACAGGCTTGACCCTAGCCAG TTGACACCTCACCTCCGCCC TTCCTCT-CACGCACGCGGC CATGGCGGAACCCGAGTTGC

MPU28721   AACTGGTGGCGCGGCGCATC CGCAGCTTCCCCGACTTCCC CATCCCGGGCGTGCTGTTCA GGTGCGGTCACGAGCCGGCG AGGCGTTGGCGCCGTACTCT
MSU28720   AACTGGTGGCGCGGCGCATC CGCAGCTTCCCCGACTTCCC AATCCCGGGCGTGCTGTTCA GGTGCGGTCACGAGCCGGCG AGGCGTTGGCGCCGTACGCT
GCU28961   AGCTGGTGGCGCGGCGCATC CGCAGCTTCCCCGACTTCCC CATCCCGGGCGTGCTGTTCA GGTGCGTCCACGAGCCGCCC AGGCGTTGGCGCTGCGTCCT

MPU28721   CATCCC-CCGGCGCAGGCGC GTGGGCAGCCTTGGGGATCT TGCGGGGCCTCTGCCCGGCC ACACGCGG-TCACTCTCCTG TCCTTGTTCCCAGGGATATC
MSU28720   CATCCC-CCGGCGCAGGCGC GTAGGCAGCCTCGGGGATCT TGCGGGGCCTCTGCCCGGCC ACACGCGGGTCACTCTCCTG TCCTTGTTCCCAGGGATATC
GCU28961   CAGCCCTCCGGCGCAGGCGC GTGAGCTGTCTCCGGGATCT TGCGGGGCCTCCGCCCAGCC ATACCCAAGTCACCATCCTG ----TGTTCCCAGGGATATC

MPU28721   TCGCCCCTCTTGAAAGATCC GGACTCCTTCCGAGCTTCCA TCCGCCTCCTGGCCAGTCAC CTGAAGTCCACGCACAGCGG CAAGATCGACTATATCGCAG
MSU28720   TCGCCCCTCTTGAAAGACCC GGACTCCTTCCGAGCTTCCA TCCGCCTCTTGGCCAGTCAC CTGAAGTCCACGCACAGCGG CAAGATCGACTACATCGCAG
GCU28961   TCGCCCCTCCTGAAAGACCC GGACTCCTTCCGAGCTTCCA TCCGTCTCCTGGCCAACCAT CTGAAGTCCAAGCATGGCGG CAAAATCGACTACATCGCAG

MPU28721   GGCAAGGTGGCCTTGCTAGG CCGTACTCATCCCCCACGGT CCTATCCCCTATCCCCTTTC CCC-TCGTGTCACCCACAGT CTACCCCACACCCATCCATT
MSU28720   GCGA--GTGGCCTTGCTAGG CCGTGCTCGTCCCCCACGGT CCTAGCCCCTATCCCCTTTC CCCCTCGTGTCACCCACAGT CTGCCCCACACCCATCCATT
GCU28961   GCGA--GTGTTCTTGCTAGG CCGTGCCCGTTCCC-ACTGT CAGGGCCGCCATCCCGTGTT CCC---------TTTTTCGT GTCACCCACACCCACCCCTC

MPU28721   CTTTCTTTAACCTCTGACTC TTCCTCCTTGGTTTCTCACT GCCTTGGACGCTTGTTCACC CCGGATGAACTCCGTAGGCG TCTCCCTTCCCTGCTTGGTA
MSU28720   CTTTCTTCAACCTCTGACAC TTCCTCCTTGGTTCCTCACT GCCTTGGACGCTTGTTCACC CCGGATGAACTATGTAGGAG TCTCCCTTCCCTGCTAGGTA
GCU28961   CTTTCTCTGACACTCCCAAG TTCCCT----GTTCCTCTCT GCCTTGGTCCCATATTCACC CCGGATGA-CTGCGGAGTCT CCCACCCTCTGACCTCTGCT

MPU28721   CCCTAAGG----TGCCCTCG GTGCTTGTTCGTAGAGACGA ACTCTGCTCT
MSU28720   CCCTAAGGCATCTGCCCTCG GTGCTTGTTCCTAGAGACGA ACTCTGCTCT
GCU28961   CTCAAAGC----------CT GTCCCTAC---TAGAGAGGA ACTCTGCTCT

    ;
END;
BEGIN TREES;
    TREE tree1 = (MPU28721, (MSU28720,GCU28961));
    TREE tree2 = (MSU28720, (MPU28721,GCU28961));
END;
BEGIN NOTES;
    PICTURE TAXON=3 FORMAT=GIF SOURCE=FILE
    PICTURE=a_rodent.gif
END;
```

The major blocks of data that the file format permits are `TAXA, CHARACTERS, UNALIGNED, DISTANCES, SETS, ASSUMPTIONS, CODONS, TREES` and `NOTES`. Only a few of these are shown above and each permits many other options. Note that the file format permits things such as the phylogeny (or tree) of a group of species to be stored, pictures of the organisms to be stored or referenced, along with many other capabilities.

5.5 PHYLIP

The PHYLIP programs are also very popular and other programs have incorporated the sequence format used by these programs. There are two formats that can be used, an interleaved and a sequential format. The `phylip-interleaved` format begins with two numbers on the first line. The first number gives the number of taxa or different sequences in the file. The second number gives the overall length of the sequences. On the next line the sequence information begins preceded by a sequence title of no more than 10 characters. The APRT sequences in this format (interleaved) would be

```
 3 650
MPU28721     ---------- ---------- ---------- ------CCTG CGGATACTCA
MSU28720     ---------- -----TCGGG ATTGACGTGA ATTTAGCGTG CTGATACCTA
GCU28961     CCTCCGCCCT TGTTCCTGGG ACAGGCTTGA CCCTAGCCAG TTGACACCTC

             CCTCCTCCTT GTCTCCTACA AGCACGCGGC CATGTCCGAG TCTGAGTTGA
             CCTCCTCCTT GCCTCCTACA CGCACGCGGC CATGTCCGAA CCTGAGTTGA
             ACCTCCGCCC TTCCTCT-CA CGCACGCGGC CATGGCGGAA CCCGAGTTGC

             AACTGGTGGC GCGGCGCATC CGCAGCTTCC CCGACTTCCC CATCCCGGGC
             AACTGGTGGC GCGGCGCATC CGCAGCTTCC CCGACTTCCC AATCCCGGGC
             AGCTGGTGGC GCGGCGCATC CGCAGCTTCC CCGACTTCCC CATCCCGGGC

             GTGCTGTTCA GGTGCGGTCA CGAGCCGGCG AGGCGTTGGC GCCGTACTCT
             GTGCTGTTCA GGTGCGGTCA CGAGCCGGCG AGGCGTTGGC GCCGTACGCT
             GTGCTGTTCA GGTGCGTCCA CGAGCCGCCC AGGCGTTGGC GCTGCGTCCT

             CATCCC-CCG GCGCAGGCGC GTGGGCAGCC TTGGGGATCT TGCGGGGCCT
             CATCCC-CCG GCGCAGGCGC GTAGGCAGCC TCGGGGATCT TGCGGGGCCT
             CAGCCCTCCG GCGCAGGCGC GTGAGCTGTC TCCGGGATCT TGCGGGGCCT
```

```
            CTGCCCGGCC ACACGCGG-T CACTCTCCTG TCCTTGTTCC CAGGGATATC
            CTGCCCGGCC ACACGCGGGT CACTCTCCTG TCCTTGTTCC CAGGGATATC
            CCGCCCAGCC ATACCCAAGT CACCATCCTG ----TGTTCC CAGGGATATC

            TCGCCCCTCT TGAAAGATCC GGACTCCTTC CGAGCTTCCA TCCGCCTCCT
            TCGCCCCTCT TGAAAGACCC GGACTCCTTC CGAGCTTCCA TCCGCCTCTT
            TCGCCCCTCC TGAAAGACCC GGACTCCTTC CGAGCTTCCA TCCGTCTCCT

            GGCCAGTCAC CTGAAGTCCA CGCACAGCGG CAAGATCGAC TATATCGCAG
            GGCCAGTCAC CTGAAGTCCA CGCACAGCGG CAAGATCGAC TACATCGCAG
            GGCCAACCAT CTGAAGTCCA AGCATGGCGG CAAAATCGAC TACATCGCAG

            GGCAAGGTGG CCTTGCTAGG CCGTACTCAT CCCCCACGGT CCTATCCCCT
            GCGA--GTGG CCTTGCTAGG CCGTGCTCGT CCCCCACGGT CCTAGCCCCT
            GCGA--GTGT TCTTGCTAGG CCGTGCCCGT TCCC-ACTGT CAGGGCCGCC

            ATCCCCTTTC CCC-TCGTGT CACCCACAGT CTACCCCACA CCCATCCATT
            ATCCCCTTTC CCCCTCGTGT CACCCACAGT CTGCCCCACA CCCATCCATT
            ATCCCGTGTT CCC------- --TTTTTCGT GTCACCCACA CCCACCCCTC

            CTTTCTTTAA CCTCTGACTC TTCCTCCTTG GTTTCTCACT GCCTTGGACG
            CTTTCTTCAA CCTCTGACAC TTCCTCCTTG GTTCCTCACT GCCTTGGACG
            CTTTCTCTGA CACTCCCAAG TTCCCT---- GTTCCTCTCT GCCTTGGTCC

            CTTGTTCACC CCGGATGAAC TCCGTAGGCG TCTCCCTTCC CTGCTTGGTA
            CTTGTTCACC CCGGATGAAC TATGTAGGAG TCTCCCTTCC CTGCTAGGTA
            CATATTCACC CCGGATGA-C TGCGGAGTCT CCCACCCTCT GACCTCTGCT

            CCCTAAGG-- --TGCCCTCG GTGCTTGTTC GTAGAGACGA ACTCTGCTCT
            CCCTAAGGCA TCTGCCCTCG GTGCTTGTTC CTAGAGACGA ACTCTGCTCT
            CTCAAAGC-- --------CT GTCCCTAC-- -TAGAGAGGA ACTCTGCTCT
```

5.6 ASN

The Abstract Syntax Notation (`asn`) format is intended to be read by computer rather than humans. It was developed at NCBI. It is included here to demonstrate the broad variety of sequence formats in use. For just the *Mus speciligus* sequence (complete entry) it would be

```
Seq-entry ::= set {
  level 1 ,
  class nuc-prot ,
  descr {
    title "Mus spicilegus adenine phosphoribosyltransferase (APRT) gene, and
 translated products" ,
    update-date
      std {
        year 1995 ,
        month 6 ,
        day 16 } ,
    source {
      org {
        taxname "Mus spicilegus" ,
        common "steppe mouse" ,
        db {
          {
            db "taxon" ,
            tag
              id 10103 } } ,
        orgname {
          name
            binomial {
              genus "Mus" ,
              species "spicilegus" } ,
          lineage "Eukaryotae; mitochondrial eukaryotes; Metazoa; Chordata;
 Vertebrata; Eutheria; Rodentia; Sciurognathi; Myomorpha; Muridae; Murinae;
 Mus" ,
          gcode 1 ,
          mgcode 2 } } } ,
    pub {
      pub {
        gen {
          serial-number 1 } ,
        gen {
          cit "Unpublished" ,
          authors {
```

```
      names
        std {
          {
            name
              name {
                last "Fieldhouse" ,
                initials "D." } } ,
          {
            name
              name {
                last "Golding" ,
                initials "G.B." } } } } ,
     title "Rates of substitution in closely related rodent species" } } } ,
  pub {
   pub {
    gen {
     serial-number 2 } ,
    sub {
     authors {
      names
        std {
          {
            name
              name {
                last "Fieldhouse" ,
                initials "D." } } } } ,
     imp {
      date
        std {
          year 1995 ,
          month 6 ,
          day 7 } ,
      pub
        str "Dan Fieldhouse, Biology, McMaster University, 1280 Main
 Street West, Hamilton, ON, L8S 4K1, Canada" } ,
     medium other } } } } ,
 seq-set {
  seq {
   id {
    genbank {
     name "MSU28720" ,
     accession "U28720" } ,
    gi 881575 } ,
   descr {
    title "Mus spicilegus adenine phosphoribosyltransferase (APRT) gene,
 complete cds." ,
    genbank {
     source "Steppe mouse." ,
     div "ROD" } ,
    create-date
     std {
      year 1995 ,
      month 6 ,
      day 28 } ,
    molinfo {
     biomol genomic } } ,
   inst {
    repr raw ,
    mol dna ,
    length 2117 ,
    seq-data
     ncbi2na 'DA8F86E0FC9B9E317175D7E5D711919A53B58178BE01EBA669935927D56
1F54356A6E7BD2B9AD189698A6FA65B19D355699299B2925DAA37E6A97795A5119AB4775ED7EF5
4A8CDD9577E0215A1D7D627D4D65DFA52D1782D46449A42361C4D9298BA5F9CA5B9DB5546B5C95
7355FD55DBB4544B7954454D4F7F7D05DE11F5D7EBD74797E867EF455A381CECA2DD5F579CAC57
0A4DE576B9FBD722181DE77B5FBB5205297577FCA91027A524D784929EA215E8175238684D7E7C
AAC977A8E07231C00F2B05FAFAA6E9B97A9217425EB27D2A9EFDD54A1C45AA4DFD5FBD5D1109FB
BC041E7B717A7539789F4804572A49E0ED4528BB522A2AEA45522048BA57AC2E74A85121FF971F
47D73EB157A539D480F2A4ECECD7D51849C8E793F80AE90894532BA5789EF48292B28D5429E23A
5152C94D06F7DC9EB2D242172EF5C90BBE17654C7E97F23D5395749D4D5105F575F15C12B721D4
AA7D7BFA57D5C9D289EA6EA7BB9D35A012A82796A551EED25D73DDE8B3A82B0989EEEC8A0A92AD
F3469C52EDCA2C0EEAE7488AF884FAB4AFC45152019D8A72A2BA51FBD93721DDDF11C7D7B7929E
27A035202397C815A9222EB4FBA385D7A5128AC0814150840487D02A5295ED7AB9E1C24089F831
77F77B57D554A053BF9A5EE379E45275A9E0BAE8BBB897AE89E176782A4A88A7285CC5BDF7775D
4B3878A27A723AD11579D5249D4A079FAE9D254A65C2E17FB89C52595FFB8BBC0'H } ,
   annot {
    {
     data
      ftable {
        {
                data
                  gene {
                    locus "APRT" } ,
                location
                  mix {
                    int {
                      from 66 ,
                      to 145 ,
                      id
                        gi 881575 } ,
                    int {
                      from 277 ,
                      to 383 ,
                      id
                        gi 881575 } ,
                    int {
                      from 1354 ,
                      to 1487 ,
                      id
                        gi 881575 } ,
                    int {
                      from 1674 ,
                      to 1752 ,
                      id
                        gi 881575 } ,
                    int {
                      from 1859 ,
                      to 2001 ,
                      id
                        gi 881575 } } } } } } } ,
    seq {
      id {
        gi 881576 } ,
      descr {
        title "adenine phosphoribosyltransferase" ,
        molinfo {
          tech concept-trans } } ,
      inst {
        repr raw ,
        mol aa ,
        length 180 ,
        seq-data
          iupacaa "MSEPELKLVARRIRSFPDFPIPGVLFRDISPLLKDPDSFRASIRLLASHLKSTHSGKID
YIAGLDSRGFLFGPSLAQELGVGCVLIRKQGKLPGPTVSASYSLEYGKAELEIQKDALEPGQRVVIVDDLLATGGTMF
AACDLLHQLRAEVVECVSLVELTSLKGRERLGPIPFFSLLQYD" } ,
      annot {
        {
          data
            ftable {
              {
                data
                  prot {
                    name {
                      "adenine phosphoribosyltransferase" } ,
                    ec {
                      "2.4.2.7" } } ,
                location
                  whole
                    gi 881576 } } } } } } ,
  annot {
    {
      data
        ftable {
          {
            data
              cdregion {
                frame one ,
                code {
                  id 1 } } ,
            comment "purine salvage enzyme" ,
            product
              whole
                gi 881576 ,
            location
              mix {
                int {
                  from 66 ,
                  to 145 ,
                  id
                    gi 881575 } ,
                int {
                  from 277 ,
                  to 383 ,
                  id
                    gi 881575 } ,
                int {
                  from 1354 ,
                  to 1487 ,
                  id
                    gi 881575 } ,
                int {
                  from 1674 ,
                  to 1752 ,
                  id
                    gi 881575 } ,
                int {
                  from 1859 ,
                  to 2001 ,
                  id
                    gi 881575 } } ,
            xref {
              {
                data
                  gene {
                    locus "APRT" } } } } } } } }
```

5.7 BSML format

The Bioinformatic Sequence Markup Language is another format that is rapidly gaining popularity and is designed to be read mostly by computers (viewers are developed to present human readable forms). The BSML format is based on the XML - extended markup language. XML is heralded as the replacement for HTML (hypertext markup language — basically the format used on the internet and read by your favorite internet browser). The primary feature that makes XML an improvement on HTML is that XML is an extenable language. New features and new objects can be defined within XML itself and does not require an entire rewriting of the language (as would HTML to add new features). BSML is a language set up with objects (data structures) predefined that are useful for bioinformatic research. The BSML data specification (DTD) was created to solve the data management problems and yet to include support for complicated structures such as sequence annotations, sequence restriction enzyme digestions, phylogenies and so on.

There is a free viewer available for PC's and more information can be found at www.bsml.org.

5.8 PDB file format

Of quite a different nature than the files listed above, this file is meant to store the three dimensional location of atoms within a molecule. This data structure file is perhaps the most difficult to maintain or to alter because programs must parse these files very precisely in order to produce three dimensional structures of the encoded molecules. There are two formats for the three dimensional structure data stored in the PDB. The first is the rather old flat file format described in detail at http://www.rcsb.org/pdb/docs/format/pdbguide2.2/guide2.2_frame.html and the second is the mmCIF format (the macromolecular Crystallographic Information File) described in detail at http://www.sdsc.edu/pb/cif/papers/methenz.html.

The flat file format (with many rows deleted — indicated by the dots in the center of a row) looks as follows . . .

```
HEADER    OXIDOREDUCTASE                          26-MAY-98   2OCC
TITLE     BOVINE HEART CYTOCHROME C OXIDASE AT THE FULLY OXIDIZED
TITLE    2 STATE
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: CYTOCHROME C OXIDASE;
COMPND   3 CHAIN: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q,
COMPND   4 R, S, T, U, V, W, X, Y, Z;
COMPND   5 SYNONYM: FERROCYTOCHROME C\:OXYGEN OXIDOREDUCTASE;
COMPND   6 EC: 1.9.3.1;
COMPND   7 OTHER_DETAILS: THIS ENZYME IS A HYBRID PROTEIN COMPLEX AND
COMPND   8 IS A HOMODIMER.  ONE MONOMER IS COMPOSED OF 13 DIFFERENT
COMPND   9 SUBUNITS AND SEVEN METAL CENTERS, HEME A, HEME A3, CUA,
COMPND  10 CUB, MG, NA AND ZN.
SOURCE    MOL_ID: 1;
SOURCE   2 ORGANISM_SCIENTIFIC: BOS TAURUS;
SOURCE   3 ORGANISM_COMMON: BOVINE;
SOURCE   4 ORGAN: HEART;
SOURCE   5 TISSUE: HEART MUSCLE;
SOURCE   6 ORGANELLE: MITOCHONDRION
KEYWDS    OXIDOREDUCTASE, CYTOCHROME(C)-OXYGEN, CYTOCHROME C
KEYWDS   2 OXIDASE
EXPDTA    X-RAY DIFFRACTION
AUTHOR    T.TSUKIHARA,M.YAO
REVDAT   1   13-JAN-99 2OCC    0
REMARK   1
REMARK   1 REFERENCE 1
REMARK   1  AUTH   S.YOSHIKAWA,K.SHINZAWA-ITOH,R.NAKASHIMA,R.YAONO,
REMARK   1  AUTH 2 E.YAMASHITA,N.INOUE,M.YAO,M.J.FEI,C.P.LIBEU,
REMARK   1  AUTH 3 T.MIZUSHIMA,H.YAMAGUCHI,T.TOMIZAKI,T.TSUKIHARA
REMARK   1  TITL   REDOX-COUPLED CRYSTAL STRUCTURAL CHANGES IN BOVINE
REMARK   1  TITL 2 HEART CYTOCHROME C OXIDASE
REMARK   1  REF    SCIENCE                        V. 280  1723 1998
REMARK   1  REFN   ASTM SCIEAS  US ISSN 0036-8075                 0038
REMARK   1 REFERENCE 2

                        ........................

REMARK   2
REMARK   2 RESOLUTION. 2.3  ANGSTROMS.
REMARK   3
REMARK   3 REFINEMENT.
REMARK   3   PROGRAM     : X-PLOR 3.84
REMARK   3   AUTHORS     : BRUNGER
REMARK   3
REMARK   3  DATA USED IN REFINEMENT.
REMARK   3   RESOLUTION RANGE HIGH (ANGSTROMS) : 2.3
REMARK   3   RESOLUTION RANGE LOW  (ANGSTROMS) : 15.
REMARK   3   DATA CUTOFF            (SIGMA(F)) : 2.0
REMARK   3   DATA CUTOFF HIGH         (ABS(F)) : 100000.0
REMARK   3   DATA CUTOFF LOW          (ABS(F)) : 0.1
REMARK   3   COMPLETENESS (WORKING+TEST)   (%) : 88.88
REMARK   3   NUMBER OF REFLECTIONS             : 278049
REMARK   3
REMARK   3  FIT TO DATA USED IN REFINEMENT.
REMARK   3   CROSS-VALIDATION METHOD          : THROUGHOUT
REMARK   3   FREE R VALUE TEST SET SELECTION  : RANDOM
REMARK   3   R VALUE            (WORKING SET) : 0.209

                        ........................

REMARK   4
REMARK   4 2OCC COMPLIES WITH FORMAT V. 2.2, 16-DEC-1996
REMARK   6
REMARK   6 THIS ENZYME IS A MULTI-COMPONENT PROTEIN COMPLEX AND IS A
REMARK   6 HOMODIMER. EACH MONOMER IS COMPOSED OF 13 DIFFERENT
REMARK   6 SUBUNITS AND SIX METAL CENTERS: HEME A, HEME A3, CUA, CUB,
REMARK   6 MG, NA, AND ZN. THE SIDE CHAINS OF H 240 AND Y244 OF
REMARK   6 MOLECULES A AND N ARE LINKED TOGETHER BY A COVALENT BOND.
REMARK   6 THE ELECTRON DENSITY OF REGION FROM D(Q) 1 TO D(Q) 3,
REMARK   6 E(R) 1 TO E(R) 4, H(U) 1 TO H(U) 6, J(W) 59, K(X) 1 TO
REMARK   6 K(X) 5, K(X) 53 TO K(X) 54 AND M(Z) 41 TO M(Z) 43 IS
REMARK   6 NOISY AND THE MODEL OF THIS REGION HAS AMBIGUITY.
REMARK 200
REMARK 200 EXPERIMENTAL DETAILS
REMARK 200  EXPERIMENT TYPE                : X-RAY DIFFRACTION
REMARK 200  DATE OF DATA COLLECTION        : MAY-1996
REMARK 200  TEMPERATURE           (KELVIN) : 283
REMARK 200  PH                             : 6.8
REMARK 200  NUMBER OF CRYSTALS USED        : 32
REMARK 200
REMARK 200  SYNCHROTRON              (Y/N) : Y
REMARK 200  RADIATION SOURCE               : PHOTON FACTORY
REMARK 200  BEAMLINE                       : 6A, 6B
REMARK 200  X-RAY GENERATOR MODEL          : NULL
REMARK 200  MONOCHROMATIC OR LAUE    (M/L) : M

                        ........................

DBREF  2OCC A    1   514  SWS    P00396   COX1_BOVIN       1    514
DBREF  2OCC B    1   227  SWS    P00404   COX2_BOVIN       1    227
DBREF  2OCC C    1   261  SWS    P00415   COX3_BOVIN       1    261
DBREF  2OCC D    4   147  SWS    P00423   COX4_BOVIN      26    169
DBREF  2OCC E    5   109  SWS    P00426   COXA_BOVIN       5    109
DBREF  2OCC F    1    98  SWS    P11949   COXB_BOVIN       1     98
```

```
                        .......................

SEQRES   1 A  514  MET PHE ILE ASN ARG TRP LEU PHE SER THR ASN HIS LYS
SEQRES   2 A  514  ASP ILE GLY THR LEU TYR LEU LEU PHE GLY ALA TRP ALA
SEQRES   3 A  514  GLY MET VAL GLY THR ALA LEU SER LEU LEU ILE ARG ALA
SEQRES   4 A  514  GLU LEU GLY GLN PRO GLY THR LEU LEU GLY ASP ASP GLN
SEQRES   5 A  514  ILE TYR ASN VAL VAL VAL THR ALA HIS ALA PHE VAL MET
SEQRES   6 A  514  ILE PHE PHE MET VAL MET PRO ILE MET ILE GLY GLY PHE
SEQRES   7 A  514  GLY ASN TRP LEU VAL PRO LEU MET ILE GLY ALA PRO ASP
SEQRES   8 A  514  MET ALA PHE PRO ARG MET ASN ASN MET SER PHE TRP LEU

                        .......................

SEQRES   1 B  227  MET ALA TYR PRO MET GLN LEU GLY PHE GLN ASP ALA THR
SEQRES   2 B  227  SER PRO ILE MET GLU GLU LEU LEU HIS PHE HIS ASP HIS
SEQRES   3 B  227  THR LEU MET ILE VAL PHE LEU ILE SER SER LEU VAL LEU
SEQRES   4 B  227  TYR ILE ILE SER LEU MET LEU THR THR LYS LEU THR HIS
SEQRES   5 B  227  THR SER THR MET ASP ALA GLN GLU VAL GLU THR ILE TRP
SEQRES   6 B  227  THR ILE LEU PRO ALA ILE ILE LEU ILE LEU ILE ALA LEU
SEQRES   7 B  227  PRO SER LEU ARG ILE LEU TYR MET MET ASP GLU ILE ASN

                        .......................

HET    HEA  A 515      60     PROTOPORPHYRIN IX CONTAINS FE(II)
HET    HEA  A 516      60     PROTOPORPHYRIN IX CONTAINS FE(II)
HET     CU  A 517       1
HET     MG  A 518       1
HET     NA  A 519       1
HET    PER  A 520       2
HET     CU  B 228       1
HET     CU  B 229       1
HET     ZN  F  99       1
HET    HEA  N 515      60     PROTOPORPHYRIN IX CONTAINS FE(II)
HET    HEA  N 516      60     PROTOPORPHYRIN IX CONTAINS FE(II)
HET     CU  N 517       1
HET     MG  N 518       1
HET     NA  N 519       1
HET    PER  N 520       2
HET     CU  O 228       1
HET     CU  O 229       1
HET     ZN  S  99       1
HETNAM     HEA HEME-A
HETNAM      CU COPPER (II) ION
HETNAM      MG MAGNESIUM ION
HETNAM      NA SODIUM ION
HETNAM     PER PEROXIDE ION
HETNAM      ZN ZINC ION
FORMUL  27  HEA    4(C49 H62 N4 O6 FE1)
FORMUL  28   CU    6(CU1 2+)
FORMUL  29   MG    2(MG1 2+)
FORMUL  30   NA    2(NA1 1+)
FORMUL  31  PER    2(O2 2-)
FORMUL  32   ZN    2(ZN1 2+)
HELIX    1   1 PHE A    2  TRP A    6  1                                   5
HELIX    2   2 HIS A   12  LEU A   41  1                                  30
HELIX    3   3 ASP A   51  ILE A   87  1                                  37
HELIX    4   4 PRO A   95  MET A  117  1                                  23
HELIX    5   5 ALA A  141  ASN A  170  1                                  30
HELIX    6   6 LEU A  183  ASN A  214  1                                  32

                        .......................

SHEET    1   A 5 LEU B 116  SER B 120  0
SHEET    2   A 5 TYR B 105  TYR B 110 -1  N  TYR B 110   O  LEU B 116
SHEET    3   A 5 LEU B  95  HIS B 102 -1  N  HIS B 102   O  TYR B 105
SHEET    4   A 5 ILE B 150  SER B 156  1  N  ARG B 151   O  LEU B  95
SHEET    5   A 5 ASN B 180  LEU B 184 -1  N  LEU B 184   O  ILE B 150
SHEET    1   B 3 VAL B 142  PRO B 145  0
SHEET    2   B 3 ILE B 209  VAL B 214  1  N  GLU B 212   O  VAL B 142
SHEET    3   B 3 GLY B 190  GLY B 194 -1  N  GLY B 194   O  ILE B 209
SHEET    1   C 2 HIS B 161  VAL B 165  0
SHEET    2   C 2 LEU B 170  ALA B 174 -1  N  ALA B 174   O  HIS B 161
SHEET    1   D 3 ASN F  47  SER F  51  0
SHEET    2   D 3 GLY F  86  PRO F  93  1  N  LYS F  90   O  ASN F  47
SHEET    3   D 3 GLN F  80  CYS F  82 -1  N  CYS F  82   O  GLY F  86
SHEET    1   E 2 LYS F  55  CYS F  60  0
SHEET    2   E 2 ILE F  70  HIS F  75 -1  N  LEU F  74   O  ARG F  56
SHEET    1   F 5 LEU O 116  SER O 120  0
SHEET    2   F 5 TYR O 105  TYR O 110 -1  N  TYR O 110   O  LEU O 116
SHEET    3   F 5 LEU O  95  HIS O 102 -1  N  HIS O 102   O  TYR O 105
SHEET    4   F 5 ILE O 150  SER O 156  1  N  ARG O 151   O  LEU O  95
SHEET    5   F 5 ASN O 180  LEU O 184 -1  N  LEU O 184   O  ILE O 150
SHEET    1   G 3 VAL O 142  PRO O 145  0
SHEET    2   G 3 ILE O 209  VAL O 214  1  N  GLU O 212   O  VAL O 142
SHEET    3   G 3 GLY O 190  GLY O 194 -1  N  GLY O 194   O  ILE O 209
SHEET    1   H 2 HIS O 161  VAL O 165  0
SHEET    2   H 2 LEU O 170  ALA O 174 -1  N  ALA O 174   O  HIS O 161
SHEET    1   I 3 ASN S  47  SER S  51  0
SHEET    2   I 3 GLY S  86  PRO S  93  1  N  LYS S  90   O  ASN S  47
SHEET    3   I 3 GLN S  80  CYS S  82 -1  N  CYS S  82   O  GLY S  86
SHEET    1   J 2 LYS S  55  CYS S  60  0
SHEET    2   J 2 ILE S  70  HIS S  75 -1  N  LEU S  74   O  ARG S  56
SSBOND   1 CYS H   29    CYS H   64
SSBOND   2 CYS H   39    CYS H   53
SSBOND   3 CYS U   29    CYS U   64
SSBOND   4 CYS U   39    CYS U   53
LINK        FE   HEA A 515                 NE2 HIS A  61
LINK        FE   HEA A 515                 NE2 HIS A 378
LINK        FE   HEA A 516                 NE2 HIS A 376
LINK        FE   HEA A 516                 O1  PER A 520
LINK        CU    CU A 517                 ND1 HIS A 240
LINK        CU    CU A 517                 NE2 HIS A 290

                        .......................

ATOM      1  N   MET A   1      55.242 340.693 224.088  1.00 68.90           N
ATOM      2  CA  MET A   1      54.908 339.282 224.487  1.00 71.09           C
ATOM      3  C   MET A   1      54.673 338.307 223.329  1.00 66.66           C
ATOM      4  O   MET A   1      55.350 337.285 223.238  1.00 67.66           O
ATOM      5  CB  MET A   1      53.723 339.248 225.450  1.00 79.30           C
ATOM      6  CG  MET A   1      54.110 339.452 226.915  1.00 87.90           C
ATOM      7  SD  MET A   1      55.300 338.229 227.515  1.00 97.07           S
ATOM      8  CE  MET A   1      54.166 336.799 228.014  1.00 96.59           C
ATOM      9  N   PHE A   2      53.673 338.579 222.494  1.00 61.89           N
ATOM     10  CA  PHE A   2      53.412 337.739 221.322  1.00 56.50           C
ATOM     11  C   PHE A   2      54.569 337.917 220.303  1.00 53.31           C
ATOM     12  O   PHE A   2      55.076 336.947 219.739  1.00 53.84           O
ATOM     13  CB  PHE A   2      52.077 338.127 220.683  1.00 55.21           C
ATOM     14  CG  PHE A   2      51.737 337.334 219.459  1.00 54.54           C
ATOM     15  CD1 PHE A   2      51.050 336.138 219.565  1.00 55.24           C
ATOM     16  CD2 PHE A   2      52.126 337.775 218.200  1.00 55.62           C
ATOM     17  CE1 PHE A   2      50.756 335.388 218.432  1.00 58.99           C
ATOM     18  CE2 PHE A   2      51.839 337.035 217.059  1.00 57.84           C
ATOM     19  CZ  PHE A   2      51.155 335.840 217.171  1.00 58.36           C
ATOM     20  N   ILE A   3      55.010 339.158 220.116  1.00 47.37           N

                        .......................

HETATM 4147 CU    CU A 517      67.173 310.978 190.358  1.00 16.27          CU
HETATM 4148 MG    MG A 518      62.605 315.176 179.115  1.00 19.26          MG
HETATM 4149 NA    NA A 519      42.250 318.661 179.405  1.00 26.18          NA
HETATM 4150  O1  PER A 520      64.953 309.772 191.618  1.00 10.28           O
HETATM 4151  O2  PER A 520      65.314 311.367 191.209  1.00 15.28           O
ATOM   4152  N   MET B   1      50.114 302.768 167.666  1.00 37.69           N
ATOM   4153  CA  MET B   1      49.455 303.851 168.484  1.00 36.15           C
ATOM   4154  C   MET B   1      48.691 303.239 169.660  1.00 34.30           C
ATOM   4155  O   MET B   1      48.549 302.024 169.753  1.00 34.54           O
ATOM   4156  CB  MET B   1      48.490 304.694 167.641  1.00 35.38           C

                        .......................
                        .......................

ATOM  28892  O   SER Z  43     155.003 299.215 171.486  1.00 99.03           O
ATOM  28893  CB  SER Z  43     152.512 300.170 170.193  1.00 99.03           C
ATOM  28894  OG  SER Z  43     151.462 300.982 169.639  1.00 99.02           O
ATOM  28895  OXT SER Z  43     154.021 299.630 173.431  1.00 99.03           O
TER   28896      SER Z  43
CONECT  351  350 4149
CONECT  474  472  473 4027
CONECT 1836 1835 1838 4147
CONECT 2239 2237 2238 4147
CONECT 2249 2247 2248 4147

                        .......................

CONECT264472635326446
CONECT265522625626551
MASTER      370    0   18   98   30    0    2    928870   26  308  292
END
```

The file begins with the keywords, HEADER, TITLE, COMPND, SOURCE, KEYWDS, EXPDTA, that describe the nature of the molecule to which this file pertains. The keywords AUTHOR and REVDAT give the authors responsible for this file and its revision history. The REMARK keyword indicates descriptive entries about the molecular structure (they are numbered according to their category). These remarks provide enormous detail regarding the structure. DBREF supplies cross references to entries of this molecule in other databases. SEQRES is the beginning of the actual sequence information of the molecule. Note the molecule can consist of multiple chains; in this case labelled chain A – chain Z. The HET, HETNAM and FORMUL fields contain information about atoms/molecules that are associated with the molecule in question (for HET, the fields here are a letter code for each “HET” atom(s), the letter identifying the chain, insertion code, number of records with a HET entry, and some descriptive text), their chemical name (in this case a HEME group, copper ion, ...) and their chemical formula. The HELIX, SHEET, and TURN (not shown above) give information about the secondrary struture of the molecule. Information about connections in the molecule are shown by SSBOND, LINK, HYDBND, SLTBRG, and CISPEP (the last three not shown in the above structure).

And much more information is provided by other fields too numerous to list here. The business end is in the ATOM field. This contains a numbered list of atoms (in this case 28,895 of them), the atom name, the (amino acid) residue name, the chain identifier number, the residue sequence number, then three numbers that describe the x, y, z coordinates of this atom in Angstrom units, an occupancy number, a temperature factor and finally an element symbol.

The TER field indicates the end a section. The CONECT section provides further information on chemical connectivity. The MASTER and END fields are used to describe the number of records of different types and to signal the end of the file.

Chapter 6

Database Searching

6.1 Are there homologues in the database?

The following are some of the common programs currently being used to search the databases to find sequences similar to a specific query sequence provided by the user. In addition to finding out the identity of an unknown sequence they are also useful to find homologues and ancestral sequences that have similar or related functions/sequences.

6.1.1 FASTA

To search through the whole genetic sequence database can take a great deal of time due to its enormous size. If some operation must be performed on each sequence in turn then this can take even longer. One such example is to look throughout the whole database for homologous or similar sequences. To do this, special programs have been developed to speed the search. The first amongst these programs was a program called FASTA written by W.R. Pearson and D.J. Lipman (1988, PNAS 85:2444-2448).

It is possible to run this program on remote machines. The obvious choice for such a remote machine would be one that has access to the latest sequence information. Both EMBL and DDBJ have permitted this type of access and have implemented FASTA type searches through their machines (NCBI prefers to use BLAST - see below).

There are several flavours to FASTA: `fasta` scans a protein or DNA sequence library for sequences similar to a query sequence. `tfasta` compares a protein query sequence to a translated DNA sequence library. `lfasta` compares two query sequences for local similarity between them and shows the local sequence alignments. `plfasta` compares two sequences for local similarity and plots the local sequence alignments. Two recent flavours `fastx` and `fasty` (Pearson *et al*. 1997 Genomics 46:24-36) permit comparison of a DNA sequence translated in all six frames to the protein databases. The 'x' form takes a DNA query sequence and translates it in all frames and then permits gaps between the resulting amino acids. The 'y' form more generally permits gaps within and between codons. The related `tfastx` and `tfasty` forms compare a protein query sequence to a DNA database by translating the DNA database in all six frames.

I will illustrate what a FASTA type of search is and what the results look like with an example. Basically the idea is to search through the complete database for any possible similar sequence.

Instructions

To carry out this type of search on the EMBL server the following must be done. Either point your web browser to FASTA3 and fill out the appropriate forms or set up a file containing the following

```
LIB UNIPROT
WORD 1
LIST 50
TITLE HALHA
```

```
HISTOGRAM yes
SEQ
PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWP
FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHL
DYLQNRVI
```

The first line contains the data library files to be searched (in this case all known protein entries). For protein searches this field may be one of

```
UniProt               A non-redundant collection of all proteins
UniRef100             As for UniProt but eliminate identical proteins
UniRef90              As for UniProt but eliminate proteins > 90% identical
UniRef50              As for UniProt but eliminate proteins > 50% identical
UniParc               As for UniProt but include archived proteins
                      (shows changes to an entry).
swiss-prot            Proteins in the SwissProt database
ipi                   Proteins in the International Protein Index
prints                Proteins in the FingerPrints database
sgt                   Proteins in the Structural Genomics Targets database
pdb                   Proteins in the 3D structural database PDB at Rutgers
imgthlap              Proteins in the Immunogenetics Database
Euro Patents          Proteins in the European patents database
Japan Patents         Proteins in the Japanese patents database
USPTO Patents         Proteins in the American patents database
```

For nucleotide searches this field may be one of

```
EMBL                  The entire EMBL database
FUNGI                 Subsection of EMBL.
INVERTEBRATES
HUMAN
MAMMALS
ORGANELLES
BACTERIOPHAGE
PLANT
PROKARYOTES
RODENTS
MOUSE
STSs                  Sequence Tagged Sites
SYNTHETIC
UNCLASSIFIED
VIRUSES
VERTEBRATES
ESTs                  Expressed Sequence Tags
GSSs                  Genome Survey Sequences
HTGs                  High throughput Genomics sequences
PATENTS
VECTORS
EMBLNEW               Sequences new since the last major database release
EMBLALL               EMBL + EMBLNEW
IMGTLIGM              Immunoglobulins and T cell receptors database
IMGTHLA               Human Major Histocompatiblity Complex (MHC/HLA) database
HGVBASE               Human Genome Variation database
```

The second line gives the word size or k-tuple value (more on this below). The third line says to LIST on the output the top 50 scores. The TITLE line is used for the subject of the mail message. Finally SEQ implies that everything below this line to the end of the message is part of the sequence. In this case the sequence is the protein sequence of the ferredoxin gene of *Halobacterium species NRC-1*.

The remaining options are - LIST n, n top scores listed in the output [50]. ALIGN n, align the top n to the query sequence [10]. ONE, compare only the given strand to the database, the default is to use the complementary strand as well. PROT will force your query sequence to be a protein (small protein sequences may be otherwise misinterpreted as DNA). PATH string mails the results back to string rather than the originator of the message.

After creating this file, mail the file by electronic mail to fasta@ebi.ac.uk and the results will be sent back to you by electronic mail. Alternatively simply point your web browser to FASTA3 and fill in the forms (they have the same options). Please, as a courtesy to others using the system please send only one job at a time. Many other people from all over the world are using these servers and the FASTA program is quite computer intensive despite its speed.

FASTA output

An example of the output is shown below. The input file is specifying the *Halobacterium species NRC-1* ferredoxin amino acid sequence to search the SWISS-PROT database.

```
FASTA searches a protein or DNA sequence data bank
 version 3.4t23 March 18, 2004
Please cite:
 W.R. Pearson & D.J. Lipman PNAS (1988) 85:2444-2448

Query library @ vs +uniprot library
searching /ebi/services/idata/v916/fastadb/uniprot library

  1>>>HALHA - 128 aa
 vs  UniProt library

       opt      E()
< 20   1429      0:=
  22     30      0:=          one = represents 2608 library sequences
  24     50      1:*
  26    246     33:*
  28    917    356:*
  30   3937   2160:*=
  32  13990   8352:===*==
  34  33145  22650:========*====
  36  63533  46517:=================*=======
  38  94343  76875:=============================*=======
  40 124411 107234:=========================================*======
  42 156468 131081:==================================================*=========
  44 150019 144594:=======================================================*==
  46 141925 147273:======================================================= *
  48 131289 140997:===================================================   *
  50 124662 128660:================================================ *
  52 106819 113114:=========================================  *
  54  83834  96619:=================================    *
  56  69228  80707:===========================   *
  58  58053  66259:=======================  *
  60  46296  53673:==================  *
  62  36284  43030:==============  *
  64  29298  34222:============ *
  66  22040  27048:========= *
  68  17474  21275:======= *
  70  13385  16672:======*
  72  10307  13028:====*
  74   7966  10157:===*
  76   6188   7906:===*
  78   4631   6145:==*
  80   3596   4772:=*
  82   2939   3650:=*
  84   2320   2891:=*
  86   1609   2237:*
  88   1387   1731:*          inset = represents 13 library sequences
  90    920   1339:*
  92    642   1036:*         :=======================================*
  94    524    802:*         :=======================================*
  96    393    620:*         :===============================        *
  98    284    480:*         :======================              *
 100    257    371:*         :====================        *
 102    209    287:*         :=================     *
 104    124    222:*         :==========       *
 106     97    172:*         :========     *
 108     91    133:*         :=======   *
 110     83    103:*         :=======*
 112     46     80:*         :====  *
 114     39     62:*         :=== *
 116     38     48:*         :===*
 118     25     37:*         :==*
>120    604     29:*         :==*=====================================
501934690 residues in 1568424 sequences
 statistics extrapolated from 60000 to 1567839 sequences
  Expectation_n fit: rho(ln(x))= 5.0598+/-0.000193; mu= 9.7296+/- 0.011
 mean_var=58.3610+/-12.418, 0's: 151 Z-trim: 257  B-trim: 877 in 1/64
 Lambda= 0.167885
 Kolmogorov-Smirnov  statistic: 0.0646 (N=29) at  44

FASTA (3.47 Mar 2004) function [optimized, BL50 matrix (15:-5)] ktup: 1
 join: 42, opt: 30, open/ext: -10/-2, width:  32
 Scan time: 272.383
The best scores are:                                      opt bits E(1568424)
```

```
UNIPROT:FER_HALN1 P00216 Ferredoxin.               ( 128)  870 217.3   1e-55
UNIPROT:Q9YGB6 Q9YGB6 Ferredoxin.                  ( 129)  761 190.9 9.1e-48
UNIPROT:FER_HALMA P00217 Ferredoxin.               ( 128)  750 188.2 5.8e-47
UNIPROT:FER_SYNP4 P15788 Ferredoxin.               (  98)  271  72.1 3.9e-12
UNIPROT:FER_SYNEL P00256 Ferredoxin I.             (  97)  263  70.2 1.5e-11
UNIPROT:FER_SYNLI P00255 Ferredoxin.               (  96)  262  69.9 1.7e-11
UNIPROT:FER_PHYPA O04166 Ferredoxin, chloroplast   ( 145)  254  68.1 9.3e-11
UNIPROT:FER1_ANASP P06543 Ferredoxin I.            (  98)  252  67.5 9.4e-11
UNIPROT:FER1_ANAVA P00254 Ferredoxin I.            (  98)  251  67.3 1.1e-10
UNIPROT:FER_NOSMU P00253 Ferredoxin.               (  98)  247  66.3 2.2e-10
UNIPROT:FER1_PLEBO Q51577 Ferredoxin I (FdI).      (  99)  245  65.8 3.1e-10
UNIPROT:FER3_CYACA P00241 Ferredoxin.              (  98)  242  65.1   5e-10
UNIPROT:Q7V0B6 Q7V0B6 Ferredoxin.                  (  99)  242  65.1 5.1e-10
UNIPROT:Q7VAM6 Q7VAM6 Ferredoxin.                  (  99)  241  64.8   6e-10
UNIPROT:Q7U8S7 Q7U8S7 Ferredoxin.                  (  99)  241  64.8   6e-10
UNIPROT:FER2_NOSMU P00249 Ferredoxin II.           (  98)  238  64.1 9.8e-10
UNIPROT:FER_CHLFR P00247 Ferredoxin.               (  98)  238  64.1 9.8e-10
UNIPROT:Q7M191 Q7M191 Ferredoxin.                  (  98)  238  64.1 9.8e-10
UNIPROT:FER1_CYAPA P17007 Ferredoxin I.            (  98)  237  63.9 1.2e-09
UNIPROT:FER1_NOSMU P00252 Ferredoxin I.            (  98)  236  63.6 1.4e-09
UNIPROT:FER_SYNY4 P00243 Ferredoxin.               (  96)  235  63.4 1.6e-09
UNIPROT:FER_EUGVI P22341 Ferredoxin.               (  96)  234  63.1 1.9e-09
UNIPROT:FER_SYNY3 P27320 Ferredoxin I.             (  96)  233  62.9 2.2e-09
UNIPROT:Q7TUS8 Q7TUS8 2Fe-2S Ferredoxin:Ferredoxi  (  99)  233  62.9 2.3e-09
UNIPROT:FER_MASLA P00248 Ferredoxin.               (  98)  232  62.7 2.7e-09
UNIPROT:FER1_SYNP7 P06517 Ferredoxin I.            (  98)  232  62.7 2.7e-09
UNIPROT:FER2_SPIOL P00224 Ferredoxin II.           (  97)  231  62.4 3.2e-09
UNIPROT:FER_CHLFU P56408 Ferredoxin.               (  94)  230  62.2 3.6e-09
UNIPROT:Q7M1S3 Q7M1S3 Ferredoxin C.                (  96)  230  62.2 3.7e-09
UNIPROT:Q6B8Y2 Q6B8Y2 Ferredoxin.                  (  98)  230  62.2 3.8e-09
UNIPROT:FER1_EQUTE P00234 Ferredoxin I.            (  95)  229  61.9 4.3e-09
UNIPROT:FER_RHOPL P07484 Ferredoxin.               (  97)  229  61.9 4.4e-09
UNIPROT:FER_GUITH O78510 Ferredoxin.               (  96)  228  61.7 5.2e-09
UNIPROT:FER_PORPU P51320 Ferredoxin.               (  98)  228  61.7 5.3e-09
UNIPROT:FER_GLEJA P00233 Ferredoxin.               (  95)  227  61.4 6.1e-09
UNIPROT:FER_MARPO P09735 Ferredoxin.               (  95)  227  61.4 6.1e-09
UNIPROT:FER1_EQUAR P00235 Ferredoxin I.            (  95)  227  61.4 6.1e-09
UNIPROT:FER_PORUM P00242 Ferredoxin.               (  98)  227  61.5 6.2e-09
UNIPROT:FER1_RAPSA P14936 Ferredoxin, root R-B1.   (  98)  226  61.2 7.4e-09
UNIPROT:O30582 O30582 Plant-type.                  (  99)  226  61.2 7.4e-09
UNIPROT:Q7XVG7 Q7XVG7 OSJNBa0073L04.7 protein.     ( 152)  227  61.5 9.1e-09
UNIPROT:FER_ODOSI P49522 Ferredoxin.               (  98)  224  60.7   1e-08
UNIPROT:FER2_RAPSA P14937 Ferredoxin, root R-B2.   (  98)  224  60.7   1e-08
UNIPROT:FER_SPIPL P00246 Ferredoxin.               (  98)  224  60.7   1e-08
UNIPROT:Q9KJL1 Q9KJL1 FdxH.                        ( 104)  224  60.7 1.1e-08
UNIPROT:Q85FT5 Q85FT5 Ferredoxin.                  (  97)  222  60.2 1.4e-08
UNIPROT:FER_BRYMA P07838 Ferredoxin.               (  98)  222  60.2 1.4e-08
UNIPROT:FER_SPIMA P00245 Ferredoxin.               (  98)  222  60.2 1.4e-08
UNIPROT:FER6_MAIZE P94044 Ferredoxin VI, chloropl  ( 155)  224  60.8 1.5e-08
UNIPROT:FER_HORVU P83522 Ferredoxin.               (  97)  221  60.0 1.7e-08

>>UNIPROT:FER_HALN1 P00216 Ferredoxin.                     (128 aa)
 initn: 870 init1: 870 opt: 870  Z-score: 1144.0  bits: 217.3 E(): 1e-55
Smith-Waterman score: 870;  100.000% identity (100.000% ungapped) in 128 aa overlap (1-128:1-128)

               10        20        30        40        50        60
HALHA  PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWP
       ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
UNIPRO PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWP
               10        20        30        40        50        60

               70        80        90       100       110       120
HALHA  FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHL
       ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
UNIPRO FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHL
               70        80        90       100       110       120


HALHA  DYLQNRVI
       ::::::::
UNIPRO DYLQNRVI


>>UNIPROT:Q9YGB6 Q9YGB6 Ferredoxin.                        (129 aa)
 initn: 761 init1: 761 opt: 761  Z-score: 1001.2  bits: 190.9 E(): 9.1e-48
Smith-Waterman score: 761;  85.156% identity (85.156% ungapped) in 128 aa overlap (1-128:2-129)

                10        20        30        40        50
HALHA   PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDW
        ::::::::..::.:::: :::.: .:.:  :::::::..:: ::::::::::::::::
```

```
UNIPRO MPTVEYLNYEVVDDNGWDMYDDDVFAEASDMDLDGEDYGSLEVNEGEYILEAAEAQGYDW
               10        20        30        40        50        60

      60        70        80        90       100       110
HALHA  PFSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKH
       :::::::::::::.:: ::.:::::::::::::::.:.::::::::: ::::::::::::
UNIPRO PFSCRAGACANCAAIVLEGDIDMDMQQILSDEEVEDKNVRLTCIGSPDADEVKIVYNAKH
               70        80        90       100       110       120

     120
HALHA  LDYLQNRVI
       :::::::::
UNIPRO LDYLQNRVI


>>UNIPROT:FER_HALMA P00217 Ferredoxin.                    (128 aa)
 initn: 750 init1: 750 opt: 750  Z-score: 986.9  bits: 188.2 E(): 5.8e-47
Smith-Waterman score: 750;  84.375% identity (84.375% ungapped) in 128 aa overlap (1-128:1-128)

               10        20        30        40        50        60
HALHA  PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWP
       :::::::::..::.:::: :::.: .:.:  :: ::::..:: :::::::::::::::::
UNIPRO PTVEYLNYEVVDDNGWDMYDDDVFGEASDMDLDDEDYGSLEVNEGEYILEAAEAQGYDWP
               10        20        30        40        50        60

               70        80        90       100       110       120
HALHA  FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHL
       ::::::::::::.:: ::.:::::::::::::::.:.::::::::: :::::::::::::
UNIPRO FSCRAGACANCAAIVLEGDIDMDMQQILSDEEVEDKNVRLTCIGSPDADEVKIVYNAKHL
               70        80        90       100       110       120


HALHA  DYLQNRVI
       ::::::::
UNIPRO DYLQNRVI


>>UNIPROT:FER_SYNP4 P15788 Ferredoxin.                    (98 aa)
 initn: 228 init1: 228 opt: 271  Z-score: 361.6  bits: 72.1 E(): 3.9e-12
Smith-Waterman score: 271;  46.739% identity (48.864% ungapped) in 92 aa overlap (32-120:7-97)

              10        20        30           40        50
HALHA  TVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYG---TMEVAEGEYILEAAEAQGYD
                                     : .:..:   :.:: . ::::..:: .: :
UNIPRO                         ASYKVTLINEEMGLNETIEVPDDEYILDVAEEEGID
                                       10        20        30

       60        70        80        90       100       110
HALHA  WPFSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAK
        :.:::::::..::. .::::::.. :..:.:...:   : :::.. ::.: . :... .
UNIPRO LPYSCRAGACSTCAGKIKEGEIDQSDQSFLDDDQIEAGYV-LTCVAYPASDCTIITHQEE
          40        50        60        70         80        90

      120
HALHA  HLDYLQNRVI
       .:
UNIPRO ELY

.......................................................................
.......................  Material Deleted  ...........................
.......................................................................


