14

Genomics

14.1 The Promise of Genomics

In Chapter 8, we learned that DNA holds the information on how to construct a living organism and make it work. This is achieved by instructing the cell how to make proteins characteristic for that organism, one cell at a time. So biologists became interested in determining the precise sequence of the nucleotides in the DNA.

At first, this was hard to do and took a great deal of time even for small segments. Simple strands of DNA were the first to be attempted, those of viruses and the plasmids of bacteria. In time, new techniques were invented and sequencing became faster. A big step was made with the invention of PCR, *polymerase chain reaction*, for making millions of copies of a strand of DNA.

Still, sequencing the *genome* of an organism, its entire complement of DNA, was beyond reach, except possibly that of a virus. And so it was a wildly ambitious plan when, in 1990, the U.S. government launched the *Human Genome Project* (HGP). The project was to sequence the entire human genome of 3 billion base pairs to an accuracy of one error in 10,000 bases.[1] The science of genomics was born, and with it, biology is forever changed.

Genomics is the science of understanding the information contained in the genomes of the world's organisms. Genomics has engendered whole new branches of biology: proteomics, comparative genomics, genomic medicine, pharmacogenomics, and structural genomics among them. Besides the biology, an essential ingredient in genomics is computation. Genomics has been called "computational biology."

How is DNA sequenced?

At the time it was announced, the Human Genome Project seemed an insurmountable task that would require at least 20 years to complete. But in fact, the job was done in less than 10. This was possible due to enormous advances in the technology of

[1] The original plan of one error in 100,000 bases, a somewhat arbitrary figure, was eased to reduce costs.

R.W. Shonkwiler and J. Herod, *Mathematical Biology: An Introduction with Maple and Matlab*, Undergraduate Texts in Mathematics, DOI: 10.1007/978-0-387-70984-0_14,

nucleotide sequencing. Today automated machines can sequence 600 to 800 bp (base pairs) in an hour and issue the results.

The basis for DNA sequencing today is the approach discovered by Frederick Sanger. The key element in this approach is the use of DNA polymerase, which adds complementary bases to a single-strand DNA template, and dideoxynucleotide, ddNTP, which stops the addition of bases. There is a different dideoxynucleotide for each base ddATP, ddTTP, ddCTP, and ddGTP.

Neglecting technical issues, such as the use of primers to get the reaction started, the essentials of the method are these. Four reaction vessels are used, one for each of the four bases. Consider the "A" reaction vessel. It includes complementary single strands of the DNA to be sequenced, a supply of the four bases, and a small amount of ddATP. By the action of DNA polymerase, bases are added one by one to these strands, forming partially complete double-stranded DNA of growing length. If by chance a ddATP is added where an A should have gone, then the chain is "frozen": No more bases will be added, and the length will not increase.

At the end of the reaction, the entire content of the vessel is now separated by electrophoresis in capillary gel tubes. Shorter segments will move farther than longer ones, and they all will end with a ddATP. For example, if the first complementary base in the sequence is A (so the original base is T), then the double strand of primer followed by the T–ddATP pair will move to the end of the capillary tube.

By lining up the four capillary tubes from each reaction and noting the tube of the segment that migrated farthest, say the "A" tube, and the tube of the segment that migrated second farthest, say the "G" tube, and the third farthest, maybe "G" again, and so on, the complimentary sequence can be read, in this case

AGG . . . .

Originally, the positions of the migrating fragments were determined by radioactive labeling, but today fluorescent dyes are used instead, a different color for each base. The DNA sequencing machine can determine the colors as the four capillary gels pass by the reading window.

Of course, only fragments of a genome can be sequenced this way in any one run. Ultimately the fragments must be stitched together. There are two main approaches to this step, *map-based sequencing* and the *whole genome shotgun* (WGS) technique.

In the first, a chromosome to be sequenced is divided into large segments of about 150 kilobases. These segments have generous overlap. The whole collection is called a library. The segments are then cloned using *bacterial artificial chromosomes* (BACs), a man-made piece of DNA that can be replicated in bacteria. Before sequencing the segments, their order must first be worked out. As an aid in this step, each segment is "fingerprinted," or given a unique tag. With their order known, these segments are then sequenced. For this they are broken up again into subsegments of about 1,500 bases long. These too have overlap. The subsegments are sequenced to a length of about 500 bp from each end. This generates thousands of A, T, C, G sequences, which are pieced together by computer algorithms that make essential use of the overlaps.

In the whole genome shotgun approach, the entire chromosome is randomly shredded into segments about 2,000 bp long. Meanwhile, separately, a copy is broken into segments 10,000 bp long. All segments are sequenced to a length of 500 bp from each end; this generates millions of sequences. Then, as above, very powerful parallel computers fit all the pieces together.

In both approaches, sequencing to 500 base pairs would seem to leave gaps. But due to the overlap, most of these are filled in. Any remaining gaps are flagged in the finishing process and further sequencing is expended to fill them.

14.2 Structural Genomics

The aim of structural genomics is to determine the three-dimensional structure of a protein from its amino acid sequence. This effort extends to RNA as well, since some of the genome codes for RNA as the end result. Structural genomics has some overlap with *proteomics*, which studies the function of proteins in addition to elucidating protein structure. Of course, the function of a protein is closely related to and determined by its structure. Proteins that serve as enzymes depend on having a carefully crafted topography and electronic presentation along their active surfaces. Those functioning as cell membrane pores rely on their array of hydrophobic and hydrophilic parts, and those intended as structural materials typically have alpha-helices or beta-pleated sheets.

A protein starts out as a linear sequence of amino acids. This is called its *primary structure*. Even before the protein is completely synthesized, the partially assembled sequence begins changing shape to bring about its ultimate, biologically active conformation. This process is called *protein folding*. The final arrangement in space of all its atoms is the *tertiary* structure of a protein. Protein *secondary structure* refers to the three-dimensional form of substructures within the protein, such as alpha-helices and beta-pleated sheets.

With every possible configuration of a molecule, proteins included, there is an associated energy. The graph of this energy vs. configuration is its *energy landscape*; it is a multidimensional graph. Important factors determining the energy landscape are hydrogen-bonding, dipolar interactions, Van der Waals attraction, and disulfide bonds between the amino acid side chains. The energy landscape also depends on the molecule's environment. For a protein, this usually means the temperature, pH, and solvent in which it is located.

For simple molecules the energy landscape is a simple bowl-shaped, or monomodal, surface. Immediately upon formation, such a molecule assumes a configuration at or near the bottom of the bowl, its energy minimum. For complicated molecules the energy landscape is multimodal, meaning it has multiple bowl-shaped regions. The energy of one basin will generally be different from that of another. The correct tertiary structure of a protein is that of its lowest energy basin.

The most successful method of determining protein structure has been *X-ray crystallography*. This method requires high-quality crystals of the protein subject. These are then placed in the path of an X-ray source, and the resulting diffraction

pattern is recorded. Using a great deal of skill and insight, the shape of the protein can be worked out. The structure of myoglobin, published in 1959, took 20 years for the job. With modern methods, heavily computational, the time is much shorter.

X-ray crystallography is limited by its need for high quality crystals and for an X-ray source. With regard to the second of these, today that source is usually a synchrotron, and they are in limited supply. The first requirement can be a bigger hurdle, since it is very hard to crystallize proteins that function in a hydrophobic environment.

Another method makes use of NMR technology. This relies on the spin of the nuclei of hydrogen, carbon, and nitrogen atoms. A protein being tested is in aqueous solution and is therefore closer to its *in vivo* condition. The drawbacks of this method are its poor resolution; the data obtained are very difficult to analyze, and this, in turn, imposes a severe limit on protein size; and finally, the sample must be at high concentration.

The ideal method would allow the structure to be worked out from its amino acid sequence and environment alone, computationally. This is the *ab initio* method. One line of attack is to simulate the events of protein folding. The physics of the simulation are well understood; the problem is that the computational requirement for anything but the very smallest protein is beyond present-day capability. Another possibility is to combine computational techniques with experimentally derived substructures. So far, the computational method has had only limited success.

The Protein Data Bank (PDB) is a repository for three-dimensional protein structures. This is a consortium of the San Diego Supercomputing Center, Rutgers University, and the National Institute of Standards and Technology. Every protein in the repository has been validated by the staff of PDB for physical plausibility. These data are available online at http://www.rcsb.org/pub/.

Structure-based drug design.

Understanding the binding properties of proteins has become an integral part of modern drug discovery. The goal is to produce a molecule that will bind, or *dock*, with an active site on the target protein. To succeed, a thorough understanding of the molecular recognition between the active site and an interacting molecule is required. The active site of the protein is a space to be filled with a molecule that complements it in terms of shape, charge, and other binding components.

An important tool in the design process is software that visualizes how the drug interacts with these components three-dimensionally. This knowledge can then guide the chemistry, called *computational chemistry*. Varieties of promising compounds are made in parallel and checked by means of some sort of activity or binding assay. This high-throughput method rapidly sorts through all these possibilities to find out which ones really work.

Prior to the use of structural genomics, combinatorial chemistry by itself was found to be an ineffectual strategy. This is because the number of molecules that can be made by the technique is still infinitesimally small compared to all the possibilities. Of the huge numbers of compounds made, nearly all were irrelevant.

Likewise, designing drugs *de novo*, purely computationally, turned out to be equally ineffective. Docking software suffered from an incomplete understanding of the energy, electron densities, and thermodynamics of the interacting molecules. The molecules derived from such studies did not conform to the expectations in potency when made.

Structure-based drug design is at its most powerful when coupled with combinatorial techniques. One of the most effective uses of structure is in optimization. An understanding of structure is key to devising methods for modifying the drug molecule to get the desired properties. Potential compounds can be screened on the computer in a very short period of time.

The binding mode of synthesized compounds to the target is often verified using X-ray crystallography and NMR. Three-dimensional structures produced by X-ray crystallography have become an integral part of the drug discovery process because of changes that have increased throughput. Synchrotron beam lines have made it possible to run a large number of crystals all at once because crystallographic information can be obtained from smaller crystals than was previously possible. The verification results are feedback into the prediction simulations to improve the process.

Speeding up drug discovery is only one of the goals of structure-based drug design. Another is to improve the quality of the drug, to determine molecules that have better pharmacological properties.

14.3 Comparative Genomics

Comparative genomics is the science of understanding and making use of similar genomic segments between two, several, or a group of organisms. Such segments can help locate gene coding and regulatory regions, clarify the importance of gene expression, elucidate protein–protein interactions, and help unravel gene evolution. As a warmup to decoding the human genome, the Human Genome Project tested and refined their techniques by decoding the genomes of brewer's yeast, *Saccharomyces cerevisiae*, the roundworm *Caenorhabditis elegans*, the plant *Arabidopsis thaliana* of the mustard family, and the fruit fly *Drosophila melanogaster*. Thus began the science of comparative genomics.

