BioInformatics: A SeqLab Introduction

Bioinformatics is tough — use a comprehensive, server-based technology to cope with the data!

July 24, 2007, a GCG¥ Wisconsin Package™ SeqLab® tutorial supplement for the Woods Hole Marine Biological Laboratory's Workshop on Molecular Evolution.

Author and Instructor: Steven M. Thompson

Steve Thompson
BioInfo 4U
2538 Winnwood Circle
Valdosta, GA, USA 31601-7953
stevet@bio.fsu.edu
229-249-9751

BioInformatics: A SeqLab introduction

It's a pretty new field, been around not quite thirty years or so, called various, often misunderstood names, that are largely subsets of one another — computational molecular biology, biocomputing, bioinformatics, sequence analysis, molecular modeling, and most lately genomics and proteomics. But what does it all mean? One way to think about it is the reverse biochemistry analogy — biochemists no longer have to begin a research project by isolating and purifying massive amounts of a protein from its native organism in order to characterize a particular gene product. Rather, now scientists can amplify a section of some genome based on its similarity to other genomes, sequence that piece of DNA, and, using sequence analysis tools, infer all sorts of functional, evolutionary, and, perhaps even, structural insight into a gene within it, and then, most likely, go on to clone that gene, express the gene product, and finally purify the protein. The process has come full circle. The computer has become an important tool to be used at the beginning and throughout a research project in assisting experimental design, not just a number cruncher used at the end of the process. This is only possible because of modern computational speed and power and the tremendous growth of the molecular databases. Biocomputing's explosive growth is reflected in and largely a result of the increase in the level of computational processing power available, along with a concurrent exponential growth of the molecular sequence databases. GenBank doubles in size almost every year! GenBank version 160, June 2007, has 77,248,690,945 bases, from 73,078,143 reported sequences.

First, a prelude — my definitions

Much confusion abounds in the area, even concerning the names of the disciplines themselves. The terms are often bantered about with little regard to what they really mean. Here's my slant on the situation. All are interdisciplinary by nature, combining elements of computer and information science, mathematics and statistics, and chemistry and biology. Each has elements of one another. Biocomputing and computational biology are the most encompassing terms and can be considered synonyms. They both describe using computers and computational techniques to analyze a biological system, whether that is a biomolecular primary sequence or tertiary structure, or a metabolic pathway, or even a complex system such as the interactions of populations within an ecological niche.

Bioinformatics necessarily intersects with this concept in that it describes using computational techniques to access, analyze, and interpret the biological information in databases. However, these databases can be the traditionally considered nucleic and amino acid sequence databases as well as three-dimensional molecular structure databases, but can even include such disparate data collections as medical records or population statistics. Therefore, bioinformatics is a type of biocomputing but also includes topics such as medical informatics that is not usually considered a part of computational biology.

Within bioinformatics the subdiscipline of sequence analysis has a clearly defined scope. It is the study of biological molecular sequence data for the purpose of inferring the function, interactions, evolution, and perhaps structure of biological molecules. Molecular modeling can also be considered a type of

bioinformatics, though it often isn't. It is necessarily a subdiscipline of computational structural biology, but uses the methodology and techniques of that discipline as well sequence analysis' similarity searching and alignment algorithms. That is why it is often referred to as "homology modeling."

Genomics is the subdiscipline of bioinformatics that is concerned not with individual molecular sequences, but rather with sequences on a genomic scale. That is, genomics analyzes the context of genes or complete genomes (the total DNA content of an organism) and transcriptomes (the total mRNA content of an organism) within and across genomes. Proteomics can be considered the subdivision of genomics concerned with analyzing the complete protein complement, i.e. the proteome, of organisms, both within and between different organisms.