>>UNIPROT:FER_SPIMA P00245 Ferredoxin.                    (98 aa)
 initn: 191 init1: 191 opt: 222  Z-score: 297.5  bits: 60.2 E(): 1.4e-08
Smith-Waterman score: 222;  45.070% identity (45.714% ungapped) in 71 aa overlap (39-109:17-86)

       10        20        30        40        50        60
HALHA  ETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWPFSCRAGAC
                                     :..  .  :::.:::  : : :.:::::::
UNIPRO                ATYKVTLISEAEGINETIDCDDDTYILDAAEEAGLDLPYSCRAGAC
                              10        20        30        40

       70        80        90       100       110       120
HALHA  ANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHLDYLQNRVI
       ..::. .  : ::.. :..:.:...:   : :::.. :..:
UNIPRO STCAGKITSGSIDQSDQSFLDDDQIEAGYV-LTCVAYPTSDCTIQTHQEEGLY
          50        60        70         80        90

>>UNIPROT:FER6_MAIZE P94044 Ferredoxin VI, chloroplast p  (155 aa)
```

```
 initn: 212 init1: 176 opt: 224  Z-score: 297.1  bits: 60.8 E(): 1.5e-08
Smith-Waterman score: 224;  39.394% identity (41.053% ungapped) in 99 aa overlap (23-121:59-153)

                       10        20        30        40        50
HALHA          PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAA
                                     ...:.  .: :: ..  .:. .  ::::::
UNIPRO NTLSFAGHARQAARASGPRLSSRFVASAAAVLHKVKLVGPDGTEH-EFEAPDDTYILEAA
       30        40        50        60        70         80

             60        70        80        90       100       110
HALHA  EAQGYDWPFSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVK
       :. : . :::::::.:..::. .. ::.:..  ..:.: .. :  . ::::. : :: :
UNIPRO ETAGVELPFSCRAGSCSTCAGRMSAGEVDQSEGSFLDDGQMAEGYL-LTCISYPKADCV-
        90       100       110       120       130        140

            120
HALHA  IVYNAKHLDYLQNRVI
        ... :. :
UNIPRO -IHTHKEEDLY
          150

>>UNIPROT:FER_HORVU P83522 Ferredoxin.                    (97 aa)
 initn: 195 init1: 195 opt: 221  Z-score: 296.3  bits: 60.0 E(): 1.7e-08
Smith-Waterman score: 221;  47.222% identity (47.887% ungapped) in 72 aa overlap (40-111:16-86)

      10        20        30        40        50        60
HALHA  TLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWPFSCRAGACA
                                     .:: .  :::. :: .: : :.:::::.:.
UNIPRO                ATYKVKLVTPEGEVELEVPDDVYILDQAEEEGIDLPYSCRAGSCS
                              10        20        30        40

      70        80        90       100       110       120
HALHA  NCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHLDYLQNRVI
       .::. .  ::::.. :..:.:...::  : ::: . : .: :
UNIPRO SCAGKLVSGEIDQSDQSFLDDDQMEEGWV-LTCAAYPKSDVVIETHKEEELTA
          50        60        70         80        90


128 residues in 1 query   sequences
501934690 residues in 1568424 library sequences
 Tcomplib [34t23] (4 proc)
 start: Tue Sep 28 13:05:09 2004 done: Tue Sep 28 13:06:32 2004
 Total Scan time: 272.383 Total Display time:  0.033

Function used was FASTA [version 3.4t23 March 18, 2004]
```

FASTA format

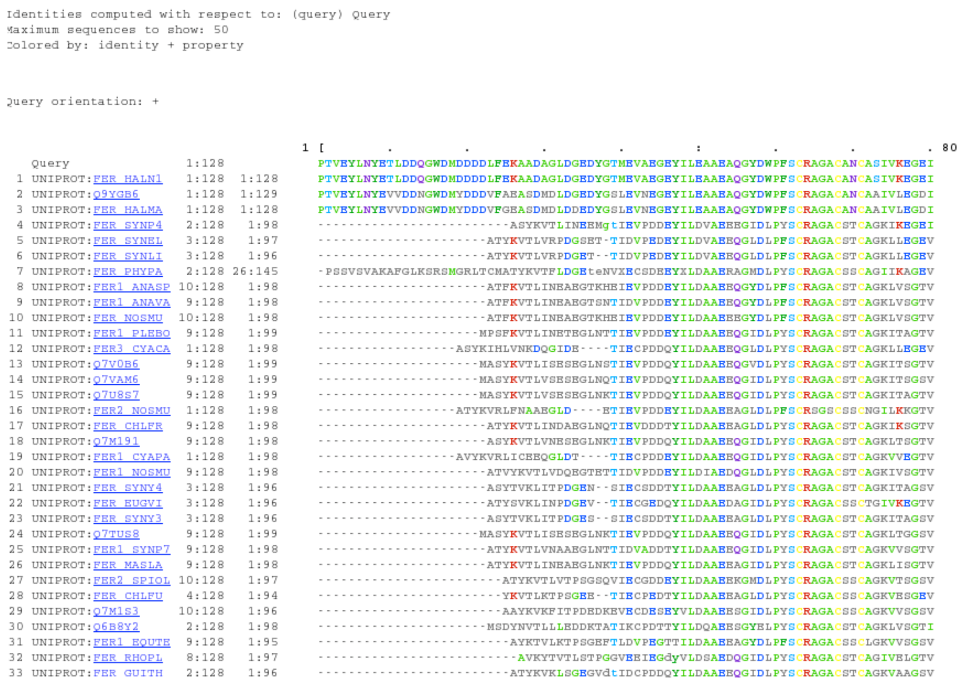
The textual output as shown above is only one possible output available. In addtition to the textual output, you can request an MVIEW (a mulitple alignment view) as in Figure 6.1 or a visual fasta view (a graphical version of the signficance) as in Figure 6.2.

The textual output from the FASTA search begins with some informational messages. This includes the reference that you should cite, the version number of the program and the libraries that were searched. In this case, an optional histogram (lying on its side) has been requested of the number of sequences found with various scores. Each equal symbol in this histogram is an indicator of 2608 sequences and the asterisk indicates the expected number. The tail of the distribution is expanded in the inset. Here each equal symbol represents 13 sequences. This histogram gives you an indication of how similar the query sequence is to some of the database sequences. For a query sequence that has found a significant match, it should be well out of the tail of the distribution. In this example there are many sequences with scores larger than 120 and they are more frequent than expected by chance. These are related ferredoxin sequences from other species.

Next comes some information about the size of the database searched (note the size of the numbers) and some statistics about the search. Next comes a section that lists the sequences (along with their locus names) that have the best scores. Finally there is a section that lists the alignments that have been found by the program.

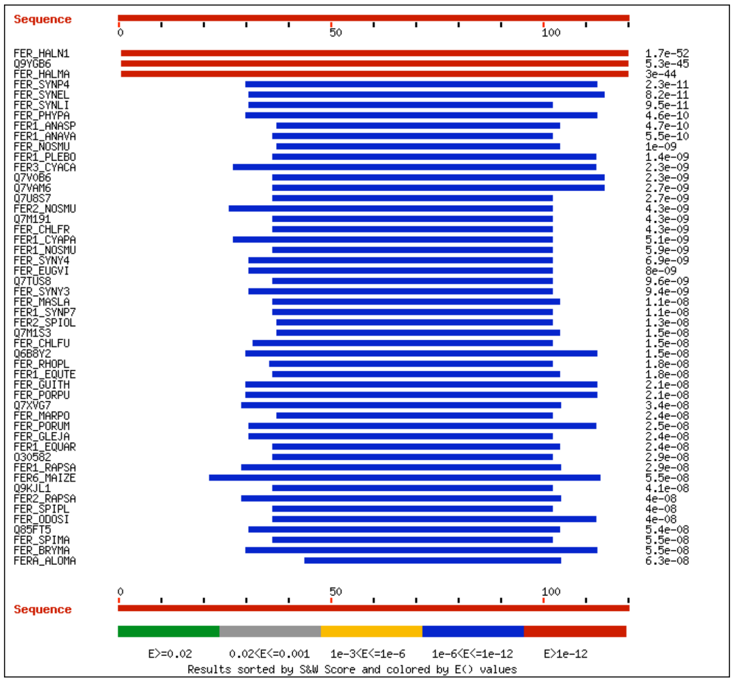
To carry out a database search in this manner, the algorithm first establishes a table containing words from the database sequences of variable length (e.g. ATCGGA, ACCCTG, GTCACA, . . . for nucleotides or MK, RS, CP, . . . for proteins). This type of preprocessing of the entire database is necessary to speed the subsequent search. This table is then sorted in alphabetical order and allows matching words (from the query sequences) to found rapidly. The length of these words is

Figure 6.1: The MVIEW output from http://www.ebi.ac.uk/fasta for the ferredoxin data



```
Identities computed with respect to: (query) Query
Maximum sequences to show: 50
Colored by: identity + property


Query orientation: +


                                      1 [         .         .         .         :         .         .         . 80
   Query                 1:128          PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWPFSCRAGACANCASIVKEGEI
 1 UNIPROT:FER_HALN1     1:128  1:128   PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWPFSCRAGACANCASIVKEGEI
 2 UNIPROT:Q9YGB6        1:128  1:129   PTVEYLNYEVVDDNGWDMYDDDVFAEASDMDLDGEDYGSLEVNEGEYILEAAEAQGYDWPFSCRAGACANCAAIVLEGDI
 3 UNIPROT:FER_HALMA     1:128  1:128   PTVEYLNYEVVDDNGWDMYDDDVFGEASDMDLDDEDYGSLEVNEGEYILEAAEAQGYDWPFSCRAGACANCAAIVLEGDI
 4 UNIPROT:FER_SYNP4     2:128   1:98   -------------------------ASYKVTLINEEMgtIEVPDDEYILDVAEEEGIDLPYSCRAGACSTCAGKIKEGEI
 5 UNIPROT:FER_SYNEL     3:128   1:97   ----------------------ATYKVTLVRPDGSET-TIDVPEDEYILDVAEEQGLDLPFSCRAGACSTCAGKLLEGEV
 6 UNIPROT:FER_SYNLI     3:128   1:96   ----------------------ATYKVTLVRPDGET--TIDVPEDEYILDVAEEQGLDLPFSCRAGACSTCAGKLLEGEV
 7 UNIPROT:FER_PHYPA     2:128 26:145   -PSSVSVAKAFGLKSRSMGRLTCMATYKVTFLDGEteNVXECSDEEYXLDAAERAGMDLPYSCRAGACSSCAGIIKAGEV
 8 UNIPROT:FER1_ANASP   10:128   1:98   ----------------------ATFKVTLINEAEGTKHEIEVPDDEYILDAAEEQGYDLPFSCRAGACSTCAGKLVSGTV
 9 UNIPROT:FER1_ANAVA    9:128   1:98   ----------------------ATFKVTLINEAEGTSNTIDVPDDEYILDAAEEQGYDLPFSCRAGACSTCAGKLVSGTV
10 UNIPROT:FER_NOSMU    10:128   1:98   ----------------------ATFKVTLINEAEGTKHEIEVPDDEYILDAAEEEGYDLPFSCRAGACSTCAGKLVSGTV
11 UNIPROT:FER1_PLEBO    9:128   1:99   ---------------------MPSFKVTLINETEGLNTTIEVPDDEYILDAAEEQGIDLPYSCRAGACSTCAGKITAGTV
12 UNIPROT:FER3_CYACA    1:128   1:98   ------------------ASYKIHLVNKDQGIDE----TIECPDDQYILDAAEEQGLDLPYSCRAGACSTCAGKLLEGEV
13 UNIPROT:Q7V0B6        9:128   1:99   ---------------------MASYKVTLISESEGLNSTIEVPDDQYILDAAEEQGVDLPYSCRAGACSTCAGKITSGTV
14 UNIPROT:Q7VAM6        9:128   1:99   ---------------------MASYKVTLVSESEGLNQTIEVPDDQYILDAAEEQGIDLPYSCRAGACSTCAGKITSGSV
15 UNIPROT:Q7U8S7        9:128   1:99   ---------------------MASYKVTLVSESEGLNKTIEVPDDQYILDAAEEQGIDLPYSCRAGACSTCAGKITAGTV
16 UNIPROT:FER2_NOSMU    1:128   1:98   ------------------ATYKVRLFNAAEGLD----ETIEVPDDEYILDAAEEAGLDLPFSCRSGSCSSCNGILKKGTV
17 UNIPROT:FER_CHLFR     9:128   1:98   ----------------------ATYKVTLINDAEGLNQTIEVDDDTYILDAAEEAGLDLPYSCRAGACSTCAGKIKSGTV
18 UNIPROT:Q7M191        9:128   1:98   ----------------------ASYKVTLVNESEGLNKTIEVPDDQYILDAAEEQGIDLPYSCRAGACSTCAGKLTSGTV
19 UNIPROT:FER1_CYAPA    1:128   1:98   ------------------AVYKVRLICEEQGLDT----TIECPDDEYILDAAEEQGIDLPYSCRAGACSTCAGKVVEGTV
20 UNIPROT:FER1_NOSMU    9:128   1:98   ----------------------ATVYKVTLVDQEGTETTIDVPDDEYILDIAEDQGLDLPYSCRAGACSTCAGKIVSGTV
21 UNIPROT:FER_SYNY4     3:128   1:96   ----------------------ASYTVKLITPDGEN--SIECSDDTYILDAAEEAGLDLPYSCRAGACSTCAGKITAGSV
22 UNIPROT:FER_EUGVI     3:128   1:96   ----------------------ATYSVKLINPDGEV--TIECGEDQYILDAAEDAGIDLPYSCRAGACSSCTGIVKEGTV
23 UNIPROT:FER_SYNY3     3:128   1:96   ----------------------ASYTVKLITPDGES--SIECSDDTYILDAAEEAGLDLPYSCRAGACSTCAGKITAGSV
24 UNIPROT:Q7TUS8        9:128   1:99   ---------------------MASYKVTLISESEGLNKTIEVPDDQYILDAAEEQGIDLPYSCRAGACSTCAGKLTGGSV
25 UNIPROT:FER1_SYNP7    9:128   1:98   ----------------------ATYKVTLVNAAEGLNTTIDVADDTYILDAAEEQGIDLPYSCRAGACSTCAGKVVSGTV
26 UNIPROT:FER_MASLA     9:128   1:98   ----------------------ATYKVTLINEAEGLNKTIEVPDDQYILDAAEEAGIDLPYSCRAGACSTCAGKLISGTV
27 UNIPROT:FER2_SPIOL   10:128   1:97   ------------------------ATYKVTLVTPSGSQVIECGDDEYILDAAEEKGMDLPYSCRAGACSSCAGKVTSGSV
28 UNIPROT:FER_CHLFU     4:128   1:94   ------------------------YKVTLKTPSGEE--TIECPEDTYILDAAEEAGLDLPYSCRAGACSSCAGKVESGEV
29 UNIPROT:Q7M1S3       10:128   1:96   ------------------------AAYKVKFITPDEDKEVECDESEYVLDAAEESGIDLPYSCRAGACSSCAGKVVSGSV
30 UNIPROT:Q6B8Y2        2:128   1:98   ----------------------MSDYNVTLLLEDDKTATIKCPDTTYILDQAEESGYELPYSCRAGACSTCAGKLVSGTI
31 UNIPROT:FER1_EQUTE    9:128   1:95   -------------------------AYKTVLKTPSGEFTLDVPEGTTILDAAEEAGYDLPFSCRAGACSSCLGKVVSGSV
32 UNIPROT:FER_RHOPL     8:128   1:97   --------------------------AVKYTVTLSTPGGVEEIEGdyVLDSAEDQGIDLPYSCRAGACSTCAGIVELGTV
33 UNIPROT:FER_GUITH     2:128   1:96   -------------------------ATYKVKLSGEGVdtIDCPDDQYILDAAEEQGIDLPYSCRAGACSTCAGKVAAGSV
```

Figure 6.2: The VISUALFASTA output from http://www.ebi.ac.uk/fasta for the ferredoxin data



set by the WORD or k-tuple parameter value. By default it is 6 for nucleic acids and 2 for amino acid searches. A lower k-tuple will give a more sensitive search but will take much longer. Although a range of 3 to 6 is permitted for nucleic acids a lower value is generally unnecessary. All places in the query sequence are determined where the k-tuple from both sequences agree perfectly. Then those regions with the highest density of these identities are found.

In comparing a query sequence to the database three scores are calculated for each and every entry in the database. These scores are init1, initn and opt. An init1 score is assigned to each of these regions of high similarity after the regions are extended at the ends to include regions shorter than the length of a k-tuple and after using a BLOSUM50 matrix (alternative distance matrices are available – more on these later) to score mismatches.

Groups of larger regions are attempted to be joined together and an initn score is generated from these. This is done by setting initn equal to the sum of the two init1 scores for each region (the final init1 score of a sequence is the maximum init1 score from all interior regions). A constant of 20 is then subtracted as a joining penalty. If the initn score is less than one of the init1 scores it is discarded, the regions are not joined and the initn score will be equal to the maximum init1 score (hence initn is greater than or equal to init1).

Sequences that have an initn score larger than a cutoff value (usually 50 but this can be altered with a "LIST n" command in the query file) are then used for a Smith-Waterman alignment (see the section on alignments) and an opt score is generated from these alignments. Only the region considered significant by the program is displayed. In these alignments, the name of the sequence will be presented, the scores, and the percent similarity over the region aligned. In general the length of the region aligned is a better indicator of homology than is the percent similarity. This is because large percentages can be found in short regions just by chance. A ':' is used to indicate a complete match, a '.' to indicate a conservative amino acid replacement, and a '-' to indicate a deletion/insertion.

Note that the opt score can be lower than the initn score. This will happen when one sequence has two (or more) regions of high similarity separated by regions that have little/no homology. The two regions are joined with high init1 scores and the initn score is high because the gap penalty/join penalty is not sufficiently large. In contrast sequences with a large number of poorly similar regions will have low init1 scores but high initn scores and then low opt scores. In general, unless a very short sequence is used, the init1 score should be much improved by the opt score for truly significant sequences. Lastly a z-score based on estimates of the statistical significance of the opt scores is presented. This estimates the probability of obtaining opt scores as good or better by chance between unrelated sequences (see below).

Remember to remove repetitive sequences from your query otherwise you will get a lot of false hits. The FASTA program itself can be obtained via anonymous `ftp` if desired.

Statistical Significance

Since version 2.0 of the FASTA program distribution, FASTA, TFASTA, and SSEARCH will provide estimates of statistical significance for library searches. Work by Altschul, Arratia, Karlin, Mott, Waterman, and others (see Altschul *et al.* 1994 Nature Genetics 6:119-129 for an excellent review) shows that local sequence similarity scores follow an extreme value distribution. The probability of a database match score larger than x arising by chance alone is therefore

$$P(s \geq x) = 1 - e^{-e^{-\lambda(x-u)}}$$

where for ungapped alignments

$$u = \frac{\ln(Kmn)}{\lambda}$$

and m, n are the lengths of the query and library sequence and K and λ are constants that depend on the substitution scores and the sequence compositions. This formula can be rewritten as:

$$1 - e^{-Kmn(e^{-\lambda x})}$$

which shows that the probability of observing larger scores for unrelated library sequences increases logarithmically with the length of the library sequence (Pearson - FASTA documentation).

FASTA and SSEARCH produce gapped alignments and hence use a simple linear regression against the log of the library sequence length to calculate a normalized "z-score" with mean 50, regardless of library sequence length, and variance 10.

These z-scores can then be used with the extreme value distribution and the poisson distribution (to account for the fact that each library sequence comparison is an independent test) to calculate the expected number of library sequences required to obtain a score greater than or equal to the score obtained in the search (Pearson - FASTA documentation).

The expected number of sequences is plotted in the histogram using an '$*$'. Since the parameters for the extreme value distribution are not calculated directly from the distribution of similarity scores, the pattern of '$*$'s in the histogram gives a qualitative view of how well the statistical theory fits the similarity scores calculated by FASTA and SSEARCH. For FASTA, optimized scores are calculated for each sequence in the database and the agreement between the actual distribution of "z-scores" and the expected distribution based on the length dependence of the score and the extreme value distribution is usually very good. Likewise, the distribution of SSEARCH Smith-Waterman scores typically agrees closely with the actual distribution of "z-scores." The agreement with unoptimized scores, $ktup = 2$, is often not very good, with too many high scoring sequences and too few low scoring sequences compared with the predicted relationship between sequence length and similarity score. In those cases, the expectation values may be overestimates (Pearson - FASTA documentation).

The statistical routines assume that the library contains a large sample of unrelated sequences. If this is not the case, then the expectation values are meaningless. Likewise, if there are fewer than 20 sequences in the library, the statistical calculations are not done (Pearson - FASTA documentation).

A complete manual for FASTA or the online FASTA3 help at EBI can be consulted for further information.

6.1.2 BLAST

While FASTA is a sensitive and rapid algorithm to search for similar sequences in the database it is not without problems. Because its initial step looks for perfect matches it might be less sensitive to more distantly related sequences that have functional homology but no longer retain complete identity. If an amino acid sequence has had many conserved replacements but no longer has identities then the FASTA algorithm might not identify these as well as it should. Fortunately, alignments where there are extensive regions of low but not exact similarity are rare enough that a small WORD or k-tuple size will pick up most regions.

A different algorithm which improves upon FASTA in speed is termed BLAST (Basic Local Alignment Search Tool). This began with a statistical paper by Karlin and Altschul (PNAS 87:2264-2268, 1990) who developed a rigorous method to obtain the probabilities of matches with a query sequence given that no gaps are permitted. This permits the use of larger WORD or k-tuple sizes with the concomitant increase in speed but permitting inexact matches between WORDs. The statistical developments permit this to be done without loss of sensitivity and allow rigorous statistical statements to be made about the matches found.

As a result of these developments Altschul, Gish, Miller, Myers and Lipman (J.Mol.Biol. 215:403-415, 1990) created the BLAST group of programs. These algorithms find ungapped, locally optimal sequence alignments. There are several versions of the BLAST programs. Some are

`BLASTN` - nucleotide query of the nucleotide database.

`BLASTP` - protein query of the protein database.

`BLASTX` - translate DNA to protein and query protein database.

`TBLASTN` - protein query of the translated nucleotide database.

`TBLASTX` - translate DNA to protein and query the translated nucleotide database.

`PHI-BLAST` - pattern-hit initiated program takes a user search pattern and finds proteins similar.

`PSI-BLAST` - use position-specific iterative score matrices to search for protein "motifs" or "profiles".

`MEGA-BLAST` - nucleotide query of the nucleotide database.

`discontiguous MEGA-BLAST` - nucleotide query of the nucleotide database.

The last two use a different algorithm than does `BLASTN`. The program `MEGA-BLAST` uses a "greedy algorithm" for nucleotide sequence alignment search and is designed to find sequences that differ slightly from the query sequence. Hence is best at identifying something "similar" in the database without concern about distant homologies. Because it is much faster than `BLASTN` it is also permitted to submit multiple queries to the database with `MEGA-BLAST` (simply add more than one query sequence). The program `discontiguous MEGA-BLAST` increases sensitvity to diverged sequences by using a discontiguous word as the initial match from which extensions are performed (see below).

To carry out this type of search go to the NCBI BLAST web server, select the desired program and fill out the forms.

Most of the options will take standard default values. The database for example, has a default of “nr”. This means that it will search the non-redundant database (it includes sequences from PDB, GenBank, GenBank updates, EMBL and EMBL updates or sequences from PDB, SWISS-PROT, PIR, GenPept and GenPept updates) but there are many others that can be chosen instead. In addition you can chose to search only specific groups of organisms or to search sequences that originated from only one organism. Filter’s will mask parts of your query so that things like repetitive elements are ignored (filter seq - will exclude regions of low compositional complexity, filter dust - is a modernized filter version that at the time of this writing has not yet been described in the literature. Other filter’s will exclude regions with repetitive elements). It is also possible to select the number of DESCRIPTIONS n, the number of described matching sequences [100]. ALIGNMENTS n, number of high scoring pairs [50], the EXPECT n, the score such that n sequences should be found by chance alone [10] (a fractional value of one or less will give only output which is statistically unusual, larger values give more output) and the WORD size used for initial matches. Other options are available.

More information about the programs and their output can be obtained from NCBI’s BLAST site including a

- BLAST overview
- BLAST FAQs
- BLAST Program Selection Guide
- BLAST course
- BLAST tutorial
- BLAST help

The BLAST programs themselves can be obtained if desired by anonymous `ftp` to NCBI (with more options possible (and permissible)) and if desired, a newwork client that works directly through TCP/IP connections (hence, no web browser required) can be obtained as BLASTcl3 from the ftp site.

BLAST output

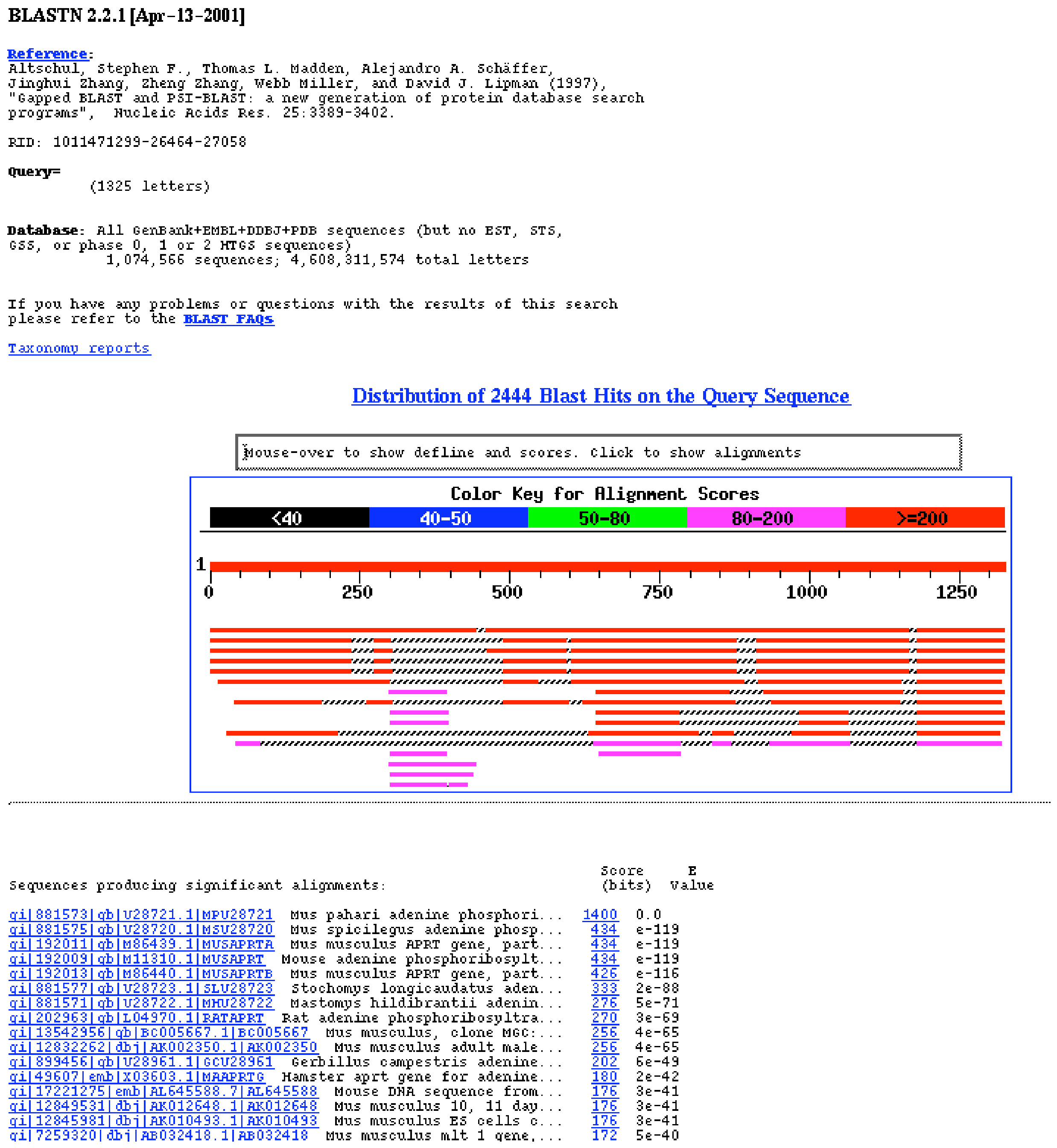
Typical BLAST output appears as in Figure 6.3 (this search was done on Jan 19 2002 with an APRT gene from *Mus pahari* as the query).

Each of the blue-highlighted pieces of text are links that leads directly to the entry in the database that matches the query. There is a diagram at the top of the entry that graphically demonstrates the hits and how they align with the query sequence. It is colour coded according to the statistical level of the match. In this diagram regions of low match are in gray hatch-marks. Note that even though the query sequence is in the database, there are these hatch-marks in the first matching sequence. This is because these sequence regions contain low complexity DNA (e.g. J.C. Wootton, 1994 Comput Chem 18:269-285) that would disrupt the statistical measures of similarity and hence they have been excluded by default from the match (this behaviour can be altered ... see above).

After the listing of hits comes a section that lists the match between the query sequence and the database match.

```
gi|10442645|gb|AF279458.1|AF279458  Mus musculus Ran-binding...    157  3e-35
gi|13752160|gb|AC091473.1|AC091473  Mus musculus chromosome ...    157  3e-35
gi|13194583|gb|AF316998.1|AF316998  Mus musculus D11lgp1 gen...    157  3e-35
gi|13160934|gb|AF304466.1|AF304466  Mus musculus adipocyte c...    157  3e-35
gi|12000469|gb|AC078930.13|AC078930  Mus musculus 10 BAC 280...    157  3e-35

                                Alignments

>gi|881573|gb|U28721.1|MPU28721 Mus pahari adenine phosphoribosyltransferase (APRT) gene, complete
           cds
         Length = 2283
```

Figure 6.3: Typical results of a BLAST search



BLASTN 2.2.1 [Apr-13-2001]

```
Reference:
Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schäffer,
Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman (1997),
"Gapped BLAST and PSI-BLAST: a new generation of protein database search
programs",  Nucleic Acids Res. 25:3389-3402.

RID: 1011471299-26464-27058

Query=
         (1325 letters)


Database: All GenBank+EMBL+DDBJ+PDB sequences (but no EST, STS,
GSS, or phase 0, 1 or 2 HTGS sequences)
           1,074,566 sequences; 4,608,311,574 total letters


If you have any problems or questions with the results of this search
please refer to the BLAST FAQs

Taxonomy reports
```

Distribution of 2444 Blast Hits on the Query Sequence

Mouse-over to show defline and scores. Click to show alignments

```
                                                                   Score     E
Sequences producing significant alignments:                        (bits)  Value

gi|881573|gb|U28721.1|MPU28721  Mus pahari adenine phosphori...   1400   0.0
gi|881575|gb|U28720.1|MSU28720  Mus spicilegus adenine phosp...    434   e-119
gi|192011|gb|M86439.1|MUSAPRTA  Mus musculus APRT gene, part...    434   e-119
gi|192009|gb|M11310.1|MUSAPRT  Mouse adenine phosphoribosylt...    434   e-119
gi|192013|gb|M86440.1|MUSAPRTB  Mus musculus APRT gene, part...    426   e-116
gi|881577|gb|U28723.1|SLU28723  Stochomys longicaudatus aden...    333   2e-88
gi|881571|gb|U28722.1|MHU28722  Mastomys hildibrantii adenin...    276   5e-71
gi|202963|gb|L04970.1|RATAPRT  Rat adenine phosphoribosyltra...    270   3e-69
gi|13542956|gb|BC005667.1|BC005667  Mus musculus, clone MGC:...    256   4e-65
gi|12832262|dbj|AK002350.1|AK002350  Mus musculus adult male...    256   4e-65
gi|899456|gb|U28961.1|GCU28961  Gerbillus campestris adenine...    202   6e-49
gi|49607|emb|X03603.1|MAAPRTG  Hamster aprt gene for adenine...    180   2e-42
gi|17221275|emb|AL645588.7|AL645588  Mouse DNA sequence from...    176   3e-41
gi|12849531|dbj|AK012648.1|AK012648  Mus musculus 10, 11 day...    176   3e-41
gi|12845981|dbj|AK010493.1|AK010493  Mus musculus ES cells c...    176   3e-41
gi|7259320|dbj|AB032418.1|AB032418  Mus musculus mlt 1 gene,...    172   5e-40
```

```
 Score = 1400 bits (706), Expect = 0.0
 Identities = 706/706 (100%)
 Strand = Plus / Plus


Query: 460  gaaagaaaggtggcaagagccaccatagtggagaaggcaggtaggatccccaaggctaag 519
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1321 gaaagaaaggtggcaagagccaccatagtggagaaggcaggtaggatccccaaggctaag 1380


Query: 520  atgctaccgagtaaccatcagtgttcttctagccatagtgggcaagacctagtgttccta 579
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1381 atgctaccgagtaaccatcagtgttcttctagccatagtgggcaagacctagtgttccta 1440

......................................................................
......................  Material Deleted  ..........................
......................................................................


Query: 1060 tggaggtaaagaaccagcccaagacaaacaggcttcaaagggccaggccctgtctggggt 1119
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1921 tggaggtaaagaaccagcccaagacaaacaggcttcaaagggccaggccctgtctggggt 1980


Query: 1120 gctgactaaacaaagcgcttgaataccttctctttctctgtccctt 1165
            ||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1981 gctgactaaacaaagcgcttgaataccttctctttctctgtccctt 2026

 Score =  882 bits (445), Expect = 0.0
 Identities = 445/445 (100%)
 Strand = Plus / Plus


Query: 1    aagcttgtgctaaacaactgctgtataccaggctccatgcttgagcttcagaaacaccct 60
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 862  aagcttgtgctaaacaactgctgtataccaggctccatgcttgagcttcagaaacaccct 921


Query: 61   agggcagctgaatgtccaccaggagtgtccagagggagggtgagcaccccaagagaacag 120
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 922  agggcagctgaatgtccaccaggagtgtccagagggagggtgagcaccccaagagaacag 981

......................................................................
......................  Material Deleted  ..........................
......................................................................

Query: 361  ttcaaatcccagcaaccacatggtggctcacaaccacctacagctacagtgtacacacat 420
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 1222 ttcaaatcccagcaaccacatggtggctcacaaccacctacagctacagtgtacacacat 1281


Query: 421  ataataaaataaataaacaaatctt 445
            |||||||||||||||||||||||||
Sbjct: 1282 ataataaaataaataaacaaatctt 1306

 Score =  287 bits (145), Expect = 1e-74
 Identities = 145/145 (100%)
 Strand = Plus / Plus


Query: 1181 aggaaccatgtttgcagcctgtgatctgctgcaccagctacgggctgaggtggtggagtg 1240
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 2042 aggaaccatgtttgcagcctgtgatctgctgcaccagctacgggctgaggtggtggagtg 2101


Query: 1241 tgtgagcctggtggagctgacctcgctgaagggcagggagaggctaggacctataccatt 1300
            ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct: 2102 tgtgagcctggtggagctgacctcgctgaagggcagggagaggctaggacctataccatt 2161


Query: 1301 cttctctctcctccagtatgactga 1325
            |||||||||||||||||||||||||
Sbjct: 2162 cttctctctcctccagtatgactga 2186

>gi|881575|gb|U28720.1|MSU28720 Mus spicilegus adenine phosphoribosyltransferase (APRT) gene,
            complete cds
          Length = 2117

 Score =  434 bits (219), Expect = e-119
 Identities = 263/277 (94%), Gaps = 3/277 (1%)
 Strand = Plus / Plus


Query: 603  tgcctctcggctccatcccacacccttccctccttaccctaacaggtctagactccaggg 662
            |||| ||| ||||||||||||| |||||||||||||||||||||||||||||||||||||
Sbjct: 1310 tgcccctcagctccatcccacaaccttccctccttaccctaacaggtctagactccaggg 1369


Query: 663  gcttcctgtttggcccttccctagctcaggagctgggcgtgggctgcgtgctcatccgga 722
            |||||||||||||||||||||||||||||||||||||||||||||| |||||||||||||
Sbjct: 1370 gcttcctgtttggcccttccctagctcaggagctgggcgtgggctgtgtgctcatccgga 1429


.....................................................................
.......................  Material Deleted  ..........................
.....................................................................
```

Note that this alignment might be in pieces as demonstrated above even for the database entry which is a perfect match. Further down the listing will be generally shorter matches such as . . .

```
>gi|17221275|emb|AL645588.7|AL645588 Mouse DNA sequence from clone RP23-452K19 on chromosome 11, complete
            sequence [Mus musculus]
          Length = 5004

 Score =  176 bits (89), Expect = 3e-41
 Identities = 95/97 (97%)
 Strand = Plus / Plus


Query: 300  agagggctggtgagatggctcagcggttaggagcactgactgctcttccaaaggtcctga 359
            ||||||||||||||||||||||||||||| ||||||||||||||||||||||||||| ||
Sbjct: 4634 agagggctggtgagatggctcagcggttaagagcactgactgctcttccaaaggtcccga 4693


Query: 360  gttcaaatcccagcaaccacatggtggctcacaacca 396
            |||||||||||||||||||||||||||||||||||||
Sbjct: 4694 gttcaaatcccagcaaccacatggtggctcacaacca 4730

>gi|12849531|dbj|AK012648.1|AK012648 Mus musculus 10, 11 days embryo whole body cDNA, RIKEN full-length
           enriched library, clone:2810002N01:related to Y39B6B.P
           PROTEIN, full insert sequence
          Length = 1026

 Score =  176 bits (89), Expect = 3e-41
 Identities = 95/97 (97%)
 Strand = Plus / Plus


Query: 302 agggctggtgagatggctcagcggttaggagcactgactgctcttccaaaggtcctgagt 361
           ||||||||||||||||||||||||||| |||| |||||||||||||||||||||||||||
Sbjct: 841 agggctggtgagatggctcagcggttaagagcgctgactgctcttccaaaggtcctgagt 900


Query: 362 tcaaatcccagcaaccacatggtggctcacaaccacc 398
           |||||||||||||||||||||||||||||||||||||
Sbjct: 901 tcaaatcccagcaaccacatggtggctcacaaccacc 937

>gi|12845981|dbj|AK010493.1|AK010493 Mus musculus ES cells cDNA, RIKEN full-length enriched library,
           clone:2410015G15:related to Y39B6B.P PROTEIN, full
           insert sequence
          Length = 1022

 Score =  176 bits (89), Expect = 3e-41
 Identities = 95/97 (97%)
 Strand = Plus / Plus


Query: 302 agggctggtgagatggctcagcggttaggagcactgactgctcttccaaaggtcctgagt 361
           ||||||||||||||||||||||||||| |||| |||||||||||||||||||||||||||
Sbjct: 833 agggctggtgagatggctcagcggttaagagcgctgactgctcttccaaaggtcctgagt 892
```

and finally at the bottom of the entry will be some statistics about the search . . .

```
Query: 303   gggctggtgagatggctcagcggttaggagcactgactgctctt 346
             ||||||| ||||||||||||| || | ||| |||| ||||||||
Sbjct: 30925 gggctggagagatggctcagctgtgaagaggactggctgctctt 30968

 Score = 40.1 bits (20), Expect = 4.9
 Identities = 44/52 (84%)
 Strand = Plus / Minus


Query: 345   ttccaaaggtcctgagttcaaatcccagcaaccacatggtggctcacaacca 396
             ||||| ||||||||||||  | ||||||||||||||   | |||||||||||
Sbjct: 30682 ttccagaggtcctgagttttattcccagcaaccacaccatagctcacaacca 30631

  Database: All GenBank+EMBL+DDBJ+PDB sequences (but no EST, STS, GSS,
  or phase 0, 1 or 2 HTGS sequences)
    Posted date:  Jan 19, 2002 12:06 AM
  Number of letters in database: 313,344,278
  Number of sequences in database:  1,074,566

Lambda     K      H
    1.37    0.711     1.31

Gapped
Lambda     K      H
    1.37    0.711     1.31

Matrix: blastn matrix:1 -3
Gap Penalties: Existence: 5, Extension: 2
Number of Hits to DB: 4,147,065
Number of Sequences: 1074566
Number of extensions: 4147065
Number of successful extensions: 57560
Number of sequences better than 10.0: 2475
length of query: 1325
length of database: 4,608,311,574
effective HSP length: 22
effective length of query: 1303
effective length of database: 4,584,671,122
effective search space: 5973826471966
effective search space used: 5973826471966
T: 0
A: 30
X1: 6 (11.9 bits)
X2: 15 (29.7 bits)
S1: 12 (24.3 bits)
S2: 20 (40.1 bits)
```

BLAST format

The program output consists of three parts. The first part is a graphical diagram of the top matches to the query sequence. The second is a listing of the best matches (along with links to their database entries), their scores and their E value. The E-value is a estimate of how many matches as good or better would occur by chance alone in a database of this size. The third part is an alignment of the matches with the query sequence. The fourth part of the output will be a listing of the parameters used and some statistics of the search. Some of these parameters can be changed (see the documentation for more information) but others cannot be changed. NCBI is aware of the tradeoffs in speed versus sensitivity and attempts to offer a service with the most sensitive parameter settings that its machines can handle.

Remember that BLAST will find matches of ungapped strings. There may be more than one "ungapped" region that give an unusually large score. These multiple regions are not ignored but rather attempts are made to put them together to yield a lower overall probability. The statistics for the ungapped strings are well worked out, but the statistics for gapped matches are still not well understood.

The BLAST algorithms are capable of speeding through the entire databases within just a few seconds. Its speed is impressive. BLAST requires time proportional to the product of the query sequence length and the length of the database. The databases are growing far more quickly than are improvements in the speed of the computers or in the design of the algorithms.

The particular example shown above is a search of the database for homologues to the *Mus pahari* APRT sequence. You will note that the algorithm has done a good job at finding these homologues. The next match is the APRT gene of *Mus* *spicilegus* (a closely related species – with a correspondingly closely related APRT sequence) and not surprisingly it has a probability of 1×10^{-119} of being just a chance match (effectively zero).

An older search of the same sequence found the same matches and if you continued down the list you would see . . .

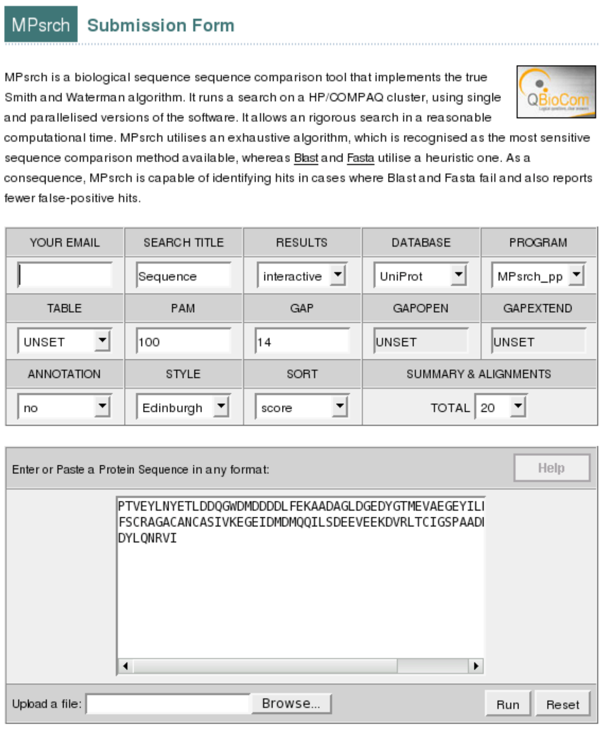
```
                                                                  Smallest
                                                                    Sum
                                                             High  Probability
Sequences producing High-scoring Segment Pairs:              Score  P(N)      N

gb|U28721|MPU28721    Mus pahari adenine phosphoribosyltr...  6451  0.0        1
gb|M86440|MUSAPRTB    Mus musculus APRT gene, partial cds.    1002  1.6e-296  10
gb|U28720|MSU28720    Mus spicilegus adenine phosphoribos...  1002  5.3e-295  12
gb|M86439|MUSAPRTA    Mus musculus APRT gene, partial cds.    1002  1.3e-290  10
gb|M11310|MUSAPRT     Mouse adenine phosphoribosyltransfe...  1002  2.6e-290  10
gb|U28723|SLU28723    Stochomys longicaudatus adenine pho...   887  5.4e-250  11

......................................................................
......................  Material Deleted  ..........................
......................................................................

gb|U13835|MMCABL1     Mus musculus c-abl oncogene (c-abl)...   446  3.6e-27    2
gb|M34073|MUSMHT10C   Mus musculus (clone T10-c) MHC clas...   417  4.5e-27    2
gb|U63716|MMU63716    Mus musculus cytochrome C oxidase s...   440  4.9e-27    2
emb|Y00629|MMG37      Murine gene 37 for pot. membrane bo...   418  5.4e-27    2
dbj|D88356|D88356     Mouse DNA for 8-oxodGTPase, complet...   445  8.3e-27    2
gb|U06950|MMU06950    Mus musculus C57BL/6 lymphotoxin-be...   433  8.4e-27    2
gb|U96726|MMU96726    Mus musculus vibrator critical regi...   440  8.9e-27    2
gb|U42467|MMU42467    Mus musculus leptin receptor (Ob-r)...   432  9.6e-27    2
gb|U22062|RNU22062    Rattus norvegicus neurogranin/RC3 p...   408  1.5e-26    2
emb|X80685|MMGMCK2B   M.musculus gMCK2-beta gene               431  3.6e-25    1
```

So as you go down the list you find more APRT genes but also, later on, some oncogene – *c-abl*. So now you get all excited — we have discovered a new class of genes involved in cancer! Major advance . . . international acclaim, . . . **Noble Prize**!! But wait, we must be cautious here, what do the statistics say. Well for this *c-abl* gene the match has a probability of 3.6×10^{-27} of occurring by chance alone. So we are home free, that is significant in anyone's statistic book. But life is seldom so exciting. As you continue to scan the list, you find cytochrome C, membrane proteins, growth factors, and all sorts of other genes all with significant homology to the query sequence. What is going on?

Remember that BLAST (and any of the other algorithms) search for similarity not of the entire sequence but rather for any piece of the query sequence. Examining the regions of significant match between the database sequences and the query sequence indicates that these are consistently from approximately nucleotides 302 to 431 but not generally outside of this region. This region encodes a very common SINE element in rodents. Hence there is no similarity of the query gene to all these other genes but there is a significant similarity of the B2 SINE element that is inserted into the APRT gene and the B2 SINE elements that have been inserted into the other gene sequences. Be careful of the interpretation of your results — no Noble prize this time.

Occassionally, other features such as a coiled-coil region or transmembrane regions will cause falsely positive matches to be predicted. In addition, although not a false match, the results of exon shuffling can copy a motif from one protein to another and might lead one to consider that the entire lengths of these two proteins are homologous (and derived evolutionarily from the other) when it is really only the motif that is similar. Sometimes, functional requirements will cause selection to pick on a pattern of amino acids that are similar again without homology.

Another common misuse of BLAST is to search for the most similar sequence to some query sequence. But the algorithm is designed to find similar ungapped subsequences, and to then piece these together. The order in which these sequences are ranked by score may not correspond to the order of overall similarity of the complete sequences, and certainly may not correspond to the phylogenetic history of these sequences (Koski & Golding 2001, J.Mol.Evol. 52:540-542). Thus a sequence with a higher score may not be more 'similar' to the query sequence than another sequence with a lower score (more later on what is meant by similarity). It is quite possible for the overall similarity to be greater for a sequence with a lower BLAST score. A sequence may also be more closely related in terms of history to the query than some other sequence with a lower score.



MPsrch Submission Form

MPsrch is a biological sequence sequence comparison tool that implements the true Smith and Waterman algorithm. It runs a search on a HP/COMPAQ cluster, using single and parallelised versions of the software. It allows an rigorous search in a reasonable computational time. MPsrch utilises an exhaustive algorithm, which is recognised as the most sensitive sequence comparison method available, whereas Blast and Fasta utilise a heuristic one. As a consequence, MPsrch is capable of identifying hits in cases where Blast and Fasta fail and also reports fewer false-positive hits.

YOUR EMAIL	SEARCH TITLE	RESULTS	DATABASE	PROGRAM
	Sequence	interactive	UniProt	MPsrch_pp
TABLE	PAM	GAP	GAPOPEN	GAPEXTEND
UNSET	100	14	UNSET	UNSET
ANNOTATION	STYLE	SORT	SUMMARY & ALIGNMENTS	
no	Edinburgh	score	TOTAL 20	

Enter or Paste a Protein Sequence in any format:

```
PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILI
FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADI
DYLQNRVI
```

Upload a file:

Figure 6.4: The webpage for input to an MPsrch

6.1.3 MPsrch

The MPsrch server at EBI runs on an HP/COMPAQ computer cluster. It uses the Smith-Waterman local similarity algorithm (see section 7.2.2 for a description of this alignment algorithm) to compare the query sequence versus the Swiss-Prot database. The advantage of this algorithm is that "is recognised as the most sensitive sequence comparison method available, whereas BLAST and FASTA utilise a heuristic one. As a consequence, MPsrch is capable of identifying hits in cases where Blast and Fasta fail and also reports fewer false-positive hits.". It will only run searches for proteins and not for nucleotides due to the time involved but also due to the discreteness of proteins. The speed achieved by MPsrch, is mainly that it is running on a "massively" parallel computer. Because of the use of a parallel computer, it was claimed that "MPsrch is the fastest implementation of the SW algorithm currently available on any machine". Many molecular biology problems lend themselves to parallel architecture computers. For many problems, intermediate steps can be effectively calculated without the need to know results from previous steps. Each of these independent steps can be given to a different processor and solved on its own. Special software has been developed for parallel computers to manage communication among individual processors and to delegate jobs to each one.

The input sequence

```
PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWP
FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHL
DYLQNRVI
```

was given to the website of MPsrch at http://www.ebi.ac.uk/MPsrch/index.html. The input webpage for MPsrch is shown in Figure 6.4. It provides several options that you should explore. Note in particular the database search options. In the example used below I selected the database UNIPROT but for initial explorations you should try UNIREF## databases. These eliminate proteins that are within ## percentage of similarity (where ## is 100, 90 or 50). This will speed an already rapid search. .

MPsrch output

The output generated by this query is ...

