Classification of Protein Domain Families
for
Genomic Sequence Analysis

Erik Leonard Laage Sonnhammer

Thesis presented for the degree of

Doctor of Philosophy

The Sanger Centre
Hinxton, Cambridgeshire

September 16, 1996

CONTENTS

ABSTRACT

Large scale genome sequencing projects have created a demand for more efficient sequence analysis methods. For a daily output in the order of 10^5 finished basepairs per lab, many traditional sequence analysis methods are too labour intense to be practical. For a genome of a higher eukaryote such as *Caenorhabditis elegans*, some of the most challenging analysis tasks are correctly predicting the exons and introns of genes, and the functional annotation of the proteins they encode. This thesis describes new methods that have been developed to assist both these tasks, in an integrated way. Gene structure prediction and functional classification are linked activities in the sense that quality improvement of one often leads to improvement of the other. The major issue for doing both on a large scale is how to improve efficiency without sacrificing quality.

Presented in part 1 are a set of software tools, that integrated in the ACEDB genomic database form a graphical analysis workbench well suited for high throughput and high quality genomic sequence analysis. They are based on single query sequence database searching and were implemented as algorithmic improvements and interactive graphical visualisation tools for results from the BLAST programs. A novel type of dot-plot program was incorporated for sensitive pairwise sequence comparison, and a database retrieval tool was developed to connect to external databases.

Part 2 describes a different line of approach, based on multiple sequence alignments for database searching. Multiple alignment based methods are often more sensitive than single sequence methods, and further automate the annotation process. In order to use such methods efficiently, a comprehensive high-quality collection of protein domain families described by multiple alignments is needed. Methods were developed to build and maintain such a database, based on hidden Markov model profiles and seed alignments. This resulted in the database **Pfam**, which currently contains 175 of the largest protein families. The high-quality alignments in Pfam are released together with automatically generated families of the remaining sequences. A graphical workbench for display of Pfam search results was developed and integrated into ACEDB. Analysis of protein domain families in all predicted *C. elegans* proteins was carried out by using Pfam and by a study of nematode-specific families.

1. Introduction

This thesis concerns bioinformatics applied to genome sequencing. These are both very young research fields, and are introduced separately below. Bioinformatics is a too wide-ranging discipline to be fully covered in detail here. The introduction therefore concentrates on the aspects that at present are most relevant for genomic sequence analysis: gene prediction and sequence similarity analysis. Because the second part of this thesis entirely focuses on protein families, a separate introduction is provided of this field.

1.1 Genome sequencing projects

One of the most important steps towards fully understanding the biology of an organism is the determination of its entire genome sequence. Thanks to recent improvements in DNA sequencing technology, the sequencing of entire genomes has become feasible. The genomes of a number of model organisms, ranging from simple bacteria to invertebrate animals, have been or are currently being sequenced, leading up to the challenge of completing the human genome early next century.

The availability of complete genome sequences will dramatically change the way that molecular biology research is done. The classical 'forward genetics' approach to find a gene is a top-down approach, using the phenotype of mutant individuals to progressively narrow down its location in the genome via genetic and physical mapping techniques. Eventually a clone of the region that contains the gene is sequenced. This procedure is very labour intensive, and relies on good genome maps. With the entire genome sequence known, many shortcuts can be taken which will accelerate traditional 'interest-driven' research. In model organisms it is also potentially possible to turn this process around: going from a known gene sequence to its function and impact on the phenotype by 'knocking it out' [Plasterk, 1992; Giese *et al.*, 1992; Johnston, 1996; Spradling *et al.*, 1995]. This 'reverse genetics' approach will initially be done on an *ad hoc* basis for particularly interesting genes, see e.g. [Zwaal *et*

al., 1993], but hopes are high that a systematic knock-out analysis of all genes with unknown function in yeast will result in a wealth of new biological knowledge [Oliver, 1996]. We cannot yet predict exactly what we will learn from the genome sequence, but it is already clear that it will be a major resource for biological research, both by direct analysis of the sequence as well as a reference for laboratory experiments.

Prokaryotic and single-cell eukaryotic genomes are more attractive for whole-genome sequencing than those of higher multi-cellular eukaryotes. The reasons are that they tend to be small (0.5 - 15 million basepairs (Mb) compared to 100 - 3000 Mb) and that the protein coding regions tend to be densely packed single open reading frames instead of dispersed exon/intron structures, thus rendering gene prediction relatively straightforward. In fact, less than 5% of the human genome codes for proteins. Because of this, most human genome sequencing projects have so far concentrated on the parts of the genome that are expressed into proteins. This is done by making libraries of cDNA clones from mRNA in cells of different tissues. For efficiency reasons, usually only 300-500 basepairs at the 5' and 3' end of these clones are sequenced in a single read, called an EST (expressed sequence tag). Although this data is enriched in protein coding sequences and has tissue-specific information attached, EST data is fragmentary and a large fraction is of poor quality. It should therefore mainly be seen as valuable complementary data to the complete genome sequence. In fact, they are invaluable for finding protein coding regions in the genome sequence, as will be discussed in section 1.2.

The complete genome sequence of four free-living organisms has to date been determined: the bacteria *Hemophilus influenzae* Rd [Fleischmann *et al.,* 1995] and *Mycoplasma genitalium* [Fraser *et al.,* 1995], the yeast *Saccaromyces cerevisiae* [Dujon, 1996] and the archeon *Methanococcus jannaschii* [Bult *et al.,* 1996]. A number of other microbial genomes are projected to be completed within two years: The two thermophilic archaeons and *Pyrococcus furiosus* [Weiss, 1996], *Mycoplasma pneumoniae* [Hilbert *et al.,* 1995], *Esherichia coli* [Wahl *et al.,* 1994], *Bacillus Subtilis* [Medigue *et al.,* 1995], a protozoan that causes typhus *Rickettsia prowazekii* [Andersson *et al.,* 1995], the spirochete that causes lyme disease *Borrelia burgdorferi* [Dunn, 1996], *Mycobacterium tuberculosis*, *Methanobacterium thermo-*

autotrophicum and *Synechocystis sp.* [Smith *et al.,* 1995]. From the first compete genome sequences, it has emerged that while Bacteria and Archea are metabolically similar, the archeal gene expression systems has much more in common with Eukaryota.

The methods described in this thesis are generally applicable to any genome. However, since they were developed in collaboration with the sequencing projects for *C. elegans* and *H. sapiens*, the two main projects at the Sanger Centre, a more detailed description of these genome projects follows.

Caenorhabditis elegans

This free-living soil nematode has been the subject of intense molecular biology research ever since Sydney Brenner's (1974) classic screen for mutants. *C. elegans* is an attractive model organism due to its completely known cell development from egg to adult hermaphrodite (959 cells) or male (1031 cells), its suitability for genetic experiments (see review [Hodgkin *et al.,* 1995]) and to its excellent genetic [Edgley and Riddle, 1990] and physical [Coulson, 1994] maps.

The systematic sequencing of its 100 Mb genome [Wilson *et al.,* 1994] (appendix A) has been underway since 1992 in two laboratories, the Sanger Centre in Cambridge, UK and the Genome Sequencing Center in St. Louis, USA, and is estimated to be finished in 1998 [Waterston and Sulston, 1995]. At present about 50 Mb from the most gene-dense regions have been completely finished, comprising more than half of all genes, while approximately 20 Mb are in various stages of completion.

Currently around 30000 *C. elegans* EST sequences [Waterston *et al.*, 1992; McCombie *et al.,* 1992; Y. Kohara, personal communication] are available, from approximately 5000 genes. About a third of all predicted genes are associated with one or more ESTs. Normally only a small part of the 5' and 3' ends are covered by EST matches, but some genes are completely covered. Such cases are very useful for calibrating the gene prediction methods. ESTs have also in some cases provided direct evidence of alternative splicing (see figure 2.1).

It has been estimated that *C. elegans* has a total of some 15000 genes [Waterston *et al.*, -1992]. This estimate was calculated by dividing the number of genes predicted in a 200 Kb region by the number of matching EST clones, and multiplying this number with the total number EST clones in the collection. Assuming a representative EST matching frequency, this would predict the total number of genes. Amazingly, this number, which was based on only four EST matches, is very close to the current estimate, which is based on nearly half the genome. The total gene estimate has fluctuated somewhat, mainly due to lower EST matching frequency on chromosome X, which is over-represented in the currently finished sequence. The number may also be overestimated because it is based on predicted genes, some of which are likely to be pseudogenes, but in the absence of conclusive evidence cannot be recognised as such. Over 7000 genes have been predicted so far, including 330 tRNA genes and numerous other structural RNAs.

The genome project is at present mainly used as a resource to aid traditional research. If somebody wants to use the nematode as a model to study the function and effects of a gene that has been discovered in another organism, the traditional cross-species hybridisation methods need no longer be used to identify the *C. elegans* homologue. Instead, a scan through the genomic sequence will rapidly identify all homologues (once the sequence is completed) and reveal their genome locations. Thanks to a frozen library of random Tc1 transposon insertions [Zwaal *et al.*, 1993], the chances are that the gene in question is near a Tc1 element. If this is the case, a deletion derivative can be made to knock the gene out, thus generating a null phenotype mutant.

The sequence itself can also been used to screen for particular gene families. For instance, a number of nematode-specific families of G-protein coupled receptors have been implicated in olfaction. From the genome sequence, a large number of these genes were found by sequence similarity, which proved to be localised to sensory neurons by reporter constructs [Troemel *et al.*, 1995].

A number of other genomic phenomena have been discovered, such as the high abundance of DNA repeat families. Some of these have the characteristics of apparently inactive, partly

degenerated parasitic transposon families. Such elements may have exploited a functional transposon in the past to spread throughout the genome.

Homo sapiens

Because the human genome has such a low density of genes (< 5% coding), the first large-scale human genome sequencing projects have concentrated on sequencing expressed sequence tags (ESTs) [Adams *et al.,* 1995; Hillier *et al.,* 1996]. Currently around 630000 human EST sequences are available in the public database dbEST, but many more ESTs exist [Adams *et al.,* 1995]. The ESTs are also used for mapping purposes by a number of groups that form an international consortium [Boguski and Schuler, 1995].

It has been estimated that the 3000 Mb human genome contains 50000 - 100000 genes [Fields *et al.,* 1994]. The 280000 ESTs sequenced by the Merck-St. Louis project [Hillier *et al.,* 1996] were derived from an estimated 29000 genes. Most other ESTs have only been sequenced at the 5' end, making the number of total genes hard to estimate, but it is clear that EST projects will never be able to find all genes. As more genes are sequenced, the chance of resequencing already known genes grows, and rarely expressed genes become increasingly difficult to sift out from the abundant ones. EST projects were fast and cost-effective in the beginning of the human genome project, but are now drawing to an end.

Emphasis is already shifting towards complete genomic sequencing. Before this can be undertaken, the genome has to be mapped at a sufficiently high resolution. This has progressed well during the past years and many people argue that the time is now right to start sequencing at large scale [Olsen, 1995]. Already some 57 Mb of human sequences are available in the EMBL database, which in contrast to *C. elegans* is more than has been produced by genome sequencing projects. At the Sanger Centre, about 8 Mb of human chromosomes 3, 4, 6, 11, 13 16, 22 and X have been finished at this time.

A substantial amount of support is now becoming available for genomic sequencing. Funding for about 300 Mb in the USA, 50 Mb in Germany and 500 Mb at the Sanger Centre have already been allocated, to be completed within the next 5 years. The amount of data produced will depend on what accuracy is aimed for. At low accuracy, the cost per base be-

comes significantly lower, but the usefulness of the sequence is also reduced due to its uncertainty. However, since the gene density in the human genome is relatively low, a lower quality is perhaps acceptable, at least in the non-coding regions. It has been proposed that lowering the quality from 1 error in 10000 bases, which is the standard quality, to 1 in 1000 might be a good compromise which would make completion of the entire genome affordable and enable it to be finished 5 years earlier than originally projected. The most important regions would be finished at a higher quality, while nonessential 'junk' DNA sequence would be left in a less accurate state [Marshall, 1995]. The feasibility of this approach depends to a large extent on how well analysis methods can be adapted to find coding regions in poor quality sequence data.

Human genomic sequencing is still in a very early phase. As old techniques are refined and new ones are developed, efficiency will improve and the cost decrease. Currently a large proportion of the resources are spent on development. For instance, it has been proposed that instead of sequencing the commonly used 40 Kb insert cosmid clones, the whole genome could be cloned into 20000 350 Kb insert BACs (bacterial artificial chromosomes) [Venter *et al.,* 1996]. By sequencing 500 bases at both ends of each clone, the ordering and selection of clones for complete sequencing would be simplified, and physical mapping would no longer be necessary. The next years will see a convergence of laboratory technologies that can be automated efficiently. With a world wide collaborative effort, the completion of the entire human genome sequence by 2005 seems feasible.

1.2 Genomic sequence analysis

Once the genome sequence has been determined, what can we learn from it? All biological phenomena are ultimately encoded in the DNA sequence, so in the long term we hope that knowing the genome sequence will provide answers to most biological questions. However, in the short term this will not be possible, since our current level of knowledge of this code is insufficient for a precise and detailed understanding of how it results in a living organism.

The basic principles for how strings of DNA encode RNA and protein molecules are known. We can find protein coding regions in DNA to some degree of confidence by looking for signals that are important for the transcriptional and RNA processing machineries, and by examining statistical effects of the usage of the genetic code (see below). Direct evidence of coding regions can also be gathered experimentally, by sequencing transcribed mRNA sequences, as was done in EST projects. However, the gene products interact with other molecules to perform life-supporting functions, and we are not able to predict exactly what these interactions are from the sequence alone.

A tremendous effort has gone into cracking the molecular enigma of protein folding, but the ultimate goal of predicting the three-dimensional structure and the function of a protein from its amino acid sequence alone is not yet in sight. Instead, related but simpler problems have been tackled, such as secondary structure (helix, sheet or coil state) prediction, fold recognition and 3D structure comparison. Genome sequencing projects are in fact helping to improve structure prediction, by rapidly producing more protein sequences. Structure prediction using multiple homologous sequences is superior to single-sequence prediction since evolutionary information contained in the multiple alignment can reveal many structural features, albeit mainly on the secondary structure level. The main hope for predicting folds is pinned on the belief that only a limited number of unique folds exist in nature, and once these structures have been determined experimentally, the structure of new sequences could be predicted by correctly assigning them to one of the known folds. This notion is based on the fact that already with only about 300 known unique fold classes, over 90% of newly solved structures has a previously seen fold class [Holm and Sander, 1996]. This may however be

due to oversampling of certain types of folds due to experimental bias, and that some fold motifs occur at unusually high frequencies. In fact, there is little evidence that the rate of discovery of new unique folds is decreasing.

Even without a detailed molecular understanding of the genomic components, many functional properties can be elucidated. One way is by direct experiment, which will now be performed exhaustively on the genes for which no function is known in the genome of *S. cerevisiae* [Johnston, 1996]. Another way is to compare the sequence of newly found genes with other known sequences of proteins with a known function.

The basis for structural and functional inference from sequence similarity is that evolution proceeds via gene duplication and speciation events, giving rise to families of related proteins that we observe in the present day. Homologous sibling proteins caused by recent gene duplication tend to have related functions, for instance enzymes that perform the same catalytic reaction but on different substrates. Homologues in different organisms may have identical functions, such as catalysis of the same step in a pathway, or may be related via both a gene duplication and speciation events. For both types of homologues, the constraints on the sequence to make a functional protein are such that the common ancestry can be observed in their sequences even after hundreds of millions of years. The sequences will differ much less than the actual mutation rate would imply, because any functionally deleterious mutation will not be propagated. Only slight variations in the sequence are usually tolerated. This is the basis for inferring function and structure from sequence similarity. Whether proteins with similar three-dimensional structures but dissimilar sequences and functions are actually highly diverged homologues or are unrelated, is a matter open to discussion. There exists a 'twilight zone', where a marginal sequence similarity may be meaningful or not. Only when two sequences are sufficiently similar to each other can functional and structural inference be done with confidence.

The first analysis that is done to new protein sequences is therefore a comparison to all other known sequences, to determine whether functional information can be inferred by homology. Traditionally, this has been done by manually perusing the output of various database searches (see below) and further analysing potentially interesting database hits with

various methods such as dot-plots (see chapter 5) and multiple alignments (see below). A researcher would be content to spend several days analysing a sequence that took months or years to determine. The main difference for genome sequencing projects is that so many proteins are found every day that the process has to be made more efficient and robust. For this, special analysis workbenches have been constructed (see below), so that a human expert can process large amounts of sequence in time-spans matching the sequencing rate.

Having access to the entire genome sequence makes certain types of questions answerable. For instance, the number of paralogues in gene families can be assessed, and a new research field of 'comparative genomics' is already emerging. Here questions such as comparing the genomic organisation and content of different species [Tatusov *et al.*, 1996], finding common sets of ancestral genes [Green *et al.*, 1993], and estimating the size of a minimal life-supporting set of genes [Mushegian and Koonin, 1996] are addressed. We can expect to learn a lot from studying homologues in different organisms. Already a number of intriguing cases have been found, such as a nitrogen fixation like gene in *S. cerevisiae* [Dujon, 1996] and homologues to human disease genes in lower organisms [Waterston and Sulston, 1995].

The remainder of this section will focus on the bioinformatics topics that are most relevant to genomic sequence analysis, namely gene prediction, database searching and automated analysis workbenches.

Gene prediction

Predicting protein coding regions in genomic DNA is a species-specific problem. Different organisms use the interchangeable genetic codes to different extents, and higher eukaryotes have their protein coding regions (exons) interrupted by non-coding regions (introns), while prokaryotes and simple eukaryotes do not. All gene finding systems are therefore fine-tuned for a particular organism.

Protein coding DNA differs from non-coding in that it contains triplets (codons), each encoding an amino acid residue. Since there are 64 triplets but only 20 amino acids, several triplets can encode the same amino acid. The frequency of usage of each codon depends on the abundance of the amino acid it encodes, and the choice of codon for a particular amino

acid. Most organisms have a preference for certain codons, which usually reflects differences in the abundance of the different tRNA species that bind to each codon. Highly expressed genes tend to use codons for abundant tRNAs [Bulmer, 1987]. The correlation between expression levels and codon usage is not perfect however, and a host of other reasons are potentially involved in biasing the codon usage (see [von Heijne, 1987] for a review). Table 1.1 shows the codon usage in *E. coli* [Krogh *et al.*, 1994b]. Many codons are as frequent as might be expected by chance, but there are some notable exceptions, e.g. the Arginine codons starting with A are hardly ever used.

Table 1.1. Codon usage frequencies and their expected frequencies in *E. coli*.

Codon	Amino acid	Codon usage	Expected frequency	Codon	Amino acid	Codon Usage	Expected frequency
AAA	Lys	3.5	1.3	GAA	Glu	4.3	1.6
AAG	Lys	1.1	1.6	GAG	Glu	1.8	1.8
AAC	Asn	2.4	1.4	GAC	Asp	2.2	1.7
AAT	Asn	1.4	1.3	GAT	Asp	3.2	1.5
AGA	Arg	0.1	1.6	GGA	Gly	0.6	1.8
AGG	Arg	0.1	1.8	GGG	Gly	1.0	2.2
AGC	Ser	1.6	1.7	GGC	Gly	3.2	2.0
AGT	Ser	0.7	1.5	GGT	Gly	2.8	1.8
ACA	Thr	0.5	1.4	GCA	Ala	2.0	1.7
ACG	Thr	1.4	1.7	GCG	Ala	3.6	2.0
ACC	Thr	2.5	1.5	GCC	Ala	2.5	1.8
ACT	Thr	0.9	1.4	GCT	Ala	1.6	1.6
ATA	Ile	0.3	1.3	GTA	Val	1.1	1.5
ATG	Met	2.5	1.5	GTG	Val	2.7	1.8
ATC	Ile	2.7	1.4	GTC	Val	1.5	1.6
ATT	Ile	2.8	1.3	GTT	Val	1.9	1.5
CAA	Gln	1.3	1.4	TAA	*	*	*
CAG	Gln	3.0	1.7	TAG	*	*	*
CAC	His	1.1	1.5	TAC	Tyr	1.4	1.4
CAT	His	1.2	1.4	TAT	Tyr	1.5	1.3
CGA	Arg	0.3	1.7	TGA	*	*	*
CGG	Arg	0.4	2.0	TGG	Trp	1.4	1.8
CGC	Arg	2.4	1.8	TGC	Cys	0.7	1.6
CGT	Arg	2.5	1.6	TGT	Cys	0.5	1.5
CCA	Pro	0.8	1.5	TCA	Ser	0.6	1.4
CCG	Pro	2.6	1.8	TCG	Ser	0.8	1.6
CCC	Pro	0.4	1.6	TCC	Ser	0.9	1.5
CCT	Pro	0.6	1.5	TCT	Ser	0.9	1.4
CTA	Leu	0.3	1.4	TTA	Leu	1.1	1.3
CTG	Leu	5.7	1.6	TTG	Leu	1.2	1.5
CTC	Leu	1.0	1.5	TTC	Phe	1.8	1.4
CTT	Leu	0.9	1.4	TTT	Phe	1.9	1.2

The non-random behaviour of triplets in coding DNA can be used to recognise it as such [Staden, 1990]. A number of different statistical modelling techniques can be used for this. The perhaps most straightforward way is to directly derive a statistical model from the difference in base frequencies in coding and non-coding DNA, as in the *C. elegans* Genefinder [P. Green, unpublished], where the likelihood of coding is estimated by the logarithmic ratio of the frequency of a triplet in coding vs. non-coding DNA. These scores summed over a win-

dow gives the coding potential. It has proved more discriminating to use di-triplets (6 bases) than single ones. It is also possible to train a neural network to recognise coding regions, either directly from the DNA sequence or as a way to combine a number of precalculated statistical properties in a weighted fashion, as is done in Grail [Uberbacher and Mural, 1991] and GeneParser [Snyder and Stormo, 1995].

A statistical framework that appears well suited for gene prediction is the hidden Markov model (HMM), in which a chain of states model the probabilities of bases in the different codon positions [Krogh *et al.,* 1994b]. GenMark [Borodovsky *et al.,* 1995] achieves improved performance by comparing the coding probabilities of the top and bottom strands, and Genie [Kulp *et al.,* 1996] combines an HMM based system with a neural network that predicts splice sites by a dynamic programming algorithm (see below).

Gene prediction in prokaryotes and lower eukaryotes is often considered trivial, since any long region without stop codons, so-called open reading frames (ORFs), is likely to contain a gene. However, to avoid errors such as accepting spurious long ORFs, missing short true ORFs and using the wrong start codon, complex statistical models are necessary.

Predicting genes with introns requires an extra statistic on splice donor and acceptor sites, which are also somewhat species-specific. The final goal of the gene prediction then becomes finding the parse of exons and introns that optimises the combined likelihood score of coding segments and splices. The length distribution of introns, and a minimal length for exons are normally also taken into account. The most common method to find the optimal exon prediction is dynamic programming (see below), which is used by Genefinder, GeneParser [Snyder and Stormo, 1993] and Grail [Xu *et al.,* 1994].

In *C. elegans,* the length of introns is usually small compared to the length of exons (the most common lengths are 50 and 100 bp respectively), which limits the number of ways potential exons can be combined together, and makes splice prediction relatively easy. Human genes on the other hand, often have introns that are thousands of basepairs long. This makes the ratio of true to false splice sites very unfavourable, and in many cases so many possible parses are equally likely that little confidence can be given to the prediction. A further complication is that multiple parses may occur naturally, so-called ‘alternative splicing’, where a

particular exon may be skipped or truncated. This may either occur at all times, producing two or more versions of a gene product, or be regulated and only occur under certain circumstances, such as in a particular developmental stage. A drawback of dynamic programming is that it normally only reports the optimal parse, while the true parse may score slightly lower. It might therefore be preferable to list more than one plausible parse, but unfortunately the number of possible combinations is often so huge that this becomes impractical.

Prediction of exons in higher eukaryotes can greatly be aided by matching EST sequences from the same organism to the genome sequence. A match provides definite proof of any spliced out introns that are observed as insertions in the alignment. Also, this is the only reliable way of predicting alternative splicing. However, *ab initio* prediction is still necessary, since the EST collection will never contain 100% of all genes, and since they do not usually cover the entire span of the genes.

Upstream of the coding region in a gene lies a control region, called the promoter, where the transcription complex binds to start transcription. Promoter sites are known for many transcription factors, and these can be used for locating promoters in new genes [Prestridge, 1995]. The main problems here are that the sites are often only a few basepairs long and that sequence conservation tends to be rather poor, which gives many false positives. So far, no algorithm has incorporated promoter finding with exon finding.

Many genes code for structural RNA molecules. Since these lack the codon bias of protein genes, there is no obvious statistical means of finding them. However, most tRNA genes can be found by a consensus method [Fichant and Burks, 1991], or by sequence similarity. More sophisticated methods to hunt for RNA genes exploit the fact that RNA molecules fold up by internal basepairing. This has made it amenable to stochastic context-free grammar approaches [Sakakibara *et al.*, 1994], which can be brought under the hidden Markov model framework [Eddy and Durbin, 1994]. In practise this has been shown to be practical only for tRNA (approximately 75 bases) or smaller RNAs.

A theoretically possible way to predict RNA genes would be to apply an RNA folding algorithm [Zuker and Stiegler, 1981], and look for regions of DNA that fold into low energy conformations. However, the reliability of available folding programs is currently not suffi-

cient to distinguish true RNA genes from the background noise. Furthermore, the known algorithms are so computationally intensive that it would not be practical for genomic analysis.

Finally, sequence similarity to homologous genes can be very useful for predicting protein coding genes. By comparing the raw translation of all six DNA frames to known protein sequences, the matching regions can give strong indications of what the correct gene prediction may be. Ignoring this information in a genome project invariably leads to incorrect gene predictions and missed frameshifts [Tatusov *et al.,* 1996]. The first part of this thesis is concerned with using sequence homology to improve both gene prediction and annotation for genome projects.

Database searching

By far the most efficient way to predict the function of a newly found protein is by comparing its sequence to other proteins with known function, and inferring the function from sequence similarity. International collaborations have set up databases that collect and distribute all known nucleotide and protein sequences. The main nucleotide database is EMBL/Genbank [Rodriguez-Tome *et al.,* 1996; Benson *et al.,* 1996], and the main protein databases are Swissprot and its supplement TREMBL [Bairoch and Apweiler, 1996], and PIR-international [George *et al.,* 1996]. Most entries of these databases have functional annotation, but a growing proportion have been sequenced by genome projects and are simply annotated as 'hypothetical proteins'.

Approximately 100000 - 150000 unique protein sequences are known today, and this number is increasing rapidly. Although many database sequences are similar to each other, usually the entire database is searched, or one where only identical or almost identical sequences have been removed. Search speed is therefore of utmost importance, which can be achieved either by heuristic algorithms or special hardware.

Sequence comparison is based on detecting evolutionarily similar sequences. The simplest scoring scheme is to give a positive score for identical residues and a negative score for non-identical residues. For all 20x20 = 400 possible amino acids, this corresponds to an

identity matrix. This scoring system only works well for closely related proteins. For distantly related proteins, it becomes important to weight similar amino acids higher than dissimilar ones. The first scoring scheme where the scores were derived from statistical analysis of related sequences was the PAM series of score matrices [Dayhoff *et al.*, 1978]. This was based on a matrix for closely related sequences that could be scaled for other evolutionary distances. The drawback of this system was that the extrapolation to longer distances, although theoretically justified, did not correspond to biological reality. The BLOSUM series of score matrices [Henikoff and Henikoff, 1992], which were derived directly from observed distant relationships are currently most widely used.

Given two sequences and a scoring scheme, an algorithm is needed to find the optimal alignment. The main challenge here is that because related protein sequences not only have diverged in amino acid types, but also in the number of residues, they must be aligned with 'padding' in the deleted or inserted regions. A suitable algorithm for this is dynamic programming [Needleman and Wunsch, 1970; Smith and Waterman, 1981] (reviewed by [Kruskal, 1983]). Dynamic programming works by filling in a matrix of additive maximal scores at all residue pair positions, allowing for gaps, and tracing back the path of the highest score. This path gives the optimal alignment, given the two sequences, the scoring scheme and the chosen gap penalties. If the scoring scheme and gap penalties are chosen carefully, dynamic programming can be more sensitive than any other method. However, these parameters can not be generalised to any type of comparison, and the wrong choice of parameters can make it perform poorly. Nevertheless, it has become very popular due to the fact that it produces gapped alignments.

In dynamic programming, constraints on the beginning and end of the alignment have a great influence on the result. It can either be global, i.e. be forced to start at the first and stop at the last residue in both sequences, or be local, i.e. the start and stop may occur anywhere. The global algorithm can be more sensitive when appropriate, but the local algorithm is more reliable and is used most often. Various improvements have been made to the original algorithm. Many of these are implementation details that make it less computationally intensive. It is possible to reduce both its space [Hirschberg, 1975] and time [Crook, 1991; Barton,

1993b] requirements. Other additions to the algorithm have permitted reporting of multiple non-overlapping [Waterman and Eggert, 1987] or overlapping [Barton, 1993b] locally optimal alignments. Improvements in speed have also been achieved by parallel implementations on DAP [Collins *et al.*, 1988], MasPar and Bioccellerator computers.

Without hardware acceleration, the dynamic programming algorithm is too time-consuming for routinely scanning large databases. Database searching can be made faster by, instead of aligning all database sequences to the query, applying a heuristic to select the sequences most likely to have a high-scoring match, and only aligning these sequences. This can be done by hashing, as is done in FastA [Pearson, 1990], or by a deterministic finite automaton to find high-scoring word matches, as is done in BLAST [Altschul *et al.*, 1990]. These algorithms are so efficient that they can be used routinely for database searches on normal workstations. FastA applies a dynamic programming step at the end, while BLAST reports ungapped matches. This is sometimes heavily criticised, but on the other hand nearly all alignment information comes from the matching segments and not from the gaps. Another reason that ungapped matches were chosen is that a rigorous statistical theory could be developed [Karlin and Altschul, 1990], using a random model. As will be discussed in chapter 4, although BLAST is rather well suited for genomic analysis, many aspects could be improved upon.

Statistical significance has been the subject of much interest in database searching. After a search, the scores of spurious matches to unrelated sequences tend to be randomly distributed around some positive value, while true matches to related sequences ideally are separated from the tail of the noise. The background noise curve can be fitted to a distribution function in order to estimate the statistical significance of the matches. This can be done regardless of the algorithm used, and has been done also for dynamic programming [Collins and Coulson, 1990]. However, the shape of the distribution of scores is not known *a priori*, so it was often approximated with the normal distribution. This has recently proved significantly less accurate than the extreme value distribution (see introduction of chapter 4). Independent of the method used to estimate the statistical significance, in most cases there is a

'twilight zone' of marginally scoring sequences that may be distant relatives or spurious matches.

There are some special requirements for database searching for genomic analysis. It is preferable to search with the raw DNA sequence as query instead of the predicted genes, especially for genomes with intron-rich genes. The search program must be capable of reporting matches to widely spaced exons without a too strong penalty for the intervening introns. It must also be fast enough to cope with a throughput of 10^5 basepairs per day. The Blastx program satisfies both of these conditions. Up to now, no dynamic programming algorithm has satisfied either of these requirements, unless special hardware is used. This may change in the future, since dynamic programming algorithms are under development that either exploit a frameshift allowance for introns [Birney *et al.*, 1996], or explicitly takes splicing into account, by not penalising gaps starting with GT (splice donor) and ending with AT (splice acceptor) [X. Huang, personal communication].

Automated and integrated analysis workbenches

Most large-scale sequencing projects are undertaken by research centres specialised in DNA sequencing that have resources to streamline the process and attain high throughput at low cost. It is not uncommon to have a daily output in the order of 10^5 basepairs. To match the speed of sequencing, a new breed of more efficient analysis methods has become necessary. The two main routes to increased efficiency are (1) automation of routine tasks and (2) an efficient analysis environment for human computer interaction. At this moment a fully automatic expert system that performs as well as a human does not seem feasible. The reason is that a human expert uses a very broad spectrum of biological knowledge and combines different pieces of evidence in an intuitive way of reasoning that is hard to express as rules that could readily be adapted to a computer program. Instead, the efficiency and quality of the analysis can be improved by providing a human analyst with a powerful set of tools in an integrated workbench environment. These tools are intended to allow the expert to quickly access and analyse information that visualises the most relevant data but hides irrelevant data to prevent exhaustion.

Genomic sequence analysis can be performed at either the DNA or protein level. For prokaryotic and simple eukaryotic genomes, where most genes correspond to single ORFs, performing the analysis on the protein level is more or less acceptable. For higher eukaryotic genomes, where genes consist of many short exons embedded in introns, gene prediction is less trivial. Because of the inaccuracy of such gene predictions, the homology analysis of the encoded proteins is more reliable when performed directly on the DNA sequence.

A system that has concentrated on ORF analysis of smaller genomes is GeneQuiz [Scharf *et al.*, 1994], which successfully has been applied to the *Hemophilus influenzae* [Casari *et al.*, 1995], *Mycoplasma capricolum* [Bork *et al.*, 1995], *Mycoplasma genitalium* [Ouzounis *et al.*, 1996] and *Saccaromyces cerevisiae* [Casari *et al.*, 1996]. For DNA analysis, a number of graphical tools for visualisation of sequence similarity and other features exist, such as ChromoScope [Zhang *et al.*, 1994], the BDGP java viewer [Rubin, 1996], WebEntrez [Kuzio, 1996] and APIC [Bisson and Garreau, 1995].

At the Sanger Centre, the analysis environment is built around the genomic database ACEDB [Durbin and Mieg, 1996]. ACEDB was initially developed for storage and distribution of genomic data such as genetic and physical mapping data, strains, authors and phenotypical information. With the arrival of the large-scale sequencing of the nematode *C. elegans*, ACEDB was extended to incorporate a sequence analysis workbench. This workbench was extended with tools for more detailed similarity analysis, which are described in this thesis.

Figure 1.1 shows schematically the different steps of the analysis process, where ACEDB forms the central foundation. The sequencing groups normally finish the sequence of an entire cosmid (40000 bp) before it is submitted to analysis. First, a number of analysis programs are run in batch mode, such as gene finding and database searching. The output of these programs is then reformatted and read in to ACEDB. At this point, an analyst will look through the gene predictions and consolidate them using the similarity to other sequences. The underlined components in figure 1.1 were developed as part of this thesis (see thesis outline below). If strong similarity is found that supports a different splicing pattern, the gene prediction can be edited manually. Once this type of analysis has been done on the

DNA level, a more sensitive analysis can be performed on the translated gene predictions. After the analysis has been done, the cosmid sequence and the attached annotations are submitted to the EMBL data library.

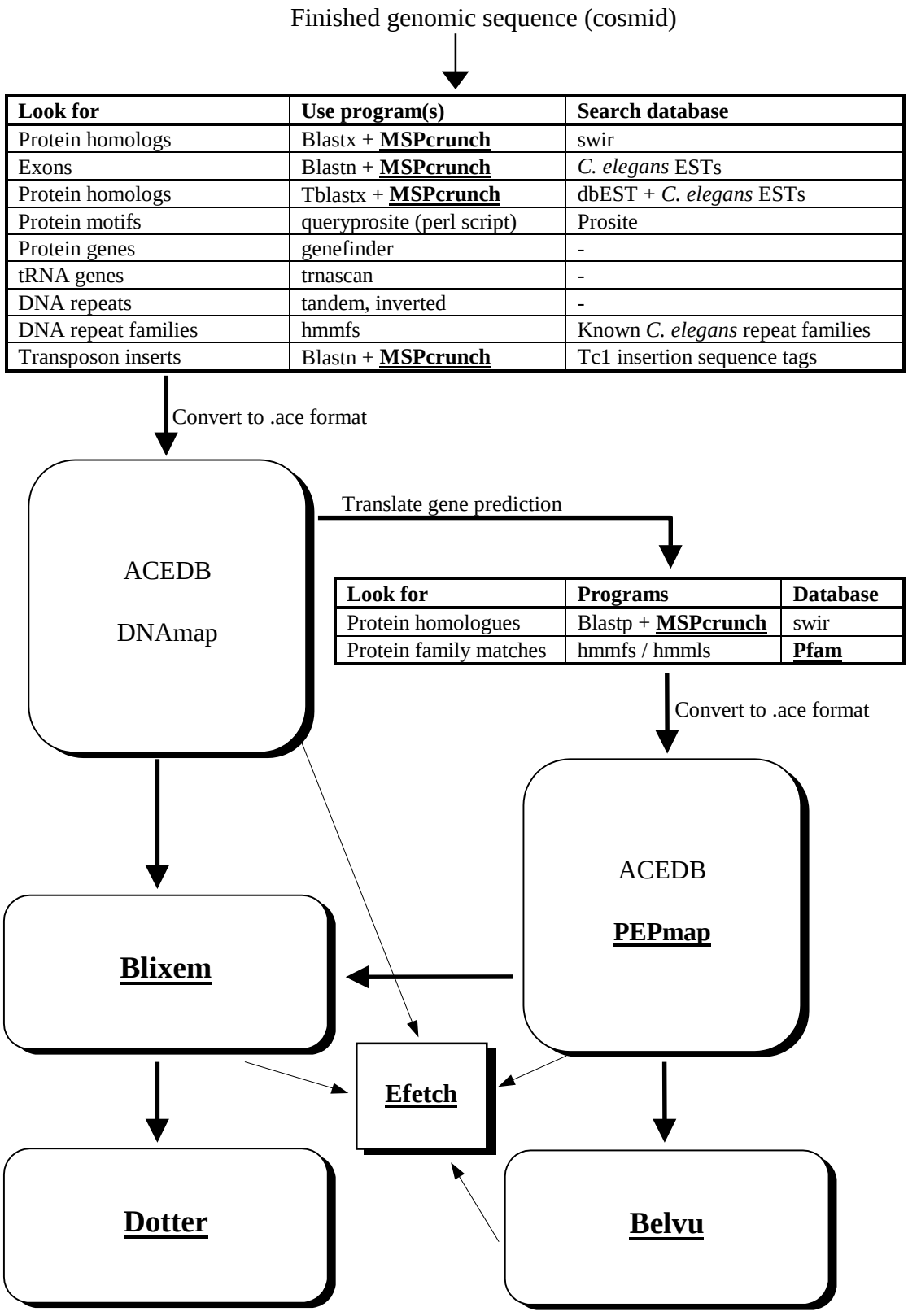


Figure 1.1. Component overview of the workbench for genomic sequence analysis built around ACEDB. The search methods in the tables are specific for *C. elegans*. Boxes with rounded corners are interactive visualisation tools. The underlined components are described in this thesis.

1.3 Protein Families

The method described above for genomic sequence analysis is essentially an extension and streamlining of traditional methods. It is clearly a significant improvement over manual methods, and it works well enough to analyse and annotate genomic DNA at sufficiently high rates and quality. The main drawbacks are: (1) The annotation is still subject to the imagination of the annotator, who also has to spend much time reading annotations of database hits to form an opinion on which domain(s) the query possesses. This is compounded for proteins containing multiple or repeated domains. (2) Sensitive analysis methods are only used as a second step, after the database search.

These problems can be mitigated by instead of searching a database of single sequences, searching a database of multiple alignments of protein domain families. A family membership thus found provides unambiguous domain annotation, and since the multiple alignment contains evolutionary information, the method can more easily discriminate between true and false members.

There are several ways to exploit the information in multiple alignments. One way is to convert it to a score matrix based profile [Gribskov *et al.*, 1987]. Here each column is compressed into a vector of 20 scores, one for each amino acid. The scores are derived from the observed amino acid frequencies and a score matrix, such as BLOSUM. Another way is to bring it under the framework of hidden Markov models [Krogh *et al.*, 1994a], in which each column is represented by a state which has a probability to produce the amino acids derived from the observed frequencies. Insertions and deletions are represented by separate states, and positions where insertion and deletions are likely to occur are reflected by high probabilities to enter these states. HMM-based profiles have some advantages over score matrix based profiles, but need many examples to train on. The two models are however similar enough to be interchangeable, and it is possible to combine them in a ‘hybrid’ model, in which the score matrix behaviour dominates at small sample sizes but the probabilistic behaviour gradually takes over with more examples (see chapter 7 for more details).

The methods for comparing a sequence to a profile are normally based on dynamic programming, usually with an extension that allows multiple non-overlapping matches in the query sequence.

Alignment databases

The second line of approach in this thesis is to exploit family based techniques to genomic sequence analysis. For this, a comprehensive collection of multiple alignments is needed. The most comprehensive protein family database is Prosite [Bairoch *et al.,* 1996], but it is only based on small motifs, or patterns. Other databases are available that contain multiple sequence alignments of the Prosite families [Gribskov *et al.,* 1988; Attwood *et al.,* 1996; Pietrokovski *et al.,* 1996], but these alignment are still only of the most conserved regions and do not span the entire protein domains. For mere family membership identification this is not a problem, but for a detailed domainwise analysis, whole-domain alignments are necessary.

Available databases such as PIRALN [George *et al.,* 1996] and ProDom [Sonnhammer and Kahn, 1994], do contain full-domain alignments, but these families tend to contain only very closely related sequences and are unlikely to be more sensitive than pairwise comparison. Therefore, it was decided to create a comprehensive database of whole-domain alignments, which contain as much evolutionary information as possible. A system for construction and maintenance of such a database, based on HMM/score matrix hybrid profiles, was developed. The resulting database is called Pfam and is described in chapters 7 and 8.

Multiple sequence alignment construction

Extending the dynamic programming algorithm to more than two sequences has a computational complexity in the order of L^N, where L is the average length, and N is the number of sequences. This makes it prohibitively time and space consuming for more than three sequences. Techniques that limit the search space can be applied to reduce the computational cost [Carrillo and Lipman, 1988; Altschul *et al.,* 1989], but this method is still only practical for a few sequences.

Another method is to look for motifs in common to all sequences and use these as anchor points to construct either a partial or complete alignment [Bacon and Anderson, 1986; Johnson and Doolittle, 1986; Waterman and Jones, 1990; Schuler *et al.,* 1991; Smith and Smith, 1992; Depiereux and Feytmans, 1992; Posfai *et al.,* 1994; Smith *et al.,* 1990]. These programs are usually best suited to find short motifs, and are often not practical for large sets of sequences.

A more efficient method is based on constructing a tree hierarchy of all sequences to be aligned, and then progressively aligning sequence pairs, starting from the closest sequences. Aligned pairs are merged into an averaged sequence, containing gap residues, which is treated as a single sequence in subsequent pairwise alignments. All sequences have a mapping to the final alignment at the root of the tree, and this gives the complete multiple alignment. Several implementations of this algorithm exist [Barton and Sternberg, 1987; Feng and Doolittle, 1987; Taylor, 1988; Higgins *et al.,* 1992]. A review [McClure *et al.,* 1994] comparing the quality of these methods concluded that they all to various extents make similar mistakes. One program may be more sensitive to inclusion of rogue sequences in the dataset, while another one may have more problems merging subsets correctly. No single program stands out as being better than the others.

A further method is based on training a hidden Markov model on unaligned sequences, which afterwards can be aligned to the model, thus generating a multiple alignment. This method was shown to be approximately as accurate as progressive pairwise methods [Eddy, 1995b], but is much slower.

Multiple sequence alignments can also be constructed from superposition of three-dimensional protein structures [Sutcliffe *et al.,* 1987; Sali and Blundell, 1990; Russel and Barton, 1992]. This method is only applicable to proteins with a known or modelled 3D structure, which is a minority of all protein families. Furthermore, although the alignment corresponds closely to the structural alignment, it is not clear whether this always corresponds to an evolutionarily correct alignment.

Sequence weighting

When generating the model from the multiple alignment, care must be taken that the alignment is representative of all members. If a group of closely related members are overrepresented, the model will be biased towards that group and may not be sensitive to underrepresented members. This can be compensated for by either removing closely related sequences from the alignment or by a weighting scheme, which downweights overrepresented sequences. The latter method is better for preserving information about the sequence variation, but on the other hand very little information is lost by removing near-identical sequences.

Various methods to calculate sequence weights are available [Sibbald and Argos, 1990; Thompson *et al.*, 1994; Gerstein *et al.*, 1994; Eddy *et al.*, 1995; Gotoh, 1995]. Using a weighting scheme often leads to substantial improvements in sensitivity for searching. The difference in sensitivity between various weighting schemes is usually negligible, so aspects such as speed of computation and robustness are of more concern. Some methods are very vulnerable to inclusion of false members [Krogh and Mitchison, 1995] and should be used with utmost care. A robust and fast method that was developed as part of this thesis is described in appendix B.

1.4 Thesis outline

This thesis is divided into two parts because two main lines of methodology have been pursued. The first part contains chapters that describe components that have been developed as parts of a sequence analysis environment for genomic sequence analysis, based on pairwise comparison algorithms. The second part is concerned with family-based comparisons, and treats different aspects of the Pfam database of protein families, which was developed with genomic analysis in mind. As seen in figure 1.1, components from part 1 and 2 are actually integrated in the same workbench.

Chapter 2 introduces the main methodology and programming environment of the ACEDB database and graphics library, which are the foundation onto which all the graphical analysis tools are integrated. Chapter 3 presents Blixem, a graphical viewer for multiple se-

quence alignments constructed from BLAST database search results, which have been processed by a program MSPcrunch, described in chapter 4. MSPcrunch uses empirical rules to filter out irrelevant BLAST matches, at the same time as it make the search more sensitive.

For detailed pairwise similarity analysis, Blixem has been integrated with a new type of dotplot program, Dotter, presented in Chapter 5, which has a new way of dynamically setting stringency thresholds. This, and its relatively fast operation makes Dotter also well-suited for comparing cosmid-size DNA sequences. Chapter 6 describes a general purpose database retrieval tool, Efetch, which is used by the other programs in the workbench. A picture of these tools is given in figure 1.2

Part 2 describes family-based analysis methods. Chapter 7 presents the database of protein families that this work is based on, Pfam, and explains why and how it was constructed. Chapter 8 then explains how Pfam can be used for genomic sequence analysis, and describes some graphical tools to assist this, which are depicted in figure 1.3. In chapter 9, an analysis of the proteins that have been discovered by the *C. elegans* genome project is presented. The analysis focuses mainly on protein families, and was done both by exploiting Pfam and by a clustering procedure to examine nematode-specific families.

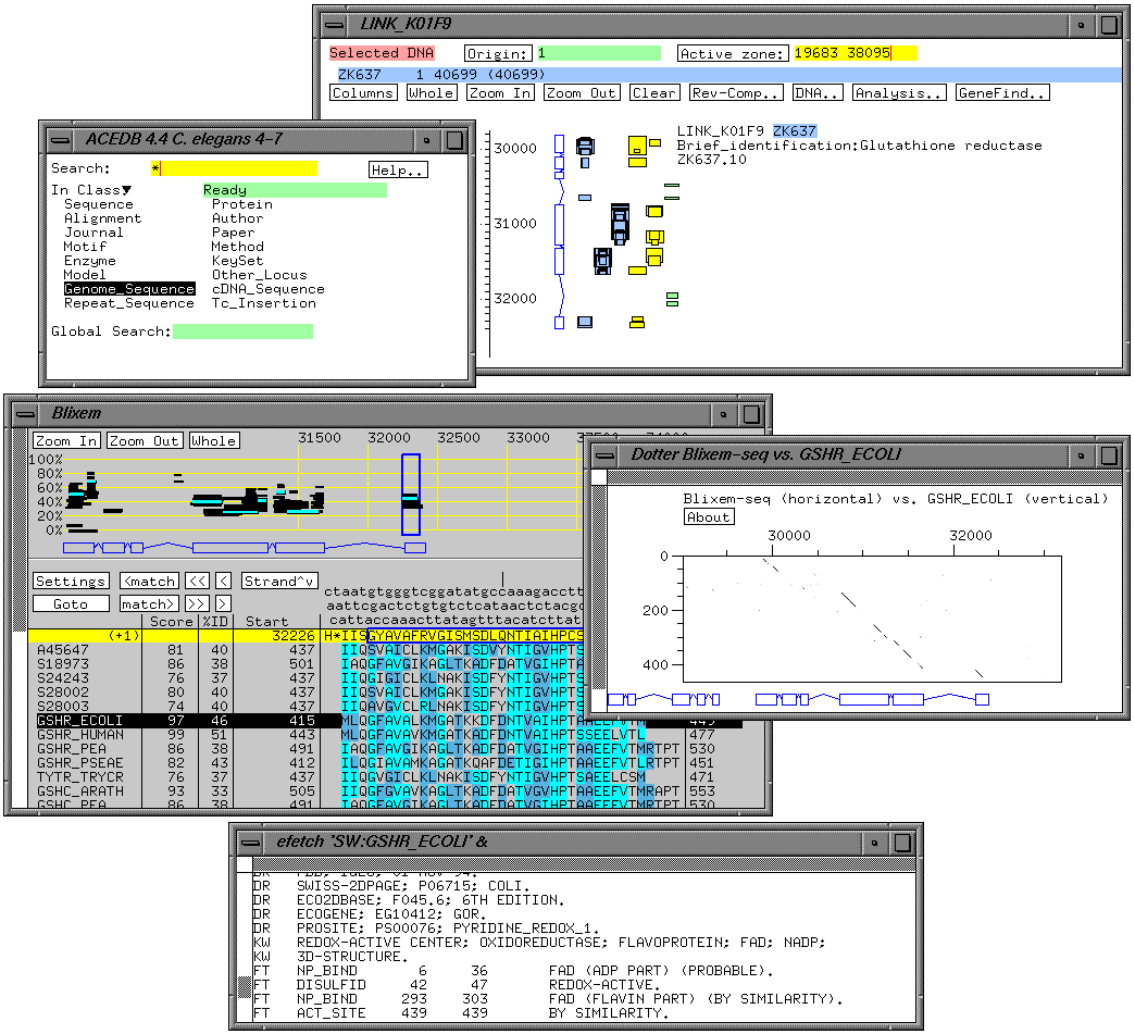


Figure 1.2. The graphical sequence analysis tools for the ACEDB genomic gene prediction workbench. Top left: the ACEDB main window. Top right: the ACEDB DNA map, showing the prediction of the *C. elegans* gene ZK637.10. The columns shown are, left to right: gene prediction (exons, blue boxes; introns, kinked lines), Blastx matches (blue boxes), Blastn matches (yellow boxes) and DNA repeats (green boxes). A large number of features are left out, such as splices, starts, stops, likely coding segments etc. (see figure 2.1). Middle left: The Blixem window of the same gene; The top part shows a schematic overview of all Blastx matches; the matches inside the scrollable box are shown as a sequence alignment in the bottom part. Middle right: A Dotter dotplot of the genome at the ZK637.10 gene prediction versus an *E. coli* homologue, called up directly from Blixem. Bottom: Swissprot annotation of the *E. coli* homologue retrieved by Efetch, activated by a double click on the match in Blixem.

Figure 1.3. The graphical sequence analysis tools for the ACEDB protein family analysis workbench. Top left: The ACEDB main window. Top right: The ACEDB PEPmap display, showing the sequence of the *C. elegans* predicted protein C14F5.5. The columns shown are, left to right: The protein sequence, a hydrophobicity plot, Blastp matches (blue boxes) and matches to Pfam protein families (green boxes). Bottom right: Blixem alignment of the Blastp matches; Pfam matches are shown schematically at the top. Middle left: Belvu alignment of the first SH3 domain of C14F5.5 to the Pfam family SH3. In Belvu, the residue colouring is based on the overall conservation of all members, while Blixem's colouring is only based on the conservation relative to the query sequence. Bottom: Swissprot annotation of a human protein with homologous domains retrieved by Efetch, activated by a double click on the match in Blixem or on the member in Belvu.

2. Materials and Methods

The work in this thesis was carried out on a cluster of UNIX workstations from Silicon Graphics running Irix 5.2, from Digital running OSF 3.2, and from SUN running SunOS 4.1.3 and 5.3. The software was written in ANSI C [Kernighan and Richie, 1988], which is easy to port between platforms, is suitable for large projects, and for which many mature debugging tools are available.

Much of this thesis concerns the development of new graphical user interfaces. For genomic sequence analysis, an efficient and powerful work environment is of paramount importance. The interfaces are based on the ACEDB database and graphics library [Durbin and Thierry-Mieg, 1996]. These were initially developed for storage, analysis and distribution of all genetic, mapping and sequence data of the nematode *C. elegans*, but have been generalised to suit any organism. The entire source code, which is freely available, is written in ANSI C, with a number of low-level API packages for accessing the database, memory management, array operations and the graphical displays. The two aspects of ACEDB that are perhaps most important for this thesis are described in more detail below.

The ACEDB database kernel

Instead of basing ACEDB on a commercially available database engine, it was decided to equip it with a native kernel. There are two main reasons for this: first, genomic data is very different in nature from the type of data for which most database systems were developed, so a specially designed system will be better tailored to handle the data efficiently. Second, the often very steep cost of commercial systems will restrain widespread usage. In fact, ACEDB's storage mechanism is very different from the dominating relational model.

The data is stored as objects, which belong to a set of predefined classes. The class models specify all attributes that *can* belong to an object of that class, but if an object omits any of the attributes, they will not use up any space. Attributes can be organised in a hierarchical tree-structure to group related attributes together in a subsection. Class models can be redefined on a live database, making incorporation of new types of data easy. Inheritance be-

tween classes is not possible, but there is no overhead in sharing one large model between related types of objects.

The efficiency of ACEDB is achieved by two-level caching system. The first cache is a raw disk cache which contains copies of disk blocks and the second cache stores in memory assembled objects. User interfaces have direct access to the second level cache, which ensures high performance. On the other hand, the direct control of the database means that only one process can access it, and although many copies can be run on different workstations in read-only mode, this effectively precludes multiple simultaneous edits. A server/client version exists, but not in conjunction with graphical interfaces.

The ACEDB graphics library

To avoid platform-dependent function calls in the graphical interfaces, a set of graphics primitives are called instead, that make the appropriate function call depending on which platform it was compiled. This allows the programmer of a graphical interface to write code that is portable throughout the platforms that are supported by the graphics library.

Some standard facilities offered by the graphics library are button, menu and keyboard event handling, postscript generation, basic scrollbars and various line and text drawing routines. The programmer can choose between a number of different window-types, depending on whether he plans to display mainly text, graphics or pixelmaps, and which scrollbars are desired. Text and graphics can be mixed in the same window, but a text window uses character based coordinates, while a pixelmap window uses pixel coordinates. Pixelmap windows can display two-dimensional arrays (matrices) of 8-bit pixels, which can be rendered dynamically to any grayscale. This is for instance exploited in the program Dotter, described in chapter 5. The graph library is based on low-level X routines and some Athena widgets. No widgets from the in some ways more evolved Motif library are used, mainly because it is under license, and ACEDB maintains a policy of free distribution.

It is possible to write light-weight graphical programs that only use the graphics library, but not the rest of the ACEDB database system. Both Blixem and Dotter exist either as stand-alone applications, or as linked-in modules in ACEDB.

The graphics library is currently supported for UNIX X-windows, Macintosh, and Windows95/NT. The two latter platforms are however somewhat experimental, and none of the tools described in this thesis are routinely released for them. Since the graphics library was initially developed for X-windows, it uses three mouse buttons. Instead of a toolbar, there is a main menu which pops up when the right mouse button is pressed anywhere in the window, except on buttons with special menus attached. On systems with fewer mouse buttons, this button is replaced by a pull-down menu on the toolbar. On single-button systems, the middle mouse button is simulated by a combination of a keyboard key and the mouse button. The system of attaching special menus to certain objects in ACEDB is widely used. For instance, Blixem and Belvu are called from the DNAmap and the PEPmap via menus under homology objects. A toolbar can be simulated by a row of buttons at the top with attached pull-down menus. This is used in most ACEDB map displays and in Belvu. Some aspects of the graphics library are only supported under X-windows. For instance, the pixelmap rendering tool used in Dotter, the Greyramp, exploits a special feature in 8-bit X-windows displays, and currently does not work on Mac or Windows.

The ACEDB genomic database front-end

A large number of genome and bioinformatics centres use ACEDB to store, analyse and distribute genomic data. At present, most end-users copy both the data and the program to a local system. Most of the tools described in this thesis are part of the distributed ACEDB code.

The front-end to ACEDB consists of a number of specialised maps. The main types are the genetic, physical and sequence maps. Objects displayed in these maps are normally 'hyperlinked', i.e. by clicking on a box or text which represents an object, the object is displayed in a new window. The class of the picked object determines whether the new window will displayed it in a certain map (e.g. markers, clones or genes), or simply in a text window (e.g. articles, alleles or strains).

Figure 2.1 shows the two most fundamental ACEDB components, the main window and the keyset window, and the DNA sequence map. The main window lists the presently stored

classes; by double clicking on one of them, all the objects of that class are listed in the keyset window ('key' means 'object' here). If a selection of the objects is desired, an expression with wildcards can be entered at the top. In the example, "Genomic_canonical" means objects of class "Sequence", that have the attribute genomic_canonical, i.e. cosmids sequenced by the *C. elegans* genome project.

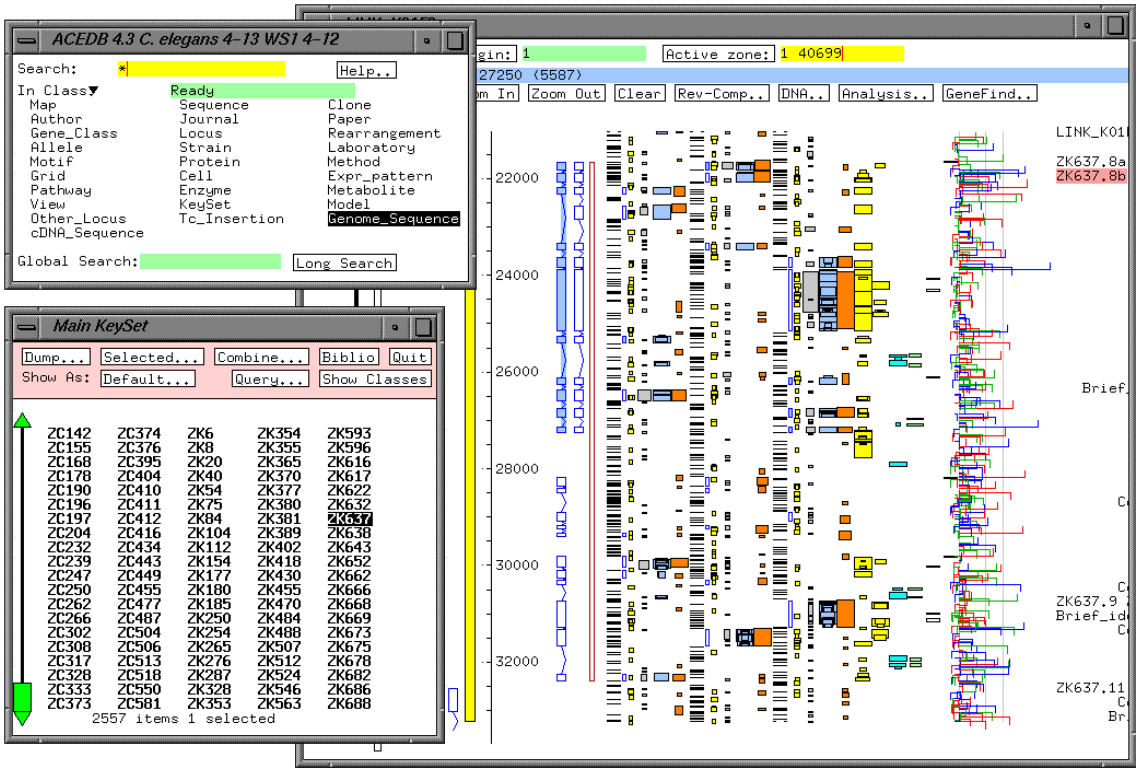


Figure 2.1. The ACEDB main window, a keyset of cosmids and the DNA sequence map. The upper gene ZK637.8 was predicted to be alternatively spliced, according to evidence from different ESTs.

Part 1: A graphical workbench for genomic sequence analysis

3. Blixem: a multiple alignment viewer for BLAST

3.1 Summary

The widely used database search programs in the BLAST suite present the results as pairwise alignments. While adequate for standard laboratory use for short query sequences, this type of output is poorly suited for high-throughput use with genomic cosmid-size queries. Not only is the large volume of output data difficult to digest, but it is also hard to form a picture of which different homology domains the query contains.

Presented here is a graphical viewer for BLAST output, Blixem, which constructs a multiple alignment of all pairwise matches and the query. A zoomable schematic global view makes it easy to see the big picture of the distribution of the matches, while another display shows the actual multiple alignment in a scrollable region. Blixem is also hyperlinked to the annotation of matching sequences via the retrieval tool Efetch (chapter 6), and to the dotplot program Dotter (chapter 5). Detailed similarity analysis and functional annotation thus become efficient enough for processing large amounts of data. Coupled to the gene prediction workbench in ACEDB, Blixem is used routinely in genomic sequencing projects. Examples of usage are taken from the *C. elegans* genome, from which cosmids totalling 50 million basepairs have been analysed this way.

3.2 Introduction

With the arrival of large scale genome sequencing projects [Wilson *et al.*, 1994; Collins, 1995; Dujon, 1996], where highly automated laboratory techniques produce DNA sequences at an ever increasing rate, the need for equally powerful sequence analysis tools has become obvious. Characterising genes found in 'blindly' sequenced DNA by searching for homologous proteins is presently the most tractable way to predict their function. Thanks to efficient database searching programs such as BLAST [Altschul *et al.*, 1991], Blaze [Brutlag *et al.*,

1993], and Flash [Califano and Rigoutsos, 1993], and specialised hardware [Collins *et al.*, 1988], searching time is of little concern. Instead, the main bottleneck lies in the manual evaluation of the matches reported by the search programs, which often form a list of many thousands of potential homologies.

Summarising the results automatically by e.g. using the annotation of the most similar sequence can often lead to misleading results. Not only is the most similar sequence not necessarily the best annotated one, but it is also easy to get misguided by partial matches to multi-domain proteins. Furthermore, similarities that are truncated or are found in predicted intronic sequence may provide evidence that a predicted gene is incorrect, and requires re-examination of both the DNA sequence and the exon predictions. These problems need human intervention to ensure a high quality of both gene predictions and annotation.

However, manual reading of exceedingly long search result lists becomes an inhuman task for sequencing projects of several megabases. What is needed is a workbench which automatically performs the routine actions of a sequence analyst as well as presents the cases where manual inspection is necessary in an interactive user-friendly environment [Bernstein, 1987; Medigue, 1995].

This chapter describes a core component, Blixem, of the large-scale sequence analysis workbench presented in this thesis. See figure 1.1 and 1.2 for an overview of all workbench components. Blixem provides interactive multiple sequence alignment analysis of database matches reported by the search programs in the BLAST suite, which produce a list of ungapped alignments, or MSPs (Maximal Segment Pairs). The MSPs should first be filtered by MSPcrunch (chapter 4), to reduce redundancy, remove low complexity 'junk' matches, and enhance the sensitivity and selectivity of multiple consistent matches.

