Master's course

Bioinformatics Data Analysis and Tools

Lecture 13: Repeat-finding methods, parameter optimization and benchmarking

Centre for Integrative Bioinformatic

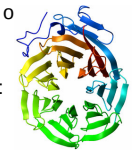
heringa@few.vu.nl

Content

- Problem example: Approaches to repeats finding
  - Background
  - FFT
  - Repro
  - Transitivity (TRUST)
- Parameterization of methods
  - GA
- Analyzing data
  - Statistics
  - PCA
- Comparing methods
  - Objective function
  - Standard of truth

Repeats

- Evolution reuses developed material developed
- Multiple stoichiometric and spatially close combined structure-function relationships
- In proteins, repeats vary from a single amino acid (e.g. poly-Gln) to complete domain sequences o combinations thereof.
- Many types of (near)identical repeats exist in genomes (Human genome > 50%) (next slide):
  - Micro- and mini-satellites
  - VNTRs
  - Interspersed repeats (http://globin.cse.psu.edu/courses/spring2000/repeats.html)
    - LINE and SINE repeats
    - LTR retroposons, also called retrovirus-like elements
    - DNA transposons



Ribbon diagram of the C-terminal WD40 domain of Tup1 (a transcriptional co-repressor in yeast), which adopts a 7-bladed beta-propeller fold. Ribbon is coloured from blue (N-terminus) to red (C-terminus).

Protein repeats and disease

A number of neurodegenerative diseases have been found to be strongly associated with proteins containing a poly-glutamine stretch. A conformational change in the expanded polyglutamine stretch is believed to form the molecular basis for disease onset.

Five neurodegenerative diseases (Huntington disease (HD), spinocerebellar ataxia type 1 (SCA1), dentatorubral-pallidoluysian atrophy (DRPLA), Macado-Joseph disease (MJD), and spinobulbar muscular atrophy (SBMA)) have been found to be strongly associated with a protein containing a polyglutamine stretch which is greatly expanded in affected individuals (for a review, see D.E.Housman, Nature Genet. 8, 10 -11, 1995). For the five diseases, the mean length of the glutamine repeat in unaffected individuals is approximately twenty, and the cutoff for pathology is about forty (the cutoff may be higher for MJD (Housman, 1995). Furthermore, long polyglutamine stretches have been found in many transcription factors.

Fibronectin repeat example

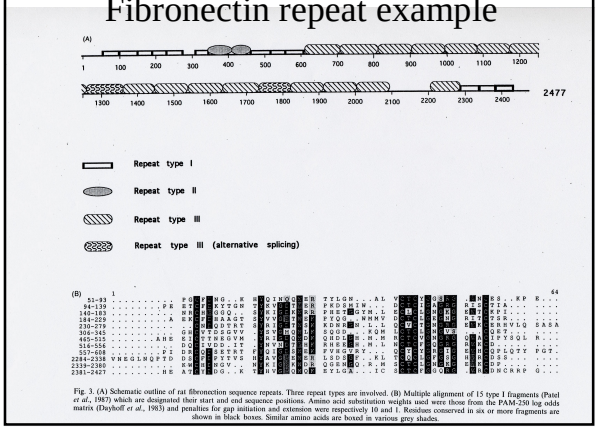


Fig. 3. (A) Schematic outline of rat fibronectin sequence repeats. Three repeat types are involved. (B) Multiple alignment of 15 type I fragments (Patel *et al.*, 1987) which are designated their start and end sequence positions. Amino acid substitution weights used were those from the PAM-250 log odds matrix (Dayhoff *et al.*, 1983) and penalties for gap initiation and extension were respectively 10 and 1. Residues conserved in six or more fragments are shown in black boxes. Similar amino acids are boxed in various grey shades.

Genome Repeats

Types of genome repeats:

- **Microsatellites**, 2-3bp (e.g. $(CA)_n$)
- **Minisatellites**, 10-100bp, occurring at more than 1000 locations in the human genome)
- **Variable number tandem repeats** (**VNTR**) range from 14 to 100 nucleotides long that is organized into clusters of tandem repeats, usually repeated in the range of between 4 and 40 times per occurrence. Clusters of such repeats are scattered on many chromosomes.

  VNTRs have been very effective in *forensic crime* investigations. When VNTRs are cut out, on either side of the sequence, by restriction enzymes and the results are visualized with a gel electrophoresis, a pattern of bands unique to each individual is produced. The number of times that a sequence is repeated varies between different individuals and between maternal and paternal loci of an individual. The likelihood of two individuals having the same band pattern is extremely improbable.
- **Interspersed repeats**
  - **SINE** (short interspersed nuclear element), **LINE** (long interspersed nuclear element) (next slide). These are also called non-LTR or poly-A retro(trans)posons
  - **LTR retroposons** Elements of several hundred bp in length, called the long terminal repeat, that appears at each end. Some autonomous elements are cousins of retroviruses (e.g., HIV) but are unable to survive outside of the cell, and are called *endogenous retroviruses*. None are known to be currently active in humans, though some are still mobile in mice. The so-called MaLR (mammalian LTR) elements, which arose before the mammalian radiation, seem to be non-autonomous repeats that move via proteins from endogenous retroviruses.
  - **DNA transposons.** Full-length autonomous elements encode a protein, called transposase, by which an element can be removed from one position and inserted at another. Transposons typically have short inverted repeats at each end.