```
 ****************************************************************************

        _/_/      _/_/ _/_/_/_/     _/_/_/   _/_/_/_/     _/_/_/   _/_/  _/_/
       _/_/_/  _/_/_/ _/_/  _/_/ _/_/  _/_/ _/_/  _/_/ _/_/  _/_/ _/_/  _/_/
      _/_/_/_/_/_/_/ _/_/  _/_/ _/_/       _/_/  _/_/ _/_/       _/_/  _/_/
     _/_/  _/  _/_/ _/_/_/_/     _/_/_/   _/_/_/_/   _/_/       _/_/_/_/_/
    _/_/      _/_/ _/_/             _/_/ _/_/  _/_/ _/_/       _/_/  _/_/
   _/_/      _/_/ _/_/       _/_/  _/_/ _/_/  _/_/ _/_/  _/_/ _/_/  _/_/
  _/_/      _/_/ _/_/         _/_/_/   _/_/  _/_/   _/_/_/   _/_/  _/_/  (TM)

  ****************************************************************************

                               Aneda Limited.
                        Release 4.2.80 Copyright (c)


MPsrch_pp:   protein-protein database search, using Smith-Waterman algorithm.

Run on:          Tue Aug  9 16:25:37 2005;  Search Time 4.80 secs
                                            1690.21 Million Cell updates/sec

Title:           >Sequence
Description:     No description found
Perfect Score:   1125
Sequence:        1 ptveylnyetlddqgwdmdddd.......adevkivynakhldylqnrvi  128
Scoring table:   PAM 100
                 Gap 14

Searched:        2088752 seqs, 683758041 residues

Database:        uniprot
                      1    uniprot_sprot
                      2    uniprot_trembl1
                      3    uniprot_trembl2

Statistics:      Mean 42.092;  Variance 64.372;  Scale 0.654

      Pred. No. is the number of results predicted by chance to have a
      score greater than or equal to the score of the result being printed,
      and is derived by analysis of the total score distribution.

                                 SUMMARIES
               %
Result         Query
   No.  Score  Match Length DB  ID          Description            Pred. No.
 ---------------------------------------------------------------------------
     1   1125  100.0    128  1  FER_HALSA   Ferredoxin.             3.48e-265
     2    978   86.9    129  2  Q9YGB6_HALJ Ferredoxin.             4.74e-225
     3    967   86.0    128  1  FER1_HALMA  Ferredoxin 1.           4.67e-222
     4    546   48.5    138  1  FER2_HALMA  Ferredoxin 2.           4.52e-109
     5    319   28.4     98  1  FER_SYNP4   Ferredoxin.             8.96e-51
     6    305   27.1     98  1  FER1_ANAVA  Ferredoxin I.           2.59e-47
     7    302   26.8     98  1  FER1_ANASP  Ferredoxin I.           1.42e-46
     8    302   26.8     98  1  FER1_ANASO  Ferredoxin I.           1.42e-46
     9    301   26.8     97  1  FER_SYNEL   Ferredoxin I.           2.50e-46
    10    301   26.8     97  1  FER_SYNEN   Ferredoxin I.           2.50e-46
    11    301   26.8     97  1  FER_SYNVU   Ferredoxin I.           2.50e-46
    12    298   26.5     96  1  FER_SYNLI   Ferredoxin.             1.37e-45
    13    297   26.4     98  1  FER_NOSMU   Ferredoxin.             2.41e-45
    14    294   26.1     98  1  FER2_NOSMU  Ferredoxin II.          1.31e-44
    15    290   25.8     99  1  FER1_PLEBO  Ferredoxin I (FdI).     1.25e-43
    16    290   25.8     99  2  Q7DK20_PLEB Ferredoxin.             1.25e-43
    17    288   25.6     99  3  Q7U8S7_SYNP Ferredoxin.             3.84e-43
    18    286   25.4     99  3  Q7VAM6_PROM Ferredoxin.             1.18e-42
    19    284   25.2     99  3  Q7V0B6_PROM Ferredoxin.             3.64e-42
    20    282   25.1     98  2  Q7M191_SYNS Ferredoxin.             1.12e-41


                                ALIGNMENTS
   (Maximum 20 Alignments, Predicted No. Cut-off is OFF )


RESULT    1
ID   FER_HALSA      STANDARD;      PRT;   128 AA.
DE   Ferredoxin.
```

```
  DB  1;  Score    1125;  Match 100.0%;  QryMatch 100.0%;  Pred. No. 3.48e-265;
  Matches   128;  Conservative    0;  Mismatches   0;  Indels   0;  Gaps   0;

          ************************************************************
Db      1 PTVEYLNYETLDDQGWDMDDDDLFEKAADAGLDGEDYGTMEVAEGEYILEAAEAQGYDWP 60
Qy      1 ptveylnyetlddqgwdmddddlfekaadagldgedygtmevaegeyileaaeaqgydwp 60

          ************************************************************
Db     61 FSCRAGACANCASIVKEGEIDMDMQQILSDEEVEEKDVRLTCIGSPAADEVKIVYNAKHL 120
Qy     61 fscragacancasivkegeidmdmqqilsdeeveekdvrltcigspaadevkivynakhl 120

          ********
Db    121 DYLQNRVI 128
Qy    121 dylqnrvi 128


RESULT    2
ID   Q9YGB6_HALJP PRELIMINARY;      PRT;   129 AA.
DE   Ferredoxin.

  DB  2;  Score     978;  Match 85.2%;  QryMatch 86.9%;  Pred. No. 4.74e-225;
  Matches   109;  Conservative    9;  Mismatches  10;  Indels   0;  Gaps   0;

          ********* .** **** ***.*  *.*  *******..** *****************
Db      2 PTVEYLNYEVVDDNGWDMYDDDVFAEASDMDLDGEDYGSLEVNEGEYILEAAEAQGYDWP 61
Qy      1 ptveylnyetlddqgwdmddddlfekaadagldgedygtmevaegeyileaaeaqgydwp 60

          ************.** **.***************.*.********* *************
Db     62 FSCRAGACANCAAIVLEGDIDMDMQQILSDEEVEDKNVRLTCIGSPDADEVKIVYNAKHL 121
Qy     61 fscragacancasivkegeidmdmqqilsdeeveekdvrltcigspaadevkivynakhl 120

          ********
Db    122 DYLQNRVI 129
Qy    121 dylqnrvi 128


RESULT    3
ID   FER1_HALMA     STANDARD;      PRT;   128 AA.
DE   Ferredoxin 1.

  DB  1;  Score     967;  Match 84.4%;  QryMatch 86.0%;  Pred. No. 4.67e-222;
  Matches   108;  Conservative    9;  Mismatches  11;  Indels   0;  Gaps   0;

          ********* .** **** ***.*  *.*  ** ****..** *****************
Db      1 PTVEYLNYEVVDDNGWDMYDDDVFGEASDMDLDDEDYGSLEVNEGEYILEAAEAQGYDWP 60
Qy      1 ptveylnyetlddqgwdmddddlfekaadagldgedygtmevaegeyileaaeaqgydwp 60

          ************.** **.***************.*.********* *************
Db     61 FSCRAGACANCAAIVLEGDIDMDMQQILSDEEVEDKNVRLTCIGSPDADEVKIVYNAKHL 120
Qy     61 fscragacancasivkegeidmdmqqilsdeeveekdvrltcigspaadevkivynakhl 120

          ********
Db    121 DYLQNRVI 128
Qy    121 dylqnrvi 128


RESULT    4
ID   FER2_HALMA     STANDARD;      PRT;   138 AA.
DE   Ferredoxin 2.

  DB  1;  Score     546;  Match 51.7%;  QryMatch 48.5%;  Pred. No. 4.52e-109;
  Matches    61;  Conservative   27;  Mismatches  29;  Indels   1;  Gaps   1;

          **.**.* *.*.** ..*.*** ***** *.. *.*   *   . .*****  *. ***.
Db      2 VEFLNFEVLEDHGWALQDEDLFAKAADANLQSTDFGRFYVDPNDTLLEAAEKNGFAWPFA 61
Qy      3 veylnyetlddqgwdmddddlfekaadagldgedygtmevaegeyileaaeaqgydwpfs 62

          **.***.***  * .**.     .**.  *. ** .**.**..* .*. ***** ***
Db     62 CRGGACTNCAVAVVDGEMPSPASHILP-PELTEKGIRLSCIAAPVSDDAKIVYNLKHL 118
Qy     63 cragacancasivkegeidmdmqqilsdeeveekdvrltcigspaadevkivynakhl 120


RESULT    5
ID   FER_SYNP4      STANDARD;      PRT;    98 AA.
DE   Ferredoxin.

  DB  1;  Score     319;  Match 53.5%;  QryMatch 28.4%;  Pred. No. 8.96e-51;
  Matches    38;  Conservative   15;  Mismatches  17;  Indels   1;  Gaps   1;
```

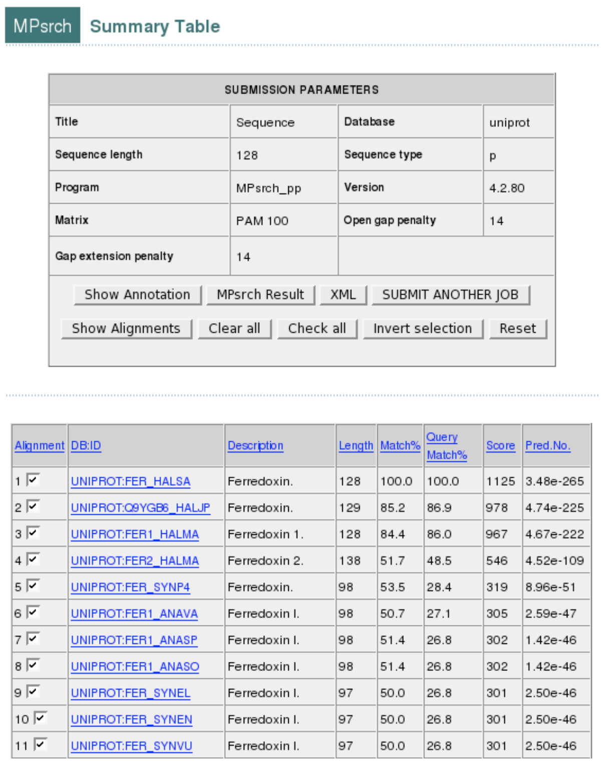
```
          *.**.. ****. ** .* * *.*******. **. .******   *  * *...*   *
Db     17 TIEVPDDEYILDVAEEEGIDLPYSCRAGACSTCAGKIKEGEIDQSDQSFLDDDQIEAGYV 76
Qy     39 tmevaegeyileaaeaqgydwpfscragacancasivkegeidmdmqqilsdeeveekdv 98

           ***.. **.*
Db     77 -LTCVAYPASD 86
Qy     99 rltcigspaad 109


RESULT    6
ID   FER1_ANAVA     STANDARD;      PRT;    98 AA.
DE   Ferredoxin I.

  DB  1;  Score     305;  Match 50.7%;  QryMatch 27.1%;  Pred. No. 2.59e-47;
  Matches    38;  Conservative   16;  Mismatches  19;  Indels   2;  Gaps   2;

          *..*.. ****.*** **** *********. **. .  * .*   *  * *...*   *
Db     17 TIDVPDDEYILDAAEEQGYDLPFSCRAGACSTCAGKLVSGTVDQSDQSFLDDDQIEAGYV 76
Qy     39 tmevaegeyileaaeaqgydwpfscragacancasivkegeidmdmqqilsdeeveekdv 98

           ***.. *..* * *
Db     77 -LTCVAYPTSD-VTI 89
Qy     99 rltcigspaadevki 113


......................................................................
.......................  Material Deleted  ...........................
......................................................................


RESULT   20
ID   Q7M191_SYNSP PRELIMINARY;      PRT;    98 AA.
DE   Ferredoxin.

  DB  2;  Score     282;  Match 47.9%;  QryMatch 25.1%;  Pred. No. 1.12e-41;
  Matches    34;  Conservative   17;  Mismatches  19;  Indels   1;  Gaps   1;

          *.**.. .***.*** ** * *.*******. **. .  * .*   *  * *...*   *
Db     17 TIEVPDDQYILDAAEEQGIDLPYSCRAGACSTCAGKLTSGTVDQSDQSFLDDDQIEAGFV 76
Qy     39 tmevaegeyileaaeaqgydwpfscragacancasivkegeidmdmqqilsdeeveekdv 98

           ***.. *..*
Db     77 -LTCVAYPTSD 86
Qy     99 rltcigspaad 109

  Search Completed:  Tue Aug  9 16:27:18 2005
Job time:  101 seconds
```

MPsrch format

This particular search took only 5 seconds of CPU time and a total of 101 seconds including input/output. This speed is a great improvement over that achieved by the FASTA algorithm. The web page output is shown in Figure 6.5. . This algorithm is as fast as BLASTP and in addition, it should also give a more sensitive search for distant homologies.

The mean and variance of the distribution of scores from the entire database are calculated. These are used to construct empirical statistics of the predicted number of random matches in the database equal to or better than that found. The algorithm then lists the best scores (50 of them here, the default for NAMES) and then lists more detailed reports for a subclass of these (30 here, the default for ALIGN). For each it calculates the raw score, the percent matches, the predicted number expected, the number of matches, the number of mismatches, the number of partial matches (residue pairs with a positive score in the PAM matrix), the number of indels and the number of gaps. This program considers these two differently in that a single gap can be composed of any number of adjacent indels.

In this case all hits have very small “pred. no.” numbers indicating that they each have statistically significant homology to the ferredoxin query sequence (not too surprising since they are all different ferredoxins). Also note that the Smith-Waterman alignment algorithm does a best local alignment (more on this later) so the entire query sequence may not be presented in the output. Rather the part of the sequence that has a good alignment with the database entry is shown. The sequence is not aligned for regions where the significance of the alignment begins to decline. Hence in the example above, for the alignment to result #20, only amino acids 17 through 86 from the database sequence and amino acids 39 to 109 from



SUBMISSION PARAMETERS

Title	Sequence	Database	uniprot
Sequence length	128	Sequence type	p
Program	MPsrch_pp	Version	4.2.80
Matrix	PAM 100	Open gap penalty	14
Gap extension penalty	14		

Alignment	DB:ID	Description	Length	Match%	Query Match%	Score	Pred.No.
1 ✓	UNIPROT:FER_HALSA	Ferredoxin.	128	100.0	100.0	1125	3.48e-265
2 ✓	UNIPROT:Q9YGB6_HALJP	Ferredoxin.	129	85.2	86.9	978	4.74e-225
3 ✓	UNIPROT:FER1_HALMA	Ferredoxin 1.	128	84.4	86.0	967	4.67e-222
4 ✓	UNIPROT:FER2_HALMA	Ferredoxin 2.	138	51.7	48.5	546	4.52e-109
5 ✓	UNIPROT:FER_SYNP4	Ferredoxin.	98	53.5	28.4	319	8.96e-51
6 ✓	UNIPROT:FER1_ANAVA	Ferredoxin I.	98	50.7	27.1	305	2.59e-47
7 ✓	UNIPROT:FER1_ANASP	Ferredoxin I.	98	51.4	26.8	302	1.42e-46
8 ✓	UNIPROT:FER1_ANASO	Ferredoxin I.	98	51.4	26.8	302	1.42e-46
9 ✓	UNIPROT:FER_SYNEL	Ferredoxin I.	97	50.0	26.8	301	2.50e-46
10 ✓	UNIPROT:FER_SYNEN	Ferredoxin I.	97	50.0	26.8	301	2.50e-46
11 ✓	UNIPROT:FER_SYNVU	Ferredoxin I.	97	50.0	26.8	301	2.50e-46

Figure 6.5: The webpage output from an MPsrch

the query sequence are shown in the alignment, even though the query protein is 128 amino acids in length. The sequence prior to amino acid 17/39 and after amino acid 86/109 are not considered to be part of the significant local alignment.

6.2 BLOCKS

The FASTA and BLAST servers are often searched for homologues in order to identify the query sequence. The BLOCKS server at http://blocks.fhcrc.org is designed to identify chunks of a protein that my encode some function. The BLOCKS server is thus somewhat related to the other servers mentioned above (and hence included here) but is designed to answer a different question. Instead of looking for similar sequences in the databases, it scans a database of protein motif signatures constructed from the INTERPRO database (a collection of of protein families, domains and functional sites found in known proteins that can be applied to explore unknown protein sequences). In this way, `BLOCKS` will search a query sequence (must be protein or optionally, it will translate your nucleotide sequence to a protein) for similar protein motifs in known proteins. Blocks are defined as short ungapped (but potentially with variable length) segments of highly conserved regions of proteins. As of August 2003 the `BLOCKS` database website reports that it consists of 8656 block patterns (version 13.0, Aug 2001). This search is particularly useful for analysing distantly related proteins.

The web form to search the `BLOCKS` database is located at http://blocks.fhcrc.org/blocks/blocks_search.html (References should cite S.Henikoff & J.Henikoff, 1991 Nucl.Acids.Res. 19:6565-6572). Again simply supply the web page with your query sequence.

Since this search only makes sense for proteins, if a nucleotide sequence is supplied to the server, it will be translated in all frames. But a nucleotide sequence with IUBPAC ambiguity codes will be interpreted as a protein and will remain untranslated.

6.2.1 BLOCKS output

In the example below, I have searched the `BLOCKS` database with the sequence

```
> Ferredoxin
GIDPNYRTHKPVVGDSSGHKIYGPVESPKVLGVHGTIVGVDFDLCIADGSCITACPVNVF
QWYETPGHPASEKKADPVNQQACIFCMACVNVCPVAAIDVKPP
```

The BLOCKS output begins with a lengthy informational message that I have deleted and then continues with the guts of the message.

```
Hits


Query=Ferredoxin n
Size=103 Amino Acids
Blocks Searched=11182
Alignments Done=         1439343
Cutoff combined expected value for hits=  1
Cutoff block expected value for repeats/other=  1
==============================================================================
                                                             Combined
Family                                       Strand  Blocks   E-value
PR00353    4Fe-4S ferredoxin signature           1   2 of 2   1.2e-06
PR00354    7Fe ferredoxin signature              1   1 of 3   0.00025
IPB000985  Legume lectin alpha domain            1   1 of 7      0.58

==============================================================================
>PR00353 2/2 blocks Combined E-value= 1.2e-06: 4Fe-4S ferredoxin signature
Block    Frame    Location (aa)      Block E-value
PR00353A    0         76-87                 4.5
PR00353B    0         88-99             0.00014
Other reported alignments:

                         |---  151 amino acids---|
           PR00353 AA........................................................BB
        Ferredoxin                                          ::::::::::::AABB

PR00353A            <->A   (1,571):75
AEGA_ECOLI|P37127  80      IQVNQQKCIGCK
                             |||| || |
Ferredoxin         76      DPVNQQACIFCM

PR00353B           A<->B   (0,338):0
FER_CLOSP|P00197   42      ACANTCPVDAIV
                           || | ||| ||
Ferredoxin         88      ACVNVCPVAAID

------------------------------------------------------------------------------
>PR00354 1/3 blocks Combined E-value= 0.00025: 7Fe ferredoxin signature
Block    Frame    Location (aa)      Block E-value
PR00354C    0         78-95             0.00027
Other reported alignments:
PR00354C    0         40-57              0.0018

                         |---   57 amino acids---|
           PR00354 AAAAA.........................BBBBB::................CCCCCCCC
        Ferredoxin                     :::::::::::::::::::::::::::::::::CCCCCCCC
        Ferredoxin                                      CCCCCCCC

PR00354C            <->C   (34,389):77
FER_BACSC|Q45560   34      IDPDVCIDCGACEAVCPV
                                || | ||  ||||
Ferredoxin         78      vNqqaCIfCmACVnVCPV
                   40      vDfDLCIadGsCitACPV

------------------------------------------------------------------------------
>IPB000985 1/7 blocks Combined E-value=    0.58: Legume lectin alpha domain
Block    Frame    Location (aa)      Block E-value
IPB000985D  0         36-45                0.67
Other reported alignments:

                         |---  126 amino acids---|
         IPB000985 AAAA:...BBBBB::..CC:......DD........EEEEEE:........FFFF:..GG
        Ferredoxin                    :::::::DD

IPB000985D          <->D   (83,186):35
```

```
LECN_PEA|P16270    145      RFVGLEFDLY
                              ||  |||
Ferredoxin         36       TIVGVDFDLC

------------------------------------------------------------------------------

3 possible hits reported
```

In this case, for ferredoxin, the program returns three possible hits. These are a 4Fe ferredoxin, a 7Fe Ferredoxin and a legume lectin alpha domain. The first signature consists of two parts (two blocks), the signature for the second hit consists of three parts (but only one was found in the query sequence) and the signature for the third hit consists of seven parts (but again only one is present in the query sequence). Each of these blocks is labelled A, B, C, etc. The E-values are calculated (as per the BLAST searches) to represent the expected number of hits with as good a similarity or better in a database of this size. Hence the last hit to a legume lection alpha domain is probably just noise.

After this initial presentation, the program returns a diagram of hits. So in the first hit, the first block (A), can typically begin anywhere from the 1st to the 571st amino acid (in bone-fide proteins with this signature). In our query it begins at position 75. The second block (B), can occur anywhere from 0 to 338 amino acids distant from the first block. In our query sequence it is 0 amino acids away. Alignments of each of these blocks to a best match is shown.

For the second hit, the query contains two possible locations for the "C" block but non of the other blocks. For the third hit, only the "D" block.

6.2.2 Getting the Block

In addition to this the BLOCKS server will allow you access to information about the individual blocks found. You can get the entry either via links on their web page. The following output is are examples from their links.

From the BLOCKS database itself, it has the following information on the block.

```
          Prints Database 37 in Blocks Format, Jun 2003
        Made available by the Fred Hutchinson Cancer Research Center
      1100 Fairview AV N, A1-162, PO Box 19024, Seattle, WA 98109-1024
   Based on PRINTS Database as described by TK Attwood, et al (1994),
   NAR 22(17):3590-3596. ID is from PRINTS gc line, AC is from
   PRINTS gx line, DE is from PRINTS gt line, BL is BLOCK information.
   Each PRINTS motif is represented by one block. For each segment, the
   sequence ID is followed by the position of the first residue in the
   segment. Sequence weights are shown to the right of each segment. The
   higher the weight (maximum 100) the more dissimilar the segment is from
   other segments in the block. These weights were obtained using the
   position-based method of S Henikoff & JG Henikoff (1994), JMB 243:574-578.
   Calibrated with position-specific scoring matrices made with pseudo-counts,
   JG Henikoff & S Henikoff (1996), CABIOS 12(2):135-143.
   ==========================================================================


Block PR00353A

ID 4FE4SFRDOXIN; BLOCK
AC PR00353A; distance from previous block=(1,571)
DE 4Fe-4S ferredoxin signature
BL adapted; width=12; seqs=171; 99.5%=733; strength=1118
P81293             ( 275) YVIDEDLCIGCR  17
FER_CLOSP|P00197   (  30) RVIDADKCIDCG  21
O27769             (  62) VVILEDRCIGCG  41
O28894             ( 233) TYVDWDKCIGCG  30
FER_CLOAC|P00198   (  30) YVIDADTCIDCG  15
FER_BACSC|Q45560   (  32) YYIDPDVCIDCG  26
Q59575             ( 147) IEIDKDTCIYCG  18
FER2_DESDN|P00211  (   5) VIVDSDKCIGCG  21
O30081             (   6) IAIDEEKCIGCG  18
O74028             ( 147) IEIDKDTCIYCG  18
FDXH_HAEIN|P44450  ( 132) VDFQSDKCIGCG  55
O26505             ( 164) AVVDESICIGCG  26
```

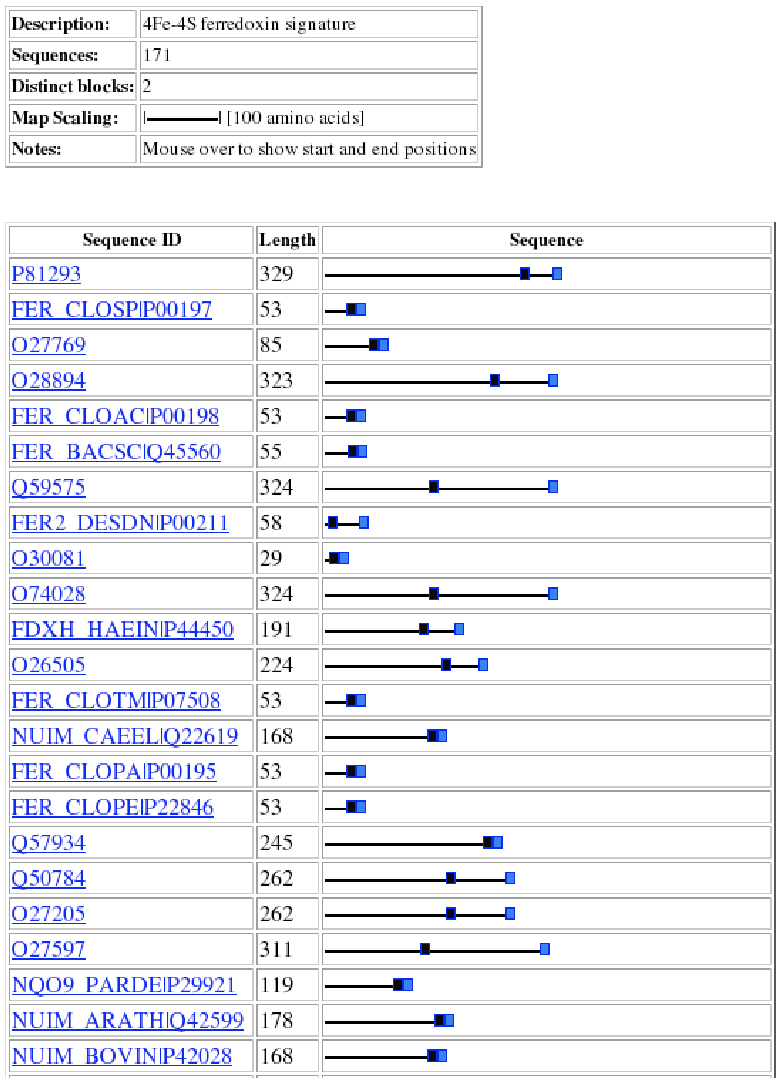
```
FER_CLOTM|P07508   (  30) YVIDADACIECG  40
NUIM_CAEEL|Q22619  ( 145) YDIDMTKCIYCG  18

......................................................................
........................  Material Deleted ..........................
......................................................................

O29066             (   9) FVHDRRKCIGCY  81
Q03195             (  48) AFISEILCIGCG  65
FER1_RHOCA|P16021  (   2) MKIDPELCTSCG  48
O28624             (  73) LIVDESLCVGCG  20
P73811             (  77) IVIDDQSCVDCG  41
Q46606             ( 145) VVRDMGKCIRCL  78
Y719_METJA|Q58129  (  55) PVISEVLCSGCG  63
O28573             (  62) AVVNYNYCKGCG  28
O27592             ( 556) YMIDPEKCDGCM  92
P74022             ( 141) FGIDHNRCILCT  59
//

Block PR00353B

ID   4FE4SFRDOXIN; BLOCK
AC   PR00353B; distance from previous block=(0,338)
DE   4Fe-4S ferredoxin signature
BL   adapted;   width=12; seqs=171; 99.5%=728; strength=1179
P81293             ( 318) ACARECPVGAIK  11
FER_CLOSP|P00197   (  42) ACANTCPVDAIV  11
O27769             (  74) LCRDACPVGAIT  17
O28894             ( 312) PCEKACPTGAIN  13
FER_CLOAC|P00198   (  42) ACAGVCPVDAPV  15
FER_BACSC|Q45560   (  44) ACEAVCPVSAIY  17
Q59575             ( 313) ACERSCPVNAIE  11
FER2_DESDN|P00211  (  47) SCIEVCPQNAIV  20
O30081             (  18) RCVNSCPTGALV  16
O74028             ( 313) ACERSCPVTAIT  21
FDXH_HAEIN|P44450  ( 180) ACVKTCPTGAIR  12
O26505             ( 213) VCEENCPTGAIR  17
FER_CLOTM|P07508   (  42) ACANVCPVDAPQ  14

......................................................................
........................  Material Deleted ..........................
......................................................................

FER1_RHOCA|P16021  (  14) DCEPVCPTNAIA  29
O28624             ( 141) VCRENCPSDAIR  26
P73811             (  89) LCTGVCPTEALS  24
Q46606             ( 200) QCTLVCPVGALA  30
Y719_METJA|Q58129  (  67) ICVKRCPFKAIS  20
O28573             (  74) ICASVCPFEAIK  14
O27592             ( 568) ACIKTCPAEAIN  18
P74022             ( 197) KCVDACPTGSIF 100
//
```

This provides a short description of the parts of each block and then representative sequences that contain these blocks (with a links to that sequence, the position of the first residue in the block, the block and a weighting score). This information can be seen in graphical format as shown in Figure 6.6.

In addition you can get more data about the blocks through the PROSITE database link for this entry

```
PROSITE: PS00198
ID   4FE4S_FERREDOXIN; PATTERN.
AC   PS00198;
DT   APR-1990 (CREATED); APR-1990 (DATA UPDATE); JUL-1998 (INFO UPDATE).
DE   4Fe-4S ferredoxins, iron-sulfur binding region signature.
PA   C-x(2)-C-x(2)-C-x(3)-C-[PEG].
NR   /RELEASE=41.21,133312;
NR   /TOTAL=523(348); /POSITIVE=482(318); /UNKNOWN=2(2); /FALSE_POS=39(28);
NR   /FALSE_NEG=16; /PARTIAL=5;
CC   /TAXO-RANGE=A?EP?; /MAX-REPEAT=6;
CC   /SITE=1,iron_sulfur; /SITE=3,iron_sulfur; /SITE=5,iron_sulfur;
CC   /SITE=7,iron_sulfur;
DR   P37127, AEGA_ECOLI, T; P26474, ASRA_SALTY, T; P26476, ASRC_SALTY, T;
DR   P31894, COOF_RHORU, T; Q49161, DCA1_METMA, T; Q49163, DCA2_METMA, T;
DR   Q57617, DCMA_METJA, T; P26692, DCMA_METSO, T; O27743, DCMA_METTH, T;
DR   P08066, DHSB_BACSU, T; Q09545, DHSB_CAEEL, T; P48932, DHSB_CHOCR, T;
DR   P51053, DHSB_COXBU, T; P48933, DHSB_CYACA, T; P21914, DHSB_DROME, T;
DR   P07014, DHSB_ECOLI, T; P21912, DHSB_HUMAN, T; O42772, DHSB_MYCGR, T;
```

Figure 6.6: A map of the BLOCKS location in representative proteins



Description:	4Fe-4S ferredoxin signature
Sequences:	171
Distinct blocks:	2
Map Scaling:	[100 amino acids]
Notes:	Mouse over to show start and end positions

Sequence ID	Length	Sequence
P81293	329	
FER_CLOSP|P00197	53	
O27769	85	
O28894	323	
FER_CLOAC|P00198	53	
FER_BACSC|Q45560	55	
Q59575	324	
FER2_DESDN|P00211	58	
O30081	29	
O74028	324	
FDXH_HAEIN|P44450	191	
O26505	224	
FER_CLOTM|P07508	53	
NUIM_CAEEL|Q22619	168	
FER_CLOPA|P00195	53	
FER_CLOPE|P22846	53	
Q57934	245	
Q50784	262	
O27205	262	
O27597	311	
NQO9_PARDE|P29921	119	
NUIM_ARATH|Q42599	178	
NUIM_BOVIN|P42028	168	

```
DR   Q59662, DHSB_PARDE, T; P80477, DHSB_PORPU, T; P21913, DHSB_RAT  , T;
DR   P80480, DHSB_RECAM, T; Q92JJ8, DHSB_RICCN, T; Q9ZEA1, DHSB_RICPR, T;
DR   Q8ZQU2, DHSB_SALTY, T; P21911, DHSB_SCHPO, T; P32420, DHSB_USTMA, T;

.......................................................................
........................  Material Deleted ............................
.......................................................................

DR   Q01642, M84A_DROME, F; Q01643, M84B_DROME, F; Q01644, M84C_DROME, F;
DR   Q01645, M84D_DROME, F; P08175, M87F_DROME, F; P55952, MT_POTPO  , F;
DR   O28002, RPOD_ARCFU, F; Q8PV16, RPOD_METMA, F; O26144, RPOD_METTH, F;
DR   Q96YW0, RPOD_SULTO, F; P23327, SRCH_HUMAN, F; P16230, SRCH_RABIT, F;
DR   P45866, YWJF_BACSU, F;
3D   1A6L; 1AXQ; 1B0T; 1B0V; 1BC6; 1BD6; 1BLU; 1BQX; 1BWE; 1C4A; 1C4C; 1CLF;
3D   1D3W; 1DUR; 1DWL; 1E7P; 1F2G; 1F5B; 1F5C; 1FCA; 1FD2; 1FDA; 1FDB; 1FDD;
3D   1FDN; 1FEH; 1FER; 1FRH; 1FRI; 1FRJ; 1FRK; 1FRL; 1FRM; 1FRX; 1FTC; 1FXD;
3D   1G3O; 1G6B; 1GAO; 1GT8; 1GTE; 1GTH; 1H7W; 1H7X; 1H98; 1HFE; 1JB0; 1K0T;
3D   1KF6; 1KFY; 1KQF; 1KQG; 1L0V; 1NEK; 1QLA; 1QLB; 1ROF; 1VJW; 1XER; 2FD2;
3D   2FDN; 5FD1; 6FD1; 6FDR; 7FD1; 7FDR;
DO   PDOC00176;
//
```

NiceSite View of PROSITE: PDOC00176 (documentation)
4Fe-4S ferredoxins, iron-sulfur binding region signature
PROSITE cross-reference(s)
PS00198; 4FE4S_FERREDOXIN
Documentation

Ferredoxins [1] are a group of iron-sulfur proteins which mediate electron transfer in a wide variety of metabolic reactions. Ferredoxins can be divided into several subgroups depending upon the physiological nature of the iron-sulfur cluster(s). One of these subgroups are the 4Fe-4S ferredoxins, which are found in bacteria and which are thus often referred as 'bacterial-type' ferredoxins. The structure of these proteins [2] consists of the duplication of a domain of twenty six amino acid residues; each of these domains contains four cysteine residues that bind to a 4Fe-4S center.

A number of proteins have been found [3] that include one or more 4Fe-4S binding domains similar to those of bacterial-type ferredoxins. These proteins are listed below (references are only provided for recently determined sequences).

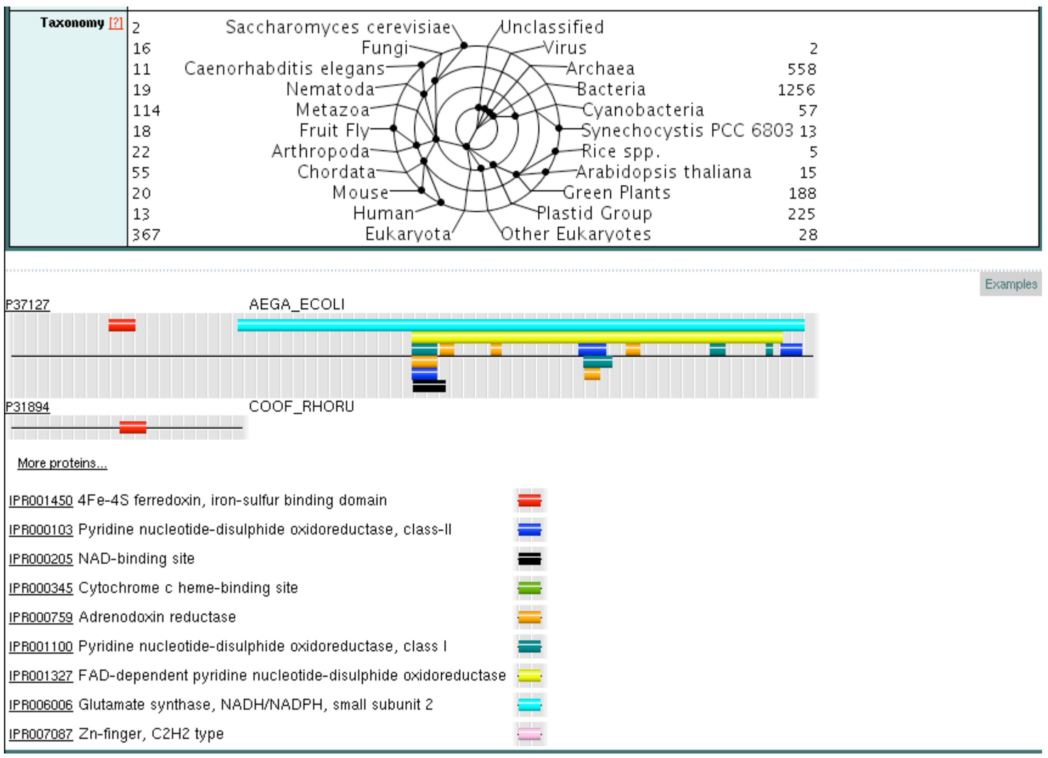
- The iron-sulfur proteins of the succinate dehydrogenase and the fumarate reductase complexes (EC 1.3.99.1). These enzyme complexes, which are components of the tricarboxylic acid cycle, each contain three subunits: a flavoprotein, an iron-sulfur protein, and a b-type cytochrome. The iron-sulfur proteins contain three different iron-sulfur centers: a 2Fe-2S, a 3Fe-3S and a 4Fe-4S.
- Escherichia coli anaerobic glycerol-3-phosphate dehydrogenase (EC 1.1.99.5) This enzyme is composed of three subunits: A, B, and C. The C subunit seems to be an iron-sulfur protein with two ferredoxin-like domains in the N-terminal part of the protein.
- Escherichia coli anaerobic dimethyl sulfoxide reductase. The B subunit of this enzyme (gene dmsB) is an iron-sulfur protein with four 4Fe-4S ferredoxin-like domains.
- Escherichia coli formate hydrogenlyase. Two of the subunits of this oligomeric complex (genes hycB and hycF) seem to be iron-sulfur proteins that each contain two 4Fe-4S ferredoxin-like domains.
- Methanobacterium formicicum formate dehydrogenase (EC 1.2.1.2). This enzyme is used by the archaebacteria to grow on formate. The beta chain of this dimeric enzyme probably binds two 4Fe-4S centers.
- Escherichia coli formate dehydrogenases N and O (EC 1.2.1.2). The beta chain of these two enzymes (genes fdnH and fdoH) are iron-sulfur proteins with four 4Fe-4S ferredoxin-like domains.
- Desulfovibrio periplasmic [Fe] hydrogenase (EC 1.18.99.1). The large chain of this dimeric enzyme binds three 4Fe-4S centers, two of which are located in the ferredoxin-like N-terminal region of the protein.
- Methanobacterium thermoautrophicum methyl viologen-reducing hydrogenase subunit mvhB, which contains six tandemly repeated ferredoxin-like domains and which probably binds twelve 4Fe-4S centers.
- Salmonella typhimurium anaerobic sulfite reductase (EC 1.8.1.-) [4]. Two of the subunits of this enzyme (genes asrA and asrC) seem to both bind two 4Fe-4S centers.
- A Ferredoxin-like protein (gene fixX) from the nitrogen-fixation genes locus of various Rhizobium species, and one from the Nif-region of Azotobacter species.
- The 9 Kd polypeptide of chloroplast photosystem I [5] (gene psaC). This protein contains two low potential 4Fe-4S centers, referred as the A and B

```
   centers.
 - The chloroplast frxB protein which is predicted to carry two 4Fe-4S centers.
 - An ferredoxin  from a  primitive  eukaryote,  the enteric amoeba  Entamobea
   histolytica.
 - Escherichia  coli  hypothetical  protein  yjjW, a protein with a N-terminal
   region belonging to the radical activating enzymes family (see <PDOC00834>)
   and two potential 4Fe-4S centers.

The pattern of cysteine  residues in the  iron-sulfur region  is sufficient to
detect this class of 4Fe-4S binding proteins.


Description of pattern(s) and/or profile(s)
Consensus pattern
        C-x(2)-C-x(2)-C-x(3)-C-[PEG] [The four C's are 4Fe-4S ligands]
Sequences known to belong to this class detected by the pattern
        the majority of known 4Fe-4S sequences, with very few exceptions.
Other sequence(s) detected in Swiss-Prot  24.
Note in some bacterial ferredoxins, one of the two duplicated domains
        has lost one or more of the four conserved cysteines. The
        consequence of such variations is that these domains have either
        lost their iron-sulfur binding property or bind to a 3Fe-3S
        center instead of a 4Fe-4S center.
Note the last residue of this pattern in most proteins belonging
        to this group, is a Pro; the only exceptions are the Rhizobium
        ferredoxin-like proteins which have Gly, and two Desulfovibrio
        ferredoxins which have Glu. It must also be noted that the three
        non 4Fe-4S-binding proteins which are picked-up by the pattern
        have Gly in this position of the pattern.
Last update
November 1995 / Text revised.
References
[ 1]
Meyer J.
Trends Ecol. Evol. 3:222-226(1988).
[ 2]
Otaka E., Ooi T.
J. Mol. Evol. 26:257-267(1987).
[ 3]
Beinert H.
FASEB J. 4:2483-2492(1990).
[ 4]
Huang C.J., Barrett E.L.
J. Bacteriol. 173:1544-1553(1991).
[ 5]
Knaff D.B.
Trends Biochem. Sci. 13:460-461(1988).
```

This is probably more information about ferredoxin than you would ever want. But should you desire more there are links to the INTERPRO entry for this domain. In this case it is (in part)

```
InterPro 4Fe-4S ferredoxin, iron-sulfur binding domain [?] = help
IPR001450
4Fe4S_ferredoxin  Matches: 2183 proteins
View matches: [Overview][...sorted by Name][of known structure][Detailed view][Table view]
Name [?] 4Fe-4S ferredoxin, iron-sulfur binding domain
Signatures [?] PF00037;fer4 (1903 proteins)
PR00353;4FE4SFRDOXIN (3 proteins)
PS00198;4FE4S_FERREDOXIN (2111 proteins)
Type [?] Domain
Dates [?] 1999-10-08 17:07:25.0 (created)
2000-06-29 10:12:25.0 (modified)
Found in [?] IPR000813; 7Fe ferredoxin
IPR001080; 3Fe-4S ferredoxin
IPR004452; Iron-sulfur cluster binding protein
IPR004453; 4Fe-4S cluster binding
IPR004460; CO dehydrogenase/acetyl-CoA synthase complex alpha subunit
IPR004489; Succinate dehydrogenase/fumarate reductase iron-sulfur protein
IPR004494; MauM/NapG ferredoxin-type protein
IPR004496; Ferredoxin-type protein NapF
IPR004497; NADH-plastoquinone oxidoreductase, subunit I
IPR006470; Formate dehydrogenase, beta subunit
IPR006547; Nitrate reductase, beta subunit
Process [?] electron transport (GO:0006118)
Function [?] electron transporter activity (GO:0005489)
Abstract [?]

Ferredoxins are iron-sulphur proteins that mediate electron transfer
```

Figure 6.7: Typical results from an INTERPRO query



IPR001450 4Fe-4S ferredoxin, iron-sulfur binding domain	🟥
IPR000103 Pyridine nucleotide-disulphide oxidoreductase, class-II	🟦
IPR000205 NAD-binding site	⬛
IPR000345 Cytochrome c heme-binding site	🟩
IPR000759 Adrenodoxin reductase	🟧
IPR001100 Pyridine nucleotide-disulphide oxidoreductase, class I	🟦
IPR001327 FAD-dependent pyridine nucleotide-disulphide oxidoreductase	🟨
IPR006006 Glutamate synthase, NADH/NADPH, small subunit 2	🟦
IPR007087 Zn-finger, C2H2 type	🟪

```
in a range of metabolic reactions; they fall into several subgroups
according to the nature of their iron-sulphur cluster(s) [1, 2]. One
group, originally found in bacteria, has been termed "bacterial-type",
in which the active centre is a 4Fe-4S cluster. 4Fe-4S ferredoxins
may in turn be subdivided into further groups, based on their sequence
properties. Most contain at least one conserved domain, including four
Cys residues that bind to a 4Fe-4S centre.

During the evolution of bacterial-type ferredoxins, intrasequence gene
duplication, transposition and fusion events occured, resulting in the
appearance of proteins with multiple iron-sulphur centres: e.g. dicluster-
type (2[4Fe-4S]) and polyferredoxins, iron-sulphur subunits of bacterial
succinate dehydrogenase/fumarate reductase, formate hydrogenlyase and
formate dehydrogenase complexes, pyruvate-flavodoxin oxidoreductase,
NADH:ubiquinone reductase and others. In some bacterial ferredoxins,
one of the duplicated domains has lost one or more of the four conserved
Cys residues. These domains have either lost their iron-sulphur binding
property, or bind to a 3Fe-4S centre instead of a 4Fe-4S centre. 3D
structures are now known both for a number of monocluster-type [3]
and dicluster-type [4] 4Fe-4S ferredoxins.

CAUTION: PRINTS signature in the current entry is known to miss protein
matches and should be updated in the near future.
```

There is even a link to give a graphical interpretation of the block's taxonomic diversity and graphical demonstrations of the block's location within proteins as shown in Figure 6.7.

A really great resource.

6.3 SSearch

At the extreme slow end of database searchers is SSEARCH. This does a universal sequence comparison using the Smith-Waterman algorithm (T.F. Smith and M.S. Waterman, J.Mol.Biol. 147:195-197, 1981). That is, it is completely rigorous comparison of each sequence with the query sequence. This program uses code developed by X. Huang, R.C. Hardison, W. Miller (1990 CABIOS 6:373-381) for calculating the local similarity score and code from the ALIGN program (see below) for calculating the local alignment. SSEARCH is about 100-times slower than FASTA with ktup=2 (for proteins). The program itself is available for download as part of the FASTA package of programs.

A study by Pearson (1995 Protein Science 4:1145-1160) compared the different methods of searching the protein databases. He found that the complete Smith-Waterman algorithm performed best to find distantly related homologies, followed by FASTA and then `blastp` when using suitable scoring matrices (BLOSUM55 – more on these later) and optimal gap penalties.

6.4 Why you should routinely check your sequence

The following is an example of why you should routinely do a search (FASTA, BLAST or whatever) for any new sequence that you are working on. This is a copy of a letter to the editor of NATURE.

Fact and fiction in alignment.
NATURE 358:271, 1992

Sir - We have discovered a startling similarity between a dinosaur DNA sequence reported in the novel Jurassic Park[1] *and a partial human brain cDNA sequence from the Venter laboratory described in Nature*[2] *(see figure).*

The dinosaur sequence (DINO1) consists of duplication, with 117 base pairs from the first member of the repeat aligning with the human sequence, HUMXT01431, at the 95 per cent level of identity with only two gaps. The extraordinary degree of nucleotide sequence conservation between organisms as distantly related as dinosaur and human suggests strongly conserved function. Expression of HUMXT01431 in human brain raises the possibility that the dinosaurs were smarter than has been supposed, arguing against the hypothesis that their extinction resulted from lack of intelligence.

Our discovery also seems to raise the interesting legal question as to whether the copyright on Jurassic Park takes precedence over the pending patent on the human sequence. However, it appears that neither group is entitled to legal protection for its sequence, because both sequences also align with cloning vector pBR322, raising the possibility that both groups inadvertently sequenced vector DNA.

Alan C. Christensen, *Dept of Biochemistry and Molecular Biology, Thomas Jefferson University, Philadelphia, Pennsylvania, 19107 USA.*

Steven Henikoff, *Howard Hughes Medical Institute and Basic Sciences Division, Fred Hutchinson Cancer Research Center, Seattle Washington 98104 USA.*

1 Crichton, M. Jurassic Park, 102 (Ballantine, New York 1990).

2 Adams, M.D. et al., Nature 355, 632-634 (1992).

