Contents

1 Major sequence repositories

Databases (USA) or databanks (UK) were created to store genome and protein sequences and others related data from biological and computational analysis. Sequence annotation and comparison allow the identification of family and functional relationships among proteins. However, text mining is limited by the lack of coherence between databases and by semantics problems.

A database is structured collection of data held in computer storage, that searchable, updatable and can be cross-referenced. It is managed by a software package called Database Management Systems (DBMs).

Databases are structure explicitly (example: a sequence in GenBank is between `origin` and //) or implicitly (example: the FASTA format). They are composed of a set of tables linked by a shared information referred as the key, which makes them easily searchable. However, there are some arguments against the relational database management, as conflict rise by integrating data from distinct origins.

The major molecular biology databanks are
- Primary sequence databases: like *ENA (EMBL/GenBank)*, *SwissPro* they contain the sequences; in order to reduce redundancy, *RefSeq* was introduced as data integration system.
- Secondary databases: they reorganize the raw data to enable new prediction (domain family) like *InterPro*, or are species-specific databases like *SGC* and *Ensembl*
- Web-base interface systems: they are form-bases query and browsing interfaces like *Entrez* (on *NCBI*).

Database entries contains
- The locus, which is the identifier and the position of the gene along the given chromosome defined by genetic analysis, and the accession number which is a unique identifier for the sequence
- The sequence itself
- Additional information referred to as annotation/ref, which even if modified do not change the accession number. They can contain the date of creation of modification (DT), the source organism (OS), the description (DE), the literature references (RX), the key cross-reference words (KW), the features of the sequence (FT), etc.

For protein sequence databanks, first there are the one containing the experimentally determined and translated acido acid sequences, like *UniProt* of *Genrept* (on *NCBI*). Then there are composite number sequence databases, like *RefSeq* (*NCBI*) which provides separate and linked records for the genomic DNA, mRNA and protein arising from the same transcript. Protein structure database also exists, like the PDB or SCOP, is which the 3D structure of a protein can be studied.

1.1 Secondary databases

Those databases are a collection of conserved amino-acid patterns that are derived from primary sequence databases by different methods
- Motif/pattern: short sequences (10 residues) that are found within the active site of proteins that have similar biochemical activity
- Domain/profile: combination of several secondary structural elements (50-100 residues) or conserved aligned residues

- Family/HMM

A domain is an extended sequence pattern that is showing context independency. A structural domain, however, is a segment of polypeptide chain that can fold independently of the rest of the polypeptide.

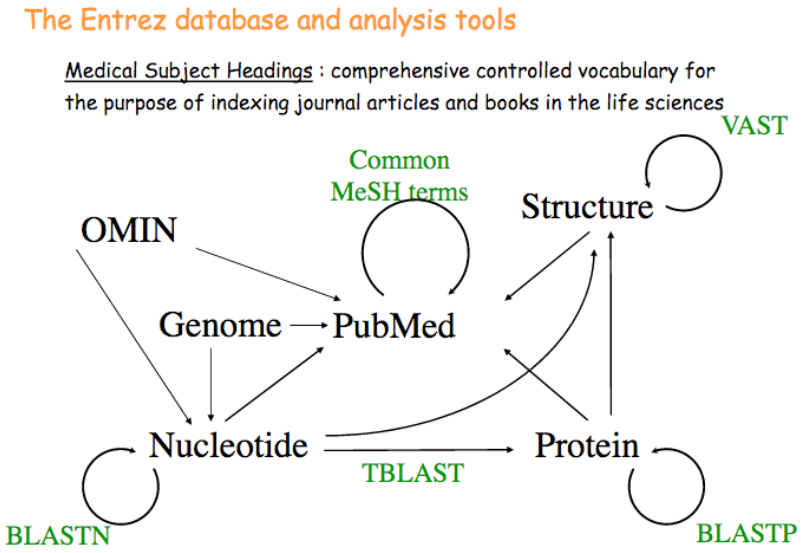
The different secondary databases are

- *Prosite*: it contains protein active sites; the motives are expressed as regular expressions or profiles. It is a great tool to for predicting the existence of actives sites in an unknown protein
- *Prints* and *Blocks*: in which several local alignemnts (fingerprints) expressed as frequency (*Prints*) pr weight (*Blocks*) matrices
- *Pfam*: contain multiple alignments including gap between the conserved regions, where profiles are describes by weight matrices and the entries are based on Hidden Markov Models.
- *Interpro*: composite databse composed of *Pfam* + *Prints* + *ProDom* + *Smart* + *Prosite*.

The *Ensembl* databases contain information about genome assembly, gene annotation, comparative genomics, regulation and variation.

1.2 Data retrieval systems

The aim of data retrieval is to fin new and relevant information. The system designed for this aim must be easy to use, fast and liked to analysis tools. *Expasy*, *Entrez* and *SRS* are such system, developed by different sites.



2 Pairwise sequence comparison

The main goal of sequence alignment is to understand the meaning of identity, similarity and homology. Sequences are compared with dot plots and through pairwise alignment. At the end of this chapter, one must be able to explain substitution score matrices, the concepts of optimal alignment and dynamic programming as well as the differences between affine and linear gap functions. One must also understand the differences between local and global alignment.

Some definitions

- Identity: the proportions of pairs of identical residues between two aligned sequences (in %), depending on the alignment

- Similarity: proportion of pairs of similar residues between two aligned sequences. Similar residues can be substituted for one another without modifying the function, depending on the used substitution matrix. Similarity thus reflects amino-acid substitutions among homologous proteins and therefore equivalent physico-chemical properties

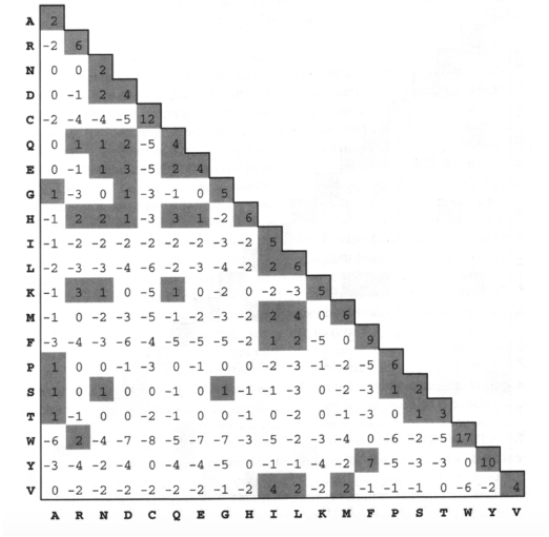


Figure 1: Substitution matrix: if the value is positive, the substitution occurs frequently; if the value is 0, the substitution does not occur more than by chance

The choice of scoring matrix affects the similarity determination. Also, gaps are considered as additional residues. The penalty values of adding gaps are linked to the scoring matrix.

- Distance: number of observed changed in an optimal alignment of two sequences, usually not counting gaps (phylogeny), depending on the sequence alignment method.

  If the score is included between 0 and 1, the simplest model states that $D = 1 - S$. One can also define the p distance as following

$$P = D/L \quad \begin{array}{rcl} D & = & \text{number of position at which two sequences differ} \\ L & = & \text{length of each of the two sequences} \end{array}$$

- Homology: focuses on common ancestry and shows how closely related species have changed from their ancient ancestors; in contrast, analogous structures are not necessarily evidence that two species came from a common ancestor.

Sequence, function and 3D fold structure should not be equally considered indicators of homology. Indeed, homologous sequences do not necessarily serves the same function and 3D structures may be conserved while the sequence is not.

Genes may share ancestry because of speciation (orthologs), meaning one ancestor gives separate species, or because of duplication (paralogs), meaning the ancestor duplicates and diverges. This distinction becomes complex when genes disappear.

2.1 Dot Plot

2.1.1 Concept

The concept of a dot plot is the comparison of the horizontal and vertical axes, which correspond respectively to two sequences. A dot is placed at each position where two residues match. A region of similarity stands out as a diagonal.

In order to reduce the noise and filter out random matches, a widow approach is used to calculate the score, with respect to a threshold. Each window of adjacent position of the first sequence is aligned, without gaps, to each window for the second second. A color indicates the score of the aligned windows.

2.1.2 Utility

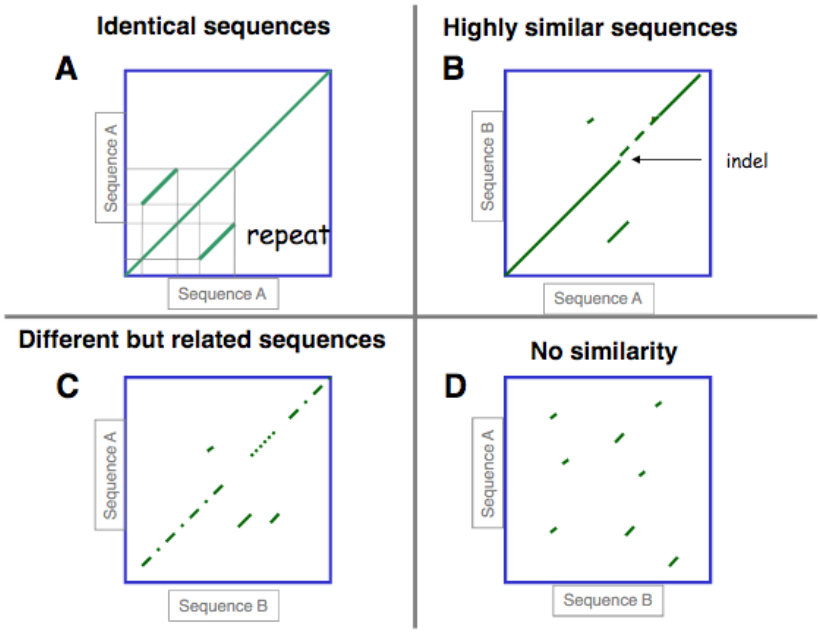


Figure 2: Dot plot utility

Dot matrix analysis can also be used to find direct and inverted repeats within sequences. If the repeat is in the same orientation, it is called a direct repeat and is a protein domain or motif. If the repeat is in the reverse orientation, it is an inverted repeat or palindrome and characterizes self-complementary sequences in mRNA.

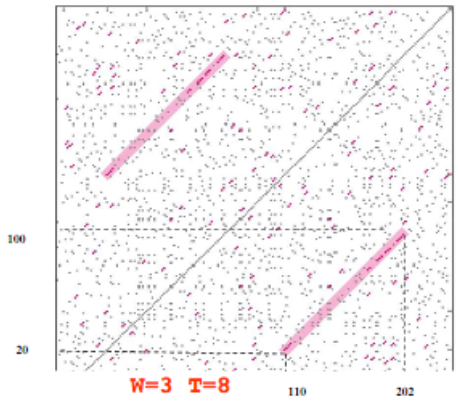


Figure 3: Dot plot with repeats: the TATA-bos binding protein TBP

Dot plot are also used to compare genomic and cDNA copies. This means one sequences is compared with the complementary of the other gene.

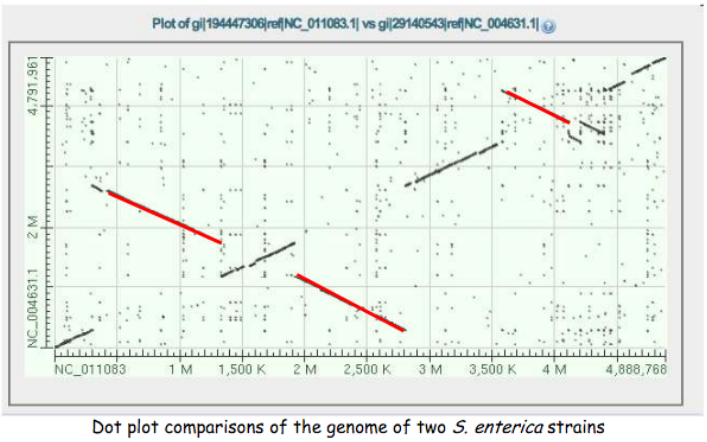


Figure 4: Dot plot for genome comparison

2.1.3 Limitations

Dot plots are a visual aid. It is a good way to explore sequence organization, but it does not provide alignment.

2.2 Pairwise alignment

2.2.1 Concept

Pairwise alignment is base of the explicit residue matching between sequences. In this way, substitutions or mismatches can be identified, but also insertions.

2.2.2 Alignment evaluation

Different alignments are possible between two sequences. Therefore, one needs a way to evaluate the biological meaning of an alignment. This must includes a tolerance to errors (mismatches, insertions/deletions, also called indels) and must evaluate the significance of the alignment is a biological concept.

Scoring system The best alignment can be found by using a scoring matrix and looking for the highest score. A simple way to score an alignment is to count 1 for each match and 0 for each mismatch. This is called the identity score.

Gaps Gaps are insertion or deletions of residues, which occur through evolution. There are constraints on where these indles can happen. The alignment can often be improved by inserting a gap.

Gap opening and extension penalties: Homologous sequences may have different lengths. Therefore, a model that simulates the evolutionary mechanisms involved in gap occurrence is needed. In some cases, one may prefer one large gap to several small ones. To model this, a gap opening penalty GOP (a) is defined and counts each time a gap is opened in an alignment. Next, a gap extension penalty GEP (b) counts each extensions of a gap. The gap cost is calculated with $w_k = a + bk$ where $b < a$ and k is the residue number. The similarity score can be defined as

$$S = x - \sum w_k z_k$$

where x is the number of match and z_k the number of gaps with length k. The GOP and GEP depend on the scoring matrix, and will affect the significance score. The obtained significance score must be compared with randomly generated sequences with the same components.

2.2.3 Use of scoring matrix for sequence alignment significance

Nucleotide scoring matrices are based on Unitary matrices (1,0), Bltast matrices (5,-4) or trasversion/transition matrices (1, -5, -1) The scoring system scale are arbitrary.

Protein scoring matrices give a score for substitution of one amino-acid by another, based on the genetic code (GCM), physico-chemical properties (hydrophobicity, acid/base, ...) or evolutionary relationships (PAM and BLOSUM).

Those scoring matrices are based on probabilistic. Differences in protein sequences might results from distinct mechanisms, namely a random or non random (evolutionary) change model. Determining occurrence probability for each model allows the identification of the most likely mechanism

$$S_{ij} = \log\left(\frac{\text{observed}}{\text{expected by chance}}\right)$$

Random system: there a no constraints on amino-acid composition. Their nature at each position is independent from other positions, which means the probability depends on amino-acid background frequency in the population and $p_i = f_i$. It a residu i is aligned with a residue j, then $p_i \cdot p_j$ is the probability of the pair occurrence.

Non random (evolutionary) system: there are some constraint on amino-acid composition. The probability of occurrence of a residue is determined by the residue at the same position in the

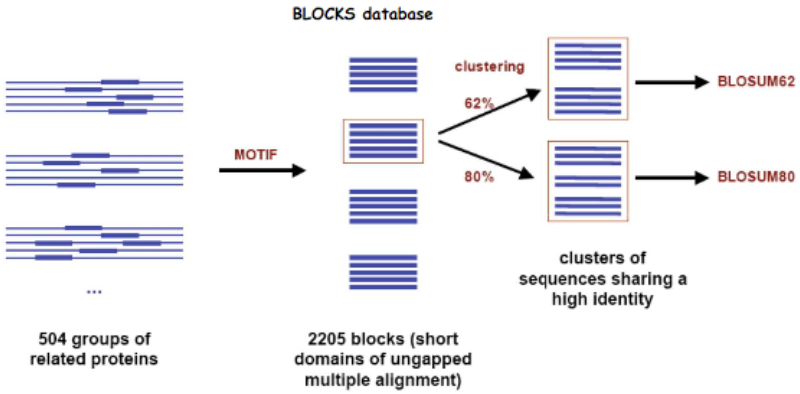
ancestral sequence. The probability of mutual substitution $q_{i,j}$ depends on the evolutionary mechanism.

Odd ratios $q_{i,j}/(p_i p_j)$ show which model is more likely. if the ratio is < 1, the non-random model is more likely to explain the alignment between residues.

Commonly used scoring matrices are based on likelihood of changes in homologous protein sequences during evolution and substitutions in conserved blocks from protein families.

The lod-odd score is defined as $S_{ij} = \log\left(\frac{q_{i,j}}{p_i p_j}\right)$.

BLOSUM matrix BLOSUM matrices are not based on the Dayhoff model of evolutionary rate.



The derivation steps are

- Calculation of the frequency of occurrence: $q_{i,j} = \frac{f_{i,j}}{\sum_{1 \le b \le a} f_{a,b}}$
- Determination of expected frequency of i being a pair p_i: $p_i = q_{i,i} + 1/2 \sum_{b \neq i} q_{i,b}$
- Calculation of odd-ratio: $R_{i,j} = \frac{q_{i,j}}{p_i p_j}$
- Calculation of the score: $S_{i,j} = 2 \log_2 R_{i,j}$

Derivation of PAM matrices It is based on the analysis of point or percentage accepted mutations in homologous sequences. It determines the probability of a residue being substituting or not.

First, the derivation starts of with a sequence alignment and the related phylogenetic tree.

The next steps to follow are

- Computation of the relative mutability, which is the probability of amino-acid in given small evolutionary interval, meaning the number of times the amino-acid has changed divided by the total exposure to mutation.

$$m_j = \frac{A_i}{N_j / N_{tot} \cdot A_t ot \cdot 100}$$

- Calculation of the mutabilition probability: $M_{ij} = \frac{m_j A_{ij}}{\sum_i A_{ij}}$

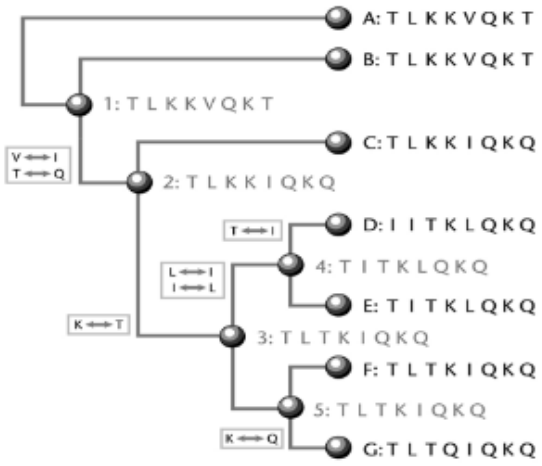


Figure 5:

- Derivation of the log relatedness odds: $R_{ij} = \frac{q_{i,j}}{p_i p_j} = \frac{M_{ij}}{f_i}$ where f_i is the frequency of occurrence. The score is $S_{ij} = \log R_{ij}$.

In a PEM scoring matrix, a positive score means the aa are similar, mutations from one into the other occur more often than expected by chance. For a negative score, it is the reverse, aa are dissimilar.

2.3 Concluding remarks

Substitution matrices and gap penalties introduce biological information into the alignment algorithms. It is not because two sequences can be aligned that they share a common biological history. The relevance of the alignment must be assessed with a statistical score. There are many ways to align two sequences. Do not blindly trust your alignment to be the only truth. Especially gapped regions may be quite variable.

3 Sequence statistics

3.1 Motivations

There are the objectives to sequence statistics. The first one is to analyze real biological data to help make sense out of it. The second one is to use appropriate algorithms and statistical methods to go beyond what can be done manually. The vast amount of available data can lead to novel insights. Automating things avoid wasting time on repetitive tasks. Also, statistics help to discover hidden patterns. finally, machine learning helps to predict on new data from past observations.

The challenges are different for everyone

- Computer scientist/statistician: avoid to apply/design algorithms blindly without considering the underlying biology
- Molecular biologist: avoid to consider algorithms or software as black boxes and go beyond the "click on the WEB" methodology
- Biomedical engineer: reconcile both worlds

The final validation must come from understanding the biology and from a proper field assessment. New algorithms need to be designed to address biological questions. Further research in machine learning and bio-statistics is required.

3.2 Simple genome statistics

One of the most fundamental properties of a genome sequence is its base composition, the proportion of A, G, C and T nucleotides present. To illustrate this, the *Haemophilus influenza* bacteria was sequences. The base composition and relative base frequencies (on one strand of DNA) are gathered Table 2.

	A	C	G	T
Base composition	567623	350723	347436	564241
Base frequencies	31.02%	19.16%	18.98%	30.83%

Table 1: Base composition and base frequencies of the *Haemophilus influenza* bacteria

However, the sum of a bases is not equal to the total length of the sequence, which is 1830138 bp. This is due to the fact that some position can be occupied by two or all bases. For example, there are 14 K bases, K standing for a G or T base. In the same way, M = 1 or C, N stands for any base, R = A or G, S = G or C, W = A or T and W = C or T. The relatives base frequencies are simply the base composition divided by the total length of the sequence.

Table 2 shows that the four nucleotides are not used at equal frequency across the genome: A and T are much more common than G and C. In fact, it is fairly unusual for all of the bases to be used in equal frequencies in any genome.

In addition to the global base composition of a genome, it is of interest to consider local fluctuations in the frequencies of nucleotides across the sequence. The local base composition can be measured by sliding a window of size k along a chromosome, measuring the frequency of each base in the window, and assigning these values to the central position of the window. This produces a vector of length $L - k + 1$ that can be plotted.

3.2.1 Change point analysis

The analysis of the base frequencies is Table 2 is however overly complex for most biological studies. Therefore, most often the the aggregate frequencies for C and G (called *GC content*) versus the aggregate frequencies for A and T (*AT content*) is used, and, as their sum equals 1, most often only the *GC* content a is reported.

A simple analysis of GC content can reveal significant biological information. For example, in bacteria these frequencies are highly dependent on an organism's replication machinery, and can be very different from species to species. Because of this, an analysis of GC content can be used to detect foreign genetic material that has been inserted into a genome by identifying locations where this content is different from the genomic average.

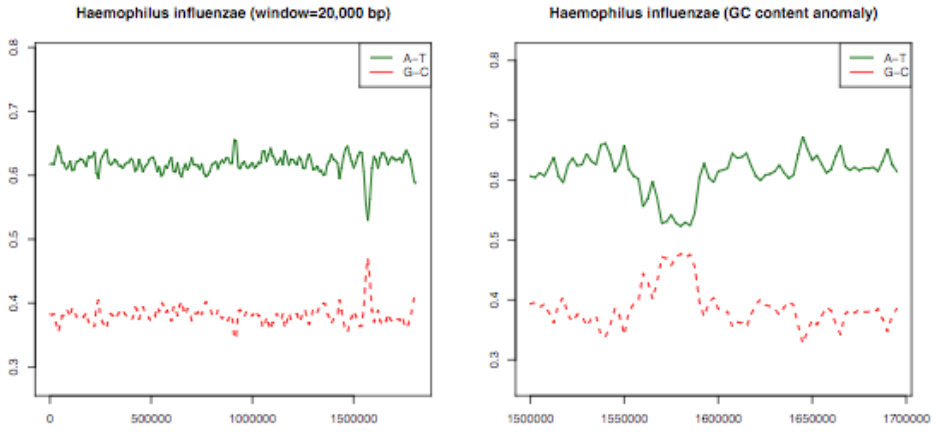


Figure 6: Sliding window plot showing local fluctuations in the frequencies of nucleotides across the sequence of *H. influenzae*

Figure 6 shows an unusual GC content 1.56 Mb into the genome (positions in circular chromosomes are generally given relative to the site where DNA replication starts during cell division). This anomaly is attributed to an ancient insertion of viral DNA into the *H. influenzae* genome.

An interesting difference between AT- and GC-rich regions is the energy needed to separate (denature) the two DNA strands: AT-rich regions separate at lower temperatures. It is believed that the ability to quickly denature DNA facilitates the insertion in the bacterial cell being infected.

A method to identify locations in the sequence where statistical properties, such as GC content, change is desired. These locations, called change points, divide the genome into regions of approximately uniform statistical behavior, and can help to identify important biological signals.

Change point analysis can be performed in various ways; one particularly effective approach based on hidden Markov models will be discussed later. The most simple-minded strategy for finding regions of different statistical behavior involves setting a threshold value that distinguishes two such regions. If this threshold between two windows is crossed, a change point has been identified. Of course setting this threshold is a statistical problem, as is the size of the window used. Both have to do with the probability of finding variation in random data, a question of hypothesis testing.

3.2.2 Finding unexpected k-mers

Another simple and important property of a genome to be measured is the frequency of all nucleotide words of length 2 (dimers), or higher (trimers, k-mers). To illustrate this let us look at the dimer frequencies of *H. influenzae*, gathered in Table **??**.

	*A	*C	*G	*T
A*	0.1202	0.0505	0.0483	0.0912
C*	0.0665	0.0372	0.0396	0.0484
G*	0.0514	0.0522	0.0363	0.0499
T*	0.0721	0.0518	0.0656	0.1189

Table 2: Dimer frequencies of the *Haemophilus influenza* bacteria

Equlatty likely dimers would appear $1/16 = 0\ 0625$ of the time. This means that AA and TT look particularly frequent, while CC, CG and GG look particularly rare. However, one has to consider the fact that A and T alone are also more frequent than G and C.