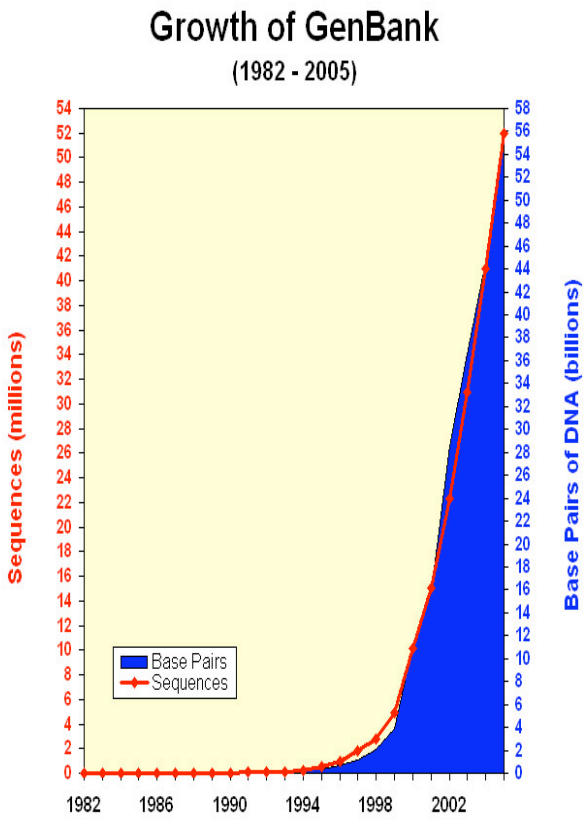
Structural genomics is the acquisition and analysis of the complete set of three-dimensional structure coordinate data for an organism's entire proteome (or a representative set thereof). Through these types of analyses it may eventually be possible to predict a completely unknown protein's structure and function just based on its deduced molecular sequence. Obviously this could be an incredible boost to the drug-design process and could go a long way toward curing many disease processes. We have come a long way in structural prediction, but are still a long way from this goal. The comparative method is crucial to all these methods but, perhaps most obvious and key to genomics and proteomics.

I. Databases: Content and Organization

The first genome sequenced was *Haemophilus influenzae*, at the Johns Hopkins University School of Medicine (Fleischmann, et al, 1995). The International Human Genome Sequencing Consortium announced the completion of a "Working Draft" of the human genome in June 2000 (Lander, et al., 2001); independently that same month, the private company Celera Genomics announced that it had completed the first assembly of the human genome (Venter, et al., 2001). As of July 2007, 44 Archaea, 496 Bacteria, and 27 Eukaryote complete genomes, and 140 Eukaryote assemblies are represented, depending on your definition of complete (not even NCBI agrees with itself on this point!), and not counting the almost 2,000 virus and viroid genomes available. Among them are a cryptomonad, *Guillardia theta*, flagellate, *Leishmania major*, apicomplexan, *Plasmodium falciparum* and *yoelli*, red alga, *Cyanidioschyzon merolae*, microsporidium, *Encephalitozoon cuniculi*, baker's yeast, *Saccharomyces cerevisiae*, fission yeast, *Schizosaccharomyces pombe*, nematode, *Caenorhabditis elegans*, mosquito, *Anopheles gambiae*, honeybee, *Apis mellifera*, fruit fly, *Drosophila melanogaster*, sea squirt, *Ciona intestinalis*, zebrafish, *Danio rerio*, chimpanzee, *Pan troglogdytes*, human, *Homo sapiens*, mouse, *Mus musculus*, rat, *Rattus norvegicus*, thale cress, *Arabidopsis thaliana*, oat, *Avena sativa*, soybean, *Glycine max*, barley, *Hordeum vulgare*, tomato, *Lycopersicon esculentum*, rice, *Oryza sativa,* wheat, *Triticum aestivum*, and corn, *Zea mays*. (somewhat conflicting genome statistics at NCBI on several of their own Web pages: http://www.ncbi.nlm.nih.gov/genomes/VIRUSES/viruses.html, http://www.ncbi.nlm.nih.gov/genomes/static/euk_g.html, http://www.ncbi.nlm.nih.gov/genomes/leuks.cgi, http://www.ncbi.nlm.nih.gov/genomes/lproks.cgi, and http://www.ncbi.nlm.nih.gov/genomes/static/gpstat.html).