LINE and SINE repeats
(elaboration of preceding slide)

- A *LINE* (long interspersed nuclear element) encodes a reverse transcriptase (RT) and perhaps other proteins.
  - Mammalian genomes contain an old LINE family, called LINE2, which apparently stopped transposing before the mammalian radiation, and a younger family, called L1 or LINE1, many of which were inserted after the mammalian radiation (and are still being inserted).
- A SINE (short interspersed nuclear element) generally moves using RT from a LINE.
  - Examples include the MIR elements, which co-evolved with the LINE2 elements. Since the mammalian radiation, each lineage has evolved its own SINE family. Primates have Alu elements and mice have B1, B2, etc.
- The process of insertion of a LINE or SINE into the genome causes a short sequence (7-21 bp for Alus) to be repeated, with one copy (in the same orientation) at each end of the inserted sequence. Alus have accumulated preferentially in GC-rich regions, L1s in GC-poor regions.

How to delineate repeats

1. Supervised: if you have a repeat motif, use profile-based methods or the like
2. Nonsupervised
   - You want to find a single repeat type
   - You want to find tandem repeats
   - You want to find interspersed repeats (intervening sequence stretches
   - You want to find multiple repeat types

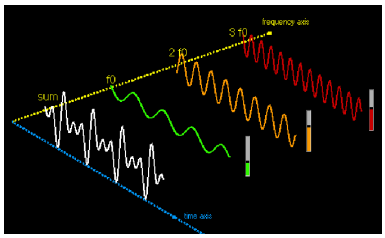
Fast Fourier Transformation

A **fast Fourier transform** (**FFT**) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse. FFTs are of great importance to a wide variety of applications, from digital signal processing to solving partial differential equations to algorithms for quickly multiplying large integers.

FFT is an intuitive algorithm for detecting repeats because it analysis **periodicity** in data

FFT

Often, you will sample a signal that is not a simple sine or cosine wave, it looks more like the "sum" wave in Figure below. However, Fourier's theorem states that any waveform in the time domain (that is, one that you can see on an oscilloscope) can be represented by the weighted sum of sines and cosines. The "sum" waveform below is actually composed of individual sine and cosine waves of varying frequency. The same "sum" waveform appears in the frequency domain as amplitude and phase values at each component frequency (that is, f0, 2f0, 3f0).



Taken from
http://zone.ni.com/devzone/cda/tut/p/id/3342

FFT

The Fourier transform converts a time domain representation of a signal into a frequency domain representation. The Fast Fourier Transform (FFT) is an optimized implementation of a DFT that takes less computation to perform. The Fourier Transform is defined by the following equation:

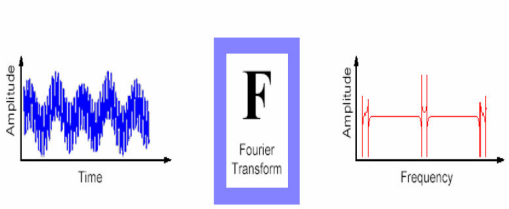
$$X(f) = \mathcal{F}\{x(t)\} = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi ft}\, dt \ , \qquad (1)$$

However, a digitizer samples a waveform and transforms it into discrete values. Because of this transformation, the Fourier transform will not work on this data. Instead, the Discrete Fourier Transform (DFT) is used, which produces as its result, the frequency domain components in discrete values, or "bins." So, the Discrete Fourier Transform (DFT) maps discrete-time sequences into discrete-frequency representations. DFT is given by the following equation:

$$X_k = \sum_{i=0}^{n-1} x_i\, e^{-j2\pi ik/n} \qquad \text{for } k = 0, 1, 2, \ldots, n\text{-}1 \ , \qquad (2)$$

Adapted from
http://zone.ni.com/devzone/cda/tut/p/id/3342

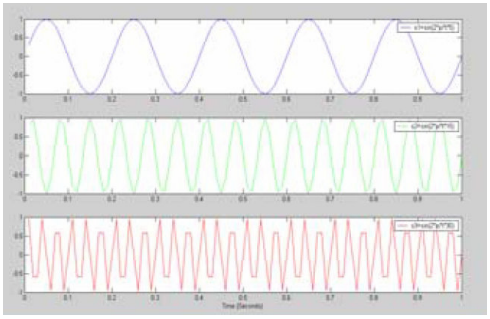
Fast Fourier Transformation



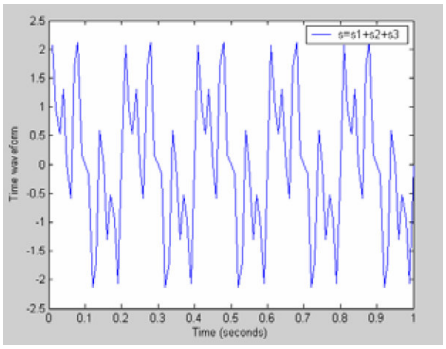
Raw data

FFT Periodogram

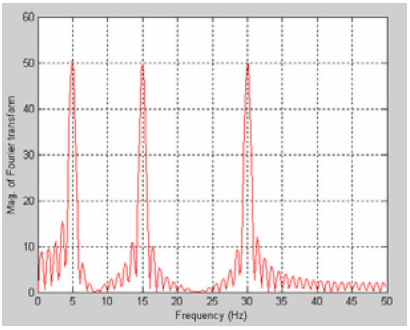
Fast Fourier Transformation



Fast Fourier Transformation



Fast Fourier Transformation



Fast Fourier Transformation

Limitations of FFT-based approaches with respect to repeats finding:

1. Repeats can be interspersed
2. Multiple repeat types
3. Incomplete repeats

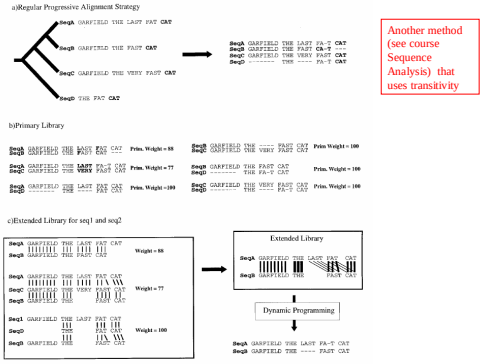
Optimised tandem repeats finding: TRUST

(Tracking repeats using significance and transitivity)

1. TRUST exploits transitivity
2. It has an dynamic profile building procedure
3. It tests for significance (using the EVD)

Szklarczyk, R. and Heringa, J. (2004) Tracking repeats using significance and transitivity. Bioinformatics 20 Suppl. 1, i311-i317.

Transitivity – T-COFFEE

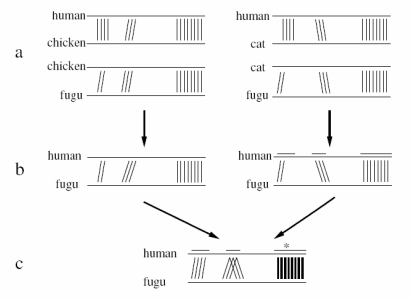


Another method (see course Sequence Analysis) that uses transitivity

Notredame C., Higgins D.G. and Heringa J. (2000) T-Coffee: a novel method for fast and accurate multiple sequence alignment, J. Mol. Biol. 302, 205-217

Transitivity – AuberGene (DNA)

Another method (see course Sequence Analysis) that uses transitivity

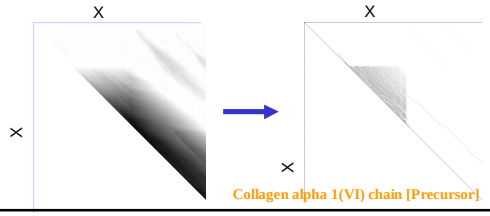


Sklarczyk, R., and Heringa, J. (2006). AuberGene - a sensitive genome alignment tool, Bioinformatics, Vol. 22 (12), 1431–1436

Remember the Waterman - Eggert method (local alignment)

$$M[i, j] = \max \begin{cases} M[i\text{-}1, j\text{-}1] + score(X[i],X[j]) \\ M[i, j\text{-}1] - 1 \\ M[i\text{-}1, j] - 1 \\ 0 \end{cases}$$

Waterman-Eggert:

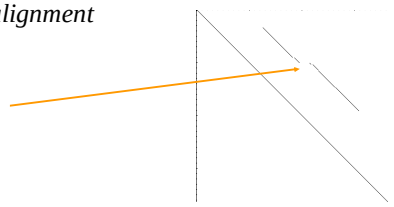


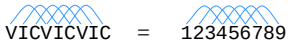
Collagen alpha 1(VI) chain [Precursor]

TRUST

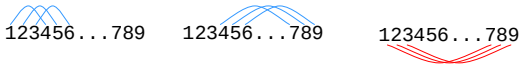
What is *a trace aka alignment*

- A trace



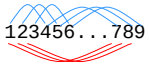


Transitivity, intro

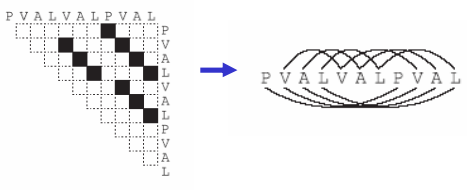


- Formally

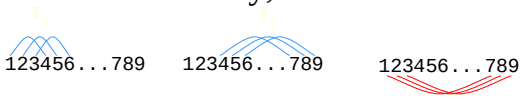
  1~4 *and* 4~7 *implies* 1~7



Transitivity

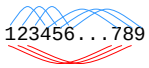


Transitivity, score



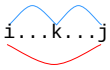
- Score for the new trace

  $score(T_{trans})=min(score(T_1), score(T_2))$

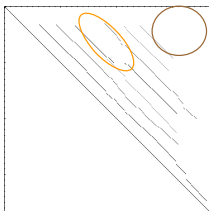


Transitive matrix

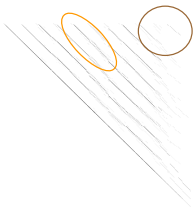
$M_{trans}[i, j] = sum_{(i, j) \text{ in } T}(score(T))$