Synteny refers to the preserved order of genes, on a chromosome, between related organisms. Synteny can be derived from genetic or physical maps of the genome. A genetic map provides a distance between two genes on the same chromosome by noting the frequency of their recombination. During meiosis, chromosomes become paired, and in the process of duplication, they may break. It can then happen that when the chromosome is put back together, a broken segment is swapped with the matching segment of the homologous chromosome. This form of recombination is called *crossing over*. The greater the distance between two genes on the same chromosome, the more likely crossing over will occur between them. In turn, these recombination rates are used to construct a genetic map along the chromosome.

A physical map gives the location of identifiable landmarks along a chromosome measured in base pairs. Examples of landmarks are restriction enzyme cutting sites

and the location of genes. Physical maps are constructed by cutting the DNA segment between landmarks into overlapping fragments called *contigs*. These are then inserted into a *vector*, an agent for cloning the DNA, such as a phage virus that can accommodate up to 45 kilobases, or as a *bacterial artificial chromosome* (BAC) spliced into *Escherichia coli* cells. These can accommodate up to 300 kb. The contigs are now sequenced, as for example by the Sanger–Hood method.

Synteny appears to be common. Long stretches of the human and mouse genomes show a remarkable degree of synteny. As a result, a human gene having a function similar to a gene of the mouse can be approximately located within the human genome if the analogue's location in the mouse genome is known.

But comparison between genomes is much more useful than that. DNA is subject to background mutation or random nucleotide drift. Several lines of research point to a gene mutation rate of about 10^{-7} per replication (as we obtained in the retinoblastoma study, Section 12.4). The rate is approximately the same even across biological divisions: Archaea, Bacteria, and Eukaryota. However, most gene mutations are detrimental to the organism. As a result, two gene sequences that started out the same tend to be conserved even over millions of years of independent evolution. Their coding sequences as well as their regulatory regions, everything functionally important, resist random drift.

Not so for the other segments of the genome. Noncoding DNA that does not have a structural or regulatory function tends to diverge much more rapidly than coding DNA. By comparing genomes, especially across related species, much can be learned about the location of exons, introns, and the regulatory segments of genes (recall Section 8.3 for the definitions of these terms). When the genome of *Fugu* (a type of blowfish) was compared with the human genome, over 1,000 new human gene candidates were discovered.

In genomics, *homology* refers to the similarity in DNA or protein sequences between individuals of the same species or among different species. In Section 14.6, we will learn how to search genomes for matching segments. This is a powerful tool for finding homologies. A surprising discovery resulting from such a homology search is that the sea squirt *Ciona intestinalis* has a homolog with the gene endoglucanase of the plant *Arabidopsis*. This gene is believed to have an important role in cellulose synthesis. The search was undertaken because the coat of *Ciona* is made primarily of cellulose.

In the science to which comparative genomics has given rise, the term *ortholog* refers to genes found in two species having a common ancestor. On the other hand, *paralogs* are similar genes found within a single species, copies having been created by duplication events. A challenge of comparative genomics is to discern true orthologs and not just similar DNA segments.

The utility of comparative genomics extends to gene expression as well. Transcription, the synthesis of mRNA, involves three main events: initiation, elongation, and termination. The main work is done by RNA polymerase. In the elongation step, RNA polymerase adds complementary nucleotides to the $3'$ end of the growing RNA chain. Termination occurs when RNA polymerase encounters a termination signal within the gene. The initial mRNA transcript is processed by splicing out introns

and ligating exons into the mature mRNA. Gene expression therefore is primarily controlled by initiation, which is a fairly involved process.

It begins with the binding of RNA polymerase to the double-stranded DNA at a site called the *promoter*. This is usually a short distance upstream of the start of transcription and contains the nucleotide sequence TATA, called the *TATA-box*. The DNA becomes single-stranded at this site. Note that a TA-rich area is more easily unwound than a CG area, since T–A is doubly hydrogen-bonded, while C–G is triply bonded.

Besides RNA polymerase, the help of other proteins, called *transcription factors*, is also required for initiation. Gene expression may also be regulated by repressors that must vacate their DNA binding sites before transcription can begin. Identifying regulatory sequences in a sequenced genome is a challenging problem.

But as we saw above, because the regulatory sites are part of the functional apparatus of the gene, they are likely to be conserved across species. By aligning orthologs from two or more species and noting matching intergenic sequences, candidate regulatory segments can be uncovered. These conserved segments are called *phylogenic footprints*.

Comparative genomics can be used to to identify proteins that interact with each other. The method exploits an insight that if multiple protein-coding segments in one species are coded as a single protein in another, then it is likely those proteins interact. The technique has been effective in finding interacting proteins in the three bacteria species *Escherichia coli*, *Haemophilus influenzae*, and *Methanococcus jannaschii*.

Probably the biggest use made of comparative genomics is in elucidating both biological and gene evolution. Before the human genome was available, it was thought that the number of human genes would be around 100,000. It did not turn out that way; it is now believed there are only 30,000 human genes. This number is not very different from those of other organisms such as chimpanzees and even the plant *Arabidopsis*. Furthermore, humans' and chimps' genomes are about 98% similar. But humans and chimps have significant differences. So how can these facts be reconciled?

One way this is possible is by differences in gene expression. In fact, it has been shown that there is over a fivefold difference in gene expression related to brain tissue between chimps and humans. This shows that gene expression is an important factor in biological evolution and that coopting existing genes is as important as creating new ones.

14.4 Genomics in Medicine

The human genome codes for about 30,000 genes. Of course, a base pair mutation in any one of them has the potential for causing disease. Hence we see that there can be a very large number of genetic diseases. Table 14.4.1 lists some of these diseases along with the gene whose mutation causes it. As of the year 2000, there were 1,000 known heredity disorders.

Table 14.4.1. Genes and their diseases (when mutated). (Source: www.ornl.gov/sci/techresources/Human_Genome/medicine/genetest.shtml.)

APKD	Adult polycystic kidney disease
AAT	Alpha-1-antitrypsin deficiency
ALS	Amyotrophic lateral sclerosis (Lou Gehrig's disease)
APOE	Alzheimer's disease
AT	Ataxia telangiectasia
CMT	Charcot–Marie–Tooth
CAH	Congenital adrenal hyperplasia
CF	Cystic fibrosis
DMD	Duchenne–Becker muscular dystrophy
DYT	Dystonia
FA	Fanconi anemia group C
FVL	Factor V-Leiden
FRAX	Fragile X syndrome
GD	Gaucher disease
HEMA, HEMB	Hemophilia A and B
HFE	Hereditary hemochromatosis
CA	Hereditary nonpolyposis colon cancer
HD	Huntington's disease
BRCA 1, BRAC 2	inherited breast and ovarian cancer
MD	Myotonic dystrophy
NF1	Neurofibromatosis type 1
PKU	Phenylketonuria
PW/A	Prader–Willi–Angelman syndromes
SS	Sickle-cell disease
SMA	Spinal muscular atrophy
SCA1	Spinocerebellar ataxia
TS	Tay–Sachs disease
THAL	Thalassemias

Genomics offers the possibility of curing heredity diseases. First, the cause of the disease must be worked out. This is easiest for *monogenic diseases*, or those attributable to a single faulty gene. The victim either lacks or has too little of some crucial protein, or has a flawed version, or some protein is made to excess. The initial task is discovering the offending protein. Sometimes this can be done via the genome itself. Another possibility is to identify the responsible protein directly and reverse the transcription, translation process. This approach works through the intermediary mRNA, which is another clue to the gene. This is the method of *expressed sequence tags* (ESTs). Complicating the EST approach is that genes are seldom contiguous, and in any case it will not find initiators or promoters.

Along with the base pair sequence, the location of the gene within the genome must also be found—on which chromosome and where along the chromosome. This step is greatly aided by having the complete human genome sequence and is mostly a

software task. Knowing the location includes knowing the exons, introns, promoters, and initiators. Then the task of deducing the problem with the faulty gene can begin.

This process has been worked through for Huntington's disease, a disease of progressive degeneration of brain cells in certain areas of the brain. The gene, HD, is on the fourth chromosome and is dominant. In the normal form, the trinucleotide CAG is repeated within the sequence up to 26 times. In Huntington's patients, the trinucleotide is repeated 36 to 125 times. With repetition between 36 to 39 times, the individual may or may not contract the disease, but with repeats over 39, disease is certain and symptoms occur sooner with increasing number of repeats.

Another genetic disease for which the responsible protein and corresponding gene has been worked out is *cystic fibrosis* (CF). It is caused by a defect in a gene called the cystic fibrosis transmembrane conductance regulator (CFTR) gene. In 1989, scientists identified the CF gene on chromosome 7. About 70% of the mutations observed in CF patients result from deletion of the three base pairs CTT (and GAA on the complementary strand) in CFTR's nucleotide sequence. This deletion causes loss of the amino acid phenylalanine (even though CTT is not a codon for phenylalanine, the error is between frames) located at position 508 in the CFTR protein; as a result, this mutation is referred to as *delta F*508.

The normal CFTR protein product is a chloride channel protein found in membranes of cells that line passageways of the lungs, liver, pancreas, intestines, reproductive tract, and skin. It controls the movement of salt and water in and out of these cells. In people with CF, the gene does not work effectively, resulting in a sodium and chloride ion imbalance. This in turn creates a thick, sticky mucus layer that cannot be removed by cilia and traps bacteria, resulting in chronic infections.

The faulty gene is autosomal (not sex-linked) recessive, so one must have two defective genes to have the disease. It also means that individuals with one defective gene are carriers.

How can these and other genetic diseases be cured? At the present time, genetic tests are available for the diseases listed in Table 14.4.1. Consequently, prospective parents can ensure that their child will be free of these diseases. If they do not have an aberrant gene themselves, then neither will their child.

On the other hand, through genetic counseling, couples can explore options. One of these is to forgo having children. Another is adoption. But there is also another possibility, *in vitro* fertilization and genotyping of embryos. In the case of an autosomal dominant disease, such as Huntington's, each embryo has a 50% chance of being free of the disease if one parent is afflicted and 25% chance even if both are. Similarly for an autosomal recessive disease, such as cystic fibrosis, even if both parents are carriers. This procedure was first performed in 1992 for cystic fibrosis carriers.

It should be noted that some people have ethical issues with *in vitro* fertilization, since several human embryos are created and genetically tested. Unused embryos are discarded.

Remedies are also sought for those with existing disease. Most of these are in the research stage of development. It may be possible to create a drug for a missing or ineffective protein. A more daring possibility is *gene therapy*. The goal here is to deliver a functional gene to the cells of the body that need it. At this time, the most

effective means for delivering therapeutic genes have been viruses. Adenoviruses are frequently used because they are associated with relatively mild illnesses or with no illness at all. Retroviruses have also been used.