Over half of the genes in many of these organisms have predicted functions based solely on previously studied bacterial genes, the comparative method in practice. Numerous worldwide genome projects have kept the data coming at alarming rates. The primary nucleotide database in the U.S.A., NCBI's GenBank, has staggering growth statistics (http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html):

Year	BasePairs	Sequences
1982	680,338	606
1983	2,274,029	2,427
1984	3,368,765	4,175
1985	5,204,420	5,700
1986	9,615,371	9,978
1987	15,514,776	14,584
1988	23,800,000	20,579
1989	34,762,585	28,791
1990	49,179,285	39,533
1991	71,947,426	55,627
1992	101,008,486	78,608
1993	157,152,442	143,492
1994	217,102,462	215,273
1995	384,939,485	555,694
1996	651,972,984	1,021,211
1997	1,160,300,687	1,765,847
1998	2,008,761,784	2,837,897
1999	3,841,163,011	4,864,570
2000	11,101,066,288	10,106,023
2001	15,849,921,438	14,976,310
2002	28,507,990,166	22,318,883
2003	36,553,368,485	30,968,418
2004	44,575,745,176	40,604,319
2005	56,037,734,462	52,016,762



A. What are primary sequences?

Remember biology's Central Dogma: DNA → RNA → protein. Primary refers to one dimensional — all of the "symbol" information written in sequential order necessary to specify a particular biological molecular entity, be it polypeptide or polynucleotide. The symbols are the one letter alphabetic codes for all of the biological nitrogenous bases and amino acid residues and their ambiguity codes (see the nice explanatory table at http://virology.wisc.edu/acp/CommonRes/SingleLetterCode.html). Biological carbohydrates, lipids, and structural information are not included within this sequence; however, much of this type of information is available in the annotation associated with primary sequences in the databases.

B. What are sequence databases?

These databases are an organized way to store the tremendous amount of sequence information that is piling up at exponential rates, as seen above. Each database has its own specific format, and access to this information is most easily handled through various software packages and interfaces, either on the World Wide Web (WWW) or otherwise.

In Bethesda, Maryland, United States, the National Center for Biotechnology Information (NCBI, http://www.ncbi.nlm.nih.gov/), a division of the National Library of Medicine (NLM), at the National Institutes of Health (NIH), supports and distributes the GenBank primary nucleic acid sequence database and the GenPept CDS (CoDing Sequence) translations database. They also maintain the derivative RefSeq genome, transcriptome, and proteome sequence databases, and they provide access to data in other sequence databases maintained by the rest of the worldwide supporters. The other primary database organization in the United States is the National Biomedical Research Foundation (NBRF, http://pir.georgetown.edu/nbrf/), an affiliate of Georgetown University Medical Center. They maintain the Protein Identification Resource (PIR, http://pir.georgetown.edu/) database of polypeptide sequences, which has now been consolidated into the UniProt database (the Universal Protein Resource, http://www.uniprot.org/).

The European Bioinformatics Institute, EBI (http://www.ebi.ac.uk/) in Hinxton, Cambridge, United Kingdom, a part of the European Molecular Biology Laboratory (EMBL http://www.embl-heidelberg.de/) in Heidelberg, Germany, maintains the EMBL nucleic acid sequence database. The Swiss Institute of Bioinformatics, SIB, at ExPASy (the Expert Protein Analysis System, http://www.expasy.org/) in Geneva, Switzerland, and EBI jointly support the excellently annotated Swiss-Prot protein sequence database, as well as the minimally annotated TrEMBL (Translations from EMBL — those EMBL translations not yet in Swiss-Prot) protein sequence databases. UniProt, a coalition of EBI, SIB, and PIR, contains sequences from all of the protein databases.

Additional, less well known, sequence databases include sites with the military, with private industry, and in Japan (the DNA Data Bank of Japan, DDBJ, http://www.ddbj.nig.ac.jp/). In most cases data is openly exchanged between the databases so that many sites 'mirror' one another. This is particularly true with GenBank, EMBL, and DDBJ; there is never a need to look in all three places. The same is now true with the creation of UniProt — the best one stop shop for protein sequence data and annotation.