Original M

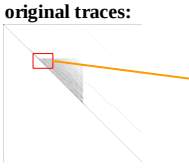


M_{trans}

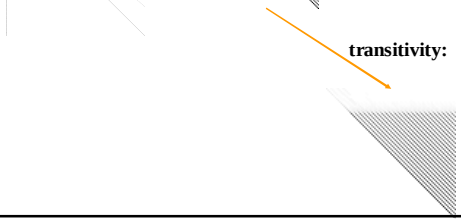


Zoom-in

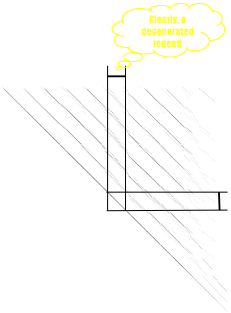
original traces:



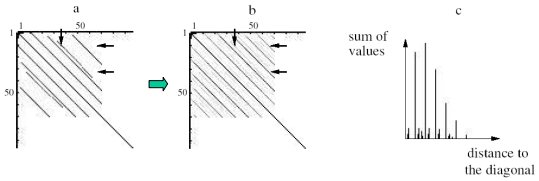
transitivity:



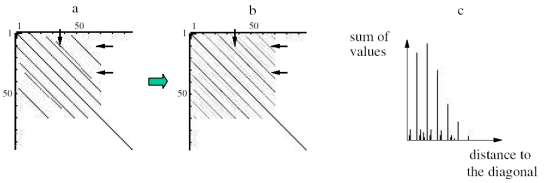
Transitivity: divergent regions



Spurious repeats



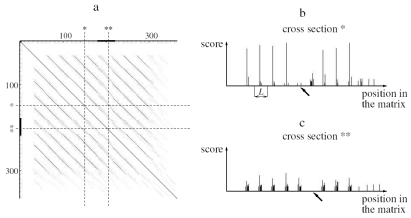
Estimating tandem repeat size L



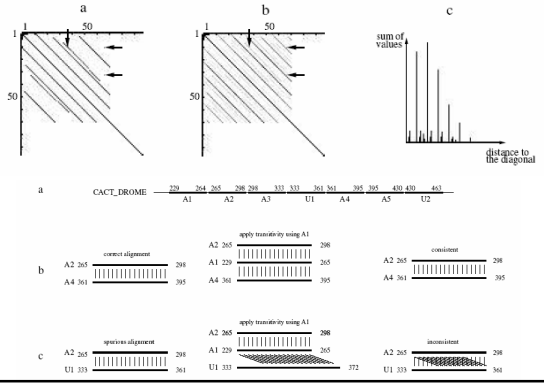
$dist[k] = sum_{j-i=k}(M_{trans}[i, j])$

Profile creation

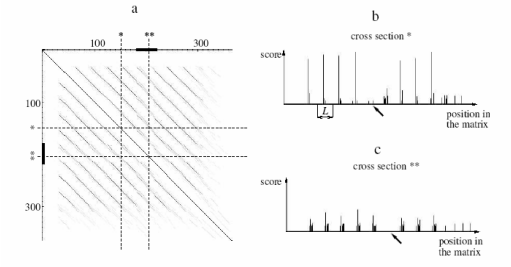
- Choose subsequence of length L
- Define the profile based on it



Transitivity – TRUST method



Transitivity – TRUST method



Transitivity – TRUST method

Method	Sensitivity	Accuracy	False positives
TRUST	43%	82%	24
TRUST -force	75%	85%	34
RADAR	63%	64%	86

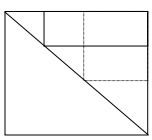
Graph-based clustering: **REPRO**

- Non-supervised algorithm for finding repeats in protein sequences, where
  - Repeats can be evolutionary distant (low sequence similarity)
  - Multiple sets of repeats can be recognised

Heringa, J., and Argos P. (1993). A method to recognize distant repeats in protein sequences. Proteins Struct. Func. Genet. 17, 391-411.

Graph-based clustering: **Repro**

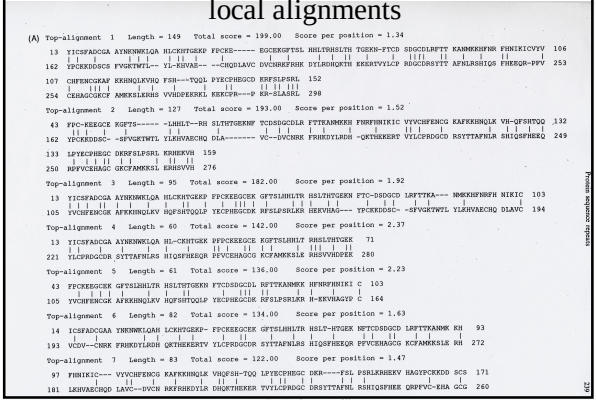
1. Calculate top-scoring non-overlapping local alignments



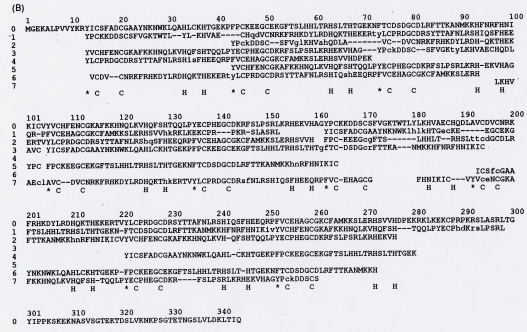
2. Stacking of local alignments
3. Make graph with N-termini of top-alignments as nodes
4. Perform graph-based clustering

Heringa, J., and Argos P. (1993). A method to recognize distant repeats in protein sequences. Proteins Struct. Func. Genet. 17, 391-411.