Although there have been some successes with gene therapy, there have also been some tragic failures. In one case, a healthy volunteer died as a result of an exaggerated immune response to an adenovirus. In another, a child being treated for SCID developed leukemia following gene therapy with a retrovirus. It is believed the virus activated a cancer-causing gene.

Genomics is a major tool in cancer research. In one application, EST sequencing is undertaken for both normal and tumor cells. The EST method of sequencing, via mRNA, targets gene expression. Once in the database, the sequences can be analyzed using nucleotide matching and other bioinformatics software, a topic we will take up later in this chapter.

Another approach to studying cancer is by means of *microarrays*. A microarray is used to determine, in a single experiment, the expression levels of hundreds or thousands of genes. The array is made up using robotic equipment to spot cDNA samples on a microscope slide or other substrate. *Complementary DNA*, or *cDNA*, is synthesized from an mRNA template or, turning this around, a segment of DNA used to bind to mRNA as here. Fluorescent labels are attached to DNA or mRNA taken from a cell under study and allowed to bind to the cDNA. The expression levels of these probes can be determined by measuring the amount bound to each site on the array. With the aid of a computer, that amount is precisely measured, generating a profile of gene expression in the cell.

The gene expression profile of metastasizing medulloblastoma (a largely childhood cancer of the brain) was determined by microarray experiments. One of the genes enhanced under metastasis was one coding for *platelet derived growth factor receptor alpha* (PDGFRα). In this way, the experiment pointed to candidate drug targets for combating metastasis.

Genomics is helping to fight microbial diseases too. In 2003, sequencing of the genome of *P. falciparum* was completed. It took an unusually long time, six years, since it contains a high A+T content, which caused problems for the software that joins overlapping segments. The organism has 5,300 protein-coding segments, which can now be studied in detail genetically. It is hoped that this will result in opportunities for new drug targets and vaccine candidates.

In fact, this has already been done and has resulted in the discovery that an existing drug, *DOXP reductoisomerase*, is effective in curing malaria in rats. The discovery began by searching for homologies between *P. falciparum*'s genes and those in other organisms. It was learned that the parasite has a homolog to the enzyme DOXP reductoisomerase found in bacteria that is part of the biosynthetic pathway synthesizing isopentenyl diphosphate, a vital precursor chemical. Animals use a different pathway from DOXP for this synthesis, providing an opportunity for a drug with few side effects. By chance, the DOXP reductoisomerase drug was already on the market as an antibacterial drug.

Alternatively, the disease might be overcome through manipulation of its mosquito vector. The genome of *Anopheles gambiae* was sequenced in 2002. A genetically

modified mosquito was subsequently created with the ability to block the transmission of the parasite to humans. This was done by inserting a gene into the mosquito to produce a 12-amino-acid peptide that blocks the oocysts' entry into the mosquito's salivary glands; see Section 11.2. This is truly a novel and unforeseen application of genomics and a hallmark of good science.

14.5 Protein Substitution Matrices

Eventually, mutations occur in the genome leading to mistakes in the encoded amino acid sequence. But these mistakes have varying effects. Most will render the protein dysfunctional and compromise its host organism. However, some amino acid substitutions are effectively harmless. And there is a spectrum of possibilities between these two extremes.

This is possible because, with respect to specific attributes, subgroups of the 20 amino acids are similar to each other. Thus with respect to size, glycine, alanine, serine, and cysteine are alike in that they are small. The peptides serine, threonine, and tyrosine are hydroxyl. And so on for several other attributes. The main antithetic pairs are large/small, hydrophobic/hydrophilic, aliphatic/aromatic, acidic/basic, and sulphur-containing/non-sulphur-containing. An amino acid substitute, sufficiently similar to the original, might not be disruptive and could even be beneficial.

To quantify this phenomenon, Margaret Dayhoff analyzed 1572 substitutions between closely related proteins. Frequencies of substitutions were noted and tabulated as shown in Table 14.5.1. Along the left side of the table are the amino acids denoted by their standard three-letter designation. Along the top are those same amino acids denoted by their single-letter designations. (These were also introduced by Dayhoff. To find the familiar three-letter protein for "D," say, move down the D column to the dash and then across to "asp." Note that Table 14.6.1, given later on in this chapter, gives the correspondence directly and includes "wildcards.") In the body of the table are the frequencies of substitution. The table is triangular, since it is assumed that the substitutions can go both ways just as well, e.g., methionine for leucine just as well as leucine for methionine.

This exchange-count table has a profound significance. With some mathematical processing, it can become the basis of a model for protein evolution. This is because the table's frequency data expresses the extent to which residues change as a result of accepted mutations as compared with purely random exchanges.

The model is *Markovian*, meaning it is based on assumptions that certain events, as given next, are independent and not affected by the other events of the model. The first, neighbor independence, is that each residue mutates independently; its mutation rate is uncoupled from that of the other residues. The second is positional independence, the probability that residue i mutates to residue j depends only on i and j and not on where i appears in the sequence. And the third, historical independence, means that mutation is memoryless, the probability of mutation at each site does not depend on how the present peptide sequence came about.

Table 14.5.1. Dayhoff exchange counts.

	G	A	V	L	I	M	C	S	T	N	Q	D	E	K	R	H	F	Y	W	P
gly	—																			
ala	58	—																		
val	10	37	—																	
leu	2	10	30	—																
ile		7	66	25	—															
met	1	3	8	21	6	—														
cys	1	3	3		2		—													
ser	45	77	4	3	2	2	12	—												
thr	5	59	19	5	13	3	1	70	—											
asn	16	11	1	4	4			43	17	—										
gln	3	9	3	8	1	2		5	4	5	—									
asp	16	15	2		1			10	6	53	8	—								
glu	11	27	4	2	4	1		9	3	9	42	83	—							
lys	6	6	2	4	4	9		17	20	32	15		10	—						
arg	1	3	2	2	3	2	1	14	2	2	12	9		48	—					
his	1	2	3	4			1	3	1	23	24	4	2	2	10	—				
phe	2	2	1	17	9	2		4	1	1					1	2	—			
tyr		2	2	2	1		3	2	2	4			1	1		4	26	—		
trp				1				2							3		1	1	—	
pro	5	35	5	4	1		1	27	7	3	9	1	4	4	7	5	1			—

The importance of the Markovian assumption is this: Given a mutation matrix M of probabilities m_{ij} that amino acid i will mutate to amino acid j over some evolutionary time period, then in two such time periods the mutation probabilities are given by M^2, and in three time periods by M^3, and so on.

The time period chosen by Dayhoff is that required to give an average amino acid mutation rate of 1%. The corresponding matrix is called the *PAM* 1 *matrix* (point accepted mutation). It is possible to compute the PAM1 matrix starting from the exchange matrix of Table 14.5.1, but one must first construct the missing diagonal elements; these are the frequencies of an amino acid not changing. Then, too, the entries above the diagonal have to be added, for example, by assuming that amino acid j changes to i with the same rate that i changes to j, which is the assumption that is, in fact, made. After that, to be a probability matrix, the exchange matrix has to be normalized so its rows sum to 1. For brevity, we will instead start with the matrix in Table 14.5.2, for which these steps have already been done.

Constructing PAM 1.

We start by figuring the average mutation rate. Each diagonal entry is the probability that its residue does not mutate, so the difference from 1 is the probability that it does. These have to be weighted according to the frequency of the amino acid. These are given in Table 14.5.3 and compiled by counting the number of occurrences of the various amino acids in biological proteins (from Jones, Taylor, and Thornton [2]).

Table 14.5.2. Assumed mutation matrix M.

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	.44	.00	.02	.04	.00	.09	.00	.01	.01	.02	.00	.02	.05	.01	.00	.12	.09	.06	.00	.00
C	.00	.96	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.02	.00	.00	.00	.00
D	.02	.00	.68	.13	.00	.02	.01	.00	.00	.00	.00	.08	.00	.01	.01	.02	.01	.00	.00	.00
E	.04	.00	.13	.68	.00	.02	.00	.01	.02	.00	.00	.01	.01	.06	.00	.01	.00	.01	.00	.00
F	.00	.00	.00	.00	.89	.00	.00	.01	.00	.03	.00	.00	.00	.00	.00	.01	.00	.00	.00	.04
G	.09	.00	.02	.02	.00	.72	.00	.00	.01	.00	.00	.02	.01	.00	.00	.07	.01	.02	.00	.00
H	.00	.00	.01	.00	.00	.00	.86	.00	.00	.01	.00	.03	.01	.04	.02	.00	.00	.00	.00	.01
I	.01	.00	.00	.01	.01	.00	.00	.77	.01	.04	.01	.01	.00	.00	.00	.00	.02	.10	.00	.00
K	.01	.00	.00	.02	.00	.01	.00	.01	.73	.01	.01	.05	.01	.02	.07	.03	.03	.00	.00	.00
L	.02	.00	.00	.00	.03	.00	.01	.04	.01	.78	.03	.01	.01	.01	.00	.00	.01	.05	.00	.00
M	.00	.00	.00	.00	.00	.00	.00	.01	.01	.03	.91	.00	.00	.00	.00	.00	.00	.01	.00	.00
N	.02	.00	.08	.01	.00	.02	.03	.01	.05	.01	.00	.65	.00	.01	.00	.07	.03	.00	.00	.01
P	.05	.00	.00	.01	.00	.01	.01	.00	.01	.01	.00	.00	.82	.01	.01	.04	.01	.01	.00	.00
Q	.01	.00	.01	.06	.00	.00	.04	.00	.02	.01	.00	.01	.01	.77	.02	.01	.01	.00	.00	.00
R	.00	.00	.01	.00	.00	.00	.02	.00	.07	.00	.00	.00	.01	.02	.81	.02	.00	.00	.00	.00
S	.12	.02	.02	.01	.01	.07	.00	.00	.03	.00	.00	.07	.04	.01	.02	.47	.11	.01	.00	.00
T	.09	.00	.01	.00	.00	.01	.00	.02	.03	.01	.00	.03	.01	.01	.00	.11	.64	.03	.00	.00
V	.06	.00	.00	.01	.00	.02	.00	.10	.00	.05	.01	.00	.01	.00	.00	.01	.03	.69	.00	.00
W	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.00	.99	.00
Y	.00	.00	.00	.00	.04	.00	.01	.00	.00	.00	.00	.01	.00	.00	.00	.00	.00	.00	.00	.92

Table 14.5.3. Amino acid frequencies.

L	0.091	E	0.062	R	0.051	Y	0.032
A	0.077	T	0.059	P	0.051	M	0.024
G	0.074	K	0.059	N	0.043	H	0.023
S	0.069	I	0.053	Q	0.041	C	0.020
V	0.066	D	0.052	F	0.040	W	0.014

It is listed in order of most to least frequent; thus leucine makes up over 9% of all proteins.