Therefore, the expected frequency of dimers observed by chance as to be calculated. This however needs a background model to characterize the non-uniform distribution of individual nucleotides. The is achieved with the *Multinomial background model*. In this model, the first step is to estimate the probability of each nucleotide independently.

$$\hat{P}(A) = f(A) = \frac{\text{number of A's}}{\text{sequence length}} = 0\ 3102$$

Those estimates are then used it a random generative model. The expected frequency of simer XY is expressed as

$$E[f(XY)] = \hat{P}(X)\hat{P}(Y) = f(X)f(Y)$$

To see if a certain dimer is unusual, the odd ratio can be used. This is the ratio between observed frequency and expected frequency

$$\frac{f(XY)}{E[f(XY)]} = \frac{f(XY)}{f(X)f(Y)}$$

It indicated how much the frequency of an observed dimer diverges from the random generative model. The odd ratios for the *H. influenzae* bacteria are gathered in Table 3.

	*A	*C	*G	*T
A*	1.2490	0.8495	0.8209	0.9533
C*	1.1180	1.0119	1.0892	0.8189
G*	0.8725	1.4348	1.0074	0.8525
T*	0.7540	0.8762	1.1202	1.2504

Table 3: Odd ratios of the *Haemophilus influenza* bacteria

Any significant deviation from 1 indicates some effect of either the mutational process or of natural selection. In Table 3 the most over-represented dimer is GC. The dimer TA is rare, but this is quite universal. The significance of the deviation from 1 should be analyzed through a statistical test.

The expected frequency with the multinomial model can be replaced by the observed frequency $f_{random}(XY)$ in a random sequence generated from this model equivalent to a random permutation of the original sequence.

All this can be generalized to k-mers. The odd ratio is expressed as

$$\frac{f(X_1 \dots X_k)}{E[f(X_1 \dots X_k)]} = \frac{f(X_1 \dots X_k)}{\prod_{i=1}^{k} f(X_i)}$$

This is useful when looking for frequent patterns of k consecutive characters. To determine if those k-mers are informative, one needs a more sophisticated background model, a statistical test to assess the significance and a biological validation.

3.3 Identification of Open reading Frames

3.3.1 Brief review of the underlying biology

While finding complex eukaryotic genes requires commensurately complex methods, finding prokaryotic genes turns out to be a relatively simple task. In order to understand how a cell finds genes, one has to first look at the production of proteins. The simplest way to explain how proteins are made – ignoring many details and condensing complicated cellular reactions – is a simple diagram that shows what is known as the central dogma:

$$\text{DNA} \rightarrow \text{mRNA} \rightarrow \text{protein}$$

The process of going from a gene encoded in DNA to a protein made up of amino acids is divided conceptually into two steps: transcription (DNA $\rightarrow$ RNA) and translation (RNA $\rightarrow$ Protein). On the whole, a sequence of nucleotides is translated into a string of amino acids. Each amino acid is related to a codon of 3 nucleotides. This is called the genetic code and is represented in Figure 7.

3.3.2 ORF finding algorithm

Even with the genetic code in hand, there is still a major problem to be overcome by the translation machinery. The translational machinery must know exactly where on the mRNA to start working. And depending on where one starts reading a DNA sequence, there are three different ways to decompose it into codons. The translation into amino acids, and hence the resulting protein, would be completely different in each of the three cases. Each non-overlapping decomposition of a DNA sequence into codons is called a reading frame. In total there are six reading frames: three on the forward strand and three on its reverse complement. A reading frame is specified by starting with a Start codon, represented by the ATG codon or the Met amino acid. After the first Met, other may appear before reaching the Stop codon. These are not true starts. An open reading frame (ORF) is the longest stretch of DNA between a Start and Stop codon, without being interrupted by another Stop on the same frame.

However, an ORF is not always a coding gene. The DNA stretch found between a Start and Stop codon might be due to chance. This solve this, the significance of the ORF has to be assessed through a statistical test. Also, the ORF might be there but the gene not expressed, as a trace of the past (or future) evolution. All regulations at the transcription/translation level are ignored. Finally, some gene sequences do not strictly follow the standard ORF structure. A careful biological validation is thus needed.

First position (5′ end)	Second position				Third position (3′ end)
	U	C	A	G	
U	Phe	Ser	Tyr	Cys	U
	Phe	Ser	Tyr	Cys	C
	Leu	Ser	Stop	Stop	A
	Leu	Ser	Stop	Trp	G
C	Leu	Pro	His	Arg	U
	Leu	Pro	His	Arg	C
	Leu	Pro	Gln	Arg	A
	Leu	Pro	Gln	Arg	G
A	Ile	Thr	Asn	Ser	U
	Ile	Thr	Asn	Ser	C
	Ile	Thr	Lys	Arg	A
	Met	Thr	Lys	Arg	G
G	Val	Ala	Asp	Gly	U
	Val	Ala	Asp	Gly	C
	Val	Ala	Glu	Gly	A
	Val	Ala	Glu	Gly	G

Figure 7: Standard genetic code

3.3.3 Significance assessment

The bigger the genome gets, the higher the possibility of the ORF to occur by chance. Therefore, a method must be devised to separate reliable patterns and background noise.

Measuring the probability of a ORF under a null model is a simple and effective way of doing this. Calculating this probability and making inferences based on it is a fundamental problem in statistics, and is referred to as hypothesis testing. To explain hypothesis testing, let us imagine two series of samples, respectively referred as μ_1 and μ_2, with respective means $\hat{\mu}_1$ and $\hat{\mu}_2$. Two hypotheses are made

$$\text{Null hypothesis } H_0 : \mu_2 = \mu_1$$
$$\text{Alternative hypothesis } H_a : \mu_2 \neq \mu_1$$

The test statistics $T = \hat{\mu}_2 - \hat{\mu}_1$ approximately follow a Student t distribution

$$T = \frac{(\hat{\mu}_2 - \hat{\mu}_1) - (\mu_2 - \mu_1)}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

where s_i it the sample variance and n_i the number of sample.

Next, a significance threshold α needs to be fixed, for example 5%. If $T > t_{1-\alpha/2}$, the hypothesis H_0 is to be rejected.

Instead of fixing a significance threshold, one can also report the p-value of the test. This is the smallest α that would to reject the test. When the p-value is smaller than α, the result is significant. Often a threshold value of $\alpha = 0\ 05$ is used to define significance, a popular (but arbitrary) choice. This value of α means that even if the null hypothesis is true, 5% of the time our data will appear to be significant.

For ORF finding, the significant ORFs in an actual sequence should be longer than ORFs observed by chance. Therefore, the null model is typically made of a random permutation of the original sequence. The ORF finding algorithm is thus the following one

- Finding all ORFs in a random permutation of the original sequence
- Reporting the length distribution of random ORFs
- Accepting as significant ORFs any ORF in the original sequence longer than the prescribed threshold (either the maximal ORF length of the random permutation of the 99% percentile of the random ORF length distribution)

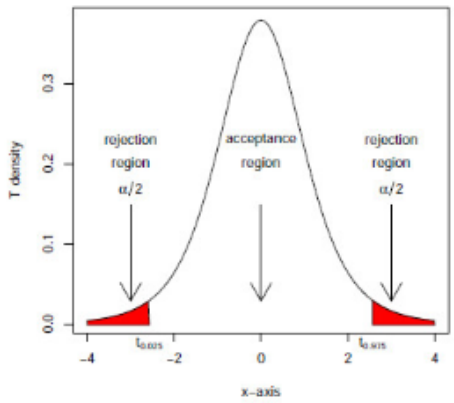


Figure 8: Parametric two-tailed t-test

4 Pairwise alignment algorithms

4.1 The alignment problem

The objective of pairwise alignment is to find the best way to aligne two sequences including matches, substitutions and possible gaps (insertions or deletions). In order to be aligned, both sequences do not need to have the same length. A scoring matrix defines the similarity scores between the resides. The higher the score, the more similar a given pair is. Also, a gap penalty is defined as a negative score. Pairwise alignment consist in finding the alignment with the maximal cumulative score.

4.1.1 Gap penalties

Linear gap penalties: $\gamma(g) = -dg$ with g the number of consecutive gaps and d the gap penalty.

Affine gap penalty: $\gamma(g) = -d - e(g-1)$ with d the gap opening penalty and e the gap extension penalty. This model is more relevant from a biological viewpoint but more complex to compute with.

4.1.2 Global versus local alignment

Global alignment: A global alignment of two sequences can be thought of as a representation of the correspondence between their respective symbols (i.e. their nucleotides). Sequence x of length n is thus alignment to sequence y of length m, considering an appropriate scoring matrix S and gap penalty $\gamma(g)$. The optimal global alignment is the one with the maximal cumulative score.

Many different alignment are possible. Let us consider Figure 9. The number of ways to describe the combinations of k, $n-k$ and $m-k$ operations is expressed as (assuming $n \approx m$

$$N = \sum_{k=0}^{min(n,m)} \frac{(n+m-k)!}{k!\,(n-k)!\,(m-k)!} \approx \frac{4^n}{\sqrt{\pi n}}$$

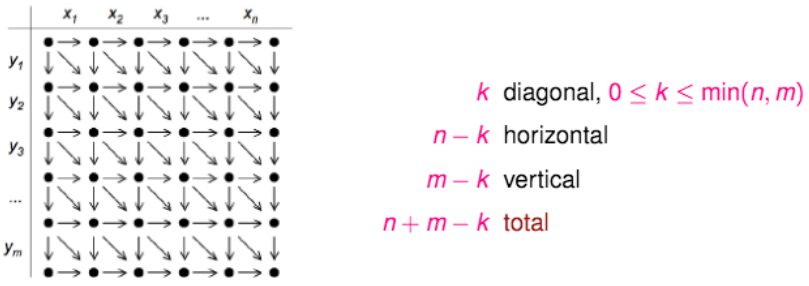


Figure 9: Number of global alignments

Local alignment: A local alignment of two sequences, s and t, is a global alignment of the subsequences $s_{i:j}$ and $t_{k:l}$, for some choice of (i,j) and (k,l). The optimal local alignment is given by the optimal choice of (i,j) and (k,l),so as to maximize the alignment score.

4.2 Global alignment: Needleman-Wunsh algorithm

The aim of this algorithm is to obtain the maximal cumulative score, meaning the highest sum of independent individual scores - $s(x_i, y_j)$ or gap penalties. The optimal alignment between sequences x and y is made of optimal alignments between sub-sequences, also called prefixes. The problem is thus decomposed into sub-problems, each needing to be computed only once and leading to an optimal sub-solution. There are three elements to the NW algorithm: a recursive relation, a tabular computation, and a trace-back procedure.

Recursive relation The idea behind the recursion in the Needleman–Wunsch algorithm is that in order to find the score of an optimal alignment between the first i symbols of sequence s and the first j symbols symbols of sequence t, all that is needed is the score of the alignment between the two sequences up to the previous position. To understand this, let us consider the first position in an alignment between two sequences. There are only three possibilities for the state of this position: (1) gap in the first sequence, (2) gap in the second sequence, and (3) no gaps, but either a match or mismatch. Given a scoring function, the score for each of these three possibilities is immediately known. Of course any one of the three options might make for a better or worse alignment of the remaining sequence, so it is not yet possible to choose which one is best. However, if the possible scores for these three alignments are known, then the optimal alignment for the sequences up to next position can chosen. This recursion can be written as

$$F(i,j) = max \begin{cases} F(i-1,j) - d & (1) \\ F(i,j-1) - d & (2) \\ F(i-1,j-1) + s(x_i, y_i) & (3) \end{cases}$$

Tabular computation The partial alignment scores are gathered in a rectangular table with $m+1 \times n+1$ cells. The first row and column are associated with the "empty sequence" and, in order to initialize the table, they are filled with multiples of the gap penalty according to the following rules:

$$F_{i,0} = \sum_{k=1}^{i} d \quad \text{and} \quad F_{0,j} = \sum_{k=1}^{j} d$$

After the initialization, the table is filled from top to bottom, and left to right, according to the recursive relation established above. The alignment score can be read in the bottom-right cell, $F(n,m)$. The alignment itself is reconstructed from the optimal path following backpointers, according to the trace-back method.

Trace-back procedure When building the table, the cell are filled using one of the three recursive rules. It possible to keep track of the chose rule by using a pointer, which shows the path used to fill the table. It is possible to trace-back the pointer starting from the right-bottom cell, up to the initial left-top cell. This path gives the optimal alignment. For each diagonal step, there is a corresponding match/mismatch. For each vertical or horizontal step, there is a gap insertion in the respective sequence.

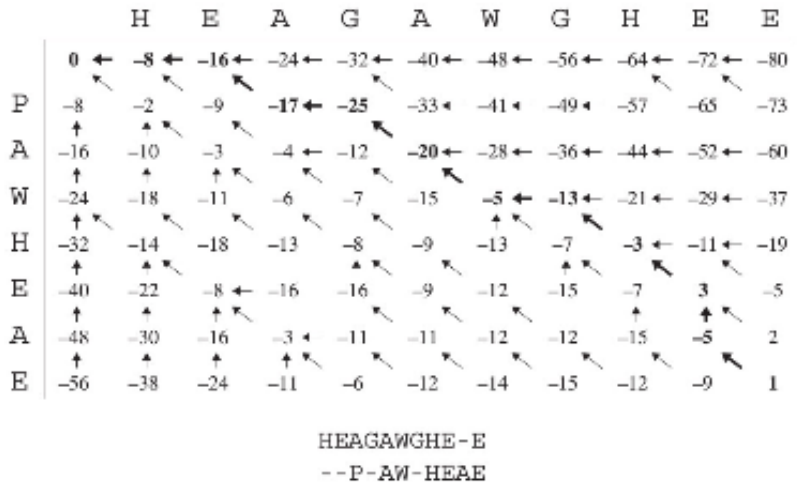


Figure 10: Example of global alignment

4.3 Local alignment: Smith-Waterman algorithm

The keys to local alignment are to use a slightly more complex scoring function, and to use a different method for reading the desired alignment from the table (the traceback).

Naïve approach In this approach, the idea is to apply the N-W algorithm for any starting point, look for the maximal ending score for each one of them and the finally look for the maximum overall score. This is however extremely time consuming and very complex. Indeed, the computation complexity of the N-W algorithm is $\mathcal{O}(n^2)$, and as there are $\mathcal{O}(n^2)$ starting point, the overall complexity is $\mathcal{O}(n^4)$.

S-W algorithm The recursive relation now counts four options

$$F(i,j) = max \begin{cases} 0 \\ F(i-1,j) - d \quad (1) \\ F(i,j-1) - d \quad (2) \\ F(i-1,j-1) + s(x_i, y_i) \quad (3) \end{cases}$$

The fourth option correspond to whenever a cell would take on a negative value. It is instead assigned a zero value. One consequence of this is that now the table is initialized with zeros. Once the table is filled, finding the best local alignment is simple. First, locate the highest value in the matrix, and trace-back as before until we reach a zero entry. This path constitutes the best local alignment. The score of the local alignment is the highest element in the table, whereas before it was the element found in the bottom-right cell.

The S-W algorithm has however some requirements

- scores for strong mismatches need to be negative, meaning worse than reset, otherwise long stretches or unrelated sub-sequences could be aligned
- scores for matching similar residues need to be positive, otherwise 0 everywhere

The computational complexity of the S-W algorithm is, for $n \approx m$, $\mathcal{O}(n^2)$.

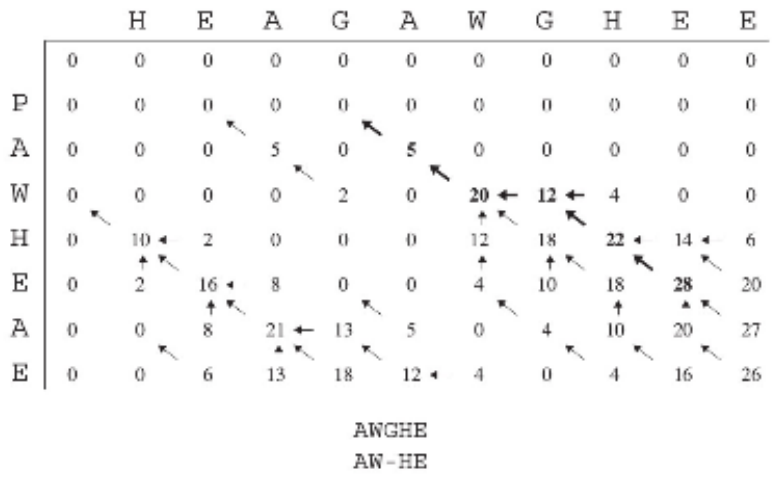


Figure 11: Example of local alignment

4.4 Several alignment variants

4.4.1 Semi-global alignment

In this type of alignment, the gap at the beginning of either sequence are not penalized. So the initialization is $F(i,0) = F(0,j) = 0$. The recursive rules are the same as for global alignment.

4.4.2 Local alignment with repeats

In a local alignment, several local matches of sequence y in sequence x can be found. In order to find those, one sets a threshold T, which will limit the considered matches scores. The initialization of the table is

$$\begin{aligned} F(0,0) &= 0 \\ F(i,0) &= max \begin{cases} F(i-1,0) \\ F(i-1,j) - T \quad \forall j = 1 \;\; m \end{cases} \end{aligned}$$

$F(i,0)$ is the best sum of completed match scores to the sub-sequence $x_{1\ i}$, assuming that x_i is an unmatched region. The recursive rules become

$$F(i,j) = max \begin{cases} F(i,0) \\ F(i-1,j) - d \\ F(i,j-1) - d \\ F(i-1,j-1) + s(x_i, y_i) \end{cases}$$

The cell $F(n+1,0)$ gathers the total scor of k matches $-kT$.

4.4.3 Gap penalties

Linear gap penalty So far a linear gap penalty was used $\gamma(g) = -dg$. However an affine gap penalty is more relevant $\gamma(g) = -d - e(g-1)$.

Affine gap penalty The first gap and the subsequent gaps need to be distinguished, for both sequences. Therefore, instead of have a single state $F(i,j)$, the are 3 states: the M state when x_i is aligned to y_i, the I_x state when x_i is aligned to a gap and the I_y state when y_j is aligned to a gap.

$$
\begin{aligned}
M(i,j) &= max \begin{cases} M(i-1,j-1) + s(x_i, y_i) \\ I_x(i-1,j-1) + s(x_i, y_i) \\ I_y(i-1,j-1) + s(x_i, y_i) \end{cases} \\
I_x(i,j) &= max \begin{cases} M(i-1,j) - d \\ I_x(i-1,j) - e \end{cases} \\
I_y(i,j) &= max \begin{cases} M(i,j-1) - d \\ I_y(i,j-1) - e \end{cases}
\end{aligned}
$$

4.5 Significance assessment

4.5.1 Statistical significance of the alignment score

Once the high alignment score is determined, it has to be decided if this is due to chance of biology. This significance is assessed through a statistical test, with the following procedure

- Compute pairwise random alignment scores

  between x and random sequences (e.g. 500 permutations of y)
  between y and random sequences (e.g. 500 permutations of x)

- Compute the histogram of random alignment scores (normalized by length)
- Determine the p-value $p - val = 1 - percentile$ of the actual alignment score of x and y. If the p-value is smaller than a certain threshold α, meaning the score is in the top α of the randomly generated scores, then the score is significant.

4.5.2 Computational complexity

As was already described before, the computational complexity of a single alignment is $\mathcal{O}(nm) \approx \mathcal{O}(n^2)$. However, many alignments (e.g 1000) have to be computed. A sound procedure is thus needed. However, the actual distribution of alignment scores is know to follow an Extreme Value Distribution, which is represented in red on Figure 12.

Extreme Value distribution For this distribution, the probability of a score larger than S is $P(x > S) = 1 - \exp(-kmne^{-\lambda S})$ where K and λ are constant fitted by the alignment software, depending on the substitution scoring matrix. The lack of normality is due to the non-independence between possible starting points of matches.

4.5.3 Concluding remarks

All the above algorithms maximizing a score can be easily adapted when minimizing an edit distance or cost. $\mathcal{O}(nm) \approx \mathcal{O}(n^2)$ is too much when aligning a query sequence to a large database of possibly homologous sequences. Therefore, BLAST and FASTA are heuristic algorithms to speed-up such computation.

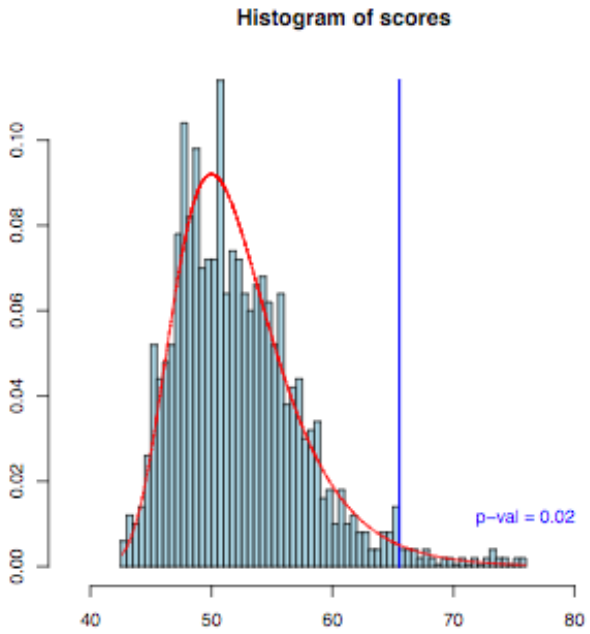


Figure 12: Example of an histogram of scores

The notions described above can be generalized to find a multiple alignment between k sequences: this is called multiple alignment. Dynamic programming scales as $\mathcal{O}(n^k)$ to find an optimal solution. Therefore, CLUSTALW is an heuristic algorithm to speed-up such computation.

Further extensions are the summary of a multiple alignment into a probabilistic model (HMM) and the computation an alignment between a new sequence and this model.

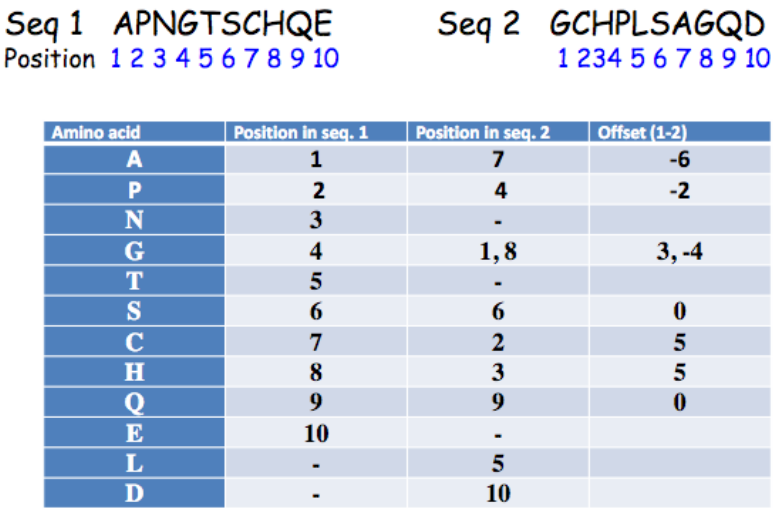
5 Database searching for similar sequences

Finding a gene in a new organism (e.g., a crop plant) with a sequence similar to a model organism gene (e.g., yeast) provides a prediction that the new gene has the same function as in the model organism. Such searches are becoming quite commonplace and are greatly facilitated by programs such as FASTA and BLAST.

At the end of this chapter, one must be able to explain the differences explain the differences between an alignment of two defined sequences and the search for homologs in databases, to use tools like Fasta and BLAST and to understand the statistical significance of a score.

Two kinds of algorithm can be used. On the one hand there are Dynamic programming algorithms, like Smith and Waterman which fins the optimal local alignment. Database searches for these algorithms can take an hour to minutes with dedicated hardware. A fast implementation of the S-W method is provided by `MPSearch`. On the other hand, there are Hash coding algorithms. These algorithms introduces heuristic methods. There is a gain of speed, but a the expense of some loss in sensitivity (50-100 times faster than S-W). These algorithms regroup the BLAST and FASTA tools.

Hashing method In this method, a lookup table showing the positions of each word of length k, or k-tuple, is constructed for each sequence. The relative positions of each word in the two sequences are then calculated by subtracting the position in the first sequence from that in the second. Words that have the same offset position are in phase and reveal a region of alignment between the two sequences. Using hashing, the number of comparisons increases linearly in proportion to average sequence length.



Amino acid	Position in seq. 1	Position in seq. 2	Offset (1-2)
A	1	7	-6
P	2	4	-2
N	3	-	
G	4	1, 8	3, -4
T	5	-	
S	6	6	0
C	7	2	5
H	8	3	5
Q	9	9	0
E	10	-	
L	-	5	
D	-	10	

Imagine the following sequence q `TCGGATTCGT ACGGTACGGA TC` with length $l = 22$ and k-tuple = 2. The k-tuples are `TC,CG,GG,GA,AT,TT,TC,CG,GT,TA,AC,CG,GG,GT,TA,AC,CG, GG,GA,AT,TC`.

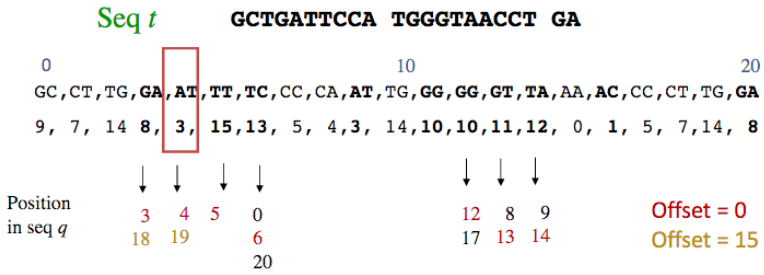
The chosen system allows vectorial registration of the pairs

$$Idc(II) = v(II_1)4^1 + v(II_2)4^0 \quad \text{and} v(A) = 0, v(C) = 1, v(G) = 2, v(T) = 3$$

The index value of the pairs are `13,6,10,8,3,15,13,6,11,12,1,6,10,11,12,1,6,10,8,3,13`.