Blixem is a general purpose viewer, although it was developed especially with the *C. elegans* and human genome sequencing projects in mind. It can either be used as a stand-alone application, or as an analysis tool incorporated into the gene prediction workbench in the genomic database ACEDB. The combination of Blixem and ACEDB forms an efficient interactive system for making gene predictions where homology to other proteins can be analysed in detail to improve both the gene structure and the functional annotation.

3.3 General features

Blixem does not parse BLAST output directly. Instead it relies on MSPcrunch to convert the BLAST results to a format readable by Blixem. These formats, ‘seqbl’ and ‘exblx’, are described in chapter 4. Any program could be used to convert BLAST output to one of these formats, but MSPcrunch is strongly recommended, since it also enhances the quality of the data.

Overview display of matches

The upper part of Blixem, the "Big Picture" display, draws all the matches to the query symbolically as lines, which on the y-axis are positioned according to the percent identity of the MSP (see figure 3.1). For DNA queries, the matches to either only one strand or to both strands can be shown simultaneously (see figures 3.2 and 3.3). By pressing the “whole” button, matches along the whole query are shown, while the zoom buttons allow the user to enlarge a particular region. If a match is clicked on, all MSPs with that database sequence become highlighted. The blue square frame in the Big Picture is the part that is aligned on the residue level in the Alignment display below. The Big Picture display is centred around this area, which can be shifted along the query with the middle mouse button.

Alignment display

For Blastp, Tblastn and Tblastx, a single query sequence is drawn on yellow background, as in figure 3.1. For Blastn, both strands are shown simultaneously (figure 3.2), and for Blastx the three translations of one strand are shown (figure 3.5). In the latter cases, the MSPs are aligned under the appropriate query sequence. Residues in the matching sequences are coloured in three different colours: Cyan (bright blue) for the same residue as the query, light blue for a conserved substitution and no colour for a mismatch.

For Blastx, the DNA query sequence is drawn on top, with staggering so that one triplet covers the width of one character. Amino acid residues in frame one thus correspond to triplets that start with a base on the top DNA row, and end with the base on the third row that is just to the right. Residues in frame two start on the second row, and residues in frame three on the third.

The start and end coordinates shown in the columns adjacent to the alignment refer to the entire match, which may extend beyond the current window. Horizontal scrolling is done either with the scroll buttons at the left, or with the middle mouse button, which starts up a crosshair, and centres the display on the crosshair position where the button is lifted. When the crosshair is on, the sequence coordinates of the query and the last clicked matching sequence are shown. For Blastx, the query coordinate is that of the base under the crosshair. The scroll buttons allow horizontal scrolling of the alignment in three different step sizes: a residue (>, <), a whole window-width (>>, <<), or to the next match (match>, <match). If not all matches fit in the window, the lower parts of the alignment can be viewed by using the vertical scrollbar on the left.

Annotation of a protein is fetched by double clicking on the sequence of interest in the bottom display. The program Efetch (chapter 6) will then retrieve the record from an external database and display it in a separate window (figure 3.1). Alternatively, a world wide web browser can be launched, which calls Efetch and marks up the retrieved entry so that references can be followed to other databases. The fetch method can be set interactively in the Settings tool. The default fetch method can be set by environment variables (see http://www.sanger.ac.uk/~esr/Blixem.html).

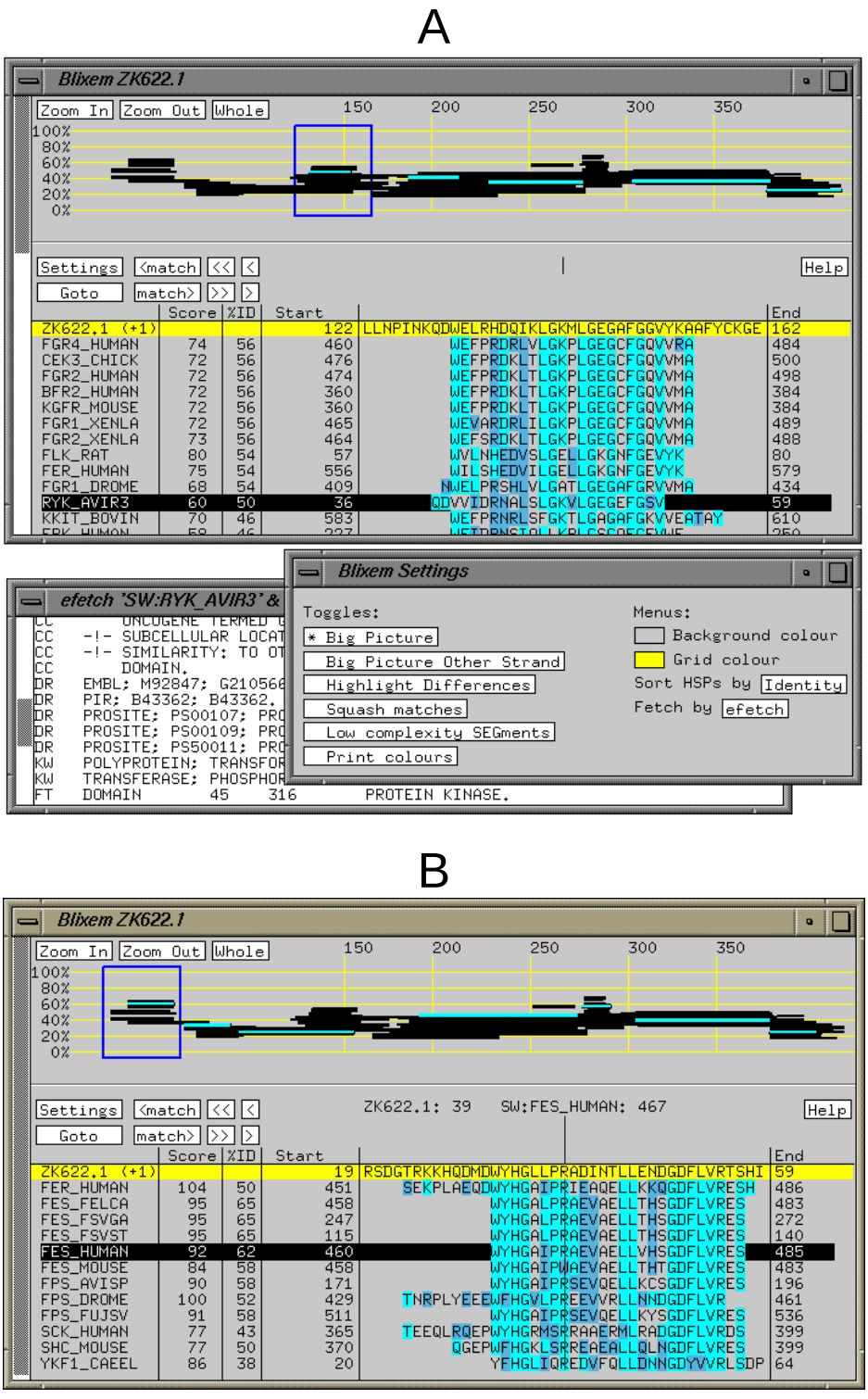


Figure 3.1. Blastp matches to the *C. elegans* protein ZK622.1 shown in Blixem. The top display shows a global overview of the MSPs in the vicinity of the alignment window in the bottom display. Each MSP is drawn as a line at its position in the query and percentage identity level in the overview. All matches to the clicked protein become highlighted, which makes it easy to identify the domains shared with other sequences. For instance, the matches to the protein kinase RYK_AVIR3 only covers the C-terminal part (a), while the matches to FES_HUMAN, which contains an SH2 domain N-terminal to the protein kinase domain, covers the entire query (b).

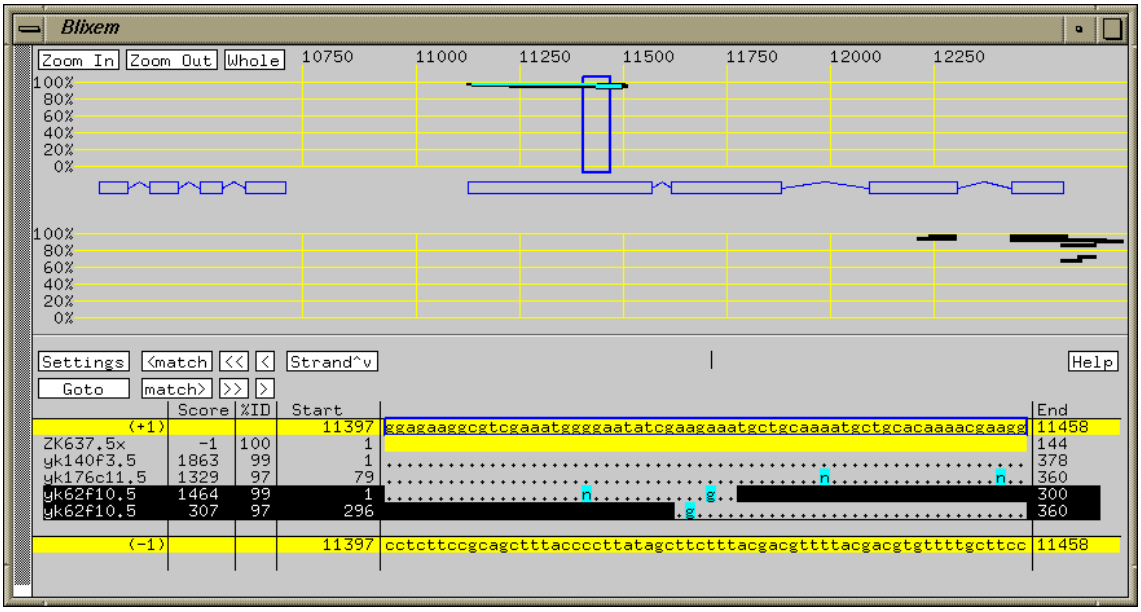


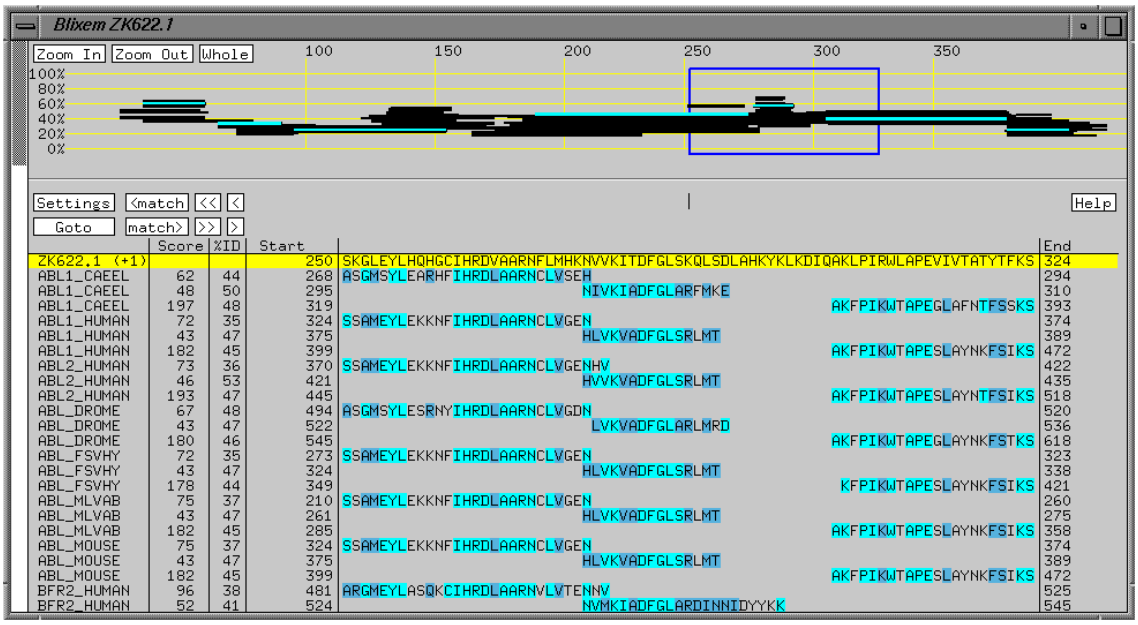
Figure 3.2. Blastn matches to *C.* elegans ESTs aligned in Blixem. By setting Blixem in "highlight differences" mode, identical residues are shown as dots, while mismatches are highlighted. The example shows a typical case of EST hits, with a number of matches in the 5' and 3' ends of the gene. The last intron and exon are confirmed by EST matches. The quality of EST sequences usually drops after 300 base pairs, which is exemplified here by a number of mismatches to the genomic sequence and a frameshift in the EST yk62f10.5.

Settings tool

The button "settings" contains most configuration options. It can either be used as a pull-down menu or as a button. Clicking on it produces a separate Settings tool window (figure 3.1). The options on the left side are on/off switches:

Big Picture	Toggle Big Picture (top display) on/off.
Big Picture Other Strand	Toggle between single and double strand display in the Big Picture.
Highlight differences	Show identical residues as a dot (.) and draw mismatching residues in bright blue.
Squash matches.	Draw multiple matches to the same sequence on one line (see figure 3.3).
Low complexity analysis	Turn on the sequence complexity display (see below).

A



B

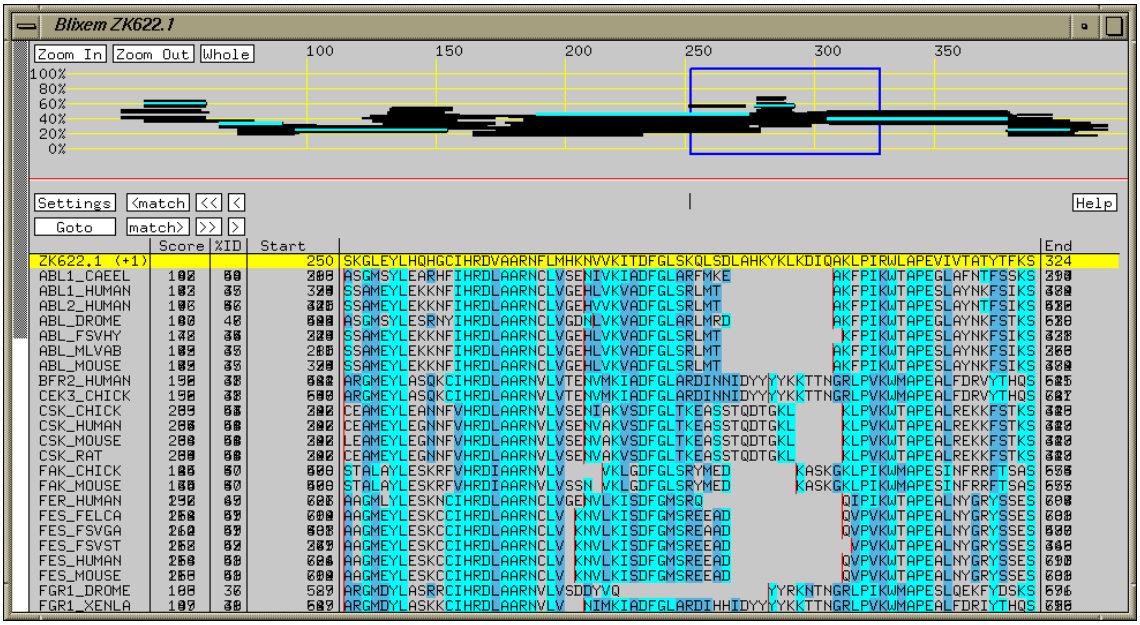


Figure 3.3. When the matches are split by many gaps, the normal display method, which draws one MSP per line, can cause the alignment to become broken up (A). In the "squash matches" mode (B), all MSPs to the same database sequence are superimposed on one line, and the start of each MSP is marked with a red vertical line.

On the right hand side in the Settings tool, a number of multiple choice settings are available. These are selected with right mouse button menus. The background and the grid in the Big Picture display can be set to 32 different colours, and the MSPs in the Alignment window can be sorted, top to bottom, in four ways: alphabetically, highest score first, highest identity first, or positionally, with the most N-terminal first. Fetching can be done by Efetch, WWW, or ACEDB (only if run inside ACEDB).

Low complexity analysis

To analyse regions of biased composition (i.e. low sequence complexity), Blixem has a special display panel which can draw three different curves of the complexity of the query sequence. The complexity is defined as the shannon entropy, i.e.

$$\sum_{i=1}^{20} f_i \ln f_i$$

for all residues i with a frequency f_i in the window of a certain length. The three curves can be assigned individual window sizes and colours. The panel can also show segments of low complexity from the Seg program [Wootton and Federhen, 1993]. Seg uses three empirically derived window sizes and complexity thresholds to decide whether a segment has stringently low complexity, medium low complexity, or "non-globular" low complexity, i.e. the segment is unlikely to have a globular fold [Wootton, 1994]. The stringent level is the default operation of Seg, and uses a window size of 12 and a threshold of 2.2 bits. The other levels have window sizes of 25 and 45, and thresholds of 3.0 and 3.4 bits. Internally, Seg uses slightly higher 'trigger' cutoffs to find initial segments, which are merged if they overlap [Wootton, 1994]. As a rule of thumb for protein sequences, a complexity below 3 bits is considered a significant deviation from what is expected of an unbiased sequence, at least with window sizes above 25. To display Seg segments in Blixem, they are read in as pseudo MSPs with scores of -4, -5 and -6 for the three different levels. The default window sizes of the complexity curves are set to the three Seg window sizes.

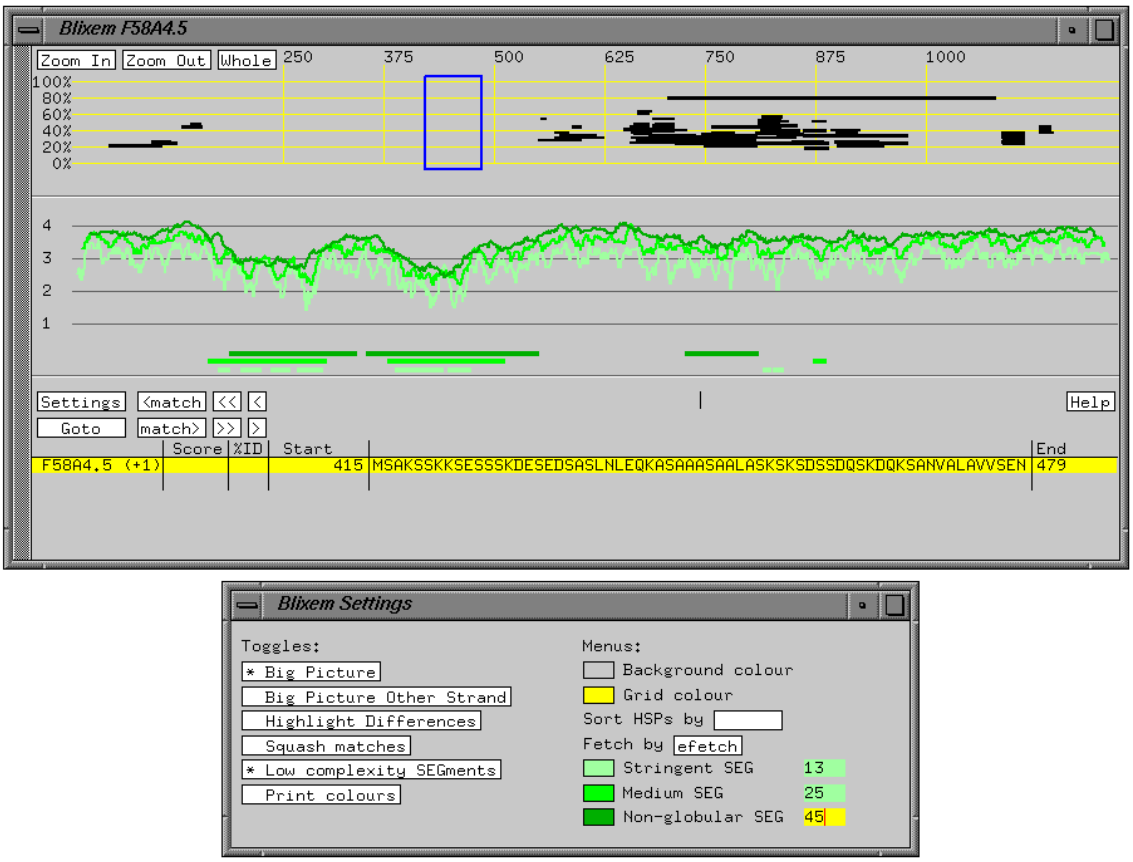


Figure 3.4. Low complexity analysis of the *C.* elegans protein F58A4.5, which has a region of biased composition between 200 and 500 (mainly serine rich). The entropy of the sequence is plotted using three different window sizes. Underneath, segments are drawn that were found by the Seg program at three different stringency levels. Blastp reports thousands of matches to unrelated sequences with a similar bias. This can be avoided by removing the region in the query with e.g. the Seg program, or, as was done here, by filtering the Blastp output with MSPcrunch.

Dotter analysis

The dot-plot program Dotter (chapter 5) is linked in with Blixem. It is normally used to make a dot-plot of the query and a database sequence, but can also be called to make a plot of the query vs. itself. The Blast MSPs are also passed on for display in Dotter, as shown in figure 5.2. If Blixem is used for Blastx data, Dotter makes three dotplots of the different frame translations, and superimposes them in one graph. When using Blixem to analyse an entire cosmid, only a part of the query is passed on for Dotter analysis. Blixem uses a heuristic to guess which region is relevant, but may get confused by repeated domains. If this

happens, it is possible to select the query region manually with the option “manual Dotter parameters” under the main Blixem menu. This also allows manual setting of the zoom factor in Dotter. Sequence features such as exons and introns are passed on from Blixem to Dotter.

If Blixem is run in stand-alone mode, Dotter relies on Efetch to retrieve the entire sequence, since BLAST does not provide this. If Efetch fails, only the matching region can be analysed in Dotter.

3.4 Special features inside ACEDB

For efficient large scale genomic sequence analysis, Blixem has been integrated in the ACEDB genomic database package. This is particularly useful if the user wants to predict genes in a DNA query sequence, since ACEDB includes a semi-automatic gene prediction environment. Blixem can be called up from ACEDB's sequence display windows so that exon predictions can be validated in the light of homology to other proteins.

When used from ACEDB, the BLAST output is first filtered by MSPcrunch, which outputs the accepted matches into .ace format (see figure 4.9c). These are then read in to and stored in the ACEDB database. When Blixem is called, ACEDB converts the data to Blixem’s internal data structure. Exons and introns are passed on as pseudo MSPs with scores of -1 and -2, respectively. As shown in figure 3.5, the genes are displayed both in the Big Picture overview, and in the alignment.

Double clicking on a match in Blixem, which normally calls Efetch to retrieve the annotation, is here by default set up to call up the corresponding ACEDB objects. This can be used to retrieve annotation independent of Efetch, as shown in figure 3.5. It also allows further analysis of the matching object in the DNAmap or PEPmap (chapter 8). Normal Efetch annotation retrieval can also be used. To make this the default when using Blixem in ACEDB, an environment variable ‘BLIXEM_FETCH_EFETCH’ must be set.

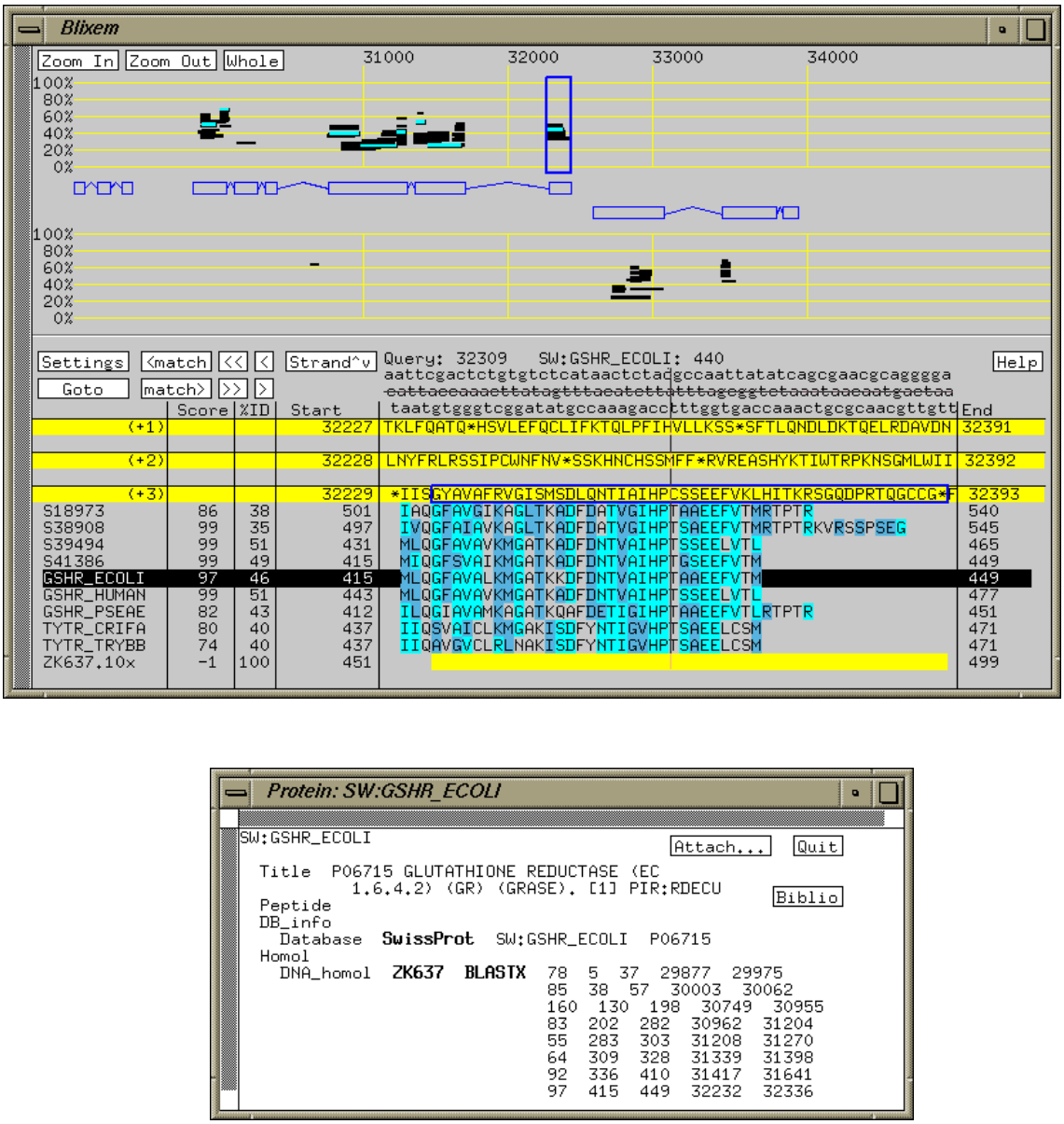


Figure 3.5. Multiple alignment of Blastx matches in Blixem when called from ACEDB. Example taken from the *C. elegans* cosmid ZK637, showing matches between the predicted gene ZK637.10 and glutathione reductases. The global overview display in the top panel shows MSPs on both the top and bottom strand. The lower alignment panel shows the DNA sequence at the top, and its translation in three frames on the yellow lines below. Predicted exons are marked in the alignment panel as blue frames on the translated genomic sequence and as yellow boxes with the other MSPs (ZK637.10x). It is very common that the MSPs extend slightly beyond the true end of the exon.

3.5 Blixelect: an organiser for multi-query Blixem analysis

For medium-scale sequence similarity analysis projects, where only inspection of BLAST matches to a set of query sequences is wanted, but no gene prediction is intended, setting up an ACEDB database may involve unnecessary overheads. For such cases, a simple organiser tool was developed, called Blixelect. It reads a list of query names from a file, and expects two files for each query: one containing the query sequence and one with MSPcrunched BLAST output in the seqbl format (figure 4.9b). The filenames must start with the query name, but the extensions may be chosen by the user. Normally they are '.seq' and '.seqbl'. Blixelect first parses the seqbl file to count the number of matches, which it lists next to each query name, as shown in figure 3.6. To save space, Blixelect can be set to only list queries with one or more matches.

Another option is to force Blixem to automatically call dotter with the first listed database match. This can be useful for e.g. efficient comparison of the proteins of two genomes, if one only wants to establish if there is any homologue in the other database. The keyboard arrow keys can be used to go to the next query in the Blixelect window. Queries with a sought feature such as 'has a homologue' can be selected, and after all of them have been inspected, a list of the selected queries can be printed out.

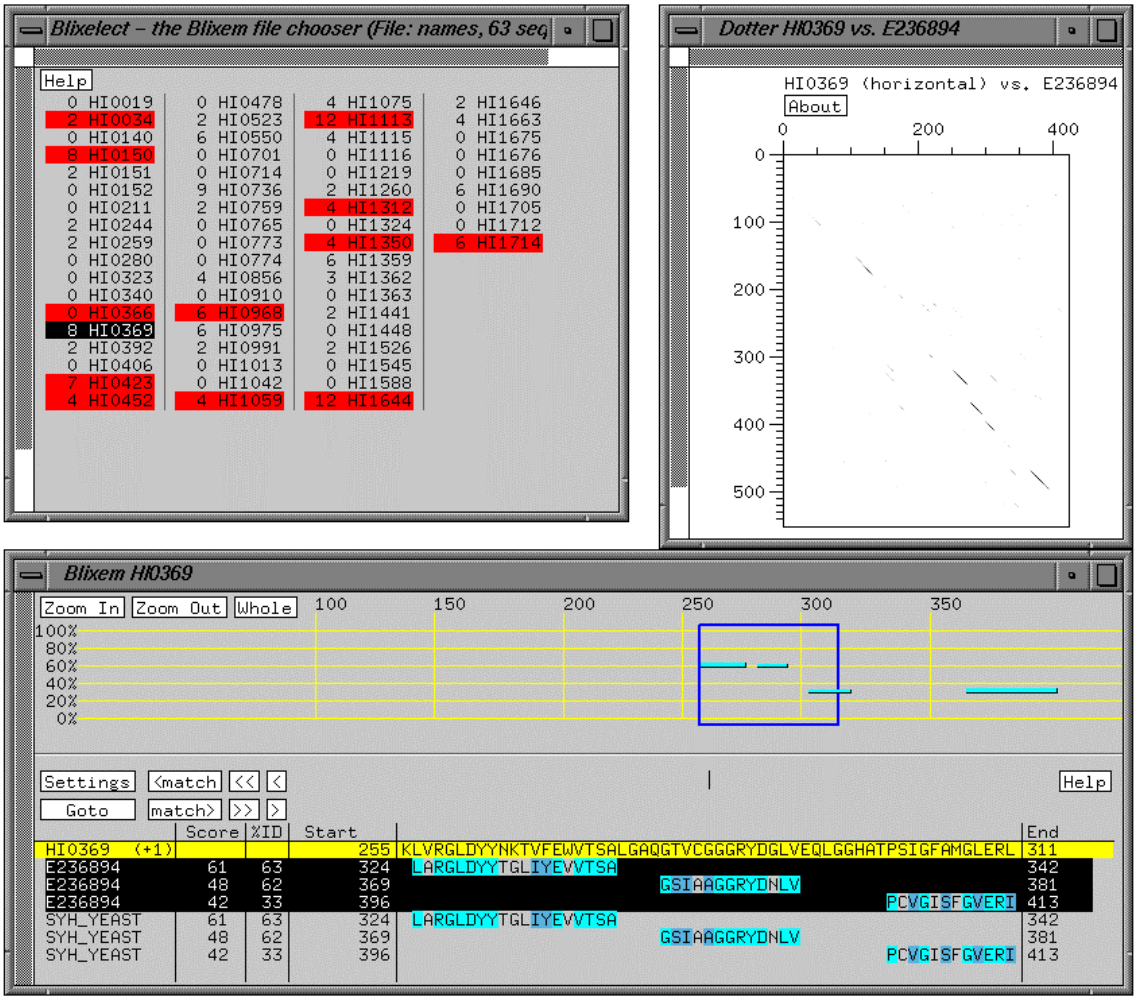


Figure 3.6. The Blixelect tool (top left) is useful for analysing a batch of BLAST searches. The number of matches to each query is listed next to its name. By clicking on a query, Blixem and Dotter (optionally) are called up. This example contains all *H. influenzae* proteins that were found to match *C. elegans* but not *S. cerevisiae* using stringent MSPcrunch criteria. After analysing the output using less stringent criteria, the proteins marked red were found likely to be homologous to a yeast protein.

3.6 Discussion

A number of graphical tools exist for schematic display of database hits [Medigue *et al.*, 1995; Rubin, 1996]. As far as we are aware, however, Blixem is the only viewer that reconstructs a multiple sequence alignment from BLAST matches. The unique combination of a schematic overview and detailed residue analysis, and the integration with the other workbench tools make Blixem a key component for complex analysis tasks. A number of other systems also employ Blixem as a BLAST viewer. It is suited for WWW servers that report BLAST results, since it can read both the query sequence and the MSPs from standard input. For web usage, one could argue that it might be preferable to re-implement Blixem in Java, which would make it platform-independent. On the other hand, progress is being made on porting the ACEDB graph library to PC and Mac platforms, which in practice would make the standard Blixem application equally portable, but faster.

Blixem can in principle be used to view the output of any database search program that produces ungapped matches. Allowing insertions in the query sequence would require a fundamental change in Blixems drawing routines, and the result of many insertions at different points could be detrimental to the alignment. We have therefore not pursued this approach. To display output from programs that report gapped matches, they would first have to be converted to a set of MSPs. Ironically, the latest version (2.0) of BLAST reports gapped alignments, which makes it a lot less suitable for Blixem than version 1.4.

A number of features may be incorporated in the future. For instance, it would be preferable to limit the vertical scrolling to the alignment part of the display only, so that the Big Picture would be visible at all times. Although such a mechanism exists in the ACEDB graph library, such windows can not be printed properly, and are therefore not used. To use Blixem as a generic graphical display tool for showing data from an arbitrary external program, a general interface specification would be needed. At present, all displays are hard-coded in the program, even the Seg displays. Instead of using “magic scores” for each method, such data should instead be converted to a standard format, which would specify the type of data and display parameters such as colour and scaling. If such a format becomes

widely used, it would allow graphical display of the output of third party sequence analysis programs, with a very small overhead for their developers.

4. MSPcrunch: a BLAST enhancement tool for large-scale sequence similarity analysis

4.1 Summary

For high-throughput genomic sequence similarity analysis, most database search programs generate prohibitively large amounts of information. To allow an annotator to concentrate on essential tasks, an automatic system was developed that makes many of the standard decisions a trained sequence analyst would, and only presents the most informative matches to the annotator.

The system is currently based on the database search programs in the BLAST suite, which have been primarily designed for short query sequences. A number of algorithmic additions were made that are especially valuable for multi-domain queries, which is the norm for genomic cosmid-size sequences. It is implemented by changing some input parameters to BLAST, and applying a number of filtering rules to the output by a post-processing program, called MSPcrunch.

The main advantages compared to default BLAST searching are: 1. Domains with weak but significant hits will not be missed due to other higher-scoring domains. 2. 'Junk' matches with biased composition are eliminated. 3. Higher sensitivity and selectivity are achieved for multiple matching segments in the 'twilight zone', thanks to strict consistency criteria. 4. A range of output formats is provided, including gapped alignment, graphical schematic and tabular data.

The default mode of operation has been calibrated empirically to suit the needs of efficient and sensitive genome annotation. This mode will remove redundant matches even if they are significant, but will report the highest scoring of the statistically insignificant matches, since these may prove to be biologically relevant after further analysis.

4.2 Introduction

Large scale genome sequencing projects [Wilson *et al.,* 1994; Dujon, 1996], generate new sequences at such a high rate that homology analysis has become a serious bottleneck. The availability of fast database searching programs such as BLAST [Altschul *et al.,* 1991], Fasta [Pearson, 1990],and Flash [Rigoutsos and Califano, 1993], and fast parallel computing hardware such as MasPar [Brutlag *et al.,* 1993], DAP [Collins *et al.,* 1988] and Bioccelerator [Esterman, 1995] ensure that the actual computation is a more or less solved problem for at least the foreseeable future. Analysing the search results generated from a hundred thousand newly sequenced basepairs per day, however, presents a major challenge. Available search programs produce results that are too time-consuming to digest on a large scale. For genome projects, there is a demand for automated analysis systems [Scharf *et al.,* 1994]. Our philosophy is that human evaluation is still a required for high-quality sequence analysis, but many monotonous time-consuming tasks can be automated by computational methods. Presented here is a program, MSPcrunch, which applies a number of rules to evaluate matches reported in a database search, and concisely presents only the most relevant information for further consideration.

Much of the recent developments of database search programs has been refinement of the statistical significance of a match [Karlin and Altschul, 1990], i.e. finding the probability that a match was caused by chance. For most search algorithms, the *extreme value distribution* [Gumbel, 1958] has proven the best model so far for describing the distribution of optimal scores to database sequences [Altschul *et al.,* 1994]. Statistical significance of a match calculated under this model has been shown to generally agree well with biological relevance.

However, sequences that are unusually rich in a few amino acid types may give rise to spurious matches that under the random model are very significant. A solution to this is to detect low complexity regions and remove them from the query [Wootton and Federhen, 1993; Claverie and States, 1993], but this has the drawback of artificially disrupting the query sequence, which may lead to missed true similarities [Koonin *et al.,* 1996b]. The risk of this is reduced if only the abnormally frequent residues are removed [Casari, personal communication]. In this chapter a method for avoiding matches caused by biased composi-

tion is described, that instead of modifying the query, compares the observed score to the expected score, given the residue composition. If the observed score was deemed to be the result of compositional bias, the match is rejected.

When the amino acid composition is not biased, the statistical significance of single matches reported by BLAST [Altschul *et al.*, 1990] is normally reliable. The significance of multiple matching segments is however difficult to calculate properly using the extreme value distribution, and BLAST uses a heuristic to decide which segments are consistently ordered with respect to each other [Karlin and Altschul, 1993]. This heuristic does not take the distance between two segments into account, and often overestimates the combined significance of what are really independent segments [Koonin *et al.*, 1996b]. In this chapter, a method is described that uses empirically derived consistency rules for multiple matches between two sequences, which explicitly takes the distance between segments into account. This has a flavour of gap penalties, but the method is very different from dynamic programming.

MSPcrunch is implemented as an add-on tool for the BLAST programs, which were chosen because of their robustness, speed and the underlying philosophy of only looking for ungapped matching segments, which allows finding of matches to multiple independent domains. For genomic analysis, which involves finding protein matches to short exons interspersed between introns, this is a big advantage. The main problem with using BLAST for large multi-domain queries is that it by default only reports the highest scores. This can cause weakly conserved domains to be missed if other domains generate too many high-scoring matches. The problem can be alleviated by changing a parameter so that BLAST reports all matches. This can cause severe over-reporting, but MSPcrunch then removes redundancy in congested regions so that only the high-scoring matches of any given region are kept.

Statistical significance tends to work well to support clear similarities. Weakly significant matches may or may not infer homology. Matches in this so-called ‘twilight zone’ are a mixture of true and false similarities, and the problem is to separate the signal from the noise.

For ungapped matches from BLAST, the problem of spurious matches is bigger than for dynamic programming methods that find the single best match.

After filtering out redundant and ‘junk’ matches, the accepted matches can be viewed either as a graphical ``Big Picture" schematic display with one database sequence per line, as a gapped alignment, or can be exported to other programs such as ACEDB or Blixem in a tabular form.

4.3 Methods and materials

MSPcrunch is in itself not a database search program, but relies on the programs Blastp, Blastn, Blastx, Tblastn and Tblastx from the BLAST suite [Altschul *et al.*, 1991]. This has the advantage that end-users can use any BLAST service provided on the internet and process the output with MSPcrunch. Another reason for implementing MSPcrunch as a post-processing filter was to keep it flexible to adaptation to output from other database search programs that may become popular in the future.

Version 1.4.6 of BLAST was used. BLAST searches for ungapped segments in two sequences and extends them until the maximum score is achieved. All such maximal segment pairs (MSPs) scoring above a certain threshold are reported. This threshold is normally set to report 10 spurious hits, but here we lower it to 25 for Blastp and 35 for Blastx, Tblastn and Tblastx, using the BLOSUM62 score matrix. A lower threshold causes BLAST to report more spurious and true MSPs, but MSPcrunch removes most of the spurious ones by consistency checks described below.

The BLAST B parameter was set to a high enough value so that it does not limit the number of MSPs reported (10^6). For Blastx on cosmids, we use the -span1 option to avid a too voluminous output, and the score matrix BLOSUM62-12, which is a slightly modified version of the BLOSUM62 matrix [Henikoff and Henikoff, 1992]. The modification was to lower the score for stop codons from -4 to -12. Such a high penalty for stop codons is preferable for DNA sequences with very low error rates. Note that for older versions than 1.4 of

BLAST one needs to set the S (first pass cutoff) parameter set to a low value, since only database sequences that have a match above this threshold will be searched in the second pass, which looks for matches above S2.

Running Blastx on long DNA query sequences (more than 10^5 bases) may prove impossible due to memory limitations. For such cases, we have developed a program Seqsplit which splits up the query into smaller chunks with overlaps. After running BLAST on the smaller chunks, another program Blastunsplit combines all the output files into one and reconstructs the positions in the original query.

The protein sequence database searched, Swir, is a low-redundancy collection of sequences from Wormpep, Swissprot and TREMBL [Bairoch and Apweiler, 1996]. Redundancy was removed by a program nuswir [P. Rice, personal communication], which rejects any sequence from TREMBL that is more than 95% identical to any Wormpep or Swissprot entry. Wormpep entries in Swissprot were also removed. 118182 Release 11 of swir consisted of 118182 sequences. 7299 of these sequences came from Wormpep release 11, 51474 from Swissprot release 33 and 59409 from TREMBL. See chapter 9 for more information on Wormpep.

4.4 MSPcrunch rules

The post-processing of MSPs from Blastx or Blastp in MSPcrunch is outlined in figure 4.1. An MSP consists of an ungapped alignment between a segment in the query sequence, simply called 'the query' hereafter, and a segment of a database sequence, called 'the subject'.

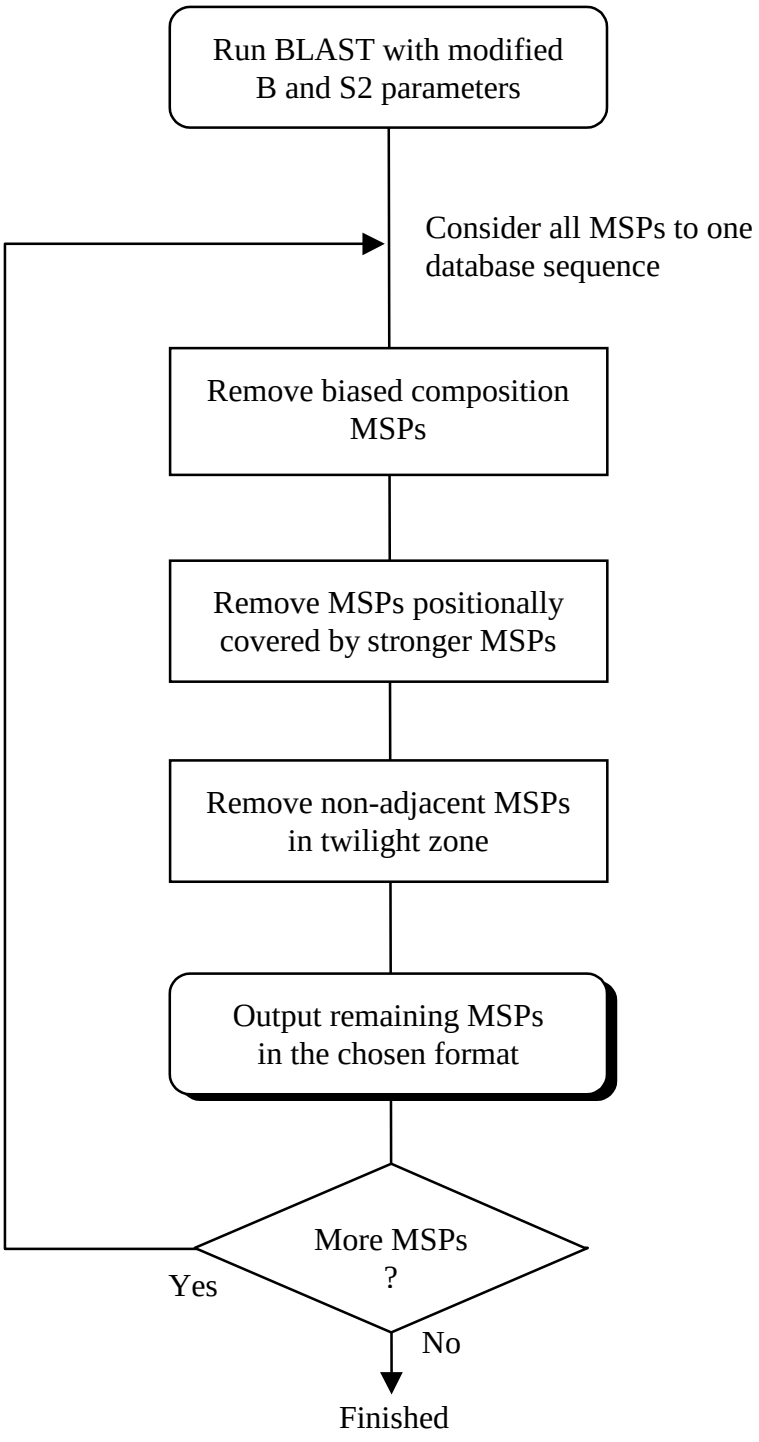


Figure 4.1. Overview of the different modules in MSPcrunch.

Biased composition matches

Biased composition MSPs are detected by a rule that compares the score of the MSP with the score of an MSP with no composition bias, in relation to the amino acid composition of the MSP in question.

The expected score of an MSP, S_{exp}, is the average score two random sequences of that particular length and amino acid composition would have. For a typical MSP the expected score is negative, but if the composition is biased the expected score may be positive. The expected score is calculated the following way: Two vectors Q and D with the observed frequencies of the amino acids in the query and database segments making up the MSP are constructed. The vectors are then scored against each other so that

$$S_{\exp} = L \sum_{i=1}^{20} \sum_{j=1}^{20} Q_i D_j M_{ij}$$

where L is the length of the MSP and M is the scoring matrix. This method yields the same result as random shuffling methods would asymptotically, but is faster. To avoid unjustified high values of S_{exp} due to small sample sizes in short MSPs, the frequencies Q_i and D_i are given pseudocounts according to

$$Q_i = \frac{Qc_i + \alpha \cdot p_i}{L + \alpha}, \quad D_i = \frac{Dc_i + \alpha \cdot p_i}{L + \alpha}$$

where Qc_i and Dc_i are the counts of residue i in the query and database segments in the MSP, and p_i is the background frequency of residue i. A good value for the pseudocount weight α was found to be 5 (cf. [Henikoff and Henikoff, 1996]). Using a lower weight tends to reject too many short true matches, while a higher weight may cause acceptance of too many biased composition matches.

To evaluate whether the score S of the MSP is the result of biased composition, we calculate the bias-ratio β:

$$\beta = \frac{S - S_{\exp}}{S - LM_{\exp}}$$

where M_{exp} is the frequency-weighted expected score of random (unbiased) sequences according to the scoring matrix used. For BLOSUM62, M_{exp} = -0.945. β can be used as an index of how biased the composition of the MSP is. As a rule, $\beta < 0.8$ is a clear sign that the MSP has a biased composition and should be rejected. Table 4.1 shows to what degree the unwanted biased composition MSPs are removed for different values of β. For values of β above 0.8, loss of good matches with slight bias becomes a problem.

A simpler and less effective version of this algorithm has previously been described. The method described here has been implemented since version 1.2 of MSPcrunch.

	YMH5_CAEEL		B0284.1		CA14_CAEEL		GRP_ARATH	
β	biased	good	biased	good	biased	good	biased	self
0.1	331	132	292	3	2503	27	1625	1
0.2	298	132	288	3	2485	27	847	1
0.3	221	132	277	3	2443	27	326	1
0.4	133	132	51	3	2418	27	66	1
0.5	23	132	191	3	2378	27	15	1
0.6	7	132	67	3	2255	27	3	0
0.7	0	132	15	3	1867	27	0	0
0.8	0	132	3	3	623	25	0	0
0.9	0	132	0	0	22	22	0	0

Table 4.1. Separation of biased composition matches from good ones by MSPcrunch as a function of the bias-ratio β. The numbers refer to counts of MSPs that passed the MSPcrunch adjacency criteria. No coverage limit was used (see below). YMH5_CAEEL (Swissprot P34472) has a stretch of biased composition (acid-rich) in the N-terminus as well as a reverse transcriptase domain and 3 C-type lectin domains (see figure 4.7. B0284.1 (Wormpep CE00650) has a charged-residue biased region. CA14_CAEEL (Swissprot P17139) is a collagen, containing mainly [Gxy] repeats. Although these matches have biased composition, they are to other collagens, and it is therefore useful that MSPcrunch does not reject all of them. GRP_ARATH is the most biased composition protein in Swissprot 28 (72% Glycine, relative entropy 2.0 bits). In this extreme case even the match to itself does not pass the biased composition test when $\beta > 0.6$.

Positional coverage limitation

For genomic cosmid-size analysis, this is perhaps the most important feature of MSPcrunch. BLAST has a settable limit for the number of highest scoring database sequences to report, which is by default set to 250. If matches to one domain fill this quota entirely, other weaker scoring domains will not be reported. To prevent this from happening, we set the limit in

BLAST to a sufficiently high number that all matches are reported (B=1000000). MSPcrunch then limits the number of matches by taking the position in the query into account. If the query segment of an MSP is already covered by many other MSPs that score higher and are accepted by MSPcrunch, the MSP is rejected. Figure 4.2 shows the MSP coverage on a cosmid sequence of 40 kbases. The two main causes of very high number of MSPs covering certain regions are strong amino acid frequency bias and large protein families. We limit the coverage by default to 10-fold on each strand. An MSP is only rejected if every residue in the query segment is covered. In practise, due to staggering of matches, this leads to a coverage up to 20-fold.

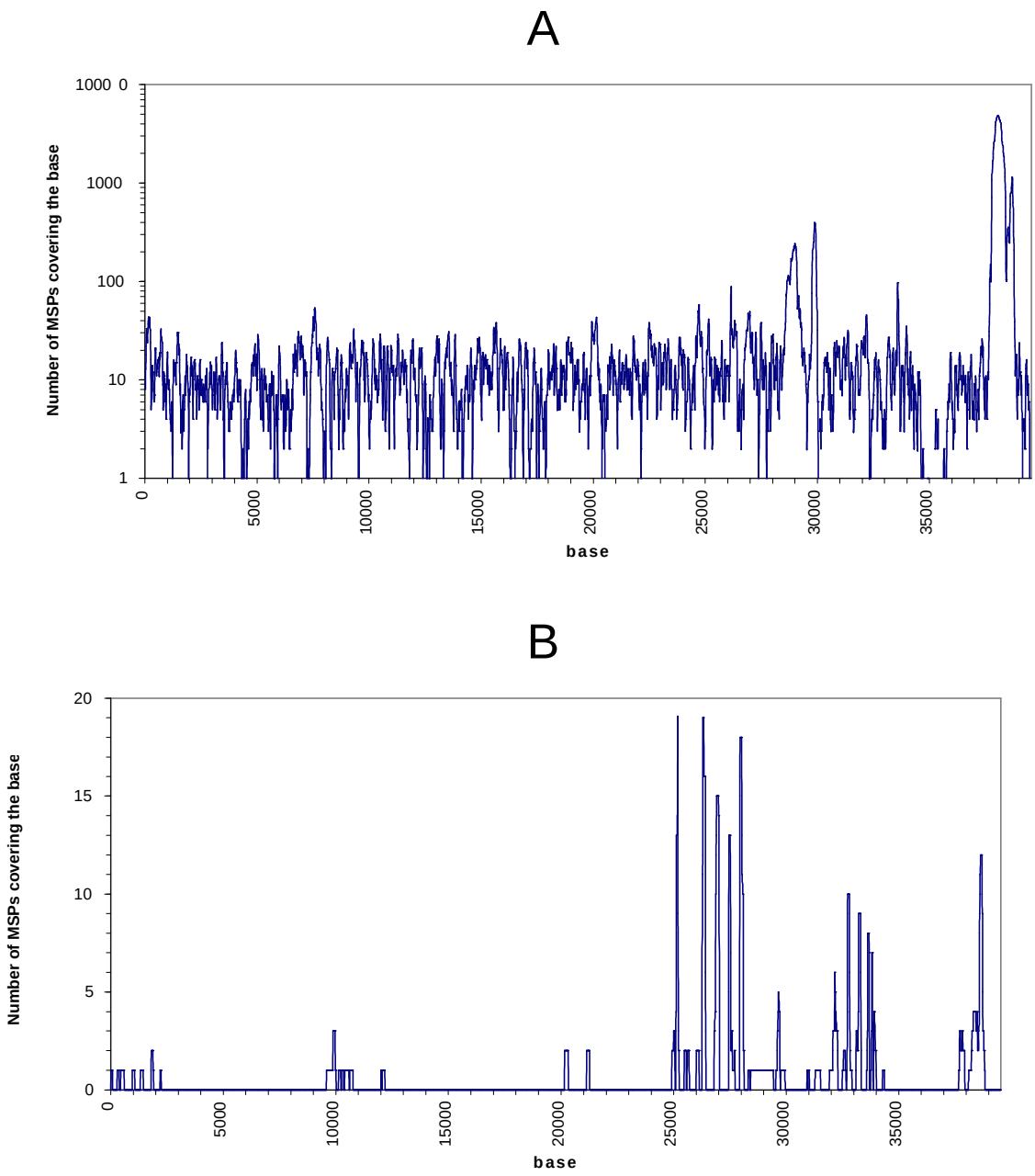


Figure 4.2. Histograms of MSP coverage on both strands of the 40 kbase DNA query sequence from the *C. elegans* cosmid ZK643 (A) before and (B) after MSPcrunch. MSPs were generated by Blastx as described in Methods. The number of MSPs was reduced from 119168 to 200 by MSPcrunch. The peak at 38000 is the result of a repetitive region which gives rise to very biased amino acid sequences (poly-G in the positive strand and poly-P in the negative) and the peaks at 29000 and 30000 have a strong bias for charged residues. Most matches to these regions were removed by the biased composition detection mechanism. The main significant homology in this cosmid is to a G-protein coupled receptor (ZK643.3), located between bases 25000 and 28000. Most matches are insignificant alone, but satisfy the adjacency criteria in MSPcrunch with neighbouring matches. There is also a motif conserved with DCMP deaminases at 32700-32800. Most other accepted matches are to predicted proteins from this cosmid.

Adjacency tests for multiple MSPs

MSPs that have an unbiased composition and score above a certain threshold are clear indicators of homology. This threshold is normally considered to be approximately 80-90 score units, using the BLOSUM62 score matrix. MSPs with scores in a region below this threshold, i.e. in the twilight zone, may have such low scores due to fragmentation caused by gaps in one of the sequences relative to the other. Since these gapped alignments are potentially real, a lower score threshold should be used for adjacent MSPs that can be concatenated within some limits of allowed overlaps and gaps in the query and subject sequences.

Figure 4.3 illustrates two cases of pairs of MSPs; the MSPs in the first pair A1/A2 is clearly consistent with a normal gapped alignment. The MSPs of the second pair B1/B2 could be caused by a gap in a true alignment, but this is less likely due to the long overlap. The wider the gaps and the more the MSPs overlap, the less likely are they to combine into a true gapped alignment. If B2 would overlap B1 with more than the length of B1, no gapped alignment could possibly join them together and the MSPs should be treated separately.

The definition of adjacency completely depends on the chosen parameters for how big the gaps and overlaps between MSP may be. BLAST itself has a consistent ordering check for multiple matches [Karlin and Altschul, 1993], which is used for calculating the combined probability. It is very conservative however, and only dismisses consistency if joining a pair of MSPs is impossible, and does not take the length of the gap into account. Still, this simple rule does reduce the noise level a fair amount.

To test adjacency between two MSPs MSP1 and MSP2, where MSP2 is C-terminal of MSP1 in the subject sequence, we define the following variables (see figure 4.3):

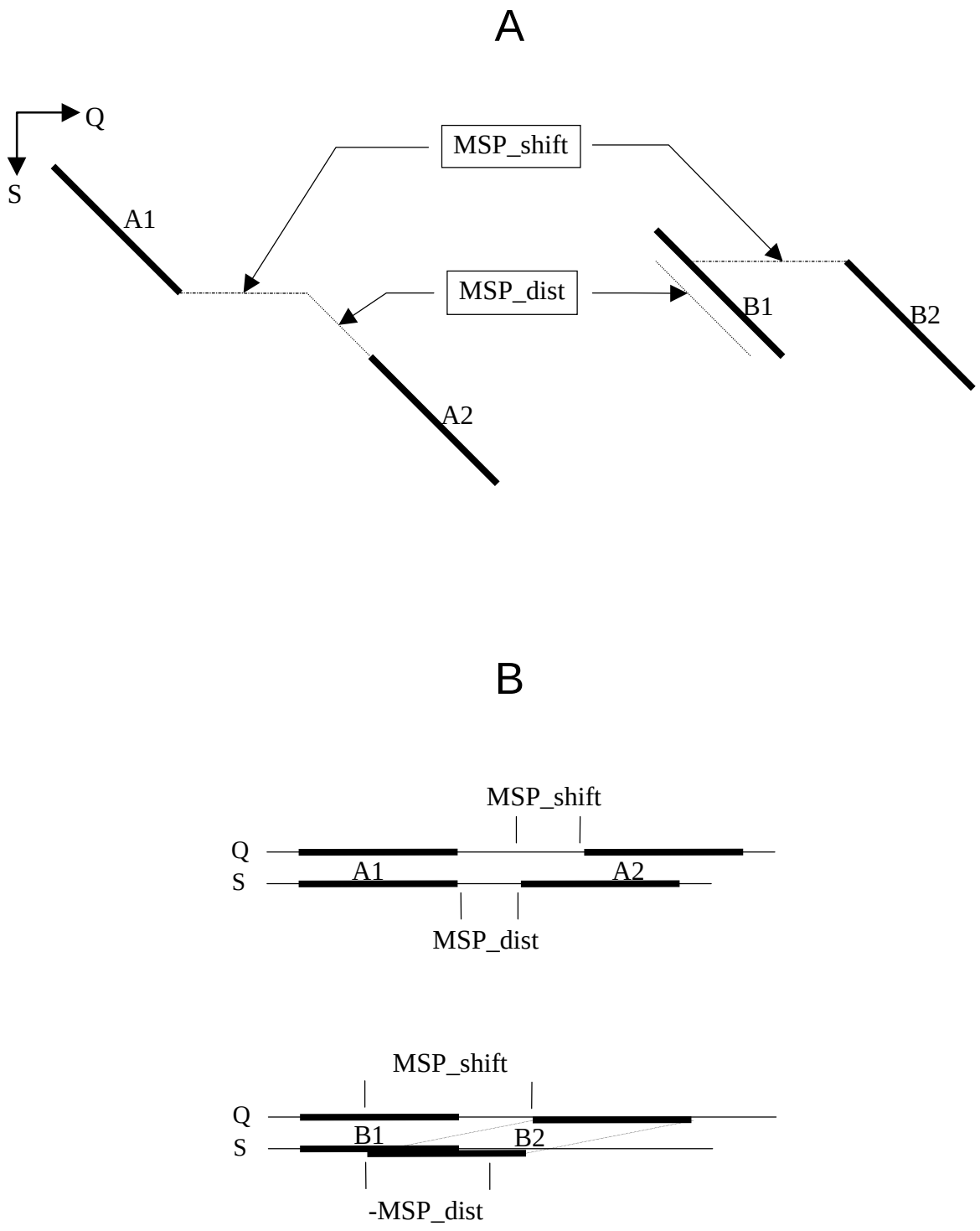


Figure 4.3. Diagrams of two cases of neighbouring MSPs in the sequences Q and S, for illustration of the MSPcrunch parameters MSP_dist and MSP_shift. The MSPs A1 and A2 do not overlap, while B1 and B2 do, yielding a negative MSP_dist value.

Query_gap = MSP2_QueryStart - MSP1_QueryEnd -1

Subject_gap = MSP2_SubjectStart - MSP1_SubjectEnd -1

MSP_dist = minimum (Query_gap, Subject_gap)

MSP_shift = | Query_gap - Subject_gap |

For Blastx, the query coordinates are converted to amino acid coordinates. Introns are essentially Query_gaps. If the Query_gap is larger than the Subject_gap but smaller than the intron limit, MSP_intron, potential introns are accommodated by setting Query_gap equal to Subject_gap before calculating MSP_dist and MSP_shift.

Consistency checking algorithm. For a truly consistent pair of MSPs, the values of MSP_dist and MSP_shift are located in a band that in order to distinguish true from false adjacency must become more narrow for lower scoring MSPs. The consistency checking algorithm is performed as follows:

```
For all pairs of MSPs between two sequences {
    Calculate MSP_dist and MSP_shift for the pair.
    Calculate the acceptance boundaries for MSP_dist and MSP_shift based on the
    lowest scoring MSP of the pair.
    If MSP_dist and MSP_shift are within the limits, mark the two MSPs as adjacent
    to each other.
}
For all MSPs between two sequences {
    If it scores above the twilight zone upper limit, accept it.
    Otherwise, reject unless it was found to be adjacent to another MSP.
}
```

Parameter estimation. The basis for adjacency analysis is that random spurious matches may occasionally end up adjacent to each other purely by chance, whereas real matches will do so very frequently. To investigate the range of the adjacency parameters in real homologies, matches to a G-protein coupled receptor and a carboxyl esterase were analysed manually to verify if they were true or false, based on the overall dotplot as reference. The MSPcrunch adjacency parameters for 78 MSPs that were verified to be correct are shown in figure 4.4. Most of the values are clustered near zero, but some low-scoring true neighbours are also present. In the lower region, there is overlap between noise and signal, but not in the upper region. A similar plot of randomly generated spurious matches would have a flat distribution along y axes in both plots, but tend to be strongly concentrated in the lower region on the x axes.

Allowed adjacency bands. We are now faced with the problem of devising a set of rules that will include as many as possible of the true MSPs, while as few as possible of the false MSPs. Usually it is more desirable to include some spurious matches, since removing all of them may reduce sensitivity to true matches.