So the average mutation rate for matrix M is

$$\begin{aligned}\text{average mutation rate} &= \sum_i f_i(1 - m_{ii}) \\ &= 0.077 * (1 - 0.44) + \cdots + 0.032 * (1 - 0.92) = 0.247,\end{aligned}$$

or 24.7%.

But for PAM1, the average mutation rate must be 1%. Since powers of PAM1 give mutations over multiples of the basic time period, we must find k such that

$$(\text{PAM1})^k = M, \quad \text{or} \quad \text{PAM1} = M^{1/k}.$$

This shows that PAM1 is some appropriate root of M for which the average mutation rate is 1%, a computationally intensive calculation but otherwise not a problem. The result is shown in Table 14.5.4.

Table 14.5.4. PAM1 matrix.

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	9867	3	10	17	2	21	2	6	2	4	6	9	22	8	2	35	32	18	0	2
C	1	9973	0	0	0	0	1	1	0	0	0	0	1	0	1	5	1	2	0	3
D	6	0	9859	53	0	6	4	1	3	0	0	42	1	6	0	5	3	1	0	0
E	10	0	56	9865	0	4	2	3	4	1	1	7	3	35	0	4	2	2	0	1
F	1	0	0	0	9946	1	2	8	0	6	4	1	0	0	1	2	1	0	3	28
G	21	1	11	7	1	9935	1	0	2	1	1	12	3	3	1	21	3	5	0	0
H	1	1	3	1	2	0	9912	0	1	1	0	18	3	20	8	1	1	1	1	4
I	2	2	1	2	7	0	0	9872	2	9	12	3	0	1	2	1	7	33	0	1
K	2	0	6	7	0	2	2	4	9926	1	20	25	3	12	37	8	11	1	0	1
L	3	0	0	1	13	1	4	22	2	9947	45	3	3	6	1	1	3	15	4	2
M	1	0	0	0	1	0	0	5	4	8	9874	0	0	2	1	1	2	4	0	0
N	4	0	36	6	1	6	21	3	13	1	0	9822	2	4	1	20	9	1	1	4
P	13	1	1	3	1	2	5	1	2	2	1	2	9926	8	5	12	4	2	0	0
Q	3	0	5	27	0	1	23	1	6	3	4	4	6	9876	9	2	2	1	0	0
R	1	1	0	0	1	0	10	3	19	1	4	1	4	10	9913	6	1	1	8	0
S	28	11	7	6	3	16	2	2	7	1	4	34	17	4	11	9840	38	2	5	2
T	22	1	4	2	1	2	1	11	8	2	6	13	5	3	2	32	9871	9	0	2
V	13	3	1	2	1	3	3	57	1	11	17	1	3	2	2	2	10	9901	0	2
W	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	1	0	0	9976	1
Y	1	3	0	1	21	0	4	1	0	1	0	3	0	0	0	1	1	1	2	9945

Scoring matrices.

Suppose two protein sequences from different species are nearly the same but a small number of amino acids are different; one of them is alanine (A), and the other glycine (G). Is the difference by chance or is it an accepted point mutation? If it is by chance alone, then the probability of its occurrence is the product of the frequency of A times that of G,

$$\text{chance alone} = f_{\mathrm{A}} f_{\mathrm{G}}.$$

If, however, it is due to the forces giving rise to the PAM1 matrix, then the probability is calculated as the probability of A's occurrence times the mutation probability of A to G,

$$f_A M_{\mathrm{AG}}.$$

A convenient way to compare these in a single number is their quotient,

$$r_{\mathrm{AG}} = \frac{\Pr(\text{A to G via accepted mutation})}{\Pr(\text{A to G by chance})} = \frac{M_{\mathrm{AG}}}{f_{\mathrm{G}}}.$$

A ratio exceeding 1 means that the process of accepted mutations is at work; a ratio less than 1 means that it is more likely pure chance. So to do the evaluation for the whole protein and its pair, we need the ratios r_{ij} for each pair of amino acids i and j. The matrix $R = [r_{ij}]$ is called the *odds ratio* matrix.

This evaluates one of the residue differences, and of course, each difference can be scored in the same way, but how are they to be combined? Treating them as independent from one another, to combine independent probabilities, one multiplies them. But it would be nicer to add the individual scores instead. Mathematically, multiplying numbers is equivalent to adding their logarithms (and exponentiating the result). This brings us to the final form; a *scoring matrix* is the elementwise logarithm of the odds matrix. (Almost: Each element might be multiplied by a constant factor to keep the values around 1; the factor is not so important, since the scores are for comparison purposes; no need to exponentiate afterward either for the same reason.) This *log-odds matrix* is the final result. In Table 14.5.5, we show the *PAM* 250 *scoring matrix*. This is the 250th power of PAM1 then converted by log-odds.

Table 14.5.5. PAM250 scoring matrix.

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	13	5	9	9	4	12	6	8	7	6	7	9	11	8	6	11	11	9	2	4
C	2	52	1	1	1	2	2	2	1	1	1	1	2	1	1	3	2	2	1	4
D	5	1	11	10	1	5	6	3	5	2	3	8	4	7	4	5	5	3	1	2
E	5	1	11	12	1	5	6	3	5	2	3	7	4	9	4	5	5	3	1	2
F	2	1	1	1	32	1	3	5	1	6	4	2	1	1	1	2	2	3	4	20
G	12	4	10	9	3	27	5	5	6	4	5	10	8	7	5	11	9	7	2	3
H	2	2	4	4	2	2	15	2	3	2	2	5	3	7	5	3	2	2	2	3
I	3	2	2	2	5	2	2	10	2	6	6	2	2	2	2	3	4	9	1	3
K	6	2	8	8	2	5	8	5	24	4	9	10	6	10	18	8	8	5	4	3
L	6	2	3	4	13	3	5	15	4	34	20	4	5	6	4	4	6	13	6	7
M	1	0	1	1	2	1	1	2	2	3	6	1	1	1	1	1	1	2	1	1
N	4	2	7	6	2	4	6	3	5	2	3	6	4	5	4	5	4	3	2	3
P	7	3	4	4	2	5	5	3	4	3	3	5	20	5	5	6	5	4	1	2
Q	3	1	6	7	1	3	7	2	5	3	3	5	4	10	5	3	3	3	1	2
R	3	2	3	3	1	2	6	3	9	2	4	4	4	5	17	4	3	2	7	2
S	9	7	7	7	3	9	6	5	7	4	5	8	9	6	6	10	9	6	4	4
T	8	4	6	5	3	6	4	6	6	4	5	6	6	5	5	8	11	6	2	3
V	7	4	4	4	10	4	5	4	10	15	4	4	5	4	4	5	5	17	72	4
W	0	0	0	0	1	0	1	0	0	1	0	0	0	0	2	1	0	0	55	1
Y	1	3	1	1	5	1	3	2	1	2	2	2	1	1	1	2	2	2	3	31

14.6 BLAST for Protein and DNA Search

One of the major changes in biology wrought by genomics is that a great deal will be on the computer and especially on the Internet. In this section, we will learn about some of the resources available via the Internet and one of the more important tools in this new biology, biopolymer database searches.

Base pair matching is a key to understanding the genome.

In the previous section, we saw that far-reaching benefits are possible if only we can decipher the secrets of the information represented by the sequence of base pairs within the genome. There are many questions: Where are the genes located in the genome? what signals their beginning and their end? what is the purpose, if any, of nucleotide segments between genes? how does the cell find the genes appropriate to itself among the thousands on its chromosomes? what initiates the transcription of a gene? how is the base pair reading oriented and what strand is used? among others.

Answering these questions begins with simple observations. In any perusal of the human genome, one notices long stretches in which the base pair sequence is simplistic, for example, a short nucleotide sequence, often one or two base pairs in length, repeated hundreds of times. It is obvious that these segments do not code for protein. Thus in the search of gene coding segments, these regions can be dismissed.

Some of these noncoding segments occur between known coding segments; thus segments that code for a particular protein are not contiguous. The intervening noncoding segments are called *introns*, whereas the coding segments are *exons*.

Help can come from studying the genomes of simple organisms such as bacteria, in which it is much easier to identify gene coding segments. Bacterial DNA as a rule do not have intron segments. From these studies, typical patterns emerge. For example, the probabilities of all the possible two-segment sequences, AA, AT, AC, AG, and so on through GG, are not equal in gene coding regions. This information is used to find genomic segments more likely to be associated with a gene.

Another major strategy is the exploitation of sequences composing messenger RNA (mRNA). These biopolymers embody the exact base sequences for encoding their target protein and have no introns. Of course, uracil, U, must be mapped to thymine, T, before a DNA search is performed. The complication here is finding the segments of the mRNA that correspond to the exons of the genome.

Further help can come from a kind of "reverse engineering"; given the sequence of amino acids of a known protein such as myoglobin, this is translated into the codon sequence that produces it. Of course, this will not be unique in general due to codon wobble. Then these several possibilities can be searched for among the genome. This technique has the same difficulty as the mRNA approach in that the exons are not obvious in the engineered sequences.

It becomes clear that a tool to search for nucleotide patterns and flag matches is quite essential, and indeed, central to studying any genome. Constructing such a tool is a challenging, and interesting, mathematical problem in bioinformatics. Increasing the difficulty is the fact that matches may not be exact or contiguous. There are three types of "errors" or complications that must be accommodated by any matching algorithm: Base pair substitution, for example, an A in the subject is for some reason a T in the query (as in sickle-cell anemia); one or more nucleotides may be missing in the query with respect to the subject; and the dual of that, one or more may be inserted in the query (or, equivalently, missing in the subject). In addition, any search algorithm must be able to handle nucleotide sequences that run into the millions or even hundreds of millions.

Fortunately, this problem is largely solved, and today there are very good search algorithms for both global and local matching for both nucleotide and amino acid sequences. We will look at the basics of these algorithms later, but now we want to investigate the immensely useful ability to do DNA and protein searches via the Internet. A few years ago, this kind of tool together with its universal availability was only a fantastic dream.

Biomolecular databases are accessible worldwide.

In addition to the software for biopolymer matching, the sequences to search, or databases, are equally important. Through largely governmentally funded programs (for instance, in universities), many databases are in the public domain and available on the Internet. The human genome is one of those, as are the genomes of many other organisms. These databases are accessible through *GenBank*.

The GenBank sequence database is an annotated collection of all publicly available nucleotide sequences and their protein translations. This database is produced at the *National Center for Biotechnology Information* (NCBI) as part of an international collaboration with the *European Molecular Biology Laboratory* (EMBL), the Data Library from the European Bioinformatics Institute (EBI), and the *DNA Data Bank of Japan* (DDBJ). The consortium is known as the *International Nucleotide Sequence Database Collaboration* (INSDC); their website is

http://www.insdc.org/.