The first tuple TC stats at the 0 position and has an index value of 13. We then go on filling each cell; if an index value already registered is met, move it to C and insert the new location in T.

The table allow easy detection of repeated tuples and similar regions.

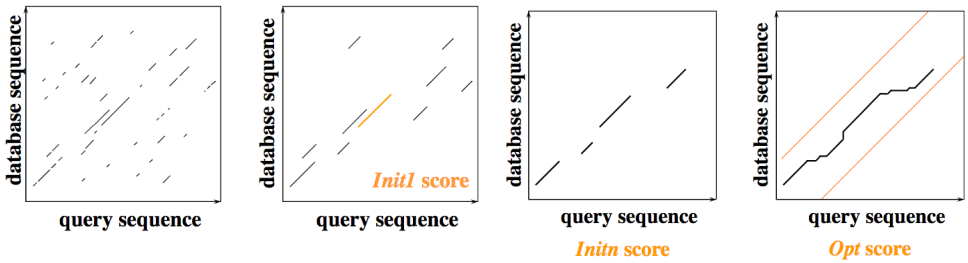


5.1 Fasta

FASTA is a program for rapid alignment of pairs of protein and DNA sequences. Rather than comparing individual residues in the two sequences, FASTA instead searches for matching sequence patterns or words, called k-tuples. The program then attempts to build a local alignment based on these word matches.

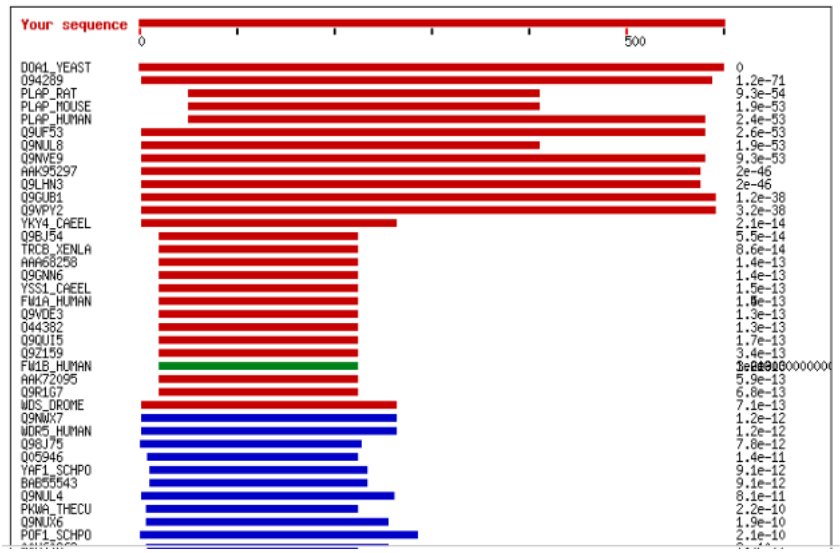
5.1.1 Method

- Search “k-tuples” common between query sequence and database sequence
- The 10 best-matching regions are evaluated using a scoring matrix and gap penalty
- Several segments on different ”diagonals” are joined into longer regions of score *initn*
- Best local alignment using the Smith-Waterman method within a restricted area (no more true in FASTA3)



5.1.2 Example of Fasta output

Graphical overview of similarity



Statistics

Comparison scores and sequence alignments

```
FASTA (3.39 May 2001) function [optimized, BL50 matrix (15:-5)] ktup: 2
 join: 38, opt: 26, gap-pen: -12/ -2, width:  16
 Scan time: 11.190
The best scores are:                                      opt bits E(693540
SWALL:D0A1_YEAST P36037 D0A1 PROTEIN.              ( 715) 4314  893       0
SWALL:O94289 O94289 WD REPEAT-CONTAINING PROTEIN.  ( 713) 1287  274 1.2e-71
SWALL:PLAP_RAT P54319 PHOSPHOLIPASE A-2-ACTIVATIN  ( 647)  996  214 9.3e-54
SWALL:Q9NUL8 Q9NUL8 CDNA FLJ11281 FIS, CLONE PLAC  ( 544)  990  213 1.9e-53
SWALL:PLAP_MOUSE P27612 PHOSPHOLIPASE A-2-ACTIVAT  ( 646)  991  213 1.9e-53
SWALL:PLAP_HUMAN Q9Y263 PHOSPHOLIPASE A-2-ACTIVAT  ( 738)  990  213 2.4e-53
SWALL:Q9UF53 Q9UF53 HYPOTHETICAL 87.2 KDA PROTEIN  ( 795)  990  213 2.6e-53
SWALL:Q9NVE9 Q9NVE9 CDNA FLJ10780 FIS, CLONE NT2R  ( 795)  981  211 9.3e-53

>>SWALL:O94289 O94289 WD REPEAT-CONTAINING PROTEIN.       (713 aa)
 initn: 792 initl: 475 opt: 1287  Z-score: 1423.7  bits: 273.9 E(): 1
Smith-Waterman score: 1291;  37.202% identity (40.717% ungapped) in 6

               10        20        30        40        50
sp      MGYQLSATLKGHDQDVRDVVAVDDSKVASVSRDGTVRLWSK-DDQWLGTVVYTGQGFLN
          :.::  : :: :::: : ....  ..:.::::  .: . . .:      . .::.:
SWALL: MTSYELSRELGGHKQDVRGVCSISNELIGSASRDGTYSVWEQINGEWTPHFYENHEGFVN
               10        20        30        40        50        60

       60        70        80        90       100       110
sp     SVCYDSEKELLLFGGKDTMINGVPLFATSGEDPLYTLIGHQGNVCSLS-FQDGVVISGSW
        :::     .    ::.:    :. :  .. ..: : :.::..:.:: : ... ..:.:::
SWALL: CVCYVPAIDKNSRGGQDKC--GI-LQEVGTNSPSYYLFGHESNICSASALNSETIITGSW
               70          80        90       100       110
```

5.2 Basic Local Alignment Search Tool (BLAST)

BLAST is a great improvement in speed, with a modest decrease in sensitivity. This tools minimizes search space instead of exploring entire search space between two sequences. It finds short exact matches ("seeds"), only explores locally around these "hits".

Like FASTA, the BLAST algorithm increases the speed of sequence alignment by searching first for common words or k-tuples in the query sequence and each database sequence. Whereas FASTA searches for all possible words of the same length, BLAST confines the search to the words that are the most significant.

5.2.1 BLAST algorithm

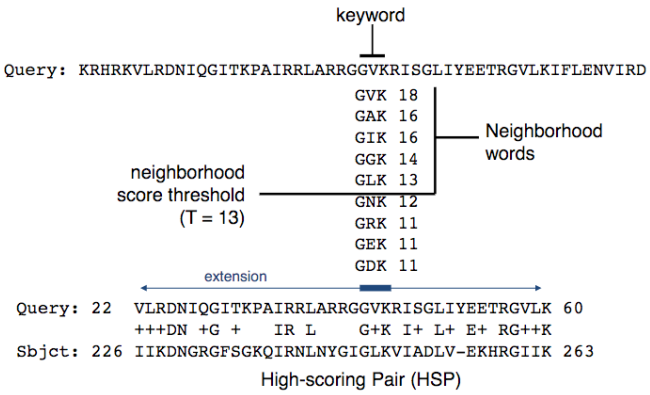
The algorithm is based on

1. Keyword search of all words of length w from the query of length n in database of length m with score above threshold

   - The word length is fixed at 3 (formerly 4) for proteins and 11 for nucleic acids (3 if the sequences are translated in all six reading frames), in order to achieve a word score that is high enough to be significant but not so long as to miss short but significant patterns
   - Matches with any other combination of 3 amino acids are also evaluated using a scoring matrix to generate a list of neighbourhood words (cutoff score T)

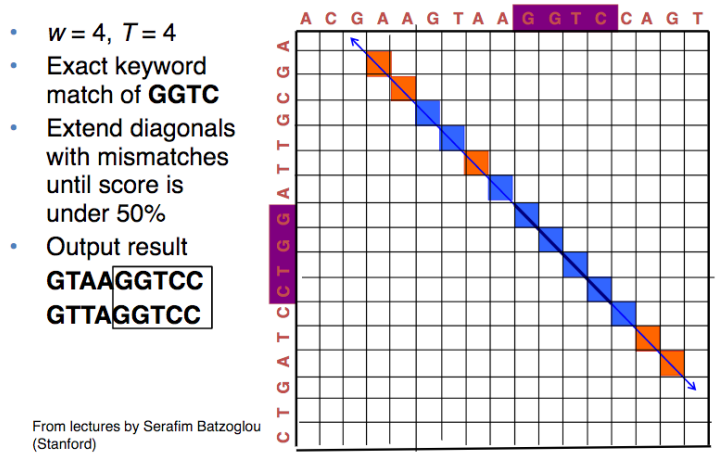
2. Local ungapped alignment extension for each found keyword (Extend result until longest match above threshold is achieved)

3. Running time $\mathcal{O}(nm)$



In the original BLAST method, an attempt was made to extend an alignment from the matching words in each direction along the sequences, continuing for as long as the score continued to increase. The extension process in each direction was stopped when the accumulated score stopped increasing and had just begun to fall a small amount below the best score found for shorter extensions. At this point, a larger stretch of sequence (called the HSP or high-scoring segment pair), which has a larger score than the original word, may have been found.

Example

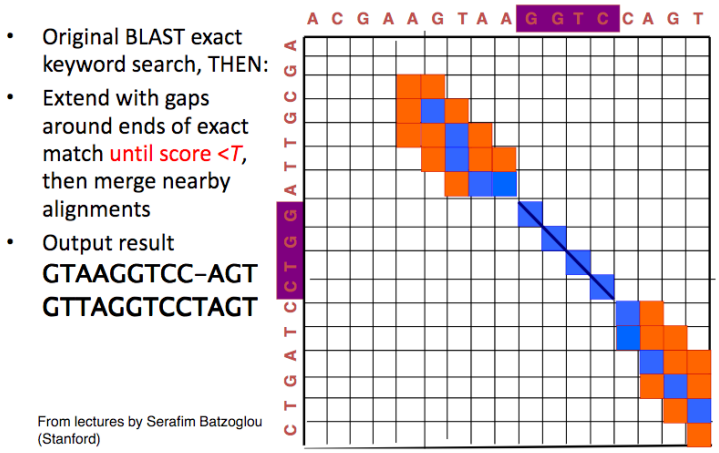


5.2.2 BLAST 2

In the first version, the extension process in each direction was stopped when the accumulated score stopped increasing and had just begun to fall a small amount (X) below the best score.

In BLAST2, only words lying on the same diagonal and within distance A of each other are joined and extended as described above.

Example



For each HSP score greater than a cut-off score S, an optimal alignment with gaps is produced with the Smith-Waterman method. The score is obtained and the expect value (E) for that score is calculated. The match is reported when the expect score satisfies the threshold parameter E.

5.2.3 Sequence filtering

The BLAST programs include a feature for filtering the query sequence through programs that search for low-complexity regions or for sequence repeats. Indeed, some regions of DNA and protein sequences consist of long repeated runs of a single residue or a pattern of residues. These repetitive sequences are not usually of interest to biologists, and close matches to these sequences may ” mask out ” lower-scoring matches to homologous sequences.

These low-complexity regions are filtered with the SEG program and then ignored by the BLAST program because they tend to give high scores that do not reflect sequence similarity but rather the occurrence of low-complexity or repetitive sequences. Removing these types of sequences increases emphasis on the more significant database hits.

5.3 Statistics of sequence similarity scores

When faced with similarity scores, the following questions arise: Given an alignment score, how strong is the similarity it represents? What is the probability of having such a high score with unrelated sequences? What is the expected number of alignments having such a high score with unrelated sequences from a DB search?

5.3.1 Modeling a random DNA sequence alignment

The first step for significance studies of similarity score consists in the modeling of a random sequence alignment. All random sequences must have the same length, without insertion or deletion. The probability that there are m matching sites for sequences of length N is given by a binomial distribution

$$P(m) = C_m^N a^m (1-a)^{N-m}$$

The mean of this binomial distribution is $\mu = Na$ and the variance is $\sigma^2 = Na(1-a)$.

The binomial distribution can be approximated normal distribution

$$Y(m) \frac{1}{\sqrt{2\pi\sigma^2}} exp\left[-\frac{(m-\mu)^2}{2\sigma^2}\right]$$

If we defined $z = \frac{m-\mu}{\sigma}$, this normal distribution has a mean $\mu = 0$ and a variance $\sigma^2 = 1$, with the following definition

$$Y(m) \frac{1}{\sqrt{2\pi}} exp\left[-\frac{z^2}{2}\right]$$

The value of a the probability for a given z is given in tables.

Example: for 120 base matches in a sequence of length $N = 400$, with mean $\mu = 0\ 25 * N$ and $\sigma^2 = 0\ 1875N$, $z = \frac{120-100}{\sqrt{75}} = 2\ 309$. From Tables, we find $p(z > 2\ 31) = 0\ 0105$. This means the match is significant even if the percentage is quite low (30%).

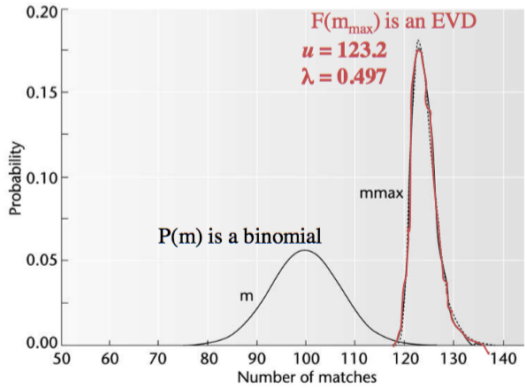
Extreme Value Distribution Simulation shows that the probability distribution for database search scores follows and Extreme Value Distribution.

For a given random sequences alignment, we calculate the score of the most closely matching sequence from the database, namely m_{max} and m the match between pairs of sequences starting at the same ordinate. For a large of alignment, m_{max} will be considered instead of m.

The EVD and its probability are defined as

$$F(S_{max} = \lambda \cdot exp\left[-\lambda(S_{max} - u)\right] \cdot exp\left[e^{-\lambda(S_{max}-u)}\right]$$

$$P(S \geq s_{obs} = 1 - exp\left[e^{-\lambda(s_{obs}-u)}\right]$$



For 2000 randons sequences of (G+C) with length $N = 200$, a score of 130 matches is just big enought to be significant according to EVD as $p(130) = 0\ 034$ while for a (wrong) normal distribution, $p(Z = 4\ 2) < 0\ 0001$.

The normalized definition of this distribution, for $S' = \lambda(S - u)$, which has $u = 0$ and $\lambda = 1$, is defined ans

$$F_{S'} = exp\left[-S' - e^{-S'}\right] \quad \text{and} \quad P(S' \geq s'_{obs} = 1 - exp\left[e^{-S'_{obs}}\right]$$

5.3.2 FASTA statistics and scores

FASTA scores are expressed as normalized Z scores: $Z = 50 + 10z$ with $z = (S - \mu)/\sigma$.

The λ and u parameters of the EVD are expressed in terms of μ and σ like

$$\lambda = \frac{1\ 2825}{\sigma} \quad \text{and} \quad u = \mu - 0\ 45\sigma$$

The probability is thus expressed as

$$p(z) \approx 1 - exp[-e^{-1\ 2825z - 0\ 5772}]$$

The expectation depends on the number of library sequences D

$$E(z) \approx p(Z \geq z) \cdot D$$

5.3.3 BLAST statistics

For BLASt, the significance of a score is expressed as the Expectation value, which is the number of alignments between your sequence and randomly chosen sequences giving a score as good as the one observed.

$$E(S > x) = KMNe^{\lambda x}$$

where M and N are the effective length of, respectively, query sequence and database sequences. Their product is the effective search space. Parameters K and λ are determined from the EVD.

The bits score allows to compare results between different database searches, even using different scoring matrices.

$$E(S_{bits}) = \frac{MN}{2^{S_{bits}}} \quad \text{with} \quad S_{bits} = \frac{\lambda S - lnK}{ln2}$$

6 Hidden Markov Model

6.1 Motivating example: gene finding

Imagine one strand of DNA sequence. The problem with gene finding is detecting the coding versus the non-coding ones. This is equal to identifying the splicing between exons and introns for eukaryotes.

The baseline approach consists in

1. finding all ORFs in the original sequences
2. finding all ORF in random permutations of the original sequence
3. accepting as significant ORFs any ORF in the original sequence longer than a prescribed length, usually the 99% percentile of random ORF lengths (p-value 1%).

The approach has however some limitations, namely that coding on non-coding fragments are not characterized by their length. Likewise, intron-exon boundaries need also to be identified for eukaryotes.

An alternative approach consist in, for a given labeled data, estimate two different statistical models M_+ and M_-, respectively for coding and non coding fragments (or exons and introns for eukaryotes). Then, for any new sequence to analyze, one has to look at a subsequence x defined by a sliding window of size L and compute the log-odds ration in order to decide if x is coding or non-coding. Indeed, x will be a coding subsequence if

$$\log \frac{P(x|M_+)}{P(x|M_-)} = \log \frac{\prod_{i=1}^{L} P(x_i|M_+)}{\prod_{i=1}^{L} P(x_i|M_-)} = \sum_{i=1}^{L} \log \frac{P(x_i|M_+)}{P(x_i|M_-)} > 0$$

But what should be the sliding door length L ? Ideally, there should be no need to use such a fixed window size. And which model should be used to present specific fragments ? There are several possibilities: multinomial models, Markov chains or Hidden Markow models.

6.1.1 Multinomial models

In this model, the nucleotides are independent and identically distributed. This means that at each sequence position i, any of the four symbols can occur, with a probability that does not depend on the position i. The probability distribution of the model is $p = (p_A, p_C, p_G, p_T)$. All transition probabilities pointing to the same state are equal. The probability of a given sequence is equal to the product of the probabilities of all individual nucleotides. There is a mutlinomial model to for coding sequences, and a different one for non-coding ones.

6.1.2 Markov chain

Graphically, a Markov chain can be seen as a collection of 'states' (ex: A, C, G, T in case of DNA), each of which correspond to a particular residue, with arrows between the states (Figure 13).

A probability parameter is associated to each arrow, which determines the probability of a certain state following another one. The parameters are called transition probabilities and, for a dimer, is defined as

$$\hat{P}(x_i|x_{i-1}, M) = \frac{f(x_{i_1}x_i)}{f(x_{i-1})} \quad \text{from segments modeled by } M$$

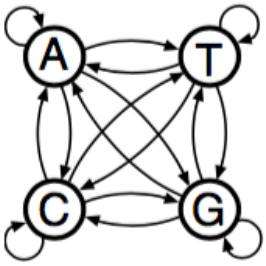


Figure 13: Transitions in Markov chain

The key property of Markov chains for a dimer is that the probability of each symbol x_i only depends on the value of preceding symbol x_{i-1}, not the entire previous sequence.

The log-odds computation then becomes

$$\log \frac{P(x|M_+)}{P(x|M_-)} = \sum_{i=1}^{L} \log \frac{\hat{P}(x_i|x_{i-1}, M_+)}{\hat{P}(x_i|x_{i-1}, M_-)} > 0$$

For a 3-mer, the probability of each symbol i only depends on the two previous symbols. This is called a second order Markov chain

$$\hat{P}(x_i|x_{i-1}, x_{i-2}, M) = \frac{f(x_{i-2}x_{i_1}x_i)}{f(x_{i-2}x_{i-1})} \quad \text{from segments modeled by } M$$

The log-odds computation then becomes

$$\log \frac{P(x|M_+)}{P(x|M_-)} = \sum_{i=1}^{L} \log \frac{\hat{P}(x_i|x_{i-1}, x_{i-2}, M_+)}{\hat{P}(x_i|x_{i-1}, x_{i-2}, M_-)} > 0$$

The limitations of the Markov Chain (MC) are that,

- as each MC from well annotated segments must be estimated, the problem needs to be solved 'manually' for a sufficiently large set of segments.
- gene lengths are variable but sliding window lengths are arbitrary? Computations with many possible sliding window lengths are expensive, so one needs to decide which window length is more relevant
- for eukaryotes, at least three models are needed (out-of-genes, introns and exons) and segment length is even more variable

The idea is thus to define a single model for all possible segments and estimate all transition probabilities simultaneously. Segmentation occur by finding the most likely state sequence to generate the whole sequence. That is why Hidden Markov Models (HMMs) were introduced. A HMM is a more compact representation of a sequence model, containing transition probabilities between states but also discrete emissions probabilities on states. There is one state for each type of segment (coding or non-coding). Sometimes states are hidden, but their emissions are still observed.

6.2 HMM definition

The idea behind HMM is to build a single model for an entire sequence, incorporating both Markov chains for coding and non-coding fragments. At each position, there is a small probability of switching from one Markov chain to the other. For a DNA sequence, there are thus

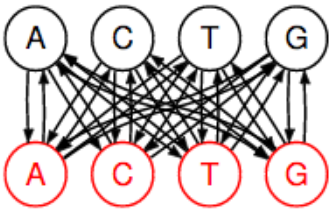


Figure 14: Hidden Markov Model

eight different labeled states: A_+, C_+, G_+, T_+ which emit A, C, G and T respectively in coding regions and states A_-, C_-, G_-, T_- correspondingly in non-coding regions. There is now no possible way to say if single symbol C was emitted by state C_+ or C_- just by looking at it.

Let us now formulate a notation for HMMs and derive the probability of a particular sequence of states. For a discrete HMM (with state emission), we define

- Σ is a finite alphabet: $\Sigma = \{A, C, G, T\}$ for DNA and $\Sigma = \{\text{the 20 a.a.}\}$ for proteins
- Q is a set of states (need not to have the same size as Σ)
- A a $|Q|\times|Q|$ transition probability matrix ($\sum_{q' \in Q} A_{qq'} = 1$)
- B a $|Q|\times|\Sigma|$ emission probability matrix ($\sum_{a \in \Sigma} B_{qa} = 1$)
- π n initial probability distribution ($\sum_{q \in Q} \pi_q = 1$)

To define the likelihood $P(s, \nu|M)$ of a sequence $s = s_1 \dots s_{|s|}$ along a path or state sequence $\nu = q_1 \dots q_{|s|}$ in a HMM M, keep in mind that the path itself follow a Markov Chain, so the probability of a state depends only on the previous state.

$$P(s, \nu|M) = \prod_{i-1}^{|s|} P(s_i, q_i|M) = \pi_{q_1} \mathbf{B}_{q_1 s_1} \prod_{i=1}^{|s|} \mathbf{A}_{q_{i-1} q_i} \mathbf{B}_{q_i s_i}$$

The likelihood $P(s|M)$ of a sequence $s = s_1 \dots s_{|s|}$ in a HMM M is defined as the sum of the likelihood $P(s, \nu|M)$ over all paths (which are $\mathcal{O}(Q^{|s|})$).

$$P(s|M) = \sum_{\nu \in Q^{|s|}} P(s, \nu|M)$$

Example For the example depicted in Figure 15, the parameters and probabilities defined above are

$\Sigma = \{a, b\}$ $\qquad$ $Q = \{1, 2\}$

$$\mathbf{A} = \begin{bmatrix} 0\,1 & 0\,9 \\ 0\,7 & 0\,3 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0\,2 & 0\,8 \\ 0\,9 & 0\,1 \end{bmatrix} \qquad \pi = [0\ 40\ 6]$$

$P(abb, 122|M) = 0\,4 \times 0\,2/a \times 0\,9 \times 0\,1/b \times 0\,3 \times 0\,1/b$

$P(abb, M) = P(abb, 111|M) + P(abb, 112|M) + P(abb, 121|M) + P(abb, 122|M) + P(abb, 211|M) + \dots$

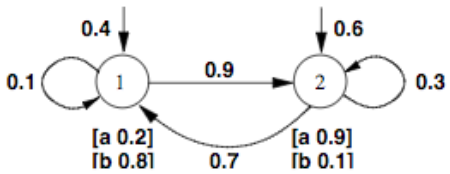


Figure 15: Hidden Markov Model: example

6.3 The three fundamental questions

Although it is no longer possible to tell what state the system is in by looking at the corresponding symbol, it is often the sequence of underlying states that we are interested in. To find out what the observation sequence 'means' by considering the underlying states is called *decoding*. Here the most common one will be described, called the **Viterbi algorithm**.

6.3.1 How to compute the most likely state sequence?

The most probable path or state sequence ν^* is defined as

$$\nu^* = argmax_\nu P(s, \nu|M)$$

It is the path with the highest probability, and can be found recursively.