C. What information do they contain, how is it organized, and how is it accessed?

Sequence databases are often mixtures of ASCII and binary data; however, they usually aren't true relational or object oriented data structures. Many expensive proprietary ones are though, and some public domain ones are MySQL. It's a complicated mess with little standardization. Typical sequence databases contain several very long ASCII text files that contain information of all the same type, such as all of the sequences themselves, versus all of the title lines, or all of the reference sections. Binary files usually help 'tie together' all of the files by providing indexing functions. Software specific routines, as exemplified by genome browsers and text search tools, are by far the most convenient method to successfully interact with these databases.

Nucleic acid databases (and TrEMBL) are split into subdivisions based on taxonomy (historical). Protein databases are often split into subdivisions based on the level of annotation that the sequences have.

Annotation sections include extremely valuable information — reference author and journal citations, organism and organ of origin, and the features table. The features table lists all sorts of important regulatory,

transcriptional and translational (CDS coding sequence), catalytic, and structural sites, depending on the database. Actual sequence data usually follows the annotation in most formats.

Becoming familiar with the general format of sequence files for the type of software you want to use can save a lot of grief. Unfortunately most databases and many different software packages have conflicting format requirements. Fortunately there are many excellent format converters available such as ReadSeq (Gilbert, 1993 and 1999). However, most sequence analysis software requires that you specify a proper sequence name and/or database identifier. These are usually discovered with some sort of text searching program, either on the WWW, e.g. Entrez (Schuler, et. al, 1996) or SRS (Sequence Retrieval System, Etzold and Argos, 1993), or with some type of a dedicated local program. This brings up a point, locus names versus accession numbers. The LOCUS, ID, and ENTRY names category in the various databases are different than the Accession number category. Each sequence is given a unique accession number upon submission to the database. This number allows tracking of the data when entries are merged or split; it will always be associated with its particular data. Entry names may change; accession numbers are forever; they just pile up, primary becomes secondary, *ad infinitum*.

D. What changes have occurred in the databases — history and development?

The first well recognized sequence database was Margaret Dayhoff's *Atlas of Protein Sequence and Structure* begun in the mid-sixties (Dayhoff, et al., 1965–1978), which later became PIR (George, et al., 1986). GenBank began in 1982 (Bilofsky, et al., 1986), EMBL in 1980 (Hamm and Cameron, 1986). They have all been attempts at establishing an organized, reliable, comprehensive, and openly available library of genetic sequences. Databases have long-since outgrown a hardbound atlas. They have become huge and have evolved through many changes. Changes in format over the years are a major source of grief for software designers and program users. Each program needs to be able to recognize particular aspects of the sequence files; whenever they change, it's liable to throw a wrench in the works. People have argued for particular standards such as XML (called BSML, Bioinformatics Sequence Markup Language, for sequence data), but it's almost impossible to enforce. NCBI's ASN.1 format (Abstract Syntax Notation One, an International Standards Organization [ISO] data representation format with inter-platform operability) and its Entrez interface were another attempt to circumvent these frustrations. Entrez, EMBL's SRS, found on the WWW at all EMBL outstations, and the Wisconsin Package's LookUp derivative of SRS all search for text in, interact with, and allow users to browse the sequence databases. Both SRS and Entrez provide 'links' to associated databases so that you can jump from, for instance, a chromosomal map location, to a DNA sequence, to its translated protein sequence, to a corresponding structure, and then to a MedLine reference, and so on. They are very helpful!