GenBank and its collaborators receive sequences produced in laboratories throughout the world for more than 100,000 distinct organisms. GenBank continues to grow at an exponential rate, doubling every 10 months. Release 169, produced in December 2008, contained over 10^{11} nucleotide bases in approximately 10^8 sequences.

Direct submissions are made to GenBank using *BankIt*, which is a Web-based form, or the stand-alone submission program *Sequin*. Upon receipt of a sequence submission, the GenBank staff assigns an *accession number*, a unique identifier, to the sequence and performs quality assurance checks. A GenBank accession number begins with a stable project ID of one to four letters assigned to each sequencing project. This is followed by six to eight digits. In the eight-digit format, the first digits following the ID are the version number of the sequence: for example, the initial accession number might be AAAX00000000; an update of the sequence would result in a nonzero version number such as AAAX01000000. In the six-digit format, version numbers are formed by appending a dot followed by the version number, for example, AF287139.1.

A master record for each assembly is created. This master record contains information that is common among all records of the sequencing project, such as the biological source, submitter, and publication information.

The submissions are then released to the public database, where the entries are retrievable by *Entrez* or downloadable by FTP. Bulk submissions may be in the form of *expressed sequence tag* (EST), *sequence tagged site* (STS), *genome survey sequence* (GSS), or *high-throughput genome sequence* (HTGS) data. The GenBank direct submissions group also processes complete microbial genome sequences.

BLAST searches are at your Internet fingertips.

Besides maintaining the DNA and protein databases, NCBI also provides tools for searching the databases. Historically, the problem of searching for a specific sequence of symbols that might be contained in a vast store of symbols was first undertaken by informaticians (computer scientists) in connection with finding words and phrases in documents and, later, libraries of documents. This work had an immediate adaptation to bioinformatic searches. The much-refined software is known generically as BLAST, which stands for *Basic Local Alignment and Search Tool*. We will learn about the basic mathematics of BLAST below; for now, we just want to illustrate a BLAST search.

There are five BLAST programs: BLASTN, BLASTP, BLASTX, TBLASTN, and TBLASTX:

- BLASTN, also called *nucleotide blast*, compares nucleotide sequences between the query and the subject..
- BLASTP does the same thing for amino acid sequences between proteins.
- BLASTX compares a nucleotide query with a protein subject. The query is first transformed into an amino acid sequence by grouping its bases into codons (threes) and mapping the codons to amino acids.
- TBLASTN and TBLASTX are both searches in which the query is a protein and the DNA subject is translated into an amino acid sequence.

We will illustrate the basic ideas by doing a BLASTN search. The *subject* of the search will be the entire DNA database at NCBI. We don't have to input that, although we have the option of limiting the subject as desired. A *query* can be input in any of three ways. A short sequence can be typed or pasted in directly. A drawback here is this is feasible only for short sequences. In addition, the sequence will not have an identifier except possibly in your own records. A second possibility is to copy in a file having a standard format known as FASTA. We will look at FASTA files below. The third choice, and the one we will take here, is to enter a sequence already known to the NCBI database, for example by its accession number.

In reviewing the literature on malaria—NCBI also archives journal articles (see http://www.ncbi.nlm.nih.gov/projects/Malaria/)—one might encounter a sequence of interest along with its accession number, for example, NM_004437. This human erythrocyte membrane protein appears in an article on *P. falciparum*; let's investigate it.

A BLAST search is initiated from the webpage:

http://www.ncbi.nlm.nih.gov/BLAST.[2]

[2] Webpage layouts and interfaces change from time to time to accommodate new capabilities and user experience. However, the goal of the website generally remains the same. The specific click sequences given here may have to be modified to get the desired information.

Under `Basic BLAST`, click on `nucleotide blast`; this brings up the NCBI search form. There are two parts: The upper form is for specifying the query, the database, and a rough trichotomy about match lengths for which to optimize. The lower form is for specifying search parameters and filters. These include the expect threshold, seed lengths, match and mismatch scores, and gap costs. Since we will be discussing these parameters in the next sections, accept the defaults for now.

In the large box under "Enter accession number…'' in the first section of the form, enter NM_004437. Under "Program Selection,'' the default optimization is for "highly similar sequences (megablast),'' as indicated by a filled radio button; this is what we want for this example.

Now click BLAST. Initially, the reference ID or RID number will appear, for example, M3JP4BJ001R (possibly very briefly). It can be used to retrieve the results at any time over the next 24 hours.

Right away or in a few seconds, depending on the load at NCBI, the results will appear. (It should not take more than two minutes. If there is a problem, an error message will eventually appear but this could be after an hour of trying.)

BLAST results contain a wealth of information.

At the top of the results page of the search some general information is given about the search itself such as a reminder of the RID number and the version of BLAST used in the search. The query identity is also repeated along with its description; notice that this segment is known by other names—in particular, it has a gi designation. We will discuss these in the next section.

Next comes a "Graphic Summary'' of the results showing the highest-scoring matches, one match per line. The most identical regions are displayed in red. More detail about these and all the "hits,'' that is, the distinct subject sequences that matched the query at or above the requested threshold level (the default level in this case), is given later on in the "Alignments'' section of the results page. The number of hits is also given here if it is less than the "Hitlist size,'' that is, the maximum number to display (100 by default).

The graphic summary is followed by a description of the hits producing significant alignments. Included here are hyperlinks to one or more databases giving more information about the subject.

Click `Help` at the very top of the results page for help with making and interpreting BLAST queries.

Score and expect rate the match.

Lastly, the "Alignments'' section lists all the hits, sorted in the order of the best first. The identity of the sequence is given, its biological description, its match score and expect value, and Web links to information about the sequence. This is followed by a pictorial comparison, base by base, between the query and match. The serial location of each base is given by the numbers at the beginning and end of each line in both sequences, and a vertical bar is used to indicate a match or not at each location. In this

way, matched bases and gaps are clearly indicated. The report also gives the strand, plus or minus, of DNA that was matched.

Upon constructing an alignment between two sequences of residues, what does it mean? Is it significant or just an expected outcome given so large a database? The quality of an alignment must be put on a numerical basis. Otherwise, the whole business devolves into emotional, contentious subjectivity and not science.

The answer lies in the Karlin–Altschul statistics developed expressly to interpret the alignment results. As we have already seen, the output of a BLAST search includes, most importantly, the score and the *expect value* of the match. Roughly, the expect value quantifies the degree to which one would expect the reported alignment by chance alone. Therefore, a small expect value is better.

The Karlin–Altschul parameters themselves are also available in the report. In small type just above "Graphic Summary,'' there is an easily overlooked section entitled "Other reports.'' Click on "Search Summary'' to reveal the alignment parameters used in the search and the important Karlin–Altschul database parameters lambda, K, and H. We will say more about this when we discuss the underlying mathematics.

Two excellent places to get started in learning about the resources available at NCBI are the "Tools'' page,

http://www.ncbi.nlm.nih.gov/Tools/index.html,

and the site map,

http://www.ncbi.nlm.nih.gov/Sitemap/index.html.

FASTA.

In 1985, David Lipman and William Pearson wrote a program, FASTP, for protein sequence similarity searching; the name stands for "FAST protein (matching).'' To simplify the algorithm, the 20 amino acids are given a single-letter designation. This mapping is given in Table 14.6.1 (and Table 14.5.1 in the previous section). Note there are three special characters, X, $*$, and $-$, as given in the table.

Later, FASTP was extended to do nucleotide–nucleotide and translated protein–nucleotide searches as well. Since it could now do All the searches, the name became FASTA. This program eventually lead to BLAST.

In entering protein (or DNA) sequences for FASTA searches, it was realized that each sequence needs an identifier. Thus was invented the *FASTA format*, which has become universal and formalized. A computer file in the FASTA format is a *FASTA file*.

A sequence in FASTA format begins with a definition line followed by lines of sequence data. The definition "line'' has four parts:

(1) a beginning greater-than sign ($>$),
(2) followed immediately (no spaces) by the identifier;
(3) this is followed by a textual description of the sequence (containing at least one space), and
(4) an end-of-line character.

Table 14.6.1. FASTP mapping.

A alanine	P proline
B aspartate or asparagine	Q glutamine
C cysteine	R arginine
D aspartate	S serine
E glutamate	T threonine
F phenylalanine	U selenocysteine
G glycine	V valine
H histidine	W tryptophan
I isoleucine	Y tyrosine
K lysine	Z glutamate or glutamine
L leucine	X any
M methionine	$*$ translation stop
N asparagine	$-$ gap of indeterminate length

The identifier can contain letters or numbers or other typographic symbols, such as pipe ("|"), but no spaces. The "line" is terminated by an end-of-line character (usually the computer carriage return, `cr`, but it could be new line, `nl`, or both). Therefore, the FASTA line could extend over multiple display lines (just keep typing without entering a return). The identifier part is distinguished from the description by a space after the identifier or the line is terminated with no description by an end-of-line. The sequence itself then follows.

The sequence lines can be of any length but usually they are 50 to 80 characters long. An example sequence in FASTA format is the following:

```
>gi|5524211|gb|AAD44166.1|cytochrome b[Elephas maximus maximus]

LCLYTHIGRNIYYGSYLYSETWNTGIMLLLITMATAFMGYVLPWGQMSF
WGATVITNLFSAIPYIGTNLVEWIWGGFSVDKATLNRFFAFHFILPFTM
VALAGVHLTFLHETGSNNPLGLTSDSDKIPFHPYYTIKDFLGLLILILL
LLLLALLSPDMLGDPDNHMPADPLNTPLHIKPEWYFLFAYAILRSVPNK
LGGVLALFLSIVILGLMPFLHTSKHRSMMLRPLSQALFWTLTMDLLTLT
WIGSQPVEYPYTIIGQMASILYFSIILAFLPIAGXIENY
```

The identifier here is actually a concatenation of identifiers in different database sources; pipe is used as the concatenation symbol. We will encounter this again shortly.

The protein sequence characters were given above. For DNA sequences, use the standard IUB/IUPAC nucleic acid codes as in Table 14.6.2.

Note that there are several types of wild cards: R denotes a purine base, Y a pyrimidine one, K a keto base, M an amino one, and so on as in the table. Also note that U is acceptable in a DNA sequence. A single dash (or hypen) can represent a gap of indeterminate length. Lowercase letters are accepted in the sequence; they will be mapped into uppercase.

Table 14.6.2. IUB/IUPAC nucleic acid codes.

A ⟶ adenosine	M ⟶ A C (amino)
C ⟶ cytidine	S ⟶ G C (strong)
G ⟶ guanine	W ⟶ A T (weak)
T ⟶ thymidine	B ⟶ G T C
U ⟶ uridine	D ⟶ G A T
R ⟶ G A (purine)	H ⟶ A C T
Y ⟶ T C (pyrimidine)	V ⟶ G C A
K ⟶ G T (keto)	N ⟶ A G C T (any)
	− gap of indeterminate length

Databases contain the sequence data.