Suppose the probability of the most probable path ending in state k at step/observation t. This probability is defined as an auxiliary quantity

$$P(s_1 \dots s_t, \nu^* = k|M) = \gamma(k, t)$$

All sequences have to start at state 1 (the begin state), so an initial condition is needed. The Viterbi recurrence algorithm is defined as

Initialization: $\gamma(k, 1) = \pi_k \mathbf{B}_{ks_1}$
Recurrence: $\gamma(k, t) = max_l[\gamma(l, t-1)\mathbf{A}_{lk}]\mathbf{B}_{ks_t}$
Termination: $P(s, \nu^*|M) = max_l \gamma(l, |s|)$

The time complexity of this algorithm is $\mathcal{O}(m|s|)$. Computations are usually done with logarithms in order to avoid underflow errors when implemented on a computer. The recurrence step then becomes

$$-\log \gamma(k, t) = min_l[-\log \gamma(l, t-1) - \log \mathbf{A}_{lk}] - \log \mathbf{B}_{ks_t}$$

The actual path ν^* can be recovered through the backpointers.It defines an alignment between states and symbols, meaning a segmentation of the sequence.

6.3.2 How to compute a sequence likelihood?

The probability of a sequence $P(s)$ for a HHM is the sum over all possible paths.

$$P(s|M) = \sum_\nu P(s, \nu|M)$$

Once again an auxiliary quantity $\alpha(k,t)$ is defined at the likelihood of emitting the first t symbols and reaching state k at time t.

$$\alpha(k,t) = P(s_1 \dots s_t, \nu_t = k|M)$$

The full probability can be calculated with the **forward recurrence algortihm**, defined as followed

Initialization: $\alpha(k,1) = \pi_k \mathbf{B}_{ks_1}$
Recurrence: $\alpha(k,t) = \sum_l [\alpha(l,t-1)\mathbf{A}_{lk}]\mathbf{B}_{ks_t}$
Termination: $P(s,\nu^*|M) = \sum_l \alpha(l,|s|)$

Like the Viterbi algorithm, the forward algorithm is often computed in a log scale in order to avoid underflow errors.

6.3.3 How to estimate HMMs parameters?

For a given HMM structure and one or several sequences to model, one must first estimate HMM parameters $\mathbf{A}, \mathbf{B}$ and π.

Supervised learning: when the state sequence is known It is easier to estimate to probability parameters when the paths are known for all the examples. When the paths are known, it is possible to calculate

- the number of times symbol i is observed on state k $f(k,i)$
- the number of times k is used $f(k)$
- the number of times a transition form state k to state l is used $f(k,l)$
- the number of times state k is used as first state $f_1(k)$

The estimation of the parameters becomes

$$\mathbf{B}_{ki} = \frac{f(k,i)}{f(k)} \qquad \mathbf{A}_{kl} = \frac{f(k,l)}{f(k)} \qquad \pi_k = \frac{f_1(k)}{\sum_{j=1}^{|Q|} f_1(j)}$$

Unsupervised learning: when the state sequence is unknown

Viterbi training In this approach, the most probable path for the training sequence are derived using the Viterbi algorithm given above, and these are used in the re-estimation process given in the previous section. In other words, Viterbi traning follows the following steps

1. Fix initial parameter values $\mathbf{A}^0, \mathbf{B}^0, \pi^0$.
2. Repeat until some stopping criterion is met (e.g. max number of iterations)
   - compute a most likely path through a Viterbi alignment for each learning sequence given the parameters $\mathbf{A}^i, \mathbf{B}^i, \pi^i$
   - estimate the emission and transition frequencies for such path(s)
   - recompute the parameter values $\mathbf{A}^{i+1}, \mathbf{B}^{i+1}, \pi^{i+1}$ from those frequencies

But Viterbi training is only an approximation as it considers that each learning sequence is generated along a single path, namely the most likely one.

Forward-Backward or Baum-Welch Algorithm A more accurate estimation is obtained if one considers all possible paths to generates each sequence. This is the case of the Baum-Welch algorithm. Actual frequencies are then replaced by expected frequencies. The overall principle is the same as for the Viterbi training. First, one must pick arbitrary model parameters. Then, for each sequence, one must calculate the forward and backward variables, and then adapt the model parameter $\mathbf{A}$ and $\mathbf{B}$, as well as π. The algorithm stop if the change in log likelihood is less than some predefined threshold or the maximum number of iterations is exceeded. This algorithm is a special case of expectation-maximization (EM) procedure.

6.4 Back to concrete examples

6.4.1 Segmentation into CpG islands

In the human genome wherever the dinucleotide CG occurs (frequently written CpG to distinguish it from the C-G base pair across the two strands) the C nucleotide (cytosine) is typically chemically modified by methylation. There is a relatively high chance of this methyl-C mutating into a T, with the consequence that in general CpG dinucleotides are rarer in the genome than would be expected from the independent probabilities of C and G. For biologically important reasons the methylation process is suppressed in short stretches of the genome, such as around the promoters or 'start' regions of many genes. In these regions we see many more CpG dinucleotides than elsewhere, and in fact more C and G nucleotides in general. Such regions are called CpG islands [Bird 1987]. They are typically a few hundred to a few thousand bases long.

Given a short stretch of genomic sequence, how would we decide if it comes from a CpG island or not? Second, given a long piece of sequence, how would we find the CpG islands in it, if there are any?

The baseline analysis uses a sliding window of a prescribed length in order to find CpG islands. But what length? A shorter length tends to be noisy while a larger on may miss some short islands. Actually, the length varies.

In order to model CpG islands, a 2-state model can also be proposer. The two states are, on one hand the GC-rich state with higher probability to emit G and C and on the other hand the alternative state, with lower probability to emit G and C. Transition probabilities reflects that it is much more likely to stay in a given state than switching regimes. Initial probabilities reflect the chance to start of not with a CpG island (by default $\pi = [0\ 50\ 5]$). Those two probabilities can either be a priori fixed or estimates through supervised learning, Viterbi or Baum-Welch algorithms.

6.4.2 Profile HMMs

See next section

7 Multiple alignment - Profile HMMs

7.1 Multiple alignment

A multiple (global) alignment of k sequences, is an assignment of gap symbols "-" into those sequences, or at their ends. The k resulting strings are placed one above the other so that every character or gap symbol in either string is opposite a unique character or a unique gap symbol in the other string. It can be represented as a $c \times k$ matrix, for some value of c, the ith row containing the ith sequence and (interspersed) gap symbols.

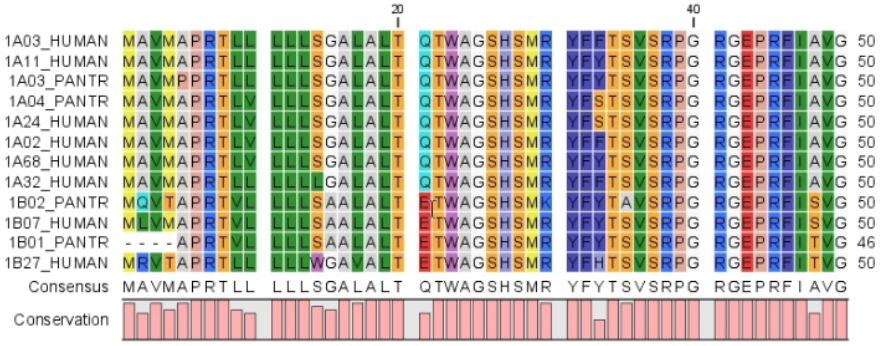


Figure 16: Multiple alignment: example

Local multiple alignment are also possible if one only looks for conserved segments.

Scoring a multiple alignment The scoring system should take into account at least two important features of multiple alignments: (1) the fact that some positions are more conserved than others, e.g. position-specific scoring; and (2) the fact that the sequences are not independent, but instead are related by a phylogenetic tree. However, the data needed to parameterise an evolutionary model is not available and so simplifying assumptions must be made. Therefore, the phylogenetic tree will be partly or entirely ignored while doing some sort of position-specific scoring of aligning structurally compatible residues.

Almost all alignment methods assume that the individual columns of an alignment are statistically independent. For a multiple alignment m with L columns, such a scoring function can be written as

$$S(m) = G + \sum_{i=1}^{L} S(m_i)$$

where $S(m_i)$ is the score of column i and G is the score of all gaps in m. Gap penalties are either linear, and in that case $s(a,-) = s(-,b) = -d$ and $s(-,-) = 0$, or either affine, in which case all gaps are scored separately.

Columns are scored by a 'sum of pairs' (SP) function using a substitution scoring matrix. The SP score for a column is defined as:

$$S(m_i) = \sum_{k<l} s(m_i^k, m_i^l)$$

where $s(a,b)$ is given by a substitution scoring matrix like PAM or BLOSUM.

7.1.1 Computation of an optimal multiple alignment

Dynamic programming It is possible to generalize pairwise dynamic programming to multiple sequences. However, this turns out to be impractical for more than a few sequence. For a pairwise alignment, there were three possible moves in the recursive relation (either a gap in the first sequence, a gap in the second sequence or a match/mismatch). But for N sequences, there are $2^N - 1$ possibles moves in the recursive relation. For example, the table that was filled using the recursive relation in pairwise alignment transforms into a three dimensions matrix (so a hyper-cube) for three sequences. Initialization, termination, and traceback steps for the algorithm also follow analogously from the pairwise dynamic programming algorithm.

The time complexity for k sequences of average length n is in $\mathcal{O}(2^k n^k)$ and the space complexity in $\mathcal{O}(n^k)$ (for storing the hyper-cube). In practice, computation must be limited to very few sequences due to the exponential growth with k.

MSA algorithm MSA is an optimized DP algorithm which first computes all pairwise alignments and then limits the exploration of the (hyper-)cube to regions consistent with those alignments. The time complexity in $\mathcal{O}(k^2 n^2)$ but somewhat complex to program. MSA can optimally align ≈ 10 sequences of up to 200-300 residues in reasonable time. A recent parallel extension G-MSA is reported to align up to 500 sequences of 236 residues on average within 10 seconds on a Linux machine including 2 cores with GPUs.

7.1.2 Heuristic algorithms

Progressive alignment methods This works by constructing a succession of pairwise alignments. Initially, two sequences are chosen and aligned by standard pairwise alignment; this alignment is fixed. Then, a third sequence is chosen and aligned to the first alignment, and this process is iterated until all sequences have been aligned.

Progressive alignment is heuristic: it does not separate the process of scoring an alignment from the optimization algorithm. It does not directly optimize any global scoring function of alignment correctness. The advantage of progressive alignment is that it is fast and efficient, and in many cases the resulting alignments are reasonable.

Greedy heuristic algorithms Succession of pairwise alignments

1. Two sequences are aligned first
2. A sequence is added to a group of already aligned sequences
   - compute all pairwise alignments between s and an existing group g of aligned sequences
   - the highest scoring pairwise alignment determines how the new sequence s is aligned to the group g
3. A group g_1 of sequences is aligned to another group g_2 of sequences
   - all sequence pairs between g_1 andg_2 are tried
   - the best pairwise alignment determines the alignment of both groups

There are however some issues with progressive pairwise alignments. For example, the degree of sequence conservation at each position should be taken into account. Also, mismatches at highly conserved positions should be more penalized. Finally, the order in which sequences are

incorporated in the multiple alignment matters. Theses aspects are all ignored by the sum of pairs scoring.

Profile alignment: ClustalW In order to solve all the issues discussed above, profiles were applied in progressive multiple sequence alignment. The exact definition of the scoring function used in profile–sequence or profile–profile alignment varies. Aligned residues are usually scored by some form of a sum-of-pairs score, but the handling of gaps varies substantially between different methods. One widely used implementation of profile-based progressive multiple alignment is the CLUSTALW program.

Algorithm: CLUSTALW progressive alignment The main step are

1. Construct a distance matrix of all $\frac{k(k-1)}{2}$ pairs by pairwise dynamic programming alignment followed by approximate conversion of similarity scores to evolutionary distances using the model of Kimura [1]
2. Construct a guide tree by a neighbor-joining clustering algorithm [1].
3. Progressively align at nodes in order of decreasing similarity, using sequence–sequence, sequence–profile, and profile–profile alignment.

Sequences are weighted to compensate for biased representation in large sub- families. The more distant the sequences, the high the weighting.

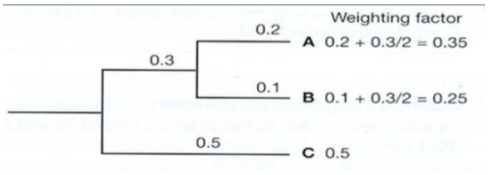


Figure 17: Weighting in a guiding tree: example

Position-specific gap-open profile penalties are multiplied by a modifier that is a function of the residues observed at the position. These penalties were obtained from gap frequencies observed in a large number of structurally based alignments. In general, hydrophobic residues (which are more likely to be buried) give higher gap penalties than hydrophilic or flexible residues (which are more likely to be surface-accessible).

Gap-open penalties are also decreased if the position is spanned by a consecutive stretch of five or more hydrophilic residues. Both gap-open and gap-extend penalties are increased if there are no gaps in a column but gaps occur nearby in the alignment. This rule tries to force all the gaps to occur in the same places in an alignment.

In the progressive alignment stage, if the score of an alignment is low, the guide tree may be adjusted on the fly to defer the low-scoring alignment until later in the progressive alignment phase when more profile information has been accumulated.

7.2 Profile HMMS

Multiple alignments are most often based on pairwise alignments. A new sequence x may be only distantly and locally related to each sequence in a known family (biological question).

[1] This will be explained in the chapter concerning phylogeny

Indeed, all pairwise alignments between x and each family members may look poor. Therefore, one needs to model statistical features shared by the family members. Computing the alignment between x and a probabilistic model of the family may be much more efficient computationally.

Several questions arise: given a multiple alignment between sequences how to build a global/local model M from it? How to compute the matching score between a query sequence x and the model M ? How to get rid of the initial alignment?

7.2.1 Position-specific scoring matrices (PSSM)

A PSSM report the frequency of each sate used at every position. It essentially represent a local model for a window length L and an ungapped score matrix from N sequences. The probability of observing sequence x (i.e., the product of nucleotide frequencies in the PSSM matrix) in the model M is given by

$$P(x|M) = \prod_{i=1}^{L} P(x_i|M) = \prod_{i=1}^{L} \frac{f(x_i)}{N}$$

To search a PSSM matrix across sequences, the following matrix score as to calculated

$$S = \sum_{i=1}^{L} \log \frac{P(x_i|M)}{q_{x_i}}$$

where q_{x_i} is the background model (e.g. multinomial model). One evaluates the score S between x and M for all positions x_i and sliding window length L

A

	1	2	3	4	5	6	7
A	1	4	1	2	0	17	13
C	28	5	5	0	3	3	2
G	0	0	4	0	25	1	7
T	2	22	21	29	4	10	9

B

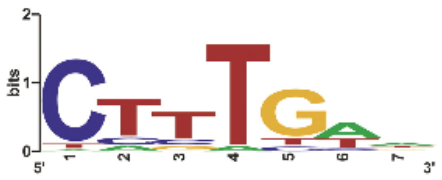


Figure 18: (A) Example of a PSSM, (B) Logo of the PSSM

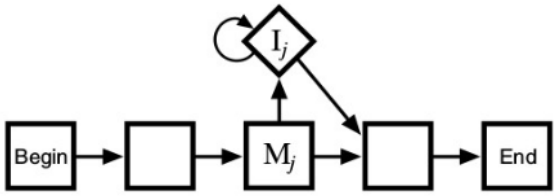
PSSM s are very simple HMMs. Indeed, one could see $P(x_i|M)$ as emission probabilities on match states. Transitions probabilities are equal to 1, which reflects a linear structure. However, one need to take into account possible gaps. Besides, one should also avoid a prescribed window length L. Therefore, this method does not work well for cases which include gaps of variable length.

7.2.2 Profile Hidden Markov Models (pHMM)

Profile HMMs encode position-specific information about the frequency of particular state as well as the frequency of insertions and deletions in the alignment. They are constructed from multiple alignments of homologous sequences. They contain match states, which describe the distribution of residues at each position, as well as insertion and deletion states that allow for the addition or removal of residues. There is a match state, insertion state, and deletion state for each column of a multiple alignment.

Let us start by only considering linear structure containing only match states.

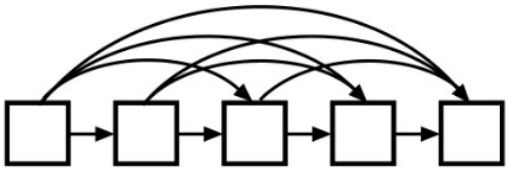
To account for portions of x that do not match anything in the model, we introduce a set of new states I_i , where I_i will be used to match insertions after the residue matching the ith column of the multiple alignment.



The I_i have emission distribution, but these are normally set to the background distribution q_{x_i}. This means that the contributions to the log-odds score is $\log \frac{P(x_i|I_i)}{q_{x_i}} = 0$. The transition probabilities to insert states and back are equivalent to affine gap penalties. The score a gap of length k is thus

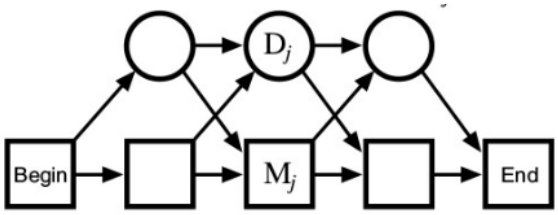
$$\log \mathbf{A}_{M_j I_j} + \log \mathbf{A}_{I_j M_{j+1}} + (k-1) \log \mathbf{A}_{I_j I_j}$$

Deletions, i.e. segments of the multiple alignment that are not matched by any residue in x, could be handled by forward 'jump' transitions between non-neighboring match states.



However, to allow arbitrarily long gaps in a long model this way would require a lot of transitions. Therefore, it is more convenient to introduce delete states which are silent states.

Because the silent states do not emit any residues, it is possible to use a sequence of them to get from any match state to any later one, between two residues in the sequence. The cost of a deletion will then be the sum of the costs of an M→ transition followed by a number of D→D transitions, then a D→M transition.



The full resulting HMM has the structure shown in Figure 19.

Derivation of a pHMM from a multiple alignment The key idea behind profile HMMs is that we can use the same structure as shown in Figure 19, but set the transition and emission probabilities to capture specific information about each position in the multiple alignment of the whole family.

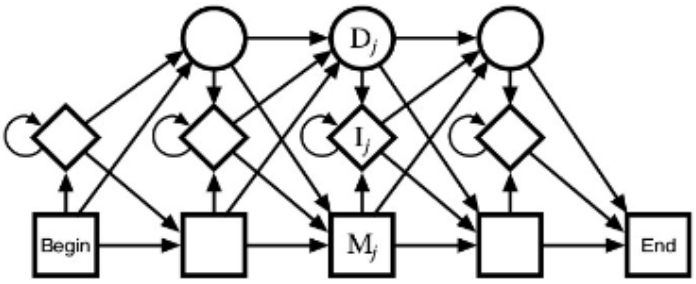


Figure 19: The transition structure of a profile HMM

From a given multiple alignment, match state are defined for each conserved position (e.g. at least 50% of the state). Inserts states are for example defined when the column contain at least 50% of gaps. Deletions state are defined when a gap occurs on match positions. Emission (for match states) and transition probabilities are estimated from the counts

- the number of times symbol i is observed on state k $f(k,i)$
- the number of times k is used $f(k)$
- the number of times a transition form state k to state l is used $f(k,l)$

The estimation of the parameters becomes

$$\mathbf{B}_{ki} = \frac{f(k,i)}{f(k)} \qquad \mathbf{A}_{kl} = \frac{f(k,l)}{f(k)}$$

Whenever the initial multiple alignment is limited to a few sequences, some emission/transition probabilities may be null. To avoid zero probabilities, pseudocounts can be added to the observed frequencies. These pseudocounts will serve a additive smoothing and are for example defined as $10^{-6} \leq \epsilon \leq 1$ and $10^{-6} \leq \epsilon' \leq 1$.

$$\mathbf{B}_{ki} = \frac{f(k,i)+\epsilon}{f(k)+\sum_i \epsilon} \qquad \mathbf{A}_{kl} = \frac{f(k,l)+\epsilon'}{f(k)+\sum_i \epsilon'}$$

Unsupervised learning The objective is to derive a pHMM starting from a collection of unaligned sequences instead of a multiple alignment. First, one needs to choose a general pHMM structure, like for example Figre 19. Next, the number of match states must be defined, for example half of the average sequence lengths. Finally the pHMM parameters are estimated through Viterbi or Baum-Welch.

Matching a sequence to a pHMM One of the main purposes of developing profile HMMs is to use them to detect potential membership in a family by obtaining significant matches of a sequence to the profile HMM. The match a sequence to a pHMM, several algorithm can be used.

The Viterbi algorithm will look for the most probable path along the pHMM generating the same sequence as the input sequence and will determine a similarity score. As this score is a multiplication of probabilities that are often smaller than zero, it is easier to look at the logarithm of the score. One must also include a background model in order to produce a log-odds score. The log-odd scale probability of the best path ending at, respectively state M, state

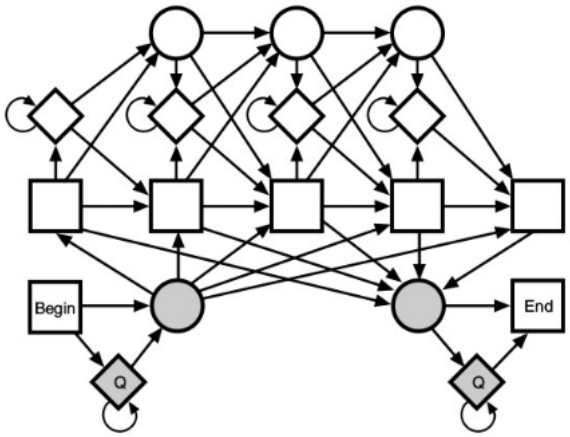
I and state D are written as

$$
\begin{aligned}
-\log \gamma^{M}(k,t) &= \log \frac{B_{M_k,s_t}}{q_{s_t}} + \min_{l} \begin{cases} -\log \gamma^{M}(l,t-1) - \log(A_{l,M_k}) \\ -\log \gamma^{I}(l,t-1) - \log(A_{l,M_k}) \\ -\log \gamma^{D}(l,t-1) - \log(A_{l,M_k}) \end{cases} \\
-\log \gamma^{I}(k,t) &= \log \frac{B_{I_k,s_t}}{q_{s_t}} + \min_{l} \begin{cases} -\log \gamma^{M}(l,t-1) - \log(A_{l,I_k}) \\ -\log \gamma^{I}(l,t-1) - \log(A_{l,I_k}) \\ -\log \gamma^{D}(l,t-1) - \log(A_{l,I_k}) \end{cases} \\
-\log \gamma^{D}(k,t) &= \min_{l} \begin{cases} -\log \gamma^{M}(l,t-1) - \log(A_{l,D_k}) \\ -\log \gamma^{I}(l,t-1) - \log(A_{l,D_k}) \\ -\log \gamma^{D}(l,t-1) - \log(A_{l,D_k}) \end{cases}
\end{aligned}
$$

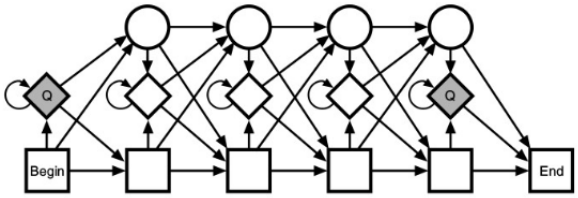
For the initialization, the algorithm must to be able to start at an insert or delete state. This happens when the beginning of the sequence does not match the first state of the model. A Begin pseudo-state is thus defined, and its log-scale probability is zero. The same is true for the termination, and therefore an End state is defined. Begin and End states are silent states like D states; they are non-emitting.

pHMM for non-global alignments The generalize the pHMM for non-global alignments, such as those that find local, repeat and overlap matches. A new model for the complete sequence x is specified, which incorporates the original profile HMM together with one or more copies of a simple self-looping model that is used to account for the regions of unaligned sequence. These behave very like the insert states that we added to the profile itself. They are called flanking model states, because they are used to model the flanking sequences to the actual profile match itself.

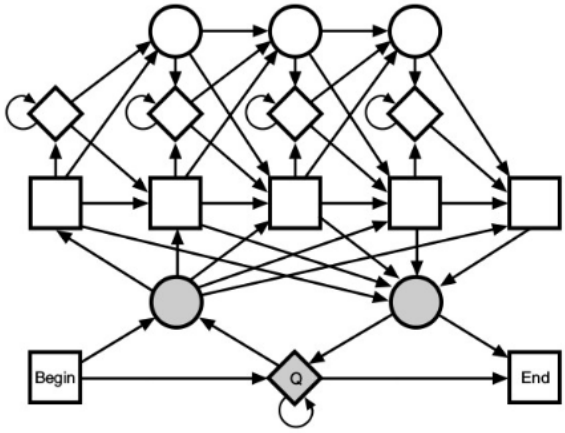
In other words, non-conserved fragments are modeled through flanking insert states using the background emission probabilities while flanking delete states allow for starting or ending the profile at any point.



The next issue is how to set all the transition probabilities from the left flanking state to different start points in the model. If all the probability is assigned to the first model state, then it forces this model to match only complete copies of the profile in the searched sequence, ensuring a type of 'overlap' match constraint. However, to allow for rare cases where the first residue might be missing, it may be wise in such cases to allow a direct transition from the flanking state into a delete state.



A final example similar to the first model for local matches allows repeat matches to subsections of the profile model.



8 Gene expression profiling

Gene expression profiling allows to determine the level at which each of a chosen set of cellular genes is expressed, measured as mRNA transcript abundance.