E. What other types of bioinformatics databases are used?

Specialized versions of sequence databases include sequence pattern databases such as restriction enzyme (e.g. http://rebase.neb.com/) and protease (e.g. http://merops.sanger.ac.uk/) cleavage sites, promoter

sequences and their binding regions (e.g. http://www.gene-regulation.com/pub/databases.html and http://www.epd.isb-sib.ch/), and protein motifs (e.g. http://us.expasy.org/prosite/) and profiles (e.g. http://www.sanger.ac.uk/Software/Pfam/); and organism or system specific databases such as the sequence portions of ACeDb (A *C. elegans* Database http://www.acedb.org/), FlyBase (*Drosophila* database http://flybase.bio.indiana.edu/), SGD (*Saccharomyces* Genome Database http://www.yeastgenome.org/), GDB (the Human Genome Database, http://gdbwww.gdb.org/), and RDP (the Ribosomal Database Project http://rdp.cme.msu.edu/). Many of these organism specific databases present their data in the context of a genome map browser (e.g. the University of California, Santa Cruz, bioinformatics group's human genome browser, http://genome.ucsc.edu/, and the Ensembl project, http://www.ensembl.org/, jointly hosted by the Welcome Trust Sanger Institute and the European Bioinformatics Institute). Map browsers attempt to tie together as many data types as possible using a physical map of a particular genome as a framework that a user can zoom in or out on in order to see more or less detail of any particular loci.

Two other types of databases are commonly accessed in bioinformatics: reference and three-dimensional structure. Reference databases run the gamut from OMIM (Online Mendelian Inheritance In Man, http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=OMIM), that catalogs human genes and phenotypes, particularly those associated with human disease states, and their excellent descriptive database Gene (http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene), that provides a 'genecentric' view of completely sequenced genomes, to PubMed access of MedLine bibliographic references (the National Library of Medicine's citation and author abstract bibliographic database of over 4,800 biomedical research and review journals, http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed). Other databases that could be put in this class include things like proprietary medical records databases and population studies databases.

Finally, the Research Collaboratory for Structural Bioinformatics (RCSB), a consortium of institutions: Rutgers University, the State University of New Jersey; the San Diego Supercomputer Center, University of California, San Diego; and the University of Wisconsin-Madison; supports the three-dimensional structure Protein Data Bank (PDB http://www.rcsb.org/pdb/). The National Institute of Health maintains "Molecules To Go" at http://molbio.info.nih.gov/cgi-bin/pdb as a very easy to use interface to PDB. And NCBI maintains the MMDB (Molecular Modeling DataBase) (http://www.ncbi.nlm.nih.gov/Structure/MMDB/mmdb.shtml) that contains all of the experimentally determined structures from PDB. Other three-dimensional structure databases include the Nucleic Acid Databank at Rutgers (NDB http://ndbserver.rutgers.edu/) and the proprietary Cambridge small molecule Crystallographic Structural Database (CSD http://www.ccdc.cam.ac.uk/products/csd/).

II. So how does one do bioinformatics?

A. Bioinformatics and the Internet: the World Wide Web

Often bioinformatics is done on the Internet through the WWW. This is possible and easy and fun, but, besides being a bit too easy too get sidetracked upon . . . the Web can not readily handle large datasets

or large multiple sequence alignments. These types of datasets quickly become intractable. You'll know you're there when you try. In spite of that . . .

Some of my favorite WWW sites for molecular biology and bioinformatics follow below:

Site	URL (Uniform Resource Locator)	Content
National Center Biotech' Info'	http://www.ncbi.nlm.nih.gov/	databases/analysis/software
PIR/NBRF	http://pir.georgetown.edu/	protein sequence database
ProteinDataBank	http://www.rcsb.org/pdb/	3D mol' structure database
Molecules To Go	http://molbio.info.nih.gov/cgi-bin/pdb/	3D protein/nuc' visualization
IUBIO Biology Archive	http://iubio.bio.indiana.edu/	database/software archive
Univ. of Montreal Genomics	http://megasun.bch.umontreal.ca/	database/software archive
Japan's GenomeNet Server	http://www.genome.ad.jp/	databases/analysis/software
European Mol' Bio' Lab'	http://www.embl-heidelberg.de/	databases/analysis/software
European Bioinformatics Inst'	http://www.ebi.ac.uk/	databases/analysis/software
The Sanger Institute	http://www.sanger.ac.uk/	databases/analysis/software
Swiss Institute Bioinformatics	http://www.expasy.ch/	databases/analysis/software
Human Genome DataBase	http://www.gdb.org/	Human Genome Project
Stanford Genomic Resource	http://genome-www.stanford.edu/	various genome projects
Inst. for Genomic Research	http://www.tigr.org/	microbial genome projects
HIV Sequence Database	http://hiv-web.lanl.gov/	HIV epidemeology seq' DB
The Baylor Search Launcher	http://searchlauncher.bcm.tmc.edu/	sequence search launcher
Pedro's BioMol Res' Tools	http://www.public.iastate.edu/~pedro/research_tools.html	extensive bookmark list
Harvard Bio' Laboratories	http://golgi.harvard.edu/BioLinks.html	nice bookmark list
BioToolKit	http://www.biosupplynet.com/cfdocs/btk/btk.cfm	annotated molbio tool links
Felsenstein's PHYLIP site	http://evolution.genetics.washington.edu/phylip.html	phylogenetic inference
The Tree of Life	http://tolweb.org/tree/	overview of all phylogeny
Ribosomal Database Project	http://rdp.cme.msu.edu/index.jsp	databases/analysis/software
PUMA2 Metabolism	http://compbio.mcs.anl.gov/puma2/cgi-bin/index.cgi	metabolic reconstructions
BIOSCI/BIONET	http://net.bio.net/	biologists' news groups
Access Excellence	http://www.accessexcellence.org/	biology teaching and learning
CELLS alive!	http://www.cellsalive.com/	animated microphotography
Genetics Computer Group	http://www.accelrys.com/products/gcg/	sequence analysis package

B. So what are the alternatives . . . ?

Desktop software solutions — public domain programs are available, but . . . complicated to install, configure, and maintain. User must be pretty computer savvy. So, commercial software packages are available, e.g. MacVector, Sequencher, DNAsis, DNAStar, etc.

But . . . license hassles, big expense per machine, and database access all complicate matters!

C. Therefore, server-based solutions (e.g. the GCG Wisconsin Package) — UNIX server computers

These offer very fast, convenient database access on local server disks, and connections can be made from any networked terminal or workstation anywhere! Public domain solutions also exist, but now a very

cooperative systems manager needs to maintain everything for users. Commercial, server-based software have one license fee for an entire institution, rather than individual licenses for every computer.

But . . . operating system and command line hassles.

1. Communications software

Most all computer systems will have some type of a WWW browser available, be it Microsoft's Explorer, Netscape's Navigator, Mozilla's Firefox, KDE's Konqueror, Opera Software ASA's Opera, Apple's Safari, on *ad infinitum*, it doesn't matter. You can use whatever is on the machine. Unfortunately a Web browser alone is not enough for serious biocomputing. More often than not you will need to directly connect to a server computer using a command line, "terminal," window where you can directly interact with the server computer's operating system. The 'old way' to do this was with a common program called telnet. However, telnet is an unsecure program from which smart hackers can 'sniff' connection account names and passwords. Therefore, in this age of the hacker, most server computers no longer allow telnet connections. A newer program named ssh, for 'secure shell,' encrypts all connections, and is now required for command line access to most servers. ssh comes preinstalled as a part of all modern UNIX operating systems but doesn't come with pre-OS X Macintosh or any Micrsoft Windows machines and, therefore, must be installed on those platforms separately in order to do most server-based biocomputing.

2. X graphics

Furthermore, since ssh is strictly a non-graphical terminal program, and since all Web browsers' graphics capability is inadequate for the truly interactive graphics that much biocomputing software requires, another type of graphical system needs to be present on the computer that you use for much biocomputing. That graphical interface is called the X Window System (*a.k.a.* X11). It was developed at MIT (the Massachusetts Institute of Technology) in the 1980's, back in the early days of UNIX, as a distributed, hardware independent way of exchanging graphical information between different UNIX computers. Unfortunately the X worldview is a bit backwards from the standard client/server computing model. In the standard model a local client, for instance a Web browser, displays information from a file on a remote server, for instance a particular WWW site, also called a Uniform Resource Locator (URL). In the world of X, an X-server program on the machine that you are sitting at (the local machine) displays the graphics from an X-client program that could be located on either your own machine or on a remote server machine that you are connected to. Confused yet?