TFIIIA: seven top-scoring non-overlapping local alignments



TFIIIA: Stacking of local alignments



TFIIIA: Graph-based clustering

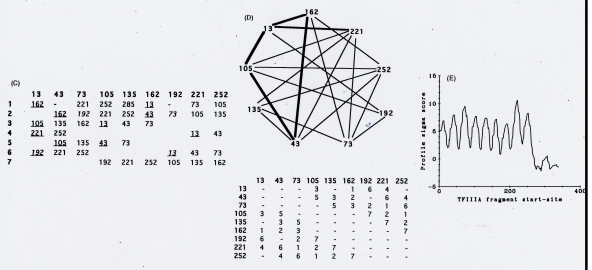
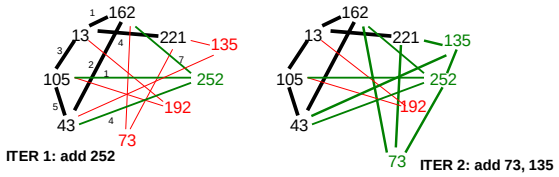


Fig. 4. (A) Seven non-overlapping top alignments for transcription factor TFIIIA from *X. laevis*. Residue substitution weights and gap penalties used are as for Fig. 3(B). (B) Stacking of the seven top alignments, designated 1–7 according to their score rank, onto the complete TFIIIA sequence. The correct start sites of the nine repeats are indicated by asterisk. The conserved Cysteine and Histidine residues are indicated with respectively "C" and "H". Lower case characters in the aligned fragments denote deletion of one or more amino acids. This avoids the occurrence of gaps within the complete sequence. (C) Nine correct start site positions from the stacked alignment. The stacked residues under each start site are indicated here by their sequence position. Underscored numbers denote N-terminal start sites of the associated top alignments; nonunderscored numbers are start sites internal to the top alignments. Italic numbers represent start sites which are not consistently matched within the corresponding top alignments and which are obtained by extending the considered top alignments in the N-terminal direction by maximally two sequence positions. Two such start site pairs are shown: Top alignment 2 yields an extra internal start pair [73, 192] from the matched residues [74, 193], while for top alignment 6 an extra N-terminal start site pair [13, 192] can be inferred from [14, 193]. Both alignments are extended by one sequence position. (D) Graph and associated adjacency matrix based on seven TFIIIA top alignments. Thick lines in the graph designate the initial graph which comprises five start sites and is assembled from the N-terminal start sites within the first five top alignments [see Fig. 4(A)]. Thin lines denote edges to new start sites included during subsequent cluster iterations. In each cell [*i*, *j*] of the adjacency matrix is placed the rank of the top alignment that matches start sites *i* with *j*. (E) Plot of the profile scores based on the multiple alignment of Fig. 1(A) in number of standard deviations from the mean profile score deduced from 20 randomized query sequences vs the N-terminal positions of the query sequence fragments searched.

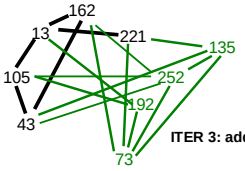
Repro: adding nodes (repeats) to graph



REPRO graph-cluster requirements:

1. ≥3 edges per node into graph

2. At least one edge in 'clan' (used top-alignments in graph)



Genetic Algorithm

A genetic algorithm is an optimisation algorithm.
It is analogous to the evolutionary optimisation procedure.

* The central component is a genome, defined as an array of genes, the status of which is represented by a number. For example [4 7 3 8] would be a genome composed of four genes with values (states) 4, 7, 3 and 8.
* A genome can also be considered as an 'individual'. In a genetic algorithm, a large number of individuals is created that form a 'population'.
* Each individual is subdued to a test, often the performance of an application (e.g. alignment), where the genes are used to specify program parameters (e.g. gap penalties). The test yields a score or fitness value.
* All individuals are ranked (sorted) according to performance.
* Only the top performing individuals pass their genes to the next generation, the rest is discarded.
* The new generation is generated using the genomes of the top performing individuals. This can be done through several 'operators': mutation, crossover, transposition inversion.
* The algorithm is performed until it converges to a stable solution.