8.1 RNA purification

The isolation of RNA starts with the tissues homogenization and disruption. The first step, the complete disruption of cell walls and plasma membranes of cells and organelles is absolutely required to release all the RNA contained in the sample. Homogenisation is necessary to reduce the viscosity of the cell lysates produced by disruption. It shears the high-molecular weight genomic DNA and other high-molecular weight cellular components to create a homogeneous lysate. Depending on the nature of the sample, these steps can be performed on glass beads, by sonication, with lytic Enzymes and/or detergents/chaotropic agents or by grinding of N2-frozen sample in a mortar.

After the protein purification, different technologies and methods are available for isolation of RNA. One possible technique is represented in Figure 20

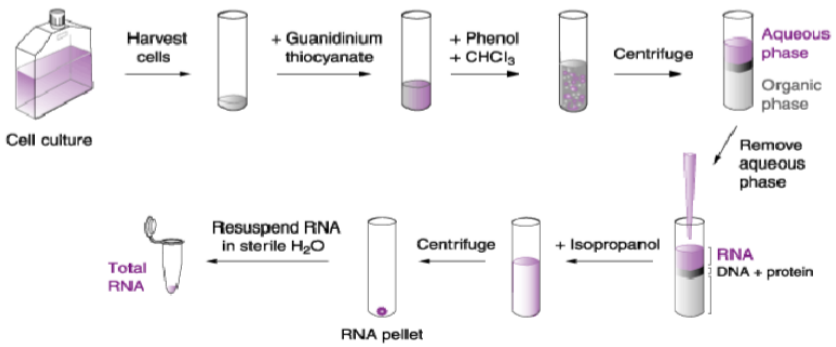


Figure 20: RNA extraction by alcohol precipitation

However, mRNA represents only a low fraction of total RNA population (5-10%). Purification is thus required for microarray analysis and sequencing. This can be done by affinity chromatography (Figure 21).

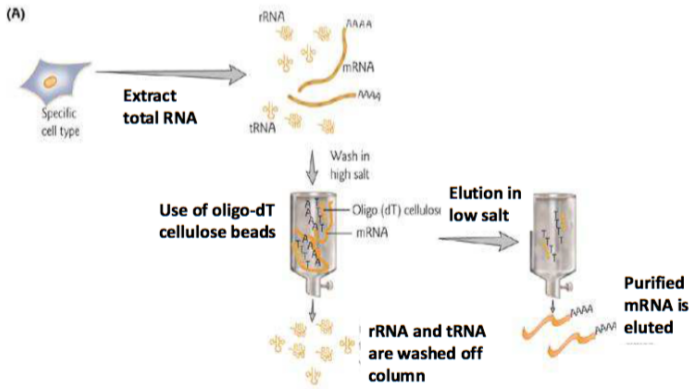


Figure 21: mRNA extraction by affinity chromatography

8.2 Qualitative detection of few targets: Northern blot

Northern blotting is a method for the detection of specific sequences in RNA preparations. Northern blot analysis is performed in three steps.

1. The RNA to be analysed is electrophoresed under denaturing conditions in an agarose/-formaldehyde gel.
2. The fractionated RNA is transferred from the denaturing agarose gel to a nitrocellulose or nylon membrane by capillary or vacuum transfer
3. The RNA sequences of interest are analysed by hybridisation using a labelled DNA or RNA probe

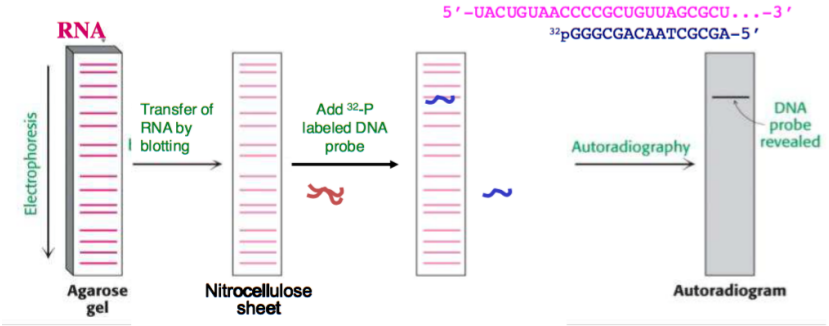
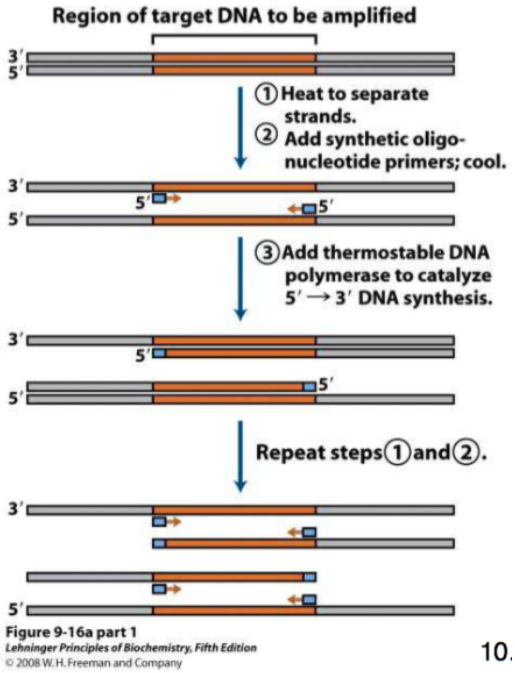


Figure 22: Northern blot analysis

8.3 Quantification of transcript expression: qRT-PCR

RT-PCR allows amplification of RNA through synthesis of complementary DNA (cDNA) ($\Rightarrow$ reverse transcription). It is used to detect RNA transcripts or virus with an RNA genome.

8.3.1 RNA amplification: Polymerase chain reaction

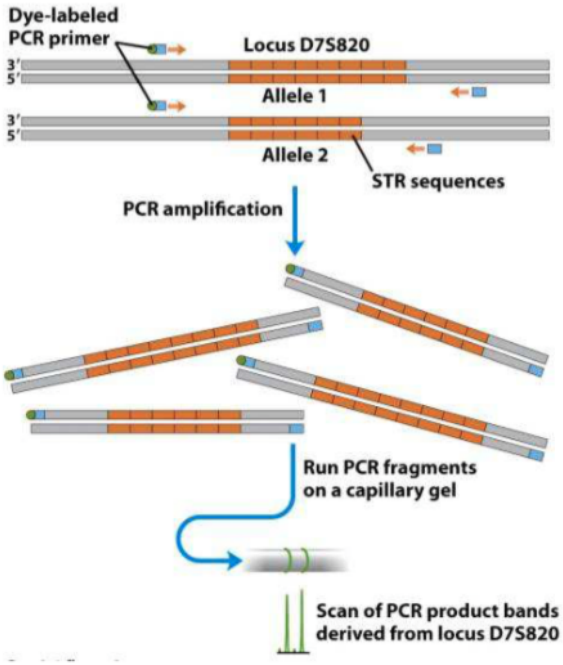


There are three steps:
(1) Strand separation (at 95 C);
(2) Hybridization of primers (at 50-60 C);
(3) DNA synthesis (68-72 C)

Selected DNA sequence (ng) is amplified (µg) in a 2-h reaction (20-30 cycles) using a thermal cycler.

10.

8.3.2 Application: PCR analysis of an Short Tandem Repeat (STR) locus



An STR (microsatellite in French) locus is a short DNA sequence ($\sim$ 4 nt), repeated many times in tandem at a particular location in a chromosome.
A STR locus from the US CODIS database may have as many as 82 different alleles.
The presence of both alleles results in two fluorescence signals separated by electrophoresis on a capillary gel (same machine as the one used for DNA sequencing).

8.3.3 Mutliplex-PCR

Multiplex-PCR consists of multiple primer sets within a single PCR mixture. The dyes linked to the PCR primers are of several different colors. The PCR primers are designed to produce products in a size range a distinct as possible from that generated by the primers targeted to other loci. Annealing temperatures for each of the primer sets must be optimized

The use of specific fluorescent oligonucleotide probes is required, like Taqman probes, Molecular beacons and scorpions. The fluorescent probes detect only the DNA being amplified The reporter probe significantly increases specificity, and enables quantification even in the presence of non-specific DNA priming. Probes with different-colored labels can be used in multiplex (provided that all targeted genes are amplified with similar efficiency).

Taqman probe Taqman probes contains a 5' fluorescent probe and a 3' quencher. During the PCR, both primers and the probe hybridize. The TAQ polymerase cleaves the fluorescent probe, with resulting PCR products and cleavage products. The fluorescent intensity increases.

8.3.4 Real-time or quantitative polymerase chain reaction (qPCR)

In real-time RT-PCR , fluorescence values are recorded during every PCR cycle and represent the amount of product amplified up to that point in the amplification reaction. The detection is based on (1) non-specific fluorescent dyes (SYBR Green) that intercalate with dsDNA or (2) specific oligonucleotides (probes).

The more template present at the beginning of the reaction, the fewer number of cycles it takes to reach a point in which the fluorescent signal is first recorded as being statistically significant above background. This point is defined as the threshold cycle (Ct). The quantity can be either

an absolute number of copies or a relative amount when normalized to DNA input or additional normalizing genes.

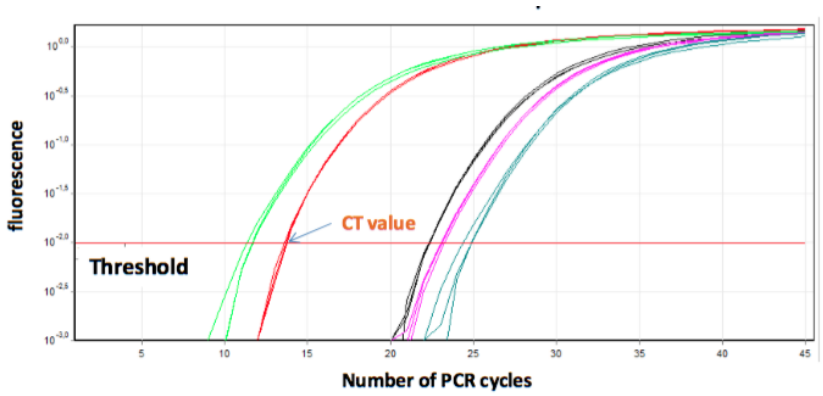


Figure 23: Threshold cycle (Ct)

Ct levels are inversely proportional to the amount of target nucleic acid in the sample. Δ Ct corresponds to the difference between the sample and a control.ΔΔ Ct is the difference of the ΔCt of the sequence of interest (SOI) and ΔCt of the reference sequence (RS), a house keeping gene sequence usually (U6 snRNA). cDNAs are balanced to yield equal Ct values for the reference.

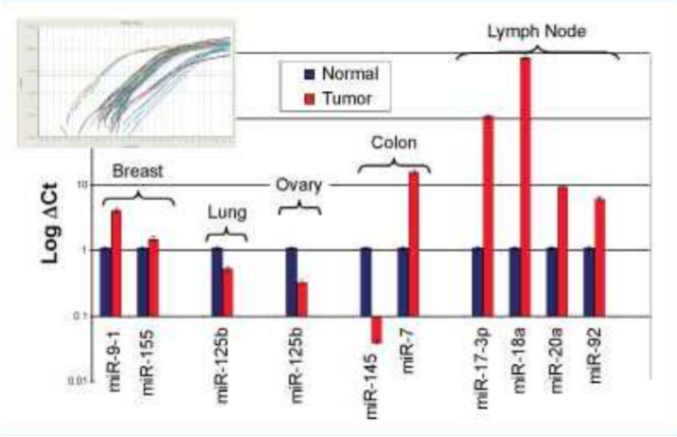


Figure 24: Quantification of qRT-PCR analysis

8.4 Genome-wide transcriptome analysis: microarrays

Reverse Transcription is followed by in vitro transcription (IVT) to allow amplification and labelling of cRNA with (i) Cy3 (green) or Cy5 (red) or (ii) with biotin.

⇒ Hybridization to cDNAs or oligonucleotide per gene.

⇒ Affymetrix only: staining with streptavidin-pephycoerythrin conjugates

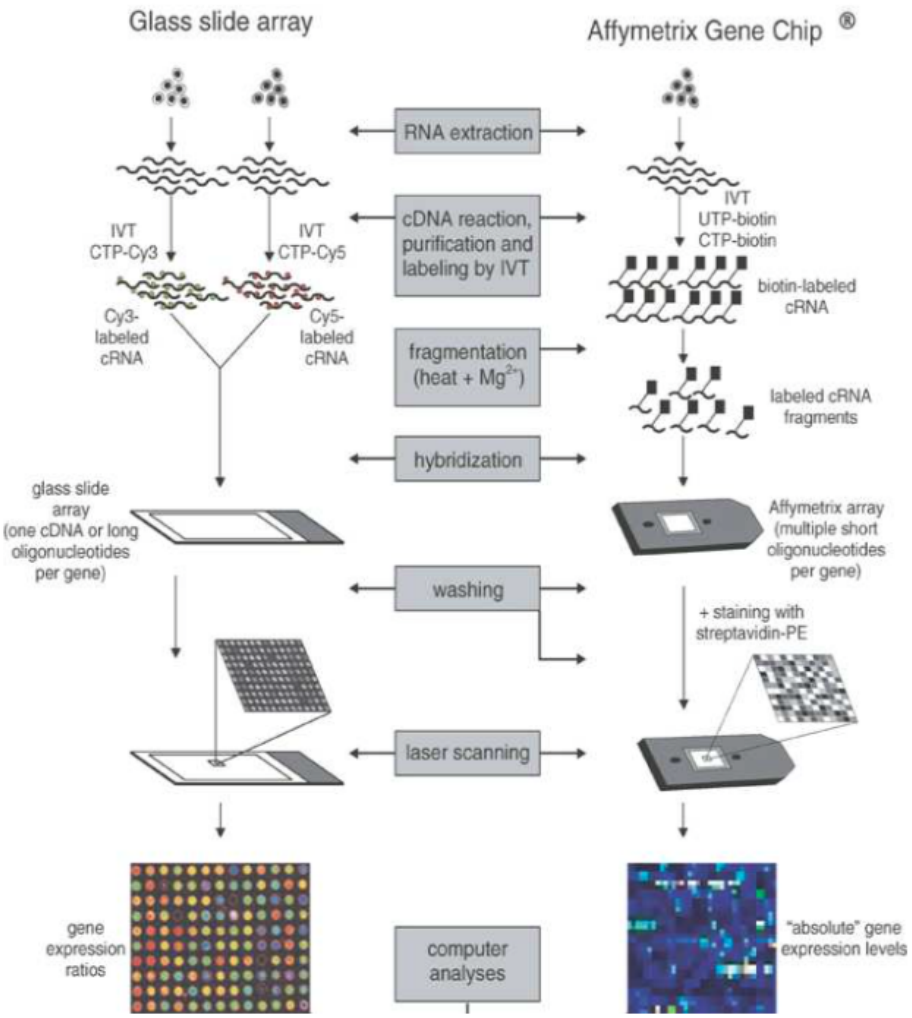


Figure 25: Glass slide vs. Affymetrix

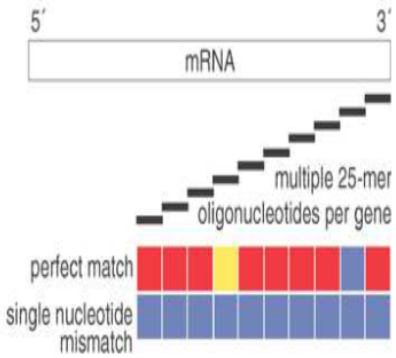
8.4.1 Glass slide experiments

RNA is extracted from two cell populations, for instance diseased and normal. cRNA is made with Cy3 (green) or Cy5 (red) labeled nucleotides. The two labeled cRNA samples are mixed and hybridized on a glass slide array, which is scanned with a laser, followed by computer analysis of the intensity image.

8.4.2 Affimetrix arrays (Genechip)

Total RNA is extracted from one cell population. Biotinylated cRNA is synthesized via coupled RT/IVT reaction. After fragmentation, cRNA is hybridized to microarrays, washed and stained with phycoerythrin (PE)-conjugated streptavidin, and subsequently scanned on a laser scanner. Affymetrix employs 11–20 different and sometimes overlapping 25-mer oligonucleotides per gene.

In addition to perfect-match (PM) oligonucleotides, Affymetrix GeneChips also contain mismatch (MM) oligonucleotides that serve as negative controls. The mismatch oligonucleotides carry a mutation at position 13 of the 25 mers.



Intensity images of perfect match (upper row of squares) and mismatch (lower row) are indicated. In cases where both perfect match and mismatch yield a strong signal, the contribution of that probe to the overall expression is ignored, because it is not specific.

The absolute expression value is calculated from the combined PM- MM differences of all the pairs in the probe set. The values can be directly compared to data for any other sample using the same probe sets.

8.4.3 Hybridization–based approaches limitations

These approaches rely upon existing knowledge about genome sequence (20% of total SNPs known). The background level is high owing to cross hybridization. The dynamic range of detection is limited due to signal saturation. Normalization methods are used to compare different experiments.

8.5 RNA-seq (Whole transcriptome shotgun sequencing WTSS)

RNA-seq is a technology that uses the capabilities of next- generation sequencing (NGS) to reveal RNA presence. It allows RNA quantification from a genome at a given moment in time. All transcripts (mRNA and ncRNA) can be analysed. Fusions and other structural variations (alternative intron splicing) are detected.

This technique has a high reproducibility and dynamic range (5 log ranges) and is not limited to genome annotation. It usually starts with a cDNA library. As reverse transcription introduces biases and artifacts single molecule real time (SMRT) sequencing methods have been developed.

RPKM-values “Reads per kilo base per million mapped reads” are introduced. They represent the amount of reads sequenced per sample. It was introduced because statististic needed to compare expression values between different samples / experiments. This values corrects for difference in sequencing depth and gene length.

$$RPKM = \frac{10^9 C}{N \cdot L}$$

with C the number of reads mapped to a gene, N the total mapped reads in the experiment and L the exon length in base-pairs for a gene.

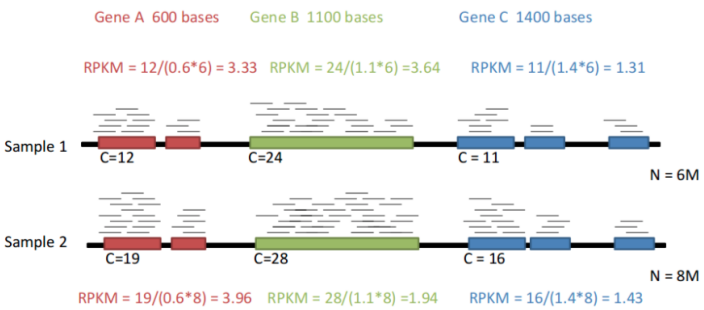
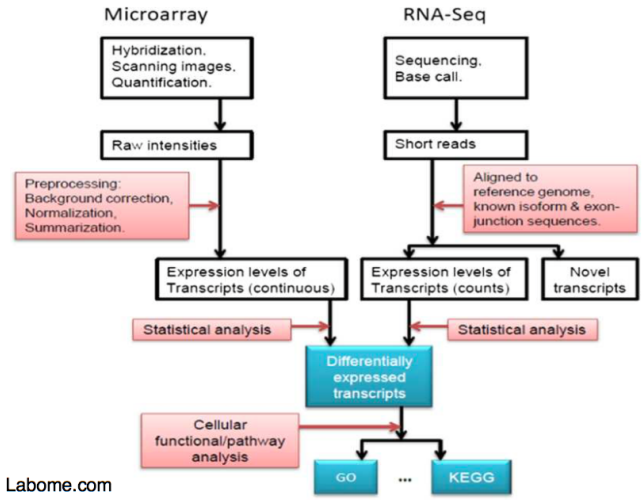


Figure 26: RPKM example

8.5.1 Microarrays vs. RNA-seq



8.6 Second Generation Sequencing technologies

1. Automated dideoxy method of Sanger (FGS)
2. Next-generation sequencing technologies
   - Template preparation (PCR amplification of single DNA versus cloned DNA in FGS)
   - Greater sequencing throughput
   - Imaging
3. Emphasis on Illumina and IonTorent
   - Need PCR/bridge amplification

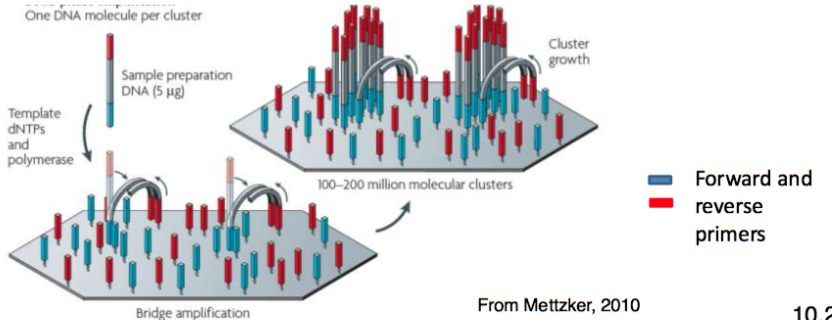
Next generation technologies are used for applications such as whole-genome assembly (metagenomics), variant detection (SNP, indels, copy number), Targeting resequencing (exons), ChIP-seq (DNA binding proteins, histone modification), expression profiling(RNA-seq splicing variants) and small RNA sequencing.

8.6.1 Illumina

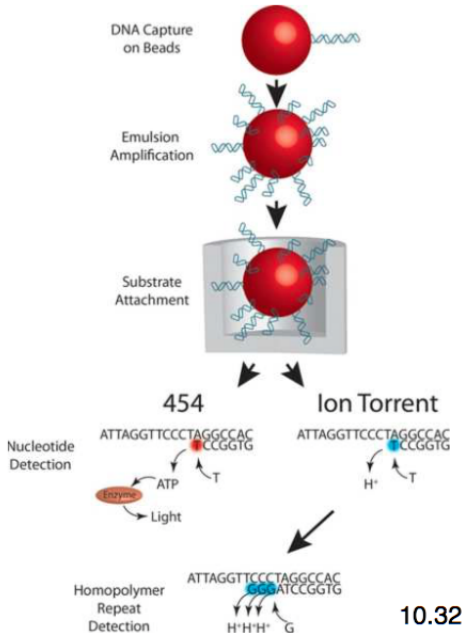
Template preparation

1. Genomic DNA is randomly fragmented and adapters are ligated to both ends
2. Solid-phase amplification of the template
   - Bind single-stranded (ss) fragments to the surface of the flow cell channels containing lawns of primers
   - Initial priming and extending of the single stranded molecule template
   - Bridge amplification of the immobilized template with immediately adjacent primers to form clusters

Cyclic reversible termination To initiate the first sequencing cycle, add all 4 labeled reversible terminators and DNA polymerase to the flow cell. After laser excitation, capture the image of emitted fluorescence (TIRF) from each cluster on the flow cell. Record the identity of the first base of each cluster. The blocked 3' terminators and the fluorophore are removed. The identity of each base of a cluster is read off from sequential images.



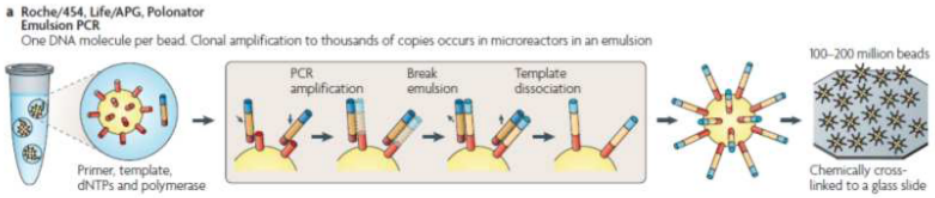
8.6.2 IonTorrent



Clonal amplification occurs on beads in an emulsion by emPCR. The beads are deposited into picoTiterPlate (PTP) wells. DNA sequencing using sequential addition of each dNTP in limited amounts. Extension is measured with a semiconductor apparatus, via the release of H+.

Analogous to pyrosquencing method Roche/454) = the PPi release produces light via a series of enzymatic reaction (luciferase/sulphurylase) Bioluminescence is imaged with a CCD camera.

Template preparation An oil-aqueous emulsion is created to encapsulate bead-DNA complexes into single aqueous droplets. PCR amplification occurs is performed at the surface of the beads via immobilised primers.



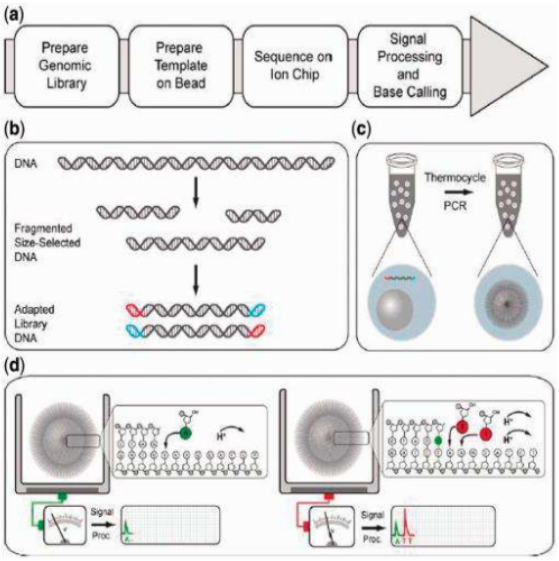
From Mettzker, 2010

Details of the CMOS semiconductor chip Single addition of nucleotide in limiting addition.