X-server graphics windows take a bit of getting used to in other ways too. For one thing, they are only active when your mouse cursor is in the window. And, rather than holding mouse buttons down, to activate X items, just <click> on the icon. Furthermore, X buttons are turned on when they are pushed in and shaded, sometimes it's just kind of hard to tell. Cutting and pasting is real easy, once you get used to it — select your desired text with the left mouse button, paste with the middle. Finally, always close X

Windows when you are through with them to conserve system memory, but don't force them to close with the X-server software's close icon in the upper right- or left-hand window corner, rather, always, if available, use the client program's own "File" menu "Exit" choice, or a "Close," "Cancel," or "OK" button.

Nearly all UNIX computers, including Linux, but not including Macintosh OS X, include a genuine X Window System in their default configuration. Microsoft Windows computers are often loaded with X-server emulation software, such as the commercial programs XWin32 or eXceed, to provide X-server functionality. Macintosh computers prior to OS X required a commercial X solution; often the program MacX or eXodus was used. However, since OS X Macintoshes are true UNIX machines, they can use one of a variety of free open source packages such as XDarwin to provide true X Windowing. Perhaps the best X solution for Max OS X is Apple's own X11 package distributed for free from their support pages: http://www.apple.com/downloads/macosx/apple/x11formacosx.html.

3. Text editing

At some point you will have to edit a file; text editing is often a necessary part of computing. This is never that much fun, but always very important. The UNIX operating system always has vi installed. It's a part of the operating system and is very powerful, but quite intimidating. Emacs or pico are often provided as alternatives. Or you can use your own favorite desktop word processing software like Microsoft Word, if you would like, followed by file transfer. Just be sure to "Save As" "Text Only" with "Line Breaks," but don't be surprised if you have subsequent line break problems, unless you can specify UNIX style line breaks.

Native word processing format contains binary control data in it specifying format and so forth; the UNIX operating system can't read it. Saving as text avoids this problem. Editing this way is a two-step process though. After the editing is done, the file needs to be transferred to the UNIX server. Therefore, it makes sense to get comfortable with at least one UNIX text editor. That will avoid the file transfer step, saving some hassle. There are several around, including many driven though a graphical user interface (GUI), but minimally I recommend learning pico (or its nearly identical clone nano).

4. File transfer

Along the lines of secure connections, there are often times when you'll need to move files back and forth between your own computer and a server computer located somewhere else. The 'old' unsecure way of doing this was a program named ftp, for file transfer protocol. Just like telnet, it has the unfortunate attribute of allowing hackers to 'sniff' account names and passwords. Therefore, an encrypted file transfer counterpart to ssh is now required by most servers. That counterpart is called sftp and scp, for 'secure file transfer protocol' and 'secure copy' respectively. It's also included in all modern UNIX operating systems, but not in pre-OS X Macintoshes, nor in Microsoft Windows, so it has to be installed on those computers separately.

III. A basic guide to UNIX for neophytes

Because this is all somewhat confusing to newcomers, here's a UNIX tutorial that we won't take the time to go through today, but I encourage you to do so at some point. I stole this tutorial from the Internet and modified it for bioinformatics use. I am indebted to the countless, but unnamed, contributors — I apologize for my lack of credit giving and flagrant copyright infringement. Hundreds of users worldwide are grateful; thank you.

The original UNIX operating system (OS) was developed in the USA, first by Ken Thompson (no relation) and Dennis Ritchie at AT&T's BELL Labs in the late 1960's; it is now used in various implementations, on many different types of computers the world over. One of the most popular variations is RedHat Linux. RedHat is a commercial distribution of the free, UNIX derived, Open Source Linux OS. Linux was invented in the early 1990's by a student at the University of Helsinki in Finland named Linus Torvalds as a part-time 'hobby.' FreeBSD (from the U.C. Berkley UNIX implementation) is another popular Open Source UNIX OS.