There are several databases of biopolymers. Some of these are given in Table 14.6.3.

Table 14.6.3. Some databases of biopolymers.

Database	Identifier format
DDBJ	dbj|accession|locus
EMBL	emb|accession|ID
NCBI GenBank	gb|accession|locus
NCBI GenInfo	gi|integer
NCBI Reference Sequence	ref|accession|locus
NBRF Protein Information Resource	pir|entry
Protein Research Foundation	prf|name
SWISS_PROT	sp|accession|entry
Brookhaven Protein Data Bank	pdb|entry|chain
Patents	pat|country|number
GenInfo Backbone ID	bbs|number
Local	lcl|identifier
General	gnl|database|identifier

Each database has its own style of identifier as given in the table. An NCBI identifier can belong to the gb, gi, or ref series. As in the example above, the GenInfo identifier is gi|5524211, the GenBank identifier of the same protein is gb|AAD44166.1|, and the description is cytochrome b [Elephas maximus maximus], the organism's biological name being contained in square brackets. The GenBank locus is blank in this example (nothing follows the final pipe).

If you want to identify your own sequences, the local and general categories are for this purpose. As you can see, the general category allows for a more hierarchical structure.

The NCBI Reference Sequence database is further broken down with a two-letter code signifying the type of data or its source. This is given in Table 14.6.4.

Table 14.6.4. Two-letter codes for the NCBI Reference Sequence database.

Prefix	Type/Source
NC_	DNA/genomic
NG_	DNA/human,mouse
NM_	AA/mRNA
NR_	AA/protein
NT_	DNA/assembled
NW_	DNA/whole genome shotgun (WGS)
XM_	AA/human mRNA
XR_	AA/human mRNA

The databases used for BLAST searches are available directly for access or downloading from the site ftp://ftp.ncbi.nih.gov/blast/db/. A particularly important one is *nr*, the nonredundant protein database. This is the database to use for a comprehensive search against all known proteins.

While FASTA format is fine for doing database searches, its description attribute must necessarily be too brief for general purposes. The *flat file* format allows for more documentation. The example below shows just the beginning of a flat file; the entire record continues beyond the ellipses (three dots) indicated at the bottom of the record. It contains the complete, and lengthy, nucleotide coding sequence.

Much of the record is self-explanatory. The locus is yet another identifier of the protein; the first three characters usually designate the organism. The nucleotide sequence for this protein is 5,028 base pairs long. The abbreviation PLN signifies that this organism is a plant, fungus, or alga. The date 21-JUN-1999 is the date of last modification of the record. The /translation entry is the amino acid translation corresponding to the nucleotide coding sequence (CDS). In many cases, the translations are conceptual.

```
     LOCUS  SCU49845 5028 bp DNA PLN 21-JUN-1999
DEFINITION  Saccharomyces cerevisiae TCP1-beta gene, partial cds, and Axl2p
            (AXL2) and Rev7p (REV7) genes, complete cds.
ACCESSION   U49845
   VERSION  U49845.1 GI:1293613
KEYWORDS    .
    SOURCE  Saccharomyces cerevisiae (baker's yeast)
  ORGANISM  Saccharomyces cerevisiae
            Eukaryota; Fungi; Ascomycota; Saccharomycotina; Saccharomyce-
            tes; Saccharomycetales; Saccharomycetaceae; Saccharomyces.
REFERENCE   1 (bases 1 to 5028)
    AUTHORS Torpey,L.E., Gibbs,P.E., Nelson,J. and Lawrence,C.W.
      TITLE Cloning and sequence of REV7, a gene whose function is required
            for DNA damage-induced mutagenesis in Saccharomyces cerevisia
    JOURNAL Yeast 10 (11), 1503-1509 (1994)
     PUBMED 7871890
```

```
REFERENCE  2 (bases 1 to 5028)
  AUTHORS  Roemer,T., Madden,K., Chang,J. and Snyder,M.
    TITLE  Selection of axial growth sites in yeast requires Axl2p, a novel plasma
           membrane glycoprotein
  JOURNAL  Genes Dev. 10 (7), 777-793 (1996)
   PUBMED  8846915
REFERENCE  3 (bases 1 to 5028)
  AUTHORS  Roemer,T.
    TITLE  Direct Submission
  JOURNAL  Submitted (22-FEB-1996) Terry Roemer, Biology, Yale University,
           New Haven, CT, USA
 FEATURES  Location/Qualifiers
     source  1..5028
             /organism="Saccharomyces cerevisiae"
             /db_xref="taxon:4932"
             /chromosome="IX"
             /map="9"
        CDS  <1..206
             /codon_start=3
             /product="TCP1-beta"
             /protein_id="AAA98665.1"
             /db_xref="GI:1293614"
             /translation="SSIYNGISTSGLDLNNGTIADMRQLGIVESYKLK
                         RAVVSSASEAAEVLLRVDNIIRARPRTANRQHM"
     gene  687..3158
             /gene="AXL2"
        CDS  687..3158
             /gene="AXL2"
             /note="plasma membrane glycoprotein"
             /codon_start=1
             /function="required for axial budding pattern of S. cerevisiae"
             /product="Axl2p"
             /protein_id="AAA98666.1"
             /db_xref="GI:1293615"
             /translation="MTQLQISLLLTATISLLHLVVATPYEAYPIG
             ...
```

14.7 The Mathematical Underpinnings of BLAST

As you might imagine, searching for approximate alignments between a query of arbitrary length within a database consisting of sequence data on the order of a trillion (10^{12}) characters is a monumental task. How can it be done so rapidly? In this section, we show how such searches are performed.

Finding alignments is only part of the problem; the other part is making objective, scientifically meaningful assertions about the matches discovered, assertions that must be independent of any unwitting correlations inherent in the database. This is the problem solved by the Karlin–Altschul statistics.

The BLAST program is the main tool for biopolymer searches.

Let's consider how to optimally align two protein sequences. Protein sequences are more of a challenge (there are more symbols) and the same ideas used here work just as well for DNA sequences.

A *global alignment* does the best job at matching the entire lengths of both sequences. For example, suppose the query and subject are

query: CIMGAPART
subject: LIDAFEGAMPAT;

a global alignment is

```
CI---MGA-PART
LIDAFEGAMPA-T.
```

A *local alignment* instead finds the best match between subsequences of the query and the subject. With the same query and subject as above, a local alignment is

```
GA-PART
GAMPA-T;
```

the residues in parentheses are left unaligned,

```
(CIM)GA-PART
(LIDAFE)GAMPA-T.
```

Saul Needleman and Christian Wunsch found an algorithm for performing a global alignment in 1970 [4]. Their algorithm is an adaptation of *dynamic programming* to the problem of protein alignment. Dynamic programming is a very general programming technique for solving large problems that can be structured into a succession of stages such that

- the initial stage of solving certain subproblems is tractable;
- partial solutions to each later stage can be calculated by recursion on a fixed number of partial solutions to earlier stages;
- the final stage contains the overall solution.

To grade the quality of an alignment, the *Needleman–Wunsch algorithm* allows you to assign a value to matches, mismatches, and gaps. For example, assign $+3$ to matches, -1 to mismatches, and -2 to gaps. Once an alignment has been forged,

its overall score is the sum of the values of each aligned pair. Thus in the global alignment above, the score is

$$-1+3-2-2-2-1+3+3-2+3+3-2+3=6.$$

Because the Needleman–Wunsch algorithm essentially computes all possible alignments, it is guaranteed to be optimal in the sense of its score. To see how the algorithm works, refer to Figure 14.7.1.

		L	I	D	A	F	E	G	A	M	P	A	T
	0	−2	−4	−6	−8	−10	−12	−14	−16	−18	−20	−22	−24
C	−2	−1	−3	−5	−7	−9	−11	−13	−15	−17	−19	−21	−23
I	−4	−3	2	0	−2	−4	−6	−8	−10	−12	−14	−16	−18
M	−6	−5	0	1	−1	−3	−5	−7	−9	−7	−9	−11	−13
G	−8	−7	−2	−1	0	−2	−4	−2	−4	−6	−8	−10	−12
A	−10	−9	−4	−3	2	0	−2	−4	1	−1	−3	−5	−7
P	−12	−11	−6	−5	0	1	−1	−3	−1	0	2	0	−2
A	−14	−13	−8	−7	−2	−1	0	−2	0	−2	0	5	3
R	−16	−15	−10	−9	−4	−3	−2	−1	−2	−1	−2	3	4
T	−18	−17	−12	−11	−6	−5	−4	−3	−2	−3	−2	1	6

Fig. 14.7.1.

The symbols of the subject have been written across the top and the symbols of the query along the side. The numbers in the body of the table are the accumulating scores for the various alignment possibilities. The first number in the table is 0 and signifies the start of the alignment. Moving to the right in the table signifies matching the subject residue at the head of the column with a gap, denoted by a dash. Likewise, moving down in the table signifies matching the query residue along the side with a dash (gap).

For example, to the right of 0 is −2 because the alignment of L in the subject with dash in the query,

```
L
-,
```

values this pair at −2. Moving cell by cell to the right adds another letter vs. gap to the alignment and hence adds another −2 to the score. This completes the first row. The first column of numbers is the same except symbols of the query are now matched to gaps in the subject. The −4 next to the I is the score for the alignment

```
--
CI.
```

Moving diagonally one right and one down signifies matching the subject letter at the top with the query letter along the side. Thus the −1 in row C, column L

corresponds to mismatching L in the subject to C in the query, a value of -1. This is added to the 0 to get the running score to that point.

There are several paths to most entries in the table. For example, the -3 in row C, column I could be reached from 0 by going two cells to the right and one down. The corresponding alignment for this is

```
--C
LI-.
```

The score for this path is that of three gaps, or -6. Alternatively, the same point in the table can be reached from 0 by a diagonal move followed by a move to the right. This gives the alignment

```
C-
LI.
```

The score for this path is -3, for a mismatch and a gap, and that is what is written in the table. The score at any place in the table is always the maximum score over all paths to that point. In the case of ties, the alternative paths signify alternative alignments that are equally good as far as the score is concerned.

Note that every possible path is, at the same time, represented in the table.

Now that we see how the table is made, how is the optimal alignment constructed from it? We will construct the alignment in reverse. Start at the bottom right of the table and work backward to 0, either up, or left, or diagonally up and left at each step, reversing the process of calculating the entry. Thus the bottom right entry for our example problem is 6, and that is the score of the complete alignment. The value to the left is 1, but a horizontal move right from that cell corresponds to matching a gap and gives a $1 - 2$ or -1, not 6; so that can't be it. Try the diagonal value 3 up and left from 6. The move from 3 diagonally to the 6 corresponds to matching the query symbol T with the subject symbol T and so adds $+3$ to the score. This does give 6, so this is the right step backward.