```
HUMXT 317 GCGTTGCTGGCGTTTTTCCATAGGCTCCGACCCCCTGACGAGCATCACAAAAATCGACGCTCAA
          ***************************** ******************************
DINO1   1 GCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCCTGACGAGCATCACAAAAATCGACGC----
          *************************************************** *
DINO1 670 GCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCCTGACGAGCATCACAAACAAGTCAGA----


HUMXT 234 GTCANAGGTGGCGGAAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTTGGAGCTTCC
                ******* **************************************** * **** **
DINO1  61 ------GGTGGCG-AAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTGGAAGCTCCC
                ******* ********************************************** * *
DINO1 730 ------GGTGGCG-AAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTGGAAGCGCTC
```

With such good jokers in the world as these gentlemen are, you don’t want to get caught by them.

Chapter 7

Sequence Alignment

7.1 Dot Plots

The comparison of sequences can be done in many different ways. The most direct method is to make this comparison via a visual means and this is what “dot plots” attempt to do. Dot plots are a group of methods that visually compare two sequences and look for regions of close similarity between them.

7.1.1 The Exact Way

The sequences to be compared are arranged along the margins of a matrix. At every point in the matrix where the two sequences are identical a dot is placed (i.e. at the intersection of every row and column that have the same letter in both sequences). A diagonal stretch of dots will indicate regions where the two sequences are similar. Done in this fashion a dot plot as shown in Figure 7.1 will be obtained. This is a dot plot of the globin intergenic region in chimpanzees plotted against itself (bases 1 to 400 vs. 1 to 300) The solid line on the main diagonal is a reflection that every base of the sequence is trivially identical to itself. As can be seen this dot plot is not very useful unless applied to protein sequences (where the background is much less dense), however some statistical methods can still be applied to the results (Gibbs and McIntyre 1970, Eur. J. Biochem. 16:1).

Maizel and Lenk (1981, PNAS 78:7665) popularized the dot plot and suggested the use of a filter to reduce the noise demonstrated in Figure 7.1. This noise is caused by matches that have occurred by chance. Because only four different nucleotides are possible, nucleotides will match other nucleotides elsewhere in the sequence without any homology present and hence are not a true reflection of the similarities between the sequences but rather reflect the limited number of bases permitted in DNA sequences. There are a wide variety of filters that can be used, indeed they are only limited by your imagination. The one suggested by Maizel and Lenk was to place a dot only when a specified proportion of a small group of successive bases match. In Figure 7.2 the same dot plot is reproduced with a filter such that a window of 10 bases is highlighted only if 6 of these 10 bases match. In Figure 7.3 the same plot is again shown with a filter of 8 out of 10 matches. Note that these plots highlight the complete window while other programs might highlight a single point centered by the window. Another common way to filter the matches is to give them a weight according to their chemical similarity (Staden 1982, Nuc. Acids Res. 10:2951).

The computational work involved with the generation of these matrices can be quite time consuming. If you are comparing a sequence of length N with another sequence of length M, then the total number of windows for which matches must be calculated is $N \times M$. Hence the amount of work increases with the square of the sequence length. This rapidly becomes a large number. For example with $N = 700$ and $M = 400$, $N \times M = 280,000$.

Figure 7.1: Dot Blot - without filtering.

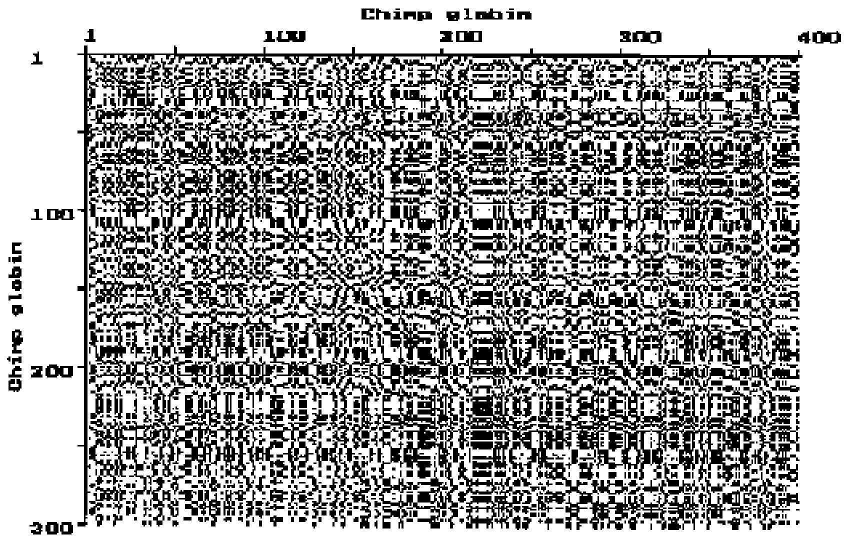


Figure 7.2: Dot Blot - filtered 6 of 10.

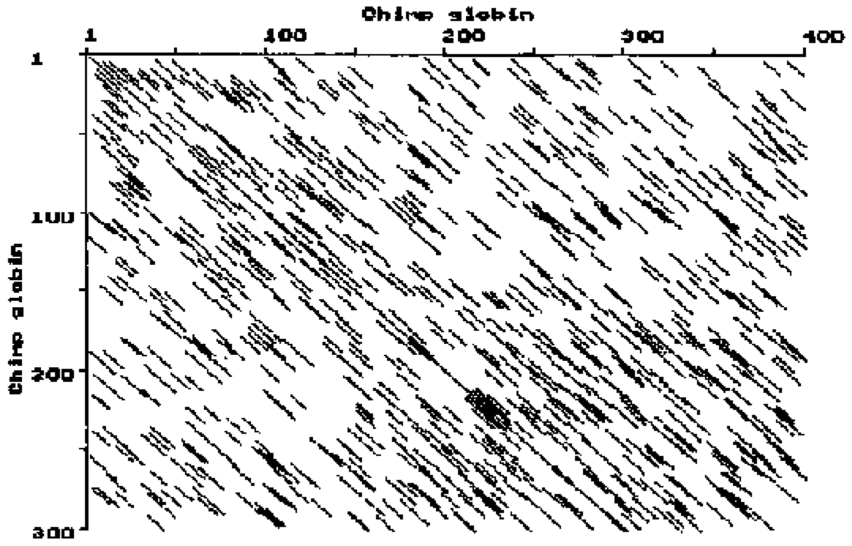
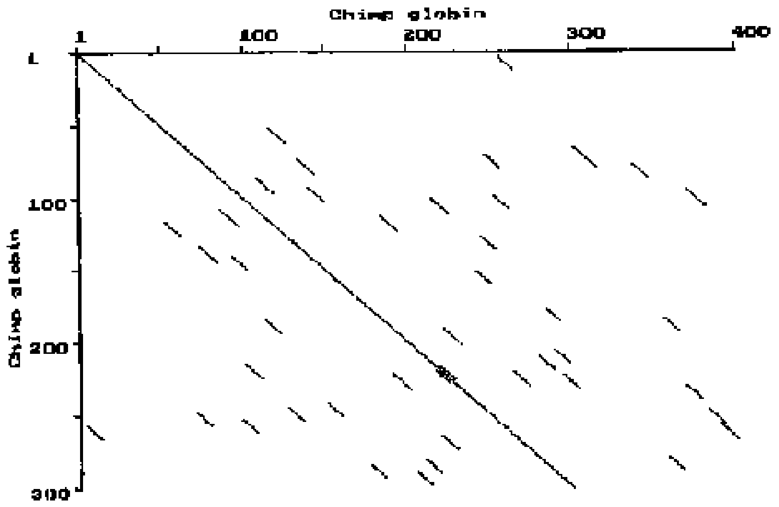


Figure 7.3: Dot Blot - filtered 8 of 10.



7.1.2 Identity Blocks

There is another way in which dot plots can be generated very quickly. This involves a computer method commonly known as “hashing” (list-sorting). As mentioned previously, these methods are incorporated into the FASTA algorithms. Basically, the idea is that instead of taking the complete matrix and calculating points for every entry in that matrix, a great saving can be made if the algorithm searches only for exact matches. Hence, this method looks only for blocks of perfect identity. The computational complexity of this algorithm grows linearly with increasing N.

The algorithm simply sub-divides the sequence into all “words” of a user specified block size. The same is done for the alternate sequence. In addition, for both sequences the location of each word is also recorded. These arrays of “words” are then sorted alphabetically and the arrays of locations are sorted in parallel with the “words”. Then, by comparing the sorted array from one sequence with that from the other sequence immediately gives the location of all identical “words”.

An algorithm which does be used to generate the dot plots shown in Figure 7.4 for identity blocks of length 5. The rapidity of this method compared to the exact method can be demonstrated by the dot plot shown in Figure 7.5 (with identity blocks of length 6). This figure extends the sequences compared in the chimpanzee globin intergenic region from (1-400 vs 1-300) up to (1-4000 vs 1-3000). The length of time required for a plot of the small region is not significantly shorter than the length of time it takes to calculate short identities on a 100 fold larger matrix.

The beauty of this method is demonstrated in Figure 7.6. This is a plot of all identities of length 6 between the chimpanzee and spider monkey sequences in the same region. The evolutionary homology between these sequences is easily discernible by the solid lines along the main diagonal despite the approx. 60 million years that separate these two groups. Further more, this is intergenic DNA with no known function to selectively maintain this homology (modulo an even more ancient eta-globin pseudogene). The insertion of some DNA is easily observed within chimpanzee sequence and then a corresponding deletion further down. These correspond to the insertion of an Alu element in the chimpanzee (and human and other ape) sequences (at approx. bp 1000) and then the presence of a truncated L1 element in the spider monkey (inserted at approx. bp 2600) that is not present in the great apes. These events are difficult to find by a simple inspection of the actual sequence code but are readily found by a visual inspection.

A more distant similarity can be seen in Figure 7.7. This is a plot of the identities of length 6 between the same region of the chimpanzee haemoglobin intergenic region and another intergenic region from the spider monkey. Note the similarity (the short diagonal line) in the circled region. This region of similarity corresponds to the location of another Alu element in the chimpanzee sequence.

There are many programs freely available to make dot plots. One which is particularly fast and interactive is the `dotter`

Figure 7.4: Identity Dot Blot.

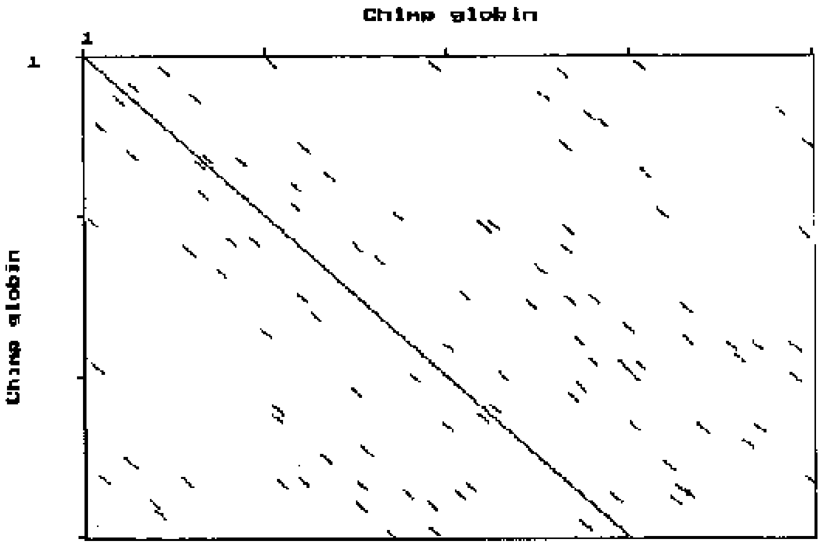


Figure 7.5: Identities of length 6bp. Chimpanzee hemoglobin intergenic DNA against itself.

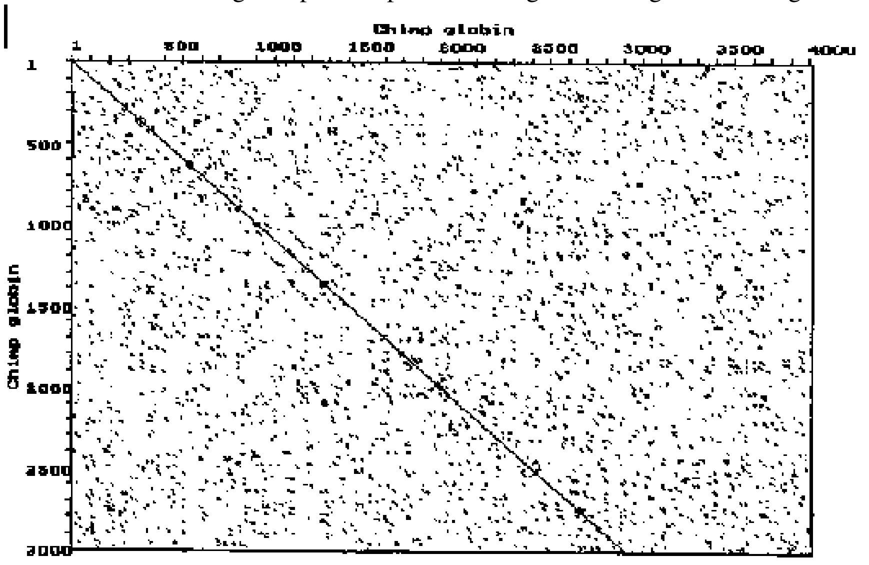


Figure 7.6: Identities of length 6bp. Chimpanzee hemoglobin intergenic DNA against spider monkey.

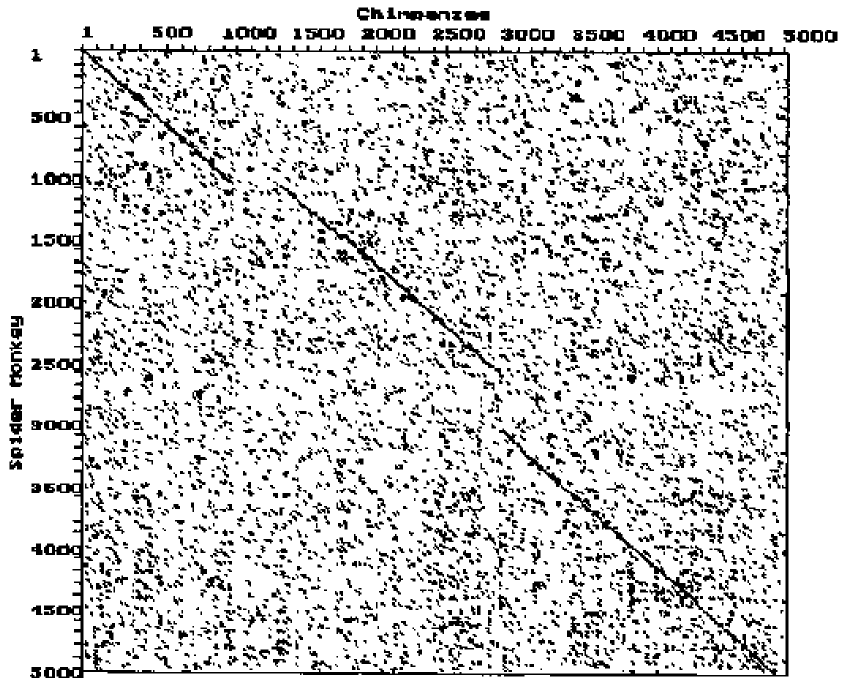


Figure 7.7: Identity dot plot. Chimpanzee hemoglobin intergenic region vs. Spider Monkey unrelated intergenic region.

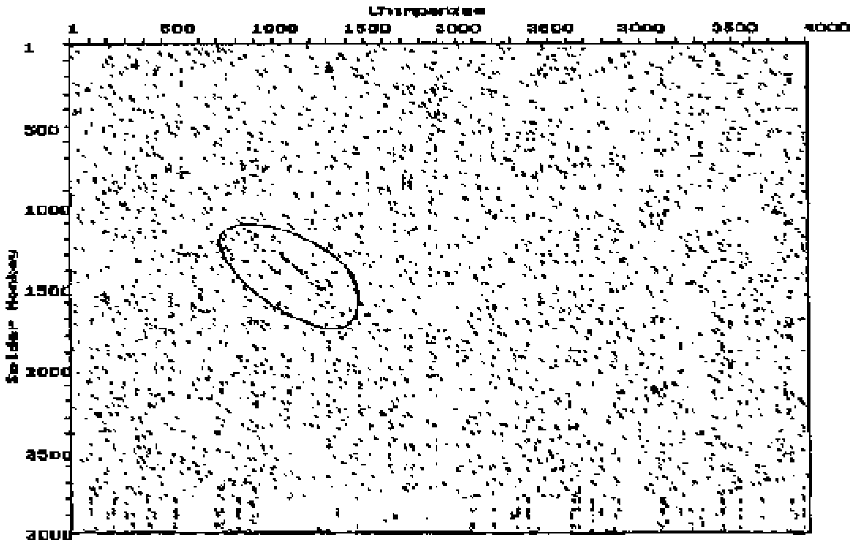
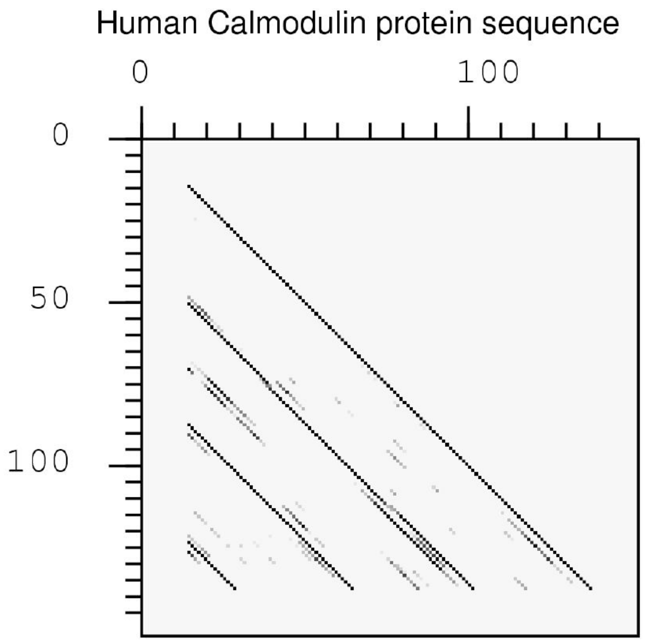


Figure 7.8: Human calmodulin protein sequence dot plotted against itself.



program. Some other interesting dot plots are comparisons of the calmodulin (Figure 7.8) protein against itself and the human epidermal growth factor (Figure 7.9) against itself. Both show internal repetitive elements. The neatest dot plot that I have yet seen is the human zeta globin (Figure 7.10) region and if you zero in on the intergenic region (Figure 7.11) the plot becomes fantastic (try to interpret this dot plot).

Figure 7.9: Human epidermal growth factor protein sequence dot plotted against itself.

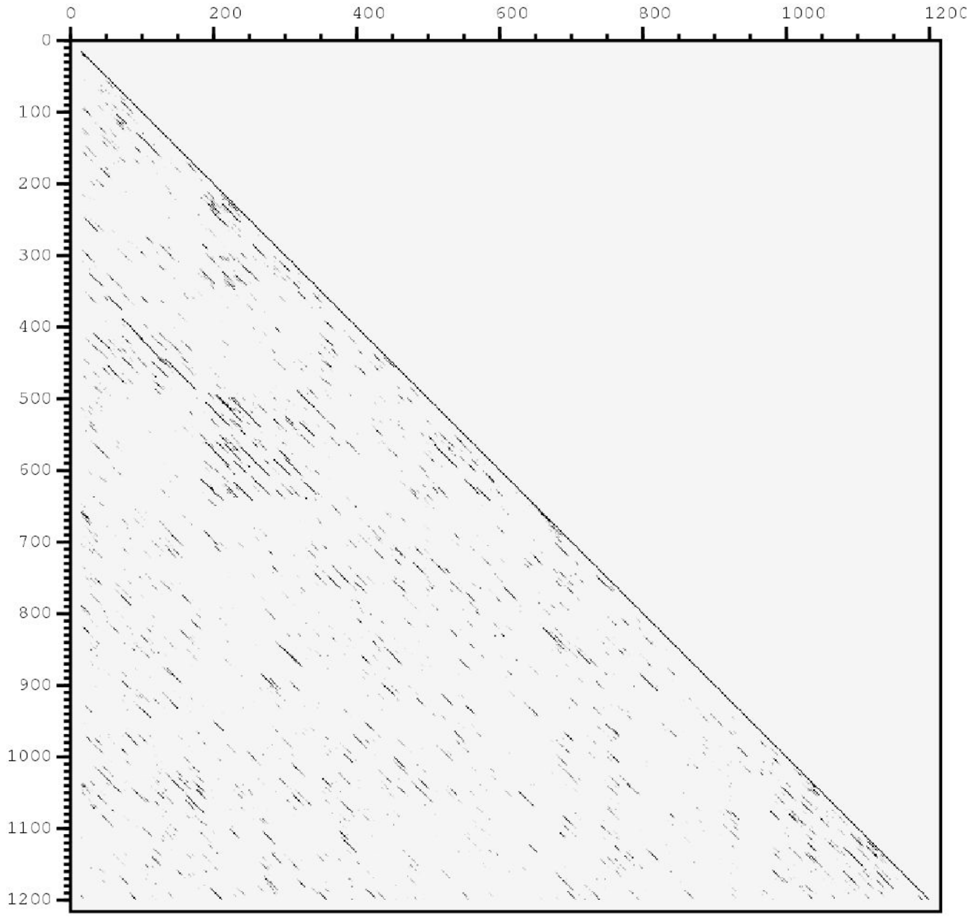


Figure 7.10: Human globin region (zeta, psizeta, psialpha1, alpha2, alpha1) dot plotted against itself.

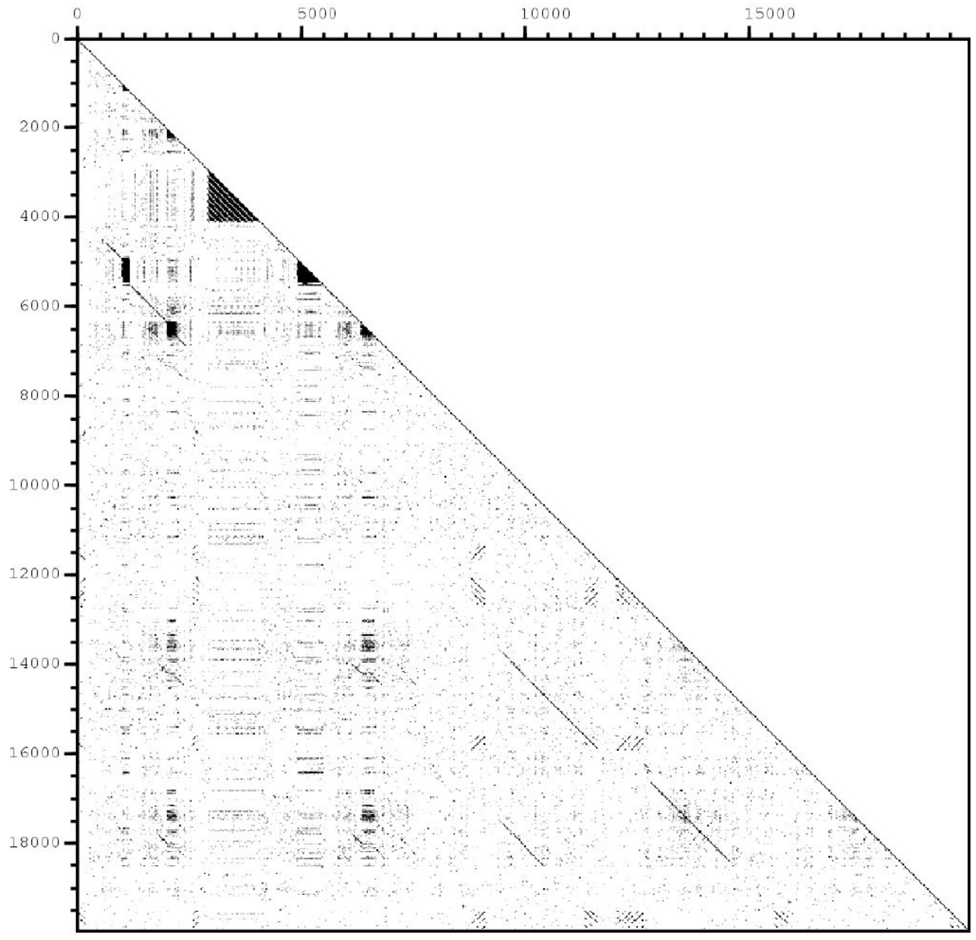
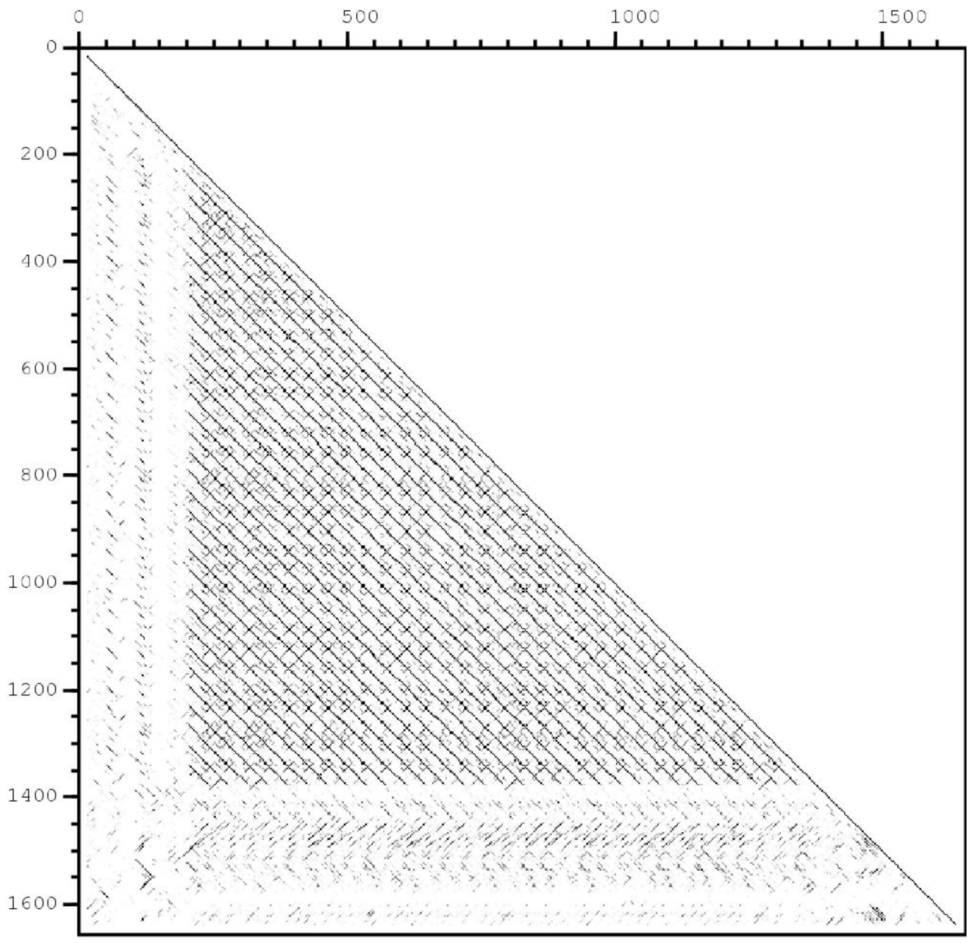


Figure 7.11: Human zeta globin intergenic region expanded from Figure 7.10.



7.2 Alignments

The dot plots provide a useful way to visualize the sequences being compared. They are not very useful however in providing an actual alignment between the two sequences. To do this, other algorithms are required.

First, a word on terminology. People in the field shudder when the terms similarity and homology are used indiscriminately. Similarity simply means that sequences are in some sense similar and has no evolutionary connotations. Homology refers to evolutionary related sequences stemming from a common ancestor.

The ability to calculate the correct alignment is crucial to many types of studies. It may, for example, alter from which part of a gene one segment was duplicated, it may alter the inferred number of point mutations, it may alter the inferred location of deletions / insertions, alter the inferred distance between species, and may alter the inferred phylogeny of the sequences along with whatever evolutionary hypotheses are dependent on these phylogenies.

An explicit and precise algorithm is also required. For example one paper in the prestigious journal NATURE stated that the alignment

------**CC**TTCAGAATACAGAA**T**AGGGACATAGAGA
ATCCCA**CC**CAGCCCCCTGGACC**T**GTAT---------

was optimal in the sense that gaps were inserted to maximize the number of base matches (the base matches are highlighted). They obviously did this alignment by eye and did not use an explicit algorithm. An alternate alignment (due to Fitch 1984, Nature 309:410) is

CCTT**CA**GAATA**CAG**AATAGG**G**ACATA**G**AGA
ATCC**CA**---CC**CAG**CCCCCT**G**GACCT**G**TAT

This alignment not only increases the number of base matches by 133 per cent, but also decreases the number of gaps by 50 per cent and reduces the number of gapped residues by 80 per cent. Hence, if the number of base matches can be increased by reducing the number of gaps, then clearly the original author's insertion of gaps did not maximize that number. Fitch recommends that the authors change their statement to the assertion that gaps were introduced to increase the number of base matches (rather than to maximize them). More generally this example shows the importance of i) using a well defined algorithm and ii) of using a computer based algorithm to perform these calculations. Even alignments that may appear simple and straightforward, if given to the computer, might yield alternatives that you did not consider.

7.2.1 The Needleman and Wunsch Algorithm

The most basic algorithm to align two sequences was developed by S.A. Needleman and C.D. Wunsch (1970, J. Mol. Biol. 48:443). The algorithm is a simple and beautiful way to find an alignment that maximizes a particular score[1]. (The score can be calculated in a variety of methods - as will be indicated below). The initial steps of the algorithm are reminiscent of the dot plot. The first step is to place the two sequences along the margins of a matrix as shown in Table 7.1.

In this first step, simply place a 1 anywhere the two sequences match and a 0 elsewhere. If done on a larger scale than is shown in Table 7.1, this would exactly recreate the dot plot shown in Figure 7.1. In this case however, we wish to find a path through this matrix which would define a more conventional alignment. For example, proceeding along the diagonal with no deviations would imply an alignment without any gaps. The introduction of a gap (either by an insertion or a deletion - an indel) in either sequence would correspond to moving either above or below the main diagonal.

To find the best route, Needleman and Wunsch suggested that you modify the matrix to represent this idea of tracing different pathways through the matrix. However, you want to include all possible pathways and from among these choose only that one which is best (in the sense of maximizing some score). Their method consists of two passes through the matrix. The first pass traces a score for all possible routes and moves right to left, bottom to top. Once the score for all possible routes are found, the maximum can be chosen (it will be somewhere on the topmost row or leftmost column) and a

[1]An alignment step by step example

Table 7.1: Initial setup for Needleman-Wunsch

The first step is to place the two sequences, in this case two protein sequences, along the margins of a matrix. Then place a one in the matrix where ever the two sequences agree.

	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	1	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	1	0	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	1	0	1	0	0	0
J	0	0	0	0	1	0	0	0	0	0	0	0	0
N	0	0	0	1	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	1	0	0	0	0	1	0	0
C	0	0	1	0	0	0	0	1	0	1	0	0	0
K	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	1	0	1	0	0	0
R	0	0	0	0	0	1	0	0	0	0	1	0	0
B	0	1	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

second pass can be carried out, this time running left to right, top to bottom to find that alignment that gives the maximum score.

The way to trace a score for all possible paths is shown in Table 7.2. For each element in the matrix you perform the following operation.

$$M_{i,j} = M_{i,j} + \max(M_{k,j+1}, M_{i+1,l})$$

where k is any integer larger than i and l is any integer larger than j. In words, alter the matrix by adding to each element the largest element from the row just below and to the right of that element and from the column just to the right and below the element of interest. This row and column for one element are shown in Table 7.2 by boxes. The number contained in each cell of the matrix, after this operation is completed, is the largest number of identical pairs that can be found if that element is the origin for a pathway which proceeds to the upper left.

We wish to have an alignment which covers the entire sequence. Hence, we can find on the upper row or on the left column the element of the matrix with maximum value. An alignment must begin at this point and can then proceed to the lower right. This is the second pass through the matrix. At each step of this pass, starting from the maximum, one moves one row *and* column to the lower right and finds the maximum in this row or column. The alignment must proceed through this point.

Continuing in this fashion one eventually hits either the bottom row or the rightmost column and the alignment is finished. This tracing pattern is shown in Table 7.3. Note that in this case the optimal alignment is not unique. There are two alignments and both give the optimal score of 8 matches.

These two alignments can be written in more familiar form as either

```
ABCNJ-RQCLCR-PM
* * * * * ** *
AJC-JNR-CKCRBP-
```

or as

Table 7.2: Half way through the second step

Add to each element of the matrix the largest number from the row or column to the lower right of each element proceeding right to left, bottom to top.

	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	1	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0	0	0	1	0	0	0	0	0	0	0	0
C	0	0	1	0	0	0	0	1	0	1	0	0	0
J	0	0	0	0	1	0	0	0	0	0	0	0	0
N	0	0	0	1	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	1	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

Table 7.3: Trace the alignment

The alignment is traced proceeding left to right, top to bottom choosing the largest numbers available. Alternate pathways both yielding the optimal alignment are shown .

	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	8	7	6	6	5	4	4	3	3	2	1	0	0
J	7	7	6	6	6	4	4	3	3	2	1	0	0
C	6	6	7	6	5	4	4	4	3	3	1	0	0
J	6	6	6	5	6	4	4	3	3	2	1	0	0
N	5	5	5	6	5	4	4	3	3	2	1	0	0
R	4	4	4	4	4	5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

```
ABC-NJRQCLCR-PM
*  *  *  *  *  **  *
AJCJN-R-CKCRBP-
```

both with 8 asterisks to denote the 8 matches. Note that in this particular case, gaps are given the same penalty as a mismatch. They simply do not add to the score.

7.2.2 The Smith-Waterman Algorithm

The Needleman-Wunsch algorithm creates a global alignment. That is, it tries to take *all* of one sequence and align it with *all* of a second sequence. Short and highly similar subsequences may be missed in the alignment because they are outweighed by the rest of the sequence. Hence, one would like to create a locally optimal alignment. The Smith and Waterman (1981, J. Mol. Biol. 147:195-197) algorithm finds an alignment that determines the longest/best subsequence pair that give the maximum degree of similarity between the two original sequences. This means that not all of the sequences might be aligned together.

Only minimal changes to the Needleman-Wunsch algorithm are required. These are

- A negative score/weight must be given to mismatches.
- Zero must be the minimum score recorded in the matrix.
- The beginning and end of an optimal path may be found anywhere in the matrix - not just the last row or column.

The first point is required to cause the score to drop as more and more mismatches are added. Hence, the score will rise in a region of high similarity and then fall outside of this region. If there are two segments of high similarity then these must be close enough to allow a path between them to be linked by a gap or they will be left as independent segments of local similarity. In general the Smith-Waterman algorithm includes gap penalties (to be discussed in section 7.4) and if this is the case, then the mismatch penalties are not required to be negative (to retain simplicity here, I have assumed that gap penalties are zero and use a negative mismatch penalty). Either way, the essence of a local alignment is that the score must decline.

The second point is required so that each pathway begins fresh at its beginning. Thus each short segment of similarity should begin with a score of zero. The third point indicates that the entire matrix must be searched for regions with high local similarity.

Again, for each element in the matrix you perform the following operation.

$$M_{i,j} = M_{i,j} + \max(M_{k,j-1}, M_{i-1,l})$$

But in this case it is easier to go left to right, top to bottom in the matrix - so here k is any integer smaller than i and l is any integer smaller than j. Also, for a local alignment $M_{i,j}$ must have a negative value if residue i is not the same as residue j. As an example the previous alignment can be reproduced with a penalty of -0.5 for each mismatch. The matrix will then be as given in Table 7.4. In this case the same alignment is found. However, the Smith-Waterman algorithm does not include the final M/P mismatch in its path as it is not part of the locally optimal solution. More generally, large chucks of each sequence may be missing from the local alignment (as in the alignment presented by BLAST).

It is seldom the case that these two approaches give the same answer. For example, a global and a local alignment of TTGACACCCTCCCAATTGTA versus ACCCCAGGCTTTACACAT give

```
TTGACACCCTCC-CAATTGTA
    ::  ::   ::  :
ACCCCAGGCTTTACACAT---


---------TTGACACCCTCCCAATTGTA              TTGACAC
         :: ::::                    or     :: ::::
ACCCCAGGCTTTACACAT-----------              TTTACAC
```

Table 7.4: Smith-Waterman example

Here a penalty of -0.5 has been added for each mismatch.

	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	1	0	0	0	0	0	0	0	0	0	0	0	0
J	0	0.5	0.5	0.5	2	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
C	0	0.5	1.5	0	0	1.5	1.5	3	1.5	3	1.5	1.5	1.5
J	0	0.5	0	1	2.5	1.5	1	1	2.5	2.5	2.5	2.5	2.5
N	0	0.5	0	2.5	0.5	2	2	2	2.5	2	2.5	2	2
R	0	0.5	0	1	2	3.5	2	2	2.5	2	4	2	2
C	0	0.5	1.5	1	2	2	3	4.5	3	4.5	3	3.5	3.5
K	0	0.5	0	1	2	2	3	2.5	4	4	4	4	4
C	0	0.5	1.5	1	2	2	3	4	4	5	4	3.5	3.5
R	0	0.5	0	1	2	3.5	3	2.5	4	3.5	6	4.5	4.5
B	0	2	0	1	2	2	3	3	4	3.5	4.5	5.5	5.5
P	0	0.5	1.5	1.5	2	2	3	2.5	4	3.5	4.5	7	5

respectively. The global alignment has considered a penalty for the end gaps but the local alignment has simply searched for the best substrings that can be put together.

If the sequences are not known to be homologous throughout their entire length, a local alignment should be the method of choice. Sometimes the two methods will give similar answers but if the homology is distant, a local alignment will be more likely to find the remaining patches of homology.

7.3 Testing Significance

The above algorithms are trying to find the best way to match up two sequences. This does not mean that they will find anything profound. For example, if I take these two sentences (and delete spaces and delete non-amino acids), they can be aligned.

```
THESEALGRITHMARETR--YINGTFINDTHEBESTWAYTMATCHPTWSEQENCES
:: :.. . ..  ...:   :    ::::..       ::  .  : ...
THISDESNTMEANTHATTHEYWILLFINDAN-------YTHIN-GPRFND------
```

Depending on your intuition you may or may not think this to be a pretty good alignment particularly at the amino-terminus. There are a total of 12 exact matches and 14 conservative substitutions. But there is obviously no homology between these two “sentence” sequences. How do we test whether or not an alignment is significant?

As a more biological example, consider the alignment of human alpha haemoglobin and human myoglobin. If you remember your basic biology, you should remember that these two proteins do similar functions of transporting oxygen in the blood and muscle respectively. But are they evolutionarily related? An alignment of the two looks like ...

```
Human alpha haemoglobin (141 aa) vs. Human myoglobin (153 aa)


VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHF-DLS-----HGSAQ
 :: ..   : ..::::.:.  ..:.:.: :.: . :.: . : .: .:.     ..:..
GLSDGEWQLVLNVWGKVEADIPGHGQEVLIRLFKGHPETLEKFDKFKHLKSEDEMKASED


VKGHGKKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLP
.: :: .: .::.. . . ..  .....:.. :: : ..    ....:.:.. .:... :
LKKHGATVLTALGGILKKKGHHEAEIKPLAQSHATKHKIPVKYLEFISECIIQVLQSKHP
```

```
AEFTPAVHASLDKFLASVSTVLTSKYR------
..:.........: :.  ..  ..:.:.
GDFGADAQGAMNKALELFRKDMASNYKELGFQG
```

Again looks like a reasonably good alignment. Or how about chicken lysozyme and bovine ribonuclease. An alignment of these gives

```
Chicken lysozyme (129 aa) vs. Bovine ribonuclease (124 aa)


KVFGRCELAAAMKRHGLDNYRGYSLGNWVCAAKFESNFNTQATNRNTDGSTDYGILQINS
:  .    :: ..:. .:.  . . ..  :.....:.  :..  . ... .. .. ....
KETA----AAKFERQHMDSSTSAASSSNYCNQMMKSRNLTKDRCKPVNTFVHESLADVQA


RWWCNDGRTP--GSRNLCNIPCSALLSSDITASVNCAKKIVSDGDGMNAWVAWRNRCKGT
   :.. ...  .... :  ..:..  .:  .. ...:   .. .. .:       :.:.
V--CSQKNVACKNGQTNCYQSYSTMSITDCRET-GSSKYPNCAYKTTQANKHIIVACEGN


DVQAWIRGCRL
   .   . ..
PYVPVHFDASV
```

Again a reasonable alignment or so it seems. How do you know which sequences (if any) are homologous?

A common and simple test to determine if the alignment of two sequences is statistically significant is to carry out a simple permutation test. This consists of

1. Randomly rearrange the order of one or both sequences
2. Align the permuted sequences
3. Record the score for this alignment
4. Repeat steps 1-3 a large number of times.

Doing this say 10000 times, gives a distribution of alignment scores that could be expected for random sequences with a similar amino acid content. If the actual alignment has a score much higher than that of the permuted sequences, then you know that they must be homologous to some extent.

A plot of 10,000 alignment scores for the human myoglobin and human alpha haemoglobin sequences are shown in figure 7.12. The permuted scores range from 14 to 75 but most are less than 50. Also note the skewness of the distribution - statistics based on a normal distribution would be strongly biased. The skew is expected since in each case the alignment algorithm is trying to maximize the score. The score for the alignment of the two actual sequences is 179 (indicated by the arrow). Obviously, myoglobin and haemoglobin are evolutionarily related and still retain many features of their homology. This alignment has a probability of less than 0.0001 of occurring by chance alone.

A plot of 10,000 alignment scores for the chick lysozyme and bovine ribonuclease sequences are shown in figure 7.13. Again note, the skew and note that this “random” distribution is somewhat different from the haemoglobin “random” distribution. This is due to the differential effects of amino acid composition in these proteins. The permuted scores range from 14 to 72. The actual score for the proteins is 30 (indicated by the arrow). Obviously whatever homology that once existed between these proteins has been completely destroyed by time.

These two examples are clear cut. There is a large grey area where the tests may be uncertain of the degree of homology between sequences. For protein sequences Doolittle’s rule of thumb is that greater than 25% identity will suggest homology, less than 15% is doubtful and for those cases between 15-25% identity a strong statistical argument is required. Personally, I would prefer the statistical test in all cases, since they are easy to do and things such as internal repeats and unusual amino acid compositions can sometimes confuse the picture.

Figure 7.12: Histogram of alignment scores

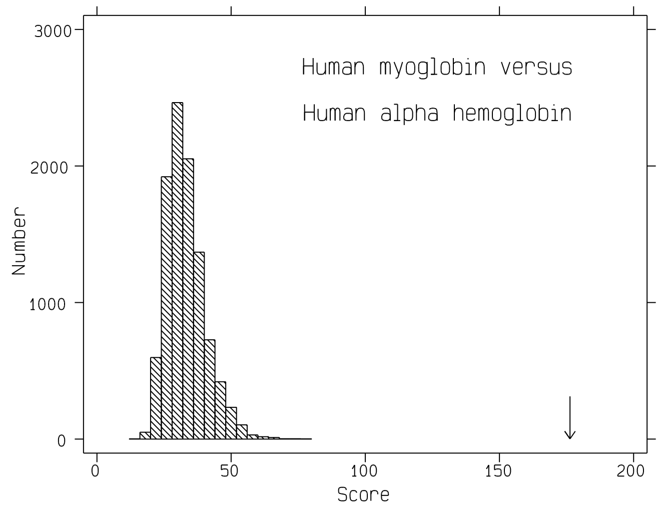
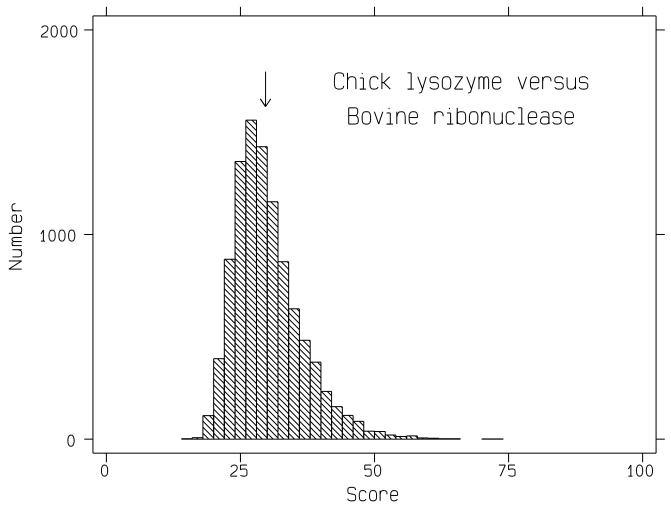


Figure 7.13: Histogram of alignment scores



7.4 Gaps and Indels

Gaps were rather freely permitted in the overly simple implementations of the Needleman-Wunsch and Smith-Waterman algorithms shown above. What would happen if you feel that gaps should be rarer events in your particular protein? It is possible to assign a different weight to gaps. This can be done by subtracting from the score, some predetermined value every time a gap is required. In this case you can define a weight as

$$W_k = a + bk$$

where k is the length of the gap. Hence you can control whether many short gaps occur or whether long gaps occur but more infrequently. Deletions do occur but when they occur it is seldom many small, short deletions but rather fewer and longer deletions.

How do you choose a gap penalty? Unfortunately, there is little knowledge to help here. Most of the tests done so far depend on an empirical basis designed to achieve some end. For example, Smith and Fitch have derived (by exhaustive search) gap penalties that will best align distantly related haemoglobin genes. But there is no guarantee that these values would work well for the protein or (worse) the nucleotide sequence that you are interested in. Typical values are

$$0.5 < a < 5.0$$

$$0.05 < b < 1.0$$

but there is nothing special about these values other than the fact that they seem to work well for some of the common comparisons. Note that in general $a > b$. This corresponds with biological knowledge of how gaps are generated - it is easier to generate one gap of two residues rather than two gaps of one residue since the former can be created by a single mutational event.

More recently, Reese and Pearson (2002, Bioinformatics 18:1500-1507) examined how these parameters might change as a function of the distance between aligned sequences. Their criteria was the "correct" identification of distant homologues. They found that b did not change but that a did change with distance. Again, through empirical tests they showed that optimal penalties were $a = 25 - 0.1 \times (PAMdistance)$ (PAM is a method of measuring distance that will be explained in section 8.2.1) and $b = 5$ where these penalties are in $1/3$ bit units (see section 10.5.1).

7.4.1 "Natural" Gap Weights - Thorne, Kishino & Felsenstein

In a series of papers Thorne, Kishino and Felsenstein (JME 33:114, 1991) and (JME 34:3, 1992) have developed a method to find maximum likelihood estimates of the gap penalties (and transition/transversion rates) while doing an alignment. This eliminates the necessity for choosing an arbitrary gap penalty.

They develop a maximum likelihood method to examine the possible paths of descent from a common ancestor for two sequences. The creation of gaps is modeled as a birth - death process with separate parameters for birth rate and death rate. The model then finds the likelihood of particular paths through the matrix given the transition parameters. It then examines alternative parameters and chooses that path and parameter set with the highest likelihood. The big difficulty with this method is the enormous computer time required to carry out the calculations.

A related question is the assignment of weights to individual differences in nucleotide or protein sequences (more on this later). There have also been advances in methods to try to find statistically bounded sets of alignments. That is, the set of alignments that are within 95% confidence limits of some best answer. Again another fertile area were many significant improvements are being made.

It is important to realize that an optimal alignment is optimal only for the particular values chosen for the mismatch and gap weights. When any of these are altered, the optimal alignment will also change. Also be aware of the fact that nature is seldom mathematically optimized. Fitch and Smith (1983, PNAS 80:1382) have derived a set of "rules of thumb" a subset of which are given in Table 7.5. Even with the very best programs it still requires some degree of experience to draw the right conclusions from the results produced and a good grasp of the biology of the problem is essential.

Table 7.5: Some rules of thumb for alignments

1.	A gap and its length are distinct quantities. Different weights should be applied to each.
2.	Weights for different mismatches should be permitted. A transition is more likely than a transversion; a Ile-Val more likely than Ile-Arg change.
3.	If the two sequences have no obvious relationship at their right and left ends, then end gaps should not be penalized.
4.	Unless two sequences are known to be homologous over their entire length, a local alignment is preferable to a global alignment.
5.	An optimal alignment is by no means necessarily statistically significant. One must make some estimate of the probability that a given alignment is due to chance.
6.	An alignment demonstrates similarity, not necessarily, homology. Homology is an evolutionary inference based on examination of the similarity and its biological meaning. Sequence similarity may result from homology but it may also result from chance, convergence or analogy.

7.5 Multiple Sequence Alignments

Conceptually, there is no reason why a Needleman-Wunsch algorithm can not be performed with more than two sequences. The matrix simply becomes multi-dimensional and the algorithm would work successively through each dimension. There are however, significant practical problems with this approach. In this case instead of growing as an N^2 problem, the computational time will grow as N^m, where m is the number of sequences. Hence, even for just 100 nucleotides from 5 species, this is

$$100^5 = 10,000,000,000$$

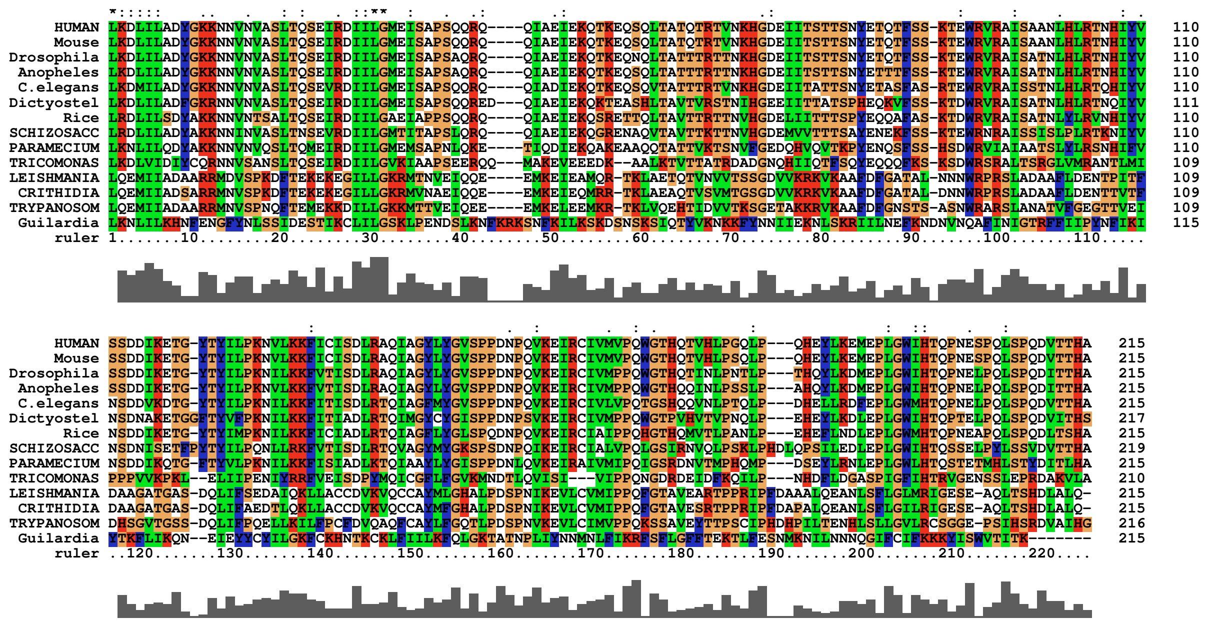
operations or the equivalent of doing an alignment for two sequences each 100,000 nucleotides long. Obviously different methods need to be employed. In general these require more assumptions and are not as precise nor “all-encompassing” as the Needleman-Wunsch or Smith-Waterman algorithms.

If desired, simple scores of similarity can be readily found using rapid techniques on all pairs of sequences. But to take these local regions of similarity and turn out an alignment is somewhat more complicated.

Bains (1986, Nuc. Acids Res. 14:159) suggested an iterative method which involves successive applications of the standard algorithms. It begins with a trial consensus alignment (say the alignment between sequences 1 and 2. Then the third sequence is aligned against the consensus sequence and a new consensus emerges. This continues until the consensus alignment converges to a global consensus. This type of method will be very dependent on the order that the sequences are introduced. Thus a different alignment could arise using the same technique and the same sequences but in a different order.

One of the most popular multiple alignment programs begins with all pairwise alignments and is called Clustal. It was written by Higgins and Sharp (1989, CABIOS 5:151). The alignments are done in four steps. In the first step, all pairwise similarity scores are calculated. This is done using rapid alignment methods. The second step is to create a similarity matrix and then to cluster the sequences based on this similarity using a cluster algorithm (see the section 9.2). The third step is to create an alignment of clusters via a consensus method. The final step is to create a progressive multiple alignment. This is performed by sequentially aligning groups of sequences, according to their branching order in the clustering. Three variants currently are being used, ClustalW, a companion program call ClustalX and the older ClustalV. An example of the output from ClustalX is shown in Figure 7.14.

Other methods make use of a multiple dimensional dot plot and then look for dots that are common to each group (Vingron



```
           *:::::    ..   .   : :   . ::**  :   .  .:       .:                     .:                    :     :    .      : .
     HUMAN LKDLILADYGKKNNVNVASLTQSEIRDIILGMEISAPSQQRQ----QIAEIEKQTKEQSQLTATQTRTVNKHGDEIITSTTSNYETQTFSS-KTEWRVRAISAANLHLRTNHIYV   110
     Mouse LKDLILADYGKKNNVNVASLTQSEIRDIILGMEISAPSQQRQ----QIAEIEKQTKEQSQLTATQTRTVNKHGDEIITSTTSNYETQTFSS-KTEWRVRAISAANLHLRTNHIYV   110
Drosophila LKDLILADYGKKNNVNVASLTQSEIRDIILGMEISAPSAQRQ----QIAEIEKQTKEQNQLTATTTRTTNKHGDEIITSTTSNYETQTFSS-KTEWRVRAISATNLHLRTNHIYV   110
 Anopheles LKDLILADYGKKNNVNVASLTQSEIRDIILGMEISAPSAQRQ----QIAEIEKQTKEQSQLTATTTRTTNKHGDEIITSTTSNYETTTFSS-KTEWRVRAISATNLHLRTNHIYV   110
 C.elegans LKDMILADYGKKNNVNVASLTQSEVRDIILGMEISAPSQQRQ----QIADIEKQTKEQSQVTATTTRTVNKHGDEIITATTSNYETASFAS-RTEWRVRAISSTNLHLRTQHIYV   110
Dictyostel LKDLILADFGKRNNVNVASLTQSEIRDIILGMEISAPSQQRED---QIAEIEKQKTEASHLTAVTVRSTNIHGEEIITTATSPHEQKVFSS-KTDWRVRAISATNLHLRTNQIYV   111
      Rice LRDLILSDYAKKNNVNTSALTQSEIRDIILGAEIAPPSQQRQ----QIAEIEKQSRETTQLTAVTTRTTNVHGDELIITTTSPYEQQAFAS-KTDWRVRAISATNLYLRVNHIYV   110
SCHIZOSACC LRDLILADYAKKNNINVASLTNSEVRDIILGMTITAPSLQRQ----QIAEIEKQGRENAQVTAVTTKTTNVHGDEMVVTTTSAYENEKFSS-KTEWRNRAISSISLPLRTKNIYV   110
PARAMECIUM LKNLILQDYAKKNNVNVQSLTQMEIRDIILGMEMSAPNLQKE----TIQDIEKQAKEAAQQTATTVKTSNVFGEDQHVQVTKPYENQSFSS-HSDWRVIAIAATSLYLRSNHIFV   110
TRICOMONAS LKDLVIDIYCQRNNVSANSLTQSEIRDIILGVKIAAPSEERQQ---MAKEVEEEDK--AALKTVTTATRDADGNQHIIQTFSQYEQQQFKS-KSDWRSRALTSRGLVMRANTLMI   109
LEISHMANIA LQEMIIADAARRMDVSPKDFTEKEKEGILLGKRMTNVEIQQE----EMKEIEAMQR-TKLAETQTVNVVTSSGDVVKRKVKAAFDFGATAL-NNNWRPRSLADAAFLDENTPITF   109
 CRITHIDIA LQEMIIADSARRMNVSPKDFTEKEKEGILLGKRMVNAEIQQE----EMKEIEQMRR-TKLAEAQTVSVMTGSGDVVKRKVKAAFDFGATAL-DNNWRPRSLADAAFLDENTTVTF   109
TRYPANOSOM LQEMIIADAARRMNVSPNQFTEMEKKDILLGKKMTTVEIQEE----EMKELEEMKR-TKLVQEHTIDVVTKSGETAKKRVKAAFDFGNSTS-ASNWRARSLANATVFGEGTTVEI   109
 Guilardia LKNLILKHNFENGFYNLSSIDESTIKCLILGSKLPENDSLKNFKRKSNFKILKSKDSNSKSIQTYVKNKKFYNNIIEKNLSKRIILNEFKNDNVNQAFINIGTRFFIIPYNFIKI   115
     ruler 1.......10........20........30........40........50........60........70........80........90.......100.......110.....