All UNIX OSs are a line-oriented system similar conceptually to the old MS-DOS OS, though many GUIs exist to help drive them. It is possible to use many UNIX computers without ever-learning command line mode and using a "shell" terminal window. However, becoming familiar with some basic UNIX commands will make your computing experience much less frustrating. The shell program is your command line interface to the UNIX OS. It interprets and executes the commands that you type. Common UNIX shells include bash, the C shell, and a popular C shell derivative called tcsh. tcsh, like bash, enables command history recall using the keyboard arrow keys, accepts tab word completion, and allows command line editing. Among numerous UNIX command line guides available on the Internet, there's a very good beginning UNIX tutorial at http://www.ee.surrey.ac.uk/Teaching/Unix/, if you would like to see an alternative approach to what I present.

The UNIX command line is often regarded as very unfriendly compared to other OSs. Actually UNIX is quite straightforward, especially its file systems. UNIX is the precursor of most tree structured file systems including those used by MS-DOS, Microsoft Windows, and the Macintosh OS. These file systems all consist of a tree of directories and subdirectories. The OS allows you to move about within and to manipulate this file system. A useful analogy is the file cabinet metaphor — your account is analogous to the entire file cabinet. Your directories are like the drawers of the cabinet, and subdirectories are like hanging folders of files within those drawers. Each hanging folder could have a number of manila folders within it, and so on, on down to individual files. Hopefully all arranged with some sort of logical organizational plan. Your computer account should be similarly arranged.

A. Generalities

In command line mode each command is terminated by the 'return' or 'enter' key. UNIX uses the ASCII character set and unlike some OSs, it supports both upper and lower case. A disadvantage of using both upper and lower case is that commands and file names must be typed in the correct case. Most UNIX commands and file names are in lower case. Commands and file names should not include spaces nor any

punctuation other than periods (.), hyphens (-), or underscores (_). UNIX command options are specified by a required space and the hyphen character (-). UNIX does not use or directly support function keys. Special functions are generally invoked using the 'Control' key. For example a running command can be aborted by pressing the "Control" key [sometimes labeled "CTRL" or denoted with the karat symbol (^)] and the letter key "c" (think c for cancel). The short form for this is generally written CTRL-C or ^C. Using control keys instead of special function keys for special commands can be hard to remember, the advantage is that nearly every terminal program supports the control key, allowing UNIX to be used from a wide variety of different platforms that might connect to the server.

The general command syntax for UNIX is a command followed by some options, and then some parameters. If a command reads input, the default input for the command will often come from the interactive terminal window. The output from a system level command (if any) will generally be printed back to your terminal window. General UNIX command syntax follows:

```
cmd
cmd -options
cmd -options parameters
```

The command syntax allows the input and outputs for a program to be redirected into files. To cause a command to read from a file rather than from the terminal, the "<" sign is used on the command line, and the ">" sign causes the program to write its output to a file (for programs that don't do this by default, also ">>" appends output to the end of an existing file):

```
cmd -options parameters < input
cmd -options parameters > output
cmd -options parameters < input > output
```

To cause the output from one program to be passed to another program as input a vertical bar (|), known as the "pipe," is used. This character is < shift > < \ > on most USA keyboards:

```
cmd1 -options parameters | cmd2 –options parameters
```

This feature is called "piping" the output of one program into the input of another.

Certain printing (non-control) characters, called "shell metacharacters," have special meanings to the UNIX shell. You rarely type shell metacharacters on the command line because they are punctuation characters. However, if you need to specify a filename accidentally containing one, turn off its special meaning by preceding the metacharacter with a "\" (backslash) character or enclose the filename in "'" (single quotes). The metacharacters "*" (asterisk), "?" (question mark), and "~" (tilde) are used for the shell file name "globbing" facility. When the shell encounters a command line word with a leading "~", or with "*" or "?" anywhere on the command line, it attempts to expand that word to a list of matching file names using the following rules: A leading "~" expands to the home directory of a particular user. Each "*" is interpreted as a