The simplest rule for confirming adjacency would be constant distance and shift cutoffs for matches in the twilight zone. This would not work well, however, since it would be too permissive for low scoring MSPs and too restrictive for high-scoring ones. To accommodate for this, a gradual tightening of the permissive distance and shift cutoffs is needed. This could be done either linearly, or according to some function. For MSP_shift and the lower bound of MSP_dist (maximum allowed overlap), we found that the rule needs to be quite strict, even for MSPs in the upper twilight zone, so a linear curve was found adequate. For the upper bound of MSP_dist and MSP_intron, however, strictness is much more required in the lower twilight zone than in the upper. Attempts to use a linear allowance function were unsatisfactory because they were either too strict in the upper zone, or would go to zero too suddenly in the lower zone. A

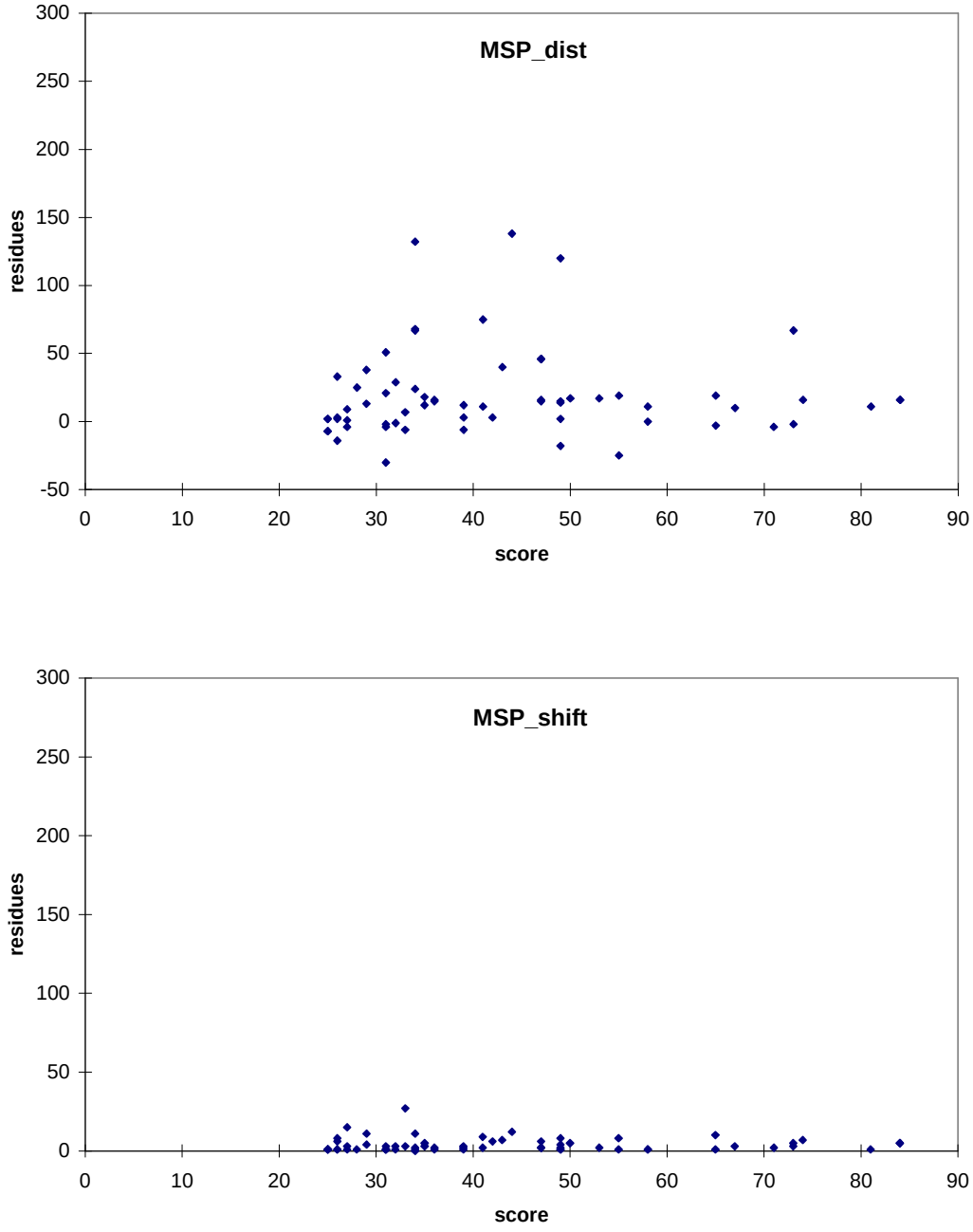


Figure 4.4 A sample of true MSP_dist and MSP_shift values from 78 MSPs that were manually verified. The score of the weakest MSP in each pair was used. Data from matches to the G-protein coupled receptor ZK643.3 (Swissprot YOW3_CAEEL P30650) and the carboxylesterase K07C11.4 (Wormpep CE07347).

quadratic function proved to gave a significant improvement over a linear. We did not pursue a more complex curve form, as we would only expect marginal improvements from this.

The bounds for the allowed bands that performed best were produced with these parameters. If the twilight zone is defined between scores *lower* and *upper*, we define the bounds for a parameter at these endpoints as distmax.lower, distmax.upper, distmin.lower, etc.. The shape of a curve is controlled by the .power exponents. A power of 1 gives a linear curve, while a higher power gives a curve which is relatively more stringent in the lower region. For scores in the twilight zone, this gives the following bounds:

MSP_dist < distmax.lower + (score - lower)^distmax.power/distmax.scale

MSP_dist > distmin.lower - (score - lower)^distmin.power/distmin.scale

MSP_shift < shiftmax.lower + (score - lower)^shiftmax.power/shiftmax.scale

MSP_intron < intronmax.lower + (score - lower)^intronmax.power/intronmax.scale

where

distmax.scale = (upper-lower)^distmax.power/(distmax.upper - distmax.lower)

distmin.scale = (upper-lower)^distmin.power/(distmin.lower - distmin.upper)

shiftmax.scale = (upper-lower)^shiftmax.power/(shiftmax,upper - shiftmax.lower)

intronmax.scale=(upper-lower)^intronmax.power/
(intronmax.upper-intronmax.lower)

(MSP_shift and MSP_intron are always positive so the lower bound is zero).

For Blastp (protein-protein comparison) we found the best definition of the twilight zone to be 25-75 (12.5-37.5 bits), while for Blastx, Tblastx and Tblastn (DNA-protein comparison) 35-75 (17.5-37.5 bits) due to the higher background noise levels. A default score of 75 was chosen as the upper limit of the twilight zone since only few spurious matches score above this value. For Blastn (DNA-DNA comparison) we set the zone by default to 70-140

(19-39 bits). We found that a linear behaviour in the distmin parameter was adequate, while a squared function for the other parameters gave better performance. The allowance bands that these functions yield are plotted graphically in figure 4.5.

To illustrate how these rules work in practice, an example is shown in figure 4.6. BLAST reports some false MSP together with the true MSPs. The false MSPs are not the lowest scoring ones, but since they lack adjacency, they can be ruled out as spurious twilight zone matches. The rule is stringent enough to allow confident use of lower scores than BLAST normally does. By default, BLAST sets the cutoff so that we expect 10 spurious matches to be reported, on a purely statistical basis. For this particular query sequence and database (ZK643.3 and swir10), this gives a lowest accepted MSP score (S2) of 33. As shown in figure 4.6c and d, lowering the BLAST score cutoff to 25 results in more true matches, but also more noise. However, thanks to the adjacency rules in MSPcrunch, the MSPs that are low-scoring due to gaps in the alignment can be separated from the spurious ones. Of course, all true MSPs are not always found. For instance, of the 78 manually verified MSPs in figure 4.4, 9 could not meet the adjacency criteria and were thus incorrectly missed by MSPcrunch.

Our results are generally applicable to any scoring scheme, but since the most popular scoring schemes, BLOSUM [Henikoff and Henikoff, 1992] and PAM [Dayhoff *et al.*, 1978], are in half bit units, we have chosen to express all scores in this unit. To convert them to other scoring schemes, they have to rescaled appropriately. Since the parameters have been estimated empirically to best suit the needs of interactive genome analysis, they may require adjustment for other purposes. All the twilight zone parameters can be changed on the command line. The above described algorithm was implemented in MSPcrunch version 2.0. Simpler and less effective algorithms have previously been described [Sonnhammer and Durbin, 1994a; Sonnhammer and Durbin, 1994b].

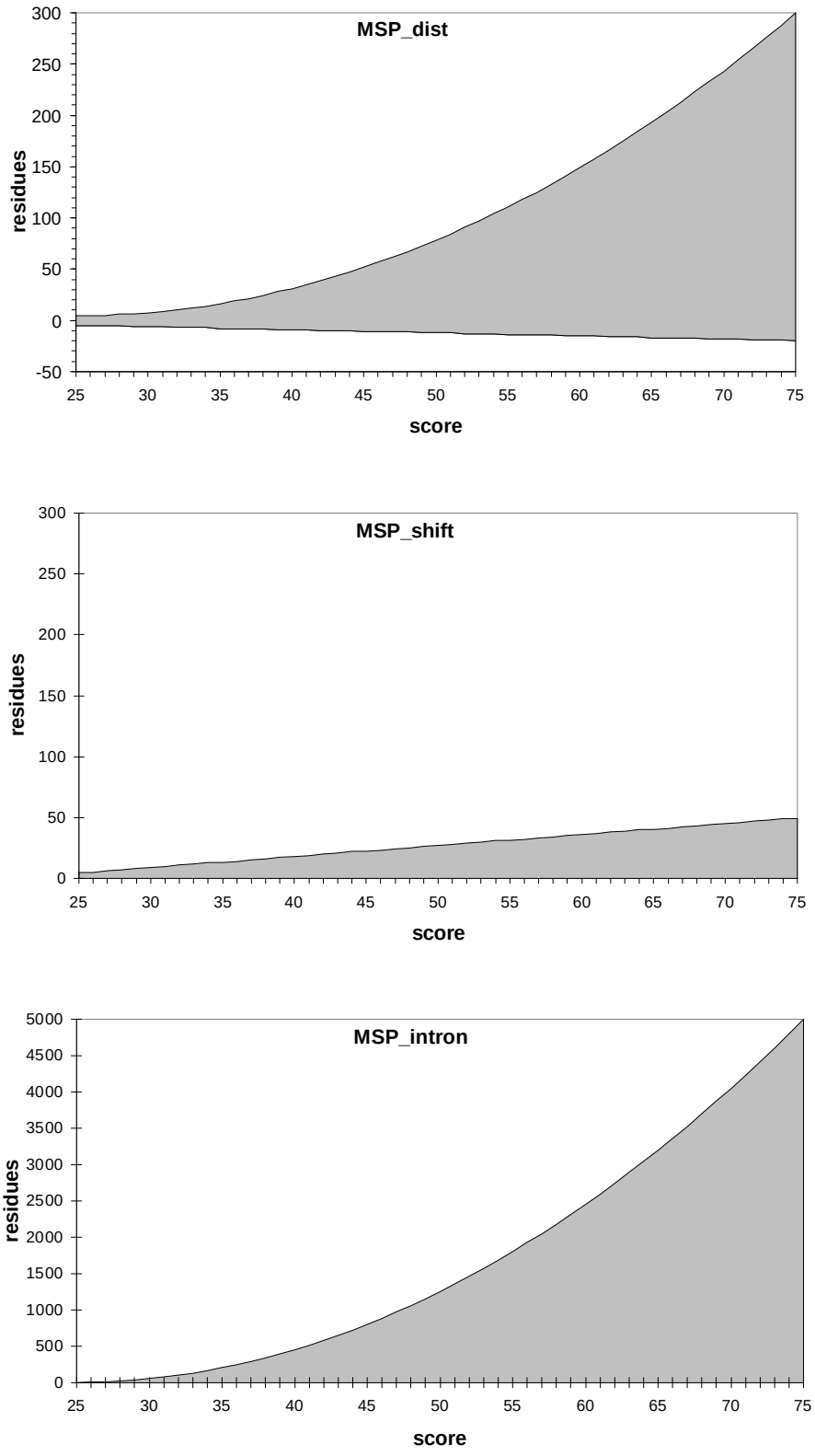


Figure 4.5. Bands of allowed MSPcrunch twilight zone adjacency parameters. If the parameters MSP_dist, MSP_shift and MSP_intron (Blastx only) between two MSPs fall within the shaded area, they are considered adjacent and will be accepted by MSPcrunch. The lower score of two MSPs is used.

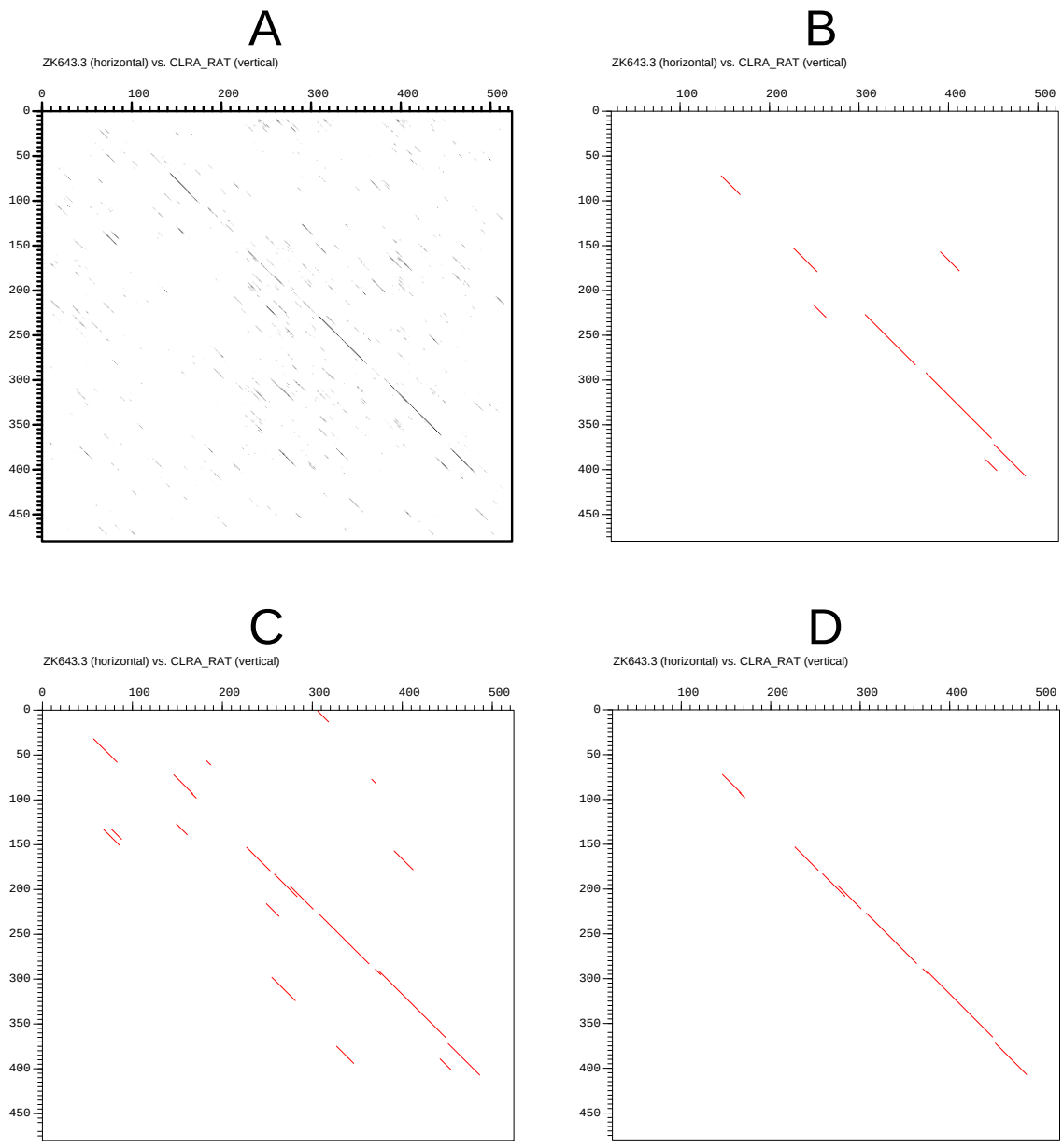


Figure 4.6. Dotplots illustrating the effect of MSPcrunch on the comparison of ZK643.3 (Swissprot YOW3_CAEEL P30650) with CLRA_RAT (Swissprot P32213). A. The full dotplot generated by Dotter (chapter 5) with a window size of 17. B. MSPs generated by Blastp using default parameters in a search against swir10 (S2=33). 5 true and 3 false MSPs are reported. C. MSPs generated by Blastp using an S2 cutoff of 25. 9 true and 11 false MSPs are reported. D. MSPs from C kept by MSPcrunch. All the false MSPs were rejected and all the true MSPs were kept, thus effectively enhancing both sensitivity and selectivity.

4.5 Displaying results

The recommended way to view MSPcrunch results is in Blixem. Nevertheless, MSPcrunch also supports a number of ASCII text output formats, that are useful for quick inspection of the results, and for exporting the data to other programs. Currently, the following output formats are available:

- A graphical ``Big Picture" schematic of the relevant matches, with one database sequence per line as shown in figure 4.7. This way one rapidly gets a good picture of which proteins match a certain region of the query. It is not unlike the Big Picture display in Blixem, except that matches that are considered adjacent are combined onto one line, and the sum of their scores is given as the score. The number of adjacent segments is also shown. Non-adjacent MSPs of the same sequences are displayed on separate lines. If an MSP with a positive expected score passes the biased composition filter, its score will be marked by an asterisk.

- Gapped pairwise alignments, as shown in figure 4.8a. This is achieved by simply concatenating adjacent MSPs. Only if BLAST reports overlapping MSPs will gaps appear as dashes in one sequence and residues in the other. For non-overlapping MSPs, BLAST does not provide the residues of the one sequence that spans the gap, and no attempt is made to retrieve it separately, since little information would be added by this. Such gaps will therefore be represented by dashes in both sequences. In practice, short gaps of a few residues can usually be reconstructed from the overlap, while long gaps can not. For a more concise report, any gap longer than ten will not be shown at full length, but will be truncated to ten dashes. The layout has been designed to be easy to read by humans as well as easy to parse by other programs.

- A detailed listing of each MSP, as shown figure 4.8b. Instead of sorting the MSPs in score order, which BLAST does, MSPcrunch sorts them by position from N to C-terminus in

the database sequence. This way a much better appreciation of the global alignment with gaps is gained, if it exists.

- In tabular format, with one line per MSP, for parsing by other programs. A variety of one-line formats are supported, one of which is shown in figure 4.9a. This format, called 'exblx' can be parsed directly by Blixem (chapter 3), which will fetch all the matching sequences, using Efetch (chapter 6). Another format, 'seqbl', contains all the information Blixem needs, including sequence data (figure 4.9b), and thus eliminates the sometimes time-consuming sequence fetching.

- In .ace format, for export to ACEDB, as shown in figure 4.9c. This is particularly useful for homology assisted gene prediction, since ACEDB includes an interactive gene prediction workbench coupled to Blixem, which integrates the display of predicted exons into the BLAST-based multiple sequence alignment (see chapter 3).

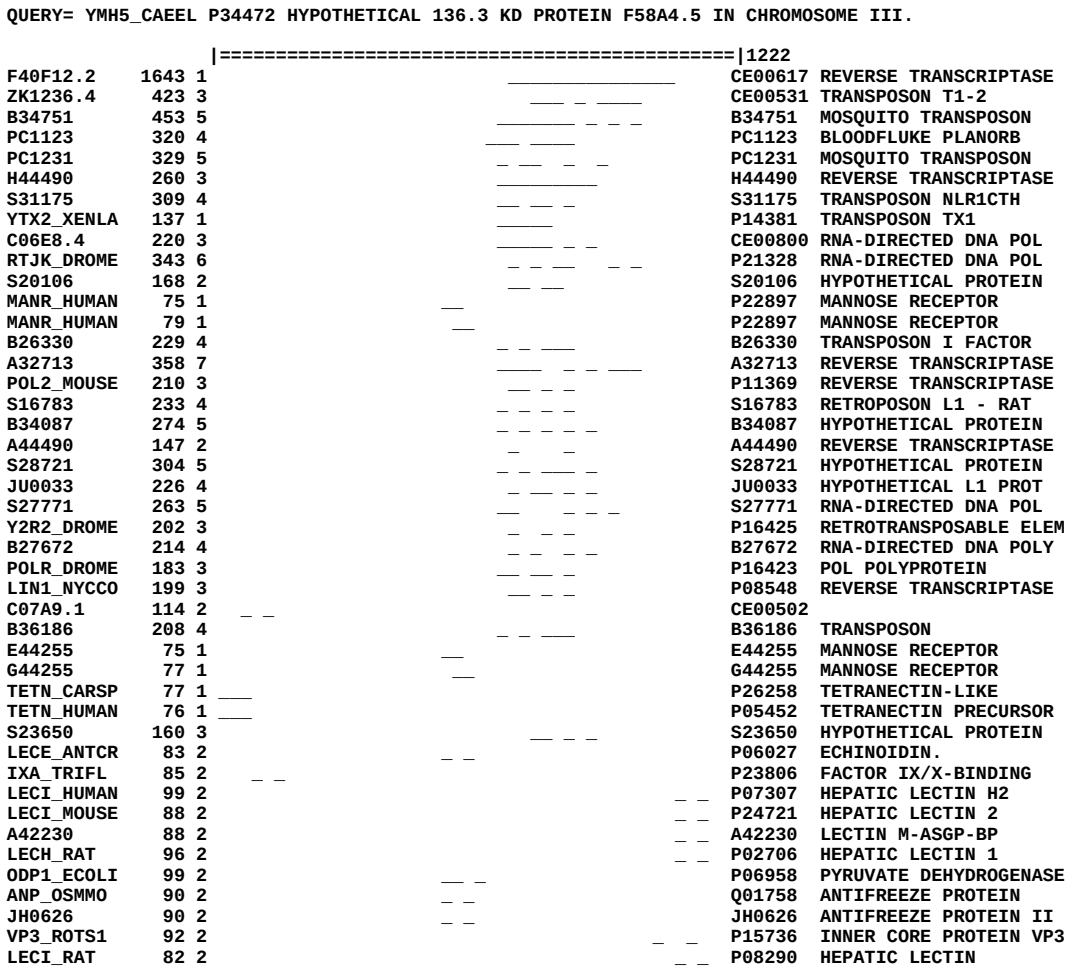


Figure 4.7. Example of the Big Picture display of MSPcrunched Blastp results. The sequence YMH5_CAEEL (Swissprot P34472) was searched against swir5. The domain organisation of this protein is C-type lectin (30-160), an acid-rich stretch (160-540), C-type lectin (540-620), Reverse Transcriptase (650-980) and C-type lectin (1080-1150). All matches to the acid-rich stretch were removed by the biased composition rule (β=0.8). In the original output from Blastp, the 62 highest-scoring MSPs were all biased composition matches, apart from the close relatives F40F12.2 and ZK1236.4 from the same chromosome. The columns are: Entry name, combined score, nr. of MSPs, schematic alignment, accession nr. and abbreviated description. Sequences from Wormpep include a dot and the ones from Swissprot an underscore. Other sequences are from PIR.

A

```
QUERY = ZK643.3  Length = 522
================================

> CLRA_RAT  P32213 CALCITONIN RECEPTOR A PRECURSOR (CT-R-A) (C1A).
------------------------------------------------------------------

Score= 96 (Sum of 2 contiguous HSPs), Identity= 48%

Query: ZK643.3 146 - 171 CPPTWDGWNCFDSATPGVVFKQ-CPNY
                         C  TWDGW C+D    GV+  Q CP+Y
Sbjct: CLRA_RAT 72 - 98  CNRTWDGWMCWDDTPAGVMSYQHCPDY


Score= 61 (Sum of 2 contiguous HSPs), Identity= 25%

Query: ZK643.3 227 - 283  LLTYSASVIFLIPAVFLLTLLRPIRCQ----LHRHLLISCLLYGAFYLITVSLFVVN
                         L+ +S S+  LI ++ +    + + CQ    LH+++ ++ +L     +I +   V N
Sbjct: CLRA_RAT 153 - 208 LVGHSMSIAALIASMGIFLFFKNLSCQ----LHKNMFLTYILNSIIIIIHLVEVVPN


Score= 322 (Sum of 5 contiguous HSPs), Identity= 33%

Query: ZK643.3 275 - 486  ITVSLFVVNDAPLSSQVFQNHLFCRLL-----RYLRLTNFTWMLAEAVYLWRLLHTAQHS
                         I + + +V   P    V ++ + C++L     +Y+   N+ WML E +YL  L+  A  +
Sbjct: CLRA_RAT 196 - 407 IIIIIHLVEVVPNGDLVRRDPISCKIL-----QYMMACNYFWMLCEGIYLHTLIVMAVFT

                         EGETLRSYKVICWGVPGVITVVYIFVRSL------CWIENSTVAWIEWMIITPSLLAMGV
                         E + LR Y ++ WG P V T+++   R++      CW+   T   + ++I  P + A+ V
                         EDQRLRWYYLLGWGFPIVPTIIHAITRAV------CWLSTET--HLLYIIHGPVMAALVV

                         NLLLLGLIVYILVKKLRCDPHLERIQYRKAVRGALMLIPVFGVQQLLTIYRFSN------
                         N   L  IV +LV K+R     E   Y KAV+  ++L+P+ G+Q ++  +R SN
                         NFFFLLNIVRVLVTKMRQTHEAEAYMYLKAVKATMVLVPLLGIQFVVFPWRPSN------

                         YQVTDQSLNGLQGMFVSFIVCYTNRSVVECVLKFWS
                         Y     SL   QG FV+ I C+ N  V   + + W+
                         YDYLMHSLIHFQGFFVATIYCFCNHEVQVTLKRQWA
```

Figure 4.8. MSPcrunch output of pairwise alignments. A (this page). Gapped alignments of the accepted MSPs in figure 4.6d. Note that only contigs of adjacent MSPs are aligned with gaps, separate contigs are not. The start and end coordinates of the entire contig is given at the start of each alignment, to make parsing easy. B (next page). Each MSP reported separately with MSP-specific information in N to C-terminal order. Matrix_expected and bias-ratio are referred to in the text as M_{exp} and β.

B

```
QUERY = ZK643.3  Length = 522
================================

> CLRA_RAT  P32213 CALCITONIN RECEPTOR A PRECURSOR (CT-R-A) (C1A).
------------------------------------------------------------------

Score= 68, Identity= 50%, Matrix_Expected= -20.8, bias-ratio= 1.04, Adjacency= Right

Query: ZK643.3 146 - 167 CPPTWDGWNCFDSATPGVVFKQ
                         C  TWDGW C+D    GV+  Q
Sbjct: CLRA_RAT 72 - 93  CNRTWDGWMCWDDTPAGVMSYQ


Score= 28, Identity= 43%, Matrix_Expected= -6.6, bias-ratio= 1.00, Adjacency= Left

Query: ZK643.3 165 - 171 FKQCPNY
                         ++ CP+Y
Sbjct: CLRA_RAT 92 - 98  YQHCPDY


Score= 34, Identity= 26%, Matrix_Expected= -25.5, bias-ratio= 0.89, Adjacency= Right

Query: ZK643.3 227 - 253  LLTYSASVIFLIPAVFLLTLLRPIRCQ
                          L+ +S S+  LI ++ +    + + CQ
Sbjct: CLRA_RAT 153 - 179 LVGHSMSIAALIASMGIFLFFKNLSCQ


Score= 27, Identity= 23%, Matrix_Expected= -24.6, bias-ratio= 0.83, Adjacency= Left

Query: ZK643.3 258 - 283  LHRHLLISCLLYGAFYLITVSLFVVN
                          LH+++ ++ +L     +I +   V N
Sbjct: CLRA_RAT 183 - 208 LHKNMFLTYILNSIIIIIHLVEVVPN


Score= 27, Identity= 22%, Matrix_Expected= -25.5, bias-ratio= 0.97, Adjacency= Right

Query: ZK643.3 275 - 301  ITVSLFVVNDAPLSSQVFQNHLFCRLL
                         I + + +V   P    V ++ + C++L
Sbjct: CLRA_RAT 196 - 222 IIIIIHLVEVVPNGDLVRRDPISCKIL


Score= 109, Identity= 35%, Matrix_Expected= -53.9, bias-ratio= 0.98, Adjacency= LeftRight

Query: ZK643.3 307 - 363  RYLRLTNFTWMLAEAVYLWRLLHTAQHSEGETLRSYKVICWGVPGVITVVYIFVRSL
                         +Y+   N+ WML E +YL  L+  A  +E + LR Y ++ WG P V T+++   R++
Sbjct: CLRA_RAT 227 - 283 QYMMACNYFWMLCEGIYLHTLIVMAVFTEDQRLRWYYLLGWGFPIVPTIIHAITRAV


Score= 27, Identity= 43%, Matrix_Expected= -6.6, bias-ratio= 0.98, Adjacency= Right

Query: ZK643.3 370 - 376  CWIENST
                         CW+   T
Sbjct: CLRA_RAT 289 - 295 CWLSTET


Score= 105, Identity= 35%, Matrix_Expected= -69.9, bias-ratio= 0.93, Adjacency= LeftRight

Query: ZK643.3 375 - 448  STVAWIEWMIITPSLLAMGVNLLLLGLIVYILVKKLRCDPHLERIQYRKAVRGALMLIPV
                         ST   + ++I  P + A+ VN   L  IV +LV K+R     E   Y KAV+  ++L+P+
Sbjct: CLRA_RAT 292 - 365 STETHLLYIIHGPVMAALVVNFFFLLNIVRVLVTKMRQTHEAEAYMYLKAVKATMVLVPL

                         FGVQQLLTIYRFSN
                          G+Q ++  +R SN
                         LGIQFVVFPWRPSN


Score= 54, Identity= 33%, Matrix_Expected= -34.0, bias-ratio= 0.99, Adjacency= Left

Query: ZK643.3 451 - 486  YQVTDQSLNGLQGMFVSFIVCYTNRSVVECVLKFWS
                         Y     SL   QG FV+ I C+ N  V   + + W+
Sbjct: CLRA_RAT 372 - 407 YDYLMHSLIHFQGFFVATIYCFCNHEVQVTLKRQWA
```

Figure 4.8b.

A

```
  68 (+1)     146     167    72    93 CLRA_RAT P32213 CALCITONIN RECEPTOR
  28 (+1)     165     171    92    98 CLRA_RAT P32213 CALCITONIN RECEPTOR
  34 (+1)     227     253   153   179 CLRA_RAT P32213 CALCITONIN RECEPTOR
  27 (+1)     258     283   183   208 CLRA_RAT P32213 CALCITONIN RECEPTOR
  27 (+1)     275     301   196   222 CLRA_RAT P32213 CALCITONIN RECEPTOR
```

B

```
# seqbl
# BLASTP
  68 (+1)     146     167    72    93 CLRA_RAT CNRTWDGWMCWDDTPAGVMSYQ
  28 (+1)     165     171    92    98 CLRA_RAT YQHCPDY
  34 (+1)     227     253   153   179 CLRA_RAT LVGHSMSIAALIASMGIFLFFKNLSCQ
  27 (+1)     258     283   183   208 CLRA_RAT LHKNMFLTYILNSIIIIIHLVEVVPN
  27 (+1)     275     301   196   222 CLRA_RAT IIIIIHLVEVVPNGDLVRRDPISCKIL
```

C

```
Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 68 146 167 72 93

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 68 72 93 146 167

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 28 165 171 92 98

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 28 92 98 165 171

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 34 227 253 153 179

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 34 153 179 227 253

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 27 258 283 183 208

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 27 183 208 258 283

Protein ZK643.3
Pep_homol CLRA_RAT BLASTP 27 275 301 196 222

Protein CLRA_RAT
Pep_homol ZK643.3 BLASTP 27 196 222 275 301
```

Figure 4.9. Examples of tabular output from MSPcrunch. A. The ‘exblx’ format, which contains score, frame, start and end coordinates and subject name and description of each MSP on one line. B. The ‘seqbl’ format, which is the same as exblx, except that it contains the sequence of the database entry instead of its description. Both formats are parsed by Blixem, but for exblx data, Efetch (chapter 6) must be installed to retrieve the sequences. C. The same data in .ace format, which is used to export the MSPs to ACEDB.

4.6 Discussion

An often heard criticism of using ungapped alignments is that distantly related proteins as a rule can only be aligned by inserting gaps. However, the regions which require gaps usually correspond to loops between secondary structure elements in the 3-dimensional structure, where the length of the loop may vary. The loop residues can often not be aligned structurally, which makes sequence alignments of these regions rather meaningless. Also, the results of algorithms that produce gapped alignment depend strongly on a somewhat arbitrary gap penalty. A further advantage of ungapped alignments is that repeated and shuffled domains in one sequence can be detected, something which is often compromised by programs that produce a gapped alignment.

One drawback of ungapped alignments is the difficulty of calculating an appropriate composite score for all MSPs with the same protein. Here we put the emphasis on making sure that a series of MSPs are truly consistent with a single gapped alignment. We then simply sum up the individual scores. The BLAST programs also calculate the probability of multiple matches by summing the individual scores of consistently ordered MSPs and correcting for the number of MSPs. However, their consistency criterion [Karlin and Altschul, 1993] is much weaker than our adjacency criteria and falsely high significance may arise from spurious hits that are not truly adjacent, especially those involving biased composition matches.

An additional practical problem with the probabilities calculated in BLAST is that they increase with the size of the database, because the expected number of spurious matches increases slowly as the database grows. However, the true match scores do not change, and because many of the new sequences are homologous to existing ones, the correction often overestimates the drop in significance. In any case it is more convenient to work with a measure of similarity that remains stable for a particular match. For these reasons we designed MSPcrunch to work only with the raw scores (which are log odds ratios).

The parameters used to deem two MSPs adjacent or not, MSP_dist and MSP_shift, were here used independently of each other. It might be worth considering treating them in a combined way, so that a large MSP_dist is more readily accepted if MSP_shift is small. We

have not found any such combinatorial rule that works satisfactorily in practice, and that make sense both biologically and statistically. Biologically one would expect a larger MSP_shift for a larger MSP_dist, but allowing this would increase the levels of accepted noise. There does not seem to exist a strong correlation between the parameters, and since it is also important for rules to be simple enough to understand, we have not pursued this any further.

The reduction of redundant results due to large protein families was achieved here by rejecting excess matches to a given region. A more subtle way of accomplishing this is to search a pre-clustered database. Instead of finding similarities to every member of the family, a single match would be found to the entire family, thus giving the relations to all other members of the family, not only the closest relatives. This is demonstrated in Part II, using the Pfam collection of protein families based on hidden Markov models. Whether searching a collection of aligned families always is more sensitive than pairwise comparison to all sequences is however not entirely clear. Sensitivity may also decrease if the family is not well defined, or if the query is much closer to one of the members than to the average of the family. Therefore, we have here pursued a higher quality of traditional single-sequence database searching, which most likely will remain an important tool complementary to family-based searching techniques.

The system described here is similar to GeneQuiz [Scharf *et al.*, 1994] in that it performs many sequence analysis tasks automatically. MSPcrunch however is primarily concerned with the proper treatment of similarities along large DNA queries, for which a solution of the multi-domain problem is needed. Together with Blixem and Dotter, integrated in ACEDB, MSPcrunch is part of a sequence analysis workbench. This workbench also has different goals than GeneQuiz, since it was designed to analyse the DNA sequence and improve the quality of exon/intron predictions using sequence similarity, as well as being an annotation tool. Given the complexity of the gene prediction process in higher eukaryotic genomes, we don't envisage a fully automatic system for this in the near future.

Many of the features in MSPcrunch have been made partly obsolete by subsequent improvements in the BLAST software. Especially the sensitivity for multiple weak matches

was improved significantly in BLAST 1.4. Our biased composition check is currently being tested as an option in BLAST, and we hope that other features will be included in the future too. Performing the filtering process during the search phase would reduce the computational load.

MSPcrunch, Seqsplit, Blastunsplit are available by anonymous FTP from ftp.sanger.ac.uk in the directory /pub/MSPcrunch.

5. Dotter: A dot-matrix program with dynamic threshold control suited for genomic DNA and protein sequence analysis

5.1 Summary

Graphical dot-matrix plots can provide the most complete and detailed comparison of two sequences. Presented here is Dotter, a dot-plot program for X-windows which can compare DNA or protein sequences, and also DNA versus protein.

The main novel feature of Dotter is that the user can vary the stringency cutoffs interactively, so that the dot-matrix only needs to be calculated once. This is possible thanks to a "Greyramp tool" that was developed to change the displayed stringency of the matrix by dynamically changing the greyscale rendering of the dots. The Greyramp tool allows the user to interactively change the lower and upper score limit for the greyscale rendering. This allows exploration of the separation between signal and noise, and fine-grained visualisation of different score levels in the dot-matrix.

Other useful features are dot-matrix compression, mouse-controlled zooming, sequence alignment display and saving/loading of dot-matrices. Since the matrix only has to be calculated once and since the algorithm is fast and linear in space, Dotter is practical to use even for sequences as long as cosmids.

Dotter was integrated in the gene-modelling module of the genomic database system ACEDB. This was done via the homology viewer Blixem in a way that also allows segments from the BLAST suite of searching programs to be superimposed on top of the full dot-matrix. This feature can also be used for very quick finding of the strongest matches. As examples, we analyse a *C. elegans* cosmid with several tandem repeat families, and illustrate how Dotter can improve gene modelling.

5.2 Introduction

Ever since the introduction of graphical dot-matrix plots to sequence analysis [Fitch, 1969] [Gibbs and McIntyre, 1970], they have been among the most popular methods for analysing similarity between two sequences, particularly for gaining a good picture of the similarity between repeated domains.

The original dot-plot concept of drawing one sequence along the horizontal axis and the other along the vertical axis of a coordinate system, and drawing a dot where two residues match has essentially stayed the same. Regions of similarity between the sequences will result in a diagonal row of dots, whereas spurious matches give rise to a background of single dots. A standard filtering technique to reduce the noise is to apply a window along the diagonals and only draw a dot in the centre of the window if the sum of all dots in the window exceeds some score threshold.

One problem is that the optimal threshold for drawing a dot is hard to guess *a priori*. Poor choice of threshold may result in dot-plots either too full of noise or lacking the relevant diagonals. This can be frustrating, since changing the threshold usually requires recalculation of the entire dot-plot, which often is very time consuming. Estimating the threshold by probabilistic methods [McLachlan, 1971] [McLachlan, 1983] [McLachlan and Boswell, 1985] [Reich and Meiske, 1987] [Argos, 1987] can be of use for finding the approximate border region between signal and noise, but still usually requires recalculation of the dot-matrix at different score levels. Inspecting the dot-plot at different thresholds is very informative since it gives a better picture of the strength of a diagonal relative to the noise and other [Staden, 1982]. Dotter was created to make this interactive aspect of dot-plots more powerful than in previous implementations.

Improvements on the classical single-bit dot-plot (where dots are either on or off) have been to encode the score of a dot by colours [Maizel and Lenk, 1981] [Reisner and Bucholtz, 1988] [Zuker, 1991] or by lines of varying thickness [Argos, 1987]. However, none of these programs can plot more than 16 different colours or shapes, and since they can not be modi-

fied dynamically to other thresholds, they do not eliminate the need for recalculation if another stringency rendering is required.

Modern graphics hardware offers new possibilities for addressing this problem. Dotter allows the user to set score thresholds dynamically *after* the dot-matrix has been calculated, using the X-windows system for changing screen colours on 8-bit displays. This is done by a mouse-controlled "Greyramp" tool which lets the user modify two score thresholds which can either be used as a strict “all or nothing” cutoff, or as a smooth rendering of many different score levels at once. Dots scoring below the first threshold are invisible and dots scoring above the second threshold get the maximum intensity while dots scoring between the thresholds are rendered with an intensity proportional to their score. Employing 128 different greyscale colours ensures a smooth range of intensity values.

Computationally, the main problem with dot-plots is that the execution time is proportional to the product of the lengths of the sequences, which makes long sequences very time consuming. This problem has been attacked by heuristic approaches [Pearson and Lipman, 1988] [Schwartz *et al.*, 1991] and trees combined with heuristics [Lefevre and Ikeda, 1994]. Such techniques can give improvements in speed of several orders of magnitude, at the cost of generating a not entirely correct dot-matrix. For long sequences, where an overview of the strongest matches is of main interest, such approximations may be acceptable, but for detailed analysis of weak similarities the full matrix still needs to be calculated. We recognise the usefulness of such fast methods and have therefore equipped Dotter with the ability to also read in matches produced by the BLAST suite [Altschul *et al.*, 1990]. Displaying ungapped matches from BLAST is also informative since it shows the extent of high-scoring segments.

Dotter is a versatile tool for dot-matrix comparisons of DNA and protein sequences. It can produce dot-plots for DNA vs. DNA, protein vs. protein, and DNA vs. Protein. For DNA, it can draw the reverse complement diagonals in the same dot-matrix as the forward ones. For DNA vs. protein, it translates the DNA sequence in the three forward frames and draws them all in the same dot-matrix. All modes feature tools to inspect the sequence alignment of any diagonal.

5.3 Methods

Generating the dot-matrix

Here, the dot-matrix will not simply contain a zero or a one (one bit) for each dot as in the traditional dot-plot, but a value between 0 and 255 (8 bits = one byte). The dot-matrix thus contains scores, averaged over a chosen window-span, but we prefer not to call it a "score matrix" to avoid confusion with the well-known pairwise score matrices, such as PAM120 and BLOSUM62. The dot-matrix needs only to be calculated once for a given window-span.

For maximum speed, we precalculate score vectors for every possible symbol in the vertical sequence along the horizontal sequence [Karreman, 1992]. For DNA, this requires 4+1 score vectors (1 extra for unknown symbols), and for protein 20+2+1 (20 amino acids, 2 for ambiguity symbols and 1 for unknowns). This makes execution faster since the few score vectors only have to be calculated once and are later added to and removed from the sliding window-sums. The window-sums are calculated for consecutive windows along the diagonal in a sliding manner by simply adding the next score and subtracting the last score inside the window. Instead of calculating the window-sums for one diagonal at a time however, we keep a horizontal vector of all window-sums and add and subtract the precalculated score vectors row by row.

The following pseudocode outlines the algorithm. The score vectors are assumed to be initialised with the scores from the pairwise score matrix used.

```
integers    N,                      // Length of horizontal sequence
            M,                      // Length of vertical sequence
            α,                      // Size of alphabet
            W                       // Span of sliding window

vectors     scoreVec[1..α+1][1..N], // The score vectors
            newsum[1..N],           // Window-sum vector 1
            oldsum[1..N],           // Window-sum vector 2
```

```
            zeroVec[1..N],      // Vector of zeros
            symbVec[1..M]       // Symbols in vertical sequence

pointers    addVec,             // Pointer to scoreVec to be added
            delVec,             // Pointer to scoreVec to be subtracted
            tmp                 // Temporary pointer


for i ← 1 to N do
{
   tmp ← oldsum
   oldsum ← newsum
   newsum ← tmp

   addVec ← scoreVec[symbVec[i]]
   if i > W then delVec ← scoreVec[symbVec[i-W]]
   else delVec ← zeroVec

   newsum[1] ← addVec[1]
   for j ← 2 to W do
      newsum[j] ← oldsum[j-1] + addVec[j]
   for j ← W+1 to M do
   {
      newsum[j] ← oldsum[j-1] + addVec[j] - delVec[j-W]
      if newsum[j] > 0 and i > W then
         dot-matrix[i-W/2][j-W/2] ← newsum[j]/W
   }
}
```

It is worth noting that the main operations are all vector additions and subtractions, which would make the program M times faster on an architecture allowing simultaneous vector operations. The above algorithm gives a performance of 5.7 million dots per second on a DEC Alpha AXP 3000/700. This means that a cosmid sequence of 40.000 basepairs can be compared against itself in about 4.5 minutes. Other programs have reported speeds of 0.0005 [Pustell and Kafatos, 1982], 0.1 [McLachlan, 1983], 0.005 [Argos, 1987] and 0.08 [Karreman, 1992] million dots/second, albeit on slower hardware. The only program we could benchmark on the same hardware as Dotter was DIAGON [Staden, 1982] which runs at 0.46 million dots/second.

The total memory usage of Dotter is $(\alpha+4)N + 2M$ plus the dot-matrix itself (1 byte/dot). The memory usage of the dot-matrix is not O(NM) since if NM is large, we only keep a

compressed matrix. Dotter calculates the compression factor based on a user-settable option S, the maximum memory usage of the dot-matrix (default 0.5 Mb). If the product NM is greater than S, we let each pixel represent a TxT region of the full matrix, where T is the smallest integer that satisfies $NM/T^2 < S$. Although all the values in the full matrix are calculated, only the maximum value in each TxT square is kept [Pustell and Kafatos, 1982]. This process increases the background noise, but this is readily compensated for by raising the thresholds in the Greyramp tool (see below). If either N or M is greater than the number of horizontal or vertical pixels of the screen, scroll bars will appear to let the user pan through the dot-matrix.

By default, Dotter sets the window-span to the length of the expected Maximal Segment Pair (MSP), which is calculated for the given sequences and score matrix the following way. Karlin and Altschul showed that for two sequences of length *n* and *m*, the MSP score $M(nm)$ has a distribution approximated by $P(M(nm) - (\ln nm)/\lambda > x) \approx 1 - \exp\{-Ke^{-\lambda x}\}$, and provided a method for solving K and λ [Karlin and Altschul, 1990]. The mode of this distribution, or the *expected MSP score* is then $(\ln nm + \ln K)/\lambda$. The expected *score per residue* in an MSP is $R = \Sigma\, q_{ij}S_{ij}$; $q_{ij} = p_i p_j \exp\{\lambda S_{ij}\}$, where p_i and p_j are the symbol frequencies in the sequences. By dividing the expected MSP score with the expected score per residue we obtain a simple approximation to the expected MSP length:

$$\frac{\frac{\ln nm + \ln K}{\lambda}}{\sum p_i p_j e^{\lambda S_{ij}} S_{ij}}$$

For typical sequences and score matrices such as BLOSUM62 for protein and {match = +5; mismatch = -4} for DNA, this usually gives a window-span of about 25 residues. If the above method gives an undesired window-span or fails because λ is undefined for the chosen scoring scheme, it can also be set manually. Because we are interested in local similarities we set *n* and *m* to a constant value of 100. This makes the noise density independent of the sequence lengths.

Dotter can also run in batch mode. In both interactive and batch mode, the dot-matrix and all used parameters can be saved to file and be inspected later. The ability to load dot-matrices from file also makes it possible to generate dot-matrices with other programs and read them in Dotter for interactive inspection. See the World Wide Web address below for details of the format.

Visualising the dot-matrix with the Greyramp tool

The Greyramp tool was designed to enhance the visualisation of greyscale images, particularly images with a delicately balanced mix of noise and signal. The simplest form of displaying the score of a dot as a greyscale is to let the intensity be directly proportional to the value of the dot. The Greyramp tool provides two additional features: A min cutoff score, below which all dots get minimum intensity, and a max cutoff score, above which all dots get maximum intensity. For dots scoring between min and max, the dot intensity is linearly proportional to the score. The score shown in the Greyramp tool is the score per residue, i.e. the total score of the sliding window divided by the window-span, multiplied by a scaling factor to use the pixel intensity range 0-255 optimally. By setting this scale factor to $256/5R$, where R is the expected score per residue in an MSP (see above), we place the expected noise level at a fifth (51.2) of the intensity range and thereby make the significance of the pixel intensities roughly the same for different scoring schemes. By starting the Greyramp tool with min=40 and max=100, the top of the noise will be just visible, and all scores above twice the expected significant level will be at maximum intensity. Empirically this gives reasonable starting points.

The min and max cutoffs can be changed dynamically and can be controlled independently by point-and-drag actions with the mouse on the little triangles seen in figure 5.1. By dragging the little box in the middle between min and max they are modified simultaneously while keeping the difference between them constant. Minimum intensity is usually white and maximum black, but this can be reversed by the "swap" function. For any setting of the min and max thresholds, the rendered dot-matrix can be printed out on a postscript printer.

Dotter changes the greyscales on the screen by modifying the colourmap cells of 8-bit X-windows displays, which are the most common. Since the colourmap cells are not needed on 24-bit graphics, Dotter will not work on such displays. Dotter uses 128 colourmap cells, which may be shared by simultaneous Dotter jobs on the same display.

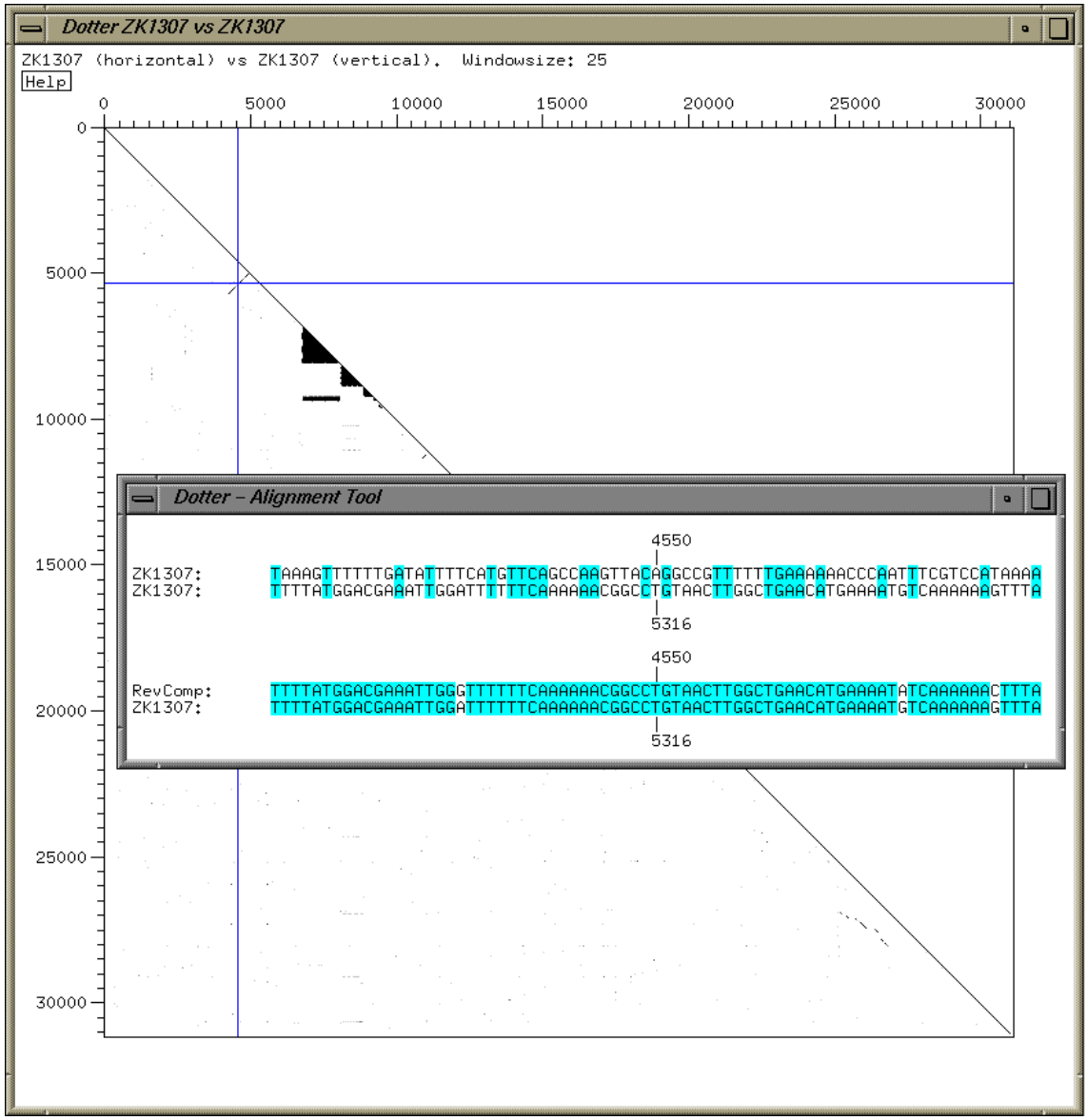


Figure 5.1a. Dot-matrix analysis of the *C. elegans* cosmid (ZK1307, EMBL Z47358) with Dotter. The entire cosmid is compared to itself, with the forward and reverse direction diagonals superimposed. Only half the dot-matrix is calculated since the other half is an identical mirror image. Features that can be seen at this level are an inverted repeat at 4000-6000, a region containing a multitude of small tandem direct and inverted repeats at 6700-9500 and a duplicated gene repeat at 25000-28000. The alignment in both directions at the position of the crosshair is shown in the Alignment tool window in the middle.

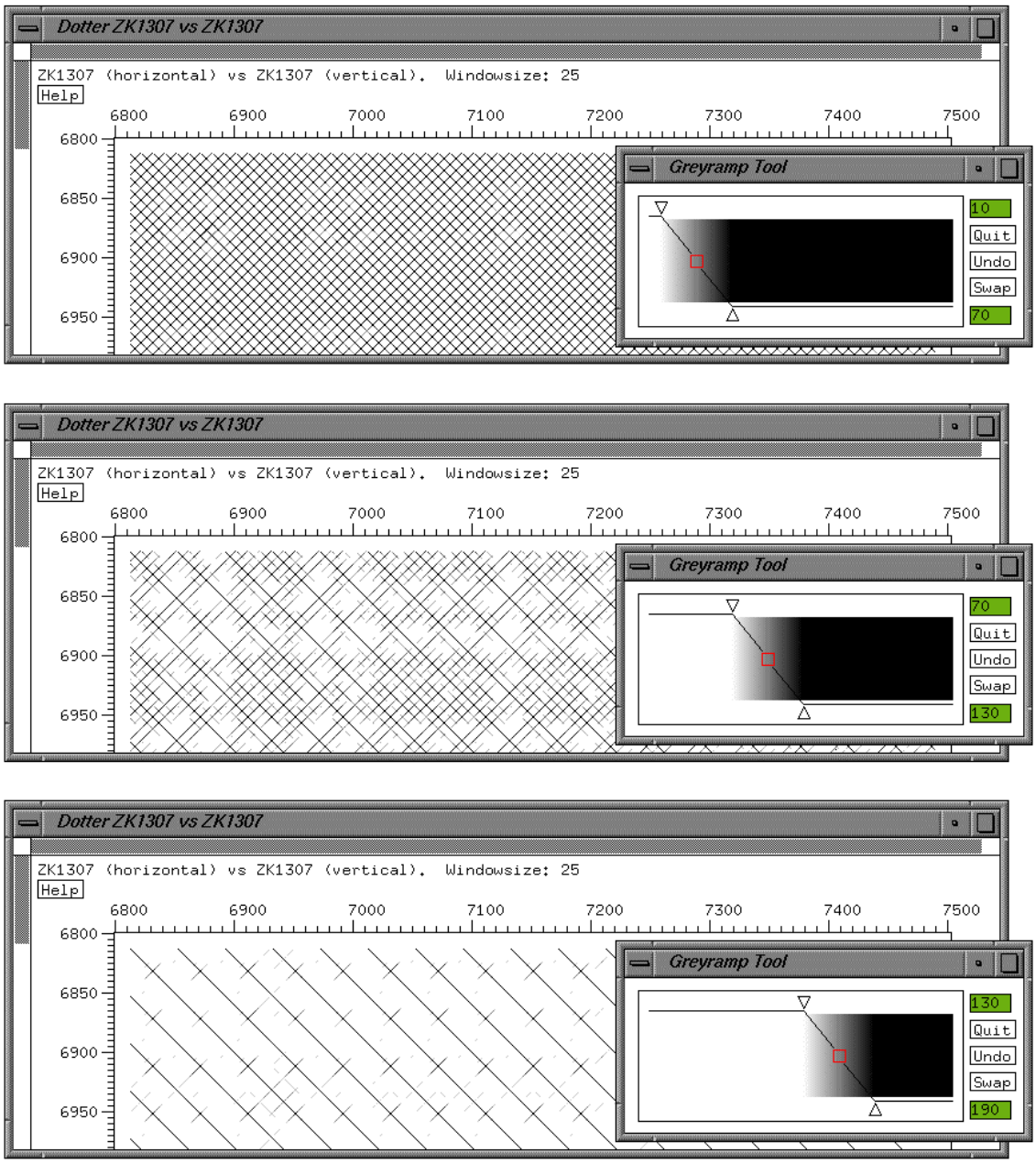


Figure 5.1b-c. (b) Zoomed in detail of (a) in a tandem repeat region of about 100 10 bp repeats between 6700 and 8100. The Greyramp Tool is used to view the dot-matrix at different stringencies. The pixel values are 50 times the average residue-score in the window, meaning that a 100% identical match would score 250, given the scoring scheme of +5 for matches and -4 for mismatches. Any dot scoring below the *min* threshold of 10 will be invisible, dots above the *max* threshold of 70 will be completely black, and dots in between will be drawn in a greyscale proportional to their score. (c) If the rendering thresholds are moved up to 70-130, it becomes clear that every 4 of the 10 bp repeats have stronger similarity with each other, suggesting a super-structure repeat unit of 40 bp. (d) Moving the thresholds up to 130-190 shows only the 40 bp repeat structure in the forward direction and only faint inverted diagonals, also with a pitch of 40 bp. The calculation of (a) took 170 seconds and of (b), (c) and (d), which are different renderings of one dot-matrix, 0.1 seconds.

The crosshair and Alignment tool

A crosshair can be moved either with the mouse or cursor keys around the dot-plot. The extent of a diagonal can be found either by reading the coordinates next to the crosshair or from the rulers on the axes. The sequence alignment of a diagonal can be displayed by moving the crosshair onto it and launching the Alignment tool from Dotter's main menu (right mouse button). The Alignment tool displays a residue by residue alignment of the two sequences corresponding to the diagonal around the crosshair. Identical matches are highlighted in bright blue and conservative substitutions in dark blue. If both sequences are DNA, two alignments are possible: of the original sequences and of the reverse complement of the horizontal sequence to the vertical sequence. The two alignments can be shown simultaneously as in figure 5.1. If the horizontal sequence is DNA and the vertical is protein, the three forward frames are translated and superimposed in the dot-matrix, keeping the maximum value in each pixel as described above for compressed matrices. The only way of telling which frame caused a diagonal is to use the Alignment tool, which displays all three reading frames aligned to the protein sequence (figure 5.2).

If the nature of some segments in one or both of the sequences is already known, Dotter can enhance the analysis by displaying such segments as coloured boxes along the border of the dot-plot, as in figure 5.3. The coloured segments seen in the border are read in from a simple data file with one line per segment. The format is: sequence (1=horizontal, 2=vertical), start, end, colour, annotation. See the World Wide Web address below for more details. In combination with the crosshair, the coloured boxes are easy to relate to a particular diagonal.

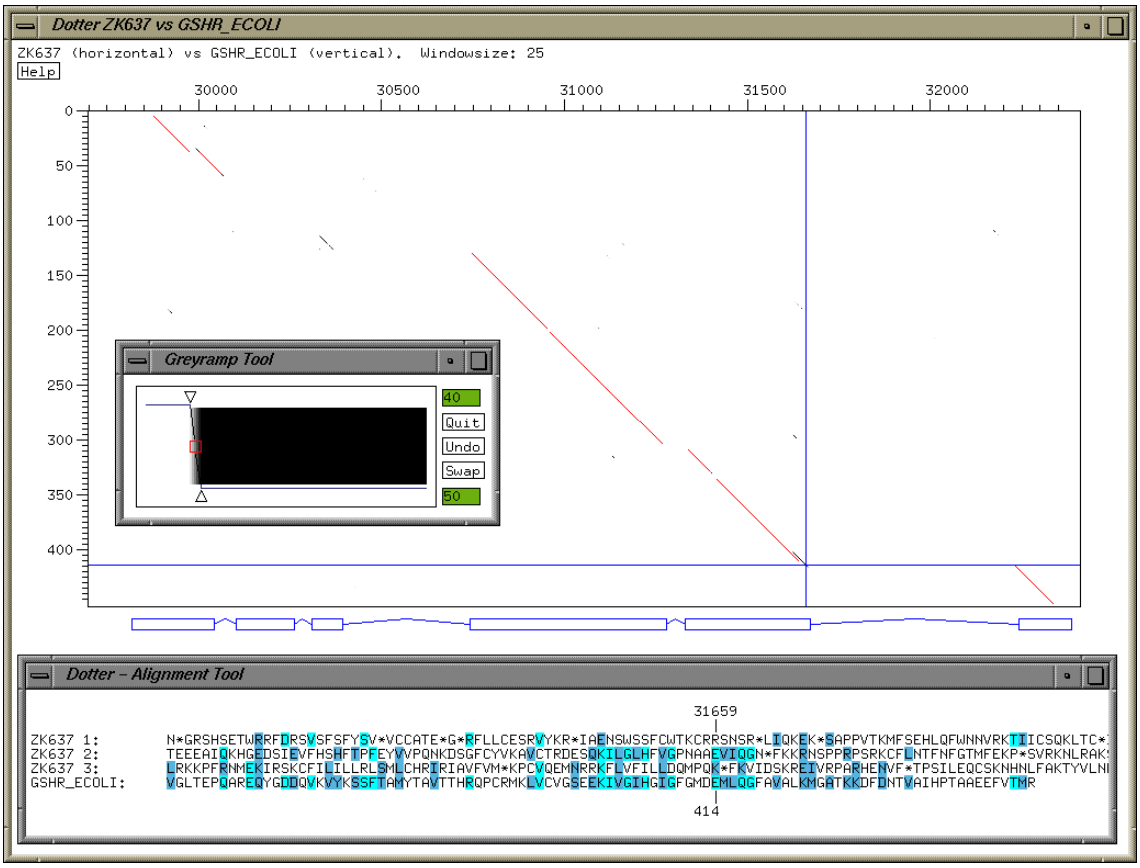


Figure 5.2. Dotter plot of DNA vs. protein with gene predictions from ACEDB. Shown here is a stretch of genomic DNA from the *C. elegans* cosmid ZK637 (EMBL Z11115) compared to the protein glutathione reductase from *E. coli* (Swissprot P06715). The gene prediction (ZK637.10) was made in ACEDB, but some exons have only very weak homology. Matches found by BLAST/MSPcrunch (chapter 4) are superimposed in the dot-plot as red lines. The match at exon 3 was too weak to be reported by BLAST/MSPcrunch, but it is visible in the dot-plot. Also, the BLAST match at the end of exon 5 was extended past an insertion, whereas the dot-plot shows the correct diagonal. The Alignment tool shows the alignment of the three translated forward frames of ZK637 with GSHR_ECOLI at the end of exon 5 (see crosshair position). Frame 2 contains the match missed by BLAST. The calculation took 0.6 seconds.