Actually this *traceback*, as it is called, can be made completely simple, a matter of following arrows, if one adds a small additional step to the forward calculation. Whenever a new score is added to a cell during the forward calculation, that is, choosing the maximum of a move down, a move right, or diagonally down and right, also add an arrow showing where the entry came from: a left arrow for a step right, an up arrow for a step down, and a northwest arrow for a diagonal step. In Figure 14.7.2, we show these arrows for every cell. Tracing back gives the global alignment above.

Local alignment goes by a similar but slightly modified algorithm.

Several years passed until finally Temple Smith and Michael Waterman developed a local alignment algorithm in 1981 [6]. Actually, the *Smith–Waterman algorithm* is just a slight modification of the Needleman–Wunsch algorithm. Namely, the cell value is never allowed to be less than zero; if it falls below, just write 0. In this case, no backward pointer is recorded.

		L	I	D	A	F	E	G	A	M	P	A	T
	0	−2 ←	−4 ←	−6 ←	−8 ←	−10 ←	−12 ←	−14 ←	−16 ←	−18 ←	−20 ←	−22 ←	−24 ←
C	−2 ↑	−1 ↖	−3 ↖	−5 ↖	−7 ↖	−9 ↖	−11 ↖	−13 ↖	−15 ↖	−17 ↖	−19 ↖	−21 ↖	−23 ↖
I	−4 ↑	−3 ↖	2 ↖	0 ←	−2 ←	−4 ←	−6 ←	−8 ←	−10 ←	−12 ←	−14 ←	−16 ←	−18 ←
M	−6 ↑	−5 ↖	0 ↑	1 ↖	−1 ↖	−3 ↖	−5 ↖	−7 ↖	−9 ↖	−7 ↖	−9 ←	−11 ←	−13 ←
G	−8 ↑	−7 ↖	−2 ↑	−1 ↖	0 ↖	−2 ↖	−4 ↖	−2 ↖	−4 ←	−6 ←	−8 ↖	−10 ↖	−12 ↖
A	−10 ↑	−9 ↖	−4 ↑	−3 ↖	2 ↖	0 ←	−2 ←	−4 ↑	1 ↖	−1 ←	−3 ←	−5 ↖	−7 ←
P	−12 ↑	−11 ↖	−6 ↑	−5 ↖	0 ↑	1 ↖	−1 ↖	−3 ↖	−1 ↑	0 ↖	2 ↖	0 ←	−2 ←
A	−14 ↑	−13 ↖	−8 ↑	−7 ↖	−2 ↖	−1 ↖	0 ↖	−2 ↖	0 ↖	−2 ↖	0 ↑	5 ↖	3 ←
R	−16 ↑	−15 ↖	−10 ↑	−9 ↖	−4 ↑	−3 ↖	−2 ↖	−1 ↖	−2 ↑	−1 ↖	−2 ↑	3 ↑	4 ↖
T	−18 ↑	−17 ↖	−12 ↑	−11 ↖	−6 ↑	−5 ↖	−4 ↖	−3 ↖	−2 ↖	−3 ↖	−2 ↖	1 ↑	6 ↖

Fig. 14.7.2.

		L	I	D	A	F	E	G	A	M	P	A	T
	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	3 ↖	1 ←	0	0	0	0	0	0	0	0	0
M	0	0	1 ↑	2 ↖	0	0	0	0	0	3 ↖	1 ←	0	0
G	0	0	0	0	1 ↖	0	0	3 ↖	1 ←	1 ↑	2 ↖	0	0
A	0	0	0	0	3 ↖	1 ←	0	1 ↑	6 ↖	4 ←	2 ←	5 ↖	3 ←
P	0	0	0	0	1 ↑	2 ↖	0	0	4 ↑	5 ↖	7 ↖	5 ←	4 ↖
A	0	0	0	0	3 ↖	1 ←	1 ↖	0	3 ↖	3 ↑	5 ↑	10 ↖	8 ←
R	0	0	0	0	1 ↑	2 ↖	0	0	1 ↑	2 ↖	3 ↑	8 ↑	9 ↖
T	0	0	0	0	0	0	1 ↖	0	0	0	1 ↑	6 ↑	11 ↖

Fig. 14.7.3.

In Figure 14.7.3, we show the resulting table along with all backarrows. This time one starts with the maximum value in the table and traces back until one encounters no backarrow.

We get

```
GA-PART
GAMPA-T.
```

BLAST is faster.

While the algorithms detailed above align optimally, they are not fast enough for really big databases and require too much computer memory. BLAST is quite different and makes use of the BLOSUM62[3] or PAM250 matrix we constructed previously to do the scoring. Again to illustrate the ideas, assume that we are going to align our query protein with a protein database.

For the first step our protein sequence is broken into overlapping groups of three consecutive symbols called *words* or 3-mers:

CIMGAPART
CIM
IMG
MGA
GAP
APA
PAR
ART

Take the first word, CIM, and score it with the BLOSUM62 matrix (see Figure 14.7.4)—C is 9, I is 4, and M is 5—for a total of 18. If CIM in the query matches CIM in the subject, this contribution to the overall score is 18. But what if CIM aligned with CLM? Since an I replacing L is a frequent occurrence, that combination will also give a high score. So along with CIM, we make a list of its modifications, called *neighbors*, which give a high score, say, greater than or equal to $T = 14$; T is called the *threshold*.

In this scheme, since C and I by themselves give 13, M can be replaced by anything giving a 1 or greater; that would be I, L, or V. The other single replacements are similarly examined. Note that C cannot be replaced by anything and still achieve the threshold. In this example, it is also possible to replace two originals; I can be replaced by V and M by L to give an acceptable modification.

The list is

$$\begin{aligned} &\text{CIM} && 9+4+5=18,\\ &\text{CII} && 9+4+1=14,\\ &\text{CIL} && 9+4+2=15,\\ &\text{CIV} && 9+4+1=14, \end{aligned}$$

[3] Another popular scoring matrix.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z	X	∗
A	**4**	−1	−2	−2	0	−1	−1	0	−2	−1	−1	−1	−1	−2	−1	1	0	−3	−2	0	−2	−1	0	−4
R	−1	**5**	0	−2	−3	1	0	−2	0	−3	−2	2	−1	−3	−2	−1	−1	−3	−2	−3	−1	0	−1	−4
N	−2	0	**6**	1	−3	0	0	0	1	−3	−3	0	−2	−3	−2	1	0	−4	−2	−3	3	0	−1	−4
D	−2	−2	1	**6**	−3	0	2	−1	−1	−3	−4	−1	−3	−3	−1	0	−1	−4	−3	−3	4	1	−1	−4
C	0	−3	−3	−3	**9**	−3	−4	−3	−3	−1	−1	−3	−1	−2	−3	−1	−1	−2	−2	−1	−3	−3	−2	−4
Q	−1	1	0	0	−3	**5**	2	−2	0	−3	−2	1	0	−3	−1	0	−1	−2	−1	−2	0	3	−1	−4
E	−1	0	0	2	−4	2	**5**	−2	0	−3	−3	1	−2	−3	−1	0	−1	−3	−2	−2	1	4	−1	−4
G	0	−2	0	−1	−3	−2	−2	**6**	−2	−4	−4	−2	−3	−3	−2	0	−2	−2	−3	−3	−1	−2	−1	−4
H	−2	0	1	−1	−3	0	0	−2	**8**	−3	−3	−1	−2	−1	−2	−1	−2	−2	2	−3	0	0	−1	−4
I	−1	−3	−3	−3	−1	−3	−3	−4	−3	**4**	2	−3	1	0	−3	−2	−1	−3	−1	3	−3	−3	−1	−4
L	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	**4**	−2	2	0	−3	−2	−1	−2	−1	1	−4	−3	−1	−4
K	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	**5**	−1	−3	−1	0	−1	−3	−2	−2	0	1	−1	−4
M	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	**5**	0	−2	−1	−1	−1	−1	1	−3	−1	−1	−4
F	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	**6**	−4	−2	−2	1	3	−1	−3	−3	−1	−4
P	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	**7**	−1	−1	−4	−3	−2	−2	−1	−2	−4
S	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	**4**	1	−3	−2	−2	0	0	0	−4
T	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	**5**	−2	−2	0	−1	−1	0	−4
W	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	**11**	2	−3	−4	−3	−2	−4
Y	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	**7**	−1	−3	−2	−1	−4
V	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	**4**	−3	−2	−1	−4
B	−2	−1	3	4	−3	0	1	−1	0	−3	−4	0	−3	−3	−2	0	−1	−4	−3	−3	**4**	1	−1	−4
Z	−1	0	0	1	−3	3	4	−2	0	−3	−3	1	−1	−3	−1	0	−1	−3	−2	−2	1	**4**	−1	−4
X	0	−1	−1	−1	−2	−1	−1	−1	−1	−1	−1	−1	−1	−1	−2	0	0	−2	−1	−1	−1	−1	−1	−4
∗	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	1

Fig. 14.7.4. BLOSUM62 matrix.

$$\begin{aligned}
&\text{CLM} \quad 9+2+5=16,\\
&\text{CMM} \quad 9+1+5=15,\\
&\text{CFM} \quad 9+0+5=14,\\
&\text{CVM} \quad 9+3+5=17,\\
&\text{CVL} \quad 9+3+2=14.
\end{aligned}$$

This same calculation of neighbors is done for each of the 3-mers listed above. As a result, we get a fairly long list of what are called *seeds*.

In step 2, for each seed we look for an exact match within the subject sequence. This process goes very fast because the subject has been scanned ahead of time and a lookup table established for all possible 3-mers. Since there are 20 amino acids and three positions, there are exactly $20^3 = 8000$ of them, a workable number. An exact match between 3-mers in the subject and a seed is called a *hit*.

For step 3, each hit is extended in both directions by adding pairs of residues, one from the query aligned with one from the subject. The scoring matrix is updated as each pair is added until the score falls below a certain preassigned value. The *dropoff value*, denoted by X, controls this cutoff. When the score of a given branch drops below the current best score minus the X-dropoff, the exploration of this branch stops. The resulting aligned segment is called a *high-scoring segment pair*, or HSP.

We will use our continuing example to demonstrate step 3 of the method. (The other steps have already been illustrated.) Because the example is small, we will take word size to be 2. This, in fact, is the word size used by the FASTA algorithm, which preceded BLAST by two years. Also, we take the dropoff value to be 4. Among the seeds produced in step 1 is GA from residue 4 in the query; this will produce a hit in the subject at residue 7 with a result score of $6 + 4 = 10$. Working upstream one residue, we match M in the query with E in the subject, a mismatch and therefore from the BLOSUM62 matrix a penalty of 2. Since the dropoff limit is 4, we continue matching I in the query with F in the subject, again a mismatch with penalty of 0. Continuing upstream, the next lineup is C with A, again a mismatch penalty of 0. Since we are at the end of the query string, we stop matching and trim back to the maximum value encountered, which was 10 with the original GA.