DNA polymerase extends the primer and pauses.

DNA synthesis is reinitiated following the addition of the next complementary dNTP.

The order and intensity of the intensity peaks are recorded as flowgrams



8.7 Third generation sequencing : single molecule real time sequencing

8.7.1 DNA synthesis based sequencing

A single DNA polymerase is affixed at the bottom of a ZMW (zero-mode waveguide on a chip. ZMW is a structure that creates an illuminated observation volume that is small enough to observe only a single nucleotide. Each of the four DNA bases is attached to one of 4 different fluorescent dyes. When a nucleotide is incorporated by the DNA polymerase the fluorescent tag is cleaved off and diffuses out the observation area of the ZMW where its fluorescence is no longer observable. A detector detects the fluorescence signal.

9 Large scale gene expression analysis

9.1 Introduction

In this chapter we will learn how modern genomic techniques allow us to monitor the internal activity of a cell in order to find the genes involved cellular processes, such as the cell cycle. This is a crucial part of both genome annotation and basic biology, revealing important clues into a gene's function. One technology in particular, known as a DNA microarray, lets us simultaneously measure the expression level of every gene in a cell. The expression level essentially tells us the number of mRNA transcripts that each gene is producing at a single point in time. Although the final objective of most experiments is to know the amount of each active protein in a cell – which we assume is an indicator of which processes are active in the cell – censusing proteins is much more difficult to do than censusing RNAs. As a consequence, biologists have turned to microarrays in the hope of better understanding cell function.

The basic idea behind all types of microarrays is the same. Complementary DNA sequences for every gene in a genome (the *probes*) are laid down in great quantity on individual spots on a glass slide or silicon chip. This is the microarray itself. Then the mRNAs from a cellular extract are washed over this array to allow them to find and stick to their complements. By counting the number of transcripts that bind to each spot, we can measure at least the relative abundance of each.

In order to count the number of transcripts bound to each spot,each transcript is labeled with a fluorescent dye; after hybridization a laser is pointed at each spot on the microarray so that the fluorescence level can be measured by a computerized system. The greater the number of mRNA molecules in a cell and therefore hybridizing, the greater the intensity of fluorescence at the spot containing the complementary probe.

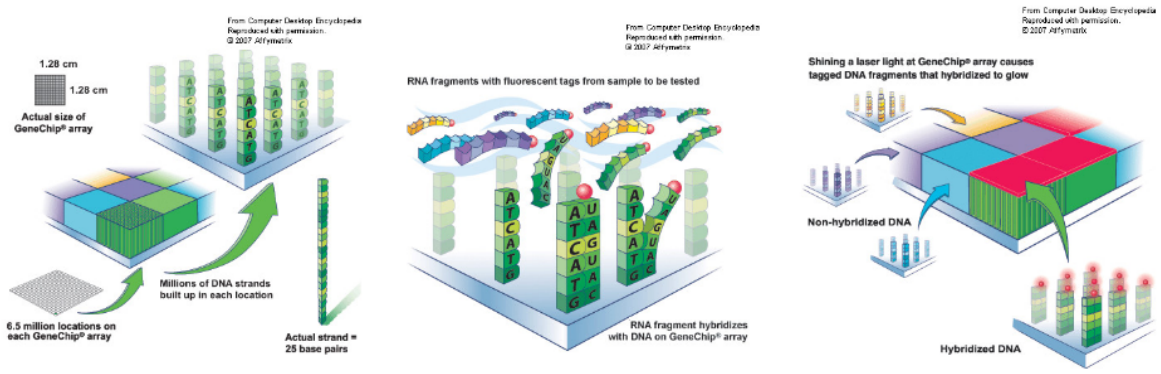


Figure 27: Example of DNA microarray: Affymetrix technology

There are **three main technologies** for producing and carrying out microarray experiments. These technologies differ in the amount of information about gene sequences needed to construct them, in the accuracy of the data they provide, and in the price of each experiment.

cDNA cDNA arrays are manufactured by spotting droplets of pre-made complementary DNAs on a glass slide. One of the great advantages of cDNA microarrays is that no prior knowledge of the gene sequences is needed in order to measure expression levels. A collection of mRNA transcripts can spotted as cDNAs on the array. RNA from new cellular extracts can be

hybridized to measure expression levels. If there are spots that show interesting patterns, then the DNA in that spot can be sequenced in order to find out what gene it represents. If it is not possible be sure that every spot has exactly the same amount of cDNA – even fluorescence intensities – then cDNA arrays are used by hybridizing two samples to each array. The two samples are labeled with different fluorescent dyes, that can be read separately by image-processing software. The data that result from cDNA array experiments are generally represented as the ratio of fluorescent intensities of the two samples, and not really as the absolute level of expression of genes.

Oligonucleotide arrays (two-dye) These methods use chemical or photolithographic technologies to synthesize oligonucleotides (short DNA sequences) on to the array. Each gene can then be represented on the array merely by synthesizing the appropriate sequence. These sequences are usually 25 to 80 bases long, depending on the exact technology used. Of course the synthesis of 6000 or more spots with specific oligonucleotides requires that we know the sequences a priori through genome annotation. Oligonucleotide arrays can either use one or two dyes (and therefore one or two samples per array). Two-dye arrays are conceptually similar to cDNA arrays, so we discuss these first.

Oligonucleotide arrays (one-dye) The most common single-dye oligonu- cleotide arrays are manufactured by the Affymetrix Corporation, and are some- times referred to by their commercial trademark name, GeneChip microarrays.

9.1.1 Examples

The following examples will illustrate the use of gene expression and the purpose of microarrays.

Diagnosis Biomarkers are used for early diagnosis of, for example, rheumatoid infections. The prediction problem selects multi-class features (e.g. for Rheumatoid arthritis, Lupus, Psoriatic rheumatismn Microcristalline arthritis, Inflammatory osteoarthritis).

Prognosis Biomarkers also predict the risk of allergies of newborns. More than 30% of the children are allergic in industrial countries. Predicting who is more likely to become allergic is a path to prevention and possible treatment.

Response to treatment prediction Gene profiling can be used for cancer treatment. The purpose of this is to identify biomarkers for predicting patient response to MAGE-A3 immunotherapy against melanoma before treatment.

9.1.2 Supervised selection

	gene 1	gene 2	...	gene p	response
sample 1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,p}$	y_1
...	...	...	...	...	...
sample n	$x_{n,1}$	$x_{n,2}$	...	$x_{n,p}$	y_n
test sample	x_1	x_2	...	x_p	?

Figure 28: Example of supervised selection

An schematic representation of supervised selection is depicted if Figure 28. The number p of input dimensions (probes, probesets or genes) may be very large, going from 10^4 to 10^6. The number n of samples is typically much smaller, around 50-100 samples.

Each samples is characterized by a vector x of p measurements. Each training sample has a known response, class labeled y (with $y \in \{-1, 1\}, y \in \mathbb{N}$ or $y \in \mathbb{R}$).

Gene selection The purpose of gene selection is to find a subset of genes (a.k.a features, attributes or input variables) in order to predict the response or class y of new samples.

Objectives

- Insight into the data and the predictive model
- Link between data analysis and medical expert
- Biological validation on a few genes rather than thousand ones
- Reduction of the financial cost of a diagnosis/prognosis kit (technological constraints)

Difficulties

- Measurements are noisy
- Gene expression varies due to many factors (gender, cell type, growth of the organism, chemical environment of the cell, ...) often not related to the response to be predicted
- Financial cost: 500...1,000 euros/experiment
- Small n (e.g. 50), large p (e.g. 50,000) problems

9.1.3 Unsupervised selection

	gene 1	gene 2	...	gene p	cluster
sample 1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,p}$	?
...	...	...	...	...	...
sample n	$x_{n,1}$	$x_{n,2}$	...	$x_{n,p}$	?
cluster	?	?	...	?	

Figure 29: Example of unsupervised selection

An schematic representation of unsupervised selection is depicted if Figure 29. The aim of unsupervised selection is to find clusters of genes and/or samples that share a similar profile: up or down regulated genes across the same samples.

9.2 Preprocessing

Once the data is measured, it needs preprocessing and sample normalization. The preprocessing step defines a single probeset expression level from the various probe intensities. This is possible with several techniques, the most common ones being MAS 5.0, RMA and GC-RMA.

There are three steps to preprocessing:

- background adjustment: optical noise correction, probe affinity adjustment (influenced by the GC content), RMA ignores the MM probes
- sample normalization: quantiles should be stable across samples, after conversion to log intensities for (GC-)RMA
- summarization: median polish

9.2.1 Feature normalization

Normalization makes sure that each gene (probeset) has roughly the same expression range across all samples.

	gene 1	gene 2	...	gene p
sample 1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,p}$
...	...	...	...	...
sample n	$x_{n,1}$	$x_{n,2}$	...	$x_{n,p}$

One possible method is Z-score normalization, which consist in replacing $x_{i,j}$ by $\frac{x_{i,j}-\mu_j}{s_j}$ with μ_j the mean level of expression of probeset j over the training samples and s_j its standard deviation.

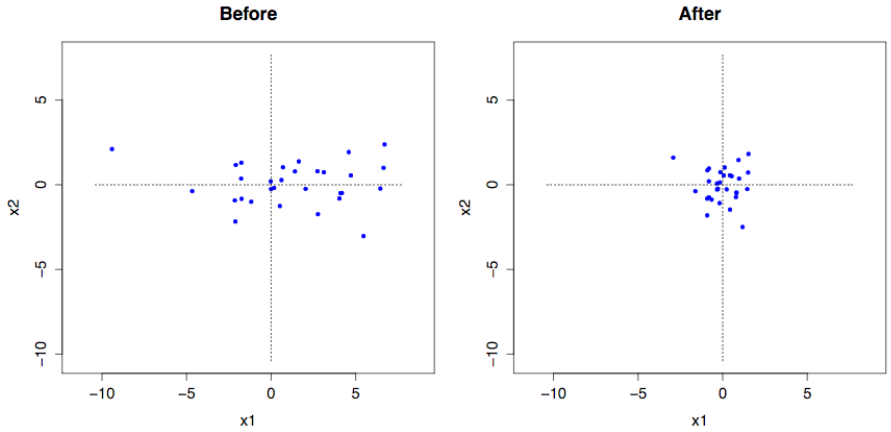


Figure 30: Example of feature normalization

9.2.2 Distance between expression values

The distance between expression values can be computed in two different ways

Euclidean distance: $d(x_1, x_2) = ||x_1 - x_2|| = \sqrt{\sum_{i=1}^{n}(x_{i,1} - x_{i,2})^2}$

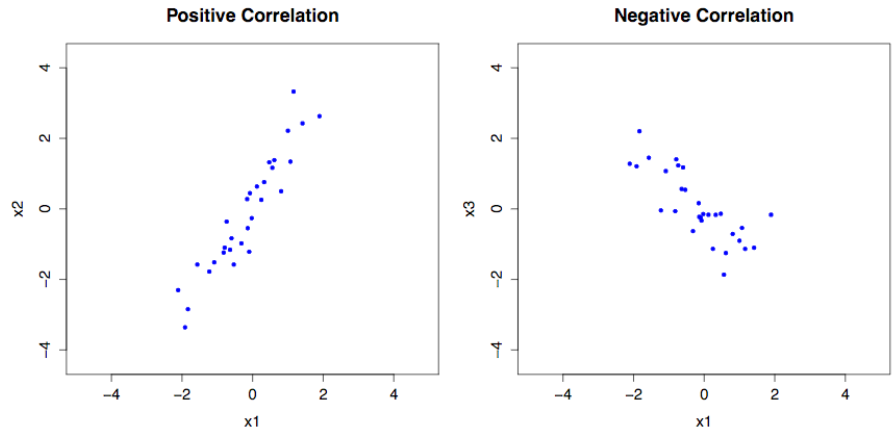
Correlation based distance: $d(x_1, x_2) = 1 - \frac{1}{2}(1 + corr(x_1, x_2))$

There are several ways to express the correlation between expression values. One of them is the Pearson Correlation.

Pearson Correlation For two random vectors (e.g.gene expression values) x_1, x_2 measured over n samples, the Pearson Correlation is defined as

$$corr(x_1, x_2) = \frac{\sum_{i=1}^{n}(x_{i,1} - \bar{x}_1)(x_{i,2} - \bar{x}_2)}{\sum_{i=1}^{n}(x_{i,1} - \bar{x}_1)^2 \sum_{i=1}^{n}(x_{i,2} - \bar{x}_2)^2}$$

If $corr(x_1, x_2) = \pm 1$, x_1 and x_2 are perfectly linearly correlated, positively or negatively depending on the sign. If both genes are over/under-expressed on the same samples, there will be a positive correlation. If one gene is over-expressed when the other is under-expressed, the correlation will be negative.



If $corr(x_1, x_2) = 0$, the genes are not linearly correlated. Finally, whenever x_1 and x_2 have a normalized to zero mean and a unit variance, the Pearson correlation becomes

$$corr(x_1, x_2) = \sum_{i=1}^{n} x_{i,1} x_{i,2} = x_1^T x_2$$

There are however some pitfalls with correlation measures. They are for example very sensitive to outliers. Besides, correlation only measures a linear dependence, while sometimes there might be a higher order dependence.

Spearman's rank correlation This correlation is less sensitive to outliers. The idea is to replace feature values by features value ranks across the observations. Then, the Pearson correlation is computed between the rank vectors. This is illustrated in Figure 31.

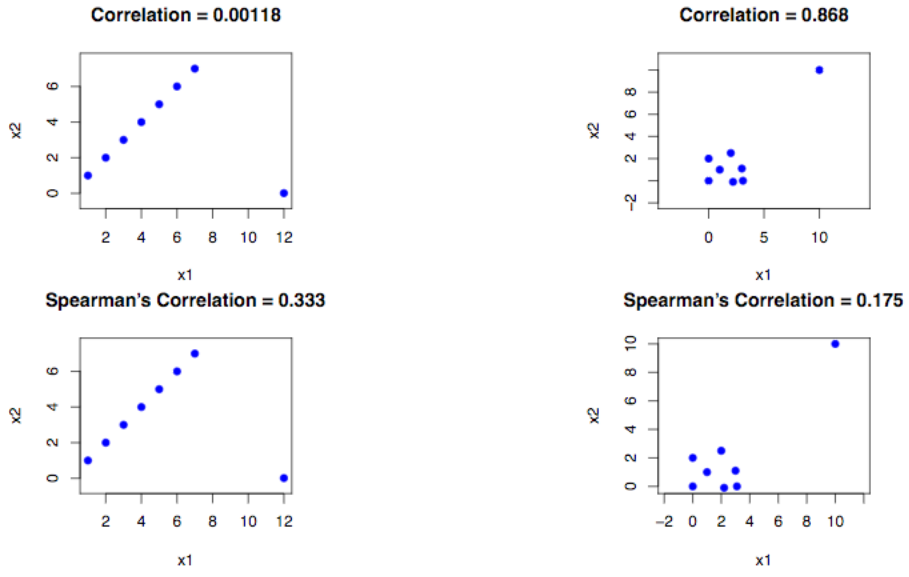


Figure 31: Spearman's correlation

Uncorrelated features When features are uncorrelated, meaning $corr(x_1, x_2) = 0$ (for both Pearson and Spearman), they are not necessarily independent. Take the example of Figure 32. The genes are uncorrelated but $P(x_2|x_1) \neq P(x_2)$.

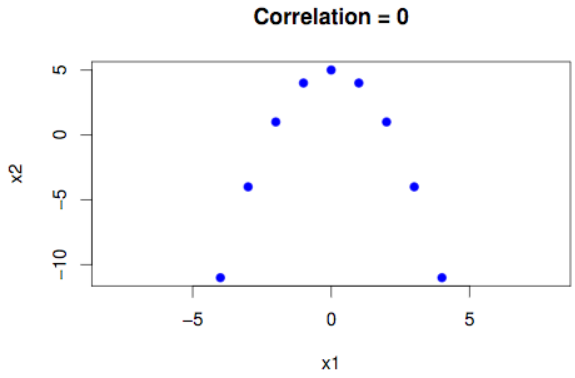


Figure 32: Uncorrelated genes

9.3 Unsupervised gene selection

Hierarchical clustering requires the user to specify a measure of dissimilarity between (disjoint) groups of observations, based on the pairwise dissimilarities among the observations in the two groups. As the name suggests, they produce hierarchical representations in which the clusters at each level of the hierarchy are created by merging clusters at the next lower level. At the lowest level, each cluster contains a single observation. At the highest level there is only one cluster containing all of the data.

Agglomerative clustering algorithms begin with every observation representing a singleton cluster, meaning at the terminal nodes. At each of the $N-1$ steps the closest two (least dissimilar) clusters are merged into a single cluster, producing one less cluster at the next higher level. Therefore, a measure of dissimilarity between two clusters (groups of observations) must be defined.

There are several ways to define the distance between two clusters D_i and D_j

- *Single-link or nearest neighbor rule:* it takes the intergroup dissimilarity to be that of the closest (least dissimilar) pair

$$distance(D_i, D_j, d) = \min_{\vec{x}\in D, \vec{y}\in D} d(\vec{x}, \vec{y})$$

  It will have a tendency to combine, at relatively low thresholds, observations linked by a series of close intermediate observations. This phenomenon, referred to as chaining, is often considered a defect of the method.

- *Complete-link or farthest neighbor rule:* it takes the intergroup dissimilarity to be that of the furthest (most dissimilar) pair

$$distance(D_i, D_j, d) = \max_{\vec{x}\in D, \vec{y}\in D} d(\vec{x}, \vec{y})$$

  It will tend to produce compact clusters with small diameters.

- *Average-link rule:* it uses the average dissimilarity between the groups

$$distance(D_i, D_j, d) = \frac{1}{|D_i| \cdot |D_j|} \sum_{\vec{x}\in D, \vec{y}\in D} d(\vec{x}, \vec{y})$$

  It represents a compromise between the two extremes of single and complete linkage and attempts to produce relatively compact clusters that are relatively far apart.

9.3.1 Agglomerative Hierarchical Clustering algorithm

Input D is a set of observations $\vec{x}_1 \dots \vec{x}_m$ and $d(\vec{x}, \vec{y})$ is a distance measure between observations.

Output A tree T of subsets of D

Steps

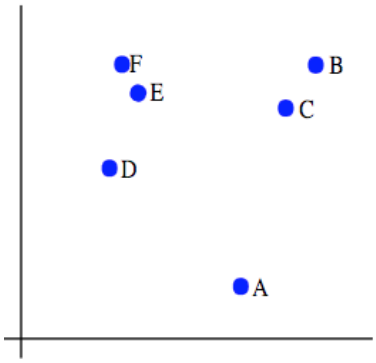
1. Initialize a set $\mathcal{D}$ of cluster $D_1 \dots D_m$

$$\begin{aligned} \mathcal{D} &\leftarrow \{\{\vec{x}_1\} \dots \{\vec{x}_m\}\} \quad \text{the initials clusters are tree leaves} \\ T &\leftarrow \quad \text{a partial tree whose leaves are the } \vec{x}_i \end{aligned}$$

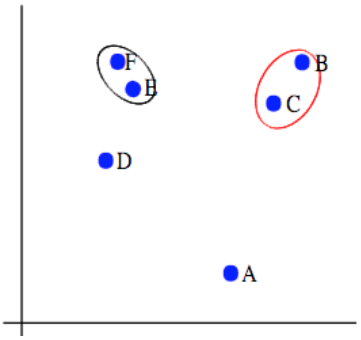
2. While $|\mathcal{D}| > 1$
   - Choose pair of clusters (D_i, D_j) in $\mathcal{D}$ such that $distance(D_i, D_j, d)$ is minimal
   - Define a new cluster $D_k = D_i \cup D_j$, such that $\mathcal{D} \leftarrow \mathcal{D} \cap D_k - \{D_i, D_j\}$
   - Add D_k as parent node of D_j and D_i in the tree T
3. Return T

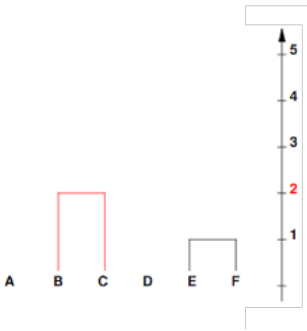
9.3.2 Example

This hierarchical clustering algorithm, with the average-link rule, is known as the UPGMA algorithm used in phylogeny. The observations are (fragments) of sequences representative of some species, called taxa. The pairwise distance measure is based on alignment scores, generally corrected by an evolutionary model (e.g. Kimura). The final tree is interpreted as a phylogenetic tree and the branch length as representative of time.

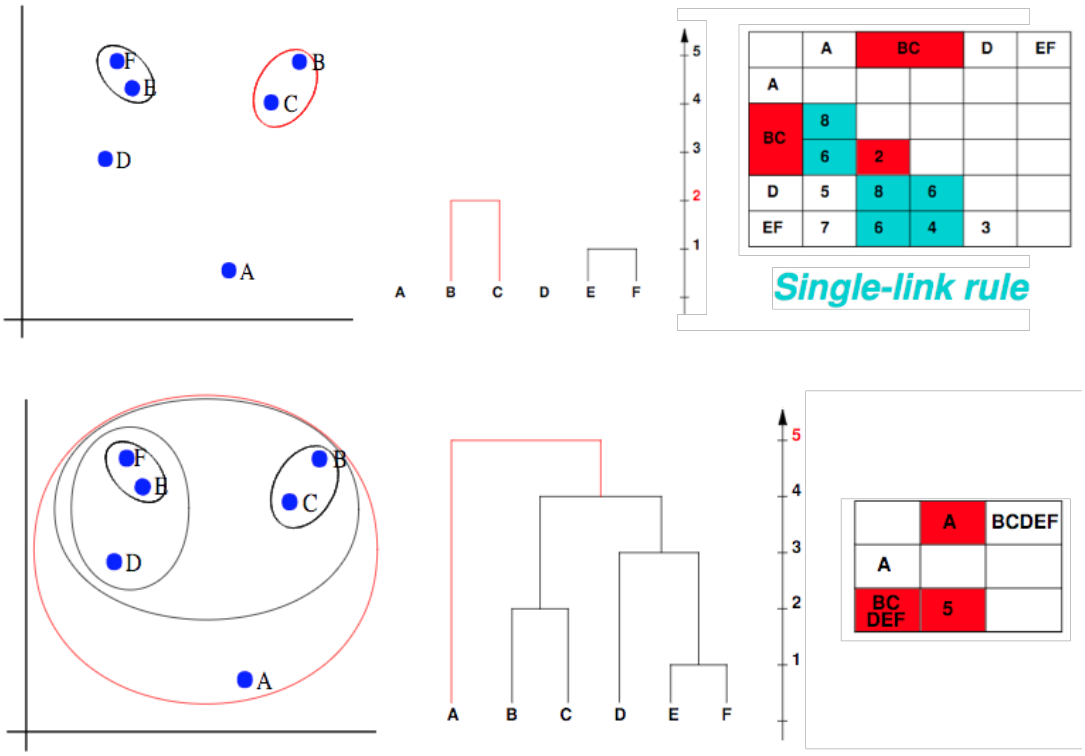


	A	B	C	D	E	F
A						
B	8					
C	6	2				
D	5	8	6			
E	7	6	4	3		
F	8	6	5	4	1	





	A	B	C	D	EF
A					
B	8				
C	6	2			
D	5	8	6		
EF	7	6	4	3	



9.4 Supervised gene selection

In the classical supervised learning task, a given training set of fixed-length feature vectors along with target values is used to derive a mathematical model that represents the mapping function between the feature vectors and the target values. The model is then used to predict the target for previously unseen instances. The problem of feature selection can be defined as finding relevant features among the original feature vector, with the purpose of increasing the accuracy of the resulting model or reducing the computational load associated with high dimensional problems.

In this section, binary classification will be discussed first: e.g. responders (+ or class 1) vs non-responders (- or class 2). Samples can be indexed by their class label; means and variances can be computed on samples of a given class. The aim is to find a subset of most discriminating genes for the prediction of the class of any new sample.

	gene 1	gene 2	...	gene p	class
sample 1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,p}$	+
sample 2	...	...	...	...	+
...	...	...	...	...	...
sample n-1	...	...	...	...	-
sample n	$x_{n,1}$	$x_{n,2}$	...	$x_{n,p}$	-

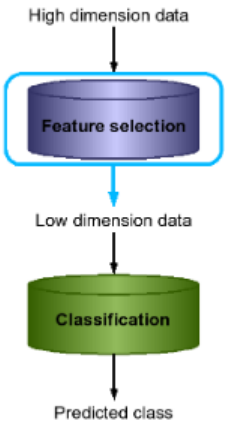
Figure 33: Example of supervised selection

Many approaches have been proposed for feature selection. In general, they can be categorized along two lines: filters and wrappers. Embedded methods are also available.

9.4.1 Filters

In filters, the feature selector is independent of a learning algorithm and serves as a pre-processing step to modeling. In other words, they attempt to assess the merits of features from the data, ignoring the effects of the selected feature subset on the performance of the learning algorithm.

Filters only use the training data and the class labels during the feature selection step. Standard techniques are based on fold changes, t-Test or mutual information. They train a single classifier, which takes the selected features as inputs. This is the simplest and less computing intensive approach.



Different types of filtering will be considered. They can be divided in two categories: nonspecific filtering and specific filtering. The first is a general procedure for removing genes that show little or no variability. The second is a selection process that is oriented towards finding genes that are associated with a particular phenotype of interest.