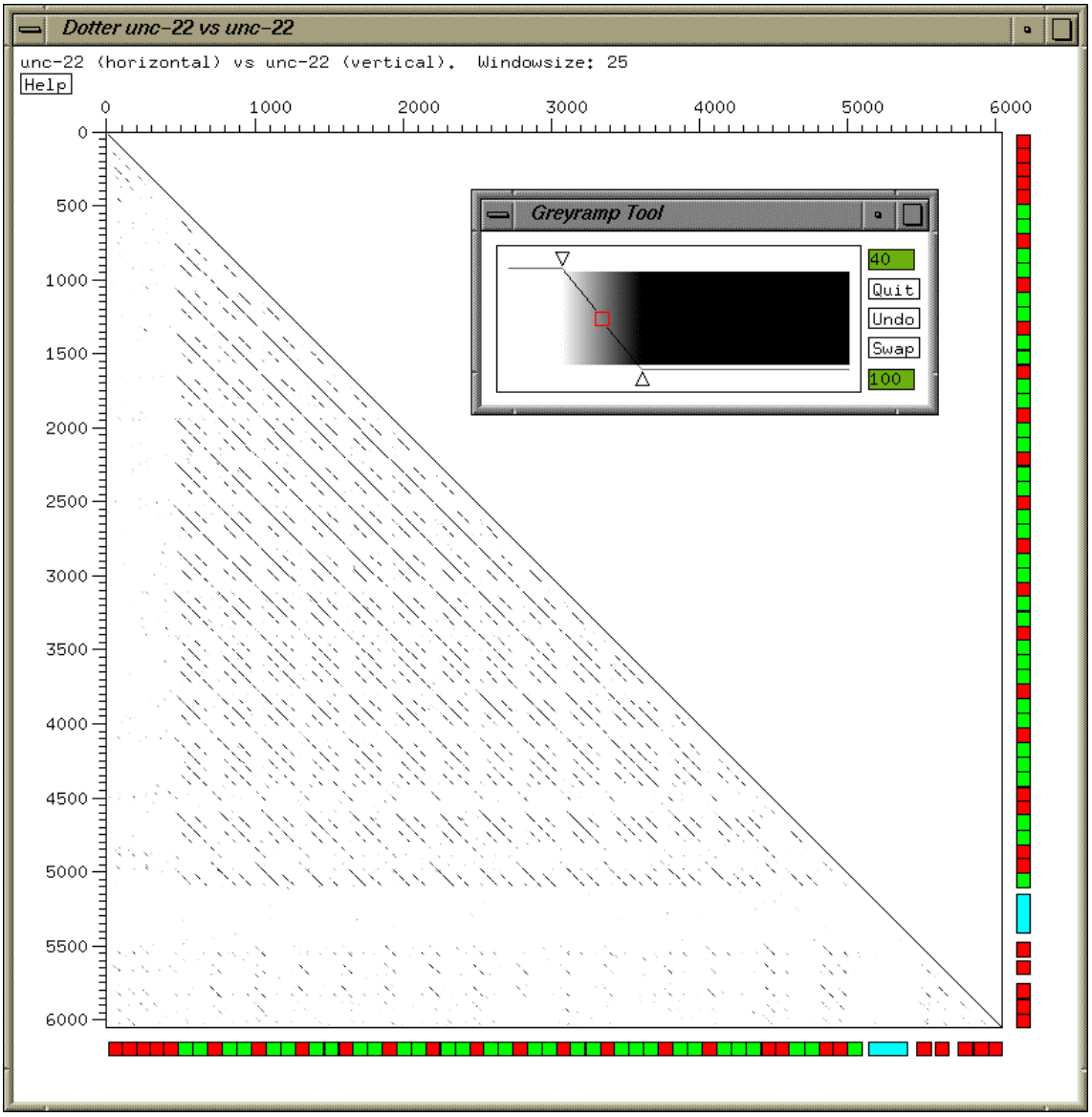


Figure 5.3. Analysis of a highly repetitive protein with symbolic domain annotation. The protein UNC-22, or twitchin (PIR S07571) from *C. elegans* is compared to itself. Pixel values are 50 times the average residue-score in the window. The colours of the segments are: green = fibronectin type III domain (FN3); red = immunoglobulin domain (IG); Blue = kinase domain. It is clear that all the FN3 domains are much more closely related to each other than the IG domains, and that the FN3-FN3-IG cassettes between 1000 and 3000 are more closely related than the other ones. The calculation took 3 seconds.

Zooming in

It is possible to zoom in to any region in a compressed dot-matrix by dragging with the middle mouse button to delimit a rectangle, or with exact coordinates via a dialogue window. A new Dotter job will then be spawned for the selected region only. The parent Dotter job will not be superseded but will remain intact on the screen. The two dot-plots will be independent of each other so that either can be killed without affecting the other one. Since all simultaneous Dotter jobs share the same colourmaps, any Greyramp tool will control the greyscale rendering of all active dot-plots.

Displaying high-scoring segments

Calculating the full dot-matrix, as described above, has two drawbacks: it is slow for very long sequences, and it does not display the maximum high-scoring extent of the diagonals. Sometimes it is informative to try to extend a diagonal in both directions until the total score doesn't increase further. The ungapped alignment giving the maximum score is called a high-scoring segment pair (HSP). The BLAST programs [Altschul *et al.*, 1990] search for HSPs in a fast, heuristic fashion. Instead of replicating the BLAST algorithm, Dotter simply reads in HSPs reported by BLAST and draws them in the dot-plot as in figure 5.2, similarly to PLFASTA [Pearson and Lipman, 1988] for FASTA output. Here it is accomplished via the BLAST output viewer Blixem (chapter 3), which constructs a multiple alignment of HSPs reported by BLAST and displays it graphically in a scrollable window. The advantage of this is that Blixem first can give an overview of all sequences that match a given query. The most interesting homologies can then be explored in much finer detail by calling up Dotter "on the fly". Blixem hands the HSPs over to Dotter, which can display the HSPs in two different ways: by greyscale according the total HSP score, or by monochrome red lines which can be superimposed over the full dot-matrix. It is also possible to superimpose four different shades of red to reflect the score of the HSP.

Using Dotter for gene prediction

The genomic database ACEDB [Durbin and Thierry-Mieg, 1996] allows interactive gene modelling, with full display of relevant features such as splice sites, open reading frames, segments of high coding potential, sequence homology, etc. If the gene in question has homologous sequences, the multiple alignment of the homologues can be viewed by calling up Blixem from ACEDB, which also passes on the tentative gene prediction coordinates. For a more detailed analysis of how the homology fits with the gene prediction, the coordinates of the predicted gene are also passed on from Blixem to Dotter, which then displays the dot-plot comparison between the genomic DNA where the gene was predicted and the homologous protein (figure 5.2). Having the gene prediction displayed in the dot-plot significantly aids the ability to accept weakly conserved exons, and to reject ones that are inconsistent with the homology.

Using Dotter to assist genomic sequencing

Dotter can also be useful in the sequencing process. If sequencing is done by a 'shotgun' approach, the reads have to be assembled into one contiguous sequence by looking for overlapping ends of contigs. In cases where the assembly algorithm fails to find overlaps, or when it joins contigs incorrectly because of repeats, Dotter can be a useful tool to find which contigs should be merged. It will produce a mosaic dot-plot as shown in figure 5.4, if the input files contain multiple sequence entries in Fasta format. At the start of each subsequence, which would correspond to the consensus sequence of a contig, a green separator line and the name of the subsequence is drawn. These features are inherited if a region is zoomed in.

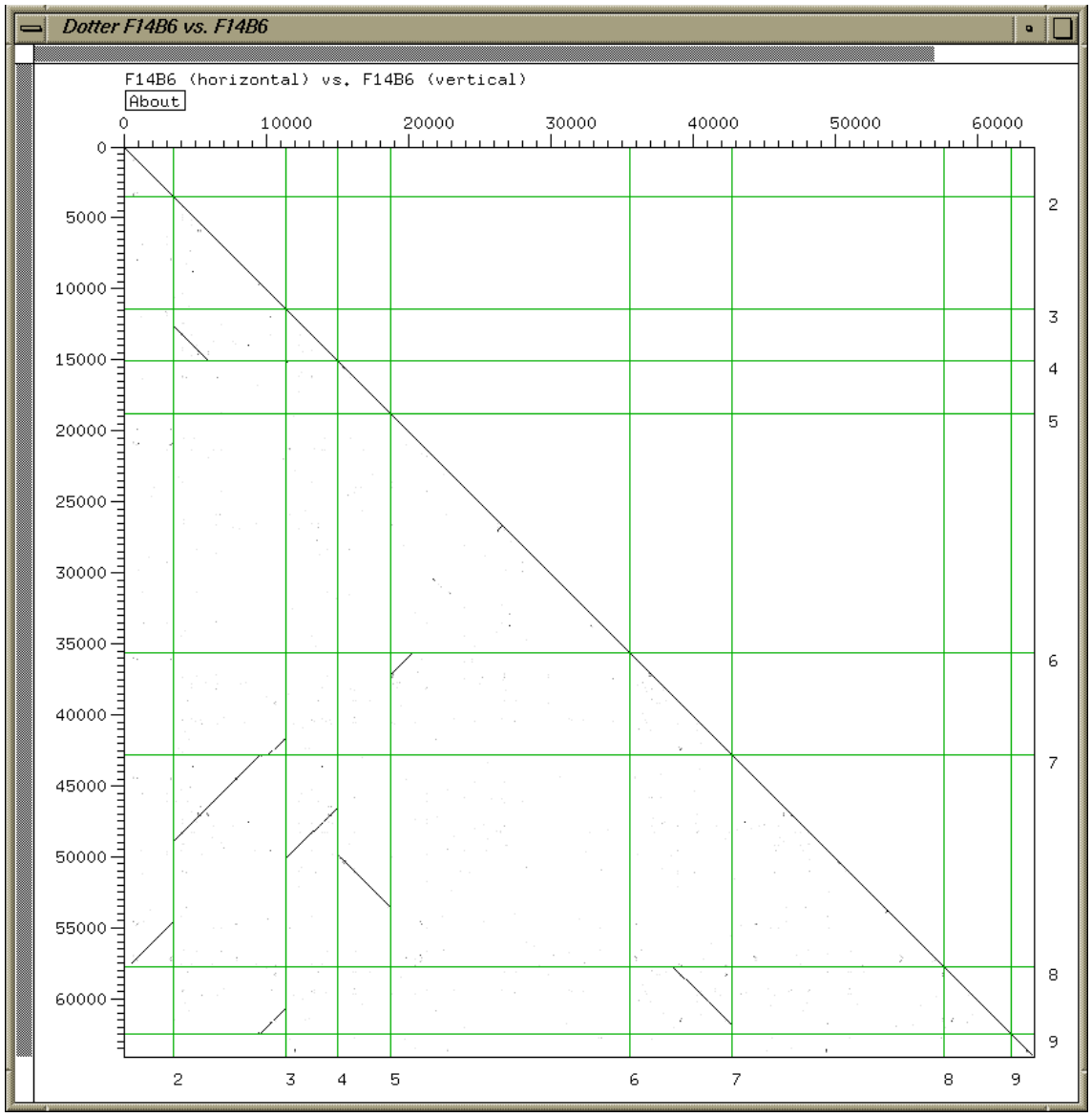


Figure 5.4. Example of how Dotter can be used to assist the fragment assembly of a cosmid sequence. In this case, contigs from two cosmids that were initially assembled independently are analysed for overlaps.

5.4 Application

Sequenced cosmids from the *C. elegans* genome sequencing project [Wilson *et al.*, 1994] are routinely compared to themselves with Dotter for analysis of the extent and nature of direct and inverted DNA repeats. Such repeats are interspersed throughout the genome, and there are many different recurring families [Naclerio *et al.*, 1992]. For example, the *C. elegans* cosmid ZK1307 (figure 5.1) contains several repeat families: 33+4 copies of a 40-mer, 22 copies of a 35-mer, 21 copies of a 15-mer and 2 copies of a 123-mer which contain 5 copies of an 11-mer in the middle. Naclerio et al. previously described the first 3 of these repeat families and named them RcC9, Rc35 and RcD1, respectively. The 40 bp repeat RcC9 [La-Volpe *et al.*, 1988], between 6750 and 8050, shown in detail in figure 5.1b-d is especially interesting since it has a less strongly conserved subunit of 10 bp which itself is palindromic, giving a minimal repeat unit in alternate orientations of only 5 bp: -TTC-. The smaller repeat units are however much less conserved than the 40 bp repeat. At very low stringency the dot-plot hence shows 10 bp spaced diagonals in both orientations (figure 5.1b). As the stringency is raised (figure 5.1c-d), the 10 bp spaced diagonals fade away, leaving only the strongest conserved 40 bp repeats in the dot-plot.

For arrays of tandem repeats such as this, Dotter makes it very easy to find the start and end of the repetitive unit and the number of repeats, which is especially important for constructing high-quality multiple alignments. As illustrated in figure 5.1, it is often far from trivial to determine the length of the main repeat unit, since multiples or fractions thereof are plausible units too. With the Greyramp and Alignment tools, this becomes a relatively easy task.

The need for a dot-plot program that can compare DNA to protein sequences was also prompted by the *C. elegans* genome project, where most primary protein homology analysis is carried out by comparing DNA to protein. The reason for doing this is that using predicted coding segments may miss homologies if the gene prediction was incorrect. Database searching is usually done with the program Blastx, in conjunction with the filtering program MSPcrunch (chapter 4) to increase sensitivity and selectivity. The DNA-protein HSPs are

then aligned in the X-windows viewer Blixem. Integration of Dotter into ACEDB and Blixem hence made it natural to carry over the DNA vs. protein philosophy to Dotter, as shown in figure 5.2. One could envisage using a different colour for each translated frame, but given that real homologies are normally confined to a single frame, and that the frame can easily be determined with the Alignment tool, we found the best solution was to leave them in the standard greyscale colours. The exons and introns of the gene prediction are shown just below the dot-plot border.

Figure 5.3 shows a self-comparison of the protein UNC-22 or twitchin, a large muscle protein which probably interacts with myosin [Benian *et al.*, 1989] [Benian *et al.*, 1993]. It consists of repeated fibronectin type III (FN3) and immunoglobulin (IG) domains and one kinase domain. At the N- and C-termini, five tandemly repeated IG domains are present, while the interior contains repeated "cassettes" of usually two FN3 and one IG domain. With the coloured segment boxes, it is easy to see how the similarity levels vary for the different domains. For instance, while the FN3 and IG domains in the N-terminal portion of the cassette repeat region are very similar, they are less conserved towards the ends. Especially the IG domains are very poorly conserved except in the middle of the cassette region. The five N-terminal IG domains are more similar to each other than to other ones, whereas for the five C-terminal IG domains this is not the case. The dot-plots in figure 5.2 and 3 were generated using the score matrix BLOSUM62 [Henikoff and Henikoff, 1992]. Dotter can read an arbitrary score matrix from file.

5.5 Discussion

Dotter is a new type of dot-plot program which is well suited to handle demanding homology analysis tasks involving weak and difficult to assess matches in both traditional protein or DNA comparisons and in more complex situations when genomic DNA is compared to proteins or DNA. Its main strength is that the dot-matrix only has to be calculated once, after which the stringency thresholds are varied dynamically, avoiding tedious reiteration of the

dot-matrix calculation. This is particularly useful when no optimal stringency exists, for instance if a diagonal can only be seen when the background noise is also visible. Such diagonals may still be biologically significant if they make good sense with other diagonals and/or if they contain important key residues. In cases like this, it is desirable to view the dot-plot under many different stringency conditions and be able to change them in a scrolling fashion.

The program XSauci [Nedde and Ward, 1993] also uses colourmaps for dynamic threshold control, for a variant of dot-plots called "correlation images", which transforms diagonals to horizontal lines. XSauci uses greyscales differently than Dotter however, in that the pixel intensity reflects the length of a match instead of the score, and it employs only one threshold.

The integration of Dotter into the multiple alignment viewer for BLAST matches, Blixem, makes a very powerful combination. With the add-on MSPcrunch, BLAST usually picks up at least one local match to homologous sequences, but may miss weak matches or matches to repeated domains. Dotter can then be called up directly from Blixem for a particular protein to show the true extent of the homology. This system provides very efficient and comfortable sequence homology analysis, with a minimal risk of overlooking similarities or assessing them incorrectly.

Alignment algorithms based on dynamic programming are a popular method of pairwise sequence similarity analysis which can be very sensitive if the gap weights are set correctly. However, for weak similarities the alignment is often very vulnerable to small changes in the gap weights, and often only a narrow range of parameters gives the correct alignment [Argos and Vingron, 1990] [Vingron and Waterman, 1994]. Dot-plots do not suffer from this problem, since no attempt is made to string matching segments together with gaps in between. Several users have asked if it would be possible to generate a gapped alignment by dynamic programming from Dotter. Since this would not improve over the standard implementations of dynamic programming, we have not included this feature. One might envisage however, that the user could select a number of diagonals, which are considered relevant. These segments could then be strung together in an alignment, possibly using dynamic programming to fill in the gaps, but allowing interactive control of the alignment path [Rechid *et al.*, 1989].

Dotter is available by anonymous FTP from ftp.sanger.ac.uk in the directory /pub/dotter, by World Wide Web on http://www.sanger.ac.uk/dotter.html or by sending E-mail to esr@sanger.ac.uk. ACEDB is available by anonymous FTP from ftp.sanger.ac.uk in /pub/acedb. Most of this chapter has previously been published [Sonnhammer and Durbin, 1996].

5.6 Acknowledgements

I am grateful to Friedemann Wobus and Darren Platt for work on the Greyramp tool and various other image handling software in the ACEDB graphics library, and to the *C. elegans* sequencing consortium, in particular Mary Berks for sequencing the cosmid ZK1307. Many thanks to Simon Dear, Gos Micklem, Sean Eddy and John Sulston for helpful discussions.

6. Efetch: a database retrieval tool

6.1 Summary

Sequence similarity data is stored in ACEDB as links to entries in external sequence databases. These links are passed on to the analysis tools Blixem, Dotter and Belvu, which need to retrieve the sequence or annotation information from the external database. Although ACEDB can store data from other databases internally, it is often preferable to only store the links to minimise the load. Moreover, a method to retrieve this information is always needed when the analysis tools are run stand-alone. To this end, an indexed sequence database retrieval tool called Efetch was developed. It is a general purpose program which supports a wide range of database formats and output formats.

6.2 Introduction

ACEDB and the tools of the sequence similarity analysis workbench store and examine database search results, and need to access the entries in the searched databases in an efficient way. As a glue between the viewers and the databases, a general-purpose database retrieval tool, Efetch, was developed. It is based on a standard indexing system, which many EMBL databases are released with.

Efetch serves several purposes. It can either be used from the command line, in scripts, or be called from other programs via a shell pipe. The main usage is to retrieve the sequence or annotation of single sequence entries in databases such as Swissprot, EMBL and TREMBL, but it can also be used to retrieve multiple alignments of from protein family databases such as Pfam (chapter 7) and Prodom [Sonnhammer and Kahn, 1994], or family annotation from Prosite [Bairoch *et al.,* 1996]. Efetch supports the flatfile format of all major sequence databases and can produce output in a number of formats.

When the workbench tools are called from ACEDB, the sequence data and annotation can either be retrieved via Efetch, or be stored internally in ACEDB. However, internal storage means duplication of data, and slower operation of other ACEDB tasks. Therefore, for large projects, it is usually preferable to only store the names in ACEDB, so that the sequence entry can be retrieved only when it is needed. To generate the Blixem alignment of a genomic sequence and homologues, all that needs to be stored internally in ACEDB is thus the names (or accession numbers) and positional coordinates of the BLAST matches.

6.3 Results

Efetch has been used extensively at the Sanger Centre and at the Genome Sequencing Center in St. Louis, and at other institutes that have enough resources to maintain their own copies of the sequence databases. Below follows more detailed descriptions of the different modes of Efetch usage.

Command line syntax

For ordinary command line usage, the syntax is:

```
efetch [options] database:entry
```

Each database needs to be stored in a separate directory. There are two methods to link that directory to a database prefix. The first method is to set a predefined environment variable to the directory. This is available for all common databases such as Swissprot, EMBL, Genbank, PIR, Prosite, etc.. The second method, which is general and can be used for any database, is based on adding the prefix and directory to the environment variable EFETCH_PREFIX. For example, to link the prefix mydb to the directory /mydbdir, the syntax would be:

setenv EFETCH_PREFIX "mydb:/mydbdir/;”

By default, mydb is assumed to be in Fasta format. If it is not, but in ‘flatfile’ format, e.g. EMBL, this must be indicated by using “mydb(flat):” instead of “mydb:” in the prefix definition. Any number of “prefix:dir;” entries can be added to EFETCH_PREFIX. A prefix can be up to 30 characters long.

Efetch can retrieve records using either entryname or accession number. When using the accession number, the option -a must be given. By giving no arguments, or the option -h, the syntax and all options are listed.

Output formats

The output format is controlled by the options on the command line. There are four main types of output: 1. The whole entry as in the flat file (default). 2. The sequence only, in the flat file format (which may contain column formatting and residue numbers). 3. In Fasta format (One line headers starting with “>“ followed by raw sequence). 4. Only the sequence, on one line. This is used to make parsing trivial when Efetch is called from other programs.

Blixem requirements

If Blixem does not get the sequences passed from MSPcrunch in the seqbl format, or from ACEDB, it will resort to calling Efetch once for each unique sequence. As shown in figure 6.1, calling up annotation for a matching sequence is done by simply double clicking on the match. Efetch is then asked to retrieve the annotation which appears in a separate window. This can be done in three ways: 1. By calling a locally installed Efetch. 2. By starting a WWW browser with a URL to the Sanger Centre Efetch server (see below). 3. If Blixem is running within an ACEDB, that contains the annotation, ACEDB object can be displayed.

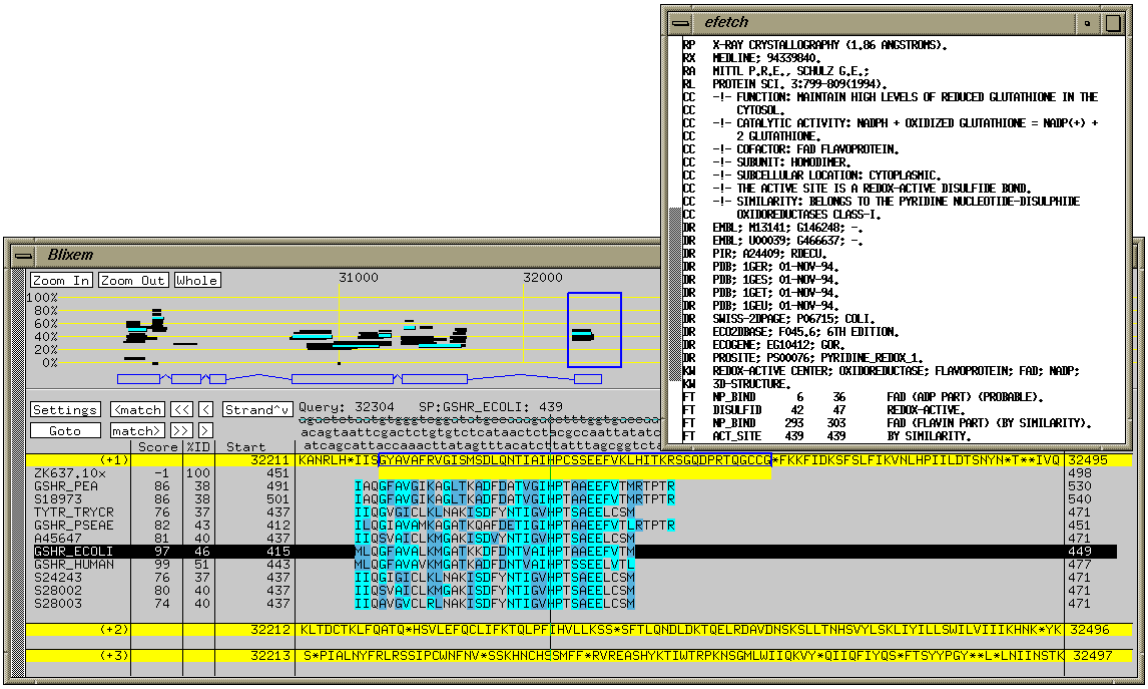


Figure 6.1. The ACEDB/Blixem environment is linked to external databases, providing sequence data and annotation, via Efetch. The example shows a Blixem alignment of glutathione reductases to ZK637.10. The annotation of GSHR_ECOLI was retrieved by clicking on it (reversed line in Blixem), which calls Efetch and displays the entry in a separate window, labelled 'efetch'. In this case, the 100% conserved histidine at position 32304 in ZK637 aligns with residue 439 in GSHR_ECOLI, which is annotated as the active site residue, and thus strongly supports that ZK637.10 would be a glutathione reductase. When Blixem is passed sequence names only, it also calls Efetch to retrieve the sequence data.

Dotter requirements

When Dotter is called from Blixem, the sequences are passed on to Dotter. If Blixem is running on seqbl data, however, it only has incomplete database sequences, corresponding to the matching segments. It will then attempt to call Efetch to retrieve the entire sequence. If this fails, Dotter will produce a dotplot of the matching segment only, and give a clear warning of this.

Belvu requirements

The multiple alignment viewer Belvu (chapter 8), is linked to annotation retrieval the same way as Blixem is; double clicking on a sequence will efetch it. The alignments displayed by Belvu may be Efetched too, if they are from Prodom or Pfam (chapter 7), which can be indexed for Efetch. When Belvu is called from ACEDB, alignments can either be stored internally in the database, or be fetched from an external database.

Efetch via World Wide Web

When Blixem and Belvu call Efetch, the databases need to be installed locally. If this is not the case, the Sanger Centre WWW Efetch server can be used via the internet (URL: http://www.sanger.ac.uk/cgi-bin/seq-query?seqname or .../seq-query_acc?accession). Instead of the usual Efetch window, a web browser will then be spawned with the appropriate URL. Even for sites with Efetch installed, this can be of use, since the WWW browser can follow references to other sequence databases and Medline abstracts. Local Efetching is however much faster. The Efetch WWW server currently uses the NCBI server at http://www3.ncbi.nlm.nih.gov/PubMed for linking to Medline abstracts.

Index files

The index system conforms to the standards proposed by the EMBL data library [Fuchs and Stoehr, 1993]. This system is used on the EMBL CD-ROM distributions and in the Staden package [Staden, 1990]. All index files for one database must be stored in the same directory. The index files used by Efetch are shown in figure 6.2. The main entry index, entry-

nam.idx, contains offsets and division numbers for each entry. The division number points to a particular flatfile as coded in the file division.lkp. This way the database can consist of many divisions in different files. The accession number index acnum.trg points to records in a 'hit file' acnum.hit, which in turn points to the main entry index. The number of different entries sharing one accession number is stored in the acnum.trg record, and pointers to all these entries are stored consecutively in the hit file. Efetch searches the accession number and entry name indices by a binary search. If the query is not unique in the database, a list of the entries that start with the query string is returned.

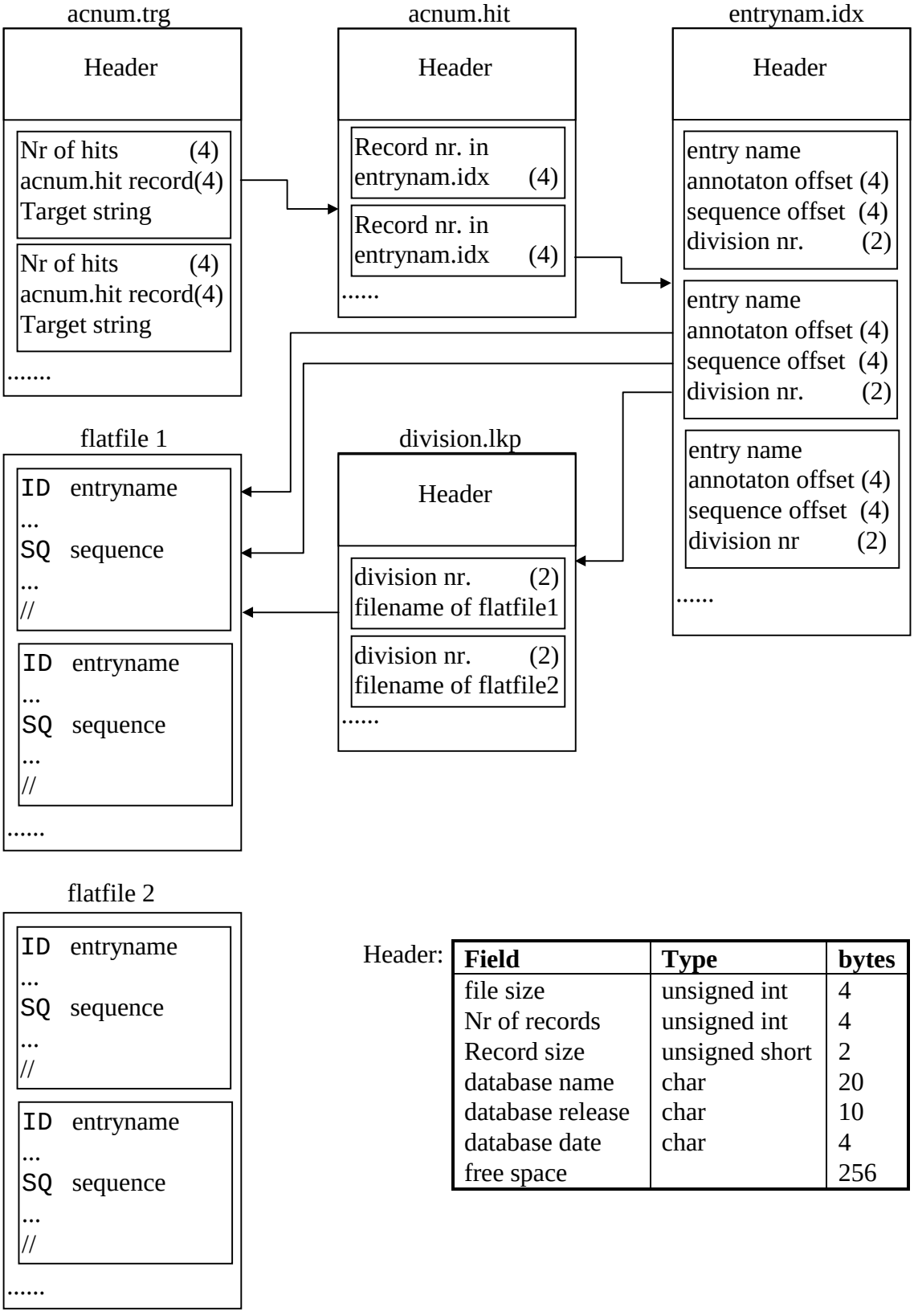
The tools for creating the indices are extensions to Staden's indexing programs [Staden and Dear, 1992]. Currently these formats are supported for indexing: Genbank, EMBL, Swissprot, PIR, Prosite (.doc and .dat) and Prodom, Pfam and any database in Fasta format. New formats can easily be incorporated.

Compatibility with other retrieval software

If other retrieval systems than Efetch are already installed locally, that do not use the flat distribution files, installing Efetch would mean a duplication of the databases. The GCG Fetch [Devereux *et al.*, 1984], Yank [White and FitzHugh, 1996] and Getz [Etzold and Argos, 1993] retrieval programs are examples of this. In such cases, Efetch can be emulated with a script that calls the other program when Blixem and Dotter calls Efetch. A script for GCG is available.

Availability

Efetch and the index making programs are available at ftp.sanger.ac.uk in /pub/MSPcrunch/efetch.tar.Z.



Header:

Field	Type	bytes
file size	unsigned int	4
Nr of records	unsigned int	4
Record size	unsigned short	2
database name	char	20
database release	char	10
database date	char	4
free space		256

Figure 6.2. The system of indices used by Efetch. The numbers within brackets are the number of bytes a field contains. When no number is given the size is variable, and is derived from the record size in the header.

Part 2: Pfam: a Comprehensive Database of Protein Domain Families

7. Construction and maintenance of Pfam

7.1 Summary

Databases of multiple sequence alignments are a valuable aid to protein sequence classification and analysis. One of the main challenges when constructing such a database is to simultaneously satisfy the conflicting demands of completeness on one hand and quality of alignment and domain definitions on the other. The latter properties are best dealt with by manual approaches, while completeness in practise is only amenable to automatic methods. Here we present a database based on hidden Markov model profiles (HMMs) which combines high quality and completeness.

Our database, **Pfam**, consists of parts A and B. **Pfam-A** is curated and contains well characterised protein domain families with high-quality alignments, which are maintained by using manually checked seed alignments and HMMs to find and align all members. **Pfam-B** contains sequence families that were generated automatically by applying the Domainer algorithm to cluster and align the remaining protein sequences after removal of Pfam-A domains.

Using Pfam, many novel family memberships in known proteins were identified, including new kazal, Fibronectin type III, and response regulator receiver domains. Pfam-A families have permanent accession numbers and form a library of HMMs available for searching and automatic annotation of new protein sequences.

7.2 Introduction

Protein sequence databases such as Swissprot [Bairoch and Apweiler, 1996] and PIR [George *et al.*, 1996] are becoming increasingly large and unmanageable, mainly as a result of the growing number of genome sequencing projects. However, many of the newly added proteins are new members of existing protein families. Typically between 40% and 60% of the proteins found by genomic sequencing show significant sequence similarity to proteins

with known function, and usually a large fraction of the rest show similarity with each other [Koonin *et al.,* 1994; Casari *et al.,* 1995; Hodgkin *et al.,* 1995]. For classification of newly found proteins, as well as orderly management of already known sequences, it would therefore be advantageous to organise known sequences in families and use multiple alignment based approaches. This requires a system for maintaining a comprehensive set of protein clusters with multiple sequence alignments.

The problem breaks down into two parts: defining the clusters, i.e. a list of members for each family, and building multiple alignments of the members. Previous approaches to construct comprehensive family databases have either concentrated on aligning short conserved regions [Gribskov *et al.,* 1988; Attwood *et al.,* 1996; Pietrokovski *et al.,* 1996], often starting from the manually constructed clusters in Prosite [Bairoch *et al.,* 1996], or full-domain alignments using either clusters that were derived manually from PIR [George *et al.,* 1996] or automatically [Sonnhammer and Kahn, 1994]. An issue here is whether to aim for conserved regions only, or whole-domain alignments. Using short conserved motifs, either in the form of a pattern or an alignment, can indicate when a protein contains a known domain. Motif matches are often useful to indicate functional sites. However, they usually do not give a clear picture of the domain boundaries in the query sequence. They may also lack sensitivity when compared to whole-domain approaches, since information in less conserved regions is ignored. The whole-domain approach therefore seems preferable for detailed family-based sequence analysis since it offers the potential for the most sensitive and informative domain annotation.

To cope with the large number of families, the existing family databases made heavy use of automatic methods to construct the multiple alignments. Almost without exception, a manually constructed alignment would have been preferred, but maintaining a comprehensive collection of hand-built alignments is not feasible. If the clustering is done at a high level of similarity, such as 50% identity, the alignment can be generated relatively reliably with automatic methods, but this will fragment true families and compromise the speed and sensitivity of searching. To avoid this, high-quality alignments of large superfamilies are needed, which frequently require manual approaches.

Apart from the multiple alignment construction problem, a fully automatic approach also has to provide a clustering and, to work for multi-domain proteins, define domain boundaries. For instance, the Domainer algorithm [Sonnhammer and Kahn, 1994] which performs the clustering of domain families based on all versus all Blastp matching, is a fully automatic approach that we have used. We are most familiar with its drawbacks and believe that other automated sequence clustering approaches share similar drawbacks. The clustering level of Domainer depends on the score level of accepted pairwise Blastp matches. Domain borders are inferred by analysing the extent of the Blast matches and from N- and C-terminal ends. The main problem with Domainer is that it does not scale well. As the sequence database grows, this will have several manifestations: 1) The computing time increases in the order of N^2. 2) Either the clustering level must go up or the risk of false family fusions will increase. 3) The domain boundaries become less reliable due to more noise in the Blastp data. 4) The quality of the alignment drops as more members are added. Further drawbacks of Domainer are that it is sensitive to incorrect data, and that it is a one-off process that does not allow incremental updates but must be completely rerun at each source database update. This is not only very costly computationally, but also means that the families are volatile, due to the heuristic character of the algorithm, and can not be permanently referenced from other databases. It is not well-suited for classification, since the families lack family-level annotation.

Presently available fully automatic methods are thus not suitable for a high-quality family-based classification system. Could a combination of manual and automatic approaches be a solution? The question here is really how much manual work has to be done to achieve a comprehensive database. This depends on the distribution of protein family sizes. Based on sequence similarity, it is clear that the universe of proteins is dominated by a relatively small number of common families [Green *et al.*, 1993]. The same type of analysis on the structural level reveals that there are a few families of very frequently occurring folds [Murzin *et al.*, 1995], and it has been estimated that a third of all proteins adopts one of nine 'superfolds' [Orengo *et al.*, 1994]. This led us to believe that a semi-manual approach initially applied to the largest families could capture a substantial fraction of all proteins. For practical reasons however, it is usually not possible to build correct alignments solely based on the sequence

data from members sharing a common fold, since often there is essentially no sequence similarity at this level. The structural information required to produce a correct alignment is available only for a fraction of proteins. It therefore makes more sense to perform the clustering at the superfamily or family level, where common ancestry and sequence similarity are reasonably clear.

A major stumbling block of manual approaches is the problem of keeping the alignments up to date with new releases of protein sequences. A robust and efficient updating scheme is required to ensure stability of the database. These requirements are met in Pfam by using two alignments: a high quality **seed** alignment, which changes only little or not at all between releases, and a **full** alignment, which is built by automatically aligning all members to a hidden Markov model based profile (HMM) derived from the seed alignment. The method that generates the best full alignment may vary slightly for different families, so the parameters used are stored for reproducibility. This split into seed/full is the main novelty of Pfam's approach. If a seed alignment is unable to produce an HMM which can find and properly align all members, it is improved and the gathering process is iterated until a satisfactory result is achieved.

The seed and full alignments, accompanied by annotation and cross-references to other family and structure databases and to the literature, and the HMMs, are what make up Pfam-A. Each family has a permanent accession number and can thus be referenced from other databases. We strived to include every family with more than 50 members in Pfam-A. All sequence domains not yet in Pfam-A were then clustered and aligned automatically by the Domainer program into Pfam-B. Together, Pfam-A and Pfam-B provide a complete clustering of all protein sequences. The quality of the Pfam-B alignments is generally not sufficient to construct useful HMMs. The main purposes of Pfam-B are instead to function as a repository of homology information and a buffer of yet uncharacterised protein families. As these families become larger they will benefit more from being incorporated into Pfam-A. Our goal is to progressively introduce the largest Pfam-B families into Pfam-A.

This chapter describes how Pfam was constructed and how it is maintained, and presents results from applying the Pfam HMM library to analyse protein families in the Swissprot protein database.

7.3 Methods

7.3.1 Pfam-A

HMMs

Hidden Markov model based profiles (HMMs) have been used extensively both for the construction of Pfam, and for detecting matches to Pfam families in database sequences. Although hidden Markov models are a general probabilistic modelling technique, we will use HMM in this chapter to mean a specific form of model which describes the sequence conservation in a family. This type of HMM consists of a linear chain of match, delete and insert states as shown in figure 7.1 [Krogh *et al.,* 1994a; Eddy, 1996]. The match state contains probabilities for amino acids in a given column, while the transition probabilities to and from insert and delete states reflect the propensity to insert a residue or skip one at a given position. The HMM parameters can either be estimated directly from a multiple alignment, or iteratively by an Expectation-Maximisation procedure from unaligned sequences. A protein sequence can be aligned to an HMM using dynamic programming to find its most probable path through the states. The logarithm of this probability over the probability of a random model gives the score of the match, usually expressed in bits (logarithm base 2).

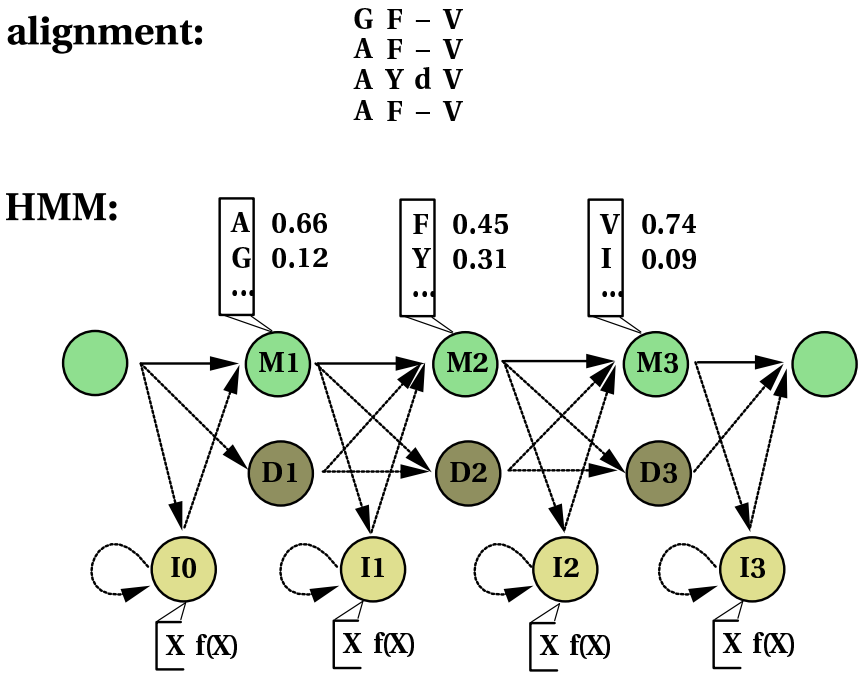


Figure 7.1. Schematic representation of the hidden Markov model used for the families in Pfam-A. This example shows an HMM corresponding to the miniature alignment on top. Circles represent states (M match; D delete; I insert) and arrows transitions between them. Probabilities for each residue (only some are shown) are stored in each match state. The residue probabilities in the insert states are set to the residue frequencies in Swissprot (f(X)).

Score matrix based profiles [Gribskov *et al.*, 1987] are similar and might also have been used throughout. However, there are reasons to believe that HMMs are a somewhat superior approach to matrix based profiles [Krogh *et al.*, 1994a]. A practical reason for choosing HMMs was the suitability to the task of the HMMER package [Eddy, 1995a], which includes the programs hmmls for finding multiple non-overlapping complete domains in a target sequence, and hmmfs for finding multiple non-overlapping partial and/or full domains.

Seed and full alignments

The philosophy behind Pfam-A is to construct a seed alignment for each family, from a non-redundant representative set of full length domain sequences trusted to belong to the family. The quality of each seed alignment was controlled by manual checking. From the seed alignment, an HMM was built, which then was used to find new members and to generate the alignment of all detected members. The process of seed alignment and member gathering was iterated as outlined in figure 7.2 if the initial seed was unsatisfactory. The HMMs were not built from the all-member alignment since this may contain incomplete or incorrect sequences which may affect the HMM adversely. The full alignments were never edited; if they were unacceptable, either the seed alignment was improved or the method to generate the full alignment from the seed was changed.

Seed alignment construction

The initial members of a seed were collected from one of several sources: Swissprot, Prosite, structural alignments [Overington, 1992], Prodom, Blast results, repeats found by Dotter (chapter 5) or published alignments. Families were chosen on an *ad hoc* basis, with a bias towards families with many members. If the source provided a complete alignment of the seed members, this was used, but usually an alignment had to be built and compared to known salient features such as active site residues or structurally important residues. Of the automated alignment methods used (Clustalw [Thompson *et al.*, 1994], Clustalv [Higgins *et al.*, 1992], HMM training [Eddy, 1995b]), Clustalw most often produced the best alignment. In a few cases, manual editing of the seed alignment was necessary. Any sequence that was suspected to contain an error such as truncation, frameshift or incorrect splicing was not included in the seed alignment, to avoid adding noise to the HMM. This is important since up to 5% of the sequences in Swissprot may contain such errors (T. Gibson, personal communication).

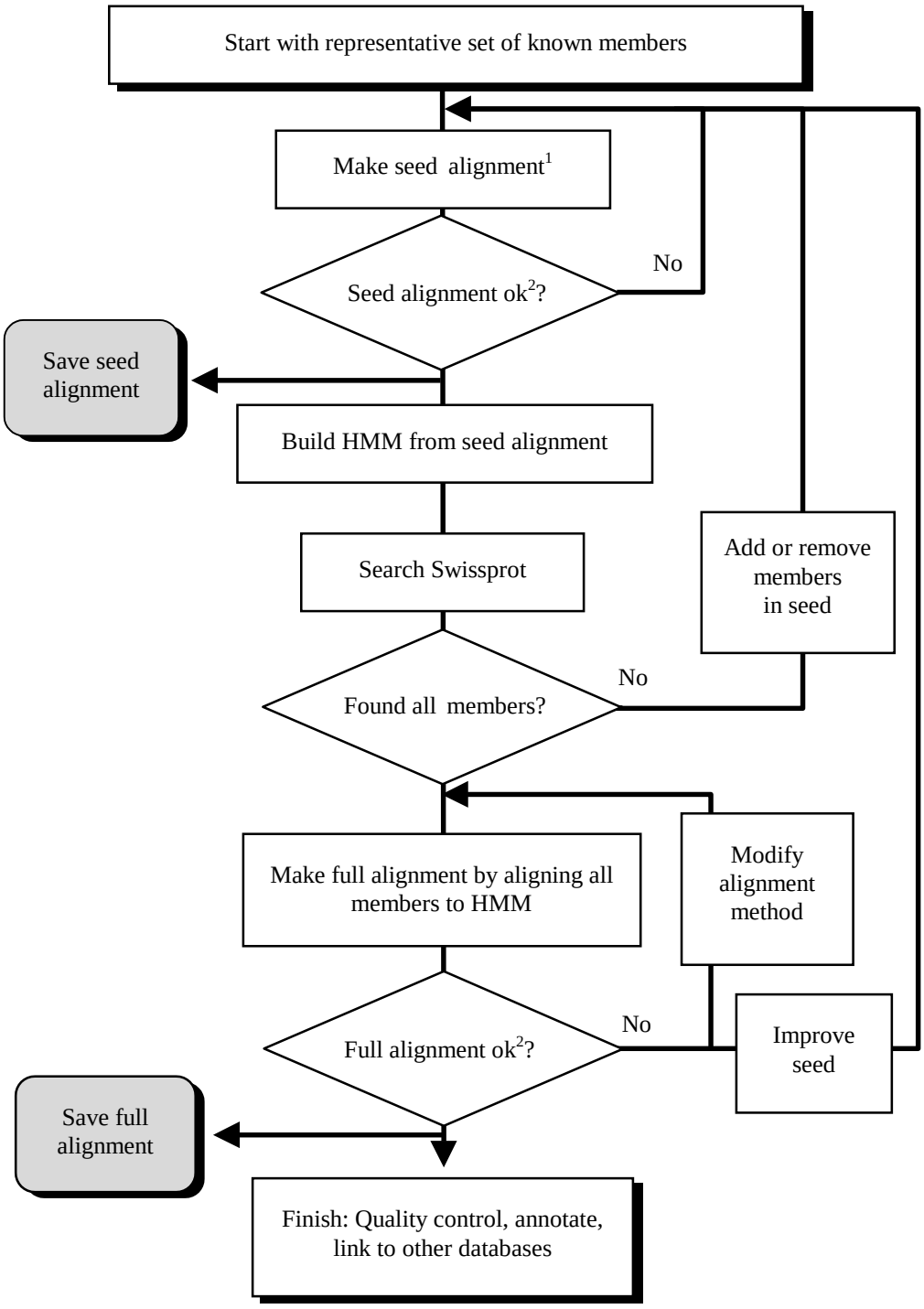


Figure 7.2. The procedure to construct the alignments and HMM for a Pfam-A family. [1]Initial seed alignments are taken either from a published alignment or are made by one of the methods described in the text. [2]By 'ok' we mean that known conserved features are correctly aligned and that the overall alignment has sufficiently high information content to separate known positives from negatives.

HMM construction

From each seed alignment an HMM was built using the hmmb program. Although care was taken to assure that the seed members did not include very similar sequences, one of two different weighting schemes [Gerstein *et al.*, 1994] [Eddy *et al.*, 1995] was applied to minimise any potential bias towards a subgroup.

To avoid overfitting and to make the HMM more general, amino acid frequency priors were normally derived according to an *ad hoc* pseudocount method [Tatusov *et al.*, 1994] using the BLOSUM62 substitution matrix. However, for some families (e.g. EGF, ef-hand, globin, ig), the less specific Laplace ('plus one') priors gave better results, and were therefore used.

Full alignment construction

Each HMM thus constructed was then compared to all sequences in Swissprot. This was either done directly with the search programs hmmls or hmmfs, or by converting the HMM to a GCG profile [Devereux *et al.*, 1984] in order to be able to use the very fast Bioccellerator hardware from Compugen [Esterman, 1995]. These programs all perform variants of dynamic programming: the programs bic_profilesearch on the Bioccellerator and hmmfs use a fully local algorithm, while hmmls is local in the query sequence but matches the entire HMM. A further difference is that bic_profilesearch only reports the highest score, while hmmls and hmmfs report all scores above a threshold, with co-ordinates. Although the Bioccellerator is about 50 times faster than a workstation, the result has to be post-processed with hmmfs or hmmls to extract the coordinates of all matches. This was done by retrieving the entire sequence of all proteins that match according to bic_profilesearch with the Efetch program (chapter 6) into a mini-database, which was then searched with hmmfs or hmmls.

If a list of known members of a family was available, the search result was compared to it to make sure that no known members were missed inadvertently. If the seed alignment is very small, one can not expect to find all members at once. In such cases, selected newly found members were incorporated in a new seed alignment, and the search was iterated. For

the families where the initial seed alignment was derived from structural superpositions, the new HMM was constructed with a modified training algorithm that constrains the known structural alignment, allowing only the sequences of unknown structure to be realigned.

By extracting all matching sequence fragments and aligning them to the HMM with the program hmma, a full alignment is created. Depending on the nature of the family, either hmmfs or hmmls will give more accurate matching segments. Hmmfs occasionally breaks a domain artificially into two or more fragments if unexpectedly large insertions or gaps are encountered. Hmmls does not do this, but may penalise partial matches (to fragments) so much that they are not found at all. Usually hmmfs is used, but in some cases hmmls was preferred. The method used for constructing the full alignment and the score cutoffs used were recorded for each family. The default score cutoff was 20 bits, but this was adjusted for some families as described below.

Quality Control

Once the seed and full alignments of a family have been constructed, a number of quality controls were performed. False positives and negatives relative to a reference clustering, usually from Prosite, were examined. Since Prosite describes motifs, the clusterings can not always agree completely. It is made sure that neither the seed nor full alignment overlaps by even a single residue with any other family. Both the alignments and the annotation are checked for format errors.

A problem with Pfam's strategy is that there is no intrinsic protection against one protein scoring high with two HMMs, if its sequence lies 'in between' the two families. This typically happens when two families are treated as separate, although they are known to be related. One case of this are the EGF domains and the related EGF-like domains found in laminins, where the laminin EGF-like modules are 20-30 residues longer than normal EGF domains and have eight instead of six conserved cysteines, possibly forming a fourth disulphide bond. When training an HMM on a cross-section of many EGF domains, this HMM will typically give a high score to laminin EGF-like domains. However, it was possible to train a tight EGF HMM where the alignment was very strict about features that are different

from laminin EGF-like domains, such as the exact spacing between some conserved cysteines. This HMM would only recognise non-laminin EGF domains. Pfam-A is checked for any overlaps between families and if this is found, either the seed alignment is modified or the score cutoffs are raised slightly.

Format

The Pfam format for the alignments is for each sequence segment: name/start-end followed by the padded sequence on one line. The name is the Swissprot acronym and the start and end are the co-ordinates of the first and last residues of the sequence segment. In the release flat file the Swissprot accession number is added to the end of each sequence line. The annotation follows the Swissprot flatfile format closely; each family in Pfam-A has a permanent referenceable accession number (Pfxxxxx), an ID name and a definition line. An example of annotation and alignment is shown in figure 7.3. The field labels in figure 7.3A follow the Swissprot syntax [Bairoch and Apweiler, 1996], with the addition of AU (alignment author), SE (seed membership source), AL (seed alignment method), GA (gathering method to find all members) and AM (alignment method of all members to HMM).

A

```
ID   response_reg
AC   PF00072
DE   Response regulator receiver domain
AU   Sonnhammer ELL
SE   Prodom
AL   Clustalw
GA   Bic_raw 25 hmmls 25
AM   hmma -qR
RA   Pao, G.M., Saier, M.H.
RL   J. Mol. Evol. 40:136-154(1995).
DR   SCOP; 3chy; fa;
CC   This domain receives the signal from the sensor partner in
CC   bacterial two-component systems.  It is usually found N-terminal
CC   to a DNA binding effector domain.
```

Figure 7.3. Example of the Pfam-A family response_reg (PF00072) with annotation (A) and alignment (B) (only part shown).

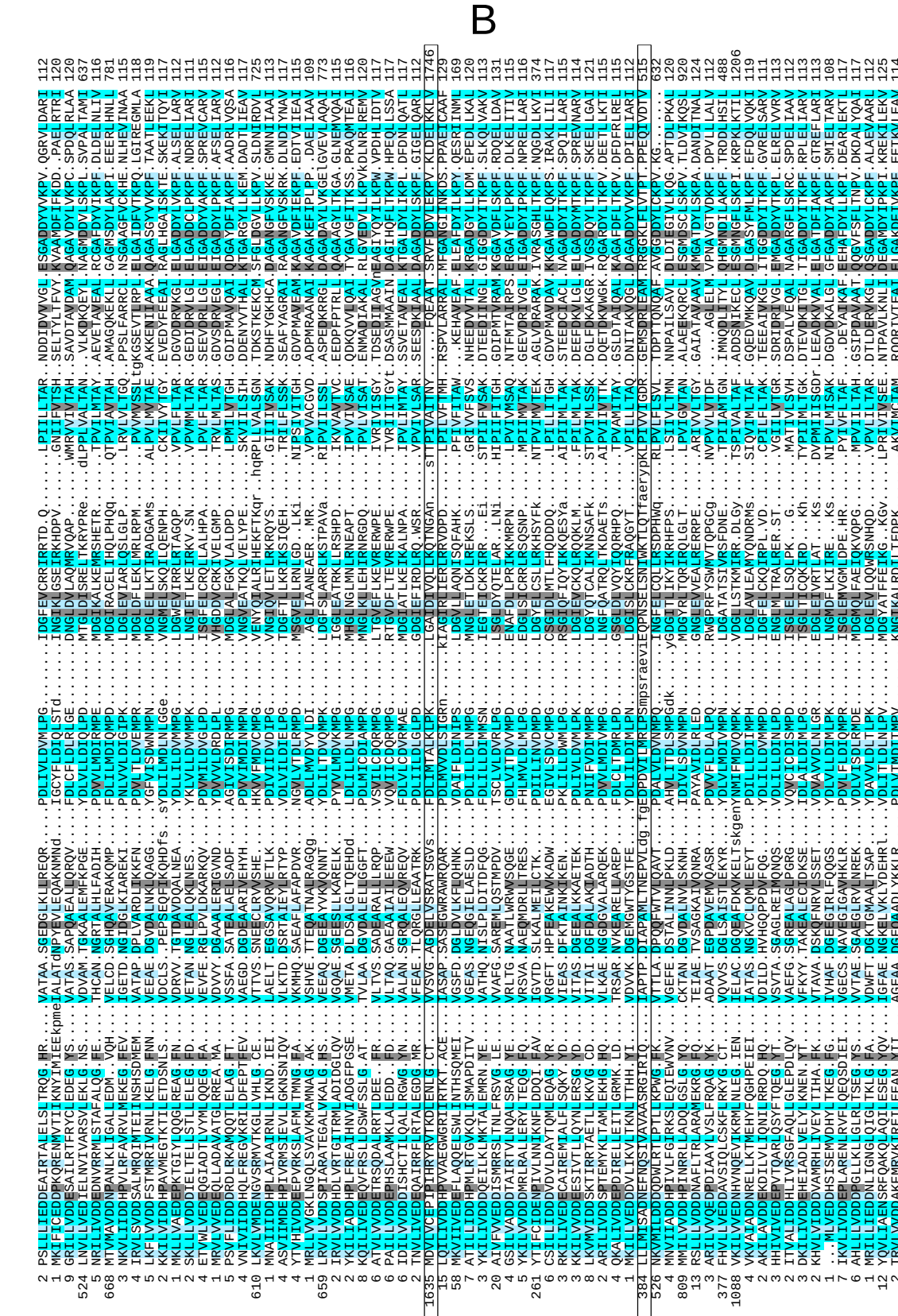


Figure 7.3b.

C

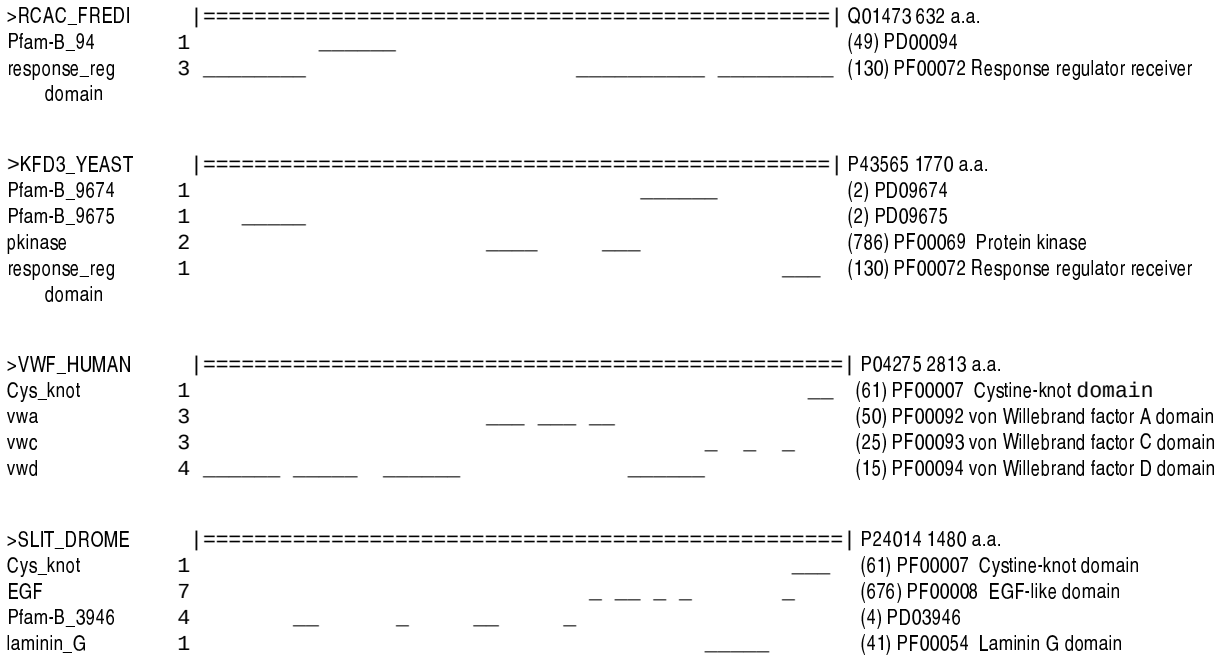


Figure 7.3c. The Pfam domain organisation of the KFD3_YEAST and the middle domain of RCAC_FREDI, where are novel member domains of the response regulator receiver domain family (see text). Two other examples of modular proteins are shown. This schematic representation is provided for each protein in Pfam in the release file swissPfam. The whole sequence is represented with '=' and the Pfam domains with '-' on the lines below. The columns of the domain lines are: Pfam ID, nr. of domains, schematic, nr. of members in the family, Pfam accession nr., description (Pfam-A families only) and start and end co-ordinates of the segments (not shown here).

D

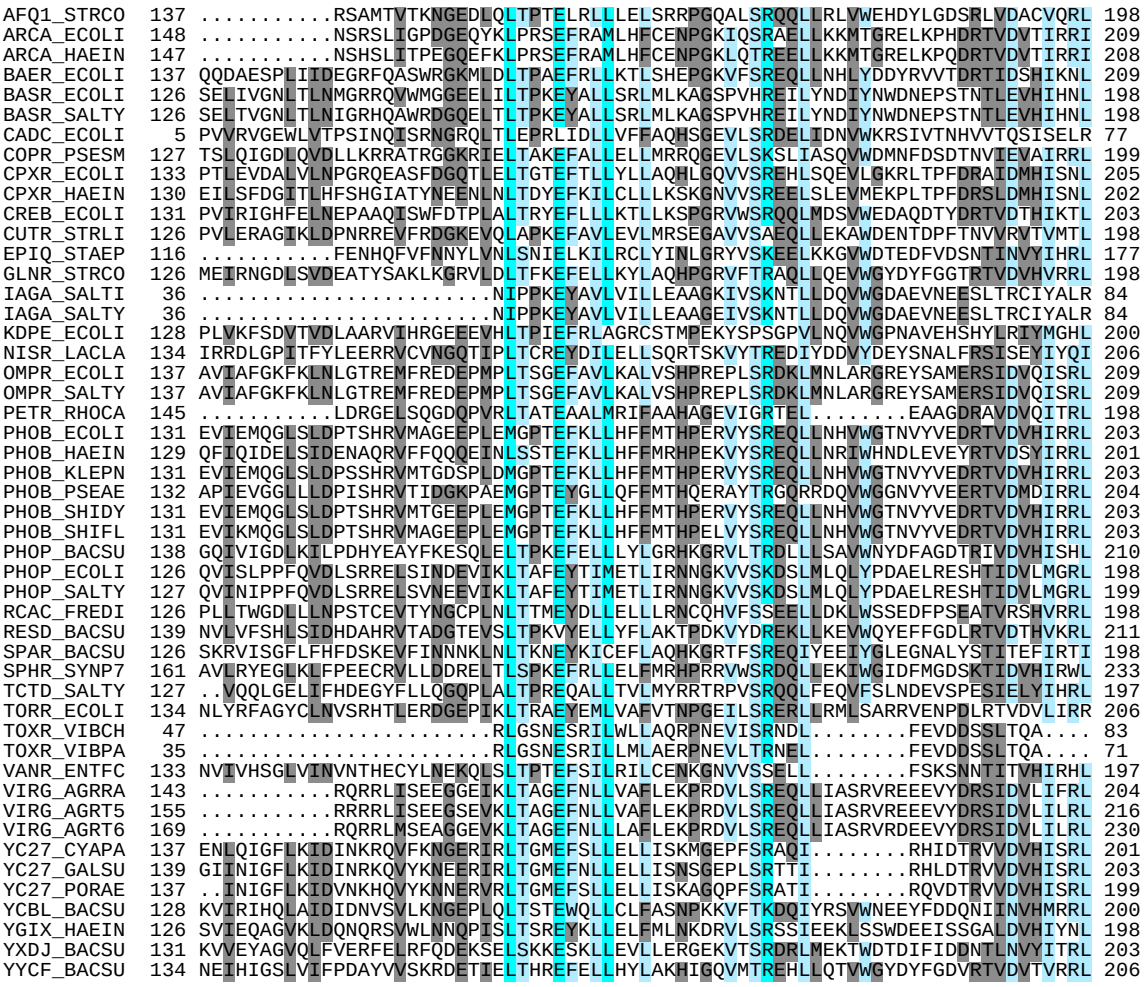


Figure 7.3d. Example of a Pfam-B family produced by Domainer. This family contains the DNA binding effector domain of RCAC_FREDI.