Genetic Algorithm A genetic algorithm is an optimisation algorithm. It is analogous to the evolutionary optimisation procedure. * The central component is a genome, defined as an array of genes, the status of which is represented by a number. For example [4 7 3 8] would be a genome composed of four genes with values (states) 4, 7, 3 and 8. * A genome can also be considered as an 'individual'. In a genetic algorithm, a large number of individuals is created that form a 'population'. * Each individual is subdued to a test, often the performance of an application (e.g. alignment), where the genes are used to specify program parameters (e.g. gap penalties). The test yields a score or fitness value. * All individuals are ranked (sorted) according to performance. * Only the top performing individuals pass their genes to the next generation, the rest is discarded. * The new generation is generated using the genomes of the top performing individuals. This can be done through several 'operators': mutation, crossover, transposition, inversion. * The algorithm is performed until it converges to a stable solution.

Example Setup

Near-optimal parametrisation by Genetic Algorithm

Genes [0-4] adopt random values, for example from 0 to 9.
They are transformed to yield the parameter space.

```
Genes         Parameters
-----------------------------
weight        =  0.05 * parameter[0];
gap-open      =  (0.5  * parameter[1]) + 10;
gap-extend    =  0.5  * parameter[2];
add-constant  =  0.4  * parameter[3];

Nr    genome      score (= fitness)
0:    [5 5 8 5]   153.6867
1:    [3 8 6 4]   153.6446
2:    [4 6 6 3]   153.3365
3:    [7 2 8 0]   153.2564
4:    [4 6 7 4]   153.0322
```

Example Setup
Near-optimal parametrisation by Genetic Algorithm

Genes [0-4] adopt random values, for example from 0 to 9.
They are transformed to yield the parameter space.
Genes Parameters
---
weight = 0.05 * parameter[0];
gap-open = (0.5 * parameter[1]) + 10;
gap-extend = 0.5 * parameter[2];
add-constant = 0.4 * parameter[3];

Nr	genome	score (= fitness)
0:	[5 5 8 5]	153.6867
1:	[3 8 6 4]	153.6446
2:	[4 6 6 3]	153.3365
3:	[7 2 8 0]	153.2564
4:	[4 6 7 4]	153.0322

Typical Genetic Algorithm Scheme

Iteration scheme of Genetic Algorithm

```
      1. generate 200 random genomes

   -> 2. run alignments with paramaters of each gene
  |
  |    3. score result (fitness)
  |
  |    4. sort (according to fitness)
  |
   <- 5. select top genomes and create new generation
         [5 5 8 5] [3 8 6 4]
                   X
               [3 8 8 5]
```

Typical Genetic Algorithm Scheme Iteration scheme of Genetic Algorithm
1. generate 200 random genomes ->
2. run alignments with paramaters of each gene | |
3. score result (fitness) | |
4. sort (according to fitness) | <-
5. select top genomes and create new generation
[5 5 8 5] [3 8 6 4]
X
[3 8 8 5]

Genetic Algorithm (III)

Iteration results

Iteration 1	Iteration 5	Iteration 8
0: [5 5 8 5 1] 153.68	0: [5 5 8 3 1] 154.90	0: [5 5 8 3 1] 154.90
1: [3 8 6 4 0] 153.64	1: [5 4 8 2 2] 154.68	1: [5 4 8 3 2] 154.90
2: [4 6 6 3 0] 153.33	2: [5 4 8 2 2] 154.68	2: [5 4 8 3 2] 154.90
3: [7 2 8 0 2] 153.25	3: [5 4 8 2 0] 154.64	3: [5 4 8 2 2] 154.90
4: [4 6 7 4 2] 153.03	4: [5 4 8 3 0] 154.64	4: [5 4 8 3 2] 154.90
5: [3 8 7 1 8] 152.48	5: [4 5 8 3 3] 154.44	5: [5 4 8 3 2] 154.90
6: [6 3 8 2 5] 152.37	6: [4 5 8 3 0] 154.39	6: [5 4 8 3 2] 154.90
7: [6 5 3 3 0] 152.37	7: [4 5 8 3 0] 154.39	7: [5 4 8 3 2] 154.90
8: [6 4 5 3 3] 152.30	8: [4 5 8 3 0] 154.39	8: [5 4 8 2 2] 154.90
9: [7 7 8 0 1] 152.01	9: [4 4 8 3 2] 154.35	9: [5 4 8 3 2] 154.90
10: [5 5 4 5 2] 151.91	10: [5 5 8 3 0] 154.32	10: [5 4 8 3 2] 154.90

Multivariate statistics – Principal Component Analysis (PCA)

Principal component analysis (PCA) involves a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called *principal components*. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.

Traditionally, principal component analysis is performed on a square symmetric matrix of type SSCP (pure sums of squares and cross products), Covariance (scaled sums of squares and cross products), or Correlation (sums of squares and cross products from standardized data).

The analysis results for objects of type SSCP and Covariance do not differ, since these objects only differ in a global scaling factor. A Correlation object has to be used if the variances of individual variates differ much, or if the units of measurement of the individual variates differ.

The result of a principal component analysis on such objects will be a new object of type PCA

Multivariate statistics – Principal Component Analysis (PCA)

Objectives of principal component analysis

To discover or to reduce the dimensionality of the data set.

To identify new meaningful underlying variables.