Now work downstream, matching P in the query with M in the subject. This gives a penalty of 2. Next match A with P, penalty of 1; then R with T, penalty of 1, so stop. Trim back to the maximum value and arrive at the HSP of GA. The original BLAST did not allow for gaps.

Improvements to BLAST allow for gaps.

Improvements to BLAST occur frequently. It was improved by its creators themselves in 1997 to gapped BLAST. Unfortunately, we must ask the reader to research gapped BLAST due to the constraints of the scope of this text.

Karlin–Altschul statistics are independent of scoring matrix.

Attached to each alignment is its score in bits and its expect value. We mentioned earlier that expect refers to the likelihood that the score occurs by chance, so small expect values are better. Here we discuss the basis on which score and expect are calculated.

The mathematical theory underlying the statistics of an alignment was worked out by Samuel Karlin and Stephen Altschul in 1990. It draws on several branches of mathematics such as the theory of information, extreme value theory, and *Poisson probability distributions*. A technical discussion is beyond the scope of this text, but we hope to give an overview sufficient for understanding the issues.

As mentioned above, clicking on "Search Summary" in the NCBI report provides additional numerical information pertaining to the run. The information there includes the size of the database and query, the "effective" lengths of the subject and query (accounting for end effects), the number of database hits, and so on. Our focus here is on the parameters lambda, K, and H. Lambda, or λ, and K are the *Karlin–Altschul parameters* and H is the *relative entropy*. These parameters are concerned with the calculation of the bit scores and expect values and are central to the statistical underpinnings of the report.

Certainly, it is necessary that bit scores and expect values have universal meaning; in particular, the results cannot be dependent on the scoring matrix. BLOSUM62 is just one among many possible scoring matrices, and individual researchers are free to use their own. The statistical results must compensate for whatever matrix is

used. This is the function of the Karlin–Altschul parameters. Lambda functions as a normalization factor and K compensates for interresidue dependencies that may be built in to the matrix, that is, the lack of probabilistic independence between the residues.

All scoring matrices, S_{ij}, have built within them an implicit probability distribution, q_{ij}, that residue j can substitute for residue i (and conversely). For example, we saw in the previous section how the PAM1 matrix was constructed from the Dayhoff data derived from empirical substitution probabilities. The implicit probabilities can be recovered from the scoring matrix using the defining equation for lambda,

$$\lambda S_{ij} = \log\left(\frac{q_{ij}}{p_i p_j}\right), \tag{14.7.1}$$

where the p_i are the *background probabilities*. For the ith residue, p_i is the frequency of its occurrence over a large number of protein sequences. Equivalently, it is the probability that it will appear at any particular position in a randomly constructed sequence. These probabilities are assumed to be independent of each other.

Solving for q_{ij}, we get

$$q_{ij} = p_i p_j e^{\lambda S_{ij}}, \tag{14.7.2}$$

provided λ is known. But since the q_{ij} are probabilities, their sum over the entire scoring matrix must be 1; hence we have

$$\sum_{i,j} p_i p_j e^{\lambda S_{ij}} = 1. \tag{14.7.3}$$

Since all the variables in this equation are known except λ, it can be used to find λ.

Finding K is harder and is estimated by statistical inference methods. For most of the scoring matrices in common use, K is on the order of 0.1.

With λ and K in hand, the *Karlin–Altschul equation* is used to find expect,

$$E = Kmne^{\lambda S}. \tag{14.7.4}$$

In this, m is the length of the subject sequence and n is the length of the query sequence. The equation gives the expected number of HSPs in an alignment that will have a score of at least S just by chance. The equation makes intuitive sense because doubling the length of either sequence should double the *number* of HSPs attaining a given score. On the other hand, in order to double a given *score*, an HSP must attain that score twice in a row, so E should decrease exponentially with score, and so it does. As we have seen, small values of E are highly significant.

In reality, the Karlin–Altschul equation is an approximation. Attempts to improve its accuracy center on making adjustments for the values of m and n and for the presence of gaps. Using the actual lengths m and n of the sequences is not completely accurate because of end effects; for example, the middle of the query cannot be matched with the first few residues of the subject, since the initial part of the query would then align with nothing. So there are techniques for figuring *effective lengths*.

With respect to gaps, the derivations culminating in Karlin–Altschul statistics are theoretically exact for gapless alignments. There is no exact theory when gaps are included as a possibility. As a result, various empirical techniques have been derived to adjust for this.

Returning to the output parameters, we turn our attention to relative entropy H. It is defined by

$$H = -\sum_{i,j} q_{ij} \lambda S_{ij}.$$

Thus H is the negative of the expected value of the normalized score. This will always be a positive value because the expected score for aligning a random pair of amino acids must necessarily be negative. Were this not the case, long alignments would tend to have high score independently of whether the aligned segments were related, and the statistical theory would break down.

As its name implies, H is also an entropy in the information theory sense, and information theory is the basis for Karlin–Altschul statistics. Its interpretation with respect to a scoring matrix is that low-entropy matrices are more specialized (i.e., have more structure) and high-entropy matrices are more general.

Bit scores compensate for the scoring system used.

Once an alignment has been constructed, the raw scores of its HSPs are available. But these scores are not useful by themselves since they depend on the scoring matrix. However, once λ and K are known, the raw score can be adjusted to calculate a *bit score*, denoted by S'. The bit score is given by the equation

$$S' = \frac{\lambda S - \log K}{\log 2} \tag{14.7.5}$$

and is comparable to bit scores calculated by other scoring matrices.

By combining (14.7.5) with (14.7.4), expect can be calculated in terms of bit score as

$$E = mne^{-\lambda S + \log K} = mne^{-S' \log 2},$$
$$E = mn2^{-S'}.$$

Thus bit scores embody the statistical essence of the scoring system employed. From the bit score, one only needs to know the size of the search space in order to calculate the expect.

P-values derive from the Poission distribution.

The number of random HSPs with score $\geq S$ is described by a Poisson distribution. This means that the probability of finding no HSPs with score $\geq S$ is e^{-E}, so the probability of finding at least one such HSP is $P = 1 - e^{-E}$.

This is the P-value associated with the score S. The BLAST programs report E-values rather than P-values because it is easier to understand the difference between, for example, E-values of 5 and 10 than P-values of 0.993 and 0.99995. However, when $E < 0.01$, P-values and E-values are nearly identical.

Exercises/Experiments

General notes. The layout of websites and the location of links change from time to time. You may find that the specific link following sequences below are not completely accurate. Please try to work your way to the destination link overcoming any such changes. These exercises are intended to get you started in the use of Internet resources in the new era of genomics biology. We will barely scratch the surface of what's available on the Web.

1. In this exercise, we want to study the map of a specific chromosome of some organism. Go to NCBI, click "Map Viewer" under "Hot Spots" (also accessible via Entrez and the nucleotide database). In the search box at the top, pick an organism (e.g., *Plasmodium falciparum*), click "Go!," and click on a chromosome (e.g., chromosome 3). Explain what information is presented on this page directly and through its links.

2. One may research proteins as well. From NCBI, click "Entrez Home"; click "Protein: sequence database"; under "Additional protein information," click "structure"; in the "Search Entrez Structure/MMDB" box, enter the protein to be studied, (e.g., tryrosine kinase); click "go"; click on the desired type of resource (e.g., 2COI for Src Family Kinase…). Explain what information is available on this page and through the links on the page. A downloadable helper application for browsers, Cn3D, is available for viewing the three-dimensional structure of the protein (click "Structure" in the toolbar across the top).

3. As noted in the chapter, *orthologous* genes between two or more species are those with a high degree of sequence similarity and usually code for proteins having similar function. They are assumed to have evolved from a common ancestral gene. The proteins coded for by such genes are also called orthologous. By comparing the genomes of the species sequenced up to the present time, several groups or clusters of orthologous proteins have been discovered. These clusters of orthologous groups are known as COGs for prokaryotic proteins and KOGs for eukaryotic ones. From the NCBI webpage, click "Clusters of orthologous groups" under "Hot Spots," click Eukaryotic Clusters, then click 16 in the first line under KOGs; finally, click KOG1019, Retinoblastoma pathway protein. Explain what information is presented on this page and its links. Note that we will discuss phylogeny diagrams in Chapter 15.

4. The objective this time is to use the Entrez reference resources to see what is available on a topic and to use BLAST to research the associated genomics. Feel free to substitute a subject of your own interest for toxoplasmosis in the example here. Use the Entrez database at NCBI (from the NCBI homepage, select "Entrez Home") to search for toxoplasmosis: enter "toxoplasmosis" in the search box and execute the search. There are many possibilities from this point; for example, click "Nucleotide," scroll down, and select "BD495032." After reading about this nucleotide sequence, do a blastn search on it as described in the chapter. Experiment with the parameters of the search, especially the expect, word size,

and database. Report on the information available and the differences you found by varying the parameters.

5. This exerise is similar to the one above, except this time do a tblastn search. Research your own example or take the following. In Exercise 3 above, we came across a reference to "At5g27610" (note it is a "gee" after 5, not a nine) in connection with KOGs. Enter this into an Entrez search and look under "proteins." One of these is NP_198113. Do a tblastn search on this protein and experiment with the search parameters. In the results page, just below the "Distribution of…BLAST hits," there is a list of "sequences producing significant alignments" along with their E-values. To the right of that are boxes labeled "U" or "E" or "G." What information is available on these links?

Questions for Thought and Discussion

What changes in biology have been brought about by genomics (knowing the complete genome of organisms) in the fields of

1. genetics;
2. taxonomy (classifying organisms);
3. evolution;
4. enzyme activity?

References and Suggested Further Reading

[1] P. Benfey and A. Protopapas, *Genomics*, Prentice–Hall, Englewood Cliffs, NJ, 2004.
[2] D. T. Jones, W. R. Taylor, and J. M. Thornton, The rapid generation of mutation data matrices from protein sequences, *Comput. Appl. Biosci.*, **8**-3 (1992), 275–282.
[3] I. Korf, M. Yandell, and J. Bedell, *Blast*, O'Reilly, Cambridge, UK, 2003.
[4] S. B. Needleman and C. D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, *J. Molecular Biol.*, **48**-3 (1970), 443–453.
[5] G. Smith, *Genomics Age*, AMACOM, New York, 2005.
[6] T. F. Smith and M. S. Waterman, Identification of common molecular subsequences, *J. Molecular Biol.*, **147**-1 (1981), 195–197.
[7] E. Ukkonen, Algorithms for string matching, *Inform. Control*, **64** (1985), 100–118.