7.3.2 Pfam-B

To cluster all protein sequences not covered by Pfam-A, the Domainer program [Sonnhammer and Kahn, 1994], version 1.6, was run. Domainer uses pairwise homology data reported from Blastp [Altschul *et al.,* 1990] to construct aligned families. Blastp was only run on the part of Swissprot that was not present in Pfam-A. In release 1.0 of Pfam this was 81% of Swissprot 33. These sequences were prepared by extracting all sequence sections larger than 30 residues that were not covered in Pfam-A into separate entries. A protein with a Pfam-A domain in the centre that has long flanking regions on either side, will thus generate two entries. By doing this, Domainer will consider each section as an independent sequence, and the boundary to the Pfam-A segment will be used as a real domain boundary. All sequences known to be fragments were omitted since these would induce false domain boundaries in Domainer.

The Domainer process was further improved by filtering the Blastp output with MSPcrunch (chapter 4) to remove biased composition matches, trim off overlapping ends of consecutive Blast matches, and to reduce redundancy. As can be seen in figure 7.4, the growth of Homologous Sequence Sets (HSSs), is practically linear with the number of homologous Sequence Pairs (HSPs) processed, while running Domainer on all of Swissprot gives rise to large plateaux in areas of large redundancy [Sonnhammer and Kahn, 1994]. Although Pfam 1.0 is based on release 33 of Swissprot, which contains more than twice as many sequences as release 21, which Prodom 21 was based on, the number of HSPs was slightly reduced. Without reduction in redundancy by Pfam-A and MSPcrunch a quadrupling would have been expected. The time consumption for processing the HSPs into HSSs was 26.3 hours on one workstation. Performing the Blastp all versus all comparison took a total of 184.6 hours, but the elapsed time was reduced by running on a number of workstations in parallel. These timings show that it is clearly feasible to rerun the process periodically.

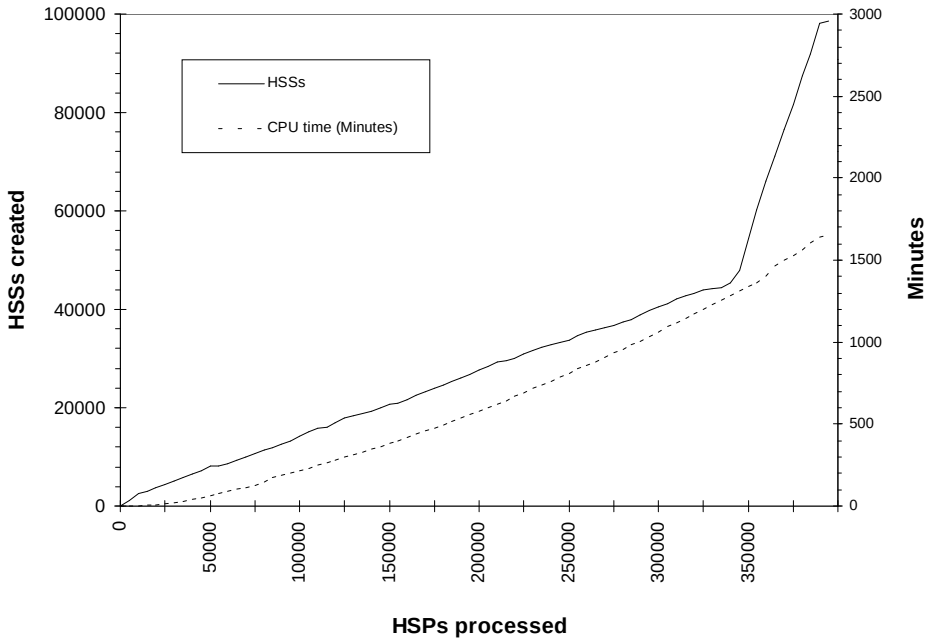


Figure 7.4. Construction of Pfam-B by Domainer. Plot of Domainer run on Swissprot 33, excluding sequences in Pfam-A. Domainer groups the pairwise matches (HSPs) into stacks of matches (HSSs) if different pairs share sequence regions. 46293 subsequences gave rise to 392207 HSPs, which resulted in 98551 HSSs in 11929 families after subsequent clustering by Domainer. When Domainer is run on the entire Swissprot, much time is spent on processing redundant pairs generated by large families, generating long horizontal plateaux in the plot (See [Sonnhammer and Kahn, 1994], figure 3). In contrast, the Pfam plot is virtually linear, since the most redundant families are already in Pfam and was thus removed before running Domainer. The sharp increase of the curve's slope at the end is caused by adding all full-length sequences as pseudo-matches after all the heterogeneous matches.

The Pfam-B alignments are released together with Pfam-A in one flat file. The format is essentially the same, but each Pfam-B cluster is assigned a volatile accession number (PDxxxxx), which is only valid for a particular release. Information sparse alignments that Domainer sometimes produces are avoided by excluding any alignment where more than 25% of the residues are gaps. In Pfam 1.0 this eliminated 34 out of 11963 alignments.

Incremental updating

Pfam was designed with easy updating in mind. When new sequences are released, they are compared to the existing models and if they score above the cutoff they are automatically added to the full alignment. Normally the seed alignment is not altered, except for updating of corrected seed sequences. However, if new sequences give rise to problems, such as strong cross-reaction between families, the seeds may have to be improved to become more specific for the respective families. Once Pfam-A is brought up to date, Pfam-B is regenerated on the rest of Swissprot as described above.

7.4 Results

We have constructed and made available a comprehensive library of protein domain families as described in the Methods section. Together with the HMM technology, this can provide an advance over traditional database searching in sequence analysis for classification purposes. Figure 7.5A illustrates the proportions of Swissprot that are covered by Pfam-A and Pfam-B. A third of all Swissprot proteins have one or more domain in Pfam-A, and a fifth of all residues are aligned in a Pfam-A family. Pfam-B is roughly twice the size of Pfam-A, leaving only 22% of all Swissprot proteins without any segment in Pfam at all. Pfam is available via anonymous FTP at ftp.sanger.ac.uk and genome.wustl.edu in /pub/databases/Pfam. There are two data files: pfam, which lists all the Pfam families with annotation and alignment, and swissPfam, which contains the Pfam domain organisation for each Swissprot entry in Pfam. There are also World Wide Web servers on

http://www.sanger.ac.uk/Pfam and http://genome.wustl.edu/Pfam which allow browsing and HMM searching against Pfam-A with a query sequence. Table 7.1 summarises the families currently in Pfam-A and the sizes of the seed and full alignments. On average, the full alignments have four times as many members as the seed alignments. The structure of 60% of the Pfam-A families is known. These families are cross-referenced to the structural classification database SCOP [Murzin *et al.*, 1995] from the Pfam WWW servers (see section 8.3).

The main use of Pfam is as a tool to identify and classify domains in protein sequences. We applied it to Wormpep 10, a database of 4874 predicted proteins from genomic sequencing of *C. elegans* [Hodgkin *et al.*, 1995]. The 2973 proteins for which no informative similarity has been found using the standard Blast/MSPcrunch approach (chapter 4) were searched for Pfam matches. As significance cutoffs, the previously recorded cutoffs that exclude negatives for each Pfam family were used. 211 Pfam matches were found in 144 unannotated sequences. Adding these to the already annotated *C. elegans* predicted proteins yields a classification rate of about 42%. As seen in figure 7.5B, already half that amount, 21%, is covered by matches to the Pfam-A HMM library.

Table 7.1. The families included in release 1.0 of Pfam-A. Since the seed alignments are smaller than the full alignments, quality control and maintenance become feasible tasks.

Pfam ID	Accession nr.	Description	HMM Length	Seed members	Seed a.a.	Full members	Full a.a.
7tm_1	PF00001	7 transmembrane receptor (Rhodopsin family)	271	64	17169	530	140214
7tm_2	PF00002	7 transmembrane receptor (Secretin family)	366	15	5339	36	12620
7tm_3	PF00003	7 transmembrane receptor (metabotropic glutamate family)	830	8	6421	12	9682
AAA	PF00004	ATPases Associated with various cellular Activities (AAA)	189	42	7990	79	15111
ABC_tran	PF00005	ABC transporters	192	63	11913	330	64017
ATP-synt_A	PF00119	ATP synthase A chain	175	30	4827	79	12304
ATP-synt_C	PF00137	ATP synthase subunit C	79	25	1884	62	4636
ATP-synt_ab	PF00006	ATP synthase alpha and beta subunits	428	47	16741	183	61617
C2	PF00168	C2 domain	92	34	2965	101	8856
COX1	PF00115	Cytochrome C oxidase subunit I	113	27	3016	80	8782
COX2	PF00116	Cytochrome C oxidase subunit II	234	36	8151	114	22075
COesterase	PF00135	Carboxylesterases	589	27	14433	62	29042
Cys-protease	PF00112	Cysteine proteases	246	36	7974	95	18000
Cys_knot	PF00007	Cystine-knot domain	88	28	2320	61	5108
DAG_PE-bind	PF00130	Phorbol esters / diacylglycerol binding domain	50	34	1677	108	5347
DNA_methylase	PF00145	C-5 cytosine-specific DNA methylases	353	31	10697	57	15915
DNA_pol	PF00136	DNA polymerase family B	845	37	29538	51	33451
E1-E2_ATPase	PF00122	E1-E2 ATPases	683	24	14568	117	62885
EGF	PF00008	EGF-like domain	30	75	2627	676	22710
FGF	PF00167	Fibroblast growth factors	136	10	1305	39	5087
GATase	PF00117	Glutamine amidotransferases class-I	201	39	7468	69	13051
GTP_EFTU	PF00009	Elongation factor Tu family (contains ATP/GTP binding P-loop)	513	63	26793	184	75117
HLH	PF00010	Helix-loop-helix DNA-binding domain	55	35	1882	133	7187
HSP20	PF00011	Heat shock hsp20 proteins	113	52	5630	132	13855
HSP70	PF00012	Heat shock hsp70 proteins	625	34	20365	171	84638
HTH_1	PF00126	Bacterial regulatory helix-loop-helix proteins, lysR family	143	65	9235	101	14331
HTH_2	PF00165	Bacterial regulatory helix-loop-helix proteins, araC family	87	42	3639	65	5655
KH-domain	PF00013	KH domain family of RNA binding proteins	50	20	984	51	2542
Kunitz_BPTI	PF00014	Kunitz/Bovine pancreatic trypsin inhibitor domain	51	44	2258	79	4062
MCPsignal	PF00015	Methyl-accepting chemotaxis protein (MCP) signaling domain	61	10	612	24	1468
MHC_I	PF00129	Class I Histocompatibility antigen, domains alpha 1 and 2	181	25	4465	151	26724
NADHdh	PF00146	NADH dehydrogenases	332	25	7837	61	16402
PGK	PF00162	Phosphoglycerate kinases	425	25	10369	51	20893
PH	PF00169	PH (pleckstrin homology) domain	104	41	4426	77	8256
Pribosyltran	PF00156	Purine/pyrimidine phosphoribosyl transferases	203	26	4857	45	8355
RIP	PF00161	Ribosome inactivating proteins	222	19	4014	37	6529
RuBisCO_large	PF00016	Ribulose bisphosphate carboxylase, large chain	506	17	8131	311	135100
RuBisCO_small	PF00101	Ribulose bisphosphate carboxylase, small chain	126	49	5569	107	12325
S12	PF00164	Ribosomal protein S12	142	23	2913	60	7238
S4	PF00163	Ribosomal protein S4	211	19	3698	54	10542
SH2	PF00017	Src Homology domain 2	81	58	4564	150	11759
SH3	PF00018	Src Homology domain 3	57	62	3500	161	9285
STphosphatase	PF00149	Ser/Thr protein phosphatases	295	17	4956	88	24549
TGF-beta	PF00019	Transforming growth factor beta like domain	108	16	1636	79	7837
TIM	PF00121	Triosephosphate isomerase	257	20	5024	42	9781
TNFR_c6	PF00020	TNFR/NGFR cysteine-rich region	41	51	1942	91	3464

UPAR_LY6	PF00021	u-PAR/Ly-6 domain	144	13	1713	18	2343
Y_phosphatase	PF00102	Protein-tyrosine phosphatase	242	38	9010	122	20901
Zn_clus	PF00172	Fungal Zn(2)-Cys(6) binuclear cluster domain	41	29	1161	54	2159
actin	PF00022	Actins	379	24	8997	160	50267
adh_short	PF00106	Alcohol/other dehydrogenases, short chain type	193	52	9931	186	35727
adh_zinc	PF00107	Zinc-binding dehydrogenases	387	45	15999	129	46025
aldedh	PF00171	Aldehyde dehydrogenases	484	34	15794	69	31826
alpha-amylase	PF00128	Alpha amylases (family of glycosyl hydrolases)	471	54	23611	114	50162
aminotran	PF00155	Aminotransferases class-I	433	29	11778	63	25487
ank	PF00023	Ank repeat	28	83	2338	305	8577
apple	PF00024	Apple domain	86	16	1344	16	1344
arf	PF00025	Arf family (contains ATP/GTP binding P-loop)	184	21	3816	43	7778
asp	PF00026	Eukaryotic aspartyl proteases	341	26	8585	72	21847
bZIP	PF00170	Basic region plus leucine zipper transcription factors	65	22	1396	95	5948
beta-lactamase	PF00144	Beta-lactamases	319	38	10288	51	13654
cNMP_binding	PF00027	Cyclic nucleotide-binding domain	123	32	3797	69	8247
cadherin	PF00028	Cadherin	104	58	5769	168	16875
cellulase	PF00150	Cellulases (glycosyl hydrolases)	333	30	8721	40	11688
connexin	PF00029	Connexin	250	16	3590	40	9005
copper-bind	PF00127	Copper binding proteins, plastocyanin/azurin family	129	31	3268	61	6261
cpn10	PF00166	Chaperonins 10 Kd subunit	96	29	2740	58	5426
cpn60	PF00118	Chaperonins 60 Kd subunit	527	32	16761	84	43948
crystall	PF00030	Crystallins beta and gamma	89	37	3061	103	8479
cyclin	PF00134	Cyclins	273	48	12580	80	19839
cystatin	PF00031	Cystatin domain	108	51	5124	88	8928
cytochrome_b_C	PF00032	Cytochrome b(C-terminal)/b6/petD	107	10	999	133	12724
cytochrome_b_N	PF00033	Cytochrome b(N-terminal)/b6/petB	215	9	1853	170	31806
cytochrome_c	PF00034	Cytochrome c	113	58	5395	175	16925
dsrm	PF00035	Double-stranded RNA binding motif	70	16	1063	22	1470
efhand	PF00036	EF hand	29	86	2493	739	21258
enolase	PF00113	Enolases	445	12	5231	41	15893
fer2	PF00111	2Fe-2S iron-sulfur cluster binding domains	89	18	1580	88	7581
fer4	PF00037	4Fe-4S ferredoxins and related iron-sulfur cluster binding domains.	64	60	3734	156	9529
fer4_NifH	PF00142	4Fe-4S iron sulfur cluster binding proteins, NifH/frxC family	280	16	4334	49	12324
fibrinogen_C	PF00147	Fibrinogen beta and gamma chains, C-terminal globular domain	255	17	4005	18	4166
filament	PF00038	Intermediate filament proteins	352	36	11459	146	41235
fn1	PF00039	Fibronectin type I domain	41	21	802	49	1849
fn2	PF00040	Fibronectin type II domain	42	17	700	37	1540
fn3	PF00041	Fibronectin type III domain	84	109	9219	456	39385
gln-synt	PF00120	Glutamine synthetase	376	35	11836	78	24502
globin	PF00042	Globin	152	62	8876	683	97276
gluts	PF00043	Glutathione S-transferases.	205	61	12034	144	28092
gpdh	PF00044	glyceraldehyde 3-phosphate dehydrogenases	352	23	7672	117	37611
heme_1	PF00173	Heme-binding domain in cytochrome b5 and oxidoreductases	79	16	1238	55	4271
hemopexin	PF00045	Hemopexin	207	14	2682	37	7120
hexapep	PF00132	Bacterial transferase hexapeptide (four repeats)	29	61	1768	82	2376
histone	PF00125	Core histones H2A, H2B, H3 and H4	125	30	3427	178	20412
homeobox	PF00046	Homeobox domain	60	64	3703	385	22470
hormone	PF00103	Protein hormones (family of somatotropin, prolactin and others)	227	17	3631	111	22870
hormone2	PF00123	Peptide hormones (family of glucagon, GIP, secretin, VIP)	28	29	810	110	3068
hormone3	PF00159	Pancreatic hormone peptides	36	15	541	53	1909
hormone_rec	PF00104	Ligand-binding domain of nuclear	165	32	5180	127	20548

		hormone receptors					
ig	PF00047	IG superfamily	47	65	4376	1280	86496
il8	PF00048	Small cytokines (inte-crine/chemokine), interleukin-8 like	70	33	2216	67	4426
ins	PF00049	Insulin/IGF/Relaxin family	88	44	3042	132	8765
interferon	PF00143	Interferon alpha and beta domains	190	17	3190	47	8834
kazal	PF00050	Kazal-type serine protease in-hibitor domain	60	53	2708	155	7814
ketoacyl-synt	PF00109	Beta-ketoacyl synthases	442	11	4648	46	18969
kringle	PF00051	Kringle domain	85	25	1970	126	9931
laminin_B	PF00052	Laminin B (Domain IV)	148	9	1225	15	1749
laminin_EGF	PF00053	Laminin EGF-like (Domains III and V)	53	72	3641	134	6707
laminin_G	PF00054	Laminin G domain	151	26	3858	41	6097
laminin_Nterm	PF00055	Laminin N-terminal (Domain VI)	264	9	2133	10	2376
ldh	PF00056	L-lactate dehydrogenases	335	30	9437	90	23842
ldl_recept_a	PF00057	Low-density lipoprotein receptor domain class A	44	43	1720	98	3914
ldl_recept_b	PF00058	Low-density lipoprotein receptor domain class B	48	23	1007	61	2651
lectin_c	PF00059	Lectin C-type domain short and long forms	128	44	4995	128	15425
lectin_legA	PF00138	Legume lectins alpha domain	49	25	1192	43	2054
lectin_legB	PF00139	Legume lectins beta domain	196	25	4634	40	6241
lig_chan	PF00060	Ligand-gated ionic channels	914	11	9296	30	23453
lipase	PF00151	Lipases	486	16	7284	23	8355
lipocalin	PF00061	lipocalins	156	58	8210	115	15945
lys	PF00062	C-type lysozymes and alpha-lactabulmin	128	21	2607	72	8468
metalthio	PF00131	Metallothioneins	74	21	1313	62	3807
mito_carr	PF00153	Mitochondrial carrier proteins	303	32	9243	62	17310
myosin_head	PF00063	Myosin head (motor domain) (contains ATP/GTP binding P-loop)	703	21	14297	52	29811
neur	PF00064	Neuraminidases	402	7	2729	55	20320
neur_chan	PF00065	Neurotransmitter-gated ion-channel	401	51	23023	145	58059
notch	PF00066	Notch	42	10	378	24	930
oxidored_fad	PF00175	FAD/NAD-binding domain in oxidoreductases	123	56	6534	101	11788
oxidored_molyb	PF00174	Molybdopterin binding domain in oxidoreductases	452	15	6291	35	14281
oxidored_nitro	PF00148	Oxidoreductases, nitrogenase component 1 and other families	457	31	13123	79	23287
p450	PF00067	Cytochrome P450	471	64	28850	204	92743
peroxidase	PF00141	Peroxidases	431	26	8336	55	17313
phoslip	PF00068	Phospholipase A2	128	37	4439	122	13886
photoRC	PF00124	Photosynthetic reaction center protein	323	27	8043	73	22068
pilin	PF00114	Pilins (bacterial filaments)	160	23	3397	56	6250
pkinase	PF00069	Protein kinase	247	67	18184	786	191228
pou	PF00157	Pou domain - N-terminal to ho-meobox domain	78	10	756	47	3359
pro_isomerase	PF00160	Peptidyl-prolyl cis-trans isomer-ases	181	28	4550	50	7966
pyr_redox	PF00070	Pyridine nucleotide-disulphide oxidoreductases class-I	496	23	10665	43	19820
ras	PF00071	Ras family (contains ATP/GTP binding P-loop)	192	61	11753	213	40953
recA	PF00154	recA bacterial DNA recombina-tion proteins	337	31	10244	74	22012
response_reg	PF00072	Response regulator receiver do-main	115	55	6256	130	14711
rhv	PF00073	picornavirus capsid proteins	306	108	27428	117	28207
rnaseA	PF00074	Pancreatic ribonucleases	128	30	3608	71	8635
rnaseH	PF00075	RNase H	157	31	4131	87	11403
rrm	PF00076	RNA recognition motif. (aka RRM, RBD, or RNP domain)	72	70	4979	279	19961
rvp	PF00077	Retroviral aspartyl proteases	113	34	3497	82	8106
rvt	PF00078	Reverse transcriptase (RNA-dependent DNA polymerase)	238	50	11031	147	31069
serpin	PF00079	Serpins (serine protease inhibi-	391	43	16190	105	34597

		tors)					
sigma54	PF00158	Sigma-54 transcription factors	323	41	12923	56	16115
sigma70	PF00140	Sigma-70 factors	236	33	7288	61	13547
sodcu	PF00080	Copper/zinc superoxide dismutases (SODC)	161	29	4411	68	9262
sodfe	PF00081	Iron/manganese superoxide dismutases (SODM)	207	28	5508	69	12029
subtilase	PF00082	Subtilase family of serine proteases	334	43	13142	91	25086
sugar_tr	PF00083	Sugar (and other) transporters	484	51	23055	107	47894
sushi	PF00084	Sushi domain	55	80	4575	346	19872
tRNA-synt_1	PF00133	tRNA synthetases class I	750	19	12728	35	22935
tRNA-synt_2	PF00152	tRNA synthetases class II	363	20	6731	29	10442
thiolase	PF00108	Thiolases	405	24	9375	25	9799
thiored	PF00085	Thioredoxins	113	52	5600	103	10850
thyroglobulin_1	PF00086	Thyroglobulin type-1 repeat	50	22	1011	49	2303
toxin	PF00087	Snake toxins	69	48	3015	172	10570
trefoil	PF00088	Trefoil (P-type) domain	43	28	1190	39	1678
trypsin	PF00089	Trypsin	230	65	14779	246	56153
tsp_1	PF00090	Thrombospondin type 1 domain	52	32	1556	91	4439
tubulin	PF00091	Tubulin	445	26	11230	197	78792
vwa	PF00092	von Willebrand factor type A domain	181	37	6634	50	9024
vwc	PF00093	von Willebrand factor type C domain	69	17	1000	25	1498
vwd	PF00094	von Willebrand factor type D domain	370	6	2133	15	4130
wap	PF00095	WAP-type (Whey Acidic Protein) 'four-disulfide core'	51	18	821	19	861
wnt	PF00110	wnt family of developmental signaling proteins	329	15	4765	105	19453
zf-C2H2	PF00096	Zinc finger, C2H2 type	23	165	3622	1452	31083
zf-C3HC4	PF00097	Zinc finger, C3HC4 type	40	52	2120	69	2796
zf-C4	PF00105	Zinc finger, C4 type (two domains)	77	27	2059	139	10500
zf-CCHC	PF00098	Zinc finger, CCHC class	18	122	2196	188	3384
zn-protease	PF00099	Zinc-binding metalloprotease domain	16	45	656	152	2274
zona_pellucida	PF00100	Zona pellucida-like domain	290	11	2936	26	6914
Total			38622	6300	1149475	22306	3561430

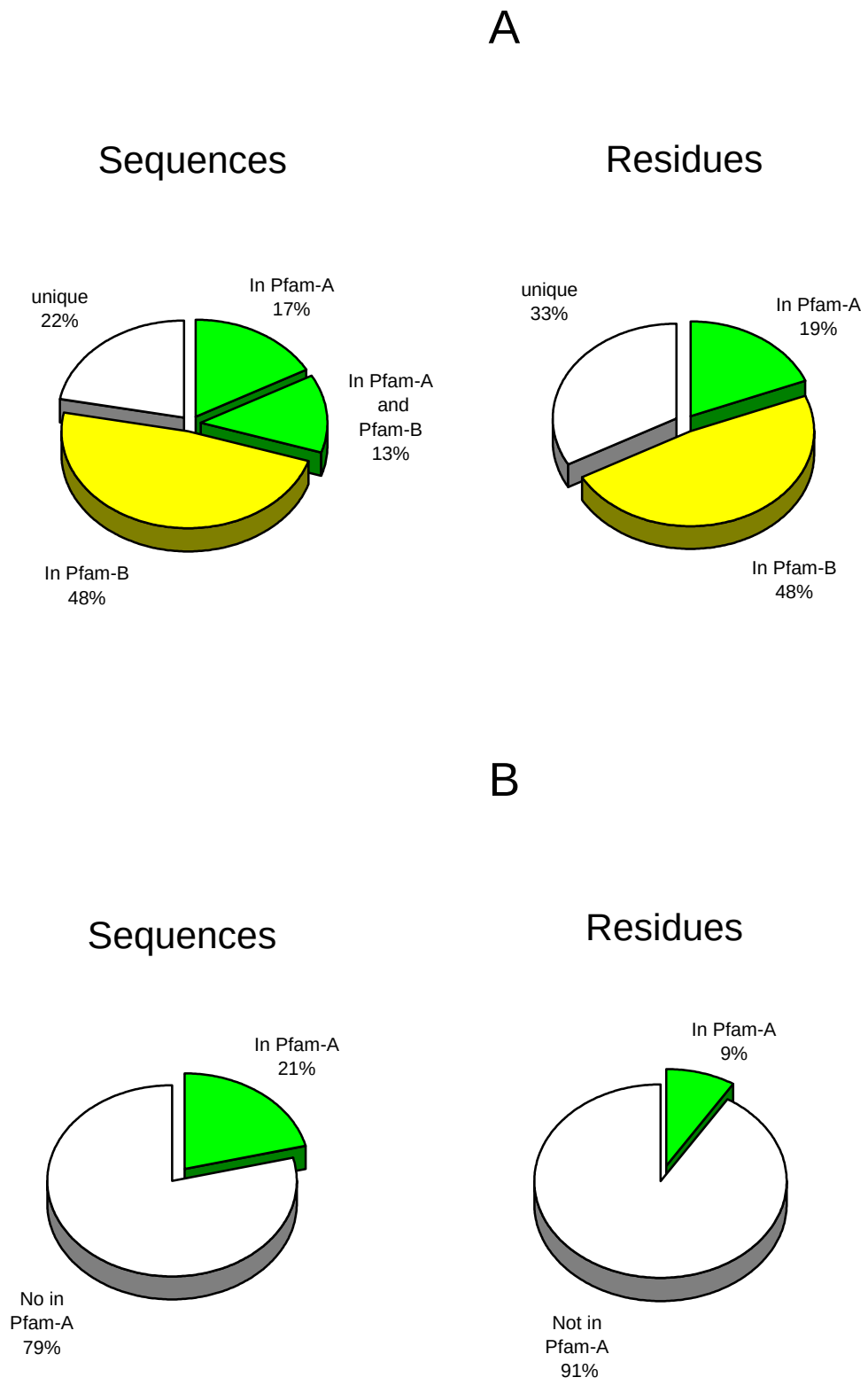


Figure 7.5. A. Proportion of Swissprot 33 in Pfam, based on sequences and residues. The portion of unique sequences is slightly overestimated due to the exclusion of fragments and sequences shorter than 30 residues from Pfam-B. B. Proportion of Wormpep 10, comprising 4874 predicted *C. elegans* proteins that is covered by Pfam matches.

An interesting case of family merging which illustrates the level of clustering in Pfam is shown in figure 7.6. Here two families that were previously not considered related could be merged. One family is the glycoprotein hormones (Prosite: PDOC00234), and the other is a family of connective tissue growth factor-like and C-terminal domains in extracellular proteins [Bork, 1993]. None of these references mention the other family. After we had noticed this family-merger, which gives a good quality alignment (figure 7.6), we learned that the structure of a glycoprotein hormone had recently been determined to be a cystine-knot fold [Lapthorn *et al.*, 1994], which is the fold adopted by TGF-ß2 [Schlunegger and Gruetter, 1993], NGF [McDonald *et al.*, 1991] and PDGF-B [Oefner *et al.*, 1992]. The link between these growth factors and the family had already been made [Bork, 1993], but ironically the sequences of TGF-ß2, NGF and PDGF-B share so few sequence features with the glycoprotein hormones and the other growth factors and extracellular C-terminal domains that they could not be included in the Pfam family.

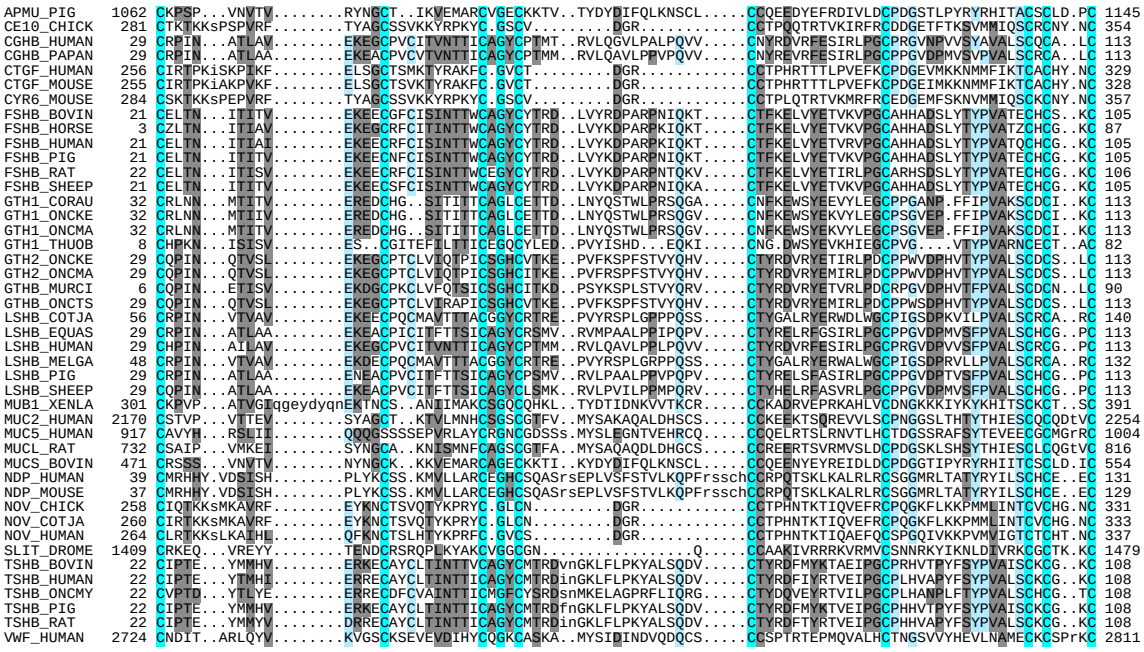


Figure 7.6. The full alignment of Pfam:Cys_knot (accession nr PF0007). This family clusters the two previously described subfamilies CTGF-like (Connective Tissue Growth Factor) and glycoprotein hormones in one single superfamily. The similarity has recently been structurally confirmed.

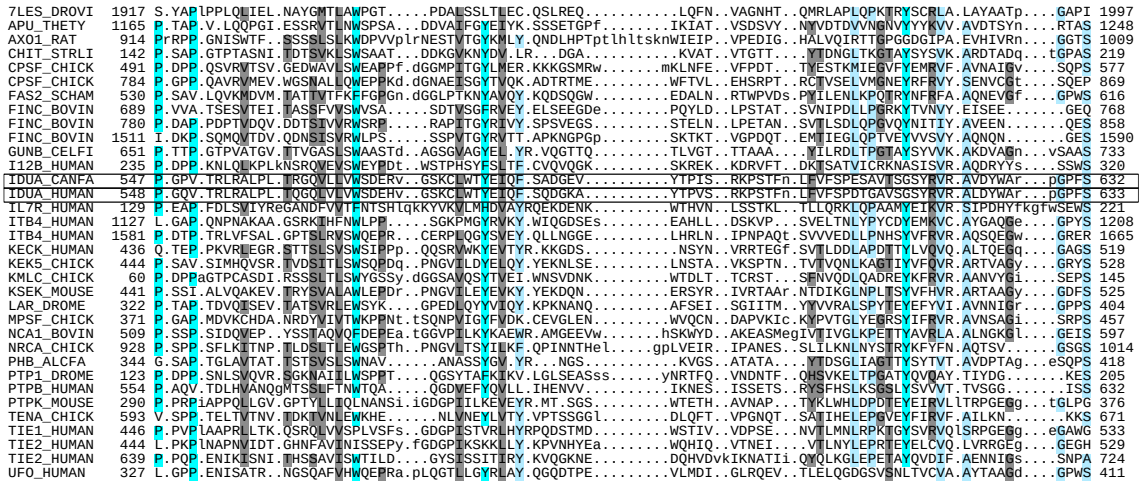
During the construction of Pfam, a number of strong matches were found that despite good sequence similarity had not been classified as true members before. The alignment in figure 7.3B and C contain two examples of this. This domain is usually found as a single N-terminal domain in response regulators of two-component systems, where it receives a signal by phosphorylation by a sensor molecule. The signal is then usually transduced to a C-terminal DNA binding transcription factor which turns on the expression of a set of down-stream genes. Sometimes the receiver domain is not combined with any other domains on the same chain, or is combined with other types of modules, such as kinase domains. The cyanobacterial protein *rcaC* (Swissprot: RCAC_FREDI Q01473), was previously found to have a duplicated receiver domain [Sonnhammer and Kahn, 1994]. We now report a third receiver-like domain between the two previously described ones. Most of the conserved features are still clearly recognisable in this third domain, although it has diverged further from the other two domains. The other novel annotation in figure 7.3B and C is in the yeast protein KFD3_YEAST (Swissprot P43565), which was found as ORF YFL033c by genomic sequencing of *S. cerevisiae* chromosome VI [Murakami *et al.,* 1995]. As seen in figure 7.3C, this protein has a protein kinase domain (split up in two matches) and one receiver domain. In the original analysis, it was only described as “protein kinase”. It further shares domains with the protein CEK1_SCHPO (Swissprot P38938), in the families Pfam-B_9674 and Pfam-B_9675, which in addition also contains one protein kinase domain, but lacks the receiver domain.

Another example is the finding of a new fibronectin type III (FN3) domain [Bazan, 1990] in a mammalian glycohydrolase. FN3 domains have already been found in many bacterial glycohydrolases [Little *et al.,* 1994] [Bork and Doolittle, 1992], but since this domain combination was found to be limited to the bacterial kingdom it was assumed that horizontal gene transfer had taken place from animal proteins with a completely different function. We have detected an FN3 domain in the C-terminal part of human, dog and mouse alpha-l-iduronidase (Swissprot IDUA_HUMAN P35475, IDUA_CANFA Q01634 and IDUA_MOUSE P48441) (see figure 7.7A). The closest homologue is ß-xylosidase from the bacterium *Thermoanaerobacter saccharolyticum*, which lacks the FN3 domain. The discov-

ery of an animal glycohydrolase linked to an FN3 domain raises questions about the conclusion that all FN3 domains in bacterial glycohydrolases have arisen by horizontal transfer of the FN3 domain from an animal source. The alternative scenario is that some ancestral glycohydrolases also possessed FN3 domains.

We have also detected previously undescribed Kazal-type protease inhibitor domains [Kazal *et al.*, 1948] in human and rat organic anion transporters (Swissprot OATP_HUMAN P46721 and OATP_RAT P46720), and in rat prostaglandin transporter (Swissprot PGT_RAT Q00910), as shown in figure 7.8. As far as we know, this is the first time a Kazal domain has been described in transmembrane proteins. From the hydrophobicity profile of these transporters [Kanai *et al.*, 1995], it is clear that the predicted Kazal domain lies in a region of about 90 residues between transmembrane helices 9 and 10. This region was predicted to protrude on the outside of the membrane by the program TopPred II [Claros and von-Heijne, 1994] for both PGT and OATP. This supports the existence of a disulphide-rich globular Kazal domain, which may well be important for substrate binding.

A



B

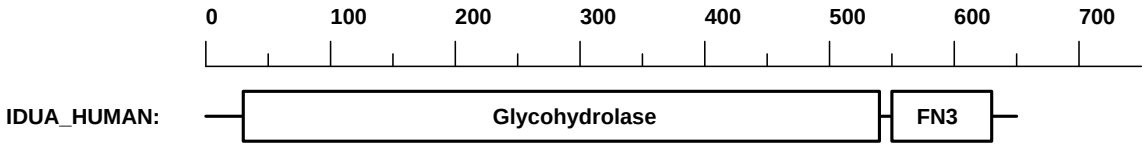


Figure 7.7. A. Selected members of the family Pfam:fn3 (accession nr. PF00041). B. The domain organisation of iduronidase from human and dog (IDUA_HUMAN and IDUA_CANFA), the first examples of a mammalian glycohydrolase combined with a fibronectin type III domain.

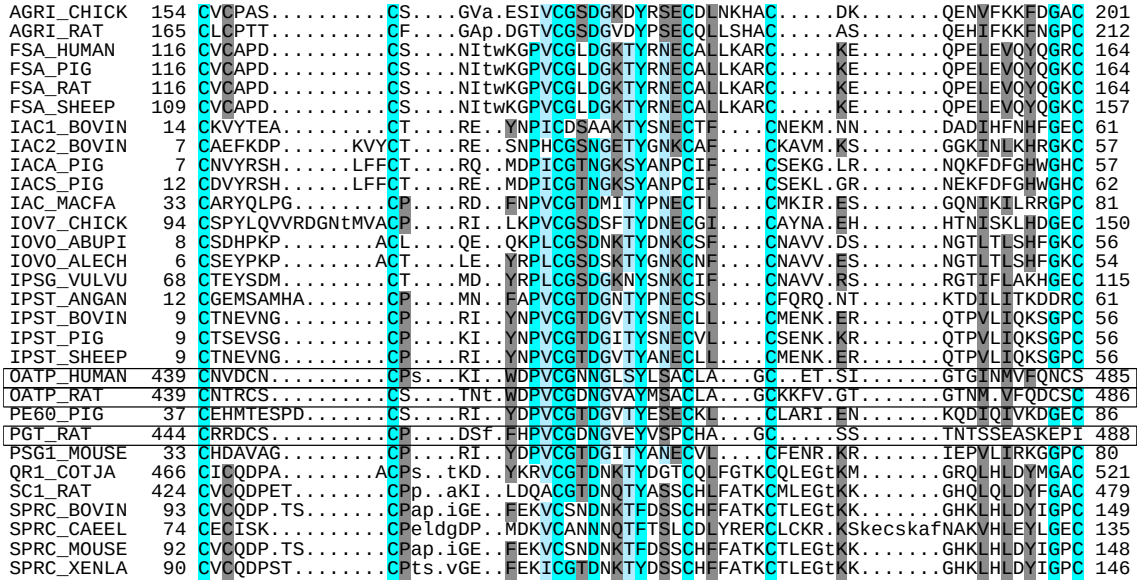


Figure 7.8. Alignment of Pfam:kazal (accession nr. PF00050), showing the novel members OATP_HUMAN, OATP_RAT and PGT_RAT, organic anion and prostaglandin transporters.

To what extent are proteins modular? With Pfam, we can address this problem with higher accuracy than before. Of the proteins in Swissprot 33 containing at least one Pfam-A domain, 17% have two or more domains. This is only a lower bound since (1) not all domains are present in Pfam-A, (2) HMMs are not perfectly sensitive and (3) it is based on proteins in Swissprot, which probably is biased towards single-domain proteins. We have done the same analysis on Wormpep 10, which should represent a relatively unbiased set of proteins. 28% of the proteins that matched Pfam-A families, matched two or more domains. We expect that this number is higher for the nematode *C. elegans* than it would be for single cell organisms. The distributions of both Pfam-A and Wormpep data are shown in figure 7.9.

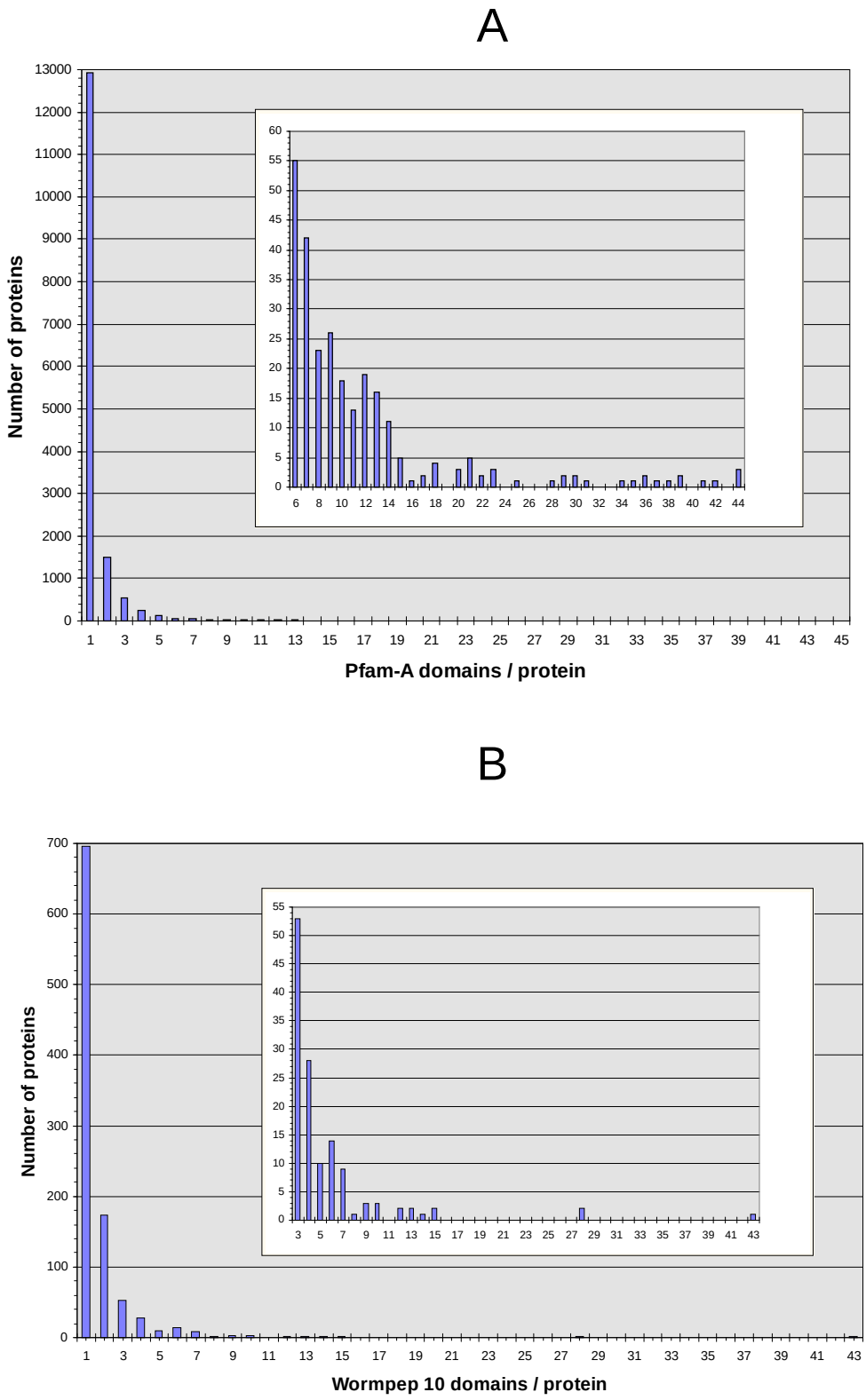


Figure 7.9. Distribution of the number of domains found per protein. A. All proteins in Pfam-A. B. Wormpep 10 proteins that match Pfam-A. The number of proteins with two or more domains is 17% for all proteins in Pfam-A and 28% among Wormpep 10 proteins.

7.5 Discussion

We have presented a database which combines high-quality alignment information with high coverage of known protein sequences. The level of clustering in Pfam-A is largely a result of the sort of alignments we aimed at: full-domain alignments. If subfamilies are too diverse, aligning them together will produce a poor alignment with poor discriminative power. The clusters are thus on a level which gives maximum cluster sizes without disrupting the alignment. In many Pfam-A families the overall sequence similarity is discernible, but not very strong. Clustering at a higher similarity level, like PIRALN [George *et al.*, 1996] where the average family only has 6.7 members (see table 7.2) would give alignments of very tight subfamilies where little evolutionary information is contained. This would diminish the advantages of multiple alignment based search methods like HMMs by rendering them less sensitive to recognising distant members. In Pfam, related subfamilies are generally merged into one family to achieve as diverse clusters as possible without compromising alignment quality.

We have chosen a flat structure of families for Pfam, rather than a hierarchy of clusters. Maintaining a hierarchy of clearly related families would have the advantage of more fine-grained classification. The current clustering of Pfam will often not permit functional inference of a match, since proteins with a common structural origin but diverged functions may be bundled in one family. However, there were a number of reasons not to choose hierarchical clustering. Creating the hierarchy of clusters for each family remains a hard and labour-intense problem, for which no efficient and robust algorithm is known to us. Subgroups of one superfamily would often be very similar to each other, which would significantly increase the complexity of maintaining the families in a non-overlapping manner. Furthermore, using subgroups for similarity searching will increase the search time substantially but preliminary experiments suggest that no significant increase in sensitivity is gained by searching with subfamilies (data not shown).

Table 7.2. Comparison of databases that contain protein family clusters and multiple alignments. [1][Murvai et al., 1996], [2][Gribskov et al, 1988], [3][Seto et al., 1990].

	Pfam-A 1.0	Pfam-B 1.0	ProDom 28.0	PIRALN 11.0	BLOCKS 13.0	PRINTS 10.0
Alignment construction	Manual, Clustal, HMM	Domainer	Domainer	Pileup	Motif	SOPMA
Source database	Swissprot 33	Swissprot 33	Swissprot 28	PIR 48	Swissprot 32	OWL 26
Nr. clusters	175	11929	8031	2059	872	500
Nr. sequences	15,604	31,931	23,048	11,367	18,593	16,231
Nr. residues	3,560,959	8,957,230	6,632,274	4,376,550	1,858,812	1,634,436
Average alignment width	297	180	154	354	32	18
Average cluster size	127	5.7	3.3	6.5	19	37
Cross-referenced to	Swissprot, Prosite, SCOP, Medline	Swissprot	Swissprot	PIR	Swissprot, Prosite	Prosite, Blocks, Sbase[1], Gribskov[2], Kanehisa[3]

It is interesting to compare Pfam clusters to those in Prosite. Although often very similar, they sometimes differ substantially. The reason is that Prosite clusters are usually constructed with a different goal in mind, i.e. describing very short motifs important for function. Prosite clusters therefore tend to include as many members as possible without destroying the pattern. The level of Prosite clustering thus depends on how well a pattern can be developed, which in turn depends on the conservation characteristics throughout the family. In some cases, several Prosite families are merged together into one Pfam family. For instance Pfam:lipocalin contains the members of both Prosite:PDOC00187 (lipocalin) and PDOC00188 (Cytosolic fatty-acid binding proteins). In other cases Pfam extends Prosite families with new members, e.g. Pfam:Cys_knot contains both Prosite:PDOC00234 (Glycoprotein hormones beta chain) and cystine knot domains from mainly growth factors and extracellular proteins (figure 7.5). Prosite families are often overlapping in the sense that one family corresponds to most members, but additional subfamilies are needed to find all members of divergent subfamilies. For example, there are four Prosite patterns for protein kinases

(PDOC00100, PDOC00212, PDOC00213 and PDOC00629), but only one Pfam HMM is needed. On the other hand, families that share only a tiny motif of only a few residues, like e.g. the P-loop [Saraste *et al.*, 1990] (defined in Prosite PDOC00017 as [AG]xxxxGK[ST]), are not merged in Pfam if there is no inter-family similarity beyond the common motif. Often such patterns are in any case too short to discriminate true matches from false, as is the case for the P-loop. Pfam-A 1.0 contains some 35 families that are absent from Prosite, possibly because no discriminative pattern could be found. Some of these families are currently being added to Prosite as 'matrix' entries instead of patterns [Bairoch *et al.*, 1996].

The protein family databases Prints [Attwood and Beck, 1994] and Blocks [Henikoff and Henikoff, 1994] are both based on a set of short ungapped blocks of aligned residues to describe each family. While the Blocks alignments were generated automatically for all Prosite families, Prints was constructed using a more manual approach to define the family clusters, similar to the Pfam member gathering step (see figure 7.1). Hence Prints also contains many clusters that are either absent from Prosite, or have a different clustering level. The ungapped block approach has the advantage that robust and fast methods can be used both to discover conserved regions within a family and to search a database for more members [Neuwald and Green, 1994]. By not allowing gaps, hard to align regions that could easily cause misalignments are avoided. However, gaps also occur in conserved regions, and not allowing them may cause either misalignments or truncation of the domain. The principal practical difference from Pfam's approach is that PRINTS and BLOCKS contain short conserved regions, whereas Pfam alignments represent complete domains, facilitating automated annotation.

Prodom is a protein family database that was entirely generated by the Domainer program [Sonnhammer and Kahn, 1994] purely from pairwise sequence homology data with no human knowledge to guide clustering or domain boundary definition. It is useful as a catalogue of comprehensive low-quality alignments, but the quality of the alignments and clusters is generally too low to produce information-rich HMMs. Unfortunately, the quality is inversely proportional to the number of family members and very poor for short domain families. For

instance, nearly all zinc finger domains were lost due to the crude ‘edge trimming’ of domain boundaries.

There are a number of other databases that contain valuable aspects of protein family classification but were excluded from the comparison in table 7.2 for a variety of reasons. For instance, Sbase [Murvai *et al.*, 1996] and the matrix entries in Prosite [Bairoch *et al.*, 1996] do not provide multiple alignments for the families. The structural clustering in FSSP [Holm and Sander, 1996] could in theory be combined with the structure-sequence alignments in HSSP [Schneider and Sander, 1996] to produce a protein family clustering with multiple alignments, but since this is not explicitly provided, and since a wide choice of different clustering levels are supplied, we have not attempted to generate this. The Conserved Regions database [Worley *et al.*, 1995], is only indirectly accessible via the Beauty Blast server on WWW and not as a complete aligned family database. The MBCRR [Smith and Smith, 1990] and Taylor’s [Taylor, 1990] databases were not included since they were based on relatively small datasets and have not been updated for many years.

The seed/full alignment strategy of Pfam was intended make updates easy; our aim is to make a new Pfam release for each new release of Swissprot. To make Pfam an integral part of the analysis process of genomic sequencing project, tools to store and display matches to Pfam families are currently being added to ACEDB [Durbin and Thierry-Mieg, 1996]. This will allow inspection of HMM matches aligned to Pfam seed alignments and significantly improve large scale classification of proteins.

Our results suggest that Pfam is valuable for genomic sequence analysis. The improvement in protein annotation relative to a human expert annotator using an integrated analysis workbench based on pairwise similarities is more than just an increase in percentage annotated proteins. It avoids many problems inherent to single sequence database searching, such as over-reliance on the annotation of the highest-scoring match and misannotation caused by multidomain proteins. Pfam thus significantly reduces the task of annotators, and helps establish a coherent nomenclature.

7.6 Acknowledgements

Many thanks to Sean Eddy for his extensive help and advice during this project, We thank C. Chothia and M. Gerstein for providing the structural alignment of the globin family, E. Birney for the RNA recognition motif alignment and Peer Bork for helpful discussions on the Fibronectin type III and cystine knot domains.

8. Tools for analysis of protein sequences and families

8.1 Summary

Presented in this chapter are a number of graphical tools for protein sequence analysis using the Pfam collection of Protein families. A multiple alignment viewer for X-windows, Belvu, was developed, which can show the match of a query sequence aligned to a Pfam alignment. It is a general purpose tool with flexible colouring schemes based on conservation or residue types, and has simple editing capabilities.

Belvu was integrated with the ACEDB via the PEPmap, which was developed as a general purpose protein sequence feature display tool. It has built-in sequence and hydrophobicity profile displays, and generic columns for any type of feature that can be displayed as a segment. The PEPmap is linked to Blixem for analysis of BLAST matches, to Dotter for dotplot analysis, and to the Pfam World Wide Web server for family browsing.

The Pfam web server also supports HMM searching of a query against all Pfam families, and domain analysis of all Swissprot proteins in Pfam.

8.2 Introduction

The ACEDB DNAmap forms the basis for the graphical genomic sequence analysis workbench described in part I. Although the DNAmap was primarily designed for DNA sequence analysis, it can also be used for protein analysis. For protein homology analysis, this relies on searching protein databases with Blastx, which translates the DNA query. In many cases, it would be better to use a protein sequence map display instead of the DNAmap. For instance, patterns or features in the protein sequence may be split by introns and therefore go undetected or be difficult to visualise. Also, a number of similarity searching programs, particularly those based on dynamic programming, do not work well on translated DNA interrupted by introns.

Furthermore, using the protein instead of the DNA sequence is often more sensitive, since the level of the background noise is lower. Blastp therefore finds more relevant similarities than Blastx. Also, most available profile-based search programs can only be applied to protein queries. To compare a protein alignment to a DNA sequence requires mechanisms for handling introns and frameshifts. Such programs are starting to appear [Birney *et al.,* 1996], but are not in mainstream use yet.

We are particularly interested in profile-based methods, since they allow us to exploit the Pfam collection of protein domain families for genomic classification (see chapter 7). Searching a query against a database of pre-built multiple alignments has the advantages of improved domain identification, which assists the annotation process, and often increased sensitivity.

To use Pfam efficiently for genomic analysis, tools for both pairwise and multiple alignment based analysis were integrated into ACEDB. To this end, a new ACEDB display, the PEPmap, was developed, which functions as an overview map and a launch pad for more specialised tools. These include Blixem, for analysis of Blast matches, Dotter, for dot-plot analysis, and Belvu, for inspection of matches to multiple alignments.

Belvu was developed as a stand-alone multiple alignment viewer. Although a number of such viewers and editors exist [Parry-Smith and Attwood, 1991; De Rijk and Wachter, 1993; Smith *et al.,* 1994], the particular demands for integration with ACEDB and display capabilities motivated the development of a new program. Belvu is also an essential tool for the construction and maintenance of Pfam. Many of its features, such as the simple editing commands, were incorporated in particular for the needs of Pfam.

For occasional Pfam users, a World Wide Web server has been set up. This allows the user to browse documentation and alignments of all Pfam families. Similarity searching against the HMMs (hidden Markov models) of all Pfam-A families is provided, and the Pfam domain organisation of all Swissprot proteins in Pfam can be looked up. Since the Pfam web server acts as a central resource for the Pfam documentation, and provides links to other WWW resources, we have linked Pfam matches displayed in the PEPmap to the corresponding family page on the Pfam web server. This way the workbench for analysing intrin-

sic and extrinsic properties of amino acid sequences is also integrated with external information resources.

8.3 A graphical genomic sequence analysis workbench for Pfam

The components of the workbench, and how they are coupled together is shown in figure 8.1. Blixem and Dotter have been described in chapters 3 and 5, and will not be treated in detail here. Dotter can be used either to analyse the similarity to a particular database match, or to make a self-comparison dot-plot of the protein in question.

When Belvu is called from the PEPmap, previously stored matches to Pfam are added to the Pfam alignment. (It is also possible to call Belvu from ACEDB with the alignment only.) Similarly, the Pfam web server may also use Belvu to display either the alignment only, or with a matching query segment. The link from Belvu to Medline abstracts is only possible for sequences from a database that is linked to Medline, such as Swissprot, PIR or EMBL/Genbank. If this is the case, a WWW browser can be called up with the sequence entry, which in turn is linked to Medline abstracts at http://www3.ncbi.nlm.nih.gov/PubMed.

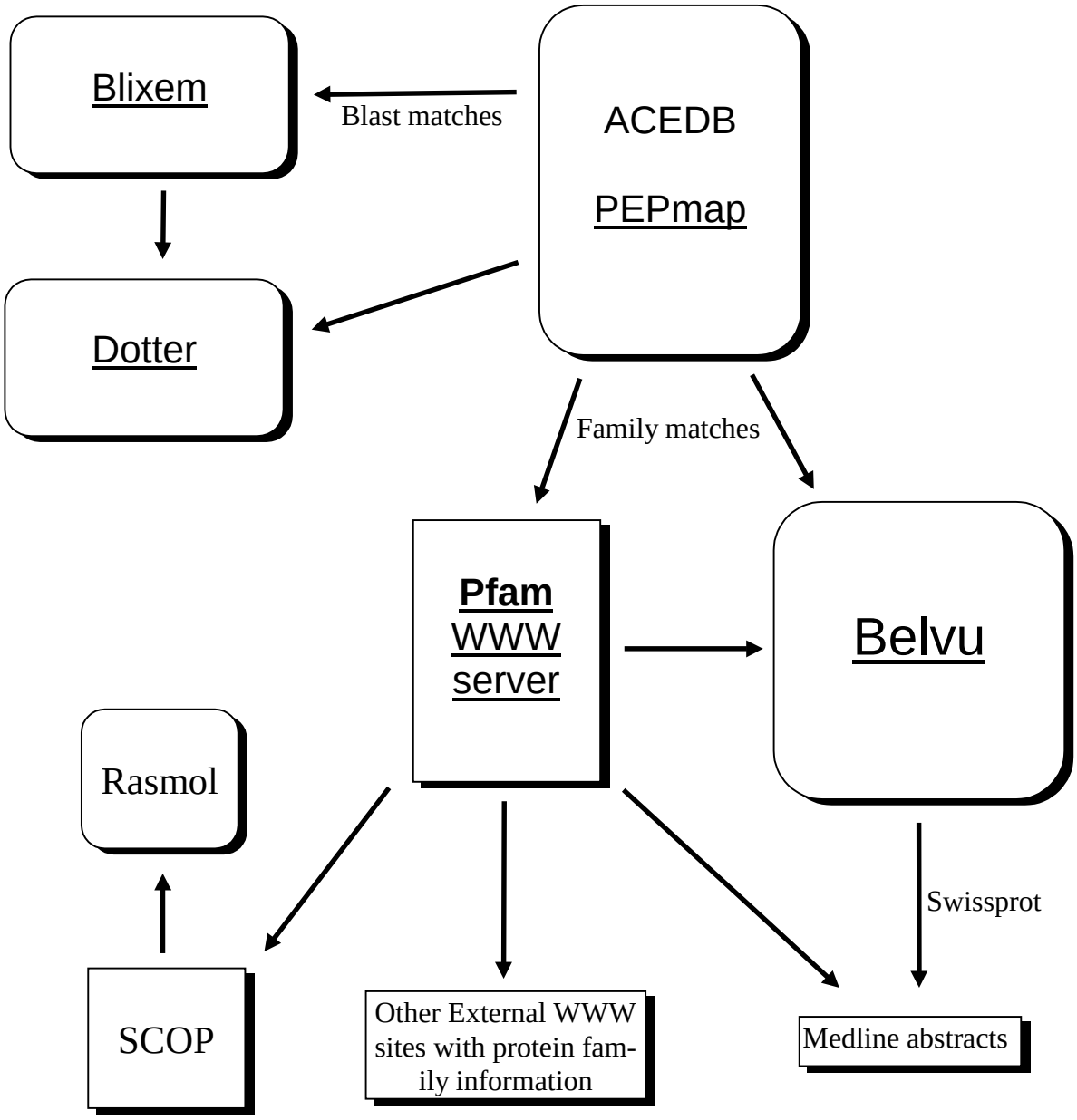


Figure 8.1. Overview of the components in the protein sequence analysis workbench. Rounded corners signifies a graphical analysis tool. Underlined components were developed as part of this work.

BELVU

The general layout of Belvu is shown in figure 8.2. On the left are four columns, containing sequence name, start and end coordinates, and (optionally) score to an HMM or profile. By clicking on a residue, its position in the sequence is displayed on the blue status bar at the top. The row that was clicked also becomes highlighted, and all rows of the same protein become highlighted in the leftmost column. The number of rows (matches) of the picked sequences is displayed on the status bar. If a sequence is highlighted, a crosshair can be activated by pressing the middle mouse button, and dragging to the left and the right can be used to locate a position in the sequence. The alignment is centred at the last position of the crosshair when the mouse button is lifted. Double clicking on a sequence will call Efetch (chapter 6) to retrieve its database annotation, either as a read-only text window, or in a WWW browser.

Belvu's main menu is activated under X-windows by pressing the right mouse button anywhere in the window, except on the buttons at the top, which are separate pull-down menus. The main menu contains general items such as save, print and exit. The pull-down menu 'Edit' contains commands for removing rows or columns, and has functions for making the alignment non-redundant to a user-settable level of identity, removing sequences below a certain score, and to remove outliers. The alignment can be sorted by name or by score (if scores are used).