Multivariate statistics – Principal Component Analysis (PCA)

How to start

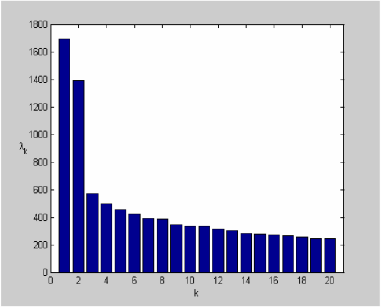
We assume that the multi-dimensional data have been collected in a table. If the variances of the individual columns differ much or the measurement units of the columns differ then you should first standardize the data.

Performing a principal component analysis on a standardized data matrix has the same effect as performing the analysis on the correlation matrix (the covariance matrix from standardized data is equal to the correlation matrix of these data).

Calculate Eigenvectors and Eigenvalues

We can now make a plot of the eigenvalues to get an indication of the importance of each eigenvalue. The exact contribution of each eigenvalue (or a range of eigenvalues) to the "explained variance" can also be queried: You might also check for the equality of a number of eigenvalues.

Multivariate statistics – Principal Component Analysis (PCA)



Eigenvectors ordered according to eigenvalues…

Multivariate statistics – Principal Component Analysis (PCA)

Determining the number of components

There are two methods to help you to choose the number of components. Both methods are based on relations between the eigenvalues.

Plot the eigenvalues: If the points on the graph tend to level out (show an "elbow"), these eigenvalues are usually close enough to zero that they can be ignored.

Limit variance accounted for and get associated number of components

Multivariate statistics – Principal Component Analysis (PCA)

Getting the principal components

Principal components are obtained by projecting the multivariate datavectors on the space spanned by the eigenvectors. This can be done in two ways:

1. Directly from the table without first forming a PCA object: You can then draw the Configuration or display its numbers (Gower, 1966).
2. Standard way: project the data table onto the PCA's eigenspace.

Multivariate statistics – Principal Component Analysis (PCA)

Gower (1966) has shown that, given $d_{ik}(i, k = 1, \ldots, n_{ref})$ which represents a matrix of pairwise Euclidean distances between n_{ref} reference objects, the co-ordinates $x_{i,j}(j = 1, \ldots, n_p)$ of the corresponding points P_i in a Euclidean subspace of n_p dimensions can be found by the following procedure.

(1) Define $a_{ik} = -\frac{1}{2}d_{ik}^2$, $i, k = 1, \ldots, n_{ref}$.

(2) Transform a_{ik} to $b_{ik} = a_{ik} - \langle a_{ik}\rangle_i - \langle a_{ik}\rangle_k + \langle a_{ik}\rangle_{ik}$, where $\langle\cdots\rangle_{i,\ldots}$ is the mean over all specified indices and $i, k = 1, \ldots, n_{ref}$.

(3) Find all $n_p \le n_{ref} - 1$ positive eigenvalues (λ_j) of the matrix with elements b_{ik} and all corresponding column eigenvectors $[x_{ij}, i = 1, \ldots, n_{ref}]$ where $j = 1, \ldots, n_p$.

(4) Scale these n_p column vectors which compose matrix $\mathbf{X}$ so that $\mathbf{X}^T\mathbf{X} = \mathbf{\Lambda}$ where matrix $\mathbf{\Lambda} = diag(\lambda_1, \ldots, \lambda_{n_p})$.

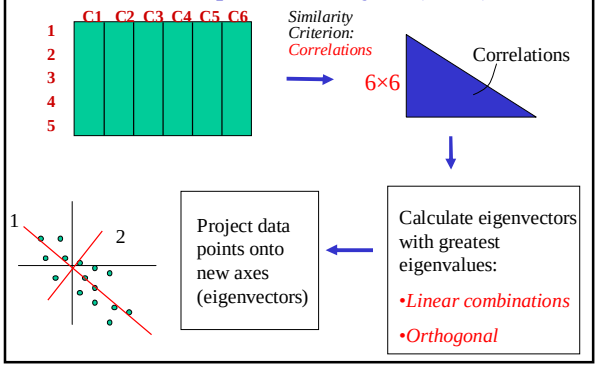
The eigenvalues determined at step 3 represent the variation along a corresponding axis. So, if eigenvalues together with eigenvectors are sorted in decreasing order, the first two co-ordinates represent the best planar projection. Negative eigenvalues, resulting from non-Euclidean distances in our applications, usually account for less than 20% of the variation.

Multivariate statistics – Principal Component Analysis (PCA)

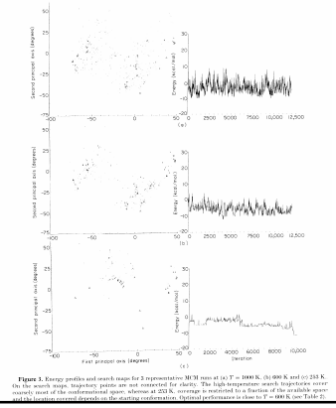
Mathematical background on principal component analysis

The mathematical technique used in PCA is called eigen analysis: we solve for the eigenvalues and eigenvectors of a square symmetric matrix with sums of squares and cross products. The eigenvector associated with the largest eigenvalue has the same direction as the first principal component. The eigenvector associated with the second largest eigenvalue determines the direction of the second principal component. The sum of the eigenvalues equals the trace of the square matrix and the maximum number of eigenvectors equals the number of rows (or columns) of this matrix.