                                 :                     .  :       .  : .          :              :  ::    .   :
     HUMAN SSDDIKETG-YTYILPKNVLKKFICISDLRAQIAGYLYGVSPPDNPQVKEIRCIVMVPQWGTHQTVHLPGQLP---QHEYLKEMEPLGWIHTQPNESPQLSPQDVTTHA   215
     Mouse SSDDIKETG-YTYILPKNVLKKFICISDLRAQIAGYLYGVSPPDNPQVKEIRCIVMVPQWGTHQTVHLPSQLP---QHEYLKEMEPLGWIHTQPNESPQLSPQDVTTHA   215
Drosophila SSDDIKETG-YTYILPKNILKKFVTISDLRAQIAGYLYGVSPPDNPQVKEIRCIVMPPQWGTHQTINLPNTLP---THQYLKDMEPLGWIHTQPNELPQLSPQDITTHA   215
 Anopheles SSDDIKETG-YTYILPKNVLKKFVTISDLRAQIAGYLYGVSPPDNPQVKEIRCIVMPPQWGTHQQINLPSSLP---AHQYLKDMEPLGWIHTQPNELPQLSPQDITTHA   215
 C.elegans NSDDVKDTG-YTYILPKNILKKFITISDLRTQIAGFMYGVSPPDNPQVKEIRCIVLVPQTGSHQQVNLPTQLP---DHELLRDFEPLGWMHTQPNELPQLSPQDVTTHA   215
Dictyostel NSDNAKETGGFTYVFPKNILKKFITIADLRTQIMGYCYGISPPDNPSVKEIRCIVMPPQWGTPVHVTVPNQLP---EHEYLKDLEPLGWIHTQPTELPQLSPQDVITHS   217
      Rice NSDDIKETG-YTYIMPKNILKKFICIADLRTQIAGFLYGLSPQDNPQVKEIRCIAIPPQHGTHQMVTLPANLP---EHEFLNDLEPLGWMHTQPNEAPQLSPQDLTSHA   215
SCHIZOSACC NSDNISETFPYTYILPQNLLRKFVTISDLRTQVAGYMYGKSPSDNPQIKEIRCIALVPQLGSIRNVQLPSKLPHDLQPSILEDLEPLGWIHTQSSELPYLSSVDVTTHA   219
PARAMECIUM NSDDIKQTG-FTYVLPKNILKKFISIADLKTQIAAYLYGISPPDNLQVKEIRAIVMIPQIGSRDNVTMPHQMP---DSEYLRNLEPLGWLHTQSTETMHLSTYDITLHA   215
TRICOMONAS PPPVVKPKL--ELIIPENIYRRFVEISDPYMQICGFLFGVKMNDTLQVISI---VIPPQNGDRDEIDFKQILP---NHDFLDGASPIGFIHTRVGENSSLEPRDAKVLA   210
LEISHMANIA DAAGATGAS-DQLIFSEDAIQKLLACCDVKVQCCAYMLGHALPDSPNIKEVLCVMIPPQFGTAVEARTPPRIPFDAAALQEANLSFLGLMRIGESE-AQLTSHDLALQ-   215
 CRITHIDIA DAAGATGAS-DQLIFAEDTLQKLLACCDVKVQCCAYMFGHALPDSPNIKEVLCVMIPPQFGTAVESRTPPRIPFDAPALQEANLSFLGILRIGESE-AQLTSHDLALQ-   215
TRYPANOSOM DHSGVTGSS-DQLIFPQELLKILFPCFDVQAQFCAYLFGQTLPDSPNVKEVLCIMVPPQKSSAVEYTTPSCIPHDHPILTENHLSLLGVLRCSGGE-PSIHSRDVAIHG   216
 Guilardia YTKFLIKQN--EIEYYCYILGKFCKHNTKCKLFIILKFQLGKTATNPLIYNNMNLFIKRFSFLGFFTEKTLFESNMKNILNNNQGIFCIFKKKYISWVTITK-------   215
     ruler ..120.......130.......140.......150.......160.......170.......180.......190.......200.......210.......220....
```

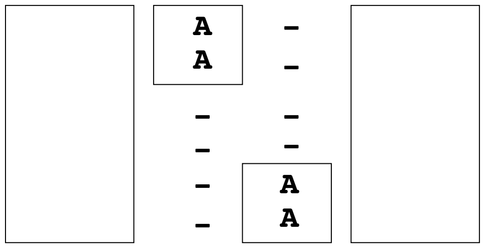
Figure 7.14: An example of the output from ClustalX, a popular multiple sequence alignment program. The shaded diagram at the base provides a measure of the similarity within each column.

& Argos 1991 J.Mol.Biol. 218:33-43). Still others rely heavily on user input such as the popular windows program MACAW (Schuler, Altschul & Lipman, 1991 Proteins Struct. Func. Genet. 9: 180-190). Others such as MSA (Gupta, Kececioglu & Schaffer, 1995 J. Comput. Biol. 2:459-472) attempt to provide a near-optimal sum-of-pairs global solution to the multiple alignment. Most of these programs attempt to find a solution such that some measure of the multiple alignment is minimized (or maximized). Most however, can only provide a guess to the best solution. Kececioglu has developed a new branch and bound algorithm that is guaranteed to converge to the true optimal solution (just no guarantees on how long that will take). This whole area is ripe for major theoretical advances and for the creation of better interface programs.

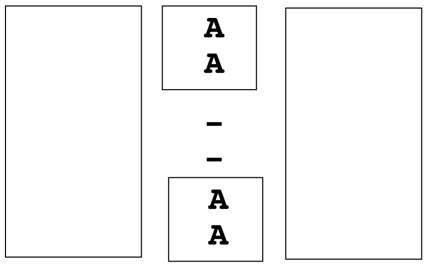
One obvious extension of these algorithms is to construct an alignment and a phylogeny for sequences all at the same time. This is because the alignment will affect distances between sequences and this will affect the inferred phylogeny. Similarly a different phylogeny will imply a different alignment of the sequences. Now you are talking about a chicken and egg problem! Never-the-less, some progress has been made in this area. Jotun Hein has come up with a program TREEALIGN which will do exactly this. It is available in the list from EMBL software given at ftp.ebi.ac.uk/pub/software/unix but again it is a very slow program.

How well do they actually work? McClure, Vasi and Fitch (1994, Mol. Biol. Evol. 11:571-592) tested how well the different algorithms could detect and correctly align, ordered functional motifs in several proteins. They used haemoglobin (5 motifs), kinase (9 motifs), aspartic acid protease (3 motifs), and ribonuclease H (4 motifs) proteins. They calculated the number of times (out of 100) that different algorithms correctly aligned these motifs for each protien. The results obviously depend on the divergence of the proteins, the number of sequences, the length of the motifs and the indel penalties but were often disappointing. As an example, the results for just ClustalV with 6 sequences were (100, 92, 100, 100, 100), (100, 83, 67, 100, 100, 100,100, 100, 100), (100, 0, 67) and (100, 67, 50, 50), respectively. Note that these motifs should be highly conserved and retain the most information enabling a correct alignment. ClustalV was one of the better algorithms but would still often miss these motifs.

When all is said and done, people will still find that the alignments produced by the programs can be improved by a judicious and critical examination by eye. Spending time to slowly and carefully examine your alignments by hand is recommended. Occasionally you might see an alignment that contains

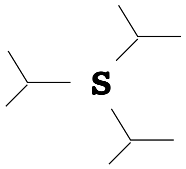


when an obviously better alignment would be



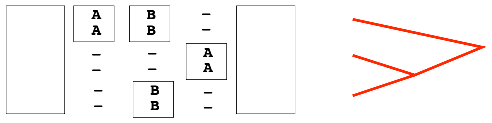
But why should this be necessary?

Many algorithms make some use of a tree or phylogeny in the construction of the alignment. It is how this information is used that can create some of these problems. If the nodes, S, containing the above deletion are central to the phylogeny, e.g.

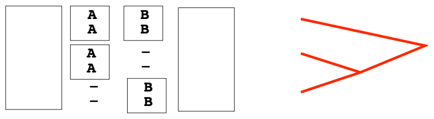


then insertions of the block 'A' must be made independently within each evolutionary branch. This will incur the same penalty in the local alignment whether it is placed to the left or to the right.

Similarly you might see



instead of



when the phylogeny shown on the right of the diagram (in red) is used. For many algorithms these are situations where the "apparent" score changes little or at all and hence the algorithm will not recognize it as a possibility for improvement (pers. comm. John Kececioglu).

In addition to these problems, all algorithms that I know of consider the penalty applied to gaps (and mismatches) as a constant throughout the length of the sequence. Yet all biologists recognize that this is not the case and understand that indels are more likely at the ends of a sequence and more likely in loop regions than in catalytic centers. We also have little idea of what are appropriate quantitative levels for the gap penalties. As a result, the alignments can always be improved by a careful examination. The algorithms can help with task. We routinely do several automated alignments for every comparison (minimally one with the default penalties, one with more severe and one with less severe penalties) and then compare these by hand.

Chapter 8

Distance Measures

8.1 Nucleotide Distance Measures

8.1.1 Simple counts as a distance measure

One of the most common measures used in computer algorithms for sequence analysis is some measure of the distance between two sequences. For many methods it is absolutely critical to get an accurate measure of distance. Past studies have shown that most algorithms that make use of a distance are not robust to small deviations in the distance matrices. This problem is also related to weighting differences between sequences.

Why bother about corrections for distances? Consider the analogy of the difference between a Ford Tempo and a Pontiac Sunbird. This is not the same difference as that between a Tempo and a Mercedes. They are all different cars but there is a greater qualitative difference between them (minimally, a big difference exists in the price between a Tempo and a Mercedes but less so between a Tempo and Sunbird). The same thing applies for sequences. Two sequences that differ by an A and G do not have the same quality of difference as do two sequences that differ by an A and a T. The former substitution is a transition and can happen readily while the latter is a transversion and occurs far less frequently. Hence it would be desirable to weight or to treat these substitutions in a different fashion. There is no reason why we should have used 1 for a residue match and 0 for a mismatch in the section on alignments. You can use any value for these that you wish (and indeed 1 and 0 are particularly poor choices).

But more generally than this problem of simple weighting, there are also subtler problems. Lets assume for the moment that all mutations occur with equal frequency. Then you might think that the difference between two sequences could be calculated simply by counting the number of nucleotide differences between the species. Lets consider how this difference, this measure of distance changes over time. Figure 8.1 shows the difference expected between two sequences that have diverged at increasing times into the past. The proportion of differences are calculated simply by counting the number of nucleotide differences divided by the total length of the sequence. Hence,

$$D = k/n,$$

$$Var(D) = D(1 - D)/n,$$

where n is the length of the sequence and k is the number of nucleotides that differ. In Figure 8.1, μ is rate of substitutions for the sequences and t is the length of time since the last common ancestor of these sequences. The rate of change is initially going up with a slope equal to twice what one might expect from the product of the mutation rate and time because both sequences are diverging from a common ancestor. Figure 8.1 shows that as the time of divergence increases the percent difference or the distance increases. Initially this occurs linearly however as time proceeds the measure of distance begins to slow its increase and finally reaches an asymptote of 0.75 and ceases to increase at all.

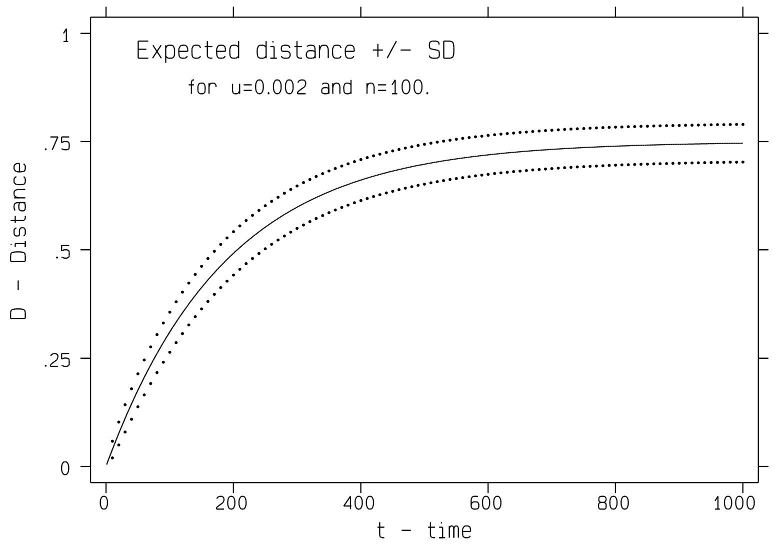


Figure 8.1: An asymptotic divergence with time.

This is quite reasonable when you think about it. There are only four types of nucleotides. A random collection based on these four possibilities will have one quarter of them identical by chance alone. But this has lots of implications for the distances that are calculated between species. A pair at time $t = 20$ are expected to have $D = 7.6$ and a pair at $t = 40$ are expected to have $D = 14.4$. These can be easily distinguished. But a pair at $t = 500$ and $t = 1000$ have D's of 69.8 and 74.6. These will be hard to tell apart. And yet, in both cases there is a doubling of the divergence between species pairs.

8.1.2 Jukes - Cantor Correction

As the time of divergence between two sequences increases the probability of a second substitution at any one nucleotide site increases and the increase in the count of differences is slowed. This makes these counts an undesirable measure of distance. In some way, this slow down must be accounted for. The solution to this problem was first noted by Jukes and Cantor (1969; Evolution of Protein Molecules, Academic Press). Instead of calculating distance as a simple count take the distance as

$$D_{JC} = -(\frac{3}{4}) \ln \left(1 - (\frac{4}{3})D\right),$$

$$Var(D_{JC}) = \frac{D(1-D)}{[n(1-(\frac{4}{3})D)^2]}$$

(Kimura and Ohta 1972; J. Mol. Evol. 2:87-90).

A plot of this function for the same range of parameters as in Figure 8.1 is given in Figure 8.2. This figure shows that this distance measure increases linearly with time (this is one property that is desirable for a distance measure). This is termed the Jukes & Cantor correction to distance and clearly indicates that divergence is a logarithmic function of time.

Observe the large increase in the variance as time increases. As D gets closer and closer over time to 0.75 the variance increases. In the limit as D approaches 0.75, the variance approaches infinity. This is an indication that the measure of distance becomes increasingly less reliable as time increases.

Note that in expectation D is less than 0.75 but in reality a single instance of D can be greater than 0.75. If this is the case then a Jukes-Cantor correction cannot be done and D_{JC} is undefined because the argument of the logarithm will be negative. In this case you can apply a method developed by Tajima (1993, Mol. Biol. Evol. 10:677-688). He suggests using the modified estimator

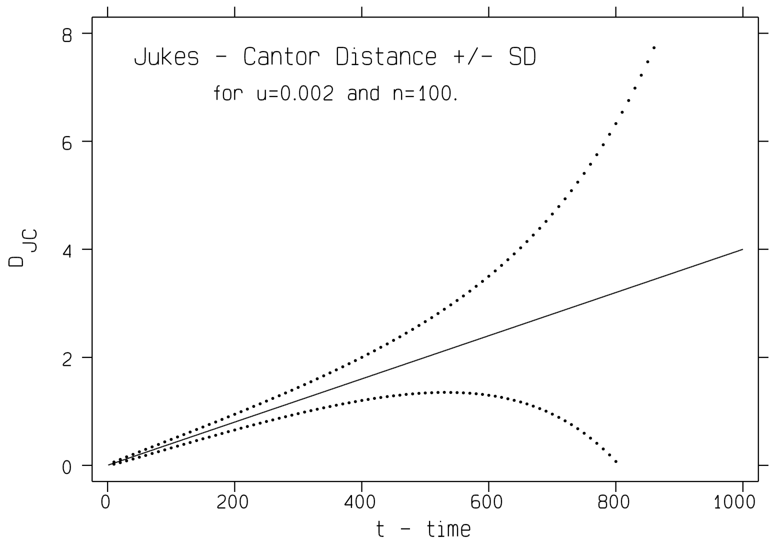


Figure 8.2: A correction leads to linear divergence with time.

$$D^*_{JC} = \sum_{i=1}^{k} \frac{k^{(i)}}{i(\frac{3}{4})^{i-1} n^{(i)}}$$

where

$$k^{(i)} = k!/(k-i)! \qquad \text{and} \qquad n^{(i)} = n!/(n-i)!$$

With variance

$$Var(D^*_{JC}) = D^*_{JC}(1-D^*_{JC})exp(8D^*_{JC}/3)/(n-1).$$

Here k is the count of differences between the two sequences and n is the length of these sequences. This is actually just a different formulation of the same quantity using a Taylor series expansion to avoid the logarithm. This estimator of distance is defined for all parameter values and actually has less bias than Jukes and Cantor's original correction for small levels of divergence. Tajima provides similar adjustments to all of the corrections noted below.

8.1.3 Kimura 2-parameter Correction

Note that this still does not correct for differences in the rates of transition and transversion. To do this you can use what is called the Kimura 2-parameter correction. This was a method established by Kimura (1980; J. Mol. Evol. 16:111-120) where the rates of transitions are assumed to be α and the rates of transversions are β. Then if the observed percentage of transitional differences are P and the observed percentage of transversion differences are Q, the estimate of distance is

$$D_{K2p} = -(\frac{1}{2})\ln(1-2P-Q) - (\frac{1}{4})\ln(1-2Q)$$

and

$$Var(D_{K2p}) = [c_1^2 P + c_3^2 Q - (c_1 P + c_3 Q)^2]/n,$$

where $c_1 = 1/(1-2P-Q)$, $c_2 = 1/(1-2Q)$ and $c_3 = \frac{1}{2}(c_1 + c_2)$. Again divergence follows a logarithmic function.

In this case you can also determine the rates of substitution via transitions and transversions separately. The rate of transition substitutions per site is

$$s = -(\frac{1}{2})\ln(1-2P-Q) + (\frac{1}{4})\ln(1-2Q)$$

$$Var(s) = [c_1^2 P + c_4^2 Q - (c_1 P + c_4 Q)^2]/n$$

where $c_4 = \frac{1}{2}(c_1 - c2)$. The rate of transversion substitutions per site is

$$v = -(\frac{1}{2})\ln(1-2Q)$$

$$Var(v) = c_2^2 Q(1-Q)/n.$$

8.1.4 Tamura - Nei Correction

Hasegawa, Kishino and Yano (1985, J. Mol. Evol. 22:160-174) suggested a model that Tamura and Nei (1993, Mol. Biol. Evol. 10:512-526) have extended. They suggest a model with different rates of transversions β, and transitions as α_1 and α_2 between purines and between pyrimidines respectively. They also consider mutation rates that yield the observed frequency of A, T, C and G (g_A, g_T, g_C, g_G). In this case, it can be shown that the distance is

$$\begin{aligned} D_{TN} = & - (2g_A g_G/g_R)\ln[1-(g_R/2g_A g_G)P_1 - (1/2g_R)Q] \\ & - (2g_T g_C/g_Y)\ln[1-(g_Y/2g_T g_C)P_2 - (1/2g_Y)Q] \\ & - 2(g_R g_Y - (g_A g_G g_Y/g_R) - (g_T g_C g_R/g_Y))\ln[1-(1/2g_R g_Y)Q], \end{aligned}$$

where P_1, P_2, Q are the proportions of transitions between A and G, between T and C, and the proportions of transversions. The variance has also be derived but is very complicated.

Other more complicated corrections are possible. For example, Felsenstein and Hasegawa have developed likelihood methods that find a maximum likelihood estimate of the distance between two sequences with mutation rates estimated from the actual sequences. It has also been demonstrated that such maximum likelihood estimates of distances are much more accurate than log-transform estimates Hoyle and Higgs (2003, Mol. Biol. Evol. 20:1-9)

8.1.5 Uneven spatial distribution of substitutions

A different sort of problem arises if substitutions are not equally spread throughout the sequence. In this case there are some spots that are “hot” - have had many substitutions and other spots that are “cold” - have had few substitutions. Hence some parts of the sequence may require strong correction for multiple substitutions and an excess of transitions/transversions while other parts of the sequence may require only minor correction.

It would be ideal if all spots along a sequence could have their own rate constants. But if this is permitted then there are so many parameters possible that any sort of data or observation could be simply explained by changing to the appropriate set of parameters.

The most common method to deal with this problem is to apply a gamma distribution to the distribution of substitutions along the sequence. The gamma distribution has been chosen because it is mathematically well characterized, it is a simple distribution, it is a continuous distribution, it is non-negative, and it can assume a variety of shapes. The gamma distribution has density

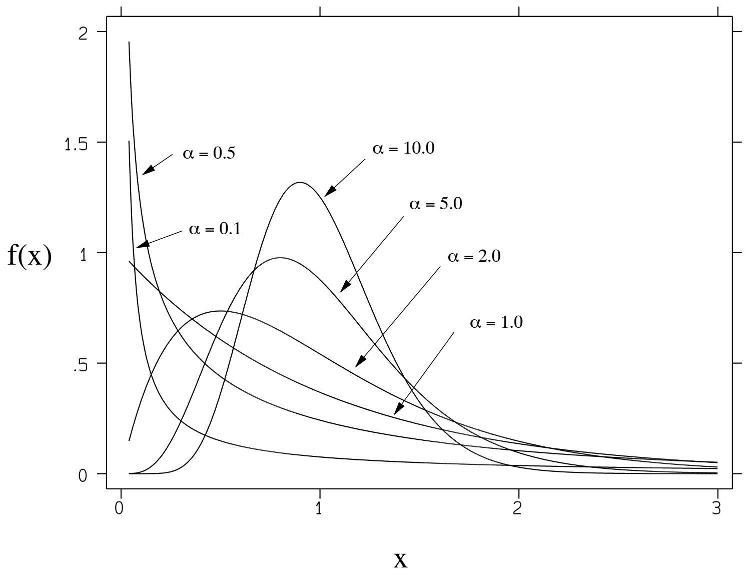


Figure 8.3: Shapes that gamma distributions can take.

$$
\begin{aligned}
f(x) &= [\beta^\alpha/\Gamma(\alpha)]x^{(\alpha-1)}e^{-\beta x}, \quad x>0 \\
&= 0, \quad elsewhere
\end{aligned}
$$

and where

$$\Gamma(\alpha)=\int_0^\infty x^{(\alpha-1)}e^{-x}dx.$$

A plot of gamma distributions is given in Figure 8.3. In distance measures, it is generally used with α set to $\mu^2/Var(\mu)$ and β set equal to α, where μ is the overall rate of substitution. This provides an interpretation such that α is the inverse of the coefficient of variation of substitutions among sites squared. Therefore the smaller the parameter α the higher the extent of variation in substitution rate. The distribution is completely determined by this mean rate of substitution and its coefficient of variation. Therefore only one extra parameter is being used to determine a variety of distributional shapes. Since the mean of the gamma distribution is α/β, the mean will always be one in this case. Thus for each of these variety of distributions the relative rate per site is constant. But unless α is very large, some sites in the sequence will have rates well above the mean and some well below the mean.

All of the distance measures discussed in previous sections can be corrected to include a gamma distribution for the distance measure. For example, the Jukes - Cantor correction becomes

$$D_{\widetilde{JC}}=(\frac{3}{4})\alpha\left[(1-(\frac{4}{3})D)^{-1/\alpha}-1\right],$$

$$Var(D_{\widetilde{JC}})=D(1-D)\left[(1-(\frac{4}{3})D)^{-2(1/\alpha+1)}\right]/n.$$

In general it would be desirable to estimate the value of the gamma parameter α and this can be done easily (given the above interpretation) and it is done by some algorithms that you might run across. However, it is also very common for algorithms that include this correction to simply request a value for α from the user. Studies of many amino acids sequences have suggested that often $\alpha < 2$ and one program package uses a default value of $\alpha = 1$. A typical example of extreme variation would be the $\alpha = 0.47$ that has been noted for some immunoglobulin genes (here much of the variation is probably due to differential selection). Values typical for your own applications will have to be calculated if you are using a program that requests supplied values.

8.1.6 Synonymous - nonsynonymous substitutions

Substitutions that result in amino acid replacements are said to be nonsynonymous while substitutions that do not cause an amino acid replacement (such as a GGG codon to GGC codon change - both codons still encode glycine) are said to be synonymous substitutions. Because of the difference in their effects on the physiology of the organism, synonymous and nonsynonymous substitutions can have quite different dynamics. For example, synonymous substitutions usually occur at a much faster rate than do nonsynonymous substitutions. Hence, for coding sequence it is often desirable to separate these two.

The most common method to estimate these parameters separately is via an algorithm set out by Li, Wu & Luo (1985; Mol. Biol. Evol. 2:150-174). It is somewhat complicated and I refer you to their paper for a complete description. Basically it counts the number of sites that are potentially 4-way, 2-way or 0-way degenerate (the third position of a glycine codon being 4-way degenerate, any second codon position being 0-way degenerate). It then counts the number of differences at each site of each category keeping tract of transversions and transitions. It then calculates

$$K_S \quad \text{and} \quad K_A$$

the rate of synonymous and nonsynonymous substitutions. It has been found that K_A can have large variation and great changes between/within specific organisms. On the other hand, K_S is generally less variable (though still shows more variation than would otherwise be predicted) and shows less changes between/within organisms.

8.2 Amino acid distance measures

Distance is powerful in the sense that it can be used with anything that can be measured. For example, the distance could be based on the strength of an immunological reaction. Using this method any form or measure of distance can be used and different types of measures can be combined into one. Hence, distances can be used with restriction site data, with allozyme data, with data on quantitative characters, with DNA fingerprints or even with real finger fingerprints. Methods to correct this type of data are not well developed because these are not as well defined characteristics.

Even with amino acids, the corrections can not be done easily and/or without some large bias. A Jukes-Cantor correction is possible. It is simply

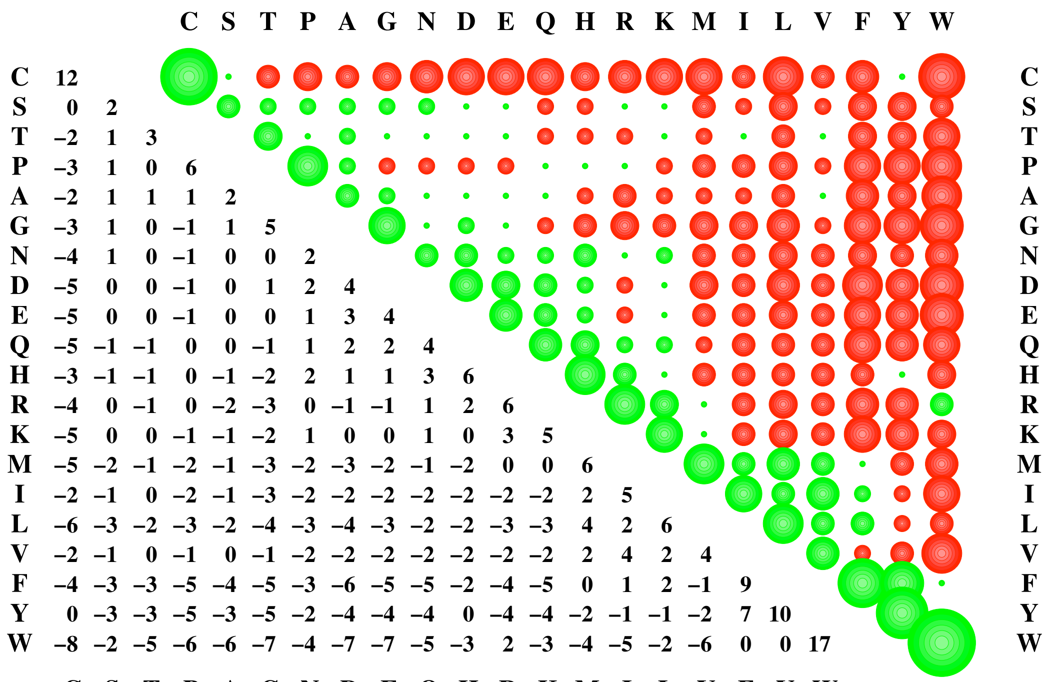
$$D_{JC} = -(\frac{19}{20}) \ln(1 - (\frac{20}{19})D),$$

or more commonly just

$$D_{JC} = -\ln(1 - D).$$

But this assumes (as does the nucleotide Jukes-Cantor correction) that for all characters the rate of substitution from one amino acid and to some other amino acid are equal and independent of the residue. This is not true of DNA and is even less true for proteins. Amino acids like cysteine and proline are very important for the structure and function of proteins. Amino acids such as tryptophan have bulky side groups and can not be inserted easily into any site in a peptide. Because of this most amino acid distances use empirical weighting schemes. The most popular of these empirical measures is the PAM family of matrices.

Table 8.1: The log odds matrix for PAM250 (multiplied by 10). The numbers in the lower left give the log odds. For the diagram in the upper right, green/red circles are proportional to the odds of an interchange more/less likely than chance alone.



	C	S	T	P	A	G	N	D	E	O	H	R	K	M	I	L	V	F	Y	W
C	12																			
S	0	2																		
T	−2	1	3																	
P	−3	1	0	6																
A	−2	1	1	1	2															
G	−3	1	0	−1	1	5														
N	−4	1	0	−1	0	0	2													
D	−5	0	0	−1	0	1	2	4												
E	−5	0	0	−1	0	0	1	3	4											
Q	−5	−1	−1	0	0	−1	1	2	2	4										
H	−3	−1	−1	0	−1	−2	2	1	1	3	6									
R	−4	0	−1	0	−2	−3	0	−1	−1	1	2	6								
K	−5	0	0	−1	−1	−2	1	0	0	1	0	3	5							
M	−5	−2	−1	−2	−1	−3	−2	−3	−2	−1	−2	0	0	6						
I	−2	−1	0	−2	−1	−3	−2	−2	−2	−2	−2	−2	−2	2	5					
L	−6	−3	−2	−3	−2	−4	−3	−4	−3	−2	−2	−3	−3	4	2	6				
V	−2	−1	0	−1	0	−1	−2	−2	−2	−2	−2	−2	−2	2	4	2	4			
F	−4	−3	−3	−5	−4	−5	−3	−6	−5	−5	−2	−4	−5	0	1	2	−1	9		
Y	0	−3	−3	−5	−3	−5	−2	−4	−4	−4	0	−4	−4	−2	−1	−1	−2	7	10	
W	−8	−2	−5	−6	−6	−7	−4	−7	−7	−5	−3	2	−3	−4	−5	−2	−6	0	0	17

8.2.1 PAM Matrices

There are several common ways in which weights can be applied for amino acid differences. Karlin and Ghandour (1985, PNAS 82:8597-8601) proposed a method of weights based on chemical, functional, charge and structural properties of the amino acids. Similarly Doolittle proposed weights based on the structural similarities and the ease of genetic interchange (Feng, Johnson and Doolittle 1985 J. Mol. Evol. 21: 112-125). However, by far the most common and most famous way to assign weights is to use Dayhoff's PAM250 matrix. This is a matrix of weights that is derived from how often different amino acids replace other amino acids in evolution (see M.O. Dayhoff, ed., 1978, Atlas of Protein Sequence and Structure, Vol. 5). This was based on a data base of 1,572 changes in 71 groups of closely related proteins appearing in earlier volumes of this amazing predecessor to electronic databases. PAM stands for percent accepted mutations and these were inferred from the types of changes observed in these proteins. Every change was tabulated and entered in a matrix enumerating all possible amino acid changes.

In addition to these counts of accepted point mutations an idea of the relative mutability of different amino acids were calculated. The information about the individual kinds of mutations and about the relative mutability of the amino acids can be combined into one distance-dependent "mutation probability matrix". The elements of this matrix give the probability that the amino acid in one column will be replaced by the amino acid in some row after a given evolutionary interval. For example, a matrix with an evolutionary distance of 0 PAMs would have ones on the main diagonal and zeros elsewhere. A matrix with an evolutionary distance of 1 PAM would have numbers close to one on the main diagonal and small numbers off the main diagonal. One PAM would correspond to roughly 1% divergence in a protein (one amino acid replacement per hundred). The model of evolution that Dayhoff used assumed that proteins diverged as a result of accumulated, uncorrelated mutations. They treat the $PAM-1$ matrix as a first order Markov chain transition model. To derive a mutational probability matrix for a protein sequence that has undergone N percent accepted mutations, a $PAM-N$ matrix, the $PAM-1$ matrix is multiplied by itself N times. This results in a family of scoring matrices.

By trial and error Dayhoff *et al.* found that for weighting purposes a PAM 250 matrix works well for distant relationships.

At this evolutionary distance (250 substitutions per hundred residues) only one amino acid in five remains unchanged and the percent divergence has increased to roughly 80%. However, the amino acids vary greatly in their mutability. According to Dayhoff *et al.* roughly 55% of the tryptophans, 52% of the cysteines and 27% of the glycines would still be unchanged, but only 6% of the highly mutable asparagines would remain. Several other amino acids particularly alanine, aspartic acid, glutamic acid, glycine, lysine and serine are more likely to occur in place of an original asparagine than asparagine itself at this evolutionary distance.

From this matrix an odds matrix is constructed. This matrix takes the elements of the previous matrix (M_{ij}) and divides each term by the frequency of the replacement residue. Hence, each term now gives the probability of replacement, j to i per occurrence of residue j.

By tradition the $\log_{10}$ of this matrix is used as weights (this is because to calculate the odds for the whole matrix requires taking the product of changes for all sites of the protein. Before calculators it was easier to find the sum of the $\log$'s rather than the product sum). This $\log$ odds PAM 250 matrix is shown in Table 8.1 (also note that amino acids have been sorted according to their similarity in this matrix).

Residue pairs with scores above 0 replace each other more often as alternatives in related sequences than in random sequences. This can be an indication that both residues can carry out similar functions. A score exactly equal to zero indicates amino acid pairs that are found as alternatives at exactly the frequency predicted by chance. Residue pairs with scores less than 0 replace each other less often than in random sequences and might be an indication that these residues are not functionally equivalent.

Some of the properties that are visible from this matrix and go into its makeup are - size, shape, local concentrations of electric charge, conformation of van der Waals surface, ability to form salt bonds, hydrophobic bonds, and hydrogen bonds. Interestingly, these patterns are imposed principally by natural selection and only secondarily by the constraints of the genetic code. This tends to indicate that coming up with your own matrix of weights based on some logical features may not be very successful because your logical features may have been over-written by other more important biological considerations.

Some of the problems with this measure of distance are that it assumes that all sites are equally mutable. But this is clearly false. Another problem is that by examining proteins with few differences, the highly mutable amino acids have been stressed. Lastly, due to the collection of proteins known at that time, the matrix is biased because it is based mainly on small globular proteins.

8.2.2 BLOSUM Matrices

The BLOSUM matrices originate with a paper by Henikoff and Henikoff (1992; PNAS 89:10915-10919). Their idea was to get a better measure of differences between two proteins specifically for more distantly related proteins. While this bias limits the usefulness of BLOSUM matrices for some purposes, for other programs such as FASTA, BLAST, etc. it should do substantially better. This is because the need for an accurate measure of distance is not as great when peptides are more closely related.

They use the BLOCKS database to search for differences among sequences but only among the very conserved regions of a protein family. Hence the term BLOSUM is from BLOcks SUbstitution Matrix. They first collect all of the sequences in the BLOCKS database and then for each one they sum the number of amino acids in each site to get a frequency table ($q_{ij}, \quad i,j = 1..20$) of how often different pairs of amino acids are found together in these conserved regions. Hence the observed frequency of occurrence of one amino acid is

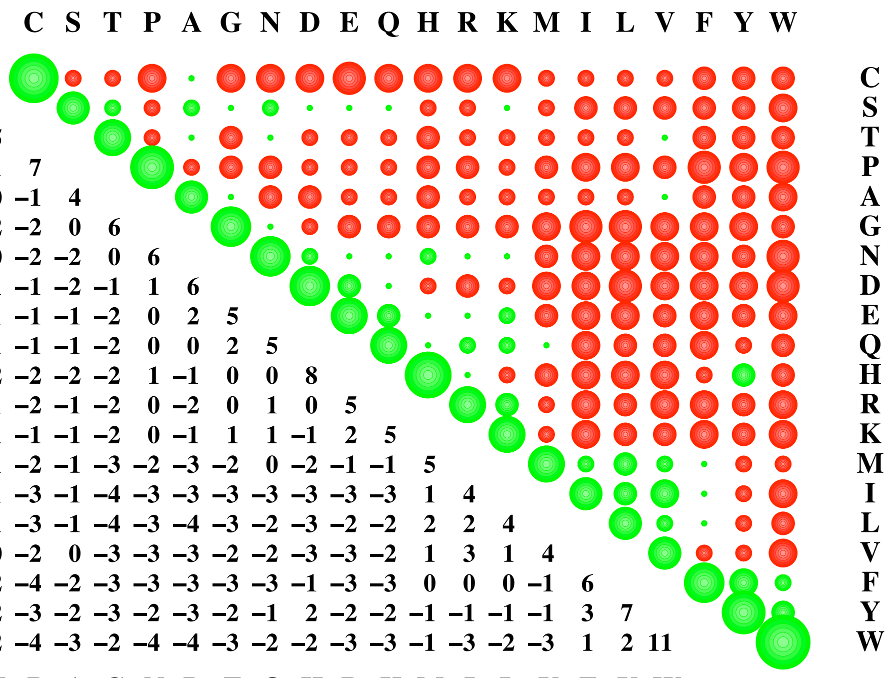
$$p_i = q_{ii} + \sum_{i \neq j} q_{ij}/2$$

Given pairs should occur with expected frequencies

$$e_{ij} = p_i^2, \quad \text{if } i = j$$

and

Table 8.2: The log odds matrix for BLOSUM62. The numbers in the lower left give the logs odds, while in the diagram to the upper right, green/red circles are proportional to the odds of an interchange more/less likely than chance alone.



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

$$e_{ij} = 2p_ip_j, \quad \text{if } i \neq j.$$

The odds matrix is q_{ij}/e_{ij}. Generally log's are taken of this matrix to give a $\log(odds)$ or *lod* matrix such that

$$s_{ij} = 2\log_2(q_{ij}/e_{ij}).$$

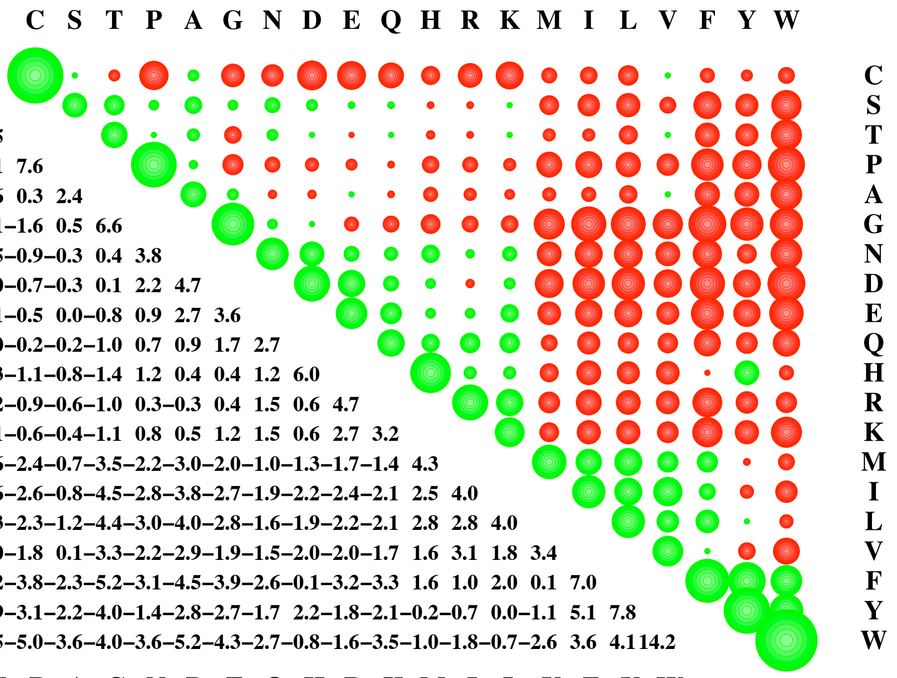
Hence if the observed number of differences between a pair of amino acids is equal to the expected number then $s_{ij} = 0$. If the observed is less than expected then $s_{ij} < 0$ and if the observed is greater than expected $s_{ij} > 0$.

All of this gives the BLOSUM matrix. Different levels of the BLOSUM matrix can be created by differentially weighting the degree of similarity between sequences. Sequences that belong to the same family (within a block) up to a critical level of similarity are clustered so that they are treated as a single entry. For example, a BLOSUM62 matrix is calculated from protein blocks such that if two sequences are more than 62% identical, then the contribution of these sequences is weighted to sum to one. In this way the contributions of multiple entries of closely related sequences is reduced.

The BLOSUM62 matrix is given in Table 8.2. If the BLOSUM62 matrix is compared to PAM160 (it's closest equivalent) then it is found that the BLOSUM matrix is less tolerant of substitutions to or from hydrophilic amino acids, while more tolerant of hydrophobic changes and of cysteine and tryptophan mismatches.

One of the significant disadvantages of the BLOSUM matrices is that they are not Markov chain matrices. Therefore Veerassamy et al. 2003, J. Comput. Biol 10:997-1010 developed a probability transition matrix, based on the BLOSUM matrices, that can be used in a Markov chain model. This is implimented as the PBM model in the PHYLIP package of programs (see below).

Table 8.3: The log odds GONNET matrix. The numbers in the lower left give the log odds, while in the diagram to the upper right, green/red circles are proportional to the odds of an interchange more/less likely than chance alone.



	C	S	T	P	A	G	N	D	E	O	H	R	K	M	I	L	V	F	Y	W
C	11.5																			
S	0.1	2.2																		
T	−0.5	1.5	2.5																	
P	−3.1	0.4	0.1	7.6																
A	0.5	1.1	0.6	0.3	2.4															
G	−2.0	0.4	−1.1	−1.6	0.5	6.6														
N	−1.8	0.9	0.5	−0.9	−0.3	0.4	3.8													
D	−3.2	0.5	0.0	−0.7	−0.3	0.1	2.2	4.7												
E	−3.0	0.2	−0.1	−0.5	0.0	−0.8	0.9	2.7	3.6											
Q	−2.4	0.2	0.0	−0.2	−0.2	−1.0	0.7	0.9	1.7	2.7										
H	−1.3	−0.2	−0.3	−1.1	−0.8	−1.4	1.2	0.4	0.4	1.2	6.0									
R	−2.2	−0.2	−0.2	−0.9	−0.6	−1.0	0.3	−0.3	0.4	1.5	0.6	4.7								
K	−2.8	0.1	0.1	−0.6	−0.4	−1.1	0.8	0.5	1.2	1.5	0.6	2.7	3.2							
M	−0.9	−1.4	−0.6	−2.4	−0.7	−3.5	−2.2	−3.0	−2.0	−1.0	−1.3	−1.7	−1.4	4.3						
I	−1.1	−1.8	−0.6	−2.6	−0.8	−4.5	−2.8	−3.8	−2.7	−1.9	−2.2	−2.4	−2.1	2.5	4.0					
L	−1.5	−2.1	−1.3	−2.3	−1.2	−4.4	−3.0	−4.0	−2.8	−1.6	−1.9	−2.2	−2.1	2.8	2.8	4.0				
V	0.0	−1.0	0.0	−1.8	0.1	−3.3	−2.2	−2.9	−1.9	−1.5	−2.0	−2.0	−1.7	1.6	3.1	1.8	3.4			
F	−0.8	−2.8	−2.2	−3.8	−2.3	−5.2	−3.1	−4.5	−3.9	−2.6	−0.1	−3.2	−3.3	1.6	1.0	2.0	0.1	7.0		
Y	−0.5	−1.9	−1.9	−3.1	−2.2	−4.0	−1.4	−2.8	−2.7	−1.7	2.2	−1.8	−2.1	−0.2	−0.7	0.0	−1.1	5.1	7.8	
W	−1.0	−3.3	−3.5	−5.0	−3.6	−4.0	−3.6	−5.2	−4.3	−2.7	−0.8	−1.6	−3.5	−1.0	−1.8	−0.7	−2.6	3.6	4.1	14.2

8.2.3 GONNET Matrix

A different method to measure differences among amino acids was developed by Gonnet, Cohen and Benner (1992; Science 256:1443-1445) using exhaustive pairwise alignments of the protein databases as they existed at that time. They used classical distance measures to estimate an alignment of the proteins. They then used this data to estimate a new distance matrix. This was used to refine the alignment, estimate a new distance matrix and so on iteratively. They noted that the distance matrices (all first normalized to 250 PAMs) differed depending on whether they were derived from distantly or closely homologous proteins. They suggest that for initial comparisons their resulting matrix should be used in preference to a PAM250 matrix, and that subsequent refinements should be done using a PAM matrix appropriate to the distance between proteins.

Their matrix is given in Table 8.3 and has been normalized to a PAM distance of 250. The matrix elements are ten times the logarithm of the probability that the amino acids are aligned, divided by the probability that these amino acids would be aligned by chance.

In addition they used these alignments to make an estimate of appropriate gap penalties. From this empirical data they suggest that P, the probability of a gap of length k should follow a relation such that

$$10\ln(P) = -36.31 + 7.44\ln(\text{PAM distance}) - 14.93\ln(k).$$

This relation would give the most accurate answer but if the PAM distance is not available, they suggest

$$10\ln(P) = -20.63 - 1.65(k-1).$$

8.3 Gap Weighting

Gap penalties are a field where a great deal more work is required. They are often applied without much justification. Dayhoff suggested using a gap penalty of 6 with PAM250 matrices. Henikoff & Henikoff suggest using a gap penalty of 8 with BLOSUM62 matrices. As noted in the previous section Gonnet, Cohen and Benner suggested yet another gap penalty. There is little reason, other than empirical support, for their choices. The MEGA program package and the PHYLIP program package go to the extreme of ignoring all gaps and any missing data. They do this because there is no accurate way to weight changes due to indels relative to substitutions. Never-the-less, the indels do contain some information but the current challenge is to correctly extract it in a precise manner. In this respect, the approach being taken by Thorne (mentioned in the section on alignments) holds great promise. Barring such a sophisticated approach, it is suggested that you use a variety of gap penalties (from some slight to some significant punishment) and from these determine the effects that this has on your results.

As an example of these problems, consider the following three sequences