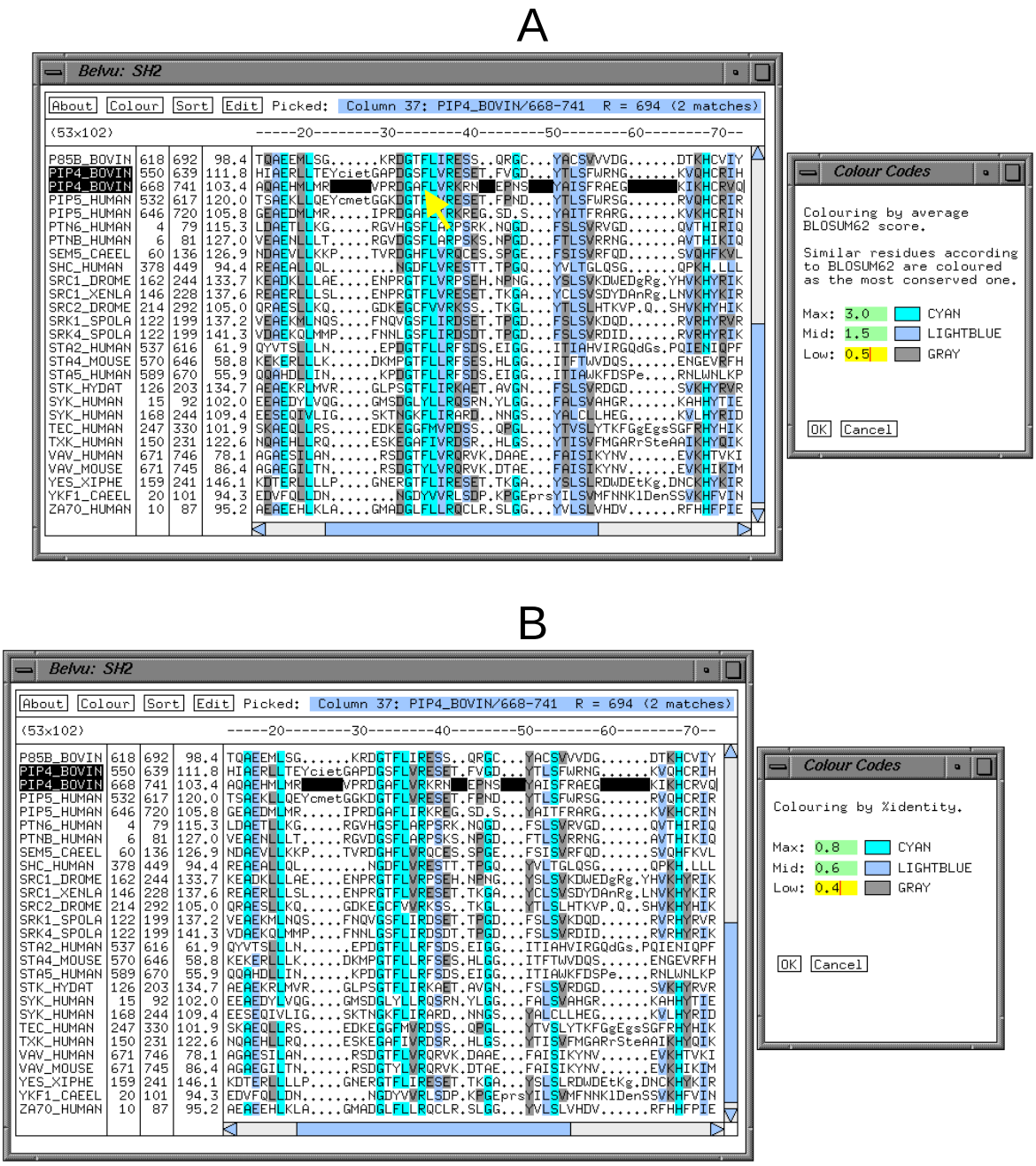
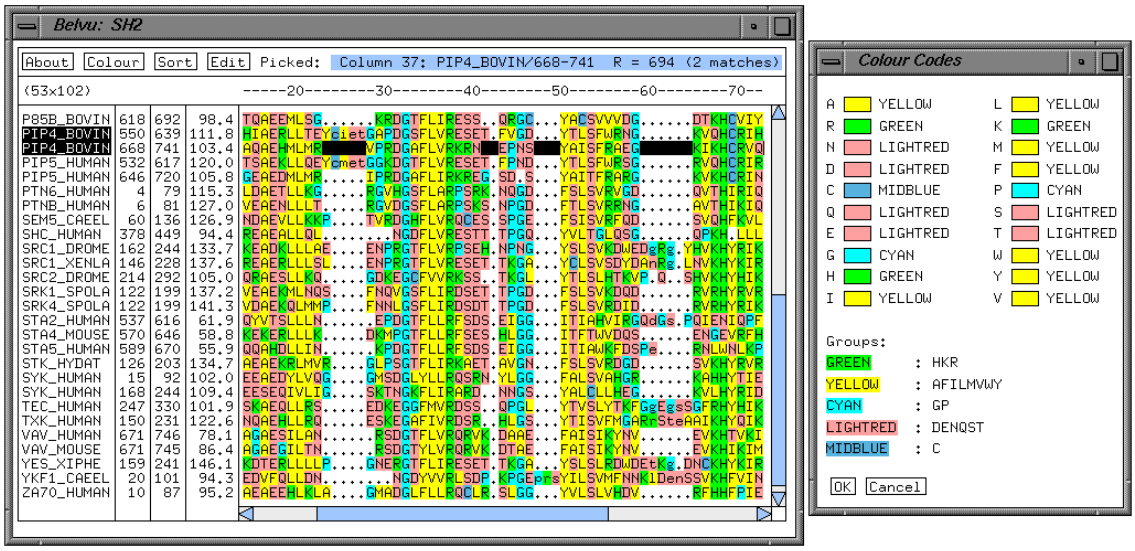


Figure 8.2a-b. The Belvu multiple alignment viewer, showing a the alignment of SH2 domains. The columns are, from the left: sequence name, start of segment, end of segment and score. Clicking on a residue shows the sequence segment and the position of the picked residue in the blue status bar at the top. A. Residue colouring by conservation according to average BLOSUM62 scores. The colours and cutoffs of the three levels are controlled in the 'Colour Codes' window. B. Residue colouring by conservation according to pure percent identity. Colouring by average BLOSUM62 score enhances columns with similar residues. An example of this is column 50, which mainly contains phenylalanines and tyrosines.

C



D

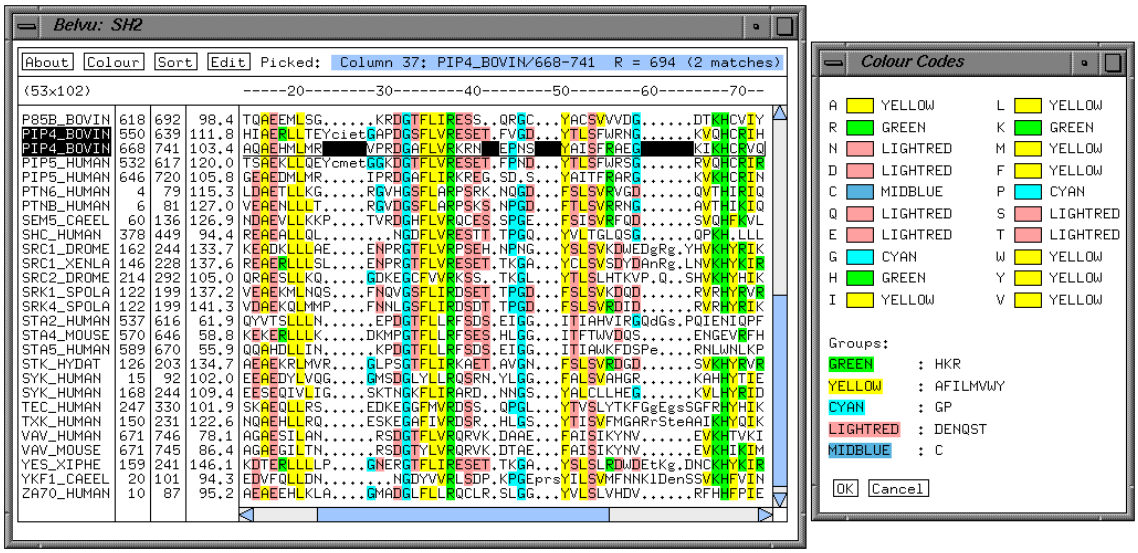


Figure 8.2c-d. C. Colouring by residue type. The colour of each residue is controlled from the 'Colour Codes' window, which also shows the currently defined groups. D. Colouring by residue type, where only residues more conserved than a user-settable identity cutoff are displayed (set to 20% here).

The alignment can be coloured according to a number of different colour schemes, as shown in figure 8.2. There are two main types of colouring: by conservation or by amino acid residue type. Perhaps the simplest way of determining the conservation is by calculating the fraction of rows that contain each residue. There is an option to colour similar residues according to BLOSUM62 [Henikoff and Henikoff, 1992], where the colour of the highest conservation dominates. This does not always give the wanted result. For instance, columns that have four types of hydrophobic residues will be no more than 25% conserved, which normally does not give any colour. To capture columns with a spread of similar residues, Belvu calculates the average pairwise score of all rows [Sander and Schneider, 1991]. This method normally produces intuitive conservation values, and is the default colouring mode. Three conservation levels are supported and the colours and thresholds can be changed interactively in the 'Colour Codes' tool. The thresholds for colouring by average score depends on the score matrix used. For BLOSUM62, average score cutoff levels of 0.5, 1.5 and 3.0 are usually suitable.

If the alignment is incorrect, or other features than conservation are of interest, it is also possible to colour residues by amino acid type. This is particularly useful for analysing fixed sequence patterns or the overall pattern of hydrophobic regions. Belvu has two built-in colour schemes, one of which was suggested elsewhere [Gibson *et al.*, 1994]. It is possible to conceive an arbitrary colour scheme, save it to file, and later load it into a new Belvu session. All residue colours can be selected interactively in the 'Colour Codes' tool. There is also a facility to only colour residues that are more conserved than a user-settable cutoff.

For printing of the entire alignment for publications, Belvu can make a wrapped alignment in a separate window, which only contains the alignment and a title. The current colour scheme is used, and local sequence coordinates are drawn to the left and the right of each line. The user can control the width of the lines and give a title. All alignments in chapters 7 and 9 were printed this way.

Belvu as a standalone program that can read multiple alignments in the MSF, HMMER and Pfam formats. It is available by anonymous FTP at ftp.sanger.ac.uk in /pub/esr/belvu, and on line documentation is available at http://www.sanger.ac.uk/~esr/Belvu.html. For im-

proved efficiency in large scale sequence analysis projects, Belvu has been coupled to the PEPmap display in ACEDB.

The ACEDB PEPmap

ACEDB contains graphical displays for genetic and physical maps, and for DNA sequence features. The latter in particular contains a large number of configurable columns, which can display most types of data as generic segments in a column. Segments from each analysis method are displayed in a separate column, and features such as colour and drawing method to reflect the score (by width or offset) are configurable.

The PEPmap works essentially as the ACEDB DNAmap. It embodies the same generic column philosophy, but also contains some protein-specific columns as well, such as hydrophobicity plots. The main reason to have a separate PEPmap, apart from modularity considerations, is that many features in the amino acid sequence are difficult to display when they are split by introns. In this chapter, we show how two types of protein similarity data can be stored in ACEDB and displayed in the PEPmap: Pairwise matches form BLAST, and HMM matches to Pfam families.

Figure 8.3a shows how the PEPmap and the integrated analysis tools can be used to analyse a newly determined protein sequence. The blue boxes (Blastp matches) and the green boxes (Pfam HMM matches) have shortcut menus. By pressing the right mouse button one of these boxes, a menu gives a choice of Blixem, Dotter or Belvu (Pfam HMM matches only) analysis. Pfam matches are also passed on to Blixem and Dotter, where they are drawn as green boxes. Blixem and Dotter are normally linked in with the ACEDB executable, while Belvu is spawned as a separate process. It is immaterial whether a viewing tool is external or internal, as long as it does not need to communicate with the ACEDB database, for which it has to be internal. Blixem can thus be used to retrieve a sequence annotation with the internal ACEDB object display, while Belvu can only retrieve it from an external database using Efetch. Using the analysis tools as external programs makes the system more modular, and more easily maintainable, however. Data is passed from ACEDB to external programs via

UNIX pipes. Blixem can also be called as an external program, which has the advantage of allowing multiple simultaneous sessions.

For proteins with matches to Pfam families, annotation and 3D structures can be accessed via the Pfam WWW server (see below). This is also an item on the shortcut menu on the (green) Pfam HMM match boxes in the PEPmap. Each family has a page in the web server, which is linked to other databases such as Prosite and Medline for documentation, and to the SCOP database [Murzin *et al.*, 1995] for structural information. The SCOP WWW server provides a hierarchical classification of the protein fold, and 3-dimensional views of the structure using either static images from the Expasy WWW server [Appel *et al.*, 1994] or RasMol [Sayle *et al.*, 1995]. An example of this is shown in figure 8.3b.

The PEPmap uses a set of generic software modules in ACEDB for drawing and controlling maps. The appearance of the columns is controlled in two ways. An object of the class 'Method' that is associated with each segment, determines in which column it is drawn, and some general features of the segments, such as the colour and the linking to analysis tools. Each column can also be configured to some extent by clicking on the green horizontal bar at the bottom of the column, which brings up a configuration window (see for example the hydrophobicity plot configuration window in figure 8.4). The general layout of the PEPmap, i.e. which columns are turned on and their relative order, is stored in a 'view'. The view is controlled in the 'Column Control' window (see figure 8.4). A particular view can be stored and be re-used at a later stage.

A

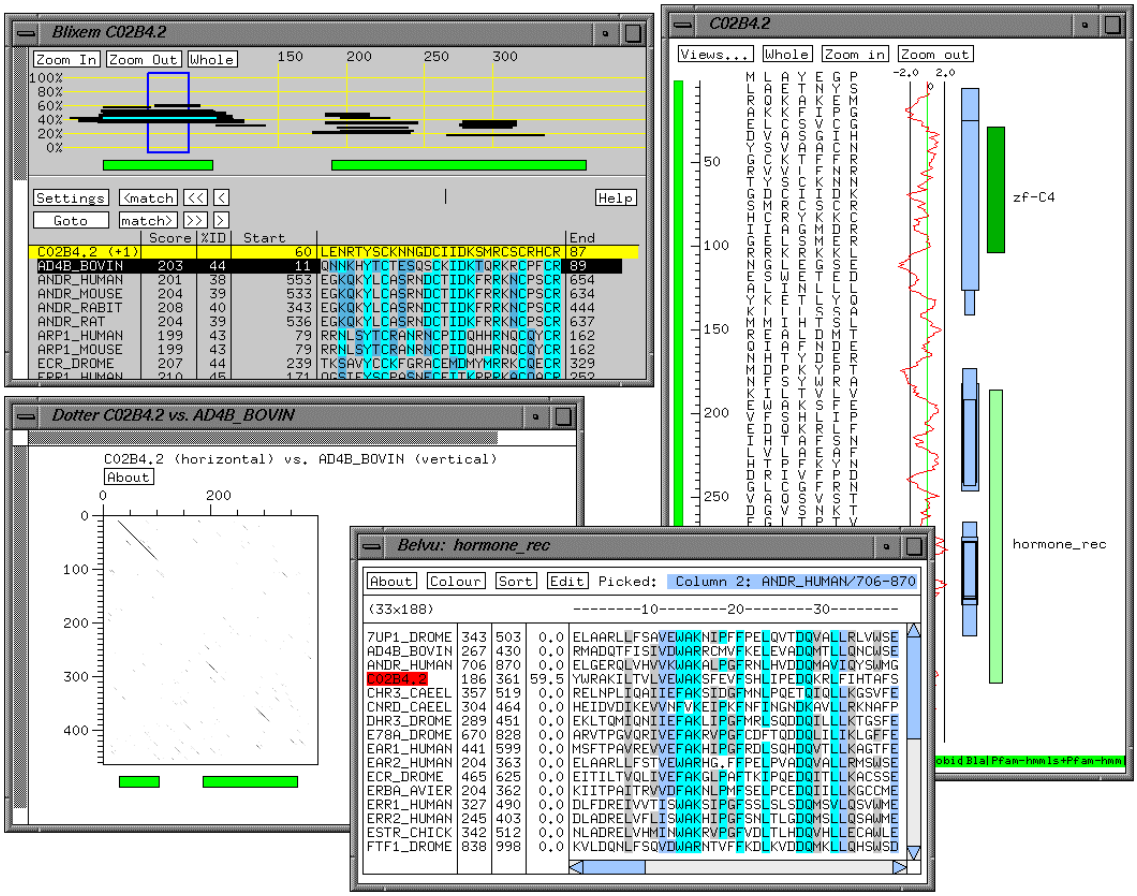


Figure 8.3a. The main graphical tools of the analysis workbench are linked together. This example shows how the *C. elegans* protein C02B4.2 can be analysed in the ACEDB PEPmap (top right), in Blixem and Dotter for pairwise protein comparison, and in Belvu for family comparison. The PEPmap shows the sequence, hydrophobicity plot, Blastp matches (blue boxes) and Pfam matches (green boxes). This protein matches the Pfam family of C4 type zinc fingers in the N-terminal part, and the family of ligand binding domains of hormone receptors in the C-terminal part.

B

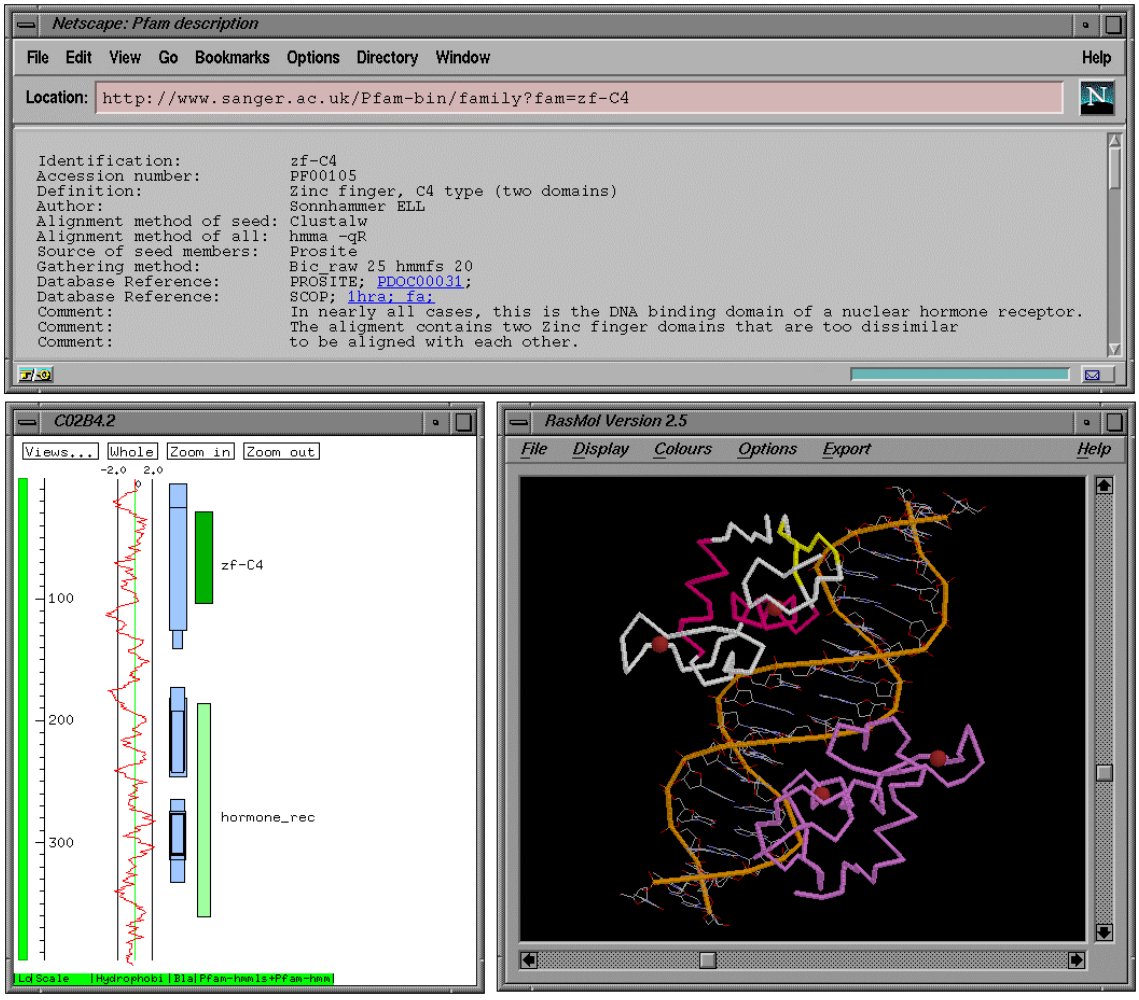


Figure 8.3b. The Pfam WWW server can act as a link between the PEPmap and protein structures, since the Pfam web pages are linked to the SCOP server. SCOP contains structural classification and provides 3D visualisation using RasMol.

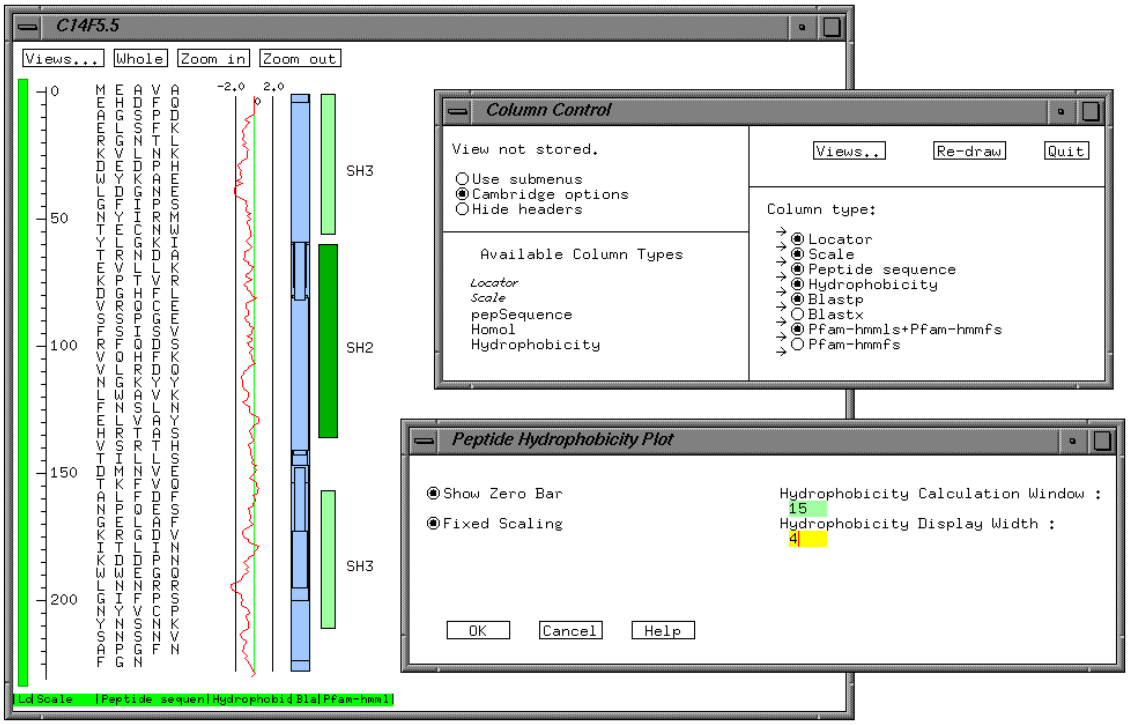


Figure 8.4. Examples of column configuration tools of the ACEDB PEPmap.

Storing matches to proteins and protein families in ACEDB

Proteins are objects of the class 'Protein' in ACEDB, and can store homology information to other proteins and protein families. Blast matches are stored under the 'Pep_homol' tag, which contains the score and extent of a match between two Protein objects (see figure 8.5a). Matches to multiple alignments are stored under the 'Align_homol' tag. Since these contain gaps, the matching segments between the gaps also need to be stored in addition to the start and end coordinates (see figure 8.5b).

The actual sequence of a protein is stored in a separate class 'Peptide', which is linked to the Protein objects. Corresponding Protein and Peptide objects must have the same name. The matching proteins do not require an internally stored peptide sequence for display in Blixem: if it is not present, Blixem will try to Efetch it. Multiple alignments are stored as 'Alignment' objects in ACEDB, and are passed on to Belvu in the Pfam format.

Matching query segments are appended to Belvu's input stream, and any insertions in the alignment are made by Belvu. We normally store the match coordinates relative to the Pfam seed alignment, which have to be calculated from the HMM match coordinates and the mapping between the HMM and the alignment it was derived from. Alternatively, one could store match coordinates relative to the HMM consensus in ACEDB. Although this would be more compact, it would require storing the HMM-alignment mapping in ACEDB, and the alignment coordinates would have to be reconstructed every time Belvu was called.

Two UNIX scripts, hmmPfam and belvuMatch, were developed to automate the searching of queries against Pfam and converting the output to .ace format for input into ACEDB (see figure 8.5). HmmPfam uses the HMM searching programs in the HMMER package [Eddy, 1995a]. ACEDB is available by anonymous FTP at ftp.sanger.ac.uk in /pub/acedb. A database containing Pfam and HMM matches to *C. elegans* proteins is also available (see section Pfamace below), which contains all the mentioned class models. The scripts hmmPfam and belvuMatch are also included.

A

```
Protein : C02B4.2
Pep_homol  YKC8_CAEEL BlastP 286.0 27 105 14 92
Pep_homol  YKC8_CAEEL BlastP 119.0 182 246 193 257
Pep_homol  YKC8_CAEEL BlastP 73.0  265 333 268 336
Pep_homol  HNF4_HUMAN BlastP 234.0 30 118 50 138
Pep_homol  HNF4_HUMAN BlastP 58.0  192 227 186 221
Pep_homol  HNF4_HUMAN BlastP 86.0  274 314 257 297
Pep_homol  RRXA_MOUSE BlastP 244.0 30 111 139 220
Pep_homol  RRXA_MOUSE BlastP 56.0  182 213 271 302
Pep_homol  RRXA_MOUSE BlastP 61.0  276 311 354 389
Pep_homol  AD4B_BOVIN BlastP 203.0 29 107 11 89
Pep_homol  THAB_XENLA BlastP 105.0 29 62 59 92
Pep_homol  THAB_XENLA BlastP 112.0 64 96 96 128
```

B

```
Protein C02B4.2
Corresponding_DNA C02B4.2
DB_searched Pfam 1.0
Align_homol hormone_rec Pfam-hmmls 59.48 186 361 1 178 Segs  1 55 \
  1 55 56 66 57 67 67 77 70 80 88 120 81 113 121 127 122 128 130 \
154 131 155 157 176 159 178

Protein C02B4.2
Corresponding_DNA C02B4.2
DB_searched Pfam 1.0
Align_homol zf-C4 Pfam-hmmfs 130.76 29 104 1 80 Segs  1 36 1 36 37 \
44 39 46 45 76 49 80
```

Figure 8.5. Examples of Blastp (A) and Pfam (B) matches in .ace format, ready to be read in to ACEDB for display in the PEPmap. The tags (fields) of the Pep_homol and Align_homol lines are: <matching object> <method> <score> <query start> <query end> <subject start> <subject end>. Since Align_homols are gapped alignments, this is followed by a list of all the matching segments in the form <query start> <query end> <subject start> <subject end>.

8.4 Public access to Pfam

World Wide Web

There are two Pfam web servers: at http://www.sanger.ac.uk/Pfam (Cambridge, UK), and http://genome.wustl.edu/Pfam (St. Louis). All figures here are taken from the Cambridge server. Figure 8.6 shows the home page and an example of a family page. Clicking on 'Browse families gives a list of the names and short descriptions of all families. Clicking on a family gives a page like in figure 8.6b, from which the full and seed alignments, and other information can be accessed. The alignments can either be displayed as text in the browser, or be viewed in Belvu, which can be run either locally or at the Sanger Centre. To install Belvu for local use from the web server, a special MIME type 'x-belvu' has to be defined. (Instructions are provided on the web page.)

Figure 8.7. illustrates how the server can be used for HMM searching of a query sequence versus the Pfam-A HMMs. At present, this facility is available on the Cambridge server only. Setting the score cutoff and the search method to default will generally only find clear similarities, that often could be found with other methods. For more sensitive searching, the cutoff should be lowered to 15 or 10 bits or even less, and both hmmls and hmmfs should be tried. Note that the example in figure 8.7 contains two 'half' PH domains, which are only found when the score cutoff is lowered below 7 bits, using the hmmfs program. These matches would each be marginal by themselves, but the alignment indicates that a whole PH domain indeed has been split up by the SH2 and SH3 domains. The search results are summarised in a table and in a graphical schematic. For each match, links to the Pfam family page and to the alignment of the match are provided. The alignment can be viewed either as a pairwise alignment of the query and the HMM consensus sequence in ASCII format, or in Belvu, with the matching segment aligned in the Pfam seed alignment.

The Pfam web server also allows inspection of the domain organisation of all proteins in Swissprot that are part of Pfam. Each protein is stored as a line-drawing schematic of the segments corresponding to Pfam families, as shown in figure 8.8. Pfam-B segments are also included. Clicking on a Pfam-A family leads to the WWW page of that family, while click-

ing on a Pfam-B family displays that family as text in the browser. When Pfam alignments are displayed in the browser, the sequence names are linked to these schematics, which makes it relatively easy to get a picture of the modularity of a particular family. All of these pages are based on marking up text retrieved by Efetch.

Anonymous FTP

A number of flat format and other files pertaining to Pfam are available by anonymous FTP at ftp.sanger.ac.uk in /pub/databases/Pfam. All files are compressed by gzip.

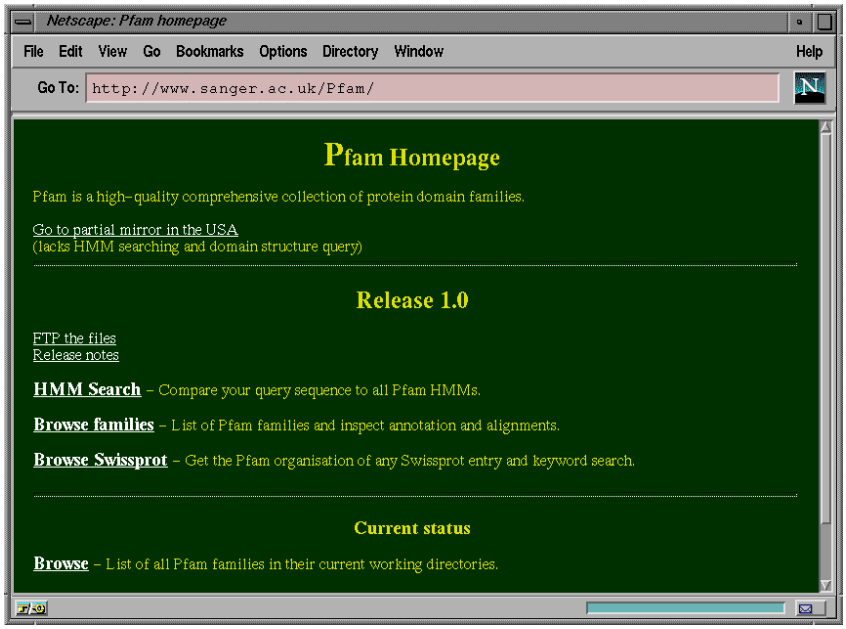
The flat file 'Pfam' contains all annotation and seed and full alignments as ASCII text. The Pfam format follows the Swissprot syntax, with the addition of AU (alignment author), SE (seed membership source), AL (seed alignment method), GA (gathering method to find all members) and AM (alignment method of all members to HMM) fields. The format of the multiple alignment is for each sequence segment: name/start-end padded sequence Swissprot accession number, all on one line.

The file 'swissPfam' contains the schematic line-drawings of the Pfam domain organisation of all Swissprot proteins in Pfam.

The HMM files are released separately in binary HMMER format in the file 'hmmPfam.tar', which also contains a simple script to search them using the Pfam default cutoffs and search programs. The HMM search programs are available separately [Eddy, 1995a].

For ACEDB curators that want to incorporate Pfam analysis in their system, a sample ACEDB database is released in the file 'Pfamace.tar'. This contains all the Pfam families stored internally and examples of proteins with Pfam matches from *C. elegans*. The ACEDB executable contains a fully functional version of the PEPmap

A



B

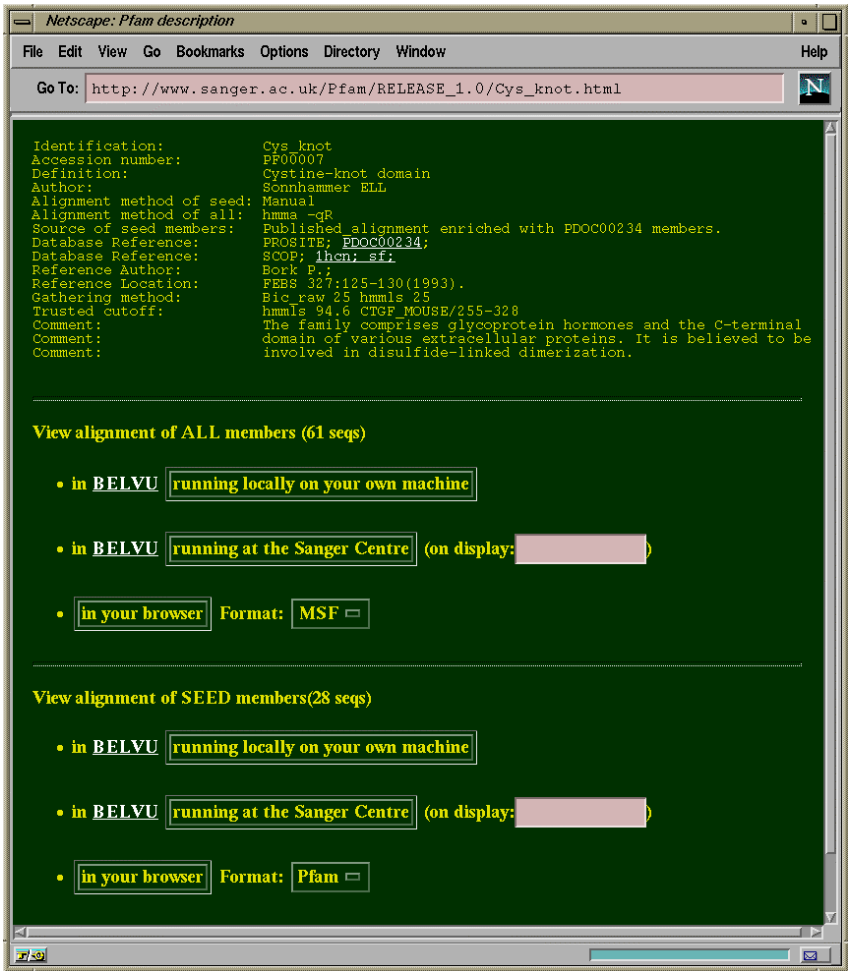
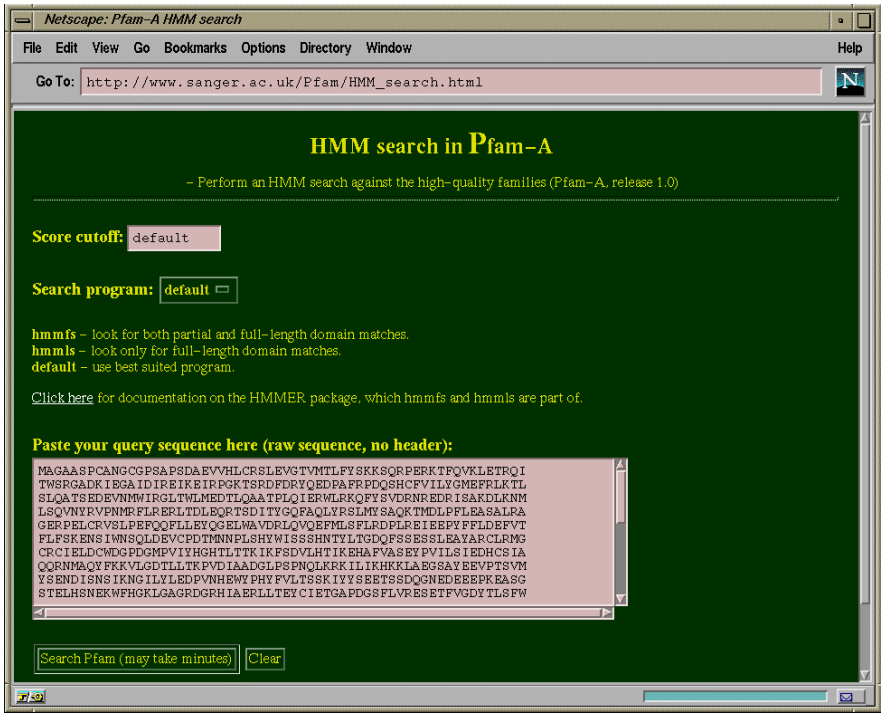


Figure 8.6. The Pfam WWW server in Cambridge. A. The home page. B. An example of a Pfam family page.

A



B

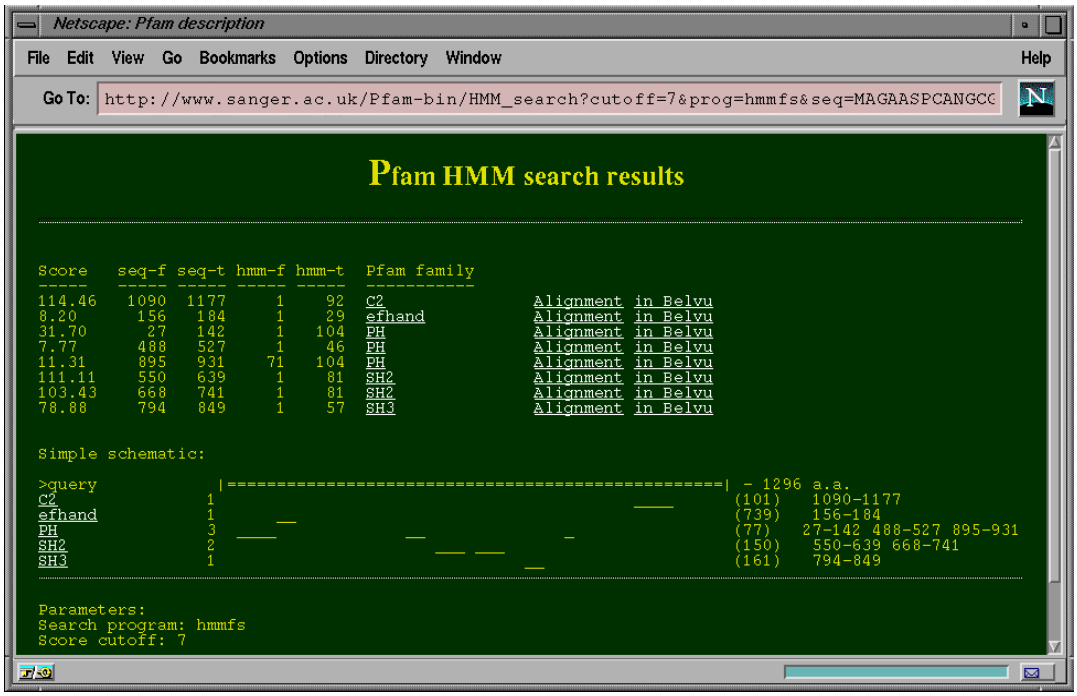
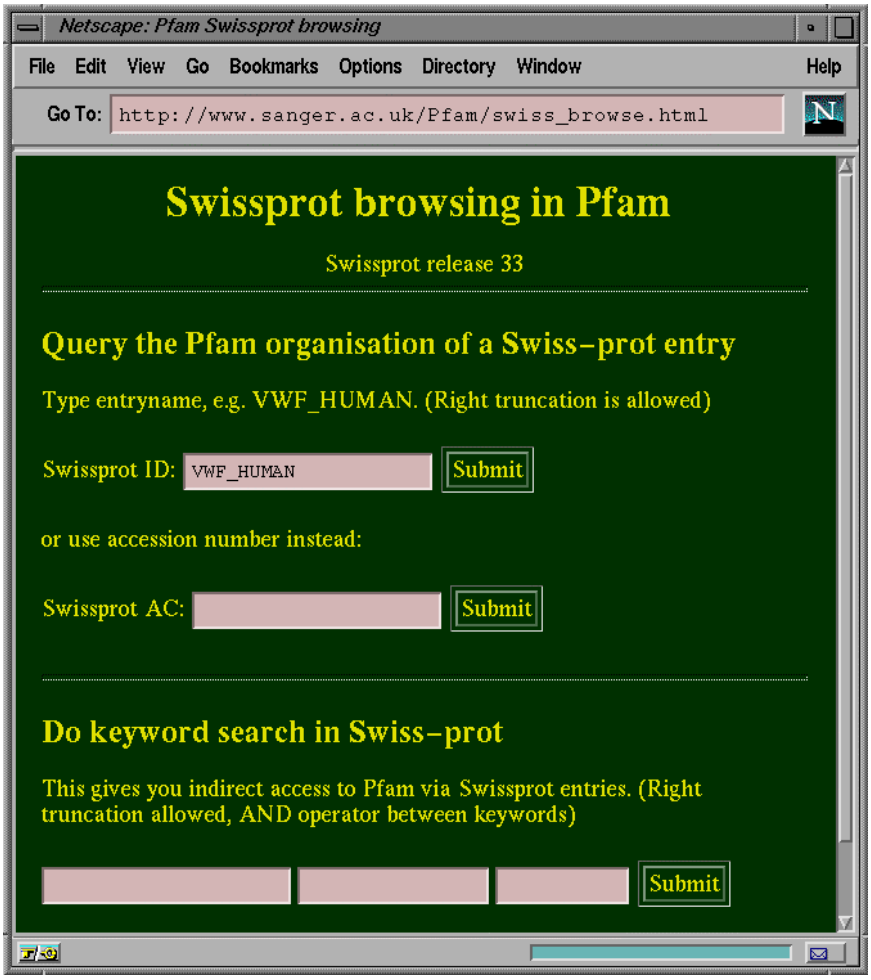


Figure 8.7. HMM searching of a query sequence versus Pfam on the Pfam WWW server. A. The query dialogue page. B. The results of an HMM search. The table of matches at the top is translated into the schematic at the bottom. Clicking on the 'alignment' lines returns the pairwise alignment of the HMM consensus sequence and the query. Clicking on the 'in Belvu' part calls Belvu with the matching query segment incorporated in the multiple alignment of the Pfam seed.

A



B

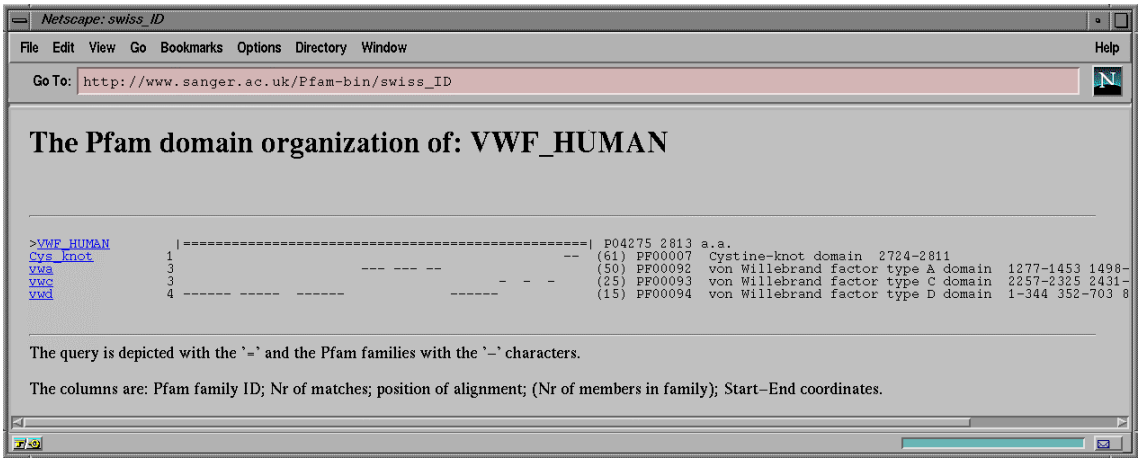


Figure 8.8. Swissprot browsing on the Pfam WWW server. The domain organisation of all proteins in Swissprot that are part of Pfam can be inspected schematically by entering their name or accession number. The blue lines are linked to WWW pages of the full Swissprot entry and the Pfam families.

8.5 Discussion

Belvu was primarily intended as an interactive tool. Its capability for printing alignments relies on colour printing, and is relatively primitive compared to e.g. Alscript [Barton, 1993a], which can use different fonts to highlight certain residues, and has a range of options for drawing consensus sequences and histograms. Belvu can of course display pre-calculated consensus sequences in the alignment, but a more flexible system that can display consensuses at different levels of conservation would be preferable.

The ACEDB PEPmap is currently an adequate platform for managing matches to database sequences and families, and for launching the analysis tools Blixem, Dotter and Belvu. The PEPmap as an analysis tool in its own right should be seen as a prototype which is still under development however. A number of improvements, such as flexible colouring and dynamic scaling of the amino acid sequence, hyperlinking of boxes representing other objects, and general layout aspects are currently being worked on. We are also investigating ways to link to the PEPmap to other external programs through a generic interface, to make it a general purpose display tool. This would be more flexible and efficient than adding new algorithms by hard coding them in the program. Any external program that can analyse a protein sequence and produce segments containing a particular type of information could in principle be engaged. For instance, predictions of secondary structure or transmembrane helices would be very useful tools to link in.

8.6 Acknowledgements

I am grateful to Simon Kelly for the basic map drawing and control routines in ACEDB and Clive Brown for work on the PEPmap.

9. Analysis of protein domain families in *C. elegans*

9.1 Summary

The *C. elegans* genome sequencing project has completed over half of this nematode's 100 Mb genome. Proteins predicted in the finished sequence are compiled and released in the database Wormpep. Presented here is a comprehensive analysis of protein domain families in Wormpep 11, which comprises 7299 proteins.

Common domains in Wormpep proteins were analysed by comparison to the Pfam collection of protein families, which is based on recognition by hidden Markov models. This identified a number of previously unannotated domains, and is a valuable complement to manual classification.

To investigate new protein families, Wormpep was clustered using several methods, which were compared to each other. Some of the new clusters were analysed in detail to assign a putative function despite lack of clear homology.

Finally, the proteins in the *C. elegans* are compared to proteins in the human, *S. cerevisiae* and *H. influenzae* genomes.

9.2 Introduction

Complete genome sequencing produces data that opens up many new areas of investigation. One of them is the analysis of all the proteins encoded in a genome. Knowing the complete set of proteins is important for studies of protein evolution and function, and for comparison of the proteins present in different organisms. This chapter addresses perhaps the most immediately interesting aspects of genome-scale protein sequence analysis: the systematic functional classification and characterisation of protein families.

Functional classification exploits the pre-existing annotation of homologous protein sequences with a known function. To carry over the annotation from other proteins, the similarity should be analysed by careful manual inspection, which for genome-scale projects preferably should be assisted by an integrated analysis/gene prediction workbench. This is important both for efficiency reasons and because the sequence similarity may influence the gene prediction (see part 1 of this thesis).

A different approach, which is not as comprehensive but is generally less ambiguous regarding the extent of homologous domains, is to search a database of pre-assembled multiple alignments of protein families. An example of such a database is Pfam, described in chapter 7. These families may also provide a more general level of annotation. This approach has been employed here to classify the proteins predicted so far in the *C. elegans* genome.

It is a principle of protein evolution that new protein functions can arise by the duplication of a gene and subsequent specialisation of the 'daughters'. In these cases of course the detailed functions are different, although the mechanisms may be the same. In fact, the majority of proteins in higher eukaryotic genomes, and 30-50 percent in prokaryotes [Brenner *et al.*, 1995; Koonin *et al.*, 1995] have clearly recognisable 'siblings' that are products of gene duplication. Such homologues are called paralogues, while proteins in different organisms that diverged due to speciation, and which normally have identical functions, are called orthologues.

To study groups of similar proteins within a genome, they first need to be clustered into families of paralogues. Many clustering methods are known (see e.g. [Romesburg, 1989]), but only a few are appropriate for protein sequences. Additional complexity for clustering proteins is caused by sequences that contain more than one protein domain, hence belonging to more than one family, and the lack of certainty in defining the relationships [States, Harris and Hunter, 1993]. At present, no clustering algorithm can solve all these problems without compromise. A choice has to be made between using a simple algorithm which ignores the nature of protein sequences, and one that tries to resolve the problems, but has other side-effects. The choice depends on the set of proteins (prokaryotic proteins seldom contain multiple domains) and what the purpose of the clustering is. If it is mainly to get an idea of the

number of clusters, particularly in prokaryotes, a simple clustering method might give the best approximation. To cluster a set of proteins from a higher eukaryote, with the aim to use multiple alignments of the clusters, a method that explicitly takes multiple domain proteins into account is necessary, such as the HHS [Hunter *et al.,* 1992] or the Domainer [Sonnhammer and Kahn, 1994] algorithms. This chapter presents results from the clustering of *C. elegans* proteins with Domainer.

Over half the *C. elegans* genome has been sequenced. The predicted protein sequences are periodically compiled and released in the database Wormpep, which is introduced in the first section. Wormpep is then analysed for content of known domains in Pfam, and is clustered in two ways, completely or only regions not matching Pfam. The paralogue clusters are then searched for homology with other proteins, and some of the largest clusters that appear to be unique to *C. elegans* are analysed in further detail. In some cases, this resulted in a tentative functional assignment. Finally, Wormpep is compared to the complete genomes of *Saccharomyces cerevisiae* and *Hemophilus influenzae*, and to a set of human proteins, to investigate the amount of conservation throughout the three kingdoms bacteria, fungi and animalia, and to examine how useful knowledge of the *C. elegans* genome will be for understanding human biology.

9.3 Wormpep - a database of predicted *C. elegans* proteins

All proteins predicted in the sequence produced by the *C. elegans* genome sequencing project are released at regular intervals as the Wormpep database, which is available at ftp.sanger.ac.uk in /pub/databases/wormpep. The data is also available in EMBL and Genbank, as cosmid DNA sequences, and in Swissprot and PIR as proteins. However, Wormpep has a number of advantages. The protein predictions are more up to date, since they are extracted directly from the latest version of ACEDB. During this process a number of quality control checks are carried out to remove erroneous predictions. For example, genes that span two cosmids are correctly represented in ACEDB, but not in the EMBL/Genbank sequence

entries, which makes it difficult to extract the complete protein sequence. A few proteins in Wormpep may still be fragments if they span two cosmids of which only one has been sequenced, or two cosmids that were sequenced in different sequencing centres, since the working ACEDB databases in Cambridge and St. Louis are not directly linked to each other. The *C. elegans* World Wide Web server (http://www.sanger.ac.uk; http://genome.wustl.edu/gsc) is the most up to date source of sequence data, but only on the DNA level.

Currently, Wormpep is released in Fasta format, which contains one line of annotation for each protein, and the sequence. The annotation line contains two compulsory fields: entry name (cosmid name.number) and Wormpep accession number, and three optional fields: locus, functional annotation and a reference to the corresponding Swissprot entry. The functional annotation is extracted from the "Brief_identification" field in ACEDB, which is the functional annotation derived from the homologues. This field is exported to EMBL/Genbank with a "similar to" prefix. In the annotation process, no distinction is made between functional inference from orthologues, i.e. when the precise function is known with high confidence, and inference from paralogues, when only general properties can be predicted, such as "transporter", or "dehydrogenase". 2868 (39%) of the 7299 proteins in release 11 are functionally annotated.

The growth of Wormpep since 1993 is plotted in figure 9.1. It has grown exponentially due to the increase in sequencing throughput, but will tail off as the sequencing project will move into less gene rich areas towards the end. These areas are harder to sequence due to repetitive DNA elements and scarcity of cosmid clones, and the overall throughput rate is also scheduled to slow down. The distribution of protein lengths in Wormpep is very skewed, as seen if figure 9.2. The mean length is 450, while the median is only 342. This is due to a small number of very long proteins. 19 predicted proteins have more than 3000 amino acid residues. The largest protein so far is K07E12.1, with 13055 residues. It contains some 10 fibronectin type 3 domains, 6 immunoglobulin superfamily domains (cell-adhesion molecule-like), 1 epidermal growth factor-like domain, 3 von Willebrand factor

type A domains, and some 60 repeats of a new type. Such multiple domain giants are nearly always extracellular proteins that often have a role in cell-cell binding.

The accuracy of the gene predictions in Wormpep depends on the amount of evidence available. Genes for which ESTs have been sequenced can be considered experimentally verified in the regions that match, and genes with strong similarity to other proteins are usually close to 100% correct. For genes that lack these extrinsic pieces of evidence, one must rely on the intrinsic properties in the DNA sequence, such as coding potential and splicing signals, for the entire prediction. The program used, Genefinder [P. Green, unpublished], generally predicts most of the exons in the middle of genes correctly. Exons at the start and end often contain weaker signals, however, and are frequently mispredicted. Occasionally, close neighbouring genes may be fused, and single genes with long introns may be fragmented. About one third of the genes have at least one EST match, and over half are similar to proteins from *C. elegans* or other organisms, so less than half of the predictions in Wormpep relied solely on intrinsic properties.

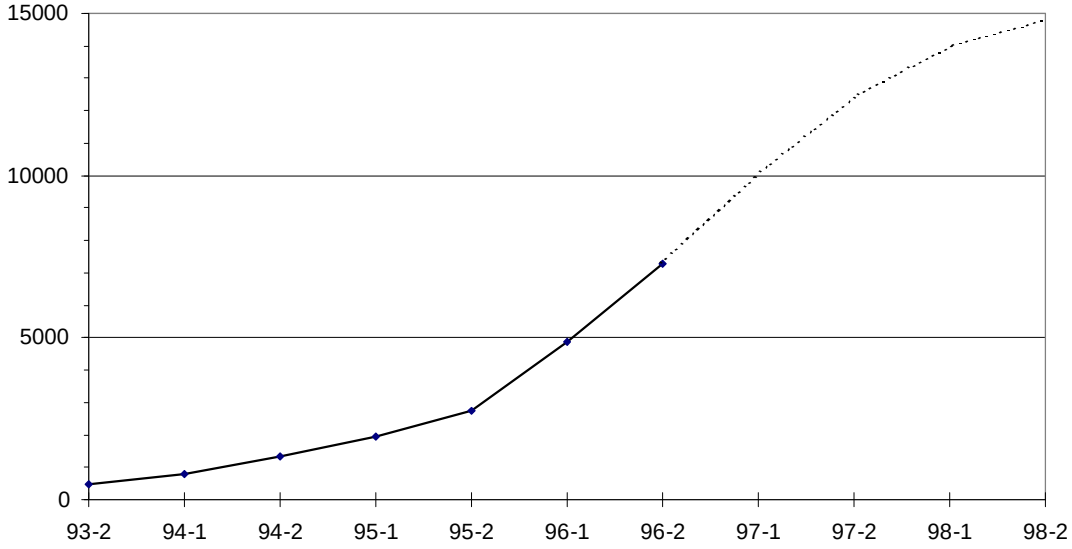


Figure 9.1. Previous and projected growth of Wormpep.

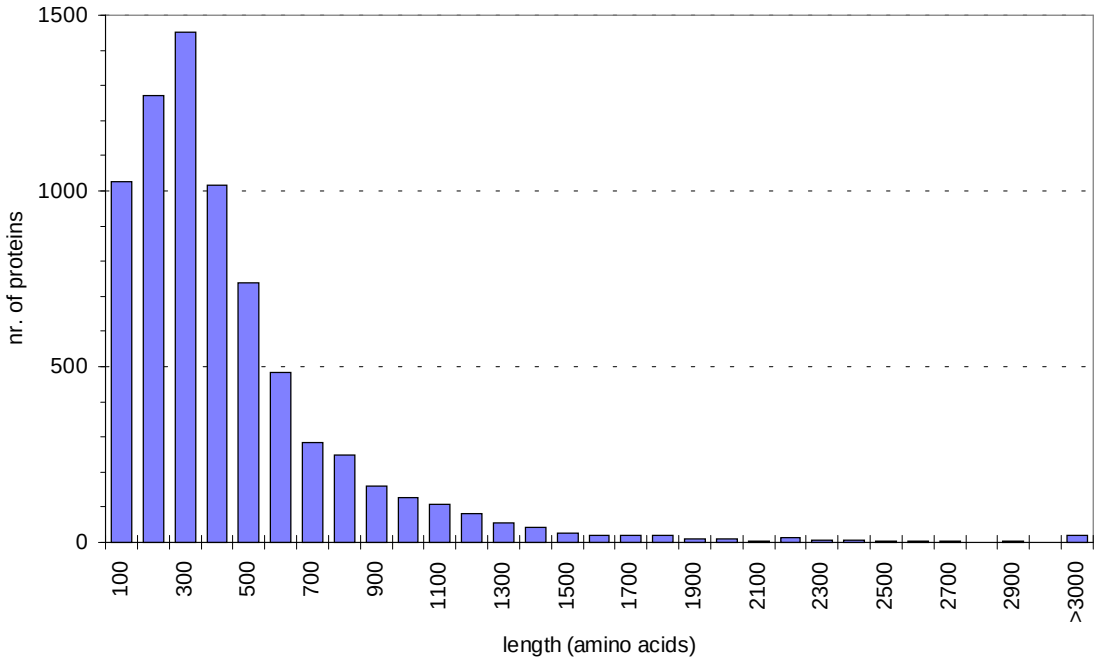


Figure 9.2. Length distribution of Wormpep entries.

9.4 Classification of Wormpep entries by Pfam

All proteins in Wormpep have been annotated manually, using the analysis workbench described in part 1. This annotation is not always easy to use for summary purposes, because the nomenclature used is variable, and it is not always complete. For example, over 20% of the eukaryotic protein kinases found by Pfam did not have the word 'kinase' in the annotation. About half of these lacked annotation completely, while the other half had other annotations, such as 'receptor' or 'cell division control protein'. Guanylate cyclases also match the protein kinase family.

A more systematic approach, which is convenient for summarising the families, is using the Pfam database (chapter 7). We compared all Wormpep 11 sequences to all Pfam families, using as significance cutoffs Pfam's previously recorded family-specific cutoffs that proved to exclude negatives. All protein domains with more than 5 examples are listed in table 9.1. Many of the most frequent domains are multiply repeated in single proteins. For example, 38 laminin type EGF domains are spread in only 5 proteins, and the 184 ank repeats in only 40 proteins. A few common *C. elegans* families are not listed in table 9.1 because the were not part of Pfam 1.0. These include DEAD/DEAH box helicases, annexin domains and collagens.

Table 9.1. The most frequent Pfam domains (n > 5) occurring in Wormpep 11, comprising about half of the proteins in *C. elegans*. The number of domains is somewhat overestimated for some families due to multiple fragment matches, and because multiple alternative splicing products were included the number of proteins may be slightly too high.

Nr. of domains	Nr. of proteins	Pfam accession	Pfam annotation
216	202	PF00069	Protein kinase
184	40	PF00023	Ank repeat
160	67	PF00096	Zinc finger, C2H2 type
158	37	PF00008	EGF-like domain
120	21	PF00041	Fibronectin type III domain
115	26	PF00047	IG superfamily
81	21	PF00090	Thrombospondin type 1 domain
74	48	PF00076	RNA recognition motif. (aka RRM, RBD, or RNP domain)
71	64	PF00001	7 transmembrane receptor (Rhodopsin family)

69	14	PF00057	Low-density lipoprotein receptor domain class A
59	41	PF00065	Neurotransmitter-gated ion-channel
57	55	PF00105	Zinc finger, C4 type (two domains)
57	15	PF00014	Kunitz/Bovine pancreatic trypsin inhibitor domain
46	45	PF00046	Homeobox domain
46	29	PF00005	ABC transporters
43	5	PF00028	Cadherin
41	31	PF00102	Protein-tyrosine phosphatase
38	5	PF00053	Laminin EGF-like (Domains III and V)
37	36	PF00099	Zinc-binding metalloprotease domain
33	22	PF00149	Ser/Thr protein phosphatases
33	16	PF00036	EF hand
32	10	PF00013	KH domain family of RNA binding proteins
31	24	PF00059	Lectin C-type domain short and long forms
31	24	PF00018	Src Homology domain 3
30	30	PF00106	Alcohol/other dehydrogenases, short chain type
28	28	PF00097	Zinc finger, C3HC4 type
27	27	PF00125	Core histones H2A, H2B, H3 and H4
26	26	PF00104	Ligand-binding domain of nuclear hormone receptors
24	24	PF00067	Cytochrome P450
24	24	PF00010	Helix-loop-helix DNA-binding domain
24	16	PF00168	C2 domain
22	21	PF00153	Mitochondrial carrier proteins
21	18	PF00017	Src Homology domain 2
20	20	PF00071	Ras family (contains ATP/GTP binding P-loop)
19	16	PF00092	von Willebrand factor type A domain
18	18	PF00083	Sugar (and other) transporters
18	18	PF00043	Glutathione S-transferases.
18	12	PF00135	Carboxylesterases
17	16	PF00078	Reverse transcriptase (RNA-dependent DNA polymerase)
16	7	PF00054	Laminin G domain
15	4	PF00084	Sushi domain
15	13	PF00004	ATPases Associated with various cellular Activities (AAA)
15	10	PF00085	Thioredoxins
14	12	PF00130	Phorbol esters / diacylglycerol binding domain
13	9	PF00038	Intermediate filament proteins
13	13	PF00169	PH (pleckstrin homology) domain
11	8	PF00060	Ligand-gated ionic channels
9	9	PF00063	Myosin head (motor domain)
9	7	PF00012	Heat shock hsp70 proteins
9	1	PF00050	Kazal-type serine protease inhibitor domain
8	8	PF00170	Basic region plus leucine zipper transcription factors
8	8	PF00011	Heat shock hsp20 proteins
8	6	PF00091	Tubulin
8	4	PF00122	E1-E2 ATPases
8	2	PF00058	Low-density lipoprotein receptor domain class B
7	7	PF00025	Arf family (contains ATP/GTP binding P-loop)
7	6	PF00112	Cysteine proteases
6	6	PF00160	Peptidyl-prolyl cis-trans isomerases

6	6	PF00155	Aminotransferases class-I
6	6	PF00171	Aldehyde dehydrogenases
6	3	PF00137	ATP synthase subunit C
6	2	PF00066	Notch

To look specifically for novel Pfam classifications, the 4431 proteins in Wormpep 11 for which no informative similarity has been found using the standard Blast/MSPcrunch approach [Sonnhammer & Durbin, 1994] were searched for Pfam matches. As significance cutoffs, the previously recorded cutoffs that exclude negatives for each Pfam family were used. Table 9.2 lists the 416 matches to 238 previously unannotated *C. elegans* sequences. A number of these matches had very high scores, indicating that they would probably have been found by Blast too, but had been missed due to human error. We have found empirically that most matches found by Pfam but not by Blast have scores below approximately 35 bits. Roughly half of the matches scored lower than this, thus representing genuinely novel classifications that are likely to have been missed because of the similarity to any one other protein was too weak. The matches above this score are more likely to have been missed due to mistakes, such as adding the annotation to the wrong field in ACEDB, or not inspecting Blastp matches, which are more sensitive than Blastx since the alignments are not disrupted by introns.

Table 9.2. Novel Wormpep classifications found by Pfam-A.