Non-specific filtering Non-specific filtering is filtering of the probes without regard to a classification or clustering objective. The data analyst wants to remove those features which have no chance of being predictive, regardless of the prediction problem.

On way to perform non-specific filtering is to keep only the genes (e.g. 25 %) with the larger variances, before normalizing the data to unit variance. Indeed, genes with a small variance across all training samples are unlikely to be discriminating between classes.

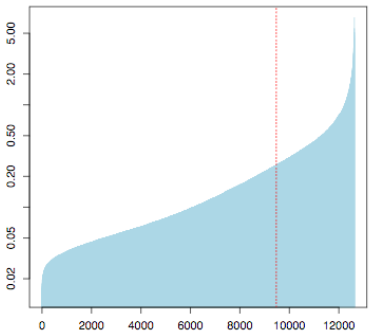


Figure 34: Example of non-specific filtering

Fold changes Fold changes is a type of specific filtering. Fold change is a measure describing how much a quantity changes going from an initial to a final value. In this approach, only the genes with the larger fold changes between both conditions are selected.

$$\frac{\bar{x_1}}{\bar{x_2}} \quad \text{or} \quad \log \frac{\bar{x_1}}{\bar{x_2}} = \log \bar{x_1} - \log \bar{x_2} \quad \text{or} \quad \bar{x_1} - \bar{x_2}$$

Whenever $\bar{x_1} < \bar{x_2}$, one considers a small value as important. This means $1 \leq \frac{\bar{x_1}}{\bar{x_2}} \leq -1$.

The significance of a fold change depends on the measurement technology and on the class conditional variance.

t-Test A feature relevance $J(x)$ can be defined according to the distance between the average feature value in each class. The larger the distance the better, relatively to standard deviations.

Welch t-statistics For the example in Figure 33, with n_1 (resp. n_2) the number of examples labeled as $+$(resp. $-$), the feature relevance is given by

$$J(x) = \frac{\bar{x_1} - \bar{x_2}}{\sqrt{S_1^2/n_1 + S_2^2/n2}}$$

The estimated variance of each given class is expressed as $S_i^2 = \frac{1}{n_i - 1} \sum_{j=1}^{n_j} (x_{ij} - \bar{x_i})^2$.

Confidence measure The Welch statistics follows a t-distribution with a number of degrees of freedom equal to

$$\frac{S_1^2/n_1 + S_2^2/n_2}{(S_1^2/n_1)(n_1 - 1) + (S_2^2/n_2)(n_2 - 1)}$$

p-values assess the significance of the difference between the two class means. A feature is selected if its associated p -value is below a prescribed threshold (e.g. 5%$\Rightarrow J(x) \geq 2\ 228$ when d.f.=10). an efficient way for computing p-value is to use the R project. For example, the `t.test(x1,x2)` function takes as input `x1` and `x2`, vectors of expression values of a given gene from samples labeled as, respectively, class 1 and class 2. If p-value > 0.05, then the difference between the two class means is not considered significant for this feature, and the feature is discarded.

Alternatives to a simple t-Test There a several alternatives. For example, Mann-Whitney rank test is an alternative non-parametric test. ANOVA offers a generalization of the t-Test in a multi-class (> 2) setting. Pairwise t-tests between one class and the others is a common alternative. Finally, Kruskal-Wallis is a generalization of Mann-Whitney to multi-class.

The Multiple test problem Imagine a microarray experiment to distinguish between patients with a positive or a negative diagnosis. Among 50,000 gene expression values measured in each experiment, only those genes that are differently expressed, with a p-value $\leq 0\ 05 = \alpha$, are selected.

A type I error of a statistical test might occur. This means the statistical test concludes that the mean expression values among the two classes are significantly different for a given gene while they are not. The feature is falsely selected with a probability α. As a statistical test is performed for each gene, this means a repetition of 50,000 times from the same experiment. If $\alpha = 0\ 05$, we expect to select wrongly $50,000 \times 0\ 05 = 2,500$ genes.

Multiple test correction In order to avoid this error, a correction is introduced for multiple tests. There a two possible correction schemes

- Bonferroni correction: the critical value (e.g. $\alpha = 0\ 05$) is divided by the number of performed tests n_t. For the example used previously, this means that only genes with associates p-value $\leq \frac{0\ 05}{50{,}000} = 10^{-6}$ are selected. This correction is conservative and often leads to select no feature.
- False Discovery Rate (FDR) correction , also called Benjamin-Hochberg correction: the steps of this correction scheme are
  1. Select a confidence level α (e.g. 0.05)
  2. Rank the p-values (one for each feature) in increasing order $p_1 \leq p_2 \leq \ldots \leq p_{n_t}$
  3. Iterate over the n_t features[2] and find the maximal index i such that the p-value $p_i < \frac{i \cdot \alpha}{n_t}$
  4. Keep all features up to index i_{max}

  Note that is $p_{n_t} < 0\ 05$, FDR correction leads to the selection of all features.

FRD correction is equivalent to Bonferonni correction whenever a single feature is selected, meaning $p_1 < \alpha/n_t$. Those corrections do not change the relative ranking of features, just the selection threshold. The used R function to achieve these correction is function `p.adjust`.

Mutual information Mutual information criterion is derived based on information theory to measure the amount of information gained for Y given X.

$$I(X;Y) = -\sum_{ij} P(x_i, y_j) \log_2 \frac{P(x_i, y_j)}{P(x_i)P(y_j)} = -\sum_{ij} P(x_i, y_j) \log_2 \frac{P(y_j|x_i)}{P(y_j)}$$

A feature X is more relevant if its mutual information with the class value is higher. If X tends to bring no information to predict Y then $P(y_j|x_i) \approx P(y_j)$ and $I(X;Y) \to 0$.

$I(X;Y) = 0$ if and only if X and Y are independent. Finally, $I(X,Y)$ is invariant under rescaling of the variables X and Y (as they are often rescaled to unit variance).

Multivariate filters Correlation measures, t-Test (ANOVA), and $I(X:Y)$ are univariate filters. Mutual information can be used to select several variables at a time $I(X_1, \ldots X_k; Y)$ but the mutual information depends on the distribution $P(X_1, \ldots, X_k, Y)$, $P(X_1, \ldots X_k)$ and $P(Y)$, which all need to be reliably estimated. The joint problem is replaced by an approximation with a greedy selection of features.

Maximal relevance minimum redundancy

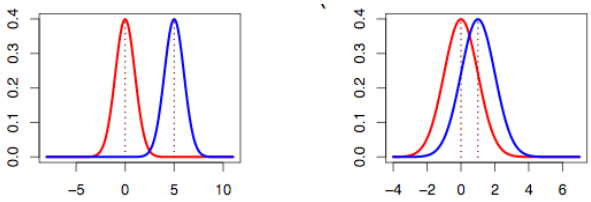
1. Select the feature with maximum mutual information with the response
   - $\hat{X} = argmax_X I(X:Y)$
   - $\phi = \{\hat{X}\}$ (initializes the set of selected features)
   - $F = \{X_1, \ldots, X_p\}/\{\hat{X}\}$ (the remaining set of features
2. repeat until an appropriate number of features are selected
   - $\hat{X} = argmax_{X \in F} \left[\underbrace{I(X;Y)}_{\text{maximize relevance}} \overbrace{-\frac{1}{|\phi|} \sum_{X_j \in \phi} I(X;X_j)}^{\text{minimize redundancy}} \right]$
   - $\phi \leftarrow \phi \cup \hat{X}$
   - $F \leftarrow F\{\hat{X}\}$

[2] $n_t = p$ with data in $\mathbb{R}^p$, not to be confused with p-value

Filters in a nutshell Filters offer interesting baselines which are fast to compute. Popular univariate filters are based on a t-Test (with multiple test correction) or mutual information. However, they ignore the interactions between genes. Maximum relevance minimum redundancy is a popular multivariate extension.

The two step approach is sometimes considered as a benefit since the features are claimed to be selected independently from the subsequent classifier/regression mode.

t-Test revisited A feature x is selected whenever the difference between the class means is significant (after correction for multiplicity). The expression for the feature relevance is still the same. Equivalently, one easily discriminates between the classes using two uni-dimensional Gaussian classifiers (with a common variance).



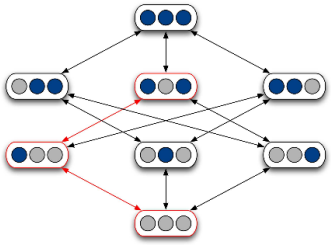
9.4.2 Wrappers

This approach searches through the space of feature subsets using the estimated accuracy from a learning algorithm as the measure of the goodness for a particular feature subset. A classifier is trained on several subsets of all possible features. However, an exhaustive evaluation of all possible subsets is unfeasible due to a the time complexity ($\mathcal{O}(2^p)$ with $p > 10000$). Typical solution to solve this problem are the use of feature ranking of forward/backward selection in order to estimate a classifier from a given subset of all possible features. The selected feature set will be the one that optimizes the performance of the classifiers (usually on an independent validation set). Feature selection depends on the evaluation protocol of the classifier.

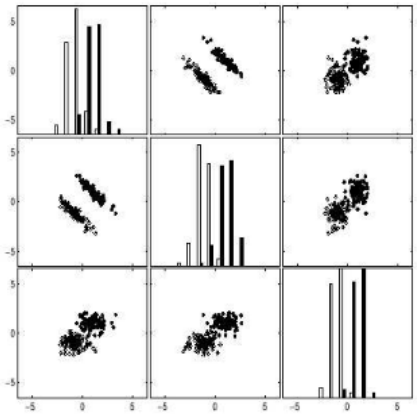
This method is restricted by the time complexity of the learning algorithm, and when the number of features is large, it may become prohibitively expensive to run.

Univariate feature ranking A common example of univariate feature ranking is hill climbing: keep adding features one at a time until no further improvement can be achieved ("forward greedy wrapping"). Alternatively we can start with the full set of predictors and keep removing features one at a time until no further improvement can be achieved ("backward greedy wrapping").

Multivariate Forward/Backward feature ranking Both phases (adding and removing) can be interlaced either in forward or backward wrapping. Forward selection goes bottom-up while Backward selection goes top-down.



Search order matters Imagine three features x_1, x_2 and x_3. The search order matters: x_3 alone is better that x_1 or x_2 alone, but x_1 together with x_2 offer the best discrimination. Depending on the order of the ranking, the selected features will be different.



2 best features:

- Univariate feature ranking selects (x_3, x_2)
- Forward selection selects (x_3, x_1)
- Backward selection selects (x_1, x_2)

Single best feature:

- Univariate feature ranking or Forward selection selects x_3
- Backward selection selects x_2

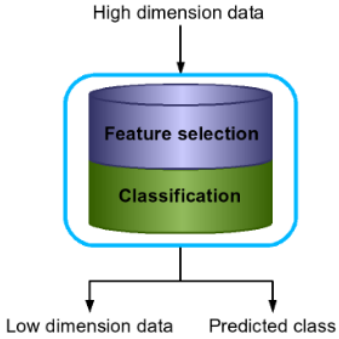
Wrappers in a nutshell A wrapper with univariate feature ranking offers a good baseline. The t-Test statistics can be used to rank features only (no need for multiple test correction nor fixing a confidence measure). Classifier performance is used to decide how many features to keep.This can outperform a pure filter approach while not increasing much the computational cost.

A backward selection may be preferable over a forward selection, but should not be used to select just a few features (what "a few" means depends on the data...). More sophisticated search strategies are possible (backward + forward, randomized search, ...).

If one can afford the computational cost of a multivariate selection, one should probably consider an embedded approach.

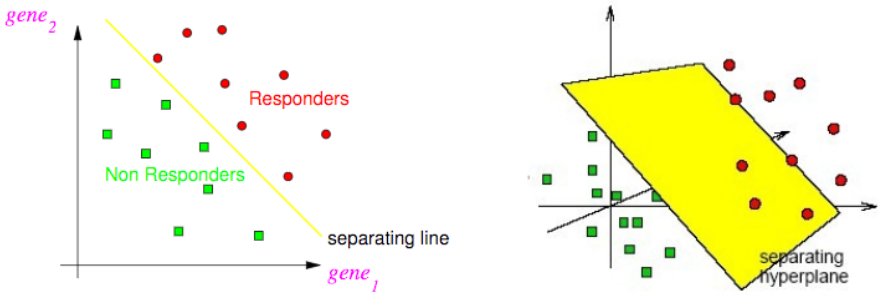
9.4.3 Embedded Methods

Embedded methods have built-in feature selection. In other words, theses methods define the feature selection and the classifier estimation as a combined optimization process. There is a classifier optimization included in the feature selection process. The features are selected as a by-product of the estimated classifier and its parameters.



This approach is more elegant and relevant but also more computing intensive than a filter.

Linear Discriminants The actual number of dimensions may be $\approx$ 10,000 for microarray classification. The linear discriminant is represented by a hyperplane in $\mathbb{R}^{\geq 10,000}$.



The decision rule is defined as followed

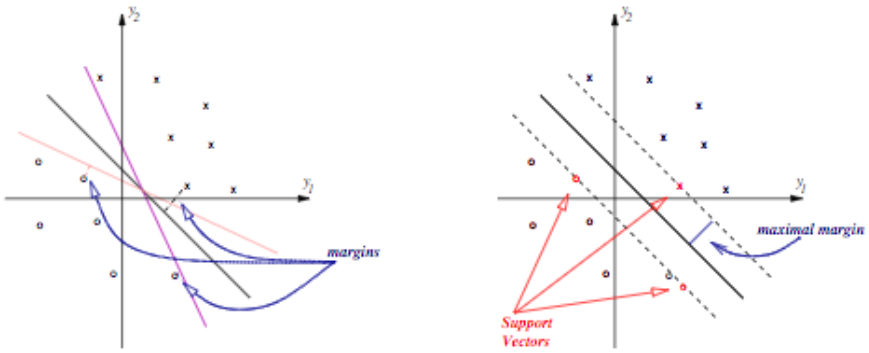
$$sign(\sum_{j=1}^{p} w_j x_j + w_0) = sign(\mathbf{w}^T \mathbf{x}$$

whit $x_0 = 1$ by definition and $|w_j|$ dictates the importance of the jth measure.

Linear Separability The data is linearly separable if the two classes can be perfectly separated by a hyperplane. A hyperplane in $\mathbb{R}^{10,000}$ can separate perfectly at least 10,001 (unaligned) points, given any possible two class labeling.

There is no problem to find a perfect linear separator of less than 100 points in $\mathbb{R}^{\geq 10,000}$ (e.g. for microarray data). The problem is that there are many apparently perfect models.

Linear Support Vector Machines (SVM) When the data is linearly separable the separating hyperplane is not unique but the maximal margin hyperplane separates the data with the largest margin. For each separating hyperplane, there is an associated set of support vectors.



Recursive Feature Elimination (RFE) Recursive feature elimination is based on the idea to repeatedly construct a model (for example an SVM or a regression model) and choose either the best or worst performing feature (for example based on coefficients), setting the feature aside and then repeating the process with the rest of the features. This process is applied until all features in the dataset are exhausted. Features are then ranked according to when they were eliminated. As such, it is a greedy optimization for finding the best performing subset of features.

Embedded Backward Selection

1. Estimate a SVM on a given set of dimensions (initially p dimensions). The decision rule is $sign(\sum_{j=1}^{p} w_j x_j + w_0)$.
2. Consider $|w_j|$ as the relevance of the jth dimension
3. Remove the least relevant dimension(s)
4. Iterate 1. to 3. on a reduced feature set

9.5 General Summary

Feature selection aims at reducing the dimensionality of the data while preserving the interpretation of the original features

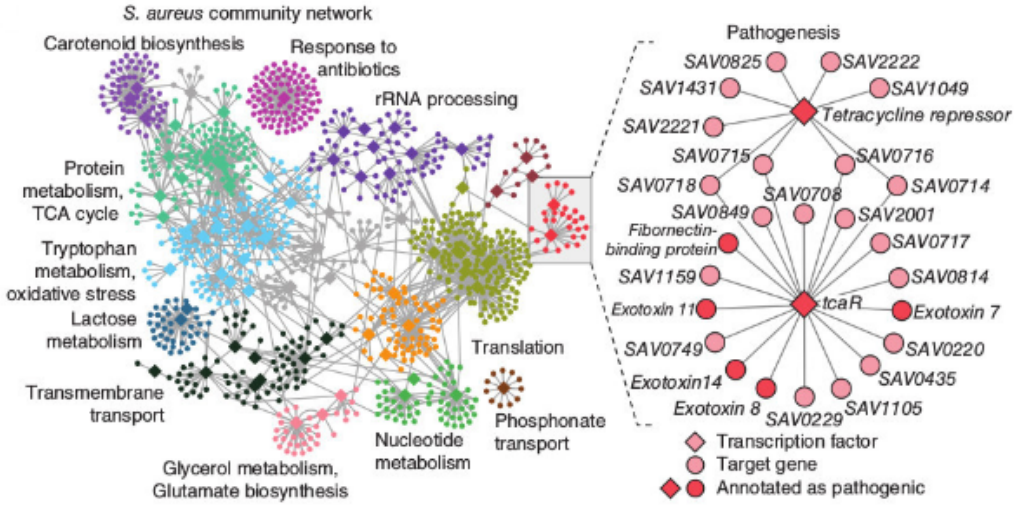
Filters methods use only the data + class labels. They are simple, fast, generally univariate (often an implicit use of a classifier).

Wrappers take the performance of the classifier into account. They are multivariate as soon as the classifier is multivariate. However, they are often computing intensive.

Finally, **embedded methods** take the structure of the classifier into account. They are more elegant and often faster than wrappers, not always better in terms of performance. They define a way to get an insight into a black-box classifier.

10 Interference to gene regulatory networks

A gene regulatory network (GRN) is a collection of molecular regulators that interact with each other and with other substances in the cell to govern the gene expression levels of mRNA and proteins. In other words, a GRN is the set of all regulatory transcription interactions in a cell.



It is typically represented as a graph with edges connecting. Genes can be viewed as nodes in the network, with input being proteins such as transcription factors, and outputs being the level of gene expression. The transcription factors (TF) can be seen as regulators while the target gene (TG) is the regulated gene. TFs are usually knows, TGs are at least partly unknown. Otherwise, each gene is considered as potential TF or TG.

Network inference, which is the reconstruction of biological networks from high-throughput data, can provide valuable information about the regulation of gene expression in cells. However, it is an underdetermined problem, as the number of interactions that can be inferred exceeds the number of independent measurements. Different state-of-the-art tools for network inference use specific assumptions and simplifications to deal with underdetermination, and these influence the inferences. The outcome of network inference therefore varies between tools and can be highly complementary.

10.1 Module-based versus direct inference

Exploiting the concept of modularity offers advantages from both the biological and the statistical points of view. Most module-based approaches not only predict regulatory interactions, but also identify the experimental conditions under which the predicted interactions take place. Assuming that modularity exists also contributes to the statistical robustness of the inferred interactions: each of the co-expressed genes in a module confirm the data for the other genes in that module by providing evidence for a certain regulatory programme, whereas for direct methods the evidence for a particular regulator–target interaction is based on only a single-gene observation.

10.1.1 Module-based interference

The network consists of overlapping modules of functionally related genes. Genes belonging to the same module act in concert under certain environmental cues, explaining their coordinated expression behavior. Modules are identified by methods that rely on clustering or biclustering. Module-based network inference procedures assign a regulatory program to these modules, rather than assigning an individual program to each single gene. This drastically lowers the number of interactions that must be evaluated during the inference process.

In other words, module-based interference

- searches for clusters (= modules) of genes that exhibit a similar expression behavior
- infers the transcriptional program of each module; it is better at finding interactions between regulators and targets with less similar expression profiles

10.1.2 Direct inference

Direct inference methods consider all genes on an individual basis. Most direct methods simplify the problem by assigning regulators to their target genes one by one and composing the combinatorial regulatory programmes in a post-processing step that finds sets of regulators which control the same target genes.

In other words, direct inference

- looks for the regulators of each gene; it is better at finding the target(s) of regulators dedicated to one or a few genes

10.2 Expression-based versus integrative inference

10.2.1 Expression-based inference

Non-integrative expression-based network inference methods extract information about regulator–target interactions from the expression data itself. Most methods restricts the interactions that can be inferred to those of regulators that are either co-expressed or inversely correlated with their targets.

In other words, expression-based inference

- takes gene expression data as sole input
- is useful for organisms for which there is little regulatory information available

10.2.2 Integrative inference

By complementing gene expression with additional transcriptional information (such as motif data or DNA–protein interaction data), integrative network inference methods can extend the scope of their predictions beyond interactions that can be inferred from co-expression behavior and usually result in more reliable predictions.

In other words, integrative inference

- complements gene expression with additional information: motif data or protein-DNA interaction data
- is more likely to predict true positive interactions
- the added value depends on the quality of the additional information

10.3 Global versus query-driven inference

The output of the inference methods can be global, indicating that they search for global patterns in the data, or query-driven, starting from a predefined set of core genes or core pathways and expanding on those. Most of the available programs can be used in either a query-driven or a global mode.

10.3.1 Global inference

Global module inference methods search for the modules that explain most of the data. This usually corresponds to identifying large pathways that consist of many genes and that are usually responsible for the general responses to major metabolic or condition shifts, such as the pathways that regulate flagellar synthesis, amino acids biosynthesis and the DNA damage response.

In other words, global inference

- provides a general view of the active GRN
- identifies large pathways that consist of many genes

10.3.2 Query-driven inference

Query-driven module detection methods, on the other hand search for genes that are co-expressed, in a condition-dependent way, with a predefined set of genes (also called query genes). These algorithms are deliberately biased towards finding a specific local solution in the search space according to the particular interests of the user.

In other words, query-driven inference

- search for genes that are co-expressed, in a condition-dependent way, with a predefined set of query genes
- finds local regulatory interactions

10.4 Supervised versus unsupervised inference

Supervised and semi-supervised methods treat the inference problem as a classification problem, whereas unsupervised methods do not.

10.4.1 Supervised inference

Supervised methods treat inference as a classification problem. They start from a set of known TF–target interactions and, on the basis of this predefined training set, characteristic features are derived, such as TF binding sites or the degree of co-expression between TF targets. These characteristics are subsequently used to evaluate a new candidate gene as a potential target of a TF. Genes that share many characteristics with the known targets of the TF are classified as true targets, and the others as non-targets. Such a classification strategy depends on the quality of the training set of true-positive and true-negative interactions.

In other words, supervised inference

- treats network inference as multiple classification problems: which target genes are regulated by a given transcription factor?

- requires a training set of known interactions between TFs and TGs and expression data
- each learned classifier is then used to predict whether further genes should be consider as target of a given TF

10.4.2 Unsupervised inference

To infer interactions in less studied organisms, unsupervised approaches are more suitable as they do not necessarily depend on previously known information and they can also infer interactions for regulators for which there is little or no prior knowledge.

In unsupervised inference, there is no such training set is available.

10.5 LASSO-based inference

The feature selection problem is solved by using a Lasso sparse regression approach. Lasso (least absolute shrinkage and selection operator) is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces.

	gene 1	gene 2	...	gene p	cluster
sample 1	$x_{1,1}$	$x_{1,2}$	...	$x_{1,p}$	?
...	...	...	...	...	...
sample n	$x_{n,1}$	$x_{n,2}$	...	$x_{n,p}$	?
cluster	?	?	...	?	