```
--GCAAACT
--GCAAGCC
ATGCTAGCC
```

Which pair of sequences has the smallest distance? If gaps are ignored then the second and third sequences are closest with one difference. But if gaps are considered (and if each gapped position is counted as one) then the first and second sequences are closest. If gaps are weighted differently then the answer might depend on the particular weighting.

Chapter 9

Reconstructing Phylogenies

9.1 Introduction

9.1.1 Purpose

The purpose of phylogenetic reconstruction is to attempt to estimate the phylogeny for some data. For any collection of data there will be some ancestral relationship between the sampled sequences. The data itself contains information that can be used to reconstruct or to infer these ancestral relationships. This involves reconstructing a branching structure, termed a phylogeny or tree, that illustrates the relationships between the sequences.

The following discussion is based mainly on Molecular Evolutionary Genetics by M.Nei, Genetic Data Analysis by B.Weir, Of URFs and ORFs by R.Doolittle, Sequence Analysis in Molecular Biology: Treasure Trove or Trivial Pursuit by H.von Gunnar, Molecular Systematics by Hollis & Moritz and J. Felsenstein (1982, Quart.Rev.Biol.57:379). Refer to these for more detailed information.

9.1.2 Trees of what

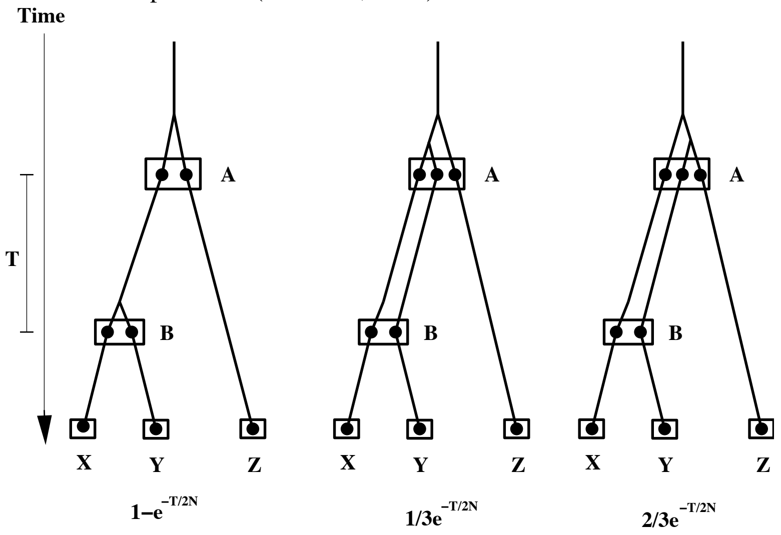
As stated, phylogenetic reconstruction attempts to estimate the phylogeny of some observed data. However, usually people are more interested in using the data to try to infer the species phylogeny and not just the phylogeny of the data. In general, these two are not always the same and estimating the species tree may not possible. Instead what is estimated has been called a "gene tree" by M.Nei. This is because your data (the sequence of some gene or some other form of data) may not have had the same phylogenetic history as the species within which they are contained.

Consider the species shown in Figure 9.1 (from Nei, 1987). The boxes represent the actual species and the dots represent the genes themselves. In the first example, a reconstructed phylogeny based on these genes would yield something similar to the true species tree. In the second example, the reconstructed phylogeny will provide the same topological tree as the species tree but the branch lengths will all be quite incorrect. In the third example, the reconstructed phylogeny will positively give an incorrect phylogeny. It would suggest that species Y and Z are more closely related when in fact X and Y are more closely related.

All of this stems from the fact that polymorphism can exist within species and the estimated age of many polymorphisms can be quite old. The problem of estimating the wrong topology will be greater when the true distance between speciation events A and B is small.

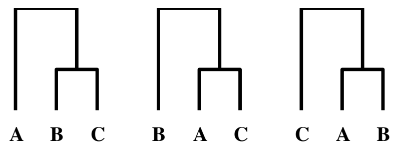
Even if the first situation applies there may still be errors introduced because the number of changes from one species to the next is often a stochastic event and subject to sampling error. Hence unless a large number of sites are examined there can be large errors introduced. In addition, if the gene is part of a multigene family it may be difficult to determine the homologous comparable gene in another species. Horizontal gene transfer and gene conversions from unrelated genes are also assumed not to have occurred.

Figure 9.1: Three possible relationships between species (X, Y, Z) and the genes they contain (indicated by dots) when polymorphism is possible at times of speciation (from Nei, 1987).

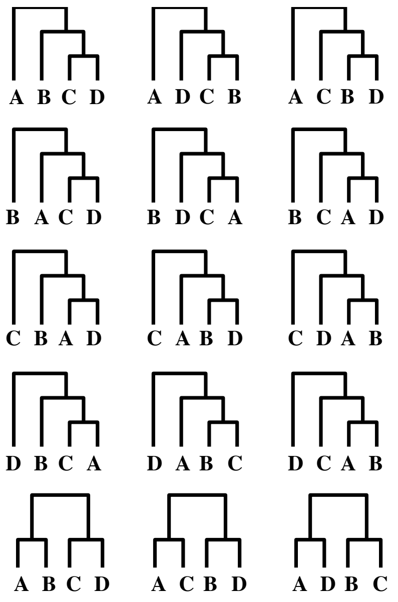


Finally, ignoring all of these caveats, people still usually consider a phylogenetic reconstruction program to act like a “black box”. It takes input, churns around for a while and then spits out the actual phylogenetic answer. This is also incorrect. First, the actual phylogenetic answer can not be obtained by any known method. All methods can only provide estimates and educated guesses of what a phylogenetic tree might look like for the current set of data. These estimates are only as good as the data itself and only as good as the algorithm. Some algorithms in common use are actually quite poor methods. Finally, since these algorithms provide just an estimate, most good methods should also provide an indication of how much variation there is in these estimates.

The problem of tree reconstruction is quite difficult. This is particularly true if all potential tree topologies must be scored or otherwise searched. For three species there are only three trees possible. They are ...



While with four species there are a total of fifteen different topologies possible.



For 5 species there are 105 different topologies. More generally, for any strictly bifurcating phylogeny with n species there are

$$(2n-3)!/(2^{n-2}(n-2)!)$$

different topologies. This number gets large very quickly[1]. With $n = 15$ species there are

$$213,458,046,676,875$$

and with $n = 20$,

$$8,200,794,532,637,891,559,375$$

different trees. Obviously if an algorithm must examine all possible trees, then only a handful of species would be permitted. Even given these, there would be an infinite number of branch length combinations that would have to be searched. Indeed this problem belongs to a class of problems that are called **??**NP-hard by computer scientists.

These numbers apply to phylogenies that are rooted. That is there is a point of origin for this phylogeny and it appears in the standard classical fashion that you are probably most familiar with. A phylogeny may also be presented in an unrooted fashion in which case it is called an unrooted tree or a network. For n species there are only

$$(2n-5)!/(2^{n-3}(n-3)!)$$

different unrooted trees possible (one step behind the number of rooted trees). Any method that purports to provide variable rates of evolution (or substitution) along each branch should generate its output in the form of an unrooted tree. This is because when rates of evolution are free to vary there is no way to determine the location of a root for a tree.

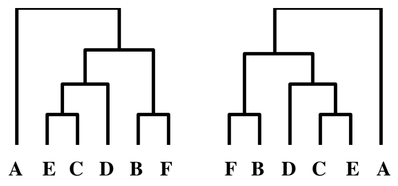
9.1.3 Terminology

There are whole dictionaries that have been created for this field of science (so that people can talk very precisely about what they mean - though this still has not helped to avoid many confusions and useless fights). All of the background

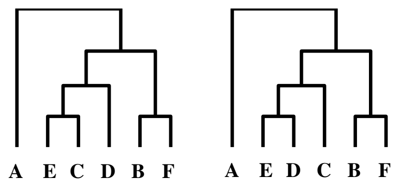
[1]A tree calculator

and terminology necessary can not be provided here. For the present, I simply want to provide you with a subset of the terminology that will enable you to understand some of the problems, some of the methods and to be able to read some of the literature.

The topology of a tree is simply the branching order of the species independent of the branch lengths. Different phylogenies can have the same topology and yet look quite different due to variable branch lengths. If however, branch lengths are all drawn to the same scale then two phylogenies might appear similar whether or not they are identical. Note for example, that



are identical trees while



are not identical.

Because these methods apply equally well to individual samples, to population samples, to species samples and so on, rather than labelling them anyone group it is common practice to label the groups OTU's. These are operational taxonomic units and are meant to represent whatever group of organisms, populations, species, families are under consideration. The individual OTU's or taxa correspond to the terminal nodes of the phylogeny (also called the tips, leaves or external nodes). Places where the interior branches meet are termed internal nodes (also called vertices). An outgroup is an OTU or taxa which is included in the study with the explicit purpose of finding the root of the tree for the remainder of the OTU's. Convergences or parallelisms of a particular character at a site are called homoplasies. These are the characters that usually provide the greatest problems for any tree reconstruction algorithm.

Willi Hennig is the person who began a very systematic approach to phylogeny reconstruction. He was looking mainly at taxonomic characters and was able to show that only shared, derived characters could be used to clearly establish a phylogenetic relationship. He showed that simply having derived characters (i.e. characters that are not ancestral), or having shared ancestral characters are not sufficient to establish a phylogeny. The terminology used for these character states is

- plesiomorphy - a primitive character state.
- apomorphy - a derived character state.
- synapomorphy - a shared derived character state.
- symplesiomorphy - a shared ancestral character state.
- autapomorphy - a unique derived character state.

Hence only synapomorphic characters are useful for determining a phylogeny. As an example, suppose that there is a genus of plants in which one species develops red petals (with the ancestral form being white petals). Suppose it underwent speciation such that there are now two red-petalled species and that there still exist five white-petalled species. Then white petals is the plesiomorphic character, red petals is an apomorphic character, the white petals among the five species is a symplesiomorphic character, the red petals among the two species is a synapomorphic character (and points to these two species as being phylogenetically related). If another species arose with purple petals, this would be an autapomorphic character. Note that this depends on being able to identify the primitive or ancestral state. It is generally not possible to unambiguously determine the direction of change for nucleotide characters (hence some strict adherents would claim that you can not produce cladograms from this data).

In addition, multistate characters can be either ordered or unordered. Nucleotide sequences are considered to be unordered since a "C" for example is not necessarily intermediate between "A" and "G". Again, it is more normal to encounter ordered characters in the analysis of morphological characters. There is also the concept of character polarity which is the assessment of the direction of character change. This most generally involves identifying one character as an ancestral state.

9.1.4 Controversy

Within the field of phylogenetic reconstruction and taxonomy there have, in the past, been two different ways and and two different philosophies to the process of reconstructing a phylogeny. The discussions between these groups about the best ways to proceed have often been acrimonious and counter-productive.

One approach is the phenetic approach. In this approach, a tree is constructed by considering the phenotypic similarities of the species without trying to understand the evolutionary pathways of the species. Since a tree constructed by this method does not necessarily reflect evolutionary relationships but rather is designed to represent phenotypic similarity, trees constructed via this method are called phenograms. A phylogenetic tree based on such information is often termed a dendrogram (a branching order that may or may not be the correct phylogeny).

The second approach is called the cladistic approach. Via these methods, a tree is reconstructed by considering the various possible pathways of evolution and choosing from amongst these the best possible tree. Trees reconstructed via these methods are called cladograms.

The phenetic philosophy as a way to do taxonomy is definitely incorrect. However, this does not mean that phenetic methods are necessarily poor estimates of the cladogram. For character data where ancestral forms are known and to construct a taxonomic classification the cladistic approach is almost certainly superior. However, the cladistic methods are often difficult to implement with assumptions that are not always satisfied with molecular data. The phenetic approaches are generally faster algorithms and often have nicer statistical properties for molecular data. Hence, there appears to be a place for both types of methods in the analysis of molecular sequence data.

9.2 Distance Methods

The archetypical phenetic approach uses distance methods. These methods take the input data and derive from them some measure of similarity/difference between species and from this construct a tree that tries to match this data.

Up until the late 1970's the methods by which the art of taxonomy was performed were never explicitly defined and the relationships between species were determined in an unspecified manner (the value of Willi Hennig's work was not clearly recognized at this time). The field of numerical taxonomy was proposed by Sokal and Sneath (1963) as a way to make the common practices of most taxonomists more rigorous. In the emergence of techniques to perform numerical taxonomy many of the methods that were first applied were based on distance matrices. This is in part, because a small group of numbers are easier to handle computationally. To manipulate the complete data set without the aid of more powerful computers than were available at that time would have been too difficult.

The simplest of the distance methods is a type of cluster algorithm that is known as UPGMA (unweighted pair group method using arithmetic averages). This method has gained popularity mostly because of this simplicity and because of its speed (though many other distance methods are as fast).

Cluster methods are a collection of methods that construct the tree by linking the least distant pairs of taxa, followed by successively more distant taxa. When two taxa are clustered they lose their individual identities and new distances are calculated from the original matrix that correspond to the loss of these two taxa and their replacement by a new joint taxa. At each step of the algorithm the total number of OTUs declines by one and the algorithm is finished when the final two OTUs are clustered. In general these methods only permit bifurcating trees. This is not a limitation since branch lengths can be zero (defining a trifurcation in practice).

This method begins with the construction of a distance matrix (d_{ij}). The two taxa that have the smallest distances are clustered together (assume that this is between the i-th and j-th taxa) and form a new OTU. The branch lengths for the i-th and j-th taxa are taken to be half of the distance between them (hence the depth of the branch between i and j is $d_{ij}/2$). A new distance matrix is constructed that replaces all distances involving the i-th and j-th taxa with the average distance to these two. Thus, for the k-th taxa its distance to the new (i, j) cluster is defined as $(d_{ik} + d_{jk})/2$. The branch length is taken to be the average distance between the OTUs. Then again, the two taxa or OTUs with the smallest distances are clustered together. If the smallest distance were between the k-th taxa and the new (i, j) cluster, the new distance to the l-th taxa is defined as $(d_{il} + d_{jl} + d_{kl})/3$. In general if OTU i and OTU j are to be clustered then the new distance is $d_{k(i,j)} = (T_i d_{ki} + T_j d_{kj})/(T_i + T_j)$ (where T_i is the number of taxa in OTU i). This process continues until all OTUs have been clustered together.

Some data come naturally in the form of a distance between species. For example measures of DNA homology through DNA hybridization / melting curves and measures from immunological data. For other forms of data, the distances are calculated from sequences of characters. The reduction of this data from sequences to a single number obviously leads to a loss of information. But you can gain a great deal from the speed and simplicity of these distance methods.

With distance methods it is generally assumed (whether intended or not) that the sum of the branch lengths in such trees correlates directly with the expected phenotypic distance between taxa and further more that this corresponds to some proportional measure of time. This is generally not a valid assumption. Hence corrections for distances and accurate measures of the distance become very important.

This method obviously assumes that the taxa are all extant and that all rates of change are equal. This is an explicit assumption of the method and yet we know of many examples where rates of evolution vary between taxa. Violation of this assumption will cause the UPGMA algorithm to perform very poorly.

Another very popular distance method is the Neighbour Joining Method (Saitou and Nei 1987, Mol. Biol. Evol. 4:406). This method attempts to correct the UPGMA method for its strong assumption that the same rate of evolution applies to each branch. Hence this method yields an unrooted tree. A modified distance matrix is constructed to adjust for differences in the rate of evolution of each taxon. Similar to the UPGMA method, the least distant pairs of nodes are linked and their common ancestral node is added to the tree, their terminal nodes are pruned from the tree. This continues until only two nodes remain.

The method begins by finding the modified matrix. To do this calculate the net difference of species i from all other taxa as

$$r_i = \sum_k d_{ik}$$

(where $d_{ii} = 0$). Then find the rate-corrected matrix as

$$M_{ij} = d_{ij} - (r_i + r_j)/(n - 2)$$

where n is the number of taxa. Saitou & Nei showed that this equation for M_{ij} (modulo the addition of a constant) is the sum of the least-squares estimates of branch lengths. The next step is to join the two nodes/taxa with the smallest M_{ij} and define the new branch lengths to this node, say u, as

$$l_{iu} = d_{ij}/2 + (r_i - r_j)/(2n - 4)$$

$$l_{ju} = d_{ij} - l_{iu}$$

Next define the new distance from node u to all others as

$$d_{ku} = (d_{ik} + d_{jk} - d_{ij})/2$$

Remove nodes i and j, decrease n by one and recalculate r_i, etc. This continues until only two nodes remain and these two are linked with a branch length of $l_{ij} = d_{ij}$.

Another common pairwise clustering algorithm is that due to Fitch and Margoliash (1967, Science 155: 279). This method yields an unrooted tree and unlike the two previous methods it does not proceed by adding taxa one at a time to a growing tree. Rather it has an optimum criterion that must be met. This method attempts to find that tree which minimizes the following sum

$$\sum (d - d')^2 / d^2$$

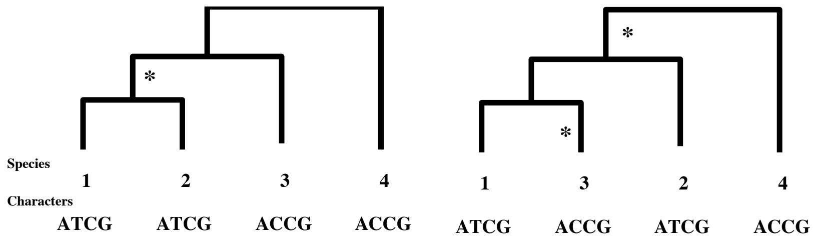
where d is the observed distance and d' is the expected distance given some phylogeny and assuming additivity between all the branch lengths. The details are not given here but are provided in the original paper or in the code and the documentation provided by Dr. Felsenstein.

There are many other methods to reconstruct trees via distance measures. Distance methods are often preferred by people that work in immunology, with frequency data or with data that has some impreciseness in its definition. In addition, almost all of these methods are very rapid and easily permit statistical tests such as bootstraps. These methods loose their accuracy as the number of substitutions goes up and since the correction for multiple substitutions at a single site will loose precision. In this case, the distance methods will increasingly begin to generate less accurate trees. For this reason, with very large trees (where the distance between the most diverged taxa is great) distance methods will do poorly in comparison with methods that are more influenced by local topologies (Rice and Warnow, unpublished).

9.3 Parsimony Methods

Maximum parsimony is perhaps the most popular method for reconstructing ancestral relationships. The method involves evaluating all possible trees (in practice usually only a subset are examined) and giving each a criterion or score that is used to choose between different trees. In maximum parsimony, this criterion is the number of evolutionary changes that need to be postulated in order to explain the observed data with a given tree. The most parsimonious tree is the one with the minimum number of evolutionary changes.

As an example consider the trees shown below.

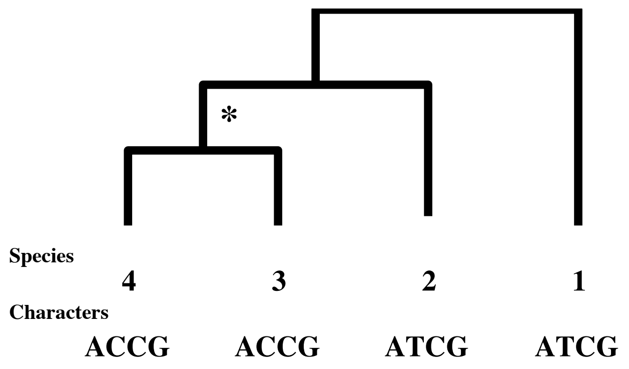


The tree on the left is the most parsimonious tree. It requires only a single evolutionary change (designated by the asterisk) in the second site (a C to T transition). The tree on the right is not as parsimonious. It requires two evolutionary changes. Hence the second tree would be rejected in favour of the first tree.

The principle of the maximum parsimony method is to infer the number of evolutionary events implied by a particular topology and to choose a tree that requires the minimum number of these evolutionary events. In general this means examining a large number of different topologies to search for those that have the minimum changes. For any particular site there are several ways to determine the minimum number of evolutionary events. The Fitch (1971; Syst. Zool. 20:406-416) parsimony criterion is a particularly easy way to count them for nucleotide or amino acid changes. For a particular topology traverse toward the root of the tree. At each node, place the intersection set of the descendant nodes. If this set is empty then place the union set at this node. Continue this for all sites and all nodes. The number of union sets equals the number of events required.

There are several problems with parsimony methods. First note that the most parsimonious tree may not be unique.

Consider the tree



This tree is just as parsimonious as that given above and yet is quite different (so long as only rooted trees are considered). A more serious problem deals with the statistics of these estimators. Suppose that you reconstructed a phylogeny based on a large amount of sequence data and found the most parsimonious tree is one that requires 486 changes. The tree that you prefer (for whatever reason) requires 484 changes. Why and/or when is one phylogeny better than another. This is quite a thorny problem and one of active current research. The best sorts of methods (including methods with algorithms other than parsimony) are those methods that will present you with a whole range of trees that are acceptable via some broad criterion.

Different sites are said to be phylogenetically informative for the parsimony criterion if they provide information that distinguishes between different topologies. Not all sites do this and these sites are, in effect, ignored by the method. Consider characters that are not ordered and can arise via mutation from any other character (such as DNA nucleotides). Then any character that exists uniquely (or locally uniquely) in one OTU is not phylogenetically informative. This is because such a character can always be assumed to have arisen by a single substitution in the immediate branch leading to the OTU in which the character exists. This change is therefore compatible with any topology. A site is phylogenetically informative only when there are at least two different kinds of characters, each represented at least two times. (Remember however, that ALL SITES provide information about the branch lengths - this is true just for the topology).

Note that there are several different kinds of parsimony and the Fitch criterion is only one. As another example, Dollo parsimony is also commonly used. It assumes that derived states are irreversible. That is, a derived character state cannot be lost and then regained. This criterion is most useful when discussing character data other than sequence data. For example if states are complex phenotypes then it is reasonable to assume that these states can evolve only once. Hence, the state can evolve and the state can be lost many times throughout evolution but it cannot be inferred to have evolved twice. An example of such a state in sequence analysis would be restriction sites - these are easier to mutationally lose than to mutationally create. Other parsimony criterion are relaxed Dollo, Wagner, Camin-Sokal, transversion, and generalized.

Many algorithms do not have a series of explicitly stated assumptions required in the derivation of the model and required for its applicability. This is particularly the case with parsimony methods which are often said to be assumption “free”. However, the lack of stated assumptions does not mean that no assumptions are necessary for the method to be valid. The assumptions are implicit rather than explicit.

There is a strong bias in parsimony methods when some lineages have experienced rapid rates of change. While this is true of many methods, parsimony methods are particularly sensitive. In general these long branches tend to “attract” each other. Nor do parsimony methods necessarily lead to “correct” trees (nor, for that matter, does any other method). Prof. Felsenstein provides an example of a comparison between four species. The “true” phylogeny is [(A,B),(C,D)], with A and B most closely related and C and D most closely related. If B and D have a more rapid rate of evolution then parsimony will usually generate a tree with [(A,C),(B,D)], with A and C most closely related and B and D related. Indeed after a certain threshold of differential rates is passed, as more and more data are collected (more and more sequences added to the database), parsimony becomes more and more certain that the “correct” tree is [(A,C),(B,D)]. Hence these methods may not be consistent estimators of the phylogeny. Consistency is a term used in statistics that implies convergence of an estimator to the true answer with increasing amounts of data. A maximum parsimony answer will however, converge to a maximum likelihood answer when the rates of evolution along each branch are small (unfortunately this is not true for most data sets). But then, maximum likelihood methods (and Bayesian methods) need not be consistent either. The arguments as to which method is "best" continue (e.g. Kolaczkowski and Thornton, 2004).

Parsimony does not require exact constancy of rates of change between branches if the number of substitutions per site is small. If the number of changes per site is large then parsimony methods will make serious errors unless rates are constant between branches. Furthermore, if the total sequence length examined is small and there are a large number of backward and parallel substitutions (as in immunoglobulins) then parsimony has a high probability of producing an erroneous tree even when substitution rates are constant between branches. Also, when the number of substitutions per site is small, a large proportion of the substitutions are autapomorphic and uninformative for constructing a parsimonious tree. In this case, a distance method may perform better since it uses all sites to compute distances.

9.4 Other Methods

9.4.1 Compatibility methods

Another class of methods are known as compatibility methods. The compatibility method assumes that the criterion for choosing between phylogenies should be the number of individual characters /sites that are strictly compatible with a given tree. Two characters are compatible if there exists some phylogeny on which both of these characters could evolve without any state having to arise more than once (no homoplasies).

With multi-state characters there have been methods developed to recode the data and to include knowledge of the ancestral states of characters and from this to determine what changes are compatible. Again compatibility methods are more accurate when there are slow rates of evolutionary change. Both compatibility and parsimony, in effect assume that homoplasies will be rare. If you expect homoplasies to be scattered at random throughout the sequence data, then a parsimony method will perform best. If homoplasies are expected to be concentrated in a few characters, whose identities are known in advance, then compatibility will perform better than parsimony. Nei (1987) notes that the compatibility method and parsimony will give the same answer when the number of OTUs is 5 or less.

9.4.2 Maximum Likelihood methods

The method of maximum likelihood attempts to reconstruct a phylogeny using an explicit model of evolution. Certainly, for this given model of evolution, no other method will perform as well nor provide you with as much information about the tree. Unfortunately, this is computationally difficult to do and hence, the model of evolution must be a simple one. Even with simple models of evolutionary change the computational task is enormous and this is the slowest of all methods.

As a typical model of simple evolutionary change consider a single site in a sequence of nucleotides. Let all sites be selectively neutral and let them spontaneously mutate at a rate μ per gamete per generation. For simplicity, let the mutation rates to and from each nucleotide be equal. Generations are assumed to be discrete and the evolution of each site is assumed to be independent of all other sites. (This may seem like a lot of assumptions but in reality the other methods will not work very well without them either).

Given this model an explicit statement can be made about the probability of change from one nucleotide to another within a specified time period. The probability that a site initially with nucleotide i will change to nucleotide j within time t is P_{ij}^t. The value of P_{ij}^t can be found easily as

$$P_{ij}^t = \delta_{ij} e^{-\mu t} + (1 - e^{-\mu t}) g_j$$

where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise and g_j is the equilibrium frequency of nucleotide j.

The likelihood that some site is in state i at the k-th node of an evolutionary tree can be designated by $L_i^{(k)}$. This likelihood can be calculated in a recursive fashion. As an example, consider a simple bifurcating tree with two branches #1 and #2, and with one root node, #3. The time between node #3 and node #1 is t and the time separating node #3 and #2 is t'. With these definitions the likelihood of having the i-th nucleotide at node #3 in an evolutionary tree can be found as

$$L_i^{(3)} = (\sum_{j=1}^{4} P_{ij}^{t} L_j^{(1)})(\sum_{k=1}^{4} P_{ik}^{t'} L_k^{(2)})$$

(Felsenstein, 1981). The terms $L_j^{(1)}$, $L_k^{(2)}$ designate the likelihoods of states j and k in the nodes or taxa #1 and #2. If nodes #1 and #2 designate extant species then these likelihoods are known explicitly. The likelihoods are either 1 or 0 depending on whether the extant species does or does not have that nucleotide at that particular site. In words, you calculate the probability that the descendant would end up having nucleotide j given that t generations in the past it had nucleotide i and multiply this by the likelihood that the descendant had nucleotide j. Sum this for all possible nucleotides and do the same for the other branch on the tree. Take the product of the two to give you the likelihood of the tree up to this point.

This information determines $L_i^{(3)}$. In more complicated phylogenies with more than two species the likelihoods of interior nodes can be calculated in a similar fashion, recursively. In this case identify node #3 with a more ancient bifurcation and nodes #1 and #2 with bifurcations that in turn give rise to more species. Begin at the tips of the phylogeny and move down the tree one node at a time. Each successive step uses the likelihoods just calculated (such as the value determined for $L_i^{(3)}$) to find the likelihood of the next node. The likelihood of every state is calculated for every node using those likelihoods calculated for the previous nodes. This continues until the root of the tree is reached and then the overall likelihood is found by summing the products of the root likelihoods with the prior probabilities of each state. Without any further information, the prior probabilities of each state are usually taken to be their equilibrium frequencies.

Since each site evolves independently, the likelihood of a phylogeny can be calculated separately for each site. The product of the likelihoods for each site provides the overall likelihood of the observed data. To maximize the likelihood different values of ut are analyzed until a set of branch lengths/substitution rates are found which provide the highest likelihood of observing the actual sequences. Finally different tree topologies are searched to find the best one.

Note that a likelihood is not quite the same thing as the probability of observing the given sequences and nor are likelihoods the same thing as a probability. For example, a set of maximum likelihoods need not sum to one. In general, you would normally have the probability of some observed data as a function of some parameter (here the parameters are the branch lengths/substitution rates). The likelihood function turns this relationship around. Instead of considering this to be a set of probabilities for alternative observations given some parameter, it considers the data as fixed and the likelihood as a function of the parameters. For more information on the powerful abilities of likelihood methods consult a text on probability.

9.4.3 Method of Invariants

The method of invariants was originally suggested by Lake (1987) under the name "evolutionary parsimony". This is probably a poor use of terminology since it holds little relationship to parsimony and is a distinct method. Others have termed the approach "invariants".

The idea behind this method is quite simple. Basically there are some patterns which are not functions of branch lengths and depend only on the topology of the tree. These are therefore invariant to the difficulties caused by different rates of evolution along each branch. Lake considered only transversions (since he was originally trying to decipher an ancient branch point and transversions occur less frequently than do transitions) and derived a set of linear invariants for transversions. These are equations for each site that should be zero for the incorrect tree topology. The values for every site are summed and he used a Chi-square to determine those values that differ significantly from zero. This defines the most probable tree topology. The method was originally proposed for four species with just two states (R or Y) per site. This is what will be discussed here. See PHYLIP below for a more general discussion.

With four species there are only three unrooted topologies that need be considered

$$((A,B),(C,D)), \quad ((A,C),(B,D)) \quad \text{and} \quad ((A,D),(B,C)).$$

If say the first tree was the correct tree then a pattern of nucleotides that would support this tree and be phylogenetically informative, would be $((R,R),(Y,Y))$. Due to the stochastic nature of mutations not all sites will have this pattern. Some sites will not support the correct tree. For example $((R,Y),(R,Y))$ supports the second topology above. Lake reasoned that some functions of the "support" for each tree might be found that would depend only on topology and not on branch lengths. The number of sites that have any particular pattern are tabulated. Let X and Y denote two different purines and Z and W denote two different pyrimidines. Then the following equations are Lake's (1987) invariants.

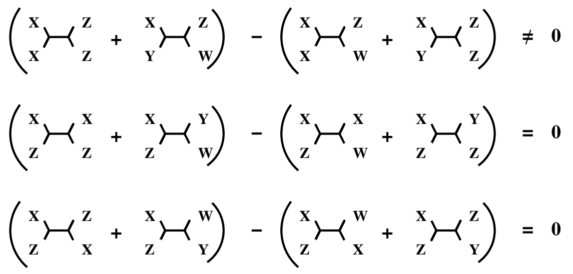
$$(XXZZ + XYZW) - (XXZW + XYZZ)$$

$$(XZXZ + XZYW) - (XZXW + XZYZ)$$

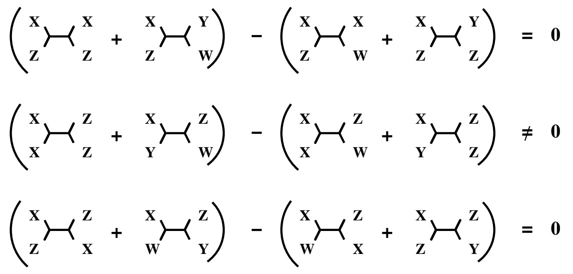
$$(XZZX + XZWY) - (XZWX + XZZY)$$

For whatever the correct topology is, one of these equations should be different from zero and the other two should be equal to zero (or close to it due to random events). The significance of all of the scores can be tested via a Chi-square or via an exact binomial test.

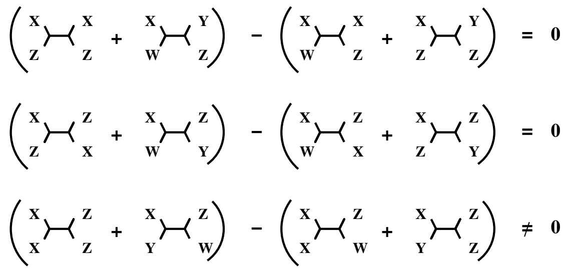
For the topology $((A, B), (C, D))\ldots$


X X Z Z + X Y Z W − X X Z W + X Y Z Z ≠ 0
X Z X Z + X Z Y W − X Z X W + X Z Y Z = 0
X Z Z X + X Z W Y − X Z W X + X Z Z Y = 0

For the topology $((A, C), (B, D))\ldots$


X Z X Z + X Z Y W − X Z X W + X Z Y Z = 0
X X Z Z + X Y Z W − X X Z W + X Y Z Z ≠ 0
X Z Z X + X W Z Y − X W Z X + X Z Z Y = 0

For the topology $((A, D), (B, C))\ldots$

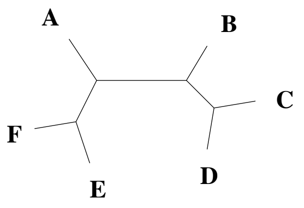

X Z X Z + X W Y Z − X W X Z + X Z Y Z = 0
X Z Z X + X W Z Y − X W Z X + X Z Z Y = 0
X X Z Z + X Y Z W − X X Z W + X Y Z Z ≠ 0

While this method was suggested for only four species there have been several extensions suggested to develop it further and apply it to more than four species. Similarly there are extensions to consider not only transversions but to consider in complete generality all four nucleotides. Steel (1995), Fu (1995), Cavender (1991), Felsenstein (1991), and Sankoff (1990), have developed quadratic and higher order invariants (or in Sankoff's words "made to order invariants"). These extensions promise that invariants will be a very useful tool in the future since these methods are dependent only on the branching order.

9.4.4 Quartet Methods

In principle if the taxa from a tree are reduced to four taxa trees, the original tree can be reconstructed from these quartets. This is the idea behind a collection of quartet methods. Quartet puzzling was suggested in a paper by Strimmer and von Haeseler (1996; Mol. Biol. Evol. 13:964-969). Their method made use of a maximum likelihood algorithm to construct the individual quartets but any algorithm that is preferred can be used for this step. Because there are only three possible trees for four species, the total number of trees that need be constructed are only $3 \times \binom{n}{4}$ for n taxa. This step is therefore quite feasible even for large n. For example, with $n = 20$ there are only $3 \times \binom{20}{4} = 14535$ trees that need be constructed.

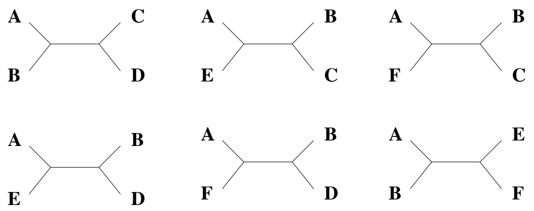
In principle, these quartets should uniquely determine the topology of the tree. For a tree with six taxa



there are $\binom{6}{4} = 15$ quartets. These are with taxa

ABCD	ABCE	ABCF	ABDE	ABDF
ABEF	ACDE	ACDF	ACEF	ADEF
BCDE	BCDF	BCEF	BDEF	CDEF

The quartets corresponding to the above tree are



and so on.

In practice the various quartets may not agree with other and some method must be chosen to weight their suggested topologies. Strimmer and von Haeseler (1996) used a method that weighted the three topologies (1,0,0), (0,1,0) or (0,0,1). They then choose four taxa at random, and began adding taxa one at a time to this four taxa tree according to the quartets. As an example, if there are four taxa (A, B, C, D) initially and if taxa E has a quartet such that ((A,B),(C,E)) then E should not be placed on branch leading to A or B. If it has a quartet ((A,D),(B,E)) then it should not be placed on a branch leading to A or D. Running through all quartets containing E a score is kept for all branches and the branch point with the minimum score is chosen as the branch point to place taxa E. The next taxa is chosen and treated in the same way and then the next taxa.

The order in which the taxa are added and the initial taxa chosen to start the process will critically influence the resulting tree. To prevent any bias due to the order, this whole process is done multiple times with random choices for the order of taxa. A majority rule consensus tree is then chosen as the final tree. This also means that a measure of variability is immediately available in the form of how many times a particular group of taxa branched together. Note that this measure is not the same as a bootstrap value and does not necessarily have the same statistical properties.

The quartet methods are useful for their comparative speed. A maximum likelihood algorithm can be applied with this algorithm to problems that would otherwise not be feasible. As a result, Strimmer and von Haeseler were able to show that this method obtained results as good as neighbor joining when the data was well behaved and results better than neighbor joining when the data had large variations in branch length (a situtation where likelihoods are known to do better). The method performed only slightly worse than Felsenstein's complete maximum likelihood method.

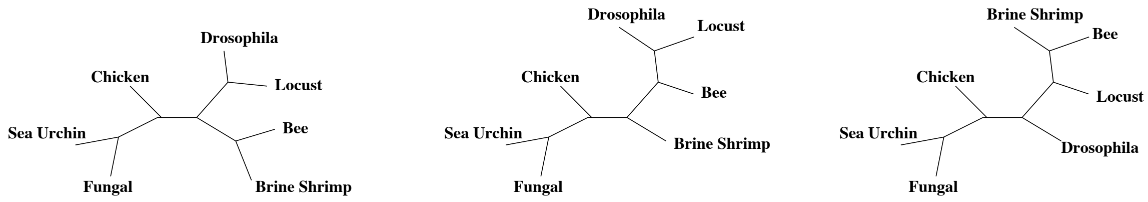


Figure 9.2: A quartet puzzling tree for eight NAD5 mitochondrial proteins. These are three of the 15 trees that quartet puzzling considers unresolved for this data.

Quartet methods are also insteresting in thier ability to separate each of the individual steps and to then easily permit the incorporation of improvements in each step. For example, the original quartets can be constructed via any algorithm. The algorithm know to give the best results for a particular data set can then be chosen (or one known to be most robust under the broadest variety of circumstances). The construction of the tree from the quartets is a completely separate step that can be optimized as well. In a subsequent paper Strimmer, Goldman and von Haeseler (1997; Mol. Biol. Evol. 14:210-211) study the influences of different weights for each quartet and develop a discrete weighting which is both efficient and improves that accuracy of the trees reconstructed. There are similarly a variety of methods possible to reconstruct consensus trees from multiple trees.

9.5 Consensus Trees

The goal of phylogenetics is to reconstruct the true tree reflecting the genetic history of groups of organisms. The true history, however, is unattainable. All we can hope to find are a collection of more or less likely trees and then only for the single gene(s) sequence under consideration. Many methods of reconstructing evolutionary relationships will generate multiple possibilities for this history. Parsimony methods, for example, usually obtain several or even many trees of the same, minimum length. Minimum spanning networks connecting a set of haplotypes regularly define many, often hundreds of equivalent trees.

Uncertainty in phylogenetic relationships also results in multiple trees. An example of multiple trees generated by quartet puzzling is shown in Figure 9.2. Statistical uncertainty arises because any set of sequences are a sample of only a finite number of sites and only one of a number of possible evolutionary events. It is inconceivable that the same process starting from the same ancestral sequence would lead twice to exactly the same result. Methods of reconstructing history must account for all conceivable paths. Reconstruction is therefore inherently probabilistic, leading only to a set of possible histories and the single true history.

Faced with conflicting alternatives, our response (in true Canadian fashion) is usually to reach some sort of consensus. It is perhaps no simpler in phylogenetics than it is in human affairs. There are not one, but many ways to find consensus trees. The PHYLIP package provides a program `consense` that will do strict consensus (a group of species must be present in all species) as well as a family of majority rule consensus tree methods labeled the M_l (M-sub-L) methods. These allow the majority rule consensus tree to consist of any percentage level between 50% and 100%. Thus, groups that occur frequently are merged into a consensus tree until uncertainties in this tree are no longer resolved.

In addition, the default method of consensus in PHYLIP is an extended majority method that finds the 50% majority rule consensus tree and then continues to add groups with a lower frequencies as long as they do not conflict higher frequency groups.

9.6 Bootstrap trees

Assessing the significance of phylogenetic trees has been a controversial problem. One method that has proved useful is the bootstrap. This method of statistical inference was invented by Bradley Efron in 1977. A popular account is given in the article by P. Diaconsis and B. Efron Scientific American 248: 116-130, 1983. Bootstrap statistics do not require

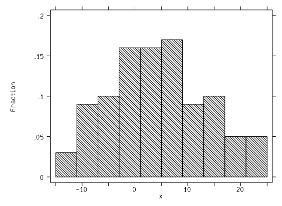


Figure 9.3: A sample of 100 data points from an underlying normal distribution

Table 9.1: Ten samples of bootstrapped data ($n = 100$ in each case)

Sample	Mean	Std.Dev.	5%	95%
1	5.5	9.71	-11.3	21.2
2	4.5	9.39	-9.0	21.2
3	5.1	9.78	-10.5	24.5
4	4.4	9.22	-9.0	22.4
5	4.6	8.92	-9.6	19.0
6	5.5	8.36	-9.6	17.7
7	4.6	9.34	-9.5	22.2
8	3.0	9.45	-11.3	20.8
9	4.0	9.71	-8.5	22.4
10	5.4	8.42	-8.6	20.8

assumptions about the underlying sample distribution. Further, they can be applied to a variety of different properties of samples beyond the traditional mean, variance and correlation.

The concept of a bootstrapped statistic depends on the concept that repeated samples are assumed to produce an accurate distribution of the data if a new data set were collected from the population. Hence, the data itself is assumed to accurately reflect the variation that might be present in the population. By repeatedly sampling (with replacement) the sample itself, you can obtain an understanding of the effect that this level of variation might have on any statistic that you might be interested in. If theoretical knowledge of the statistics is available then it should be used in preference to a bootstrap. If the data itself is biased, then bootstraps tend to exaggerate this bias and again bootstraps should either not be used or corrected for the suspected bias. The bootstrap is advantageous when there is no knowledge of the true statistical properties such as when the underlying distribution is unknown. This is the case in phylogenetic studies.

To illustrate, consider the problem of estimating 95% confidence limits on the mean. Suppose we draw 100 samples from a population with a underlying normal distribution with mean $\mu = 5$ and standard deviation $\sigma = 10$ (a graph of this data is given in Figure 9.3). Due to sample effects, the mean of this data is $\bar{x} = 4.3$ and its standard deviation is $s = 9.15$. To estimate 95% confidence limits for the mean, we then sample with replacement each value of x until a hundred new samples have been drawn. The statistic of interest (mean, median, confidence limits, or whatever else is of interest) is recalculated for this data set and then the whole is process is repeated (here we will repeat it ten times but in practice bootstraps **must** be repeated with much larger sample numbers; on the order of 1000 or more). The statistics from ten potential samples from the original $n = 100$ data set are shown in the accompanying Table 9.1. For each repeated sample statistics are calculated just as one would traditionally with the original data set. These can then be combined to yield bootstrapped estimates. Bootstrap statistics allow an alternate estimate of the 95% confidence interval. For example, in this case, the 10% confidence limit on the mean would be calculated from the values in Table 9.1 and would be 3.0 and the 90% confidence limit on the mean would be 5.5 (of course based on such a limited sample size of 10, neither is a useful estimate).

Felsenstein first recommended applying the bootstrap method to phylogenies. A useful review of this and other methods

Original data set (x)

```
Species          Sites
        abcdefghij ...
   1    ATACCAGCAC ...
   2    ATACCAACAC ...
   3    ATACCGGGAT ...
   4    ATACCCGAAA ...
```

Bootstrap data set (x^1)

```
Species          Sites
        abbjigccfb ...
   1    ATTCAGAAAT ...
   2    ATTCAAAAAT ...
   3    ATTTAGAAGT ...
   4    ATTAAGAACT ...
```

Bootstrap data set (x^2)