Pfam family accession number and description	**C. elegans protein (score)**
PF00001:7 transmembrane receptor (Rhodopsin family)	B0244.7(27.9) B0563.6(24.8) C01F1.4(68.0) C02H7.2(64.3) C26F1.6(89.4) C30B5.5(92.6) D1014.2(24.2) F10D7.1(24.5) F36D4.4(52.6) R11F4.2(63.7) T07F8.2(24.4) T14C1.1(55.7) T19F4.1(30.0) ZK418.6(62.7) ZK418.7(27.9) C54A12.2(33.1) F21C10.9(80.4) F47D12.1(93.9) F55E10.7(52.1) ZK1307.7(85.6)
PF00002:7 transmembrane receptor (Secretin family)	B0286.2(26.9)
PF00004:ATPases Associated with various cellular Activities (AAA)	F54B3.3(75.5)
PF00005:ABC transporters	C56E6.1(90.6) F43E2.4(45.9) C05D10.3(226.7)
PF00137:ATP synthase subunit C	R10E11.8(146.6)
PF00168:C2 domain	2xF07A5.5(67.3-82.9) K07G5.3(30.7) T12A2.4(20.5) 3xT12A7.1(22.6-86.1)

PF00135:Carboxylesterases	2xC01B10.4(26.4-36.0) 2xC01B10.10(40.4-53.0)
PF00130:Phorbol esters / diacylglycerol binding domain	2xF13B9.5(46.3-86.4) F42A9.1A(29.0) F42A9.1B(53.8)
PF00122:E1-E2 ATPases	W09C2.3(53.8)
PF00008:EGF-like domain	2xC37C3.7(76.3-343.7) C54D1.5(17.9) F09E8.2(18.4) F35D2.3(17.0) F58B3.8(17.6) F58G4.4(20.8) K06A9.3(25.5) K07D8.2(26.1) R05G6.9(22.3) 5xR13F6.4(24.2-30.6) 5xZK783.1(18.2-27.1) 13xF28E10.2(17.4-30.4) F40F11.4(25.5) F55G1.13(17.3)
PF00010:Helix-loop-helix DNA-binding domain	7xC17C3.7(17.8-30.1) C17C3.8(26.4) C43H6.8(25.5) F31A3.2(60.0) F31A3.4(31.8) F48D6.3(31.8) T01E8.2(66.9) C17C3.10(43.5) C28C12.8(26.4) C44C10.8(62.7) F46G10.6(54.3)
PF00011:Heat shock hsp20 proteins	F43D9.4(76.1)
PF00012:Heat shock hsp70 proteins	F43E2.8(88.4) T24H7.2(1276.0)
PF00013:KH domain family of RNA binding proteins	C56G2.1(214.1)
PF00014:Kunitz/Bovine pancreatic trypsin inhibitor domain	B0222.5(44.0) C37C3.5(38.1) 3xC37C3.6(80.4-92.5) 8xT22F7.3(53.9-103.4)
PF00169:PH (pleckstrin homology) domain	5xF38B7.3(41.8-100.3) F41F3.2(34.9) K10B2.5(33.9) F52D10.6(44.1) ZK1248.10(40.8)
PF00017:Src Homology domain 2	T27F7.2(34.8) T06C10.3(28.1)
PF00018:Src Homology domain 3	B0336.6(34.5) F32A5.6(92.0) F35A5.8(67.1) F49E2.3(40.2) K11E4.4(35.4) F09E10.8(60.8)
PF00020:TNFR/NGFR cysteine-rich region	T02C5.1(51.6)
PF00102:Protein-tyrosine phosphatase	C07E3.4(35.7)
PF00022:Actins	F42C5.9(77.3)
PF00023:Ank repeat	M60.7(86.8) 4xC01H6.2(28.4-40.7) 2xC18H2.1(38.4-39.2) C18H2.3(49.5) 2xC43H6.3(39.5-40.4) 2xK04C2.4(36.1-37.0) 2xC18F10.7(33.1-40.5)
PF00028:Cadherin	B0034.3(41.8)
PF00134:Cyclins	2xR02F2.1(27.7-41.8) T12C9.4(29.6)
PF00037:4Fe-4S ferredoxins and related iron-sulfur cluster binding domains.	C25F6.3(56.9)
PF00147:Fibrinogen beta and gamma chains, C-terminal globular domain	D1009.3(23.7)
PF00041:Fibronectin type III domain	C36B1.2(70.4) 2xK09E2.4(33.5-73.7) 2xR07E4.2(28.6-50.7) T22E5.3(36.7) ZC374.2(45.2) 3xZK617.1B(34.3-40.3)
PF00043:Glutathione S-transferases.	31xC25H3.7(39.4-97.1)
PF00125:Core histones H2A, H2B, H3 and H4	F17E9.12(25.4) F17E9.13(69.5) W05B10.1(206.9)
PF00046:Homeobox domain	K03A11.3(120.6) W05E10.3(109.3)
PF00104:Ligand-binding domain of nuclear hormone receptors	C25B8.6(33.8) F16H9.2(43.3) F25E5.6(32.3) T07C5.2(38.5) T07C5.3(50.3) T07C5.5(25.8) ZK418.1(40.5)
PF00047:IG superfamily	C18F3.3(45.0) C37C3.5(23.9) C53B7.1(27.6) 2xF48C5.1(15.0-17.3) K09E2.4(16.0) 3xT02C5.3(15.9-30.2) C18A11.7(22.8) F21C10.7(18.1) 3xK02E10.8(15.9-22.2) 4xZK617.1B(17.8-25.4)

PF00052:Laminin B (Domain IV)	15xC54D1.5(15.7-34.8)
PF00053:Laminin EGF-like (Domains III and V)	C54D1.5(104.6)
PF00054:Laminin G domain	10xF41G3.12(39.5-80.9)
PF00055:Laminin N-terminal (Domain VI)	C54D1.5(76.2)
PF00057:Low-density lipoprotein receptor domain class A	F44C4.1(335.9)
PF00059:Lectin C-type domain short and long forms	B0218.6(30.6) 2xB0218.8(81.5-94.0) 2xB0286.2(69.6-100.1) C25G4.1(81.7) 2xF09G8.8(55.0-56.1) F52E1.2(35.1) K02F3.5(84.5) M02F4.7(50.6) 2xT05A7.2(44.9-53.9) T19E7.1(55.2) ZK666.7(113.5) ZK1193.2(30.5)
PF00061:lipocalins	ZK742.5(84.8)
PF00153:Mitochondrial carrier proteins	K01C8.7(29.4) K02F3.2(189.3)
PF00065:Neurotransmitter-gated ion-channel	C35C5.5(242.0) 2xF17E9.7(138.3-236.7) F17E9.8(100.7)
PF00067:Cytochrome P450	K09A11.3(93.2)
PF00069:Protein kinase	B0496.3(417.6) C25H3.1(320.6) D2024.1(87.8) EEED8.9(65.0) F22D6.5(73.0) F35C8.1(176.4) F35C8.2(119.4) F35C8.3(74.1) F54H5.2(192.7) K10D3.5(90.0) R13F6.7(34.9) R13H9.5(240.5) R13H9.6(70.4) W03A5.1(79.9) C36B1.10(186.9) F59E12.2(150.6) T06C10.3(204.5) W07A12.4(196.4) ZK617.1B(32.1)
PF00160:Peptidyl-prolyl cis-trans isomerases	D1009.2(274.2)
PF00071:Ras family	C35C5.4(312.9) F43D9.2(265.7)
PF00075:RNase H	ZK1290.6(200.3)
PF00076:RNA recognition motif. (aka RRM, RBD, or RNP domain)	M18.7(53.5) C01F6.5(26.5) EEED8.1(26.0) K08F4.2(27.1) T04A8.6(27.9) T11G6.8(68.3) W04D2.6(66.4) C26E6.9A(56.8) F07A11.6(30.9) F18H3.3B(27.5)
PF00078:Reverse transcriptase (RNA-dependent DNA polymerase)	4xB0478.2(82.7-103.7) F56C9.2(49.1) T07E3.1(87.5) F28E10.3(107.9)
PF00083:Sugar (and other) transporters	F14B8.3(108.8) K05F1.6(61.8) T22F7.1(93.6)
PF00084:Sushi domain	T07H6.5(46.7)
PF00085:Thioredoxins	7xC06A6.5(29.0-68.8) F47B7.2(27.3) C35D10.10(50.5)
PF00086:Thyroglobulin type-1 repeat	B0222.5(23.3)
PF00088:Trefoil (P-type) domain	D2096.3(58.8)
PF00090:Thrombospondin type 1 domain	C11H1.1(50.5) C37C3.6(27.4) 5xD1022.2(17.6-35.5) F11C7.2(20.0) 2xF53B7.5(18.9-39.3) F58F9.6(44.8) T19D2.1(23.0) 4xF01F1.13(18.3-55.4) 2xF14H12.3(30.5-41.1) 3xF23H12.5(35.2-64.0) F57C12.1(28.1)
PF00091:Tubulin	C54C6.2(27.2)
PF00092:von Willebrand factor type A domain	C16E9.1(1005.8) 2xF09G8.8(92.6-102.7) ZK666.3(152.8) ZK666.6(31.2) ZK666.7(38.8) ZK673.9(33.9) R10H10.3(32.8) T19D12.4(43.7) T25C12.3(74.4) ZK1193.2(39.7)
PF00096:Zinc finger, C2H2 type	B0035.1(45.9) C09F5.3(26.0) 2xC28H8.9(23.7-25.6) D1046.2(20.9) F21D5.9(20.6) F26F4.8(28.1) 2xF52E4.7(24.2-31.1) F53B3.1(21.7) 4xK04C1.3(22.3-32.9) T20H4.2(26.6) T21C9.2(26.6) W04D2.4(21.3) ZC395.9(21.3) 2xF15C11.1(23.1-31.4)
PF00097:Zinc finger, C3HC4 type	7xC01B7.6(23.6-39.1) C11H1.3(21.1) C26B9.6(34.4) EEED8.9(27.8) F26F4.7(30.4) F47G9.4(27.5) C32D5.10(30.4) C32D5.11(42.8)

	F19G12.1(42.8) F54B11.5(31.6) T13A10.2(35.6)
PF00105:Zinc finger, C4 type (two domains)	C26B2.4(27.1) F16H9.2(39.5) F54D1.4(41.9) T07C5.2(67.7) T07C5.5(34.8) ZK418.1(67.3) F36A4.14(53.3) F21D12.1B(68.4)
PF00098:Zinc finger, CCHC type	C27B7.5(32.7)
PF00099:Zinc-binding met-alloprotease domain	F53A9.2(24.2) F57B7.4(21.2) F58A6.4(30.2) T19D2.1(23.5) F57C12.1(32.2) K11G12.1(28.6) K11G12.1(22.8)

About 20% of all Wormpep proteins have at least one domain that matches a Pfam-A family. The matching regions are on average about half the length of the proteins, so about 10% of the residues in Wormpep are covered by Pfam-A. Since Pfam-A only contains the most common protein domain families, these numbers are necessarily much lower than the fraction of Wormpep that has matches found by all-protein searches using BLAST. Figure 9.3 illustrates the relative proportions of annotation in Wormpep. Overall, about 40% of the proteins have functional annotation based on BLAST/MSPcrunch analysis. Some 3% of the Pfam matches are to previously unannotated proteins. Some of these were not annotated due to human error, and not because BLAST failed to pick up any similarity. Although Pfam-A currently adds only a few more percent to the fraction annotated proteins, the analysis of proteins with BLAST matches benefits from Pfam matches too, by clearer indication of domains and family annotation. Furthermore, cases where Pfam detected previously unidentified domains in previously annotated proteins are not reflected in figure 9.3. Considering that this analysis is based on the first release of Pfam, already a substantial fraction of Wormpep is covered. The two approaches thus complement each other well, and the best analysis is achieved by combining them.

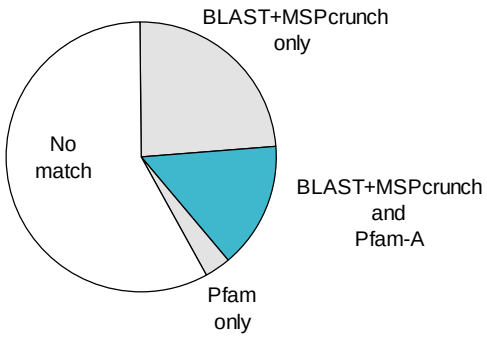


Figure 9.3. Fractions of *C. elegans* proteins that can be annotated based on homology, using the pairwise BLAST+MSPcrunch approach (chapters 3-6) and the family-based Pfam approach (chapters 7-8). Together they add up to an annotation level of about 42%.

9.5 Clustering of Wormpep proteins

Many Wormpep proteins do not match a Pfam family. To examine the complete distribution of paralogue families in Wormpep, a clustering analysis was performed. The nature of protein sequences renders many standard clustering techniques inappropriate.

To illustrate this, let us consider the simplest clustering method, 'single linkage' [Watanabe and Otsuka, 1995]. The principle works as follows: All proteins are compared to each other and all significant pairwise matches are stored. The proteins are then linked together in clumps by joining all proteins that have at least one match to one of the proteins in the group.

This procedure would work perfectly if all proteins only had a single domain, and if a clear significance cutoff to separate related from unrelated proteins existed. Unfortunately, neither of these is true: unrelated clusters may be joined by multi-domain proteins, or by false links. As a consequence, it may be difficult to generate multiple alignments from the resulting clusters, especially if the proteins only share a domain.

An algorithm which reduces the space requirements of the clustering, and to some extent the joining of unrelated clusters is the Hunter, Harris and States 'minimal spanning tree' method [States *et al.,* 1993], which add sequences incrementally and only stores the highest scoring link for each new sequence. However, this method has the opposite effect, that related families are sometimes not merged with each other.

The first drawback, false linkage due to multiple domains, has been tackled by the CLUSDOM program [Koonin *et al.,* 1996b], which only clusters on links that overlap in sequence. However, CLUSDOM does not indicate which part of multi-domain proteins belong to which cluster, so the alignment problem remains unsolved. Also, the program was unavailable for this study.

We applied single linkage clustering to Wormpep 11. Using Blastp filtered by MSPcrunch (chapter 4) with a cutoff that is normally considered stringent (twilight zone between scores 40 and 80) resulted in a super-cluster of about a third of all sequences. MSPcrunch effectively removes biased composition matches, but some spurious links will be accepted with these parameters. By raising the stringency to not include any matches scoring below 90 eliminates all spurious matches, but extensive joining of unrelated clusters occurred due to multi-domain proteins. The largest cluster contained 585 proteins, including such diverse protein families as protein kinases, phosphatases, proteases, protease inhibitors, transcription factors, extracellular domains, etc.

To make the clustering useful for further studies, ideally the clustering algorithm should not only avoid these drawbacks as much as possible, but also generate multiple alignments of the resulting clusters. This is important for generating consensus sequences or profiles for further characterisation of the clusters. To produce useful multiple alignments, a strict definition of homology domains is needed.

A method that explicitly attempts to find domain boundaries and that produces multiple alignments of the found domains is the Domainer algorithm [Sonnhammer and Kahn, 1994]. It also takes repeated domains within one protein into account. Domainer first lets multi-domain proteins join unrelated clusters, and then analyses the resulting graph of clusters for likely domain boundaries, at which the super-cluster is cleaved. The main drawback of Domainer is that is vulnerable to imperfections in its input of pairwise similarity data. Incomplete matching regions can cause Domainer to infer too many domain boundaries, resulting in fragmentation of real domains. On a dataset such as Wormpep, which contains predicted genes with unverified N and C-termini, the fragmentation will be compounded. However, the core domains are usually of reasonable quality to use as a starting point, and the risk of merging unrelated families is small. Because of the fragmentation, the Domainer output overestimates the number of domain families. These ‘pseudo-domains’ often need to be processed manually to produce true domain families.

Domainwise clustering of entire Wormpep

About 60 % of all *C. elegans* proteins match another *C. elegans* protein. There is thus an abundance of paralogue clusters. When the Domainer program was applied to Wormpep 11, this generated 1818 clusters in the range of 2-89 members. This is probably more than the true number of domain families because of over-fragmentation in Domainer however. The distribution of the cluster sizes is plotted in figure 9.4. To analyse what proportion of these are specific for *C. elegans* and other species in the family rhabditida, the consensus sequence of each cluster was searched against Swissprot 33 with Blastp and was filtered by MSPcrunch. It appears that most of the large clusters are domains that are found in other phyla, while a large fraction of the smaller clusters appear to be specific for nematodes.

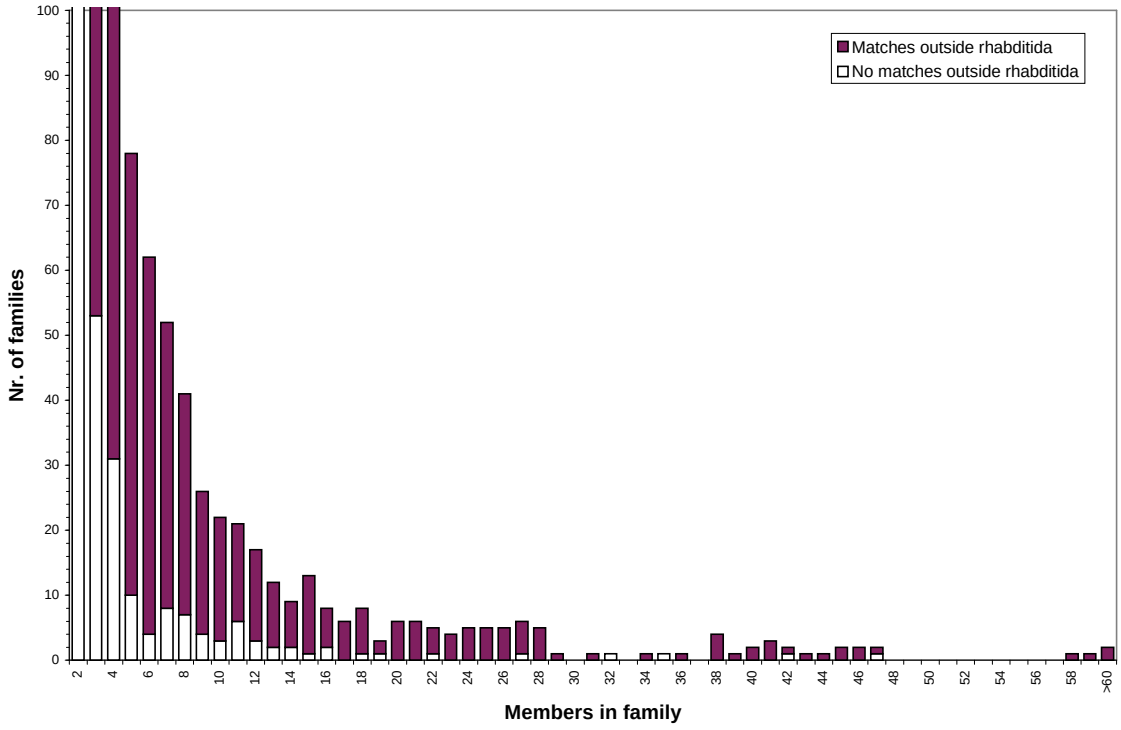


Figure 9.4. Histogram of Wormpep 11 family sizes based on Domainer clustering and BLAST analysis of the domain family consensus sequences. Families for which similarity to proteins outside rhabditida was detected are filled columns; apparently nematode-specific families are empty columns.

Domainwise clustering of Wormpep using Pfam

The main problem with the above cluster analysis is that it depends on Blastp and Domainer, which do not produce perfect data. For a rough estimate of how many proteins have paralogues and the size range of the largest clusters, Domainer analysis is adequate, but the number of domain clusters is likely to be overestimated.

In this case however, we can improve the Domainer clustering by using the previously found matches to Pfam-A families, as described above. By removing these matching segments, most of the large families, which are most prone to errors, are avoided, and correct domain boundaries are introduced. The procedure is to extract all sequence sections larger than 30 residues that were not covered in Pfam-A into separate entries. A protein with a Pfam-A domain in the centre that has long flanking regions on either side, will thus generate two entries. By doing this, Domainer will consider each section as an independent sequence, and the boundary to the Pfam-A segment will be used as a real domain boundary. Furthermore, members of previously found nematode-specific families were removed.

After extracting these segments from Wormpep, the remaining 8221 segments (90% of Wormpep 11) were clustered by running Blastp and Domainer, in the same way as described in the previous section. As shown in figure 9.5, the nematode-specific families become larger this way, and most of the large families that match outside rhabditida are no longer present.

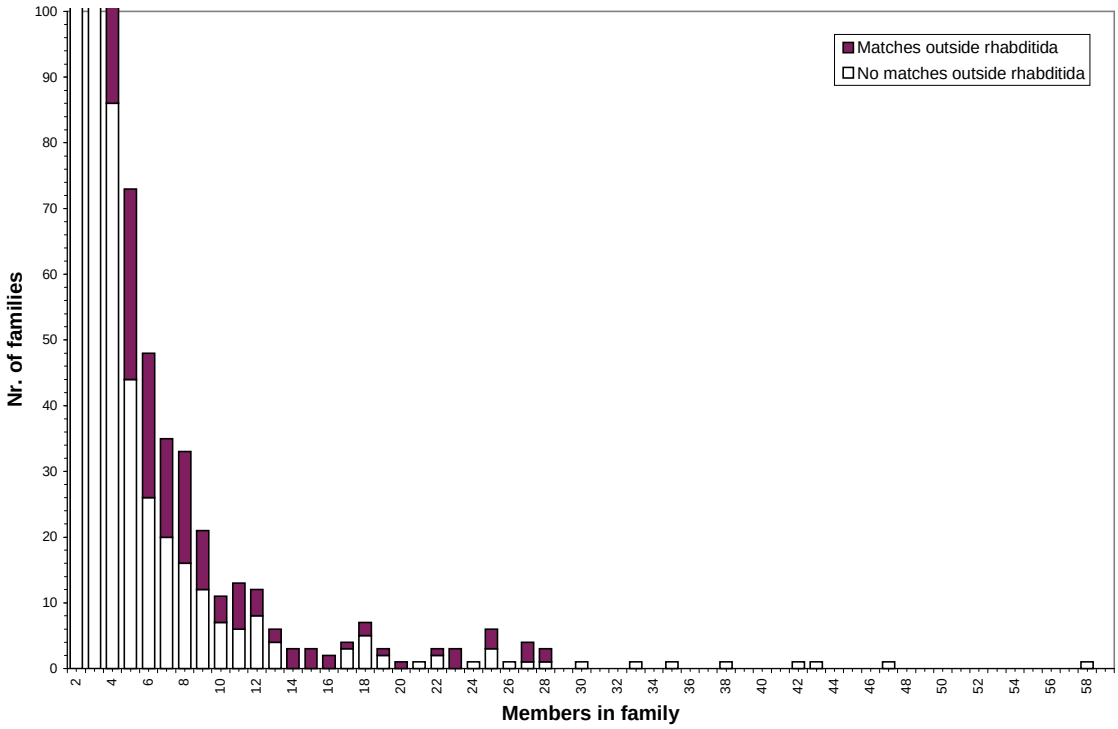


Figure 9.5. Histogram of Wormpep 11 family sizes based on Domainer clustering after removal of known protein families and BLAST analysis of the domain family consensus sequences.

9.6 Nematode-specific protein families

The largest *C. elegans* protein clusters that were found to be unique to rhabditida in the previous section were analysed in further detail. To improve the quality of the alignments they were rebuilt from complete sequences. These alignments were searched against Swissprot and Swissprot-TREMBL using sensitive HMM (hidden Markov model) methods as a second pass to look for matches to other organisms. Only families lacking clear homology outside rhabditida were considered. Hydrophobicity patterns, coiled-coil predictions and Prosite pattern matches were inspected too. This way we have collected 10 nematode-specific families, which are listed in table 9.3.

Three of the families are probably G-protein coupled receptors. Although the sequence similarity is weak, it is supported by alternating hydrophobic/hydrophilic regions typical for receptors, and there is also a characteristically conserved arginine at the end of the third pre-

dicted transmembrane helix. Some of the members have been found to be expressed in sensory neurons, and are likely to function as olfactory receptors [Troemel *et al.,* 1995]. The fact that so many G-protein coupled receptor families appear to be nematode-specific is not surprising, since divergence rates of transmembrane proteins is much higher than for globular proteins.

Three examples of these families are shown in figures 9.6-8. Family nr. 2 has weak similarity to transthyretin (formerly called prealbumin), which transports thyroid hormones. The hydropathy plot of family nr. 8 suggests it may have a transmembrane location, but there is no detectable sequence similarity to G-protein coupled receptors.

The members in family nr. 9 are not randomly distributed throughout the genome. As seen in figure 9.8b, large clusters are present on chromosomes V and X, while chromosome I and III only contain one member together. This indicates that these families arose by local gene duplication. There is also a strong correlation between the similarity and the distance between two members, which is illustrated in figure 9.8. Members on the same cosmid are nearly always most similar to each other. In only one case of the ten cosmids that have more than one member, are they not all very similar to each other (C42D4.4, which does not cluster with the three other members on C42D4).

Multiple alignments of these families are available by anonymous FTP at ftp.sanger.ac.uk in /pub/databases/wormpep/wormPfam.

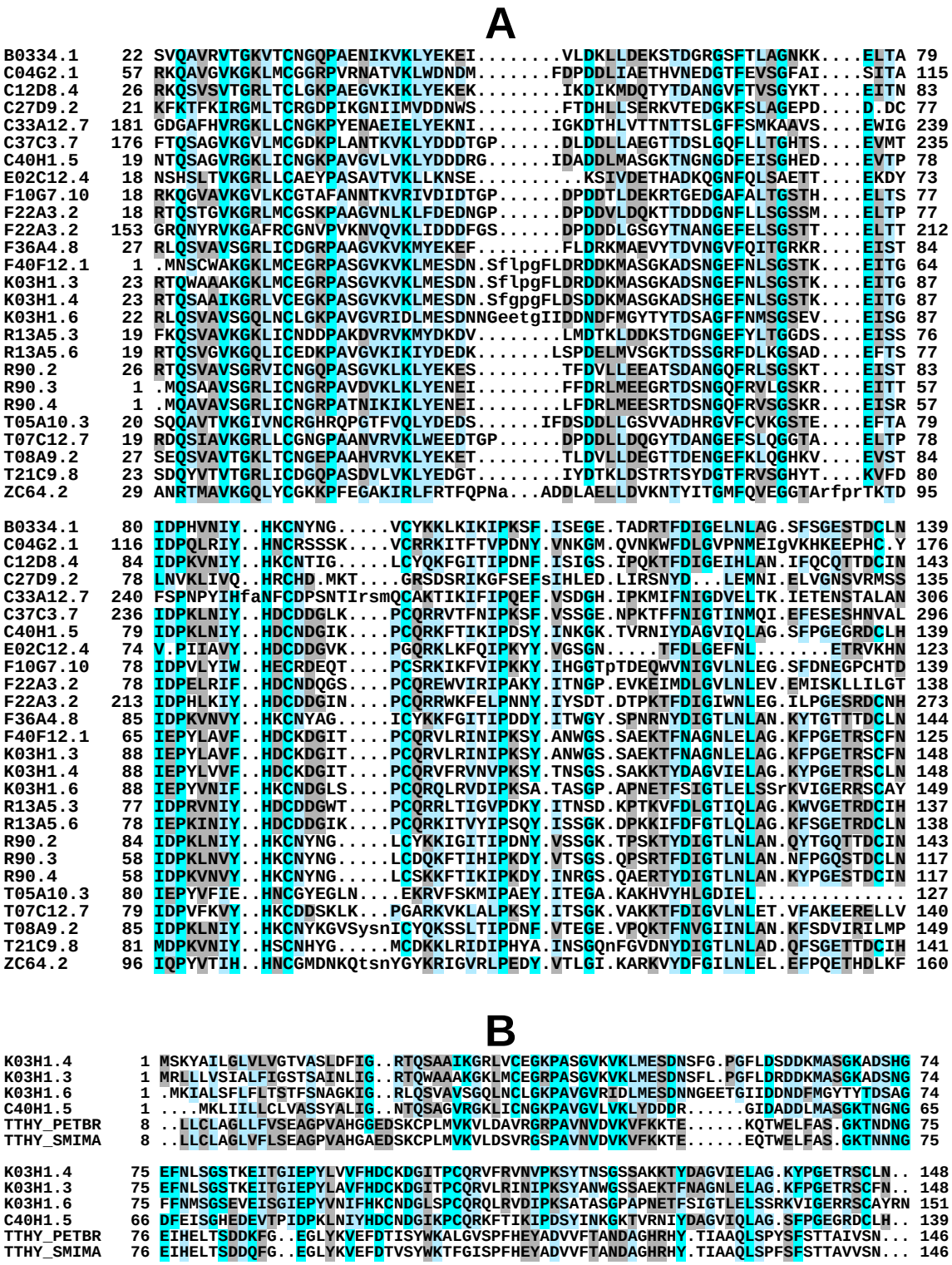


Figure 9.6 A. Alignment of a selection of the members in nematode-specific family nr 2. This family has weak similarity to transthyretins (B), suggesting a putative function as hormone transporter.

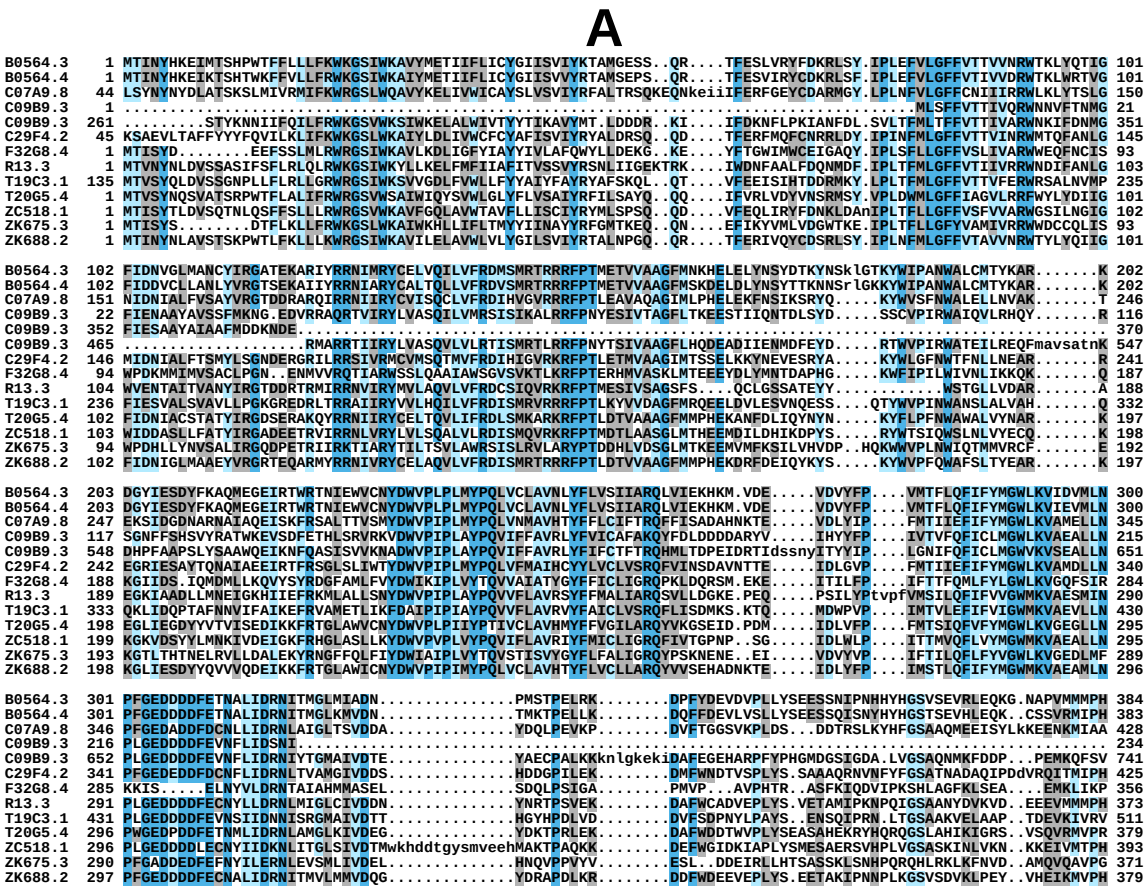


Figure 9.7. A. Alignment of all members of nematode-specific family nr 8. No putative function has been assigned to this family. The hydropathy plot (B) suggests that the proteins are likely to have a transmembrane location.

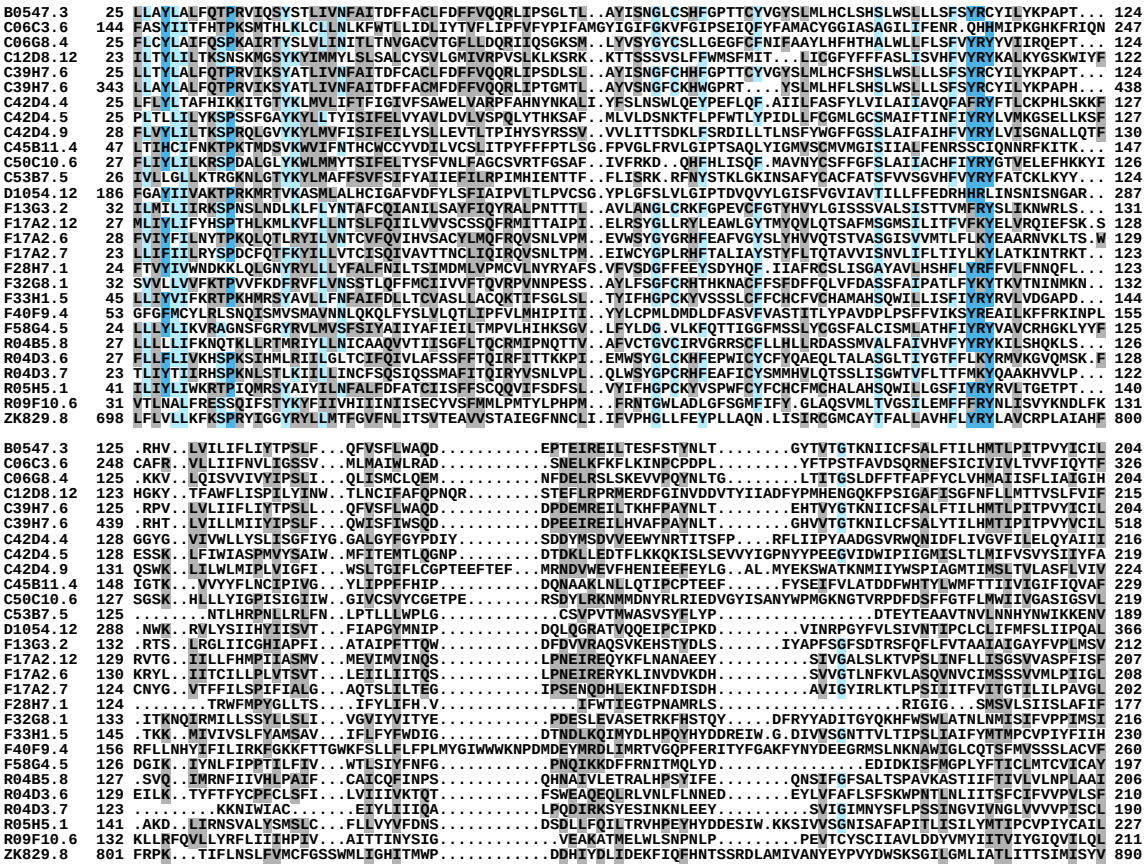


Figure 9.8a. Alignment of a selection of the members in nematode-specific family nr 9. This family has weak similarity to G-protein coupled receptors, which is supported by the hydropathy profile (not shown).

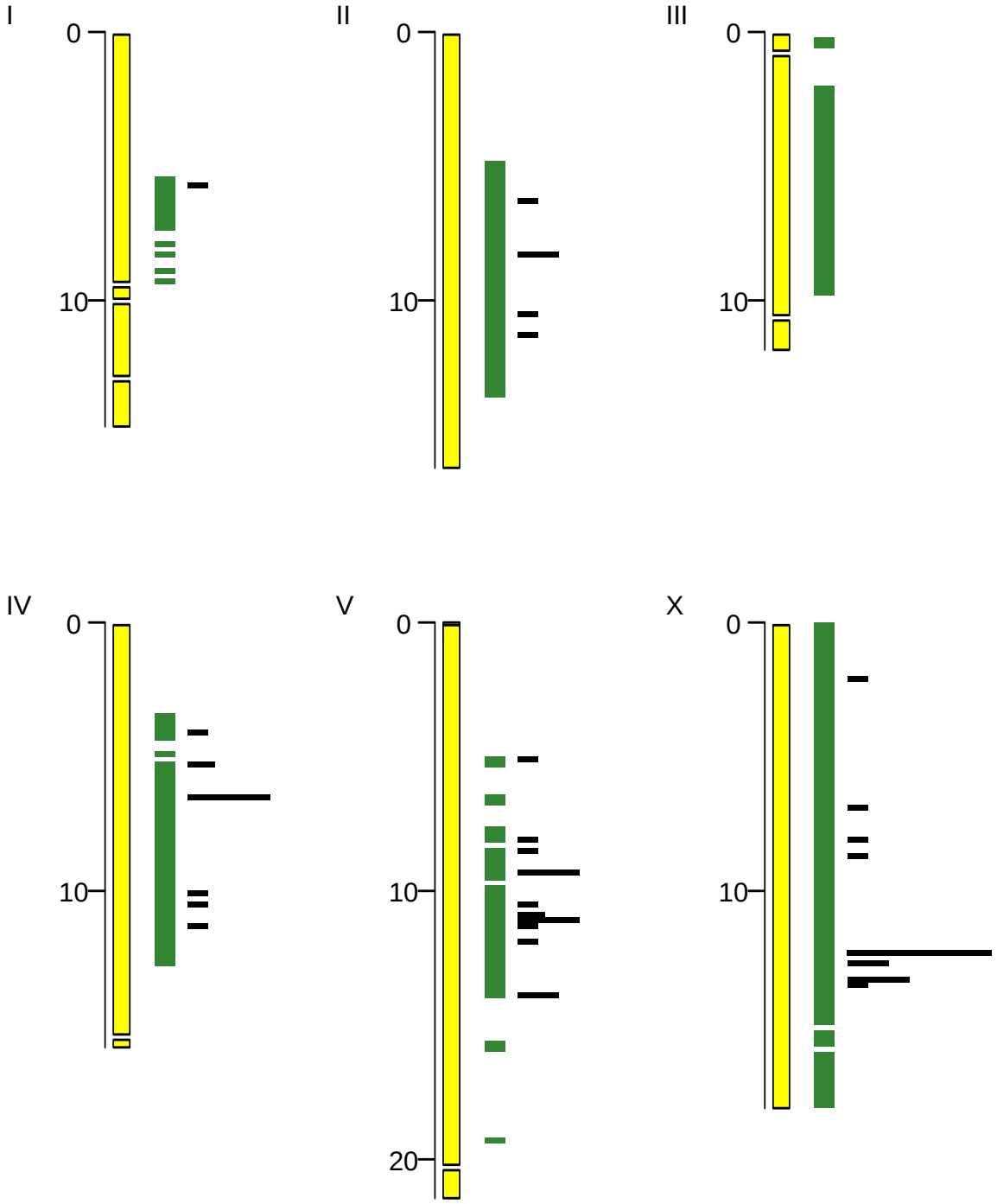


Figure 9.8b. The distribution of all members in nematode-specific family nr. 9 over the six chromosomes of *C. elegans* (black bars). The yellow areas represent the regions that have been cloned and the green areas are the sequenced regions that Wormpep 11 is based on.

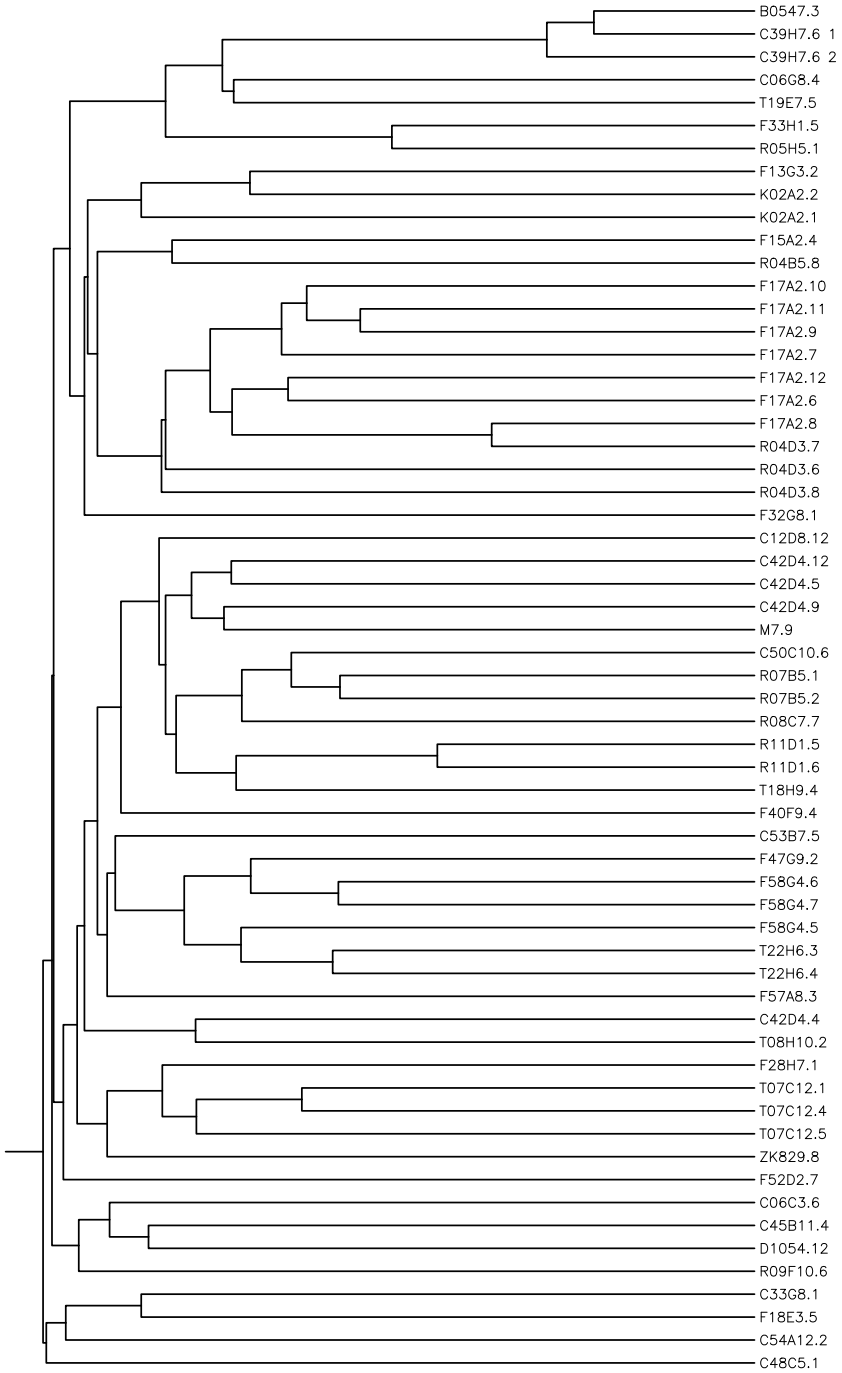


Figure 9.8c. Tree of all members in nematode-specific family nr 9. The most similar sequences are almost invariably close in the genome; either on the same cosmid (e.g. the seven proteins on F17A2), or on the neighbour cosmid (e.g. B0547 and C39H7, F33H1 and R05H5).

Table 9.3 **N**ematode-specific protein families. GPCR stands for G-protein coupled receptor. [1]These families have been found independently and are describe elsewhere [Troemel *et al.*, 1995].

Family nr	Members (nr. of domains in brackets)	Alternative name[1]	Domains	Proteins	Length	Putative function
1	C18H2.1 C18H2.3(2) C18H2.4 F37A4.4 F56D5.9		6	5	1000	- (Contains one ank repeat)
2	B0334.1 C04G2.1 C12D8.4 C14C10.2 C27D9.2 C33A12.7 C37C3.7 C40H1.5 E02C12.4(2) F10G7.10 F22A3.2(2) F26G1.3 F36A4.8 F40F12.1 K03H1.3 K03H1.4 K03H1.6 R13A5.3 R13A5.6 R90.2 R90.3 R90.4 T05A10.3 T07C12.7 T07C4.5 T08A9.2(2) T14G10.3 T14G10.4 T21C9.8 ZC64.2		33	30	160	Hormone transporter
3	B0244.4 B0244.5 B0244.6(3) B0244.7(3) ZK418.6 ZK418.7		9	6	195	- (trans-membrane)
4	C14A4.10 C18F10.4 C18F10.5 C18F10.6 C18F10.8 C33A12.10 C33A12.11 C33A12.8 C33D9.4 C34C6.1 F48D6.2 R07B5.6 R13F6.3 T01B7.2 T04A8.1 T04A8.2 T12A2.10 T12A2.11 T12A2.12 T12A2.13 T12A2.9 T13A10.13 T19C4.3 T21C9.7 T23F11.5	srg	25	25	400	GPCR
5	AH6.10 AH6.11 AH6.12 AH6.13(2) AH6.14 AH6.4 AH6.6 AH6.7 AH6.8 AH6.9 B0304.5(2) B0304.6 B0304.7(3) C27D6.10 C27D6.6 C27D6.7 C27D6.8 C27D6.9(2) C33G8.5 C56C10.5 F18C5.1(2) F18C5.6 F18C5.8 F23F12.10 F37C12.15 F37C12.16 F44F4.13 F44F4.5 F44F4.7 F49E12.5 F58A6.10 F58A6.11 F58A6.6 K11E4.4 R04B5.10 R05H5.6 R10H1.2 T11A5.3 T11A5.4(2) T19D12.8 T21H8.2 T21H8.3	sra	43	42	335	GPCR
6	B0228.3(13)		13	1	230	-
7	F26C11.3(9)		9	1	80	-
8	B0564.3 B0564.4 C07A9.8 C09B9.3(3) C29F4.2 F32G8.4 R13.3 T19C3.1 T20G5.4 ZC518.1 ZK675.3 ZK688.2		13	12	386	- (trans-membrane)
9	B0547.3 C06C3.6 C06G8.4 C12D8.12 C33G8.1 C39H7.6(2) C42D4.12 C42D4.4 C42D4.5 C42D4.9 C45B11.4 C48C5.1 C50C10.6 C53B7.5 C54A12.2 D1054.12 F13G3.2 F15A2.4 F17A2.10 F17A2.11 F17A2.12 F17A2.6 F17A2.7 F17A2.8 F17A2.9 F18E3.5 F28H7.1 F32G8.1 F33H1.5 F40F9.4 F47G9.2 F52D2.7 F57A8.3 F58G4.5 F58G4.6 F58G4.7 K02A2.1 K02A2.2 M7.9 R04B5.8 R04D3.6 R04D3.7 R04D3.8 R05H5.1 R07B5.1 R07B5.2 R08C7.7 R09F10.6 R11D1.5 R11D1.6 T07C12.1 T07C12.4 T07C12.5 T08H10.2 T18H9.4 T19E7.5 T22H6.3 T22H6.4 ZK829.8	srd	60	59	315	GPCR
10	K07E12.1(58)		58	1	195	-

9.7 Comparison of *C. elegans* to other genomes

One of the motivations to sequence the genome of the invertebrate *C. elegans*, is its potential usefulness as a model organism. Insights in nematode biology can often be extrapolated to human biology. For example, in a study of 44 human disease genes, up to 32 had a homologue in 25% of the *C. elegans* genome [Hodgkin *et al.*, 1995]. Naturally there are differences, but many of the basic life-supporting functions involved in e.g. energy metabolism, replication, gene expression and signalling are conserved throughout all phyla. Since *C. elegans* is a multi-cellular animal with a complete nervous system, it is hoped that many, if not most, human proteins will have a homologue in the worm. Many events during early development and differentiation, such as body patterning by the Hox cluster, are similar in the two organisms. To address the question of how much protein homology can be expected between human and *C. elegans*, all presently available proteins of these two organisms were compared.

It has been proposed that most protein domains that are present in two species belonging to different phyla, are also found in many other phyla. In 1993, it was estimated that over 90% of these 'anciently conserved domains' (ACRs) were already present as functionally characterised entries in the sequence databases [Green *et al.*, 1993]. To examine the amount of conservation between organisms from different kingdoms, we have also compared the *C. elegans* proteins to the proteins in two completely sequenced genomes: the yeast *S. cerevisiae* and the bacterium *H. influenzae*.

Pairwise comparison of proteins in H. sapiens, C. elegans, S. cerevisiae and H. influenzae

To explore the amount of conservation between these organisms, all proteins from each genome were compared to all proteins of the other genomes (see Materials and Methods). The results are listed in table 9.4 and are summarised in figure 9.9. The animals *H. sapiens* and *C. elegans* had the highest level of similarity, with 60% of the human proteins matching *C. elegans*. In general, the organism with the smaller genome has a larger proportion of its ge-

nome matching, and the organism with the larger genome has a larger number of proteins matching. In terms of percentages, *H. influenzae* is most similar to *S. cerevisiae*, which in turn is most similar to *C. elegans*, which in turn is most similar to *H. sapiens*. This is in agreement with the phylogenetic tree of these organisms.

Table 9.4. Cross-species protein comparison. The percentages within brackets in the second column indicate what fraction of the genome the set of proteins represent. *Only yeast TREMBL entries that were non-identical to Swissprot entires.

H. sapiens proteins in Swissprot 33	3475 (~5%)	
H. sapiens proteins that match Wormpep 11	2077	60%
H. sapiens proteins that match *S. cerevisiae*	1432	41%
H. sapiens proteins that match *H. influenzae*	323	9%
C. elegans proteins in Wormpep 11	7263 (~50%)	
C. elegans proteins that match *H. sapiens*	2378	33%
C. elegans proteins that match *S. cerevisiae*	2146	30%
C. elegans proteins that match *H. influenzae*	454	6%
S. cerevisiae proteins in Swissprot 33 and TREMBL*	6719 (~100%)	
S. cerevisiae proteins that match *H. sapiens*	1929	29%
S. cerevisiae proteins that match *C. elegans*	2447	36%
S. cerevisiae proteins that match *H. influenzae*	901	13%
H. influenzae proteins	1680 (100%)	
H. influenzae proteins that match *H. sapiens*	282	17%
H. influenzae proteins that match *C. elegans*	340	20%
H. influenzae proteins that match *S. cerevisiae*	482	29%

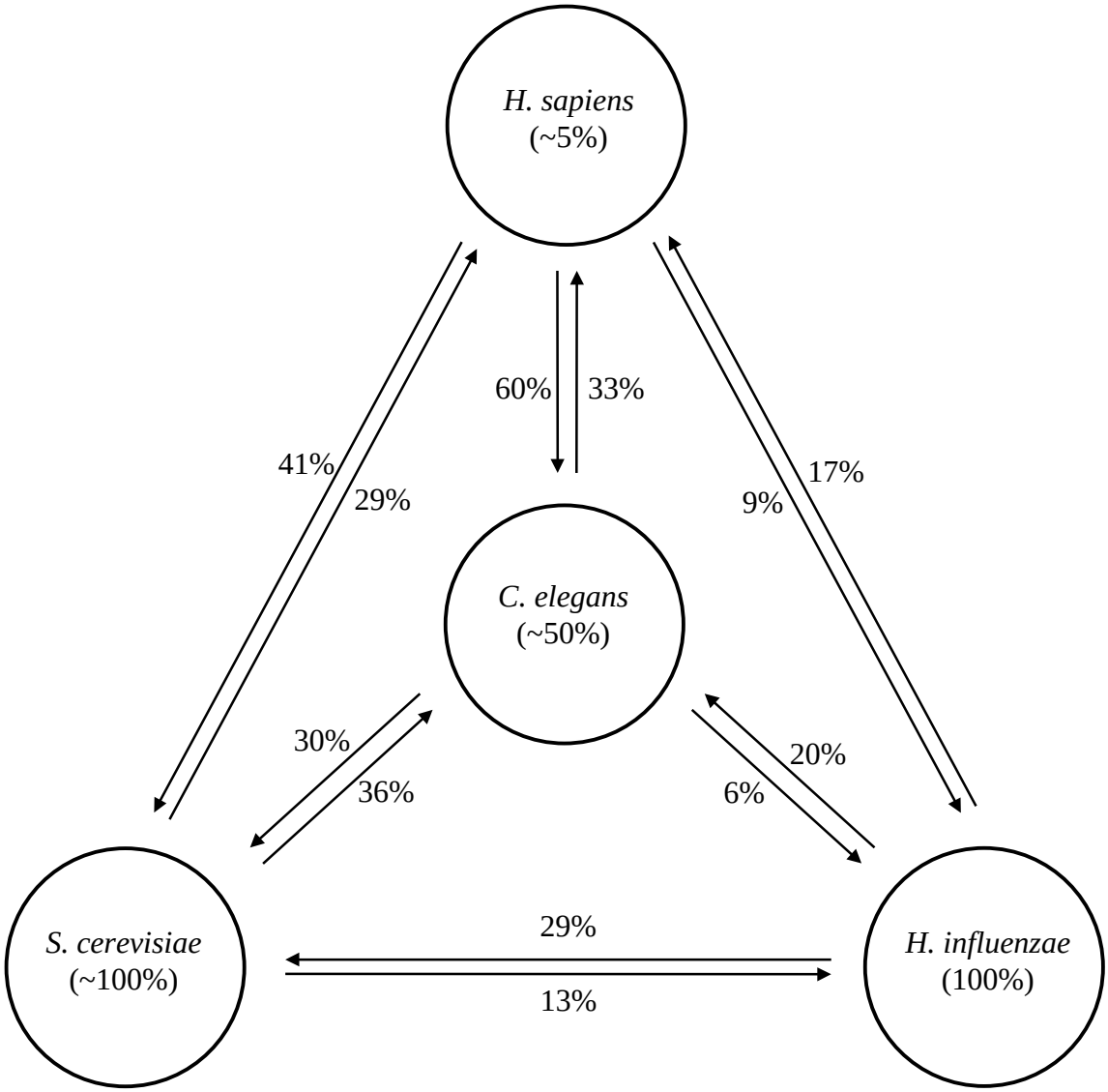


Figure 9.9. Percentages of the proteins in genomes representing bacteria, fungi and animalia that match one another. The percentage inside the circles indicates what fraction of the genome that was available for the analysis.

Common proteins in subsets of C. elegans, H. influenzae and S. cerevisiae

To further investigate to what extent protein families are shared among organisms from different kingdoms, we also looked for proteins that intersect these genomes. We excluded *H. sapiens* from this analysis, since only about 5% of the human proteins have been sequenced completely. The number of proteins common to two or more genomes can be counted from the point of view of any of the organisms in the union. They were therefore counted separately from all participating genomes. All these numbers are listed in table 9.5, and the lowest of the numbers are illustrated in the cartoon in figure 9.10. The fact that *S. cerevisiae* in most cases contains more proteins than its counterparts is partly due to the fact that a completely non-redundant set of *S. cerevisiae* proteins was not obtained (see Materials and Methods).

Almost all of the proteins shared between all three organisms, have a function that can be inferred by sequence similarity. Of the 294 *H. influenzae* proteins, 251 had functional annotation provided by TIGR. We analysed the remaining 43 proteins and found that 38 could be assigned a function with high confidence, leaving only 5 genes without a putative function (HI0090, HI0174, HI0271, HI0719 and HI1715). It will be interesting to find out what the function of these proteins is. Given that these proteins are conserved throughout so many phyla, they are likely to be of fundamental importance.

Table 9.5. Common subsets of proteins shared between genomes from organisms representing bacteria, fungi and animalia. In the overlap cases, where the numbers can be counted from either genome, they are listed in the same order as the species in the left column. Within brackets are the percentages of the genome that was counted from.

Organism combination	Proteins
C. elegans NOT (*S. cerevisiae* OR *H. influenzae*)	5049 (70%)
S. cerevisiae NOT (*C. elegans* OR *H. influenzae*)	3973 (59%)
H. influenzae NOT (*C. elegans* OR *S. cerevisiae*)	1135 (68%)
(*C. elegans* AND *S. cerevisiae*) NOT *H. influenzae*	1760 (24%), 1843 (27%)
(*C. elegans* AND *H. influenzae*) NOT *S. cerevisiae*	58 (1%), 46 (3%)
(*S.* cerevisiae AND H. *influenzae*) NOT *C. elegans*	299 (4%), 205 (12%)
C. elegans AND *S. cerevisiae* AND *H. influenzae*	396 (5%), 604 (9%), 294 (17%)

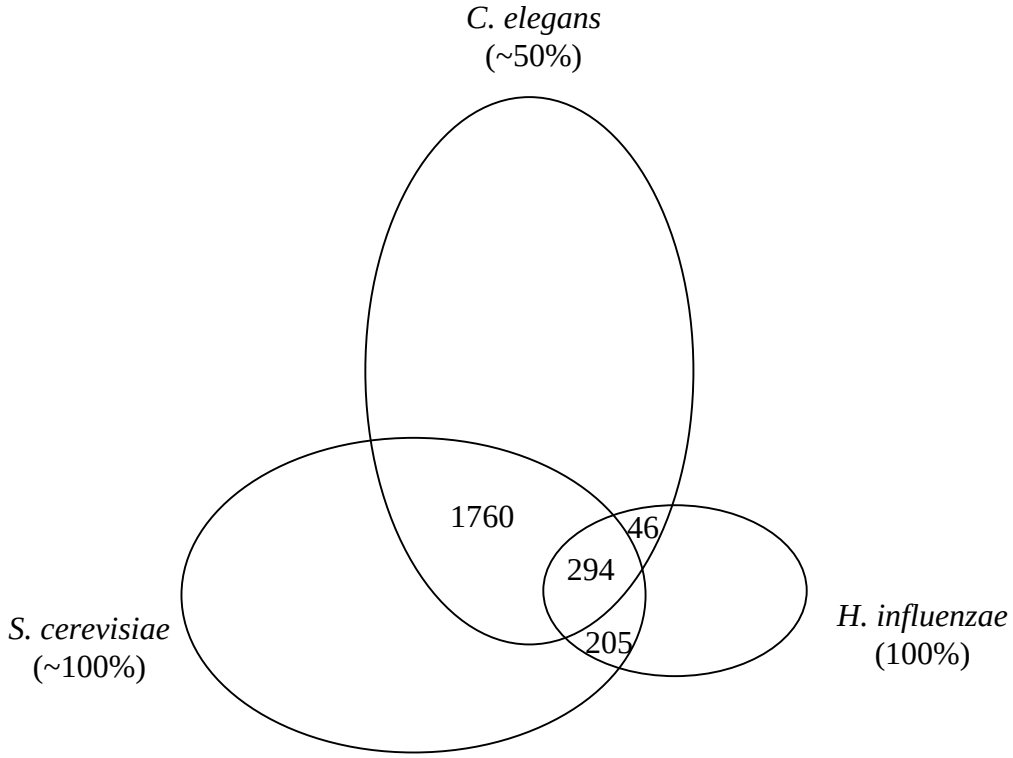


Figure 9.10. Diagram of common proteins shared between three kingdoms. The number shown in intersecting areas is the lowest of the participating genomes.

The most obvious reason that *C. elegans* and *S. cerevisiae* have so many more similar proteins than with *H. influenzae* seems to be eukaryotic protein kinases, which apparently are absent from *H. influenzae*. A number of other protein families are also specific to eukaryotes, such as histones, tubulin, and much of the proteins involved in transcription, translation and replication. Only a few proteins were found to be unique to *C. elegans* and *H. influenzae*. These are listed in table 9.6. Many of these appear to be metabolic enzymes involved in biosynthesis, but most cellular roles seem to be represented. We were suprised to note strong similarity between DNA polymerase I (HI0856) in *H. influenzae* and W03A3.2 in *C. elegans*, yet no similarity was found to a yeast protein. No other DNA polymerases were similar between *C. elegans* and *H. influenzae*. This type of DNA polymerase I was also absent from the human proteins, but the yeast and human polymerases are very similar to each other. There are thus instances, where conservation of molecular mechanisms does not follow the groupings of the traditional phylogenetic tree of life.

Seven of the 46 *H. influenzae* genes did not have functional annotation provided by TIGR. One of these (HI0323) had been assigned a function in an previous reanalysis [Casari *et al.*, 1995], and three were assigned a putative function (HI0152, HI0392 and HI1663) here, leaving three genes without a putative function (see table 9.6).

Table 9.6. The 46 *H. influenzae* proteins that match *C. elegans* proteins but not *S. cerevisiae*. The functional assignments were taken from TIGR except in four cases that lacked annotation (HI0152, HI0323, HI0392 and HI1663). Three ORFs did not match any functionally characterised proteins.

H. influenzae ORF	Functional annotation
HI0019	
HI0140	N-acetylglucosamine-6-phosphate deacetylase (nagA)
HI0151	Protease specific for phage lambda cII repressor (hflK)
HI0152	Transcription factor
HI0211	Phosphatidylglycerophosphate phosphatase B (pgpB)
HI0244	tRNA-guanine transglycosylase (tgt)
HI0259	UDP-glycose:glycoprotein glycosyltransferase
HI0280	Uridine phosphorylase (udp)
HI0323	Lactoylglutathione lyase
HI0340	
HI0392	Acyl transferase
HI0406	Acetyl-coenzyme A carboxylase (accA)

HI0478	ATP synthase F1 epsilon subunit (atpC)
HI0550	Lipooligosaccharide biosynthesis protein
HI0701	
HI0714	ATP-dependent clp protease proteolytic component (clpP)
HI0736	Sodium-dependent noradrenaline transporter
HI0759	A/G-specific adenine glycosylase (mutY)
HI0765	Lipooligosaccharide biosynthesis protein
HI0773	3-oxoacid CoA-transferase
HI0774	Butyrate-acetoacetate coenzyme A transferase subunit A (ctfA)
HI0856	DNA polymerase I (polA)
HI0910	Mutator mutT (AT-GC transversion)
HI0975	Pantothenate permease (panF)
HI0991	DNA/ATP binding protein (recF)
HI1013	Glyoxylate-induced protein
HI1042	Methyltetrahydrofolate transmethylase (metH)
HI1075	Cytochrome oxidase d subunit II (cydB)
HI1115	Thioredoxin (trxA)
HI1116	Deoxyribose aldolase (deoC)
HI1219	Cytidylate kinase (cmk)
HI1260	Folylpolyglutamate-dihydrofolate synthetase expression regulator (accD)
HI1324	Lon protease (lon)
HI1362	NAD(P) transhydrogenase subunit alpha (pntA)
HI1363	NAD(P) transhydrogenase subunit beta (pntB)
HI1441	Stringent starvation protein A (sspA)
HI1448	Molybdopterin biosynthesis protein (chlE)
HI1526	Autotrophic growth protein (aut)
HI1545	C4-dicarboxylate transport protein
HI1588	Formyltetrahydrofolate hydrolase (purU)
HI1646	Cytidylate kinase (cmk)
HI1663	Glyoxalase
HI1675	Molybdenum cofactor biosynthesis protein (moaC)
HI1676	Molybdenum cofactor biosynthesis protein A (moaA)
HI1690	Na+ and Cl- dependent gamma-aminobutryic acid transporter
HI1705	Aminopeptidase a/i (pepA)

Human homologues in C. elegans

Nearly two thirds of human proteins have a homologue in 50% of *C. elegans* proteins. This figure is inflated mainly by two factors: the known human sequences are biased towards well-known and ubiquitous families, and because most *C. elegans* proteins occur in families of paralogues. We expect that most protein families in *C. elegans* already have at least one representative in Wormpep 11, and that a majority of the human proteins that have a homologue in *C. elegans* already should have a match. To estimate the fraction of human proteins that will have a match to the entire *C. elegans* genome, we fitted a curve to a number of smaller sets of *C. elegans* proteins, as shown in figure 9.11. This curve suggests that approximately 70% of the human proteins in the set would match the entire *C. elegans* genome, which is only 10% more than the fraction that matches half of it. This means that 85% of the

human proteins, for which a *C. elegans* homologue exists, would already have a detectable match to at least a paralogue.