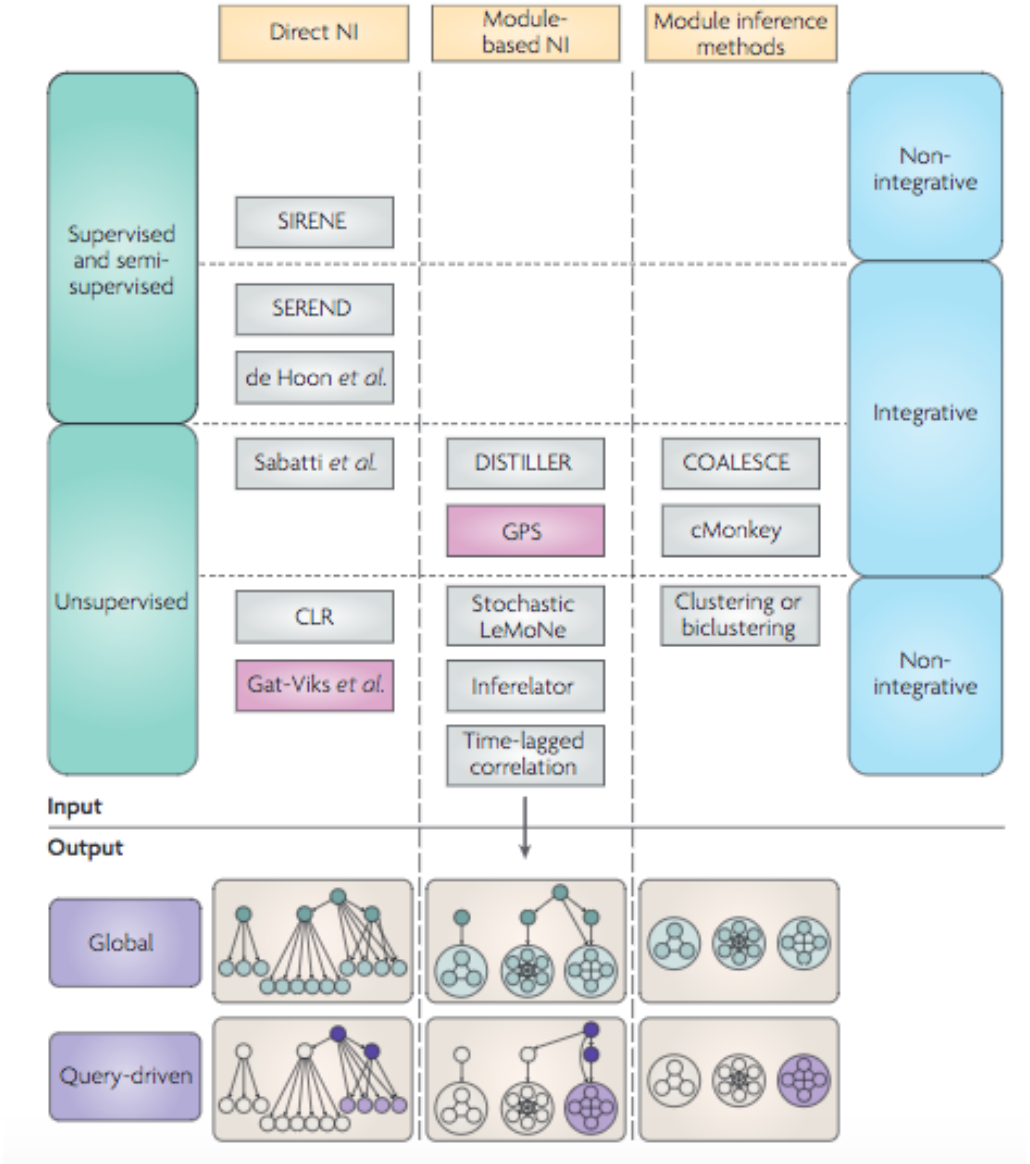
In short, a linear model was estimated in order to predict the expression of the target gene from the expression of the candidate regulators. The Lasso procedure led to a sparse linear model, i.e., to a linear model based only on a few transcription factors. The transcription factors selected by Lasso are therefore good candidates to regulate the target gene

- Each gene G_i is separately regressed on the TFs (a known subset of columns).
- Linear regression with L1-regularization

$$\hat{\mathbf{w}} = argmin_{\mathbf{w}} ||\mathbf{x}_{i*} - \mathbf{X}_{TF}\mathbf{w}||_2^2 + \lambda ||\mathbf{w}||_1$$

- For a specific $\hat{\mathbf{w}}$, $\hat{\mathbf{w}} \neq 0$ means dependence $TF_j \rightarrow G_i$.
- Direct, non-integrative, global and unsupervised inference can be used: expression data + list of known TFs is the only required information.

10.6 Taxonomy of GNR inference


Direct NI
Module-based NI
Module inference methods
Supervised and semi-supervised
Unsupervised
Non-integrative
Integrative
Non-integrative
SIRENE
SEREND
de Hoon et al.
Sabatti et al.
DISTILLER
GPS
COALESCE
cMonkey
CLR
Gat-Viks et al.
Stochastic LeMoNe
Inferelator
Time-lagged correlation
Clustering or biclustering
Input
Output
Global
Query-driven

11 Molecular phylogeneitics

11.1 Definition and concepts

Phylogeny is the inference of evolutionary relationships among biological species. Molecular phylogenetics is the use of molecular sequences to construct evolutionary trees. The evolution of proteins and nucleic acids is much more regular than for genomes. Sequence analysis is more amenable to quantitative treatments and statistical evaluation.

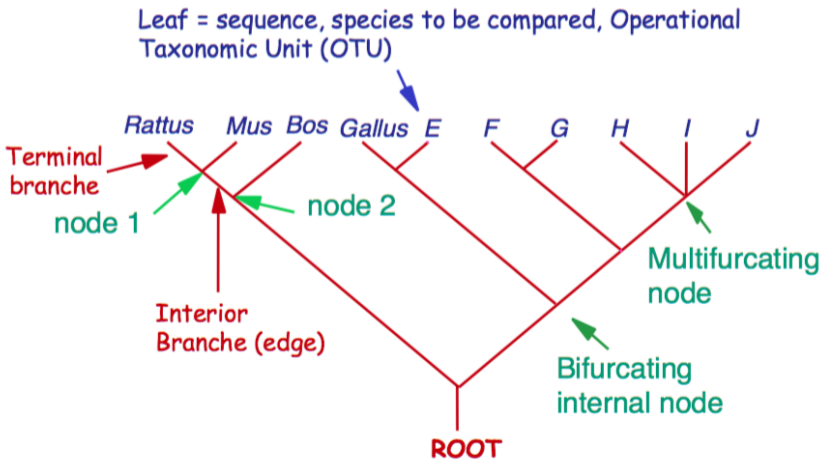
At the molecular level, evolution is a process of random drift of neutral mutations ($\neq$ from phenotypic evolution and natural selection).

11.1.1 Phylogenetic tree

A phylogenetic tree is a branching diagram that shows:
- changes in genes and proteins,
- the order these sequences diverged from one another,
- the orthologs and paralogs within a gene family

It is a two-dimensional graph showing ancestral relationships among homologous genes. If for example, two sequences are assumed to have evolved from a common ancestor, this means that some residues of the ancestor sequence can be inferred from conserved positions.

A phylogenitic tree is made of tree components: (internal) nodes, which represent the common ancestor taxonomic unit (taxon); leaves, which are the sequences or species to be compared (terminal nodes); and the branches. The length of the branches sometimes gives information about the number of mutation between two nodes.



A tree is qualified as dichotomic if there are two *Operational Taxonomic Units (OTU)* at the end of each branch.

A phylogenetic tree is also defined as either rooted or unrooted. In a rooted tree a special internal node is defined, called the root. The root node is the common ancestor to all the other nodes in the tree and all evolutionary paths lead back to the root. The branches of a rooted tree have an orientation going from the root to each external node. Unrooted trees, on the other hand, are un-oriented; they show the topological relationships among taxa, but are agnostic with respect to the identity of the common ancestor of all taxa.

There are more rooted trees ($\sim 3\ 4 \cdot 10^7$) than unrooted ones ($\sim 2 \cdot 10^6$). Most phylogenetic methods produce unrooted trees. However, some methods may consider every possible tree. It is easier to work with methods that do not consider rooted trees.

11.1.2 The four steps of molecular phylogenetics

1. Selection of sequences to be analysed
2. Multiple sequence alignment (overall similarity vs. evolved character)
3. Tree building (with distance matrix or character state method)
4. Tree evolution (bootstrapping).

11.1.3 Phenetics vs. cladistics methods

Within molecular phylogenetics, there are different approaches, like phenetics and cladistics methods.

Phenetics are base on quantitative global dissimilarities. They do not directly consider evolutionary history. The inferred tree is called a phenogram. UPGMA is a typical phenetic method.

Cladistics methods are based on common ancestry and evolutionnary history. They use subsets of characters/sites and like the evolved one to the ancestral one. To build the tree, these methods consider shared characters that come from the last common ancestor and are absent in more distant ancestors. The cladogram focuses more on the tree topology, meanning branch lengths are meaningless. Parsimony and maximum likelihood are examples of cladistics methods.

UPGMA A rooted tree is build from a distance matrix. The mean distance measured along the tree to the sequences on one side of the root is equal to the mean distance to the sequences on the other sides = *midpoint method*; hypothesis of a molecular clock or similar rate of evolution.

Parsimony with morphological characters A character alone does not say anything about the branching order within the "haves" and the "have-nots". A large set of characters that evolved at different point since time is needed. The sums overall characters is called the *tree length*. The most parsimonious trees with the minimum tree length are selected = Maximum parsimony.

11.2 Algorithms

11.2.1 Maximum parsimony

Character-state approach In this approach, we define a sets of binary character states, 0 and 1, where 0 represents the ancestral state and 1 represents a derived state. The aim is to find the tree giving the simplest possible explanation of the data. In Figure 35, the parsimony criterion says that tree (a) is to be preferred. Note that all trees in this example are unrooted

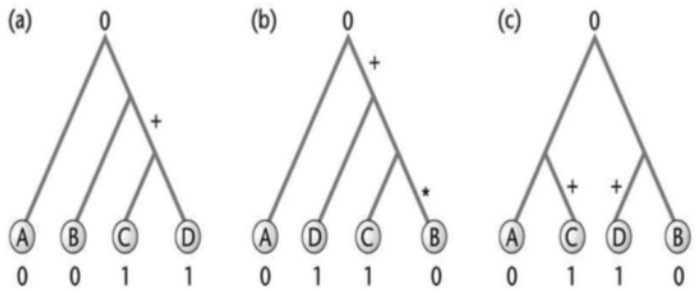
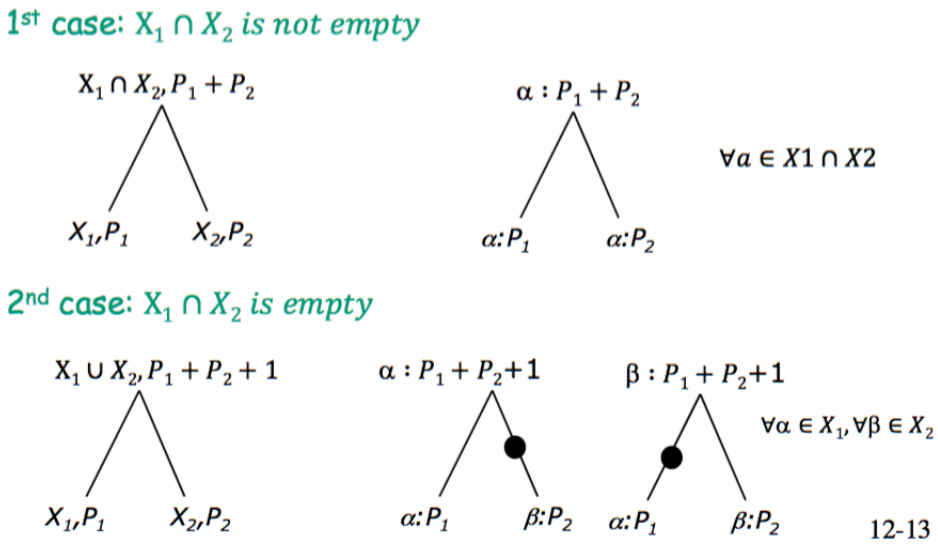


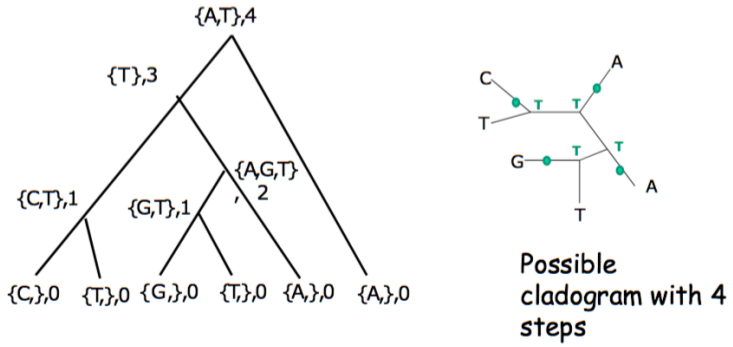
Figure 35: Character-state approach

Fitch algorithm The aim of this approach is to compute the minimal number of changes that are consistent with all the leaves of a tree.

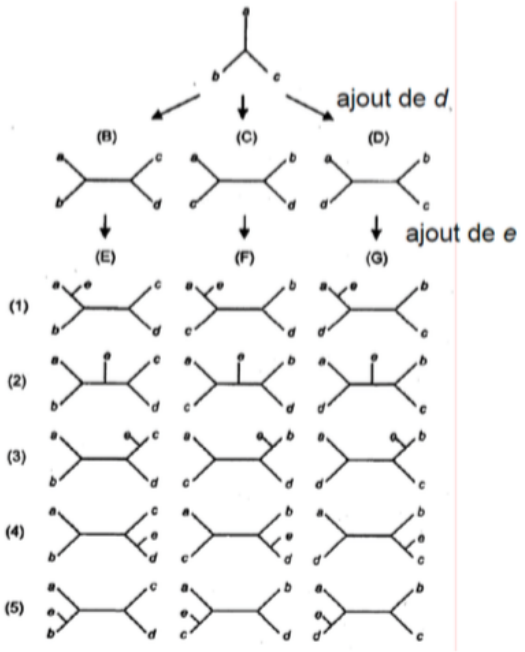


If $X_1 \cap X_2 \neq \emptyset$ then $X = X_1 \cap X_2$ and $P = P_1 + P_2$.

If $X_1 \cap X_2 = \emptyset$ then $X = X_1 \cup X_2$ and $P = P_1 + P_2 + 1$, as one change is observed.



Parsimony analysis is time consuming, so there should not be more than 10 leaves. There exists however an alternative approach, called the branch-and-bound methods, which can be used for $m < 20$. In this method, instead of looking at all configuration, the right tree is chosen from the start.



1. Start with a 3 OTU-tree (arbitrary order)
2. Add one more OTU to every branch of the tree and keep the position with the lowest cost
3. Assess the nearest neighbor trees and keep the tree with the lowest score
4. Go back to step 2

Repeat the whole procedure changing the order of the sequences to be added.

For the construction of the most parcimonious cladogram, on must first assess all possible trees. The next step is the localisation of the transformation on the tree. The most parcimonious tree is chosen in order to *minimise homoplasy* (character shared by a set of species but not present in their common ancestor = convergence) and to *maximise congruence* by looking for synapomorphies (character shared by two or more taxonomic groups).

Conclusions Maximum Parsimony methods are not suitable for more than 20 sequences without heuristics. When, one or more most parsimonious trees can be found, one must define a consensus tree. Parsimony does not seem to depend on an explicit model of evolution. It gives both trees and associated hypotheses of character evolution. The branch lengths do not have an unique meaning. Example of Parsimony Programs are PAUP and PHYLIP DNAPARS and PROTPARS.

11.2.2 Distance matrix method

The goal of this method is to reconstruct an evolutionary tree from a distance matrix. With a $N \times N$ distance matrix D, a weighted tree T with N leaves fitting D can be constructed.

Additive tree If the branches within a tree each have a specified length, then the distance between any two nodes can easily be computed as the total length of the (unique) path connecting them. In this way a tree can specify a distance matrix between its leaf nodes. However, not all distance matrices have this "additivity" property.

Biologically, additivity is an important property for a distance matrix: the actual number of substitution events separating two taxa from their last common ancestor (their genetic distance) forms an additive distance.

If distance matrix D is additive, the problem of building a tree has solution and there is a simple algorithm to solve it.

Ultrametric tree Let a pairwise distance be defined as any numeric variable that expresses the difference between two objects a and b and has the properties of being metric, meaning that

1. $d(a,a) = 0; d(b,b) = 0; d(a,b) \geq 0$
2. if $a \neq b$, then $d(a,b) > 0$
3. $d(a,b) = d(b,a)$
4. $d(a,c) \leq d(a,b) + d(b,c)$

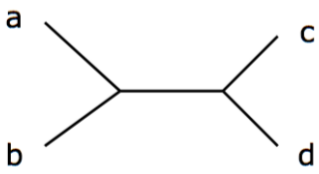
This last properties can be relaxed to $d(a,c) \leq d(a,b) + d(b,c) \rightarrow d(a,c) \leq max d(a,b), d(b,c)$.

Any set of distances having these properties will produces an ultrametric tree.

A generalisation of ultrametric trees are additives trees (4-point conditions):

$$d(a,b) + d(c,d) \leq max d(a,c) + d(b,d), d(a,d) + d(b,c)$$

$$d(a,b) + d(c,d) \leq d(a,c) + d(b,d) = d(a,d) + d(b,c)$$



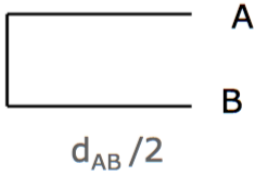
Unweighted Pair Group Method with Arithmetic mean (UPGMA) This method considers a molecular clock and rooted trees. The change rate along the branches of a tree is constant and the distances are ultrametric.

UPGMA is a simple approach for making rooted trees considering an ultrametric distance matrix. Sequential clustering algorithm in which local topological relations hops are inferred in order of decreasing similarity. This method assumes equal distribution rates (molecular clock hypothesis). It is however not a reliable method because when there are many sequences, the molecular clock hypothesis is not valid anymore.

Let us consider a case with four OUT. In the first step, the lowest distance is considered

	OTU		
OTU	A	B	C
B	d_{AB}		
C	d_{AC}	d_{BC}	
D	d_{AD}	d_{BD}	d_{CD}

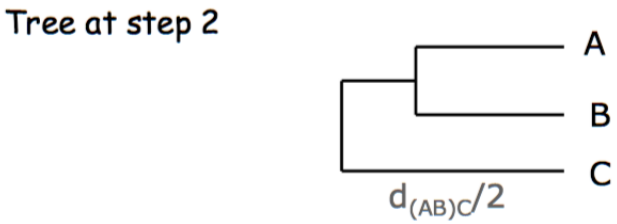
Tree at step 1



After the first clustering, A and B are treated as a single composite OTU and a new distance matrix is calculated. The smallest distance of this new matrix is once again considered.

	OTU	
OTU	(AB)	C
C	$d_{(AB)C}$	
D	$d_{(AB)D}$	d_{CD}

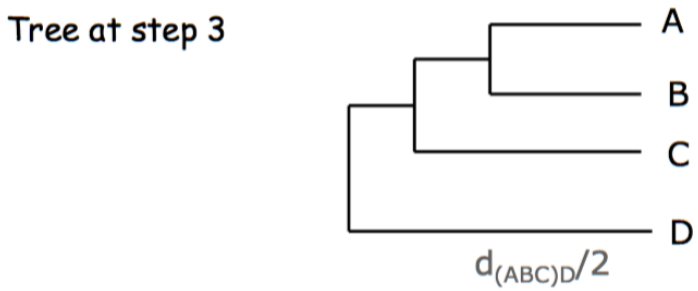
$d_{(AB)C} = (d_{AC} + d_{BC})/2$ and $d_{(AB)D} = (d_{AD} + d_{BD})/2$



The distance between two composition OTUs is computed as the arithmetic mean of the pairwise distances between constituent OTUs of the two composite OTUs:

$$d_{(ij),(mn)} = \frac{1}{4}\left(d_{im} + d_{in} + d_{jm} + d_{jn}\right)$$

. In general, $d_{XY} = \sum_{i,j} \frac{d_{ij}}{nXnY}$.



$d_{(ABC)D} = (d_{AD} + d_{BD} + d_{CD})/3$

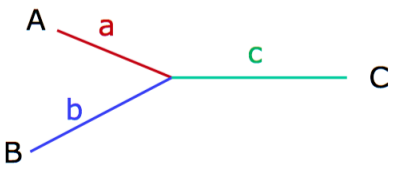
Fitch and Margoliash method In this method, no molecular clock is considered, therefore the banches linking the leaves to the shared node might have different lenghts. Also, unrooted trees are considered. Change rates are variable (different branch lengths) and least-squares fit of alternative trees is used.

Branch length is inferred from a set of 3 pairwise distances (3-point formula).

The FM method follows the next steps:

1. Select the two species/sequences with the minimal distance and calculate the mean distance to the other.

2. Calculate the branch length for these two OTUs



$a+c = d_{AC}$
$b+c = d_{BC}$
$a+b = d_{AB}$

The solution is:

$a = (d_{AB} + d_{AC} - d_{BC})/2$
$b = (d_{AB} + d_{BC} - d_{AC})/2$
$c = (d_{AC} + d_{BC} - d_{AB})/2$

3. Treat the two selected species as a single composite OTU and calculate the new distance table
4. the next more closely related sequences are identified and repeat all the steps

Normally the FM method repeat the process starting with another sequence pair (all combinations are tested) to find a tree that best predicts the data in the distance table. The percent change from the actual to the predicted distance is determined for each sequence pair. These values are squared and summed over all possible pairs. This sum divided by the numbers of pairs (n(n-1)/2 less one (the number of degrees of freedom) provides the square of the percent standard deviation of the result.

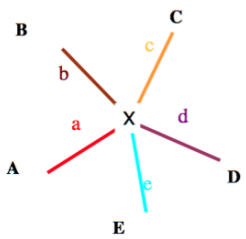
⇒ Least-scare method the tell how well the tree fit the experimental data.

Neighbor-joining method In this method, there is no molecular clock and unrooted trees are considered.

It is similar to the FM method except that the choice as to which sequences to pair at the very first step is determined so to give one tree only.

The Neighbor-joining method follows the next steps

1. Begin by placing all the taxa in a star-like topology.

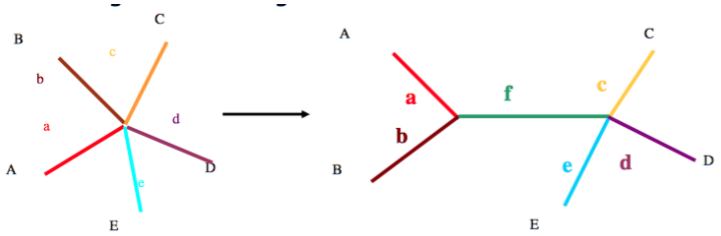


$$\sum_{i=1}^{m} L_{iX} = \frac{1}{m-1} \sum_{i<j}^{m} d_{ij}$$

L_{ix}= branch length
d_{ij}=distance

2. Decompose the star-like tree by combining pairs of sequences: A and B are connected to other OTUs via an internal branch, f. Calculate the sum of the branch lengths. Repeat the calculation for each combination , S_{AB}, $S_{BC} = 81$, $S_{CD} = 76$, $S_{DE} = 70 + 6$ others, and select the neighbors reducing the total branch lengths to the largest extent. The sum of branch length is expressed as

$$\sum_{ab} = \left[\frac{\sum_{i \neq a,b} d_{ia} + d_{ib}}{2(m-2)} \right] + \frac{d_{ab}}{2} + \frac{\sum_{i<j; i,j \neq a,b} d_{ij}}{m-2}$$



3. Once the choice of neighbors has been made, the branch lengths a and b and the average distance from AB to CDE is calculated by the F-M method
4. A new table of distance with A and B forming a single composite sequence is produced.

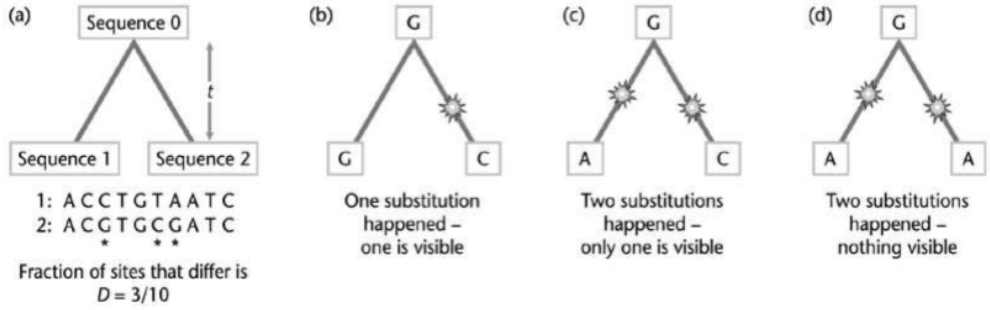
The N-J algorithm is used to find the next pair and the F-M algorithm is then used to find the next branch lengths.

The cycle is repeated until the corrected branched tree and the branch distances on that tree have been identified.

11.2.3 Models of sequence evolution

Evolutionary models are used in phylogenetic methods as the basis of : defining evolutionary distance and calculating the likelihood of a set of sequences evolving on a phylogenetic tree. The PAM scoring matrices used in protein sequence alignment algorithms are also related to evolutionary models (parsimony method).

Distance-base methods This methods counts what fraction of sites is different in aligned sequences.



If the time since divergence of the sequence is short, there will be few differences and these will happen at a different site. Thus each of the substitution is visible by comparing the 2 sequences (b).

If the time is larger, it is possible that there has been more than one substitution occurring at the same site. In (c), one substitution is detected although two occurred. In (d), there is no visible difference between the present-day sequences.

The last case (d) shows the need for a model to calculate the phylogenetic distance. Evolutionary distance d is defined as the average number of substitutions that have occurred per site between two sequences. This means d increases linearly with time and is additive.

Observed Hamming distance

$$D = \frac{n_d}{n}$$

Evolutionary distance

$$d = F(D)$$

11.3 Evaluating trees

The main criteria by which the accuracy of a phylogenetic tree is assessed are consistency, efficiency, and robustness.

Evaluation of accuracy can refer to an approach (e.g. UPGMA) or to a particular tree.

11.3.1 Bootstrapping

Bootstrapping is a statistical approach to measuring the robustness of a tree topology. It answers to following question: Given a branching order, how consistently does an algorithm find that branching order in a randomly permuted version of the original data set?

Bootstrapping considers a subset of data and look at the modification of the tree according to this subset. If the tree is modified, the method is not robust enough. If there are no change, to model is adequate.

The following steps are thus followed

1. Make an artificial data set obtained by randomly sampling columns from your multiple sequence alignment. Repeat this sampling a fixed number of times and analysed each bootstrap replicate data set.
2. Observe the percent of cases in which the assignment of clades in the original tree is supported by the bootstrap replicates.$>$ 70% is considered significant
3. Programme seqboot de la suite phylip