```
Species          Sites
        aafegghiaa ...
   1    AAACGGCAAA ...
   2    AAACAACAAA ...
   3    AAGCGGGAAA ...
   4    AACCGGAAAA ...
```

Figure 9.4: Bootstraped data sets are made by randomly sampling nucleotide (or amino acid) sites with replacement. Some sites may, therefore, be omitted altogether from some of the bootstrap samples.

of assessing the reliability of phylogenies is given in J. Felsenstein Annu. Rev. Genet. 22: 521-565, 1988. Again, the idea is resample the sequence data (with replacement) site by site to construct a new sequence data set of the same length as the original and then to estimate a set of bootstrap trees. The original sequences (x) are used to estimate a distance matrix (D) by some method that measures differences between sequence pairs. This distance matrix is converted into a tree by an algorithm that connects sequence pairs into an unrooted, bifurcating tree (T). Alternatively the tree (T) can be obtained directly from the sequences by parsimony (or your method of choice). Felsenstein's method is to randomly sample sites (columns of sequence set x) from the sequence data with replacement to form a bootstrap data set (x^y; Figure 9.4). The original algorithm is then applied to this data to yield a bootstrap tree (T^y). Repeated bootstrap samples yield a set of bootstrap trees T^{1-n}. These trees are derived from sequences containing representative sites sampled from the actual data. The assumption of this method is that the sampled sites are independent of one another and representative of what the evolutionary process would produce if repeated.

There are many ways in which the bootstrap set of trees could be used to answer questions about significance. Felsenstein suggested that the significance of phylogenetic relationships could be assessed from their frequency of occurrence in the bootstrap set $T^1, T^2, \ldots T^n$. More specifically, suppose one wanted to know if a subgroup of taxa (called G) were monophyletic (exclusively comprised of descendants of a common ancestor). We determine the fraction of trees in the bootstrap set $T^1, T^2, \ldots T^n$ in which G is, in fact, monophyletic (call this F_G). Obviously if F_G is small there is little support form monophyly while if F_G is close to 100% we feel that a monophyletic grouping is more likely. Felsenstein

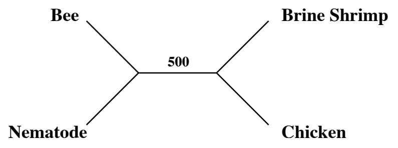


Figure 9.5: The consensus of 500 bootstrapped trees based on NAD5 gene sequences

recommended that $F_G \geq 95\%$ be considered as significant support of a monophyletic relationship.

Analysis of NAD5 mitochondrial genes illustrates this method of using bootstrap trees to determine phylogenetic significance. NAD5 is one of the larger proteins encoded within the mitochondria and hence it is easy to obtain the DNA and to sequence this gene. Phylogenetic relationships for four diverged examples of the NAD5 sequences were determined. Bootstrap DNA sequences (PHYLIP: seqboot) were used to determine pairwise distance matrices (PHYLIP: dnadist, Kimura 2-parameter, $Ts/Tv = 2.0$) and the Fitch-Margoliash, least squares method (PHYLIP: fitch) for merging taxa was then used to make a set of 500 bootstrap trees. The consensus of all 500 of these trees grouped the brine shrimp with the chicken and grouped the bee with the nematode (Figure 9.5). Hence, in this case the support is clear and unequivocal for this one of the three possible relationships.

Although bootstrap methods are widely used, there is considerable debate over the measurement of the actual level of statistical validity. Many have criticized that the 95% criterion is too conservative and accept smaller F_C values (e.g. $\geq 80\%$) as indicating significant monophyletic grouping. Support for this less conservative interpretation of F_G comes from simulation and theoretical work. However, some considerations in favor of a more conservative approach are the following.

1. The consensus tree itself is often used to suggest monophyletic groups. Thus, these are not tested a priori. It is possible that phylogenetically uninformative data may sometimes generate groupings by chance. Since it is not clear how often this could happen, it is best to be conservative and demand a larger value of F_G.

2. The conensus tree effectively tests all possible groups formed from the set of taxa. A few such groups could reach high F_G just by chance.

3. If the group G is actually monophyletic, the value of F_G estimates the probability that it would appear as monophyletic in sequence data of the type obtained. It is not a confidence interval. It determines the stability of the phylogenetic relationship if more sequence data of the same type were to be accumulated.

4. Bootstrap statistics are assumed to vary smoothly and continuously in sample space. However, a taxonomic group can either be monophyletic or not monophyletic. Methods to extend bootstrap methods to bivariate statistics have not been developed.

5. The bootstrap method assumes that sites are independent and that a sufficient number have been sampled to give a complete representation of the evolutionary process.

Finally, the bootstrap trees are only as good as the method that generates them. Again, if the method is biased, the trees will not be representative examples of evolution. In particular, the trees generated for the *NAD5* genes are certainly biased (and were purposely used to illustrate this point – no, bees and nematodes are not close relatives). The DNA distance method not only did not account for rate variation among sites (e.g. first, second and third codon positions), but also ignored large base composition differences between sequence pairs. The substitution model for DNA distance assumes that all sequences are subject to the same mutational forces leading to equilibrium nucleotide frequencies of 25% for each of the four nucleotides. This is certainly not the case for the *NAD5* genes. Thus the method artificially makes sequences of similar nucleotide composition (e.g. bee and nematode) closer because the expected number of substitutions is underestimated. The significant grouping of these taxa is entirely a result of this bias. Indeed, otherwise this would provide strong evidence that the brine shrimp and the bee do not form a monophyletic grouping commonly known as arthropods. Steel *et al.* Nature 364: 440-442, 1993 have discussed a randomization test that corrects for such nucleotide biases and can be used to show that the evidence for the association displayed in Figure 9.5 is due to the nucleotide bias and is not an accurate phylogenetic reconstruction.

9.7 Warnings

Remember that each of these methods have their advantages and their disadvantages. They provide estimates of what the phylogenetic history of the sample may be like - they do not provide "truth". When you run an algorithm for your data set, consider this simply as the starting point of your analysis.

There are several approaches that can be taken to begin an in depth phylogenetic analysis of your data. These are a few suggestions but they are not exhaustive - for different data sets additional steps should be taken.

1. The first rule that should be followed is to apply several different algorithms to your data set. Each one will provide a different picture of the phylogenetic history reflecting the assumptions of the methods.
2. Your data should be bootstrapped or jackknifed to sample your data. These are techniques to create new data sets either by sampling with replacement from the original set or by successively dropping individual data points. They will help to determine how sensitive the phylogenetic history is to changes in the data set (preferably the data should also be aligned all over again with the bootstrapped data). (The actual statistics for these cases are non-standard and difficult to calculate but it will provide a rough measure of variability).
3. If the data and tree inference technique were ideal, analysing any two subsets of taxa would yield congruent trees (i.e., the trees would be identical after pruning taxa absent from one or both trees). Try this and see what happens for different subsets.
4. In this regard, if the tree changes dramatically when a single OTU is dropped this is usually an indication that that OTU is causing systematic errors (such as would be caused by a significantly different rate of change).
5. Worry about long unbranched lineages and any subtrees on either side of long branches. Long branches tend to attract each other !!!
6. Remember that these are gene trees and hence the trees from different genes may or may not be the same. If your taxa are each sufficiently diverged then the trees should be similar. If not then check for non-orthologous genes, check for lateral gene transfer or for other events that would cause systematic errors.
7. Always include more than one outgroup taxa. In this way you can check that the outgroups are indeed "out".
8. If possible choose your outgroup species such that they are evenly spaced on the tree. You will obtain more reliable information from these. Two outgroups that are closely related to each other will not add much information.
9. Even if you are interested in the relationships of just a few taxa it is best to include as many intermediate taxa as possible. These will help to highlight the multiple substitutions that confound any analysis.
10. Others have suggested that because large branch lengths confound many methods, one should limit an analysis to those sequence regions that exclude the most variable positions. (I personally disagree with this rule of thumb but hey ..!)

9.8 Available Packages

The following section of these notes will provide you with some background to the package of programs distributed by Dr. Felsenstein - the PHYLIP package. These are excellent programs, work on any platform, they are free (!) and they are easily obtained.

They are not however unique. A list of other phylogenetic reconstruction programs is maintained by Dr. Felsenstein and parts of it are reproduced here. A listing from Prof. Felsenstein's PHYLIP (see http://evolution.genetics.washington.edu/phylip.html) homepage includes

- General-purpose packages
  - PHYLIP
  - PAUP*
  - MEGA
  - VOSTORG
  - Fitch programs
  - Phylo_win
  - ARB
  - DAMBE
  - PAL
  - Bionumerics
- Parsimony programs
  - PAUP*
  - Hennig86
  - MEGA
  - Tree Gardener
  - RA
  - Nona
  - PHYLIP
  - TurboTree
  - Freqpars
  - Fitch programs
  - CAFCA
  - Phylo_win
  - sog
  - gmaes
  - LVB
  - GeneTree
  - TAAR
  - ARB
  - DAMBE
  - MALIGN
  - POY
  - DNASEP
  - SEPAL
  - Gambit
  - TNT
  - GelCompar II
  - Bionumerics
  - TCS
- Distance matrix methods
  - PHYLIP
  - PAUP*
  - MEGA
  - MacT
  - ODEN
  - Fitch programs
  - ABLE
  - TREECON
  - DISPAN
  - RESTSITE
  - NTSYSpc
  - METREE
  - TreePack
  - TreeTree
  - GDA
  - Hadtree, Prepare and Trees
  - GCG Wisconsin Package
  - SeqPup
  - PHYLTEST
  - Lintre
  - WET
  - Phylo_win
  - njbafd
  - Gambit
  - gmaes
  - DENDRON
- Molecular Analyst Fingerprinting
  - BIONJ
  - TFPGA
  - MVSP
  - SOTA
  - ARB
  - BIOSYS-2
  - Darwin
  - T-REX
  - sendbs
  - nneighbor
  - DAMBE
  - weighbor
  - QR2
  - DNASIS
  - minspnet
  - PAL
  - Arlequin
  - vCEBL
  - HY-PHY
  - Vanilla
  - GelCompar II
  - Bionumerics
  - qclust
  - TCS
- Computation of distances
  - PHYLIP
  - PAUP*
  - RAPDistance
  - MULTICOMP
  - MARKOV
  - RSVP
  - Microsat
  - DIPLOMO
  - OSA
  - DISPAN
  - RESTSITE
  - NTSYSpc
  - TREE-PUZZLE
  - Hadtree, Prepare and Trees
  - GCG Wisconsin Package
  - AMP
  - GCUA
  - DERANGE2
  - POPGENE
  - TFPGA
  - REAP
  - MVSP
  - SOTA
  - RSTCALC
  - Genetix
  - BIOSYS-2
  - RAPD-PCR package
  - DISTANCE
  - Darwin
  - sendbs
  - K2WuLi
  - GeneStrut

  - Arlequin
  - DAMBE
  - DnaSP
  - PAML
  - puzzleboot
  - MATRIX
  - PAL
  - Sequencer
  - Vanilla
  - GelCompar II
  - Bionumerics
  - qclust

- Maximum likelihood and related methods
  - PHYLIP
  - PAUP*
  - fastDNAml
  - MOLPHY
  - PAML
  - Spectrum
  - SplitsTree
  - PLATO
  - TREE-PUZZLE
  - Hadtree, Prepare and Trees
  - SeqPup
  - Phylo_win
  - PASSML
  - ARB
  - Darwin
  - BAMBE
  - DAMBE
  - Modeltest
  - TreeCons
  - VeryfastDNAml
  - PAL
  - dnarates
  - TrExMl
  - HY-PHY
  - Vanilla
  - MEGA
  - Bionumerics
  - fastDNAmlRev
  - RevDNArates
  - rate-evolution
  - MrBayes
  - Hadtree, Prepare and Trees
  - CONSEL

- Quartets methods
  - TREE-PUZZLE
  - STATGEOM
  - SplitsTree
  - PHYLTEST
  - GEOMETRY
  - PICA95
  - Darwin
  - PhyloQuart
  - Willson quartets programs
  - Gambit

- Artificial-intelligence methods
  - SOTA

- Invariants (or Evolutionary Parsimony) methods
  - PHYLIP
  - PAUP*

- Interactive tree manipulation
  - MacClade
  - PHYLIP
  - PDAP
  - TreeTool
  - ARB
  - WINCLADA
  - TreeEdit
  - UO
  - TreeExplorer
  - TreeThief
  - RadCon
  - Mavric

- Looking for hybridization or recombination events
  - PLATO
  - Bootscanning Package
  - TOPAL
  - reticulate
  - RecPars
  - partimatrix
  - homoplasy test
  - LARD
  - Network
  - TCS

- Bootstrapping and other measures of support
  - PHYLIP
  - PAUP*
  - PARBOOT
  - ABLE
  - Random Cladistics
  - AutoDecay
  - TreeRot
  - RASA
  - DNA Stacks
  - OSA
  - DISPAN
  - TreeTree
  - PHYLTEST
  - Lintre
  - sog
  - njbafd
  - MEGA
  - PICA95
  - ModelTest
  - TAXEQ2
  - BIOSYS-2
  - RAPD-PCR package
  - TreeCons
  - BAMBE
  - DAMBE
  - puzzleboot
  - CodonBootstrap
  - DNASEP
  - SEPAL
  - Gambit
  - MEAWILK
  - TrExMl
  - Sequencer

  - PAL
  - PHYCON
  - MrBayes
  - CONSEL

- Compatibility analysis

  - COMPROB
  - PHYLIP
  - PICA95
  - reticulate
  - partimatrix
  - SECANT
  - CLINCH
  - MEAWILK

- Consensus trees and distances between trees

  - COMPONENT
  - TREEMAP
  - NTSYSpc
  - PHYLIP
  - PAUP*
  - REDCON
  - TAXEQ2
  - TreeCons
  - QUARTET2
  - RadCon

- Tree-based sequence alignment

  - TreeAlign
  - ClustalW
  - MALIGN
  - GeneDoc
  - GCG Wisconsin Package
  - TAAR
  - Ctree
  - DAMBE
  - POY
  - ALIGN
  - DNASIS

- Biogeographic analysis and host-parasite comparison

  - COMPONENT
  - TREEMAP

- Comparative method analysis

  - PHYLIP
  - CAIC
  - COMPARE
  - PA
  - CMAP
  - CoSta
  - PDAP
  - ACAP
  - ANCML
  - RIND
  - MacroCAIC
  - Fels-Rand
  - Phylogenetic Independence

- Simulation of trees or data

  - COMPONENT
  - Bi-De
  - SEQEVOLVE
  - TheSiminator
  - Seq-Gen
  - Treevolve and PTreevolve
  - PSeq-Gen
  - COMPARE
  - ROSE
  - PAML
  - ProSeq
  - PAL
  - Vanilla

- Examination of shapes of trees

  - End-Epi
  - MacroCAIC
  - Genie
  - PAL
  - Vanilla
  - RadCon
  - BRANCHLENGTH

- Clocks, dating and stratigraphy

  - StratCon
  - QDate
  - Diversi
  - K2WuLi
  - Modeltest
  - PAML
  - TipDate
  - RRTree
  - vCEBL
  - TreeEdit
  - HY-PHY
  - PAL
  - rate-evolution
  - BRANCHLENGTH

- Description or prediction of data from trees

  - CONSERVE
  - TreeDis

- Tree plotting/drawing

  - PHYLIP
  - PAUP*
  - TreeTool
  - TreeView
  - Fitch programs
  - NJplot
  - DendroMaker
  - Tree Draw Deck
  - Phylodendron
  - ARB
  - unrooted
  - DAMBE
  - TREECON
  - Mavric
  - TreeExplorer
  - TreeThief
  - Bionumerics

- Sequence management/job submission

  - PARBOOT
  - Random Cladistics
  - Tree Gardener
  - GDE
  - MUST
  - DNA Stacks
  - SeqPup
  - ARB
  - BioEdit
  - Singapore PHYLIP web interface
  - PHYCON
  - Bionumerics

- Teaching about phylogenies
  - Phylogenetic Investigator

9.9 PHYLIP

This is the package of programs distributed by Professor Felsenstein. It is distributed free and Joe is a very friendly character and can help with whatever problem you might have (but carefully read the documentation before contacting him). I have reproduced parts of the documentation here but I urge you to get your own copy of the programs so that Dr. Felsenstein can know how many copies are out there and can update/modify programs etc. Also if you do use these programs in a publication you must quote Dr. Felsenstein (the same applies for any other program obtained from the file servers). Remember that the value of a scientist's work is often measured by quotations and if you use someone's programming work you should quote it just as you would quote their experimental work.

The PHYLIP package is distributed for free. Programs are written in a standard subset of "C" and the source code is provided with the package. You can reach Dr. Felsenstein at joe@genetics.washington.edu and the complete package can be obtained via anonymous ftp from evolution.genetics.washington.edu.

9.9.1 PHYLIP Contents

On the following pages you will find extracts of the documentation for the PHYLIP package of programs. The complete documentation is not reproduced - you should get your own official copy.

```
                                      What The Programs Do

Here is a short description of each of the programs. For more detailed
discussion you should definitely read the documentation file for the
individual program and the documentation file for the group of programs
it is in. In this list the name of each program is a link which will
take you to the documentation file for that program. Note that there is
no program in the PHYLIP package called PHYLIP.

PROTPARS
      Estimates phylogenies from protein sequences (input using the
      standard one-letter code for amino acids) using the parsimony
      method, in a variant which counts only those nucleotide changes
      that change the amino acid, on the assumption that silent changes
      are more easily accomplished.

DNAPARS
      Estimates phylogenies by the parsimony method using nucleic acid
      sequences. Allows use the full IUB ambiguity codes, and estimates
      ancestral nucleotide states. Gaps treated as a fifth nucleotide
      state. Can use 0/1 weights, reconstruct ancestral states, and
      infer branch lengths.

DNAMOVE
```

Interactive construction of phylogenies from nucleic acid sequences, with their evaluation by parsimony and compatibility and the display of reconstructed ancestral bases. This can be used to find parsimony or compatibility estimates by hand.

DNAPENNY

Finds all most parsimonious phylogenies for nucleic acid sequences by branch-and-bound search. This may not be practical (depending on the data) for more than 10 or 11 species.

DNACOMP

Estimates phylogenies from nucleic acid sequence data using the compatibility criterion, which searches for the largest number of sites which could have all states (nucleotides) uniquely evolved on the same tree. Compatibility is particularly appropriate when sites vary greatly in their rates of evolution, but we do not know in advance which are the less reliable ones.

DNAINVAR

For nucleic acid sequence data on four species, computes Lake’s and Cavender’s phylogenetic invariants, which test alternative tree topologies. The program also tabulates the frequencies of occurrence of the different nucleotide patterns. Lake’s invariants are the method which he calls "evolutionary parsimony".

DNAML

Estimates phylogenies from nucleotide sequences by maximum likelihood. The model employed allows for unequal expected frequencies of the four nucleotides, for unequal rates of transitions and transversions, and for different (prespecified) rates of change in different categories of sites, with the program inferring which sites have which rates. It also allows different rates of change at known sites.

DNAMLK

Same as DNAML but assumes a molecular clock. The use of the two programs together permits a likelihood ratio test of the molecular clock hypothesis to be made.

PROML

Estimates phylogenies from protein amino acid sequences by maximum likelihood. The PAM or JTTF models can be employed. The program can allow for different (prespecified) rates of change in different categories of amino acid positions, with the program inferring which posiitons have which rates. It also allows different rates of change at known sites.

DNADIST

Computes four different distances between species from nucleic acid sequences. The distances can then be used in the distance matrix programs. The distances are the Jukes-Cantor formula, one based on Kimura’s 2-parameter method, Jin and Nei’s distance which allows for rate variation from site to site, and a maximum likelihood method using the model employed in DNAML. The latter method of computing distances can be very slow.

PROTDIST

Computes a distance measure for protein sequences, using maximum likelihood estimates based on the Dayhoff PAM matrix, Kimura's 1983 approximation to it, or a model based on the genetic code plus a constraint on changing to a different category of amino acid. Rate variation from site to site is also allowed. The distances can be used in the distance matrix programs.

RESTDIST

Distances calculated from restriction sites data or restriction fragments data. The restriction sites option is the one to use to also make distances for RAPDs or AFLPs.

RESTML

Estimation of phylogenies by maximum likelihood using restriction sites data (not restriction fragments but presence/absence of individual sites). It employs the Jukes-Cantor symmetrical model of nucleotide change, which does not allow for differences of rate between transitions and transversions. This program is very slow.

SEQBOOT

Reads in a data set, and produces multiple data sets from it by bootstrap resampling. Since most programs in the current version of the package allow processing of multiple data sets, this can be used together with the consensus tree program CONSENSE to do bootstrap (or delete-half-jackknife) analyses with most of the methods in this package. This program also allows the Archie/Faith technique of permutation of species within characters.

FITCH

Estimates phylogenies from distance matrix data under the "additive tree model" according to which the distances are expected to equal the sums of branch lengths between the species. Uses the Fitch-Margoliash criterion and some related least squares criteria. Does not assume an evolutionary clock. This program will be useful with distances computed from molecular sequences, restriction sites or fragments distances, with DNA hybridization measurements, and with genetic distances computed from gene frequencies.

KITSCH

Estimates phylogenies from distance matrix data under the "ultrametric" model which is the same as the additive tree model except that an evolutionary clock is assumed. The Fitch-Margoliash criterion and other least squares criteria are assumed. This program will be useful with distances computed from molecular sequences, restriction sites or fragments distances, with distances from DNA hybridization measurements, and with genetic distances computed from gene frequencies.

NEIGHBOR

An implementation by Mary Kuhner and John Yamato of Saitou and Nei's "Neighbor Joining Method," and of the UPGMA (Average Linkage clustering) method. Neighbor Joining is a distance matrix method producing an unrooted tree without the assumption of a clock. UPGMA does assume a clock. The branch lengths are not optimized by the least squares criterion but the methods are very fast and thus can handle much larger data sets.

CONTML

Estimates phylogenies from gene frequency data by maximum likelihood under a model in which all divergence is due to genetic drift in the absence of new mutations. Does not assume a molecular clock. An alternative method of analyzing this data is to compute Nei’s genetic distance and use one of the distance matrix programs. This program can also do maximum likelihoodn analysis of continuous charactersn that evolve by a Brownian Motion model, but it assumes that the characters evolve at equal rates and in an uncorrelated fashion, so that it does not take into account the usual correlations of characters.

GENDIST

Computes one of three different genetic distance formulas from gene frequency data. The formulas are Nei’s genetic distance, the Cavalli-Sforza chord measure, and the genetic distance of Reynolds et. al. The former is appropriate for data in which new mutations occur in an infinite isoalleles neutral mutation model, the latter two for a model without mutation and with pure genetic drift. The distances are written to a file in a format appropriate for input to the distance matrix programs.

CONTRAST

Reads a tree from a tree file, and a data set with continuous characters data, and produces the independent contrasts for those characters, for use in any multivariate statistics package. Will also produce covariances, regressions and correlations between characters for those contrasts. Can also correct for within-species sampling variation when individual phenotypes are available within a population.

PARS

Multistate discrete-characters parsimony method. Up to 8 states (as well as "?") are allowed. Cannot do Camin-Sokal or Dollo Parsimony. Can reconstruct ancestral states, use character weights, and infer branch lengths.

MIX

Estimates phylogenies by some parsimony methods for discrete character data with two states (0 and 1). Allows use of the Wagner parsimony method, the Camin-Sokal parsimony method, or arbitrary mixtures of these. Also reconstructs ancestral states and allows weighting of characters (does not infer branch lengths).

MOVE

Interactive construction of phylogenies from discrete character data with two states (0 and 1). Evaluates parsimony and compatibility criteria for those phylogenies and displays reconstructed states throughout the tree. This can be used to find parsimony or compatibility estimates by hand.

PENNY

Finds all most parsimonious phylogenies for discrete-character data with two states, for the Wagner, Camin-Sokal, and mixed parsimony criteria using the branch-and-bound method of exact search. May be impractical (depending on the data) for more than 10-11 species.

DOLLOP

Estimates phylogenies by the Dollo or polymorphism parsimony criteria for discrete character data with two states (0 and 1). Also reconstructs ancestral states and allows weighting of characters. Dollo parsimony is particularly appropriate for restriction sites data; with ancestor states specified as unknown it may be appropriate for restriction fragments data.

DOLMOVE

Interactive construction of phylogenies from discrete character data with two states (0 and 1) using the Dollo or polymorphism parsimony criteria. Evaluates parsimony and compatibility criteria for those phylogenies and displays reconstructed states throughout the tree. This can be used to find parsimony or compatibility estimates by hand.

DOLPENNY

Finds all most parsimonious phylogenies for discrete-character data with two states, for the Dollo or polymorphism parsimony criteria using the branch-and-bound method of exact search. May be impractical (depending on the data) for more than 10-11 species.

CLIQUE

Finds the largest clique of mutually compatible characters, and the phylogeny which they recommend, for discrete character data with two states. The largest clique (or all cliques within a given size range of the largest one) are found by a very fast branch and bound search method. The method does not allow for missing data. For such cases the T (Threshold) option of PARS or MIX may be a useful alternative. Compatibility methods are particular useful when some characters are of poor quality and the rest of good quality, but when it is not known in advance which ones are which.

FACTOR

Takes discrete multistate data with character state trees and produces the corresponding data set with two states (0 and 1). Written by Christopher Meacham. This program was formerly used to accomodate multistate characters in MIX, but this is less necessary now that PARS is available.

DRAWGRAM

Plots rooted phylogenies, cladograms, and phenograms in a wide variety of user-controllable formats. The program is interactive and allows previewing of the tree on PC or Macintosh graphics screens, and Tektronix or Digital graphics terminals. Final output can be to a file formatted for one of the drawing programs, on a laser printer (such as Postscript or PCL-compatible printers), on graphics screens or terminals, on pen plotters (Hewlett-Packard or Houston Instruments) or on dot matrix printers capable of graphics (Epson, Okidata, Imagewriter, or Toshiba).

DRAWTREE

Similar to DRAWGRAM but plots unrooted phylogenies.

TREEDIST

Computes the Robinson-Foulds symmetric difference distance between trees, which allows for differences in tree topology (but does not use branch lengths).

CONSENSE

Computes consensus trees by the majority-rule consensus tree method, which also allows one to easily find the strict consensus tree. Is not able to compute the Adams consensus tree. Trees are input in a tree file in standard nested-parenthesis notation, which is produced by many of the tree estimation programs in the package. This program can be used as the final step in doing bootstrap analyses for many of the methods in the package.

RETREE

Reads in a tree (with branch lengths if necessary) and allows you to reroot the tree, to flip branches, to change species names and branch lengths, and then write the result out. Can be used to convert between rooted and unrooted trees.

OVERVIEW OF THE INPUT AND OUTPUT FORMATS

When you run most of these programs, a menu will appear offering you choices of the various options available for that program. The data that the program reads should be in an input file called (in most cases) "infile". If there is no such file the programs will ask you for the name of the input file. Below we describe the input file format, and then the menu.

Input File Format

I have tried to adhere to a rather stereotyped input and output format. For the parsimony, compatibility and maximum likelihood programs, excluding the distance matrix methods, the simplest version of the input file looks something like this:

```
   6   13
Archaeopt CGATGCTTAC CGC
HesperorniCGTTACTCGT TGT
BaluchitheTAATGTTAAT TGT
B. virginiTAATGTTCGT TGT
BrontosaurCAAAACCCAT CAT
```

```
B.subtilisGGCAGCCAAT CAC
```

The first line of the input file contains the number of species and the number of characters, in free format, separated by blanks (not by commas). The information for each species follows, starting with a ten-character species name (which can include punctuation marks and blanks), and continuing with the characters for that species. In the discrete-character, DNA and protein sequence programs the characters are each a single letter or digit, sometimes separated by blanks. In the continuous-characters programs they are real numbers with decimal points, separated by blanks:

```
Latimeria  2.03  3.457  100.2  0.0  -3.7
```

The conventions about continuing the data beyond one line per species are different between the molecular sequence programs and the others. The molecular sequence programs can take the data in "aligned" or "interleaved" format, with some lines giving the first part of each of the sequences, then lines giving the next part of each, and so on. Thus the sequences might look like this:

```
   6   39
Archaeopt CGATGCTTAC CGCCGATGCT
HesperorniCGTTACTCGT TGTCGTTACT
BaluchitheTAATGTTAAT TGTTAATGTT
B. virginiTAATGTTCGT TGTTAATGTT
BrontosaurCAAAACCCAT CATCAAAACC
B.subtilisGGCAGCCAAT CACGGCAGCC

TACCGCCGAT GCTTACCGC
CGTTGTCGTT ACTCGTTGT
AATTGTTAAT GTTAATTGT
CGTTGTTAAT GTTCGTTGT
CATCATCAAA ACCCATCAT
AATCACGGCA GCCAATCAC
```

Note that in these sequences we have a blank every ten sites to make them easier to read: any such blanks are allowed. The blank line which separates the two groups of lines (the ones containing sites 1-20 and ones containing sites 21-39) may or may not be present, but if it is, it should be a line of zero length and not contain any extra blank characters (this is because of a limitation of the current versions of the programs). It is important that the number of sites in each group be the same for all species (i.e., it will not be possible to run the programs successfully if the first species line contains 20 bases, but the first line for the second species contains 21 bases).

Alternatively, an option can be selected to take the data in "sequential" format, with all of the data for the first species, then all of the characters for the next species, and so on. This is also the way that the discrete characters programs and the gene frequencies and quantitative characters programs want to read the data. They do not allow the "interleaved" format.

In the sequential format, the character data can run on to a new line at any time (except in a species name or in the case of continuous character and distance matrix programs where you cannot go to a new line in the middle of a real number). Thus it is legal to have:

```
Archaeopt 001100
1101
```

or even:

```
Archaeopt
0011001101
```

though note that the FULL ten characters of the species name MUST then be present: in the above case there must be a blank after the "t". In all cases it is possible to put internal blanks between any of the character values, so that

```
Archaeopt 0011001101 0111011100
```

is allowed.

If you make an error in the input file, the programs will often detect that they have been fed an illegal character or illegal numerical value and issue an error message such as "BAD CHARACTER STATE:", often printing out the bad value, and sometimes the number of the species and character in which it occurred. The program will then stop shortly after. One of the things which can lead to a bad value is the omission of something earlier in the file, or the insertion of something superfluous, which cause the reading of the file to get out of synchronization. The program then starts reading things it didn't expect, and concludes that they are in error. So if you see this error message, you may also want to look for the earlier problem that may have led to this.

The other major variation on the input data format is the options information. Many options are selected using the menu, but a few are selected by including extra information in the input file. Some options are described below.

The Options Menu

The menu is straightforward. It typically looks like this (this one is for DNAPARS):

```
DNA parsimony algorithm, version 3.5c

Setting for this run:
  U                 Search for best tree?  Yes
  J   Randomize input order of sequences?  No. Use input order
  O                        Outgroup root?  No, use as outgroup species  1
  T              Use Threshold parsimony?  No, use ordinary parsimony
  M           Analyze multiple data sets?  No
  I          Input sequences interleaved?  Yes
  0   Terminal type (IBM PC, VT52, ANSI)?  ANSI
  1    Print out the data at start of run  No
  2  Print indications of progress of run  Yes
  3                        Print out tree  Yes
  4          Print out steps in each site  No
  5  Print sequences at all nodes of tree  No
  6       Write out trees onto tree file?  Yes

Are these settings correct? (type Y or the letter for one to change)
```

If you want to accept the default settings (they are shown in the above case) you can simply type "Y" followed by a carriage-return (Enter) character. If you want to change any of the options, you should type the letter shown to the left of its entry in the menu. For example, to set a threshold type "T". Lower-case letters will also work. For many of the options the program will ask for supplementary information, such as the value of the threshold.

Note the "Terminal type" entry, which you will find on all menus. It allows you to specify which type of terminal your screen is. The options are an IBM PC screen, an ANSI standard terminal (such as a DEC VT100), a DEC VT52-compatible terminal, such as a Zenith Z29, or no terminal type. Choosing "0" toggles among these four options in cyclical order, changing each time the "0" option is chosen. If one of them is right for your terminal the screen will be cleared before the menu is displayed. If none works the "none" option should probably be chosen. Keep in mind that VT-52 compatible terminals can freeze up if they receive the screen-clearing commands for the ANSI standard terminal! If this is a problem it may be helpful to recompile the program, setting the constants near its beginning so that the program starts up with the VT52 option set.

The other numbered options control which information the program will display on your screen or on the output files. The option to "Print indications of progress of run" will show information such as the names of the species as they are successively added to the tree, and the progress of global rearrangements. You will usually want to see these as reassurance that the program is running and to help you estimate how long it will take. But if you are running the program "in background" as can be done on multitasking and multiuser systems such as Unix, and do not have the program running in its own window, you may want to turn this option off so that it does not disturb your use of the computer while the program is running.

The Output File
--- ------ ----

Most of the programs write their output onto a file called (usually) "outfile", and a representation of the trees found onto a file called "treefile".

The exact contents of the output file vary from program to program and also depend on which menu options you have selected. For many programs, if you select all possible output information, the output will consist of (1) the name of the program and its version number, (2) the input information printed out, (3) a series of phylogenies, some with associated information indicating how much change there was in each character or on each part of the tree. A typical rooted tree looks like this:

```
                                     +-------------------Gibbon
        +----------------------------2
        !                            !      +------------------Orang
        !                            +------4
        !                                   !  +---------Gorilla
  +-----3                                   +--6
  !     !                                      !    +---------Chimp
  !     !                                      +----5
--1     !                                           +-----Human
  !     !
  !     +-----------------------------------------------Mouse
  !
  +------------------------------------------------Bovine
```

The interpretation of the tree is fairly straightforward: it "grows" from left to right. The numbers at the forks are arbitrary and are used (if present) merely to identify the forks. In some of the programs asterisks ("*") are used instead of numbers. For many of the programs the tree produced is unrooted. It is printed out in nearly the same form, but with a warning message:

```
   remember: this is an unrooted tree!
```

The warning message ("remember: ...") indicates that this is an unrooted tree (mathematicians still call this a tree, though some systematists unfortunately use the term "network". This conflicts with standard mathematical usage, which reserves the name "network" for a completely different kind of graph). The root of this tree could be anywhere, say on the line leading immediately to Mouse. As an exercise, see if you can tell whether the following tree is or is not a different one from the above:

```
             +-----------------------------------------------Mouse
             !
   +---------4                                   +------------------Orang
   !         !                            +------3
   !         !                            !      !       +---------Chimp
---6         +----------------------------1      !  +----2
   !                                      !      +--5    +-----Human
   !                                      !         !
   !                                      !         +---------Gorilla
   !                                      !
   !                                      +-------------------Gibbon
   !
   +-------------------------------------------Bovine

   remember: this is an unrooted tree!
```

(it is NOT different). It is IMPORTANT also to realize that the lengths of the segments of the printed tree may not be significant: some may actually represent branches of zero length, in the sense that there is no evidence that the branches are nonzero in length. Some of the diagrams of trees attempt to print branches approximately proportional to estimated branch lengths, while in others the lengths are purely conventional and are presented just to make the topology visible. You will have to look closely at the documentation that accompanies each program to see what it presents and what is known about the lengths of the branches on the tree. The above tree attempts to represent branch lengths approximately in the diagram. But even in those cases, some of the smaller branches are likely to be artificially lengthened to make the tree topology clearer. Here is what a tree from DNAPARS looks like, when no attempt is made to make the lengths of branches in the diagram proportional to estimated branch lengths:

```
                +--Human
             +--5
          +--4  +--Chimp
          !  !
       +--3  +-----Gorilla
       !  !
    +--2  +--------Orang
    !  !
 +--1  +-----------Gibbon
 !  !
--6  +--------------Mouse
 !
 +-----------------Bovine

 remember: this is an unrooted tree!
```

Some of the parsimony programs in the package can print out a table of the number of steps that different characters (or sites) require on the tree. This table may not be obvious at first. A typical example looks like this:

```
 steps in each site:
        0   1   2   3   4   5   6   7   8   9
     *-----------------------------------------
    0!      2   2   2   2   1   1   2   2   1
   10!  1   2   3   1   1   1   1   1   1   2
   20!  1   2   2   1   2   2   1   1   1   2
   30!  1   2   1   1   1   2   1   3   1   1
   40!  1
```

The numbers across the top and down the side indicate which site is being referred to. Thus site 23 is column "3" of row "20" and has 2 steps in this case.

The Tree File
--- ---- ----

In output from most programs, a representation of the tree is also written into the tree file (usually named "treefile"). The tree is specified by the nested pairs of parentheses, enclosing names and separated by commas. If there are any blanks in the names, these must be replaced by the underscore character "_". Trailing blanks in the name may be omitted. The pattern of the parentheses indicates the pattern of the tree by having each pair of parentheses enclose all the members of a monophyletic group. The tree file for the above tree would have its first line look like this:

```
((Mouse,Bovine),((Orang,(Gorilla,(Chimp,Human))),Gibbon));
```

In the above tree the first fork separates the lineage leading to Mouse and Bovine from the lineage leading to the rest. Within the latter group there is a fork separating Gibbon from the rest, and so on. The entire tree is enclosed in an outermost pair of parentheses. The tree ends with a semicolon. In some programs such as DNAML, FITCH, and CONTML, the tree will be completely unrooted and specified by a bottommost fork with a three-way split, with three "monophyletic" groups separated by two commas:

```
(A,(B,(C,D)),(E,F));
```

The three "monophyletic" groups here are A, (B,C,D), and (E,F). The single three-way split corresponds to one of the interior nodes of the unrooted tree (it can be any interior node). The remaining forks are encountered as you move out from that first node, and each then appears as a two-way split. You should check the documentation files for the particular programs you are using to see in which of these forms you can expect the user tree to be in. Note that many of the programs that estimate an unrooted tree produce trees in the treefile in rooted form! This is done for reasons of arbitrary internal bookkeeping. The placement of the root is arbitrary.

For programs estimating branch lengths, these are given in the trees in the tree file as real numbers following a colon, and placed immediately after the group descended from that branch. Here is a typical tree with branch lengths:

```
((cat:47.14069,(weasel:18.87953,((dog:25.46154,(raccoon:19.19959,
bear:6.80041):0.84600):3.87382,(sea_lion:11.99700,
seal:12.00300):7.52973):2.09461):20.59201):25.0,monkey:75.85931);
```

Note that the tree may continue to a new line at any time except in the middle of a name or the middle of a branch length, although in trees written to the tree file this will only be done after a comma.

These representations of trees are a subset of the standard adopted on June 24, 1986 at the annual meetings of the Society for the Study of Evolution at an meeting (the final session in a local lobster restaurant) of an informal committee consisting of Wayne Maddison (MacClade), David Swofford (PAUP), F. James Rohlf (NTSYS-PC), Chris Meacham (COMPROB and plotting programs), James Archie (character coding program), William H.E. Day, and me. This standard is a generalization of PHYLIP’s format, itself based on a well-known representation of trees in terms of parenthesis patterns which has been around for almost a century. The standard is now employed by most phylogeny computer programs but unfortunately has yet to be decribed in a formal published description.

Chapter 10

Pattern Analysis

What is "random"? Intuitively, our idea of randomness is closely connected with homogeneity. Properties of a random sequence should somehow look the same at different scales. If they don't, we describe the sequence as "patchy". All genomes are complex and patchy. Some examples of DNA sequence heterogeneity are protein-coding regions, introns, CpG islands and dispersed tandem repeats such as the 171 human alpha satellite repeat.

What forces create heterogeneity in DNA sequences? Mutation is often though of as random. However, it is a complex process that does not occur uniformly across a genome. The process of replication, for example, may favor the expansion of repetitive regions by slippage. Transcriptionally active DNA may be subject to different mutational forces than non-transcribed regions. Regulatory elements may have different compositional requirements than coding regions. Natural selection is strong force creating DNA heterogeneity. Protein-coding regions experience complex selection intensities that vary among different codon positions and near splice junctions. Evolutionary history also affects sequence composition. Bacterial genomes are a mosaic of resident and horizontally transferred segments. Regions recently acquired from another organism with different base composition may appear as compositional heterogeneity.

10.1 Base Composition: first order patchiness

The fraction of bases that are G or C in a sequence varies dramatically among organisms. The range is greatest among bacterial taxa, which vary from about 30% to 75% (G+C). Genomes of mitochondria and chloroplasts tend to have higher (A+T) content then their host's nuclear genome as do introns compared to flanking exons. Causes for such variation are largely unknown although (G+C) content influences replication, transcription and translation through effects on secondary structure and the stability of double stranded molecules. Mutation and repair processes also affect DNA composition. It is tempting to speculate that higher (G+C) content is associated with thermostability since GC base pairs increase the melting temperature of DNA. However, there is no correlation between (G+C) and optimum growth temperature among prokaryotic genera (Galtier and Lobry, 1997 J. Mol. Evol. 44: 632).

10.1.1 Genome Patchiness

Differences in nucleotide composition are observed within genomes as well as between genomes. Karlin and Brendel (Science 259: 677-680, 1993) discussed the statistical analysis of DNA patchiness. Base content fluctuates at many different scales. One example is the large (>100 kb) regions in vertebrate genomes called "isochores" (Bernardi, Annu. Rev. Genetics 29: 445-476, 1995). Isochores are correlated with the staining properties of vertebrate chromosomes (Giemsa-positive and -negative bands). They have been revealed by physical analysis of DNA fragments as well as from DNA sequences (Ikemura *et al*., Genomics 8: 207-216, 1990). Genes tend to be concentrated in (G+C)-rich regions, but both coding and non-coding portions are subject to similar influences on composition. DNA sequence analysis of one isochore boundary indicated that it is sharp (Fukagawa *et al*., Genomics 25: 184-191; 1995). The origin of isochores is not clear. Bernardi favors an evolutionary explanation based on composition differences between warm and cold-blooded

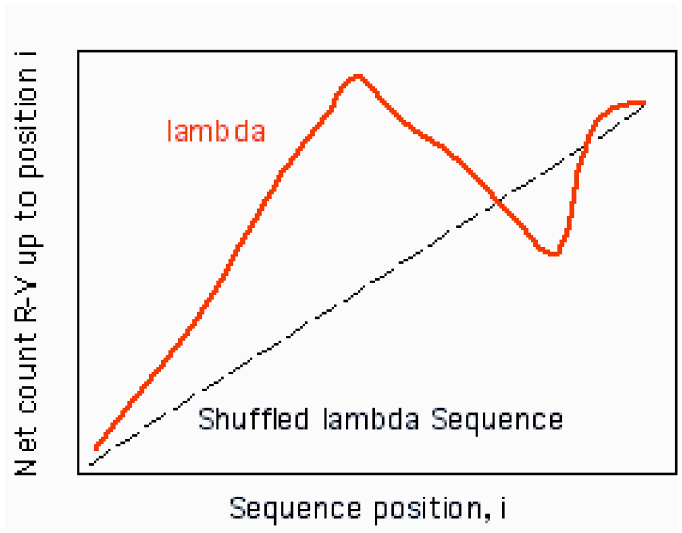


Figure 10.1: Sequence walk plot of the lambda genome after Karlin and Brendel, 1993.

animals. (G+C)-rich isochores are prominent in mammals and birds, although gene clustering and composition patchiness has also been observed in plants. Bernardi suggests that the (G+C)-rich isochores of mammals and birds originated about 200 million years ago from corresponding (G+C)-rich regions in their ancestors.

How can compositional patches be detected and what do they mean? These are questions that are actively pursued but not satisfactorily answered. Sequence walks are a simple method used to detect patchiness (Karlin and Brendel, 1993). As position is increased along a DNA sequence, the value of a variable is incremented +1 or 1 depending on a compositional parameter. Figure 10.1 is a sequence walk plot for the bacteriophage lambda genome where +1 is taken if the position is A or G (R=purine) and -1 if T or C (Y=pyrimidine) as described by Karlin & Brendel (1993). A randomly shuffled lambda sequence shows a steady increase in R-Y, while the actual lambda sequence has a patchy distribution of purines and pyrimidines.

Patchiness can also be visualized using a sliding window approach. Compositional parameters such as the (A+T) fraction are evaluated within a window that slides along the DNA sequence. Figure 10.2 is an (A+T) plot for the *E. coli* K12 genome. No unusual features are revealed in spite of the fact that the K12 chromosome contains several horizontally transferred regions.

10.2 Dinucleotide Composition: second order patchiness

Kornberg and his colleagues in the 1960s developed biochemical techniques for determining the dinucleotide content of DNA (Josse, J, Kaiser, AD and Kornberg, A, J. Biol. Chem. 261: 864-875, 1961). DNA is copied from a template using a 5′p labeled nucleoside triphosphate (pp*pY). The product is then cleaved with an enzyme that leaves a 3′p (Xp*). The radioactivity in Xp* is proportional to the amount of XpY in the DNA. Relative XpY values (nearest neighbor frequencies) are normalized by the amounts of X and Y to give a dinucleotide spectrum. These spectra were found to be characteristic of groups of organisms and were called the "general design" of DNA. They were used in cluster analysis to group organisms according to similarity in dinucleotide composition (Russell, GJ and Subak-Sharpe, JH, Nature 266: 533-536, 1977).

Karlin and his coworkers (Karlin, S, Campbell, AM and Mrázek, J, Annu. Rev. Genet. 32: 185-225, 1998) extended these biochemical methods to the computational analysis of DNA sequences. Following earlier work with "general design", Karlin suggested that dinucleotide frequencies can be used as a "genome signature". The normalized dinucleotide frequencies (called dinucleotide signatures) for a single DNA strand are given by equation 10.1 where f_{XY} is the frequency of XpY in the single strand and f_X is the frequency of X.

$$\rho_{XY} = f_{XY}/f_X f_Y \qquad (10.1)$$

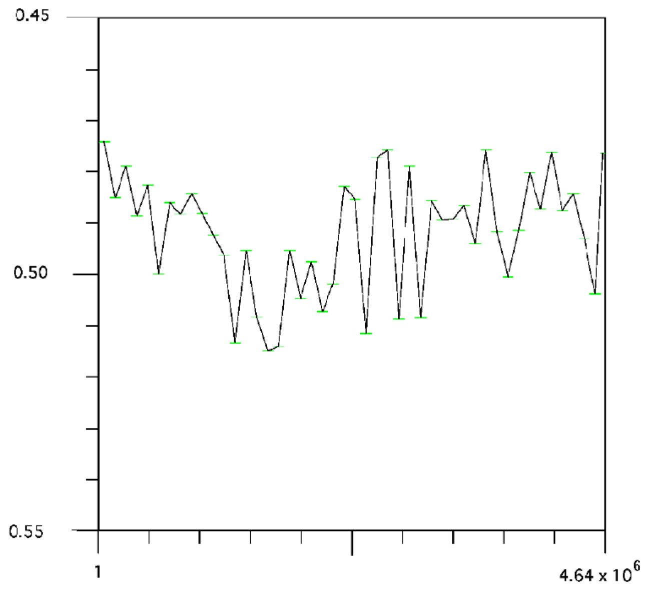


Figure 10.2: (A+T) fraction of the *E. coli* K12 chromosome, window = 100,000 nt

For normalized dinucleotide frequencies in dsDNA, the forward strand is concatenated with its complement in the above calculation. When the dinucleotide signature of XpY is $>$1.0, it is more frequent than expected from the nucleotide composition, while $\rho_{XY} < 1.0$ indicates under-representation. Karlin *et al.* (1998) suggest that $\rho_{XY} < 0.78$ or $\rho_{XY} > 1.23$ in 50 kbp or more of DNA are significant.

Genomic signatures may be useful for determining similarity within broad groups of organisms. They may also be able to detect horizontal transmission of DNA, provided the foreign DNA is from an organism with a different dinucleotide signature. For example, GpC dinucleotides are over-represented in the *E. coli* genome but not in some other bacteria such as *Pseudomonas*. There are many unexplained peculiarities about dinucleotide frequencies. For example, TpA is almost universally under-represented in DNA. Although this was observed in the biochemical studies of the 1960s, it has never been explained. The avoidance of CpG in vertebrate genomes is the one significant signature that has a theoretical basis. Vertebrates, but not invertebrates, methylate CpG (CpG $\rightarrow$ ^{5m}CpG). Deamination of ^{5m}C produces T so that ^{5m}CpG frequently mutates to TpG (mismatch repair is unable to correct TG pairs). Presumably, as CpG methylation evolved, the frequency of CpG dinucleotides decreased through mutation. With an important exception, CpG islands remain where methylation does not occur (Bird, AP, Nature 321: 209-213, 1986, see Figure 10.3). These unmethylated CpG islands are found in the 5$'$ regions of many genes, especially those that are constitutively expressed. Interesting, these CpG islands become hypermethylated in many tumors and gene expression is silenced (Esteller, M, Corn, PG, Baylin, SB and Herman, JG, Cancer Res. 61: 3225-3229, 2001). CpG methylation cannot be the complete story for the wide avoidance of this dinucleotide because CpG is also under-represented in mitochondrial genomes where it is not methylated.

10.3 Strand Asymmetry

10.3.1 Chargaff's Rules

Chargaff's rules express the fact that double stranded DNA obeys Watson-Crick base pairing. The two stands of dsDNA are sometimes labeled "Watson" and "Crick". Chargaff's first rules are A$_c = T_w$, $T_c = A_w$, $C_c = G_w$ and $G_c = C_w$, where the letters represent the molar fraction of a base on one strand. These rules result from formation of Watson- Crick base pairing between strands and are very precisely obeyed by dsDNA molecules.

Less well known are Chargaff's second rules. These apply only approximately and separately to each of the two strands

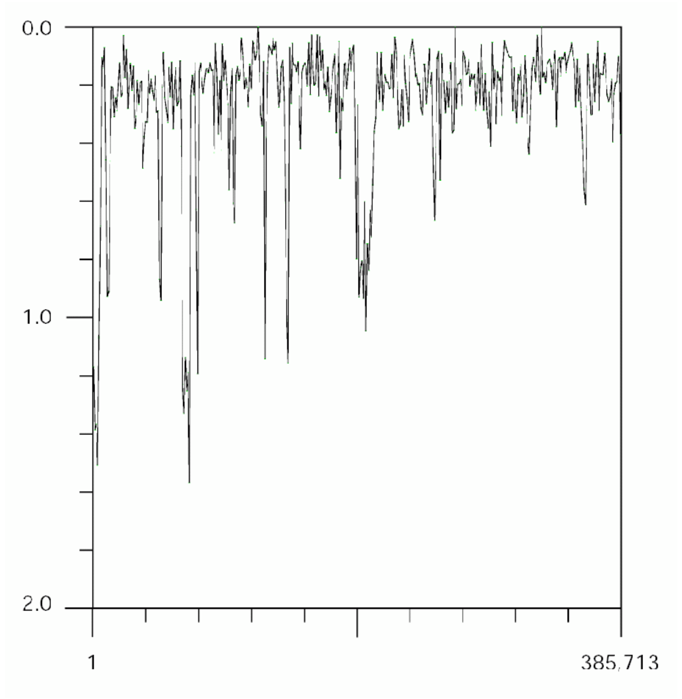


Figure 10.3: CpG islands in a 385 kbp segment of human DNA from chromosome 10 (Accession: AL031601). Dinucleotide signature (ρ_{GC} for CpG), window = 1,000 nt

of dsDNA. They are: $A_c \sim T_c$, $T_w \sim A_w$, $C_c \sim G_c$ and $G_w \sim C_w$. Chargaff's second rules express the fact that complementary strands are approximately symmetric in nucleotide content. If they are true, then $A_c = A_w$, T$_c$ $= T_w$, $C_c = C_w$ and $G_c = G_w$. Departures from strand symmetry (Chargaff asymmetry) are expressed by differences: (A-T)/(A+T) and (G-C)/(G+C) on a single strand.

$$\begin{aligned} \phi_{AT} &= (f_A - f_T)/(f_A + f_T) \\ \phi_{GC} &= (f_G - f_C)/(f_G + f_C) \end{aligned} \tag{10.2}$$

Strand symmetry originates from identical substitution processes affecting each strand, for example, when changing $A_c \rightarrow T_c$ has the same probability as $A_w \rightarrow T_w$. Under these circumstances, the number of AT base pairs will approximately equal the number of TA base pairs (and likewise for GC and CG). However, some mutation processes are known to be strand asymmetric (Francino and Ochman, Trends Genet. 13: 240-245, 1997). Furthermore, nucleotide substitution is subjected to selection that may depend on information contained in only one strand.

10.3.2 Replication Asymmetry

The leading- and lagging-strands are replicated by different mechanisms. The leading-strand is copied by a continuous process, while the lagging strand is synthesized discontinuously using multiple, short RNA primers. Additional enzymes are needed to synthesize primers and then later remove them and fill in gaps. Leading- and lagging-strand replication may involve different polymerases with disparate error rates. As well, the structure of the replication fork exposes the leading- and lagging-strands to different environments. The lagging-strand is more open as a longer, single-stranded structure, which could lead to increased DNA damage and repair.

Mutagenesis experiments in *E. coli* have shown that deletions and replication errors are more frequent on the lagging strand. Differences depend on the agent inducing replication errors. Excess dTTP causes more errors on the lagging strand, while

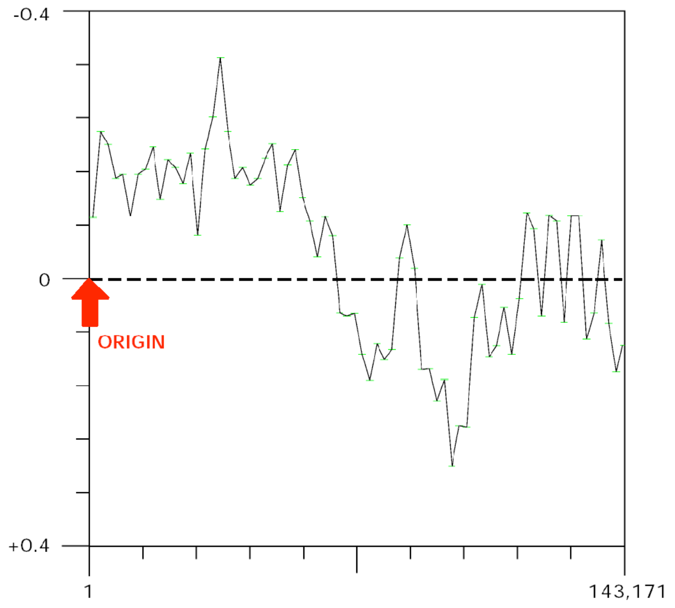


Figure 10.4: Strand asymmetry for the *Euglena gracilis* chloroplast chromosome (Accession: X70810) after Morton (1999). The chromosome is circular and strand asymmetry changes sign quickly at the replication origin and at a point about 180^0 from the origin. There are also peaks associated with the open reading frames and the three rRNA operons. $\phi_{AT} = (f_A - f_T)/(f_A + f_T)$ for window = 1,000 nt.

excess dCTP makes little difference. In general, it seems that $Y \geq R$ (pyrimidine $\geq$ purine) changes are more frequent on the lagging-strand, causing an accumulation of purines.

Replication bias may cause a switch in Chargaff asymmetry across a replication origin because at this point the leading- and lagging-strands change identity. An example is the *Euglena gracilis* chloroplast genome as reported by Brian Morton (Proc. Natl. Acad. Sci. USA. 96: 5123-5128, 1999), see Figure 10.4.

Loby (Mol. Biol. Evol. 13: 660-665, 1996) analyzed the chromosomes of several bacteria for replication bias. The expected switch in strand asymmetry occurred across the replication origins. Changes in (G-C)/(G+C) were much more dramatic than changes in (A-T)/(A+T). The replication effect was partly obscured by protein-coding sequences, which introduce their own bias (see also the *Euglena* chromosome in Figure 10.4). Wherever one strand had a higher density of coding sequences, that strand was found to increase G>C and T>A. Contrary to the expectation from mutagenesis, the lagging-strand accumulated more A and C (instead of A and G).

No evidence has been found for replication bias in eukaryotes. Chargaff asymmetries switch rapidly over short regions of the chromosome although they are generally higher around protein-coding exons. Apparently, the effect of mutational bias and/or codon selection obscures the asymmetry (if any!) caused by a replication origin.

10.3.3 Transcriptional Asymmetry

Transcription can also introduce Chargaff asymmetry since the two strands may be subject to different mutational effects. During transcription, the non-template strand is in an open single-stranded conformation that is more sensitive to certain mutations such as C $\geq$ T (U) deamination. The template strand, on the other hand may be subject to transcription-dependent repair. DNA damage (for example a pyrimidine dimer) can stall the RNA polymerase and promote the action of nucleotide excision repair. This repair may be error-prone, inducing mutations on the template strand. Or unrepaired damage on the non-template strand may lead to substitution.

10.3.4 Codon Selection

Selection for specific amino acids in protein-coding DNA produces strand asymmetry. For example, suppose selection favors glycine in a protein. Thus, GGN codons tend to occur on one strand and complementary NCC nucleotides on the other. The content of G increases relative to C in the sense strand. Thus, protein amino acid composition can impose strand asymmetry. Certain kinds of codon bias in the synonymous position also produce strand asymmetry. One site to find a general description of DNA walks can be found at http://www2.unil.ch/comparativegenometrics/DNA_walk.html.

10.4 Simple Sequence Repeats

Runs of a repeated amino acid are common in the proteins of all organisms. The first triplet repeats, called "Opa", were discovered in the Drosophila Notch gene (Wharton, KA, Yedvobnick, B, Finnerty, VG, Artavanis-Tsakonas, S. Cell 40: 55-62, 1985). These are CAG (or CAA) repeats that code for glutamine (GLN = Q) when translated. Glutamine domains often form protein-protein inter- or intra-molecular contacts. They are an example of the general class of triplet repeats. CAG is the best-known, but CTG, GCC CGG as well as others are also common. Triplet repeats have an been associated with a number of genetic syndromes (Paulson, HL and Fischbeck, KH. Annu. Rev. Neurosci. 19: 79-107, 1996). They are not always found in protein-coding domains, but are also observed in non-coding sequences. They are a subset of minisatellite repeats that have been used for studies of DNA polymorphism, evolution and fingerprinting.

The reiteration of a single amino acid is only one way in which the complexity of protein- coding DNA is reduced. Brian Golding has found that "simple sequence" motifs are a common feature of proteins (Golding, GB. Protein Sci. 8: 1358-1361, 1999). Regions that contain repeated amino acids of varying complexity represent protein sequence simplification. An example is splicing factors that contain repeated "SR" (serine-arginine) domains. These domains are involved in protein-protein contacts that take place during dimerization.

Protein simplification can be detected by using information theory, which will be described in more detail in section 10.5.2. The Shannon-Weaver index is used as a measure of complexity. Figure 10.5 shows how information content reveals regions of low complexity in a yeast nuclear localization protein. The most dramatic is from about 10% to 30% of the protein sequence where reiterated serines frequently occur. Another region of high glycine content occurs at about 90% of the sequence.

10.5 Sequence Complexity

There are a number of ways that complexity in DNA or protein sequences might be represented. The best is based on information theory. Information theory describes the information content of a sequence of symbols. There is little information in repetitive symbols because the number of possible messages that can be made from them is small. On the other hand, sequences that appear random or complex can potentially contain a great deal of information.

10.5.1 Information Theory

Shannon and Weaver developed their theory of information in order to understand the transmission of electronic signals. Gatlin (Information Theory and the Living System. Columbia University Press, New York,1972) describes its extension to Biology. Information theory is an obvious tool to look for pattern and complexity in DNA and protein sequences (Schneider, 1995, Information theory primer. ftp://ftp.ncifcrf.gov/pub/delila/primer.ps). However, results from this area have so far been somewhat disappointing. Shannon and Weaver developed a measure for the information content of messages made from L elements, each element chosen from a set of (S$_i$) symbols with probability of occurrence p_i.

$$H = -L \sum p_i log_2(p_i) \qquad [log_2(p_i) = 1.4427 log_e(p_i)] \tag{10.3}$$

The units of H are called "bits". Since logarithms are additive, L in equation 10.3 can be removed (H/L) to give the average value in bits per nucleotide (or amino acid) site. For a DNA sequence of length L containing four bases, each

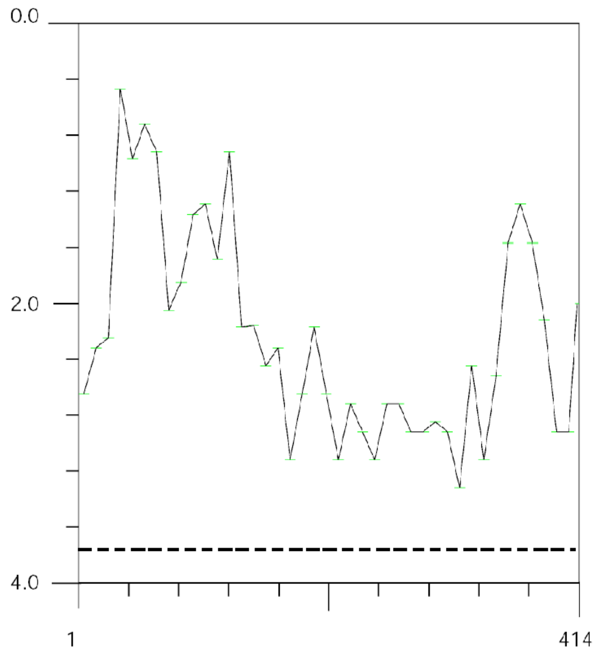


Figure 10.5: Amino acid complexity (Shannon-Weaver information content) in the *Saccharomyces cerevisiae* nuclear localization sequence binding protein; Nsr1p (Accession: NP_011675). The dashed line shows the Shannon-Weaver index for the entire protein sequence, the solid lines connect windows of 10 amino acids.

having $p_i = 0.25$, $H_{max} = Llog_2(4) = 2L$ bits or 2 bits per nucleotide site. Each nucleotide site can be represented by a two bit number (11, 10, 01, 00). This is the maximum information content of a DNA message. Less information is contained in sequences that depart from equal probability. At the other extreme is a sequence comprised of a single base ($p_i = 1$, $H/L = 0$). The Shannon-Weaver index can be regarded as a measure of the complexity of a sequence. $H/L = 0$ represents a sequence of minimum complexity, $H/L = 2$ bits has maximum possible complexity.

One way to think about the Shannon-Weaver index is in terms of uncertainty. Suppose the four bases are equally likely. The uncertainty of a single base is 2 bits before it is read by a functional device (enzyme). After the base is decoded, its uncertainty is zero. The information content of the message is the decrease in uncertainty as a result of decoding.

There is a paradox in the use of the Shannon-Weaver index to express the information content of a message (Hariri et al, J. Theor. Biol. 147: 235-254, 1990). When noise is introduced into a message, its uncertainty increases and the Shannon-Weaver index increases. However, in a real sense the useful information content decreases. Information theory distinguishes between information at the transmitter and at the receiver. The uncertainty in the message may not be zero after reception if noise has been introduced during transmission. However, when decoding DNA sequences, only the message received is available so this distinction cannot be made.

Consider this paradox is from an evolutionary standpoint. Natural selection reduces variability leading to conservation (constraint) of DNA (or protein) sequences. Natural selection reduces the potential information in a DNA message, thereby decreasing its uncertainty. Any constraint in a DNA sequence brings about a difference between potential and stored information. Natural selection is only one way to constrain DNA sequences. They may also be historically constrained by having a recent ancestor or they may be constrained by mutation if generated by a non-random process.

Information theory has been applied to the analysis of DNA and protein sequences in three ways.

1. Analyzing sequence complexity from the Shannon-Weaver indices of smaller DNA fragments (windows) contained in a long sequence as was done in Figure 10.5.

2. Comparing homologous sites in a set of aligned sequences by means of their information content. That is, determin-

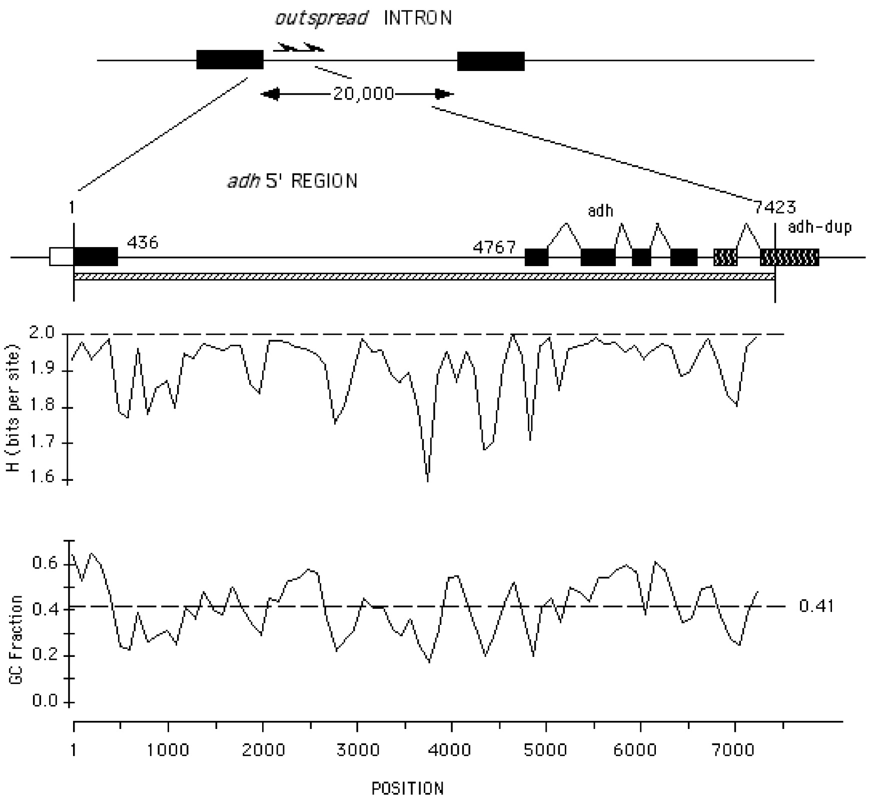


Figure 10.6: Nucleotide complexity (Shannon-Weaver information content H/L) of the *D. melanogaster* ADH gene region (Accession: Z00030), windows of 100 nucleotides overlapped by 50 nucleotides.

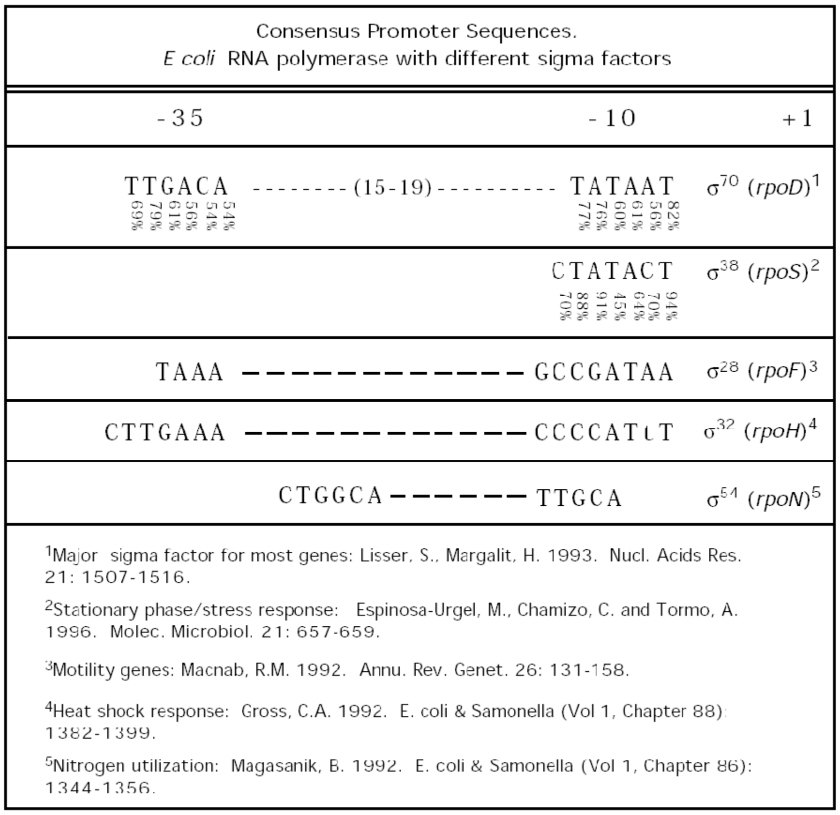
ing the complexity of homologous sites.

3. Examining the pattern of information content of a sequence divided into successively longer words (symbols) consisting of a single base pairs, triplets and so forth. This is a method to look at clustering of nucleotides and will not be considered.

10.5.2 Sequence Window Complexity

An analysis of the *D. melanogaster* alcohol dehydrogenase (ADH) gene illustrates the application of information theory to DNA sequence data (Figure 10.6).

The ADH gene lies within a 20 kb intron of a larger gene, outspread. Generally, maximum complexity is found in exons of either ADH or outspread. In fact, the existence of the left-most exon in outspread was first deduced from an open reading frame 5′ to the ADH gene before the outspread gene had been mapped. Figure 10.6 also shows a correlation between complexity and base composition. In principle, increasing the relative frequency of any of the nucleotides should have the same effect, to decrease complexity. However, in this region of the *Drosophila* genome, only increased (A+T) decreases complexity, while increased (G+C) has the opposite effect. High GC is associated with protein-coding exons while high AT is associated with non-coding DNA such as introns. Although natural selection produces more constrained messages, proteins do not usually use highly patterned or repetitive codon choices except where simple amino acid repeats are found (see Figure 10.5). The Shannon-Weaver index reaches nearly the maximum value of 2 bits per site for the protein-coding exons of these two *Drosophila* genes. Regions of repetitive DNA, on the other hand, have low complexity. In the ADH region of the *Drosophila* genome, these are associated with AT-rich sequences. It is also interesting that intron DNA between ADH and outspread exons appears to be organized into sub-regions with different complexities. It remains to be seen if intronic regions of high complexity and GC content are functional and constrained by natural selection, as are protein-coding exons, or simply a different kind of neutral DNA.



[1]Major sigma factor for most genes: Lisser, S., Margalit, H. 1993. Nucl. Acids Res. 21: 1507-1516.

[2]Stationary phase/stress response: Espinosa-Urgel, M., Chamizo, C. and Tormo, A. 1996. Molec. Microbiol. 21: 657-659.

[3]Motility genes: Macnab, R.M. 1992. Annu. Rev. Genet. 26: 131-158.

[4]Heat shock response: Gross, C.A. 1992. E. coli & Samonella (Vol 1, Chapter 88): 1382-1399.

[5]Nitrogen utilization: Magasanik, B. 1992. E. coli & Samonella (Vol 1, Chapter 86): 1344-1356.

Figure 10.7: Consensus sequence analyses of *E. coli* promoters. +1 is the transcriptional start position.

10.6 Finding Pattern in DNA Sequences

A different and perhaps more important problem than compositional heterogeneity is the location of regulatory elements. Functionally important sequences are conserved across homologous DNA segments from different species. Conservation of DNA information is not restricted to evolution by descent. Convergence may produce similar patterns within a single genome. For example, DNA sequences recognized by the same or a similar DNA-binding protein will be conserved in order that the protein functions properly. I will describe a conserved sequence motif, the *E. coli* sigma70 promoter as an example of finding a conserved pattern within a genome.

10.6.1 Consensus Sequences

Promoter sequences, in conjunction with other DNA elements and proteins, activate RNA polymerase binding and transcription. *E. coli* promoter elements are recognized by an RNA polymerase holoenzyme which contains a bound sigma factor (core enzyme plus sigma factor = holoenzyme). The sigma factor is though to provide most of the sequence recognition capability of the holoenzyme. *E. coli* has a number of different sigma factors, each associated with a specific promoter consensus sequence (Figure 10.7).

The consensus sequnce is defined by majority rule. Analysis of the sigma-70 promoter by Lisser and Margalit (Nucleic Acids Res. 21: 1507-1516, 1993) revealed the consensus sequence shown in Figure 10.7. A pattern search for the sigma-70 promoter based on the consensus sequence would look for TTGACA $(N)_{n=15-19}$ TATAAT.

A major drawback to using the consensus sequence in pattern matching is that rarely will an actual promoter perfectly match the consensus sequence. No known sigma-70 promoter matches the consensus sequence at all 12 nucleotides (although this pattern does occur in the *E. coli* genome). Thus, searching for the consensus sigma-70 promoter sequence in front of genes is an exercise in futility. The search must be for "something like" the consensus sequence. But how alike?

Variation found among the promoters of individual *E. coli* genes is indicated under the majority base in Figure 10.7. Some sites in the promoter sequence are more conserved than others. The cause of variation, however, is unknown. It could be due to mutational drift under the influence of selection. There may also be gene-specific effects. For example, genes requiring lower expression may use "weaker" promoters.

It is possible to take variation into account in a pattern search by defining alternative nucleotides. For example, if the most frequent alternative to the first T in TTGACA is A, the pattern search could be for (T/A)TGACA. Problems with simple pattern searches are obvious. The number of possible patterns grows exponentially with alternatives, but all of them are not equally useful as matches. A pattern with 10 mismatches from the consensus is probably not a promoter, but one with two mismatches might be. To account for this, pattern-matching programs will allow up to a specified number of mismatches. Another problem is that there may be no clear alternatives to the consensus nucleotide. This is the case with the *E. coli* sigma-70 promoter where minority nucleotides are more-or-less evenly distributed (Table 10.1).

Base	T	T	G	A	C	A	T	A	T	A	A	T
A	0.10	0.06	0.09	**0.56**	0.21	**0.54**	0.05	**0.76**	0.15	**0.61**	**0.56**	0.06
C	0.10	0.07	0.12	0.17	**0.54**	0.13	0.10	0.06	0.11	0.13	0.20	0.07
G	0.10	0.08	**0.61**	0.11	0.09	0.16	0.08	0.06	0.14	0.14	0.08	0.05
T	**0.69**	**0.79**	0.18	0.16	0.16	0.17	**0.77**	0.12	**0.60**	0.12	0.15	**0.82**

Table 10.1: Fractional occurrence of nucleotides at each position for 298 *E. coli* sigma-70 promoters (Lisser and Margalit, 1993)

10.6.2 Matrix Analysis of Sequence Motifs

Hertz and Stormo discuss the analysis and prediction of *E. coli* promoters (Methods in Enzymol. 273: 30-42, 1996). The basic method of analyzing sequence motifs and their conservation is to compute a score using a scoring matrix. The simplest scoring matrix assigns a score of one for each match with the consensus sequence (Figure 10.8). A perfect match to the consensus nucleotide produces the maximum score. Partial matches produce intermediate scores.

An assumption of using scoring matrices to evaluate potential sequence patterns is that each site must act independently. No covariance is allowed between nucleotide changes at one position with those at another position.

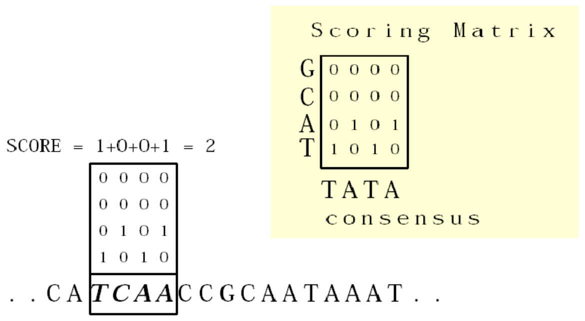


Figure 10.8: Matrix analysis of a sequence motif using a scoring matrix based on the consensus sequence (TATA). The score of 2 for TCAA indicates it matches the consensus sequence at two sites.

Scoring matrices can be developed that use more information about the pattern than contained in the consensus sequence. One approach is to find the matrix that gives the best correlation between the scores it produces and the measured activities of actual promoter sequences. If the activities of an example group of promoter sequences are equal, the maximum likelihood matrix elements will be the logarithms of observed frequencies for each nucleotide at a position divided by the probability that the nucleotide occurs by chance (equation 10.4). The latter can be estimated from the genome nucleotide frequency (e.g., p_i 0.25 for each base in *E. coli*).

$$W_{in} = \log_{10}(F_{in}/p_n) \tag{10.4}$$

W_{in} is the scoring matrix element at the ith position in the pattern for the nth type of nucleotide (G, C, A, or T); F_{in} is the frequency of the nth nucleotide at the ith position among the group of patterns used to derive the consensus sequence;

p_n is the probability that the nth nucleotide occurs by chance. For example, among the group of promoters used to derive the sigma-70 consensus sequence in Figure 10.7, the T at -10 (TATAAT) occurs 82% of the time (Table 10.1). The scoring element for a T at this position is

$$W_{TT} = \log_{10}(0.82/0.25) = 0.516 \tag{10.5}$$

(assuming that T occurs with a frequency of 1/4 in the *E. coli* genome).

Scores for DNA patterns can also be obtained using neural network methods. Examples of such techniques are discussed in Hénaut and Danchin (Analysis and predictions from Escherichia coli sequences, or *E. coli* in silico In: *E. coli* and *Salmonella* Vol. II, Chapter 114: 2047-2066, 1992). A computer program is "trained" on examples of good and bad promoters. Matrix elements are flexible and optimized to discriminate between the training set. Such methods do not usually give appreciably better results than the maximum likelihood approach. However, they can be more easily adapted to include additional information about what makes a good promoter. Many promoters require several proteins to initiate transcription. These recognize other DNA sequence motifs, usually located near the sigma factor binding site. DNA curvature is often a factor. Upstream sequences that bend DNA increase the activity of some promoters (Travers, Cell 60: 177-180, 1990). DNA bending depends mainly on runs of A or T since the dinucleotide AA/TT has the largest tilt angle (Trifonov, CRC Revs. Biochem. 19: 89-106, 1985). DNA curvature can be calculated by accumulating AA and TT pairs. DNA curvature is more easily incorporated into the analysis of promoter scores by using training methods.

10.6.3 Sequence Conservation and Sequence Logos

DNA regulatory elements such as promoter sequences are examples of a constraint placed on the evolution of DNA sequences by natural selection. Variability across genomes or among genes is reduced because a conserved protein molecule must recognize different forms of the element. Variability results from a balance between mutation and selection.

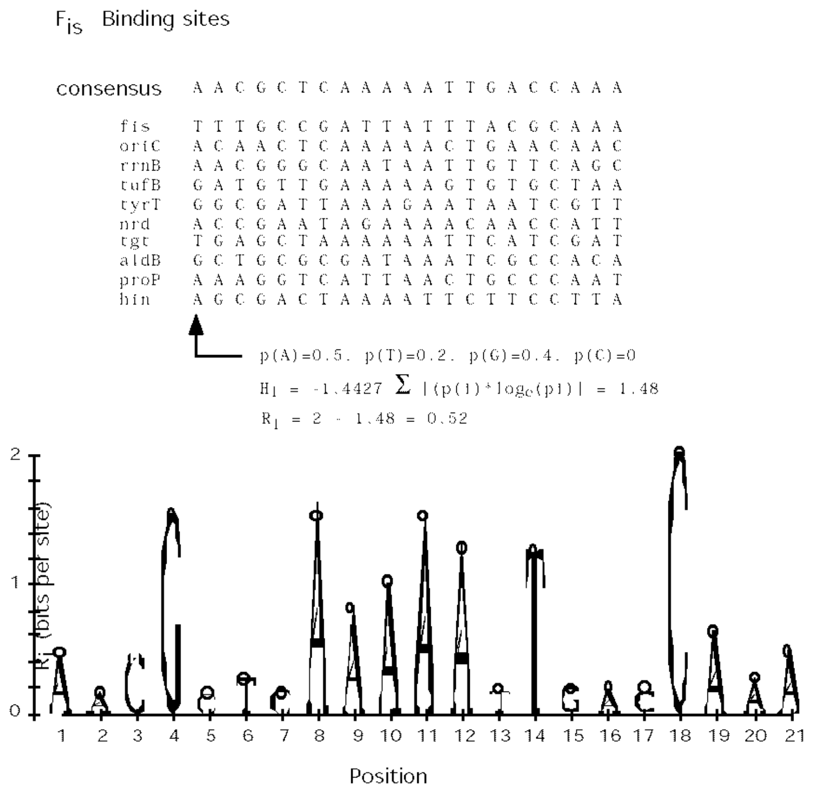
Information theory can be used to analyze the effectiveness of selection. Although the approach can be applied to any conserved DNA or protein sequence, its theoretical basis is clearest for DNA binding sites that are recognized by a protein molecule. In this case, the protein can be thought of as decoding information contained in its binding site. This information can be evaluated by comparing variation among different binding sites. Unlike the consensus sequence in which every position in the binding site is equivalent, information analysis evaluates the relative importance of individual sites. A good description of this method is the paper by Shaner, Blair and Schneider (1994, Sequence logos: a powerful, yet simple, tool. http://www-lecb.ncifcrf.gov/ toms/paper/hawaii/). A "sequence logo", is obtained from a set of aligned DNA (or protein) examples. The information content (R_i) of each site (i) is calculated from equation 10.6.

$$R_i = H_{max} - H_i - e(N) \tag{10.6}$$

R_i is the information content of the site. H_{max} is the maximum uncertainty, 2 bits if the four bases are equally probable before the site is decoded (see section 10.5.1). After decoding (e.g., by the RNA polymerase for promoter sequences), the uncertainty (H_i) is given by equation 10.3 with p_i are nucleotide frequencies calculated from each position of the aligned, example sequences. e(N) is a correction factor to account for the fact that only a finite number of example sequences (N) are used to estimate the information content of the binding site (see Schneider et al, 1986). Figure 10.9 illustrates the method by analysis of the *E. coli* FIS binding site using data from Hengen et al (Nucleic Acids Res. 25: 4994- 5002, 1997).

F_{is} binds to and bends DNA at specific sites. It regulates the transcription of a subset of genes in conjunction with RNA polymerase, and is also involved in the process of recombination. F_{is} sites have been identified in a number of genes as well as the f_{is} gene itself. Some genes (e.g., f_{is}) have a cluster of sites in their promoter region. Figure 10.9 displays the information content of the F_{IS} binding site as a "sequence logo", where each consensus nucleotide is given a size proportional to its information. Hengen *et al.* analyze 60 example sequences (30 sites in both directions since the F_{is} site is known to be symmetrical) from which I selected 10 for illustration in Figure 10.9. Sequence logos can be constructed at the internet site: http://www.bio.cam.ac.uk/cgi-bin/seqlogo/logo.cgi.

An advantage of sequence logos is that sequence conservation can be quantitatively interpreted as the information that the decoder (e.g., F_{is} protein) obtains from potential sites in order to recognize a valid site. For example, the information content of the two GC base pairs in the F_{IS} binding site is approximately 2 bits, close to the maximum information available. The F_{IS} protein contacts the major groove of dsDNA at these positions and can obtain information about base pair identity (e.g., CG vs GC). On the other hand, in the central region of the F_{IS} binding site, the protein contacts the



F_{is} Binding sites

```
consensus   A A C G C T C A A A A A T T G A C C A A A

     fis    T T T G C C G A T T A T T T A C G C A A A
     oriC   A C A A C T C A A A A A C T G A A C A A C
     rrnB   A A C G G G C A A T A A T T G T T C A G C
     tufB   G A T G T T G A A A A A G T G T G C T A A
     tyrT   G G C G A T T A A A G A A T A A T C G T T
     nrd    A C C G A A T A G A A A A C A A C C A T T
     tgt    T G A G C T A A A A A A T T C A T C G A T
     aldB   G C T G C G C G A T A A A T C G C C A C A
     proP   A A A G G T C A T T A A C T G C C C A A T
     hin    A G C G A C T A A A A T T C T T C C T T A
```

$p(A)=0.5,\ p(T)=0.2,\ p(G)=0.4,\ p(C)=0$

$H_1 = -1.4427 \sum |(p(i)*\log_e(pi))| = 1.48$

$R_1 = 2 - 1.48 = 0.52$

Figure 10.9: Information theory analysis of F_{IS} binding sites according to equation 10.6 with $e(N)$ taken as zero for simplicity.

minor groove and can only distinguish GC from AT pairs, but not their orientation. The information available in this region is approximately 1 bit.

The information content of a binding site can be calculated by summing the information at each position. It is approximately 9 bits for the F_{is} consensus sequence (Hengen *et al.*, 1997). This contrasts with a maximum of $21 \times 2 = 42$ bits of information available in a 21 bp binding site. The F_{IS} protein uses only a faction of the this information in order to recognize a site. Nine bits of information is sufficient to allow approximately 16,000 sites to be distinguished in the *E. coli* genome [$9 = -\log_2(x/G)$, where G is the genome nucleotide content, about 8×10^6 nucleotides because each nucleotide begins a potential site (see Schneider *et al.*, 1986). The number of nucleotides in the *E. coli* genome was doubled because the F_{is} site is symmetric. Solution gives $x = 16,000$]. More stringent binding site recognition requires that more information to be used by the protein.

The total information of a potential binding site can be calculated using a scoring matrix derived from equation 10.7.

$$W_{bj} = H_{max} - \log_2(F_{bj}) - e(N) \tag{10.7}$$

W_{bj} is the matrix element for nucleotide of type b at position j in the pattern. F_{bj} is the frequency of this nucleotide in the example set at the same position, and e(N) is a correction for the finite size (N) of the example set. The information content of a test pattern is obtained by using its sequence in equation 10.7. Hengen *et al.* (1997) used this approach to scan the *E. coli* genome for F_{IS} binding sites. A sliding window of 21 nucleotides was moved along the genome sequence and the information content of potential sites evaluated. Segments with information above 2 bits were considered potential F_{IS} sites.

Chapter 11

Exon Analysis

Locating protein-coding genes is an important goal of genomics. This, together with locating RNA genes and regulatory elements is the process of annotation. Annotating DNA is based on three tools; 1) aligning cDNA with genomic DNA, 2) similarity to previously identified genes and 3) theoretical prediction. Annotating the human genome is an ongoing process. The 3×10^9 bp of DNA is estimated to contain the order of 5×10^4 genes. Approximately 1×10^4 complete cDNA sequences have so far been identified. It is likely that complete cDNA sequences will never be obtained for all genes so that computational techniques will be necessary to obtain a complete understanding of its coding potential.

11.1 Open Reading Frames

Prediction of protein-coding genes is primarily based on identifying open reading frames (ORF). Many programs determine open reading frames, among them the GDE software on Life and "Translate" on the ExPasy Molecular Biology Server (http://ca.expasy.org). This however, is only the first step. There are a number of problems in determining if an ORF is actually used to code for protein.

1. Sequencing errors, internal "stop" codons that are removed by editing, and codons for selenomethionine.
2. Spurious ORFs that are not part of any protein-coding gene. The non-coding strand of exons often contains ORFs. That is, the reverse complements of stop codons (TTA [TAA], CTA [TAG], TCA [TGA]) are often statistically avoided, creating ORFs on the complementary strand.
3. Intron exon structure combines several ORFs into a single gene. The splice junction fusion may create the in-phase codon.
4. Splicing creates multiple transcripts and multiple proteins. Certain exons may only be used in a subset of transcripts. The *D. melanogaster Adh* gene, for example has different transcripts during larval and adult phases of growth.

11.2 Gene Recognition

Two general approaches are used to recognize genes within a DNA sequence.

Local alignment methods such as BLAST detect sequence similarity to ESTs or genes already in a database. These are very powerful at finding the approximate location of exons, but do not accurately determine their boundaries. Nor can they combine exons into a gene without additional information.

Global approaches calculate a vector that estimates the protein-coding capacity of a window within the sequence. This vector is simply a one-dimensional array of numbers that incorporate various features that the algorithm uses to determine protein-coding capacity. The measured vector is compared to one obtained from a set of standard genes.

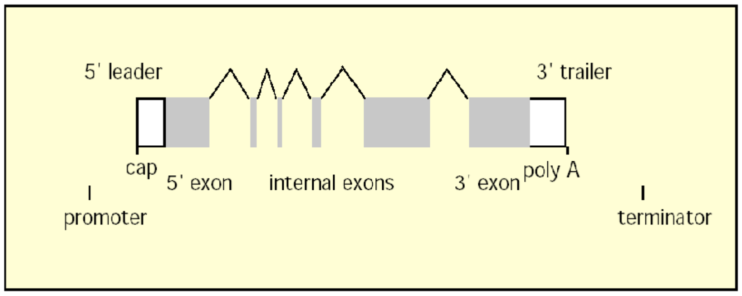


Figure 11.1: Eukaryotic gene with exon - intron structure, protein-coding is gray.

Both of these approaches are combined in annotation software that is used for complete genome annotation. An example is GenomeScan, used extensively to annotate the human genome sequence (Yeh *et al*. 2001. Genome Res. 11: 803-816). Gene prediction combines the location of ORFs with other sequence information to make a model of the entire gene. Data about possible promoters, transcription initiation (cap sites), translation signals (initiation, termination codons), splice signals, and transcription terminators are combined to make an inference that rejects unlikely ORFs and includes likely ORFs in a consistent gene model (Figure 11.1).

Gene prediction algorithms calculate an overall statistic and make a decision as to whether or not to present the model as a potential gene. Neural network methods are often used in which the algorithm is trained on a set of test genes and learns what weights should be assigned to the various measures in order give the best discrimination between valid and invalid test genes.

The ability of various approaches to predict protein-coding genes was assessed by Fickett and Tung (1992. Nucleic Acids Res. 20: 6441-6450). They identified several features that are particularly useful.

1. Codon usage. A codon usage vector (frequencies of the 64 possible codons) for a potential exon is compared to that of a reference sets of genes, preferably from the same or closely related organism. Methods differ in how the reference set is obtained and how the measure of fit is calculated. Reference sets that incorporate information about the amino acid composition of the potential gene are superior to those that do not.

2. In-phase words. A vector similar to the codon vector is calculated for longer words (oligonucleotides of length n). Hexamers have proven useful. These take into account tendencies of codon use to be correlated over short ranges (e.g., a codon ending in G tends not to be followed by one beginning in G).

3. The presence of STOP codons. Most methods only consider ORFs. However, it is possible to incorporate stop codons into a measure of amino acid content.

4. Amino acid content. Measures of protein function, such as vectors of amino acids, dipeptides and hydrophobicity, can be obtained for a potential exon. Like the codon usage vectors, these are compared to a reference set. This, however, may limit identification to particular types of protein-coding genes.

5. Nucleotide periodicity. Nucleotides do not appear at random in coding sequences (nor in non-coding ones). The statistical average codon is RNY, leading to a periodicity of 3 nucleotides. Periodicity vectors are calculated for potential exons (e.g., using Fourier transforms or autocorrelation functions).

11.2.1 Splice Sites

Gene prediction programs must locate splice sites in genes that have exon / intron organization. Information about the splice site is mainly contained within a few nucleotides of the boundary. The dinucleotides ...GT and AG... form the canonical splice sites of most exon-intron junctions (Figure 11.2). The method of sequence logos has been used to represent the contributions of various positions to the information content of human splice sites.
http://www.lecb.ncifcrf.gov/ toms/gallery/SequenceLogoSculpture.gif

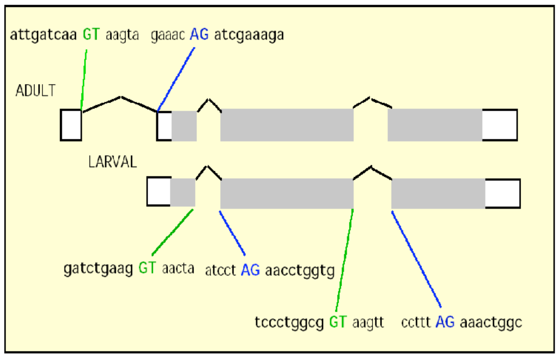

attgatcaa GT aagta gaaac AG atcgaaaga
ADULT
LARVAL
gatctgaag GT aacta atcct AG aacctggtg
tccctggcg GT aagtt ccttt AG aaactggc

Figure 11.2: Exon - intron boundaries of the *D. melanogaster* Adh gene.

11.2.2 Codon Usage

Codon use and nucleotide periodicities are interdependent properties of protein-coding regions that influence exon prediction.

Base Composition. Base composition is a major factor influencing codon usage. Organisms, especially bacteria, have variable GC content. This alters both the types of amino acids and the codons used to code for these amino acids As an example, AAA and AAG both code for lysine. As the genome content of (A+T) increases, proteins tend to use more lysine and more AAA codons (Figure 11.3).

This trend across genomes is repeated within a genome across different genes, although with much more variability. To illustrate, *E. coli* genes that have greater (A+T) content tend to use more AAA codons (Figure 11.4).

Mutational bias is thought have a major effect in determining overall base composition. Other influences, such as selection for compact genome size, have also been suggested.

Mutational bias could reflect replication error, repair efficiency, nucleotide pools or other, unidentified factors. The causes of high, low or intermediate GC content among organisms are not known. Neither are the causes of variation among genes within a genome. Amino acid composition is an obvious possibility, but even with constant composition, GC content can vary because of synonymous codon choice. The problem of GC content and codon choice is a chicken-or-egg situation. They are correlated, but which is driving which and what are the underlying forces?

Codon Position. Codon choice is patterned differently at each of the three codon positions (c1, c2, c3). Figure 11.5 shows nucleotide choices for *E. coli*. The average nucleotide frequencies of all genes are to the right of histograms showing deviations from this average at each position.

In all organisms, G is preferred in the first position. T and, less obviously for *E. coli*, A are avoided. The second position is less consistent, but A is often preferred, especially at moderate or high GC content. The third (synonymous) position shows most clearly the effect of variable GC content. In organisms with high GC content, G and C are preferred in the third position, but are avoided in organisms with high AT content. In *E. coli*, which has an even distribution of nucleotides, G and C are slightly preferred and A slightly avoided.

The choice of codon at the second position is very dependent on the hydrophopicity of the protein because of a pattern in the universal genetic code. T (U in RNA) at c2 is confined to hydrophobic amino acids, while A at c2 is confined to hydrophilic ones.

The effect of this bias in the genetic code is clearly seen in the distribution of nucleotides at c2 in the *E. coli* genome (Figure 11.6). There is a peak of relatively hydrophobic proteins that prefer T (U in RNA) instead of A.

The patterns of codon use described above are complex, but they are not taken individually into account by gene prediction programs. Rather they create trends in protein-coding regions that are utilized by algorithms as frequency distributions of "words" (for example, hexamer frequencies).

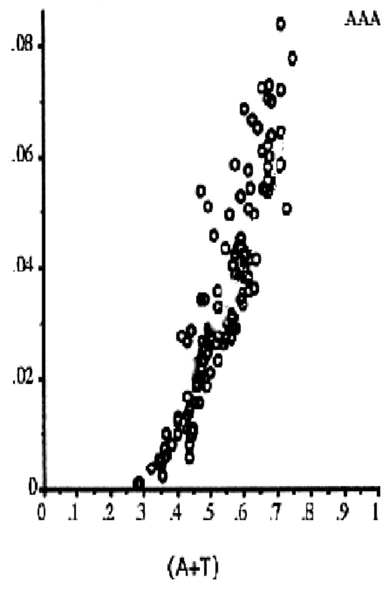


Figure 11.3: The fraction of all codons that are AAA across genomes with different AT contents.

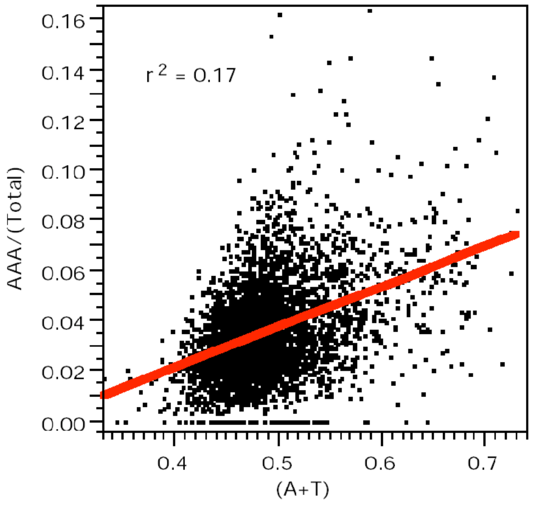


Figure 11.4: The fraction of codons that are AAA for genes of the *E. coli* genome as a function of the gene's (A+T) fraction.

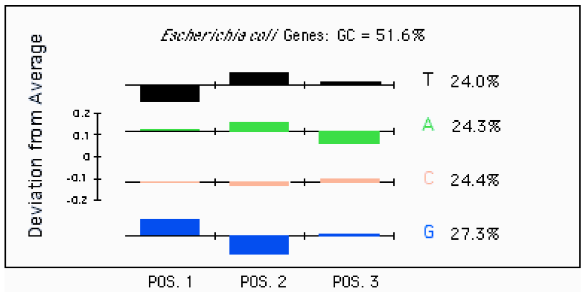


Figure 11.5: Nucleotide composition by codon position for *E. coli* genes.

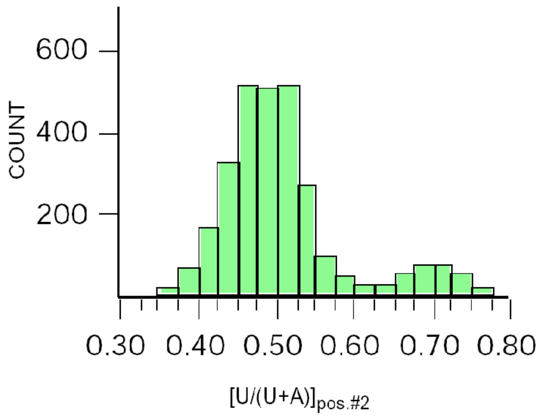


Figure 11.6: The relative content of T to (T+A) at the second position of 3180 *E. coli* genes.

11.2.3 Gene Prediction Software

Many programs are available to build gene models, such as FGENEH, GENMARK, GRAIL, GeneParser. Burset and Guigo (1996. Genomics 34: 353-367) and Guigó *et al.* (2000. Genome Res. 10: 1631-1642) compared many of them and found that their accuracy is often overrated because they have been evaluated on genes similar to the test set used to build the discrimination functions. Three of the most commonly used programs are summarized.

GeneFinder is a group of programs for gene identification written by Victor Solovyev's group of the Computational Genomics Group at the Sanger Centre (Solovyev V and Salamov A. 1997. Proc Int Conf Intell Syst Mol Biol 5:294-302). They were used to predict genes in the *Drosophila* genome (Solovyev V and Salamov A. 2000. Genome Res. 10: 516-22). The software can be accessed for testing at the commercial site http://www.softberry.com/berry.phtml. FGENES is designed to identify and piece exons together to predict multiple genes on both strands. There is a version, FGENES-M that predicts multiple models of a single gene, useful if there are alternate splice forms. FGENESH is a variant using a Hidden Markov Model (HMM, section 11.2.4). FGENESH+ is a program that uses a protein sequence similar to the predicted gene product (possibly obtained from BLAST) in conjunction with FGENESH to more accurately predict exon structure.

FGENES relies on identifying exon donor and acceptor splice sites as described by Solovyev *et al.* (1994. Nucleic Acids Res. 22: 5156-5163). Flanking ($5'$ and $3'$) and internal exons are treated with separate algorithms. The program examines each ORF that terminates in a GT or begins with AG and calculates a linear discriminant function, $z = \sum \alpha_i x_i$, where x_i are measures of a splice site and α_i are weights. The discriminant function is used to classify an exon as valid if z is above a critical value determined from the analysis of test (learning) data. The measures in the discriminate function are triplet nucleotides frequencies at the exon-intron boundaries. Because these are organism dependent, discriminant function weights must be obtained for each species or from a closely-related relative.

GENIE is a program written by the Computational Biology Group at the University of California, Santa Cruz and the Genomic Informatics Group at LBNL (Kulp D, Haussler D, Reese MG, Eeckman FH. 1996. Proc. Int. Conf. Intell. Syst. Mol Biol. 4:134-42). It uses a Generalized Hidden Markov Model (HMM, section 11.2.4) to develop gene models. It has been extensively used to predict genes in the human and fruitfly genomes (Reese MG, Kulp D, Tammana H, Haussler D. 2000. Genome Res. 10:529-38). The web version of Genie is available through the Berkeley *Drosophila* Genome Project (http://www.fruitfly.org/seq_tools/genie.html).

GENSCAN is a program developed by Burge and Karlin (1997. J. Mol. Biol. 268: 78-94). Although designed for human genes, it has been tested successfully on other vertebrate sequences and plants. It also works for *Drosophila*. A large, non-redundant set of human genes (2.58×10^6 nucleotides containing 1492 exons and 1254 introns) was used to develop GENSCAN. GENSCAN is generally regarded as one of the best gene prediction programs and has been extensively used in the human genome project. It incorporates a number of features to build a model.

1. Transcriptional and translational signals are evaluated by weight matrices. Potential signals are: polyadenylation, cap site, promoter (both TATA and TATA-less promoters are allowed with variable distance to the cap site), translational start sites (6 nt prior to start codon) and stop sites (3 nt following stop codon).

2. Splice signals. A modified weight matrix method is used to examine potential splice sites (3 nt in exon, 6 nt in intron). The modified method takes into account correlations between positions.

3. Exon models. Potential coding portions of exons are evaluated using a Markov model. This computes transition probability matrices for hexamers ending at each codon position. Scores are dependent on similarity between the GC-content of the training sequences and the sequence to be evaluated. GENSCAN uses one of two sets of expected transition probabilities that were generated from training sets having either $GC < 43\%$ or $GC > 43\%$.

The internet site for GENSCAN is (http://genes.mit.edu/GENSCAN.html).

Each of the programs described above uses general features of genes (Fig. 11.1) to develop its model. They derive their parameters from analyzing a group of example genes and will perform best if the target gene is similar. Another, potentially more powerful, approach is based on homology to closely related genes. In fact, it is even better to combine this with gene prediction methods. GENOMESCAN is an outgrowth of GENSCAN that evaluates a gene model by making it's probability conditional on similarity results from a BLASTX search of a protein database. In this respect it is similar to FGENESH+,

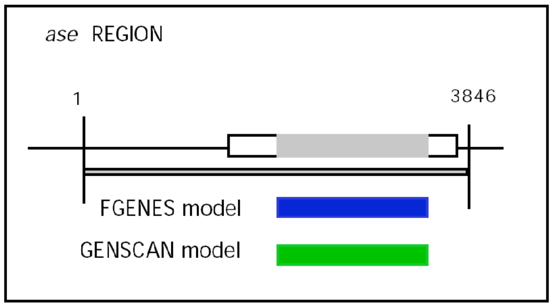


Figure 11.7: Gene models of the *D. melanogaster* ase region.

but more comprehensive and designed for genome annotation. It has been used in annotating the human genome project. It may be accessed at http://genes.mit.edu/genomescan.html. In the web version, you are required to input a similar protein sequence (rather than having the program obtain sequences from BLASTX).

11.2.4 Hidden Markov Models (HMM)

Hidden Markov Models are statistical methods for evaluating sequences labeled with biologically relevant information. Data may be promoter sites, exon positions and termination signals for a gene model. The gene model can be thought of as represented by an array of symbols (called a parse), To obtain the probability distribution for possible arrays of symbols, a "hidden" set of transition states and transition probabilities between these hidden states is assumed. This allows the parse of maximum likelihood to be obtained. HHM must be trained on a set of learning sequences in order to obtain the hidden transition probabilities.

11.2.5 Comparison of Programs

Figure 11.7 shows how FGENES and GENSCAN performed on the *D. melanogaster* asense gene (*ase*, accession: X52892), which does not have introns. The protein-coding portion of the exon was correctly defined and the 486 amino acid gene product returned. This is all that can be expected since none of the gene prediction programs can find non- translated regions of transcribed RNA.

The *Drosophila* Adh region has been extensively examined by genetically (Ashburner M. 1999. Genetics 153:179-219) and results compared with gene prediction programs. Figure 11.8 shows that both FGENES and GENSCAN precisely defined the three protein-coding Adh exons and combined them to give the Adh gene. The correct amino acid sequence of ADH was deduced. The adult promoter was not located, perhaps because it is too far from the first protein-coding exon, but the larval promoter was found. Neither the portion of the outspread exon at the beginning of the sequence nor the adh-dup exons at its end were located by FGENES. The polyA site was incorrectly located in the $3'$ mRNA trailer, however, it is possible that other, shorter transcripts exist that use this site. GENSCAN was unable to locate the outspread exon. Like FGENES, it performs poorly at the boundaries of sequences. It did, however, make a good attempt at the adh-dup exons, locating the beginning of the second exon correctly, but not the first. Interestingly, GENSCAN identified a potential exon at nucleotide position 1388-1566. This region of the DNA sequence has high complexity and GC content.

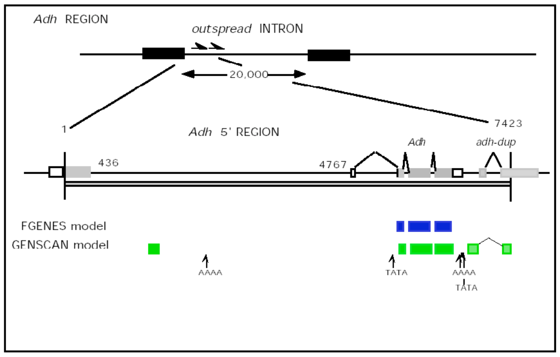


Figure 11.8: Gene models of the *D. melanogaster* Adh region.