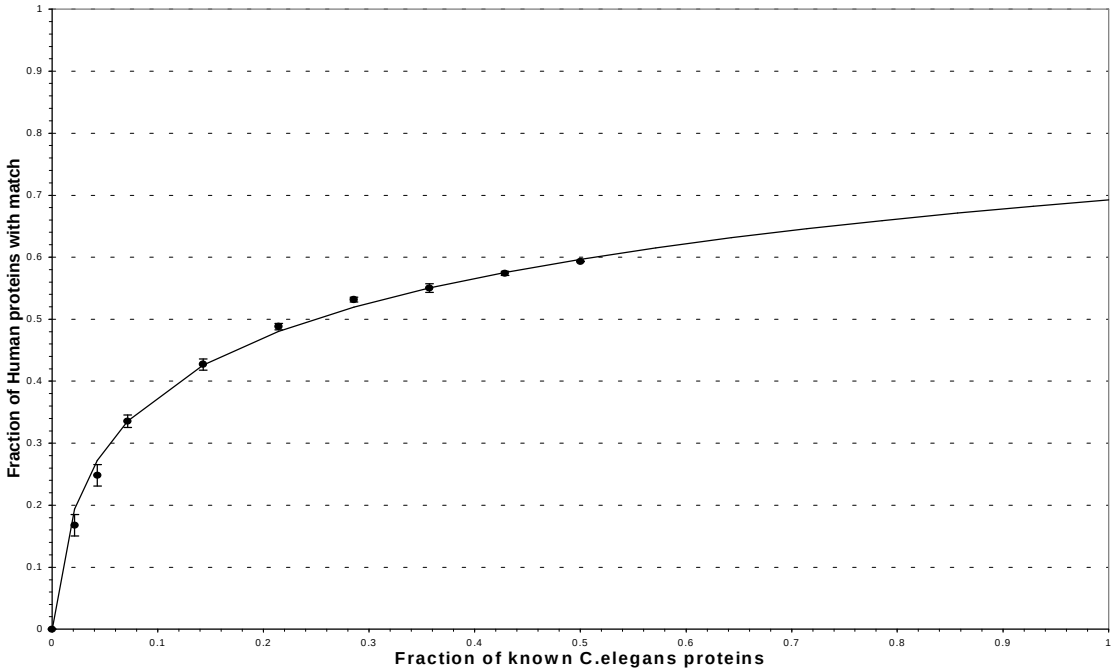


Figure 9.11. Projection of the fraction of human proteins that match *C. elegans* proteins for different fractions of known *C. elegans* proteins. The datapoints below 50% were simulated by taking fractions of the currently known *C.* elegans proteins in Wormpep 11. The values are averages from 3 independent experiments and the errorbars are standard deviations.

As mentioned before, the number of true orthologues is significantly lower than the number of matching proteins. This is illustrated in figure 9.12. Since non-orthologous homologues may have diverged in function, they are often less useful for precise inference of biological information. We have estimated the number of orthologues between the human and *C. elegans* datasets by looking for homologues that are the most similar pair of proteins, as seen from both genomes. This is usually the case for true orthologues.

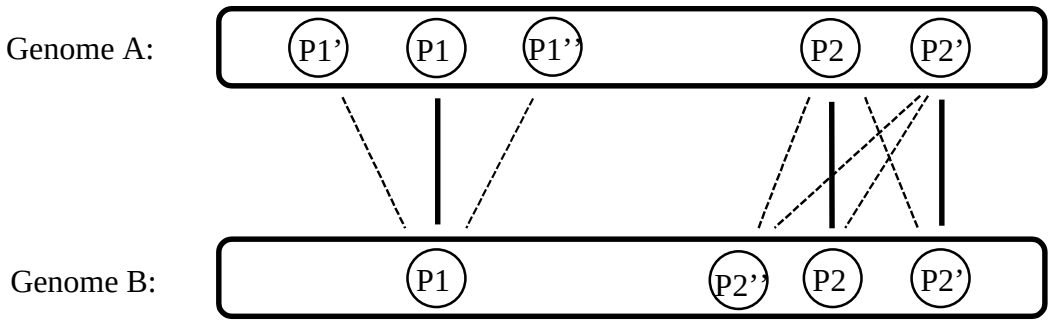


Figure 9.12. When comparing two genomes, the number of matching proteins can be higher than the number of orthologues due to cross-reaction with paralogues. In the example above there are three orthologous proteins, P1, P2 and P2'. P2 and P2' are paralogues that arose by gene duplication before the species A and B separated, while P1'' and P2'' arose afterwards. The number of orthologues will be overestimated by counting every matching protein. This effect can be reduced by only counting proteins that are reciprocally the most similar pair.

Further evidence for orthology is that both proteins are equally long and match over the entire length. This property should be used with care here, since the *C. elegans* proteins were predicted from genomic DNA and may not always have the correct N and C-terminus due to lack of experimental evidence, and because the extent of the match was estimated using Blastp, which sometimes only report part of the match.

Of the 2077 human proteins that match *C. elegans*, 744 had reciprocally best partners. This number of proteins are thus likely to have true orthologues. Requiring that both proteins have to match each other by more than 80% reduces the number to 257. Given that only about 5% of the human proteins were used in the analysis, many of the true orthologues may not have been sequenced yet. However, the human and *C. elegans* proteins that fulfil the stringent criteria mentioned above are likely to have very similar functions even if they are not true orthologues.

The number of detectable orthologue relationships should grow more linearly than the curve of homologues in figure 9.11. When the *C. elegans* genome is finished, we would expect a significant increase in orthologue relationships compared to now, although probably not twice as many. A greater increase in orthologous partners will be the result of complet-

ing the human genome. Given that *C. elegans* is estimated to contain no more than 15000 genes, and only a third of the homologues above were deemed likely orthologues, it is not reasonable to expect more than 5000 eventual orthologue pairs.

We refrained from performing the opposite extrapolation of what fraction of *C. elegans* proteins would match larger fractions of the human genome, since basing such an estimate on only 5% of all human proteins will make the number at 100% highly unreliable.

9.8 Materials and Methods

The clustering of Wormpep was performed by version 1.6 of the Domainer program [Sonnhammer, 1996], using pairwise homology information from Blastp version 1.4. Blastp [Altschul *et al.*, 1990] was used with the BLOSUM62 substitution matrix, and only matches scoring above 90 were used. The Blastp output was filtering by MSPcrunch (chapter 4) to remove biased composition matches, trim off overlapping ends of consecutive matches, and to reduce redundancy.

Wormpep 11 contains 36 protein sequences that are alternatively spliced versions of other genes. Since these are not true paralogues from a different locus, they were excluded from the clustering analysis. Arbitrarily, the first listed version was included. Normally the difference between alternatively spliced genes is just one exon, coding for a few tens of amino acids, so the loss of information by this procedure is negligible.

The Pfam matching was performed with the hmmfs and hmmls search programs, which are part of the HMMER package [Eddy, 1995a].

The nematode-specific families were analysed by running HMMs derived from the multiple alignments against swir11, which is a non-redundant combination of Wormpep 11, Swissprot 33 and Swissprot-TREMBL. Prosite patterns were searched with the perl script queryprosite, and coiled coil predictions were done with the program Pepcoil, which is part of the EGCG package [Rice *et al.*, 1995] and uses the algorithm by Lupas *et al.* [1991].

The proteins sets for the pairwise genome to genome comparisons were assembled the following way. *H. sapiens*: all entries in Swissprot 33. *C.* elegans: all entries in Wormpep 11, except alternatively spliced versions of the same gene (the A version was arbitrarily selected). *S. cerevisiae*: all entries in Swissprot 33 and all entries in Swissprot-TREMBL that were not 100% identical to (a part of) a Swissprot entry. *H. influenzae*: all entries in the TIGR set (ftp://ftp.tigr.org/pub/data/h_influenzae). The human and yeast datasets contained both nuclear and mitochondrial encoded proteins. The 13 mitochondrial *C. elegans* proteins in Swissprot 33 were not included. The *S. cerevisiae* dataset was somewhat redundant even after excluding the 100% matching and included sequences. We did not wish to remove less than 100% identical proteins on the basis of similarity only, to avoid removing very similar paralogues.

For the pairwise genome to genome comparisons, Blastp was used with MSPcrunch (chapter 4), to efficiently filter out false matches in the twilight zone. Compared to a straight score cutoff of 80 or 90 for each ungapped matching segment, this method detects a great deal more true matches. The MSPcrunch parameters were set more stringently than the default, to reduce the number of spurious matches. We found that raising the score range of the 'twilight zone' from 35-75 to 45-80, and the bias composition criterion to 0.8 with no pseudocounts, removed virtually all spurious matches with only a small loss of true matches. The accuracy was assessed by manual inspection of a few genome comparisons in Blixem (chapter 3) and Dotter (chapter 5) using the Blixelect multi-query results organiser. In the *C. elegans* to *H. sapiens* comparison, 125 protein assignments were removed by the increase in MSPcrunch stringency. Of these, only 9 were found likely to be true matches. We also performed the same analysis on the 150 assignments (2% of the *C. elegans* proteins) in the *C. elegans* to S. *cerevisiae* comparison that only had matches scoring below 80. Of these, only about 10 were dubious. Very few matches with scores above 80 were false.

Our method should thus be a good compromise between sensitivity and selectivity for genomic comparison purposes. An alternative approach would be to apply other types of programs for postprocessing matches in the twilight zone, such as dynamic programming and multiple alignment methods [Koonin *et al.*, 1996b; Tatusov *et al.*, 1996]. MSPcrunch could

also have been used in a less stringent mode, combined with manual processing of twilight zone matches. For a detailed analysis, manual inspection of the results is essential. However, the accuracy achieved by MSPcrunch without manual processing in the twilight zone seems adequate for a reasonably reliable estimate of the overall percentage similarity between genomes. If anything, our method is conservative; we accept missing some weak matches as a tradeoff for rejecting most spurious ones.

A further uncertainty in the numbers of matching proteins is caused by counting whole proteins as units. A more accurate method would be to give the numbers of matching domains. Pfam could be exploited for this, but many cases would require labour-intense manual processing, which is unsuitable for large-scale analysis.

When comparing three genomes with each other, a 'bridging' situation that often occurs is when one genome contains a protein which is significantly homologous with proteins from the two other genomes, which do not show significant similarity to each other. We noted several cases of this in the *C. elegans* / *S. cerevisiae* / *H. influenzae* comparison. For example, the killer toxin-resistance protein WP:F48E3.3 is similar to KRE5_YEAST (P22023) and HI0259, but there is no discernable similarity between the *S. cerevisiae* and the *H. influenzae* proteins. In such cases, one could in principle infer homology indirectly. We have not pursued this strategy here, since most of the bridging cases require a much more thorough manual analysis to provide conclusive evidence of homology.

The smaller sets of *C.* elegans proteins in the Human to *C. elegans* comparison were generated by leaving out randomly chosen proteins from Wormpep 11. It was not done by removing all proteins from the same cosmid at once, which might have yielded slightly lower values due to clustering of gene families. The regression was performed by fitting a logarithmic function to the simulated datapoints in Microsoft Excel.

A further criterion that can be used for orthology, is that the two genes in question must be more similar to each other than to homologues from phylogenetically more distantly related organisms [Tatusov *et al.*, 1996]. For the human to *C. elegans* comparison, this would be a fungi, plants or bacteria. We only found one such case (The putative DNA helicase M03C11.2 is more similar to the yeast protein CHL1_YEAST (Swissprot: P22516) than to

the inferred human orthologue XPD_HUMAN (Swissprot P18074). One reason for finding so few cases is that a large proportion of the proteins are only found in animalia.

The human dataset could have been augmented by using EST data. We chose to not use this data, since its fragmentary nature makes the estimate of the number of matches and their extent uncertain.

9.9 Discussion

This chapter has provided a glimpse into what can be learned from data generated by genomic sequencing projects. Some results were surprising while others were more or less expected. Molecular biology research before the genome projects had already indicated that many protein domains are conserved between distantly related organisms, while some appear to be unique. With entire genome sequences such notions can be quantified, and detailed answers can be given about which families are most widespread. This knowledge will have a profound impact on biology and guide experimental research in new interesting directions.

Perhaps one of the most striking results is the estimate that about 70% of the currently known human genes will have a homologue in the invertebrate *C. elegans*. This underlines the appropriateness and usefulness of studying this nematode, and we can expect that the unravelling of molecular biological phenomena in it will greatly assist the understanding of human biology. One should keep in mind however, that the proportion of human homologues is likely to decrease in the future as a less biased set of human genes is produced by genomic sequencing.

Another striking result is that most protein domains that are conserved in distantly related organisms have been biochemically characterised already. This is exemplified by the fact that of the 294 *H. influenzae* proteins also found in *C. elegans* and *S. cerevisiae*, only 5 had no functionally annotated homologues. This was also the case for only 3 of the 46 proteins found in *C. elegans* and *H. influenzae* but not in *S. cerevisiae*. This strongly supports the ancient conserved region (ACR) theory [Green *et al.*, 1993], which was based on that over 90%

of newly found ACRs were already in the databases. We found that this figure is now at least 95%.

One of the most fundamental questions in bioinformatics is how much functional information can be inferred from a particular similarity. Obviously, the more sequence similarity between two proteins, the more likely they are to have similar functions. Here the concept of orthologous pairs comes in, which are usually proteins with identical functions. We have addressed this in the *C. elegans* to *H. sapiens* comparison, and found that it is likely to be true for 15-30% of the homologies. Non-orthologous homology, which often has a lower level of similarity, still allows many general features to be inferred, such as putative nucleotide binding moieties, protein-protein interaction domains, or catalytic activities. In such cases, the substrate(s) and the cellular role(s) can not be inferred from the homology. The scenario is more complicated if proteins in different organisms that perform identical functions, for instance a catalytic step in a metabolic pathway, have evolved from different ancestors. A number of such cases of 'non-orthologous gene displacement' have recently been discovered [Koonin *et al.,* 1996a].

Homologous proteins (i.e. that were derived from a common ancestor) often have similar sequences, because of the functional and structural constraints imposed on it. After long time spans, however, mutations accumulate, and the amino acid sequences may drift beyond the point of recognition. Performing the analysis on the basis of sequence similarity may therefore not necessarily give the ultimate answer. The methods based on comparing one sequence with another are probably close to being as sensitive as they can ever get. It is possible to look further back in time by using multiple alignment methods, since truly important features stand out more prominently. This is exemplified by the fact that many of the nematode-specific families initially could not be assigned a function when only a few members were known. But as more members were gathered in families nr. 4 and 5, it became increasingly clear that they were likely G-protein coupled receptors. Some families, e.g. nr. 8, which still only has a small number of (very similar) members, defy functional prediction today. This may change in the future as more members are found.

Still, *C. elegans* and other organisms seem to contain a large number of unique families with few members. Are these truly unique protein families with novel folds? The answer must be sought with more sophisticated analysis methods than pure sequence comparison. Structural threading methods, that fit a sequence to known structures in order to find the most likely fold, may give an answer. However, it is not always clear what functional information can be transferred in such cases.

Our capability to recognise homologues was shown to be improved by searching a database of pre-assembled protein families such as Pfam, as an addition to traditional single-sequence database searching. Although the number of proteins that changed status from 'function unknown' to 'putative function' was not enormous, a large number of novel and supportive domain classifications were found. Since this analysis was based on version 1.0 of Pfam, we can expect a significant increase in the usefulness of this approach in the future.

The clustering of *C. elegans* proteins and the distribution of the families that appear to be nematode-specific provides insights in the general mechanism of evolution of paralogues within a genome. It seems that the genome is constantly shuffled around; chunks of DNA are duplicated, preferentially next door to the original. In many cases, this must have lethal effects, but what we observe today in the living organism is the result of the accumulation of a countless number of 'lucky accidents'.

The fact that 46 proteins occur in both *C. elegans* and *H. influenzae*, but not in yeast, suggests that evolution in many cases proceeds by gene loss followed by replacement, or by horizontal transfer from one organism to an other. Some of the observations could also be caused by different rates of genetic drift in yeast of these proteins, which made it impossible to recognise a true homologue. A more thorough analysis would be necessary to establish which hypothesis is most likely for each individual case.

An aspect of protein function that is of vital importance to biology is how proteins interact with each other in the network of pathways that make up a living cell. That regulation and signal transduction is a major aspect of metazoan life is evident from the large number of protein kinases, receptors and transcription factors found in *C. elegans*. The quest for under-

standing molecular mechanisms on this level will be one of the greatest future challenges in biology.

10. Conclusion and future perspectives

Just as Columbus little could have imagined how the world would change from his discovery, so it is equally hard for us to envisage the changes to science and society that large-scale genome sequencing will eventually bring about. One thing is certain: it will have a profound impact on medical research, both directly by for example improving the ability to detect gene mutations, and indirectly by providing a crucial information resource for biological research. Knowing the complete genomic sequence is of course only a beginning; the work required to discover new biological functions and mechanisms will still be needed. But we can expect that this work will be greatly accelerated by having the sequence as a blueprint. In some ways, genome research can be likened with the extensive efforts to map the continents of the world in the 16th and 17th centuries. Just as then, the global goal of the endeavour is clear, which was to find new sources of riches and trade routes to bring them home, but exactly where they will be discovered can not be known beforehand.

We are still in the early days of genome research. Many of the issues that need to be addressed at this time are of a fundamental nature. Most scientists in the field are not yet used to working with the vast quantities of data that are being produced, and it will take a long learning process before the full potential of the information is realised. The increase in sequence data therefore has to be accompanied by a new breed of efficient analysis and visualisation tools. Most of this thesis is taken up with methods that address this very point: how computers can assist humans optimally in performing large-scale sequence analysis.

Already, the high sequencing rates may seem a burden for the person who has to analyse the sequence. Compared to the past, when a year's experimental work could be analysed manually in a few days, this may seem true. But in fact today it only takes a few hours to analyse a cosmid sequence, which it has taken one person about a month to finish. However, using the analysis methods of the past, it may well have taken over a month to analyse it. As sequencing becomes more automated, so do analysis methods, and there is no indication that analysis will ever be the bottleneck. To keep abreast of the sequence flow will however most

likely continue to require a significant amount of analysis labour, and a steady improvement of the software tools.

The long term building up of knowledge about genes and proteins is one of the most concrete goals of genome sequencing. By transferring functional information from biochemically characterised genes using homology analysis, a vast body of genes with predicted functions is gathered. The proportion of proteins with an experimentally determined function in sequence databases is therefore steadily dropping. This is not necessarily a problem, since strong sequence similarity has been shown to correspond well to homologous functions. Computational methods for separating spurious similarities from true homologies have been a central theme in this thesis.

In recent years there has been a clear increase in the fraction of proteins from genome projects that can be assigned a function on the basis of sequence similarity. As discussed in chapter 9, there is still a significant proportion of proteins for which this is not the case, but most of these 'hypothetical' proteins are only found in one species and its closest relatives. Whether they are homologues of known genes, but have drifted too far in the sequence to be recognised as such, or whether they are truly unique lines of proteins must still be unravelled. To some extent, this will be done by a refinement of homology analysis methods, such as systematic use of multiple alignments in order to look back further in time. However, there is still a large need for systematic biochemical characterisation of these newly found proteins. The effort to undertake this effort for all such proteins in yeast [Oliver, 1996] will soon indicate just how possible such an endeavour is.

Genomic sequence analysis plays a central role in the concerted genome sequencing effort. A network of different disciplines, such as genetic and physical mapping, and all other aspects of biological research can be linked together with the sequence as a reference. One of the main goals is to bring data together from these different sources, and make them available for 'in silico' analysis. This will become an increasingly important approach for molecular biology research. Figure 10.1 illustrates the main flows of data between experimental disciplines and computational analysis in a genome sequencing project. The results of such projects are not well suited for traditional paper publishing. It is therefore essential that both

the data and the visualisation tools are electronically distributed throughout the world, and that the data is curated at a high quality.

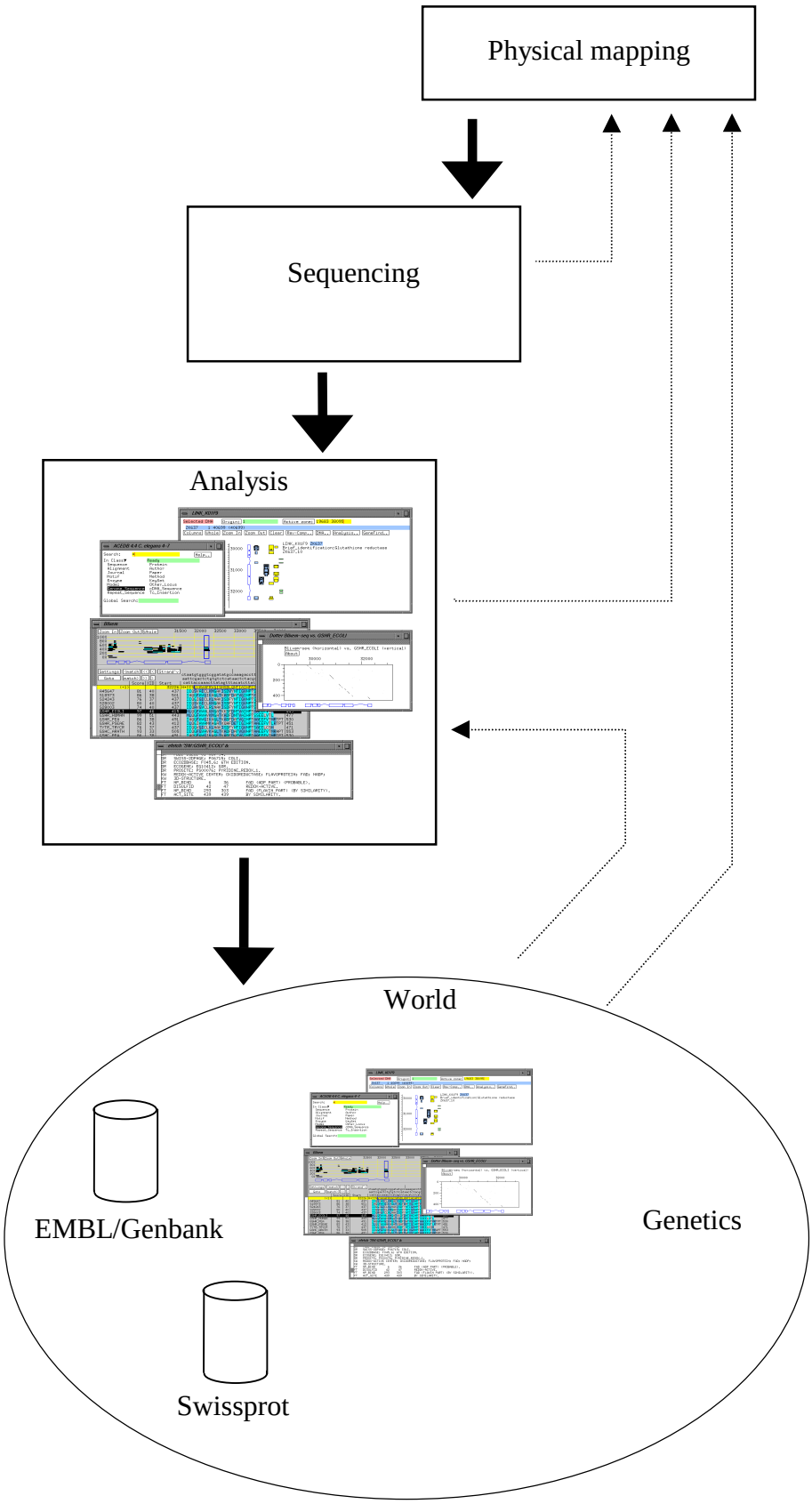


Figure 10.1. Global view of the information flow in the *C. elegans* genome sequencing project. Solid lines indicate the main production line flow information, while the dashed lines are feedback channels that reinforce the sources.

11. Acknowledgements

I would like to express my profound gratitude to my primary supervisor Richard Durbin, without whom this work would never have been possible. I am also very grateful to my second supervisor John Sulston, who has been very encouraging and stimulating during my time at the Sanger Centre. A very special thanks goes to Sean Eddy for his input to this work, especially for the collaboration on the Pfam database. The whole informatics group has been a great help throughout. I wish to particularly thank Anders Krogh, Daniel Lawson and Steve Jones and Julie Ahringer for their comments on parts of the text in this thesis. Many thanks also go to members of the Computational Molecular Biology Discussion Group at the LMB for fruitful discussions. I also thank the colleagues at the EBI for discussions and for the colour prints in this thesis, the worm sequencing teams at the Sanger Centre, and the worm group at the LMB.

Very special personal thanks go to Ana Vaz Gomes for her support, and for comments on parts of this thesis. Finally, I am very grateful to my parents and family in Sweden who have been very supportive and patient during my long exile.

The Sanger Centre is supported by the Wellcome Trust and the MRC.

References

Adams MD, Kerlavage AR, Fleischmann RD, Fuldner RA, Bult CJ, Lee NH, Kirkness EF, Weinstock KG, Gocayene JD, White O, *et al.*, Venter JC. (1995) Initial assessment of human diversity and expression patterns based upon 83 million nucleotides of cDNA sequence. *Nature* **377 supplement**:3-174

Altschul SF. (1989) Gap costs for multiple sequence alignment. *J. Theor. Biol.* **138**:297-309

Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. (1990) Basic local alignment search tool. *J. Mol. Biol.* **215**:403-410

Altschul SF, Boguski MS, Gish W, Wootton JC. (1994) Issues in searching molecular sequence databases. *Nat. Genet.* **6**:119-129

Andersson SG, Zomorodipour A, Winkler HH, Kurland C. (1995) Unusual organization of the rRNA genes in *Rickettsia prowazekii*. *J. Bacteriol.* **177**:4171-4175

Appel RD, Bairoch A, Hochstrasser DF. (1994) A new generation of information retrieval tools for biologists: the example of the ExPASy WWW server. *Trends Biochem. Sci.* **19**:258-260

Argos P. (1987) A sensitive procedure to compare amino acid sequences. *J. Mol. Biol.* **193**:385-396

Argos P, Vingron M. (1990) Sensitivity comparison of protein amino acid sequences. *Meth. Enzymol.* **183**:352-365

Attwood TK, Beck ME. (1994) PRINTS - a protein motif fingerprint database. *Protein Eng.* **7**:841-848

Attwood TK, Beck ME, Bleasby AJ, Degtyarenko K, Parry Smith DJ. (1996) Progress with the PRINTS protein fingerprint database. *Nucleic Acids Res.* **24**:182-189

Bacon DJ, Anderson WF. (1986) Multiple sequence alignment. *J. Mol. Biol.* **191**:153-161

Bairoch A, Apweiler R. (1996) The SWISS-PROT protein sequence data bank and its new supplement TREMBL. *Nucleic Acids Res.* **24**:21-25

Bairoch A, Bucher P, Hofmann K. (1996) The PROSITE database, its status in 1995. *Nucleic Acids Res.* **24**:189-196

Barton GJ. (1993a) ALSCRIPT: a tool to format multiple sequence alignments. *Protein Eng.* **6**:37-40

Barton GJ. (1993b) An efficient algorithm to locate all locally optimal alignments between two sequences allowing for gaps. *Comput. Appl. Biosci.* **9**:729-734

Barton GJ, Sternberg MJE. (1987) A strategy for the rapid multiple alignment of protein sequences. *J. Mol. Biol.* **198**:327-337

Bazan JF. (1990) Structural design and molecular evolution of a cytokine receptor superfamily. *Proc. Natl. Acac. Sci. U.S.A.* **87**:6934-6938

Benian GM, Kiff JE, Neckelmann N, Moerman DG, Waterston RH. (1989) Sequence of an unusually large protein implicated in regulation of myosin activity in C. elegans. *Nature* **342**:45-50

Benian GM, L'Hernault SW, Morris ME. (1993) Additional Sequence Complexity in the Muscle Gene, unc-22, and Its Encoded Protein, Twitchin, of Caenorhabditis elegans. *Genetics* **134**:1097-1104

Benson DA, Boguski M, Lipman DJ, Ostell J. (1996) GenBank. *Nucleic. Acids. Res.* **24**:1-5

Bernstein M. (1987) Reducing the man - machine barrier: the sequence analysis workbench. *Comput. Appl. Biosci.* **3**:229-232

Birney E, Thompson JD, Gibson TJ. (1996) PairWise and SearchWise: finding the optimal alignment in a simultaneous comparison of a protein profile against all DNA translation frames. *Nucleic Acids Res.* **24**:2730-2739

Bisson G, Garreau A (1995) APIC: a generic interface for sequencing projects. In *ISMB-95; Proceedings Third International Conference on Intelligent Systems for Molecular Biology*, 57-65, AAAI Press, Menlo Park.

Boguski MS, Hardison RC, Schwartz S, Miller W. (1992) Analysis of conserved domains and sequence motifs in cellular regulatory proteins and locus control regions using new software tools for multiple alignment and visualization. *New Biol.* **4**:247-260

Boguski MS, Schuler GD. (1995) ESTablishing a human transcript map. *Nat. Genet.* **10**:369-371

Bork P. (1993) The modular architecture of a new family of growth regulators related to connective tissue growth factor. *FEBS Lett.* **2**:125-130

Bork P, Doolittle RF. (1992) Proposed acquisition of an animal protein domain by bacteria. *Proc. Natl. Acac. Sci. U.S.A.* **89**:8990-8994

Bork P, Ouzounis C, Casari G, Schneider R, Sander C, Dolan M, Gilbert W, Gillevet PM. (1995) Exploring the Mycoplasma capricolum genome: a minimal cell reveals its physiology. *Mol. Microbiol.* **16**:955-967

Borodovsky MY, Rudd KE, Koonin EV. (1994) Intrinsic and extrinsic approaches for detecting genes in a bacterial genome. *Nucleic Acids Res.* **22**:4756-4767

Borodovsky M, McIninch JD, Koonin EV, Rudd KE, Medigue C, Danchin A. (1995) Detection of new genes in a bacterial genome using Markov models for three gene classes. *Nucleic. Acids. Res.* **23**:3554-3562

Brenner SE, Hubbard T, Murzin A, Chothia C. (1995) Gene duplications in *H. influenzae*. *Nature* **378**:140

Brunak S, Engelbrecht J, Knudsen S. (1991) Prediction of human mRNA donor and acceptor sites from the DNA sequence. *J. Mol. Biol.* **220**:49-65

Brutlag DL, Dautricourt J-P, Diaz R, Fier J, Moxon B, Stamm R. (1993) BLAZE (tm): an implementation of the Smith-Waterman sequence comparison algorithm on a massively parallel computer. *Comput. Chem.* **17**:203-207

Bulmer M. (1987) Coevolution of codon usage and tranfer RNA abundance. *Nature* **325**:728-730

Bult CJ, White O, Olsen GJ, Zhou L, Fleischmann RD, Sutton GG, Blake JA, FitzGerald LM, Clayton RA, Gocayne JD, Kerlavage AR, Dougherty BA, Tomb J-F, Adams MD, Reich CI, Overbeek R, Kirkness EF, Weinstock KG, Merrick JM, Glodek A, Scott JL, Geoghagen NS, Weidman JF, Fuhrmann JL, Nguyen D, Utterback TR, Kelley JM, Peterson JD, Sadow PW, Hanna MC, Cotton MD, Roberts KM, Hurst MA, Kaine BP, Borodovsky M, Klenk H-P, Fraser CM, Smith HO, Woese CR, Venter JC. (1996) Complete Genome Sequence of the Methanogenic Archaeon, *Methanococcus jannaschii*. *Science* **273**:1058-1073

Califano A, Rigoutsos I (1993) FLASH: A Fast Look-Up Algorithm for String Homology. In *ISMB-93; Proceedings First International Conference on Intelligent Systems for Molecular Biology*, 56-64, AAAI Press, Menlo Park.

Carrillo H, Lipman DJ. (1988) The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.* **48**:1073-1082

Casari G, Andrade MA, Bork P, Boyle J, Daruvar A, Ouzounis C, Schneider R, Tamames J, Valencia A, Sander C. (1995) Challenging times for bioinformatics. *Nature* **376**:647-648

Casari G, De Daruvar A, Sander C, Schneider R. (1996) Bioinformatics and the discovery of gene function. *Trends Genet.* **12**:244-245

Claros MG, von-Heijne G. (1994) TopPred II: an improved software for membrane protein structure prediction. *Comput. Appl. Biosci.* **10**:685-686

Claverie J-M, States DJ. (1993) Information enhancement methods for large scale sequence analysis. *Comput. Chem.* In Press

Collins FS. (1995) Ahead of schedule and under budget: The Genome Project passes its fifth birthday. *Proc. Natl. Acad. Sci. U. S. A.* **92**:10821-10823

Collins JF, Coulson AFW. (1990) Significance of protein sequence similarities. *Meth. Enzymol.* **183**:474-487

Collins JF, Coulson AFW, Lyall A. (1988) The significance of protein sequence similarities. *Comput. Appl. Biosci.* **4**:67-71

Crook J. (1991) In *New algorithms and methods for protein and DNA sequence comparison, Ph.D. Thesis,* University of Edinburgh, United Kingdom.

Dayhoff MO (1978) Survey of new data and computer methods of analysis. In *Atlas of Protein Sequence and Structure,* 5 suppl. 3:1-8, Nat. Biomed. Res. Found., Washington D.C.

De Rijk P, De Wachter R. (1993) DCSE, an interactive tool for sequence alignment and secondary structure research. *Comput. Appl. Biosci.* **9**(6):735-740

Depiereux E, Feytmans E. (1992) MATCH-BOX: a fundamentally new algorithm for the simultaneous alignment of several protein sequences. *Comput. Appl. Biosci.* **8**:501-509

Devereux J, Haeberli P, Smithies O. (1984) A comprehensive set of sequence analysis programs for the VAX. *Nucleic Acids Res.* **12**:387-395

Dujon B. (1996) The yeast genome project: what did we learn? *Trends Genet.* **12**:263-270

Dunn J. (1996) *Human Genome News* **7**:13

Durbin R, Thierry-Mieg J (1996) In *The ACEDB genomic database,* World Wide Web URL: ftp://ftp.sanger.ac.uk/pub/acedb.

Eddy SR. (1996) Hidden Markov Models. *Curr. Opinion Struct. Biol.* **6**:361-365

Eddy SR (1995a) In *The HMMER package,* World Wide Web URL: http://genome.wustl.edu/eddy/hmm.html.

Eddy SR (1995b) Multiple alignment using hidden Markov models. In *ISMB-95; Proceedings Third International Conference on Intelligent Systems for Molecular Biology,* 114-120, AAAI Press, Menlo Park.

Eddy SR, Durbin R. (1994) RNA sequence analysis using covarience models. *Nucleic Acids Res.* **22**:2079-2088

Eddy SR, Mitchison G., Durbin R. (1995) Maximum Discrimination Hidden Markov Models of Sequence Consensus. *J. Comput. Biol.* **2**:9-23

Edgley ML, Riddle DL. (1990) *Genetic Maps* **5**:3

Esterman L. (1995) Bioccelerator: a currently available solution for fast profile and Smith-Waterman searches. *Embnet News* **2**:5-6

Etzold T, Argos P. (1993) SRS - an indexing and retrieval tool for flat file data libraries. *Comput. Appl. Biosci.* **9**:49-57

Farber R, Lapedes A, Sirotkin K. (1992) Determination of eukaryotic protein coding regions using neural networks and information theory. *J. Mol. Biol.* **226**:471-479

Feng D-F, Doolittle RF. (1987) Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.* **25**:351-360

Fichant GA, Burks C. (1991) Identifying potential tRNA genes in genomic DNA sequences. *J. Mol. Biol.* **220**:659-671

Fickett JW. (1996) Finding genes by computer: the state of the art. *Trends Genet.* **12**:316-320

Fields C, Adams MD, White O, Venter JC. (1994) How many genes in the human genome? *Nat. Genet.* **7**:345-346

Fitch WM. (1969) Locating gaps in amino acid sequences to optimize the homology between two proteins. *Biochem. Genet.* **3**:99-108

Fleischmann RD, Adams MD, White O, Clayton RA, Kirkness EF, Kerlavage AR, Bult CJ, Tomb J-F, Dougherty BA, Merrick JM, *et al.*, Venter JC. (1995) Whole-genome random sequencing and assembly of *Haemophilus influenzae* Rd. *Science* **269**:496-512

Fraser CM, Gocayne JD, White O, Adams MD, Clayton RA, Fleischmann RD, Bult CJ, Kerlavage AR, Sutton G, Kelley JM, Fritchman JL, Weidman JF, Small KV, Sandusky M, Fuhrmann J, Nguyen D, Utterback TR, Saudek DM, Phillips CA, Merrick JM, Tomb J-F, Dougherty BA, Bott KF, Hu P-C, Lucier TS, Peterson SN, Smith HO, Hutchison CA, Venter JC. (1995) The Minimal Gene Complement of *Mycoplasma genitalium*. *Science* **270**:397-403

Fristensky B. (1986) Improving the efficiency of dot-matrix similarity searches through use of an oligomer table. *Nucleic Acids Res.* **14**:597-610

Fuchs R, Stoehr PJ. (1993) EMBL-Search - a CD-ROM based database query system. *Comput. Appl. Biosci.* **9**:71-77

George DG, Barker WC, Mewes H-W, Pfeiffer F, Tsugita A. (1996) The PIR-International Protein Sequence Database. *Nucleic Acids Res.* **24**:17-21

Gerstein M, Sonnhammer ELL, Chothia C. (1994) Volume Changes in Protein Evolution. *J. Mol. Biol.* **236**:1067-1078

Gibbs AJ, McIntyre GA. (1970) The diagram: a method for comparing sequences. Its use with amino acid and nucleotide sequences. *Eur. J. Biochem.* **16**:1-11

Gibson TJ, Hyvönen M, Musacchio A, Saraste M. (1994) PH domain: the first anniversary. *Trends Biochem. Sci.* **19**:349-353

Giese KP, Martini R, Lemke G, Soriano P, Schachner M. (1992) Mouse P0 gene disruption leads to hypomyelination, abnormal expression of recognition molecules, and degeneration of myelin and axons. *Cell* **71**:565-576

Gotoh O. (1995) A weighting system and algorithm for aligning many phylogenetically related sequences. *Comput. Appl. Biosci.* **11**:543-551

Green P, Lipman DJ, Hillier L, Waterson R, State D, Claverie J-M. (1993) Ancient conserved regions in new gene sequences and the protein databases. *Science* **259**:1711-1716

Gribskov M, Homyak M, Edenfield J, Eisenberg D. (1988) Profile scanning for three-dimensional structural patterns in protein sequences. *Comput. Appl. Biosci.* **4**:61-66

Gribskov M, McLachlan M, Eisenberg D. (1987) Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad. Sci. U.S.A.* **84**:4355-4358

Gumbel EJ (1958) In *Statistics of extremes*, Columbia Univ. Press, New York.

Henikoff S, Henikoff JG. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U.S.A.* **89**:10915-10919

Henikoff S, Henikoff JG. (1994) Protein family classification based on searching a database of blocks. *Genomics* **19**:97-107

Henikoff S, Henikoff JG. (1996) Using substitution probabilities to improve position-specific scoring matrices. *Comput. Appl. Biosci.* **12**:135-143

Higgins DG, Bleasby AJ, Fuchs R. (1992) CLUSTAL V: improved software for multiple sequence alignment. *Comput. Appl. Biosci.* **8**:189-191

Hilbert H, Himmelreich R, Pirkl E, Plagens H, Proft T, Herrmann R. (1995) Analysis of the *Mycoplasma pneumoniae* genome. *Abstr. 95th Gen. Meet. Am. Soc. Microbiol.* 505

Hillier L, Lennon G, Becker M, Bonaldo M, Chiapelli B, Chissoe S, Dietrich N, DuBuque T, Favello A, Gish W, Hawkins M, Hultman M, Kucaba T, Lacy M, Le M, Le N, Mardis E, Moore B, Morris M, Parsons J, Prange C, Rifkin L, Rohlfing T, Schellenberg K, Soares M, Tan F, Trevaskis E, Underwood K, Wohldmann P, Waterston R, Wilson R, Marra M. (1996) Generation and analysis of 280,000 human expressed sequence tags. *Genome Research*, **in press**

Hirschberg DS. (1975) A Linear Space Algorithm for Computing Maximal Common Subsequences. *Comm. ACM* **18**:341-343

Hodgkin J, Plasterk R.H., Waterston RH. (1995) The nematode *Caenorhabditis elegans* and its genome. *Science* **270**:410-414

Holm L, Sander C. (1996) Mapping the Protein Universe. *Science* **273**:595-602

Hunter L, Harris N, States DJ (1992) Efficient Classification of Massive, Unsegmented Datastreams. In *International Machine Learning Workshop*, 224-232, San Mateo, CA.

Jagadeeswaran P, McGuire PMJr. (1982) Interactive computer program in sequence analysis. *Nucleic Acids Res.* **10**:433-447

Johnson MS, Doolittle RF. (1986) A method for the simultaneous alignment of three or more amino acid sequences. *J. Mol. Evol.* **23**:267-278

Johnston M. (1996) Towards a complete understanding of how a simple eukaryotic cell works. *Trends Genet.* **12**:242-243

Kanai N, Lu R, Satriano JA, Bao Y, Wolkoff AW, Schuster VL. (1995) Identification and Characterization of a Prostaglandin Transporter. *Science* **268**:866-869

Karlin S, Altschul SF. (1993) Applications and statistics for multiple high-scoring segments in molecular sequences. *Proc. Natl. Acad. Sci. U.S.A.* **90**:5873-5877

Karlin S, Altschul SF. (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. U.S.A.* **87**:2264-2268

Karreman C. (1992) A dotplot program for the Atari ST, for the analysis of DNA and protein sequences. *Comput. Appl. Biosci.* **8**:75-77

Kazal LA, Spicer DS, Brahinsky RA. (1948) Isolation of a Crystalline Trypsin Inhibitor-Anticoagulant Protein from Pancreas. *J. Am. Chem. Soc.* **70**:3034-3040

Kernighan BW, Ritchie DM (1988) In *The C Programming Language*, Prentice Hall, USA.

Koonin EV, Bork P, Sander C. (1994) Yeast chromosome III: new gene functions. *EMBO J.* **13**:493-503

Koonin EV, Mushegian AR, Bork P. (1996a) Non-orthologous gene displacement. *Trends Genet.* **12**:334-336

Koonin EV, Tatusov EV, Rudd KE. (1996b) Protein Sequence Comparison at Genome Scale. *Meth. Enz.* **266**:295-322

Krishnan G, Kaul RK, Jagadeeswaran P. (1986) DNA sequence analysis: a procedure to find homologies among many sequences. *Nucleic Acids Res.* **14**:543-550

Krogh A, Brown M, Mian IS, Sjoelander K, Haussler D. (1994a) Hidden Markov model in computational biology. Applications to protein modelling. *J. Mol. Biol.* **235**:1501-1531

Krogh A, Mian IS, Haussler D. (1994b) A Hidden Markov Model that finds genes in *E. coli* DNA. *Nucleic Acids Res.* **22**:4768-4778

Krogh A, Mitchison G (1995) Maximum Entropy Weighting of Aligned Sequences of Proteins or DNA. In *ISMB-95; Proceedings Third International Conference on Intelligent Systems for Molecular Biology*, 215-221, AAAI Press, Menlo Park.

Kruskal JB (1983) An overview of sequence comparison. In *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison, Sankoff D., Kruskal J.B., Eds.,* 1-44, Addison-Wesley, Reading.

Kulp D, Haussler D, Reese MG, Eeckman FH (1996) A Generalized Hidden markov Model for the Recognition of Human Genes in DNA. In *ISMB-96; Proceedings Fourth International Conference on Intelligent Systems for Molecular Biology*, AAAI Press, Menlo Park.

Kuzio J. (1996) Genomic sequence presentation. *Trends Genet.* **12**:321-322

Landes C, Henaut A, Risler J-L. (1993) Dot-plot comparisons by multivariate analysis (DOCMA): a tool for classifying protein sequences. *Comput. Appl. Biosci.* **9**:191-196

Lapthorn AJ, Harris DC, Littlejohn A, Lustbader JW, Canfield RE, Machin KJ, Morgan FJ, Isaacs NW. (1994) Crystal structure of human chorionic gonadotropin. *Nature* **369**:455-461

LaVolpe A, Ciaramella M, Bazzicalupo P. (1988) Structure, evolution and properties of a novel repetitive DNA family in Caenorhabditis elegans. *Nucleic Acids Res.* **16**:8213-8231

Lefevre C, Ikeda J-E. (1994) A fast word search algorithm for the representation of sequence similarity in genomic DNA. *Nucleic Acids Res.* **22**:404-411

Little E, Bork P, Doolittle RF. (1994) Tracing the Spread of Fibronectin Type III Domains in Bacterial Glycohydrolases. *J. Mol. Evol.* **39**:631-643

Lupas A, van Dyke M, Stock J. (1991) Predicting coiled coils from protein sequences. *Science* **252**:1162-1164

Maizel JVJr, Lenk RP. (1981) Enhanced graphic matrix analysis of nucleic acid and amino acid sequences. *Proc. Natl. Acad. Sci. U.S.A.* **78**:7665-7669

Marshall E. (1995) Emphasis Turns From Mapping to Large-Scale Sequencing. *Science* **268**:268-269

McClure MA, Vasi TK, Fitch WM. (1994) Comparative Analysis of Multiple Protein-Sequence Alignment Methods. *Mol. Biol. Evol.* **11**:571-592

McCombie WR, Adams MD, Kelley JM, FitzGerald MG, Utterback TR, Khan M, Dubnick M, Kerlavage AR, Venter JC, Fields C. (1992) Caenorhabditis elegans expressed sequence tags identify gene families and potential disease gene homologues. *Nat. Genet.* **1**:124-131

McDonald NQ, Lapatto R, Murray-Rust J, Gunning J, Wlodawer A, Blundell TL. (1991) New protein fold revealed by a 2.3-A resolution crystal structure of nerve growth factor. *Nature* **354**:411-414

McLachlan AD. (1983) Analysis of Gene Duplication Repeats in the Myosin Rod. *J. Mol. Biol.* **169**:15-30

McLachlan AD. (1971) Test for comparing related amino acid sequences. Cytochrome c and cytochrome c551. *J. Mol. Biol.* **61**:409-424

McLachlan AD, Boswell DR. (1985) Confidence limits for homology in protein or gene sequences. The c-myc oncogene and Adenovirus E1A protein. *J. Mol. Biol.* **185**:39-49

Medigue C, Moszer I, Viari A, Danchin A. (1995) Analysis of a *Bacillus subtilis* genome fragment using a cooperative computer system prototype. *Gene* **165**:GC37-51

Murakami Y, Naitou M, Hagiwara H, Shibata T, Ozawa M, Sasanuma SI, Sasanuma M, Tsuchiya Y, Soeda E, Yokoyama K, Yamazaki M, Tashiro H, Eki T. (1995) Analysis of the nucleotide sequence of chromosome VI from Saccharomyces cerevisiae. *Nat. Genet.* **10**:261-268

Murvai J, Gabrielian A, Fabian P, Hatsagi Z, Degtyarenko K, Hegyi H, Pongor S. (1996) The SBASE protein domain library, Release 4.0: a collection of annotated protein sequence segments. *Nucleic Acids Res.* **24**:210-214

Murzin AG, Brenner SE, Hubbard T, Chothia C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* **247**:536-540

Mushegian AR, Koonin EV. (1996) A minimal gene set for cellular life derived by comparison of complete bacterial genomes. *Proc. Natl. Acac. Sci. U.S.A.* **in press**

Naclerio G, Cangiano G, Coulson A, Levitt A, Ruvolo V, La Volpe A. (1992) Molecular and Genomic Organization of Clusters of Repetitive DNA Sequences in Caenorhabditis elegans. *J. Mol. Biol.* **226**:159-168

Nedde DN, Ward MO. (1993) Visualizing relationships between nucleic acid sequences using correlation images. *Comput. Appl. Biosci.* **9**:331-335

Needleman SB, Wunsch CD. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.* **48**:443-453

Neuwald AF, Green P. (1994) Detecting Patterns in Protein Sequences. *J. Mol. Biol.* **239**:698-712

Oefner C, D'Arcy A, Winkler FK, Eggimann B, Hosang M. (1992) Crystal structure of human platelet-derived growth factor BB. *EMBO J.* **11**:3921-3926

Oliver S. (1996) A network approach to the systematic analysis of yeast function. *Trends Genet.* **12**:241-242

Oliver SG, Van der Aart QJM, Agostini-Carbone ML, Aigle M, Alberghina L, Alexandraki D, Antoine G, Anwar R, Ballesta JPG, Benit P, Berben G, Bergantino E, Biteau N, Bolle PA, Bolotin-Fukuhara M,

Brown A, Brown AJP, et al. (1992) The complete DNA sequence of yeast chromosome III. *Nature* **357**:38-46

Olsen M. (1995) A Time to Sequence. *Science* **270**:394-396

Orengo CA, Jones DT, Thornton JM. (1994) Protein superfamilies and domain superfolds. *Nature* **372**:631-634

Ouzounis C, Casari G, Valencia A, Sander C. (1996) Novelties from the complete genome of *Mycoplasma genitalium*. *Mol. Microbiol.* **20**:895-900

Overington JP. (1992) Comparison of three-dimensional structures of homologous proteins. *Curr. Opin. Struct. Biol.* **2**:394-401

Parry-Smith DJ, Attwood TK. (1991) SOMAP: a novel interactive approach to multiple protein sequences alignment. *Comput. Appl. Biosci.* **7**:233-235

Pearson WR. (1990) Rapid and sensitive sequence comparison with FASTP and FASTA. *Meth. Enzymol.* **183**:63-98

Pearson WR, Lipman DJ. (1988) Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. U.S.A.* **85**:2444-2448

Pietrokovski S, Henikoff JG, Henikoff S. (1996) The Blocks database-a system for protein classification. *Nucleic Acids Res.* **24**:197-201

Plasterk R. (1995) Reverse genetics: from gene sequence to mutant worm. *Methods. Cell. Biol.* **48**:59-80

Plasterk R. (1992) Reverse genetics of *Caenorhabditis elegans*. *Bioessays* **14**:629-33

Posfai J, Szaraz Z, Roberts RJ. (1994) VISA: Visual Sequence Analysis for the comparison of multiple amino acid sequences. *Comput. Appl. Biosci.* **10**:537-544

Prestridge DS. (1995) Predicting Pol II promoter sequences using transcription factor binding sites. *J. Mol. Biol.* **249**:923-32

Pustell JM, Kafatos FC. (1982) A high speed, high capacity homology matrix: zooming through SV40 and Polyoma. *Nucleic Acids Res.* **10**:4765-4782

Rechid R, Vingron M, Argos P. (1989) A new interactive protein sequence alignment program and comparison of its results with widely used algorithms. *Comput. Appl. Biosci.* **5**:107-113

Reich JG, Meiske W. (1987) A simple statistical significance test of window scores in large dot matrices obtained from protein or nucleic acid sequences. *Comput. Appl. Biosci.* **3**:25-30

Reisner AH, Bucholtz CA. (1988) The use of various properties of amino acids in color and monochrome dot-matrix analyses for protein homologies. *Comput. Appl. Biosci.* **4**:395-402

Rice P, Lopez R, Doelz R, Leunissen J. (1995) EGCG 8.0. *Embnet News* **2**:5-7

Rodriguez-Tome P, Stoehr PJ, Cameron GN, Flores TP. (1996) The European Bioinformatics Institute (EBI) databases. *Nucleic Acids Res.* **24**:6-12

Romesburg HC (1989) In *Cluster Analysis for Researchers*, Krieger, Melbourne, Florida.

Rubin G. (1996) Around the Genomes: The *Drosophila* Genome Project. *Genome Res.* **6**:71-79

Russell RB, Barton GJ. (1992) Multiple protein sequence alignment from tertiary structure comparison: assignment of global and residue confidence levels. *Proteins* **14**:309-323

Sakakibara Y, Brown M, Hughey R, Mian IS, Sjoelander K, Underwood R, Haussler D. (1994) Stochastic context-free grammars for tRNA modelling. *Nucleic Acids Res.* **22**:5112-5120

Sali A, Blundell TL. (1990) Definition of general topological equivalence in protein structures. A procedure involving comparison of properties and relationship through simulated annealing and dynamic programming. *J. Mol. Biol.* **212**:403-428

Sander C, Schneider R. (1991) Database of homology-derived protein structures and the structural meaning of sequence alignment. *Proteins* **9**:56-68

Saraste M, Sibbald PR, Wittinghofer A. (1990) The P-loop - a common motif in ATP- and GTP-binding proteins. *Trends Biochem. Sci.* **15**:430-434

Sayle R, Milner White EJ. (1995) RASMOL: biomolecular graphics for all. *Trends Biochem. Sci.* **20**:374-376

Scharf M, Schneider R, Casari G *et al*. (1994) GeneQuiz: a workbench for sequence analysis. In *ISMB-94; Proceedings Second International Conference on Intelligent Systems for Molecular Biology,* 348-353, AAAI Press, Menlo Park.

Schlunegger MP, Gruetter MG. (1993) Refined crystal structure of human transforming growth factor beta 2 at 1.95 Å resolution. *J. Mol. Biol.* **231**:445-458

Schneider R, Sander C. (1996) The HSSP database of protein structure-sequence alignments. *Nucleic Acids Res.* **24**:201-205

Schuler GD, Altschul SF, Lipman DJ. (1991) A workbench for multiple alignment construction and analysis. *Proteins* **9**:180-191

Schwartz S, Miller W, Yang C-M, Hardison RC. (1991) Software tools for analyzing pairwise alignments of long sequences. *Nucleic Acids Res.* **19**:4663-4667

Seto Y, Ikeuchi Y, Kanehisa M. (1990) Fragment peptide library for classification and functional prediction of proteins. *Proteins* **8**:341-351

Sibbald PR, Argos P. (1990) Weighting aligned protein and nucleic acid sequences to correct for unequal representation. *J. Mol. Biol.* **216**:813-818

Smith DR, Lee HM, Dubois J, Qui D, Caubet W, Bashirzadeh R, Parenteau P, Weirzbowski JWX, Shimer S, Nolling J, Reeve J. (1995) Microbial genome sequencing. *Gen. Sci. Technol.* **1**:P-48

Smith HO, Annau TM, Chandrasegaran S. (1990) Finding sequence motifs in groups of functionally related proteins. *Proc. Natl. Acad. Sci. U.S.A.* **87**:826-830

Smith RF, Smith TF. (1992) Pattern-induced multi-sequence alignment (PIMA) algorithm emplying secondary structure-dependent gap penalties for use in comparative protein modelling. *Protein Eng.* **5**:35-41

Smith RF, Smith TS. (1990) Automatic generation of primary sequence patterns from sets of related protein sequences. *Proc. Natl. Acad. Sci. U.S.A.* **87**:118-122

Smith SW, Overbeek R, Woese CR, Gilbert W, Gillevet PM. (1994) The genetic data environment an expandable GUI for multiple sequence analysis. *Comput. Appl. Biosci.* **10**:671-675

Smith TF, Waterman MS. (1981) Identification of common molecular subsequences. *J. Mol. Biol.* **147**:195-197

Snyder EE, Stormo GD. (1993) Identification of coding regions in genomic DNA sequences: an application of dynamic programming and neural networks. *Nucleic Acids Res.* **21**:607-613

Snyder EE, Stormo GD. (1995) Identification of protein coding regions in genomic DNA. *J. Mol. Biol.* **248** :1-18

Sonnhammer ELL, Durbin R. (1996) A dot-matrix program with dynamic threshold control suited for genomic DNA and protein sequence analysis. *Gene* **167**:GC1-10

Sonnhammer ELL, Durbin R (1994a) An expert system for processing sequence homology data. In *ISMB-94; Proceedings Second International Conference on Intelligent Systems for Molecular Biology*, 363-368, AAAI Press, Menlo Park.

Sonnhammer ELL, Durbin R. (1994b) A workbench for large-scale sequence homology analysis. *Comput. Appl. Biosci.* **10**:301-307

Sonnhammer ELL, Eddy SR, Durbin R. (1996) Pfam: a Comprehensive Database of Protein Domain Families Based on Seed Alignments. *Proteins* **in press**

Sonnhammer ELL, Kahn D. (1994) Modular arrangement of proteins as inferred from analysis of homology. *Protein Sci.* **3**:482-492

Spradling AC, Stern DM, Kiss I, Roote J, Laverty T, Rubin GM. (1995) Gene disruptions using P transposable elements: an integral component of the *Drosophila* genome project. *Proc. Natl. Acad. Sci. U. S. A.* **92**:10824-10830

Staden R. (1990) Finding protein coding regions in genomic sequences. *Meth. Enzymol.* **183**:163-180

Staden R. (1982) An interactive graphics program for comparing and aligning nucleic acid and amino acid sequences. *Nucleic Acids Res.* **10**:2951-2961

Staden R, Dear S. (1992) Indexing the sequence libraries: software providing a common indexing system for all the standard sequence libraries. *DNA Seq.* **3**:99-105

States DJ, Harris NL, Hunter L. (1993) Computationally efficient cluster representation in molecular megaclassification. In *ISMB-93; Proceedings First International Conference on Intelligent Systems for Molecular Biology*, 387-394, AAAI Press, Menlo Park.

Sulston J, Du Z, Thomas K, Wilson R, Hillier L, Staden R, Halloran N, Green P, Thierry-Mieg J, Qiu L, Dear S, Coulson A, Craxton M, Durbin RK, Berks M, Metzstein M, Hawkins T, Ainscough R, Waterston R. (1992) The *C. elegans* genome sequencing project: a beginning. *Nature* **356**:37-41

Sutcliffe MJ, Haneef I, Carney D, Blundell TL. (1987) Knowledge based modelling of homologous proteins, Part I: Three-dimensional frameworks derived from the simultaneous superposition of multiple structures. *Protein. Eng.* **1**:377-384

Swindells, M.B. (1992) Structural similarity between transforming growth factor-beta 2 and nerve growth factor. *Science* **258**:1160-1162

Tatusov RL, Altschul SF, Koonin EV. (1994) Detection of conserved segments in proteins: iterative scanning. *Proc. Natl. Acad. Sci. U.S.A.* **91**:12091-12095

Tatusov RL, Mushegian AR, Bork P, Brown N, Hayes WS, Borodovsky M, Rudd KE, Koonin EV. (1996) Metabolism and evolution of *Haemophilus influenzae* deduced from a whole-genome comparison with *Escherichia coli*. *Curr. Biol.* **6**:279-291

Taylor WR. (1988) A flexible method to align large numbers of biological sequences. *J. Mol. Evol.* **28**:161-169

Taylor WR. (1990) Hierarchical method to align large numbers of biological sequences. *Meth. Enzymol.* **183**:456-474

Thompson JD, Higgins DG, Gibson TJ. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.* **22**:4673-4680

Troemel ER, Chou JH, Dwyer ND, Colbert HA, Bargmann CI. (1995) Divergent seven transmembrane receptors are candidate chemosensory receptors in C. elegans. *Cell* **83**:207-218

Uberbacher EC, Mural RJ. (1991) Locating protein-coding regions in human DNA sequences by a multiple sensor-neural network approach. *Proc. Natl. Acad. Sci. U.S.A.* **88**:11261-11265

Venter JC, Smith HO, Hood L. (1996) A new strategy for genome sequencing. *Nature* **381**:364-366

Vingron M, Waterman MS. (1994) Sequence alignment and penalty choice. Reviews of concepts, case studies and implications. *J. Mol. Biol.* **235**:1-12

von Heijne G. (1987) In *Sequence analysis in molecular biology. Treasure trove or trivial pursuit,* Academic Press, London.

Wahl R, Rice P, Rice CM, Kröger M. (1994) ECD - a totally integrated database of *Escherichia coli*. *Nucleic Acids Res.* **22**:3450-3455

Watanabe H, Otsuka J. (1995) A comprehensive repesentation of extensive similarity linkage between large numbers of proteins. *Comput. Appl. Biosci.* **11**:159-166

Waterman MS, Eggert M. (1987) A new algorithm for best subsequence alignments with application to tRNA-rRNA comparisons. *J. Mol. Biol.* **197**:723-728

Waterman MS, Jones R. (1990) Consensus methods for DNA and protein sequence alignment. *Meth. Enzymol.* **183**:221-237

Waterston R, Martin C, Craxton M, Huynh C, Coulson A, Hillier L, Durbin R, Green P, Shownkeen R, Halloran N, et al. (1992) A survey of expressed genes in Caenorhabditis elegans. *Nat. Genet.* **1**:114-123

Waterston R, Sulston J. (1995) The genome of Caenorhabditis elegans. *Proc. Natl. Acad. Sci. U. S. A.* **92** :10836-10840

Weiss R. (1996) *Human Genome News* **7**:13

White O, Fitzhugh W. (1995) A rapid retrieval tool for operating on large flat archive files. *Comput. Appl. Biosci.* **11**:45-47

Wilson R, Ainscough R, Anderson K, Baynes C, Berks M, Bonfield J, Burton J, Connell M, Copsey T, Cooper J, Coulson A, Craxton M, Dear S, Du Z, Durbin R, Favello A, Fulton L, Gardner A, Green P, Hawkins T, Hillier L, Jier M, Johnston L, Jones M, Kershaw J, Kirsten J, Laisster N, Latreille P, Lightning J, Lloyd C, Mortimore B, O'Callaghan M, Parsons J, Percy C, Rifken L, Roopra A, Saunders D, Shownkeen R, Sims M, Smaldon N, Smith A, Smith M, Sonnhammer E, Staden R, Sulston J, Thierry-Mieg J, Thomas K, Vaudin M, Vaughan K, Waterston R, Watson A, Weinstock L, Wilkinson-Sproat J, Wohldman P. (1994) 2.2 Mb of contiguous nucleotide sequence from chromosome III of C. elegans. *Nature* **368**:32-38

Wootton JC. (1994) Non-globular domains in protein sequences: automated segmentation using complexity measures. *Comput. Chem.* **18**:269-285

Wootton JC, Federhen S. (1993) Statistics of local complexity in amino acid sequences and sequence databases. *Comput. Chem.* **17**:149-163

Worley KC, Wiese BA, Smith RF. (1995) BEAUTY: An Enhanced BLAST-based Search Tool that Integrates Multiple Biological Information Resources into Sequence Similarity Search Results. *Genome Research* **5**:173-184

Xu Y, Mural RJ, Shah M, Uberbacher EC (1994) Recognizing exons in genomic sequence using GRAIL II. In *Genetic engineering: principles and methods, Setlow J., Ed.*, 16:241-253, Plenum Press, New York.

Zhang J, Ostell J, Rudd K. (1994) ChromoScope: a graphic interactive browser for *E. coli* data expressed in the NCBI data model. In *Proc. 27th Annual Hawaii International Conference on System Sciences* **V**:58-67

Zuker M. (1991) Suboptimal sequence alignment in molecular biology. Alignment with error analysis. *J. Mol. Biol.* **221**:403-420

Zuker M, Stiegler P. (1981) Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Res.* **9**:133-148

Zwaal RR, Broeks A, van MeursJ, Groenen JT, Plasterk R. (1993) Target-selected gene inactivation in *Caenorhabditis elegans* by using a frozen transposon insertion mutant bank. *Proc. Natl. Acad. Sci. U. S. A.* **90**:7431-7435

Appendices