Multivariate statistics – Principal Component Analysis (PCA)



Multivariate statistics – Principal Component Analysis (PCA)



A recap on Benchmarking

Benchmarking

How accurate?
Sequence searching:
- Specificity
- Selectivity

Sequence alignment:
- Column score or pairscore (correctness)
- Alignment score (significance)

General:
- Z-score

How fast?
- Time complexity
- CPU time

Statistics

Random variables described by probability density of normal distribution

$$p(x) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

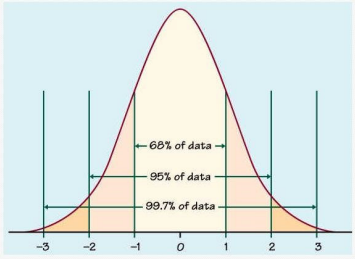
The distribution belongs to the class of Gaussian distributions

$$p(x) = e^{-((a-b)^2/c^2)}$$

In benchmarking we have two random distributions:
The background distribution of all true negative scores and the target distribution of all true positive scores.

Z-score

sigma = 1 -> 68% of area
sigma = 2 -> 95% of area
sigma = 3 -> 99.7% of area



http://www.ltcconline.net/greenl/courses/201/probdist/zScore.htm

Z-Transformation

Normalisation by linear transformation

Yields distribution with mean mu=0 and standard deviation sigma = 1

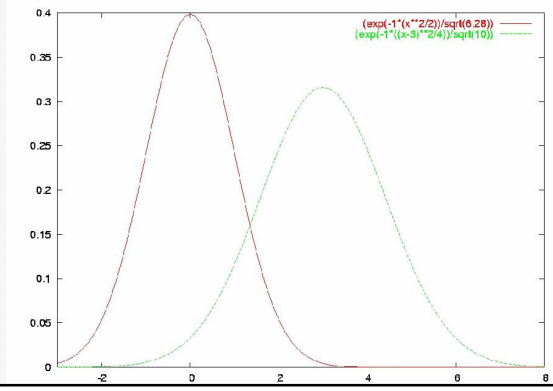
$$z = \frac{x - \mu}{\sigma}$$

$$\mu = \sum_{i}^{N} \frac{x_i}{N}$$

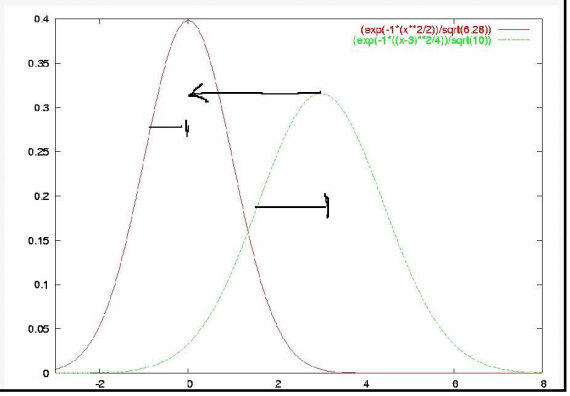
Variance

$$\sigma^2 = \sum_{i}^{N} \frac{(x_i - \mu)^2}{N - 1}$$

Background and Target Distribution



Z-Transformation



Benchmark Curve

What is a benchmark curve?
Every program that computes something can give true and false answers. In a benchmark curve one plots, for example, the sum of true answers in y-direction and the sum of false answers in the x-direction.

The terms 'true' and 'false' might be replaced by 'sensitivity' and 'specificity', or 'coverage' and 'error per query'; however, the basic idea is always the same.

An easy way to visualise a benchmark curve is a walk based on answers to a quiz: if the answer is correct go straight, if it is incorrect go right. The curve you walk is a benchmark curve: if you go mostly straight then you know a lot, if you go mostly right you know a little.

Receiver Operator Characteristics

Plot p(TP) [= sensitivity] over p(FP) [= 1 - specificity]

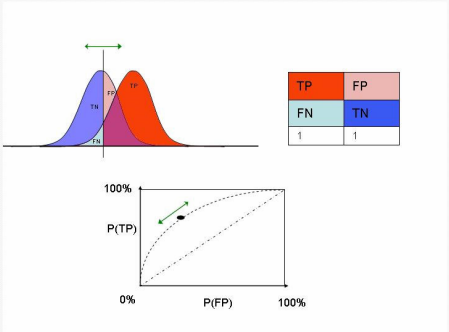
ROC curves are used to evaluate the results of a prediction and was first employed in the study of discriminator systems for the detection of radio signals in the presence of noise in the 1940s. In the 1960s they began to be used in psychophysics, to assess human (and occasionally animal) detection of weak signals. They also proved to be useful for the evaluation of machine learning results, such as the evaluation of Internet search engines. They are also used extensively in epidemiology and medical research.

Sometimes, the ROC is used to generate a summary statistic. Two common versions are:

- the intercept of the ROC curve with the line at 90 degrees to the no-discrimination line
- the area between the ROC curve and the no-discrimination line

However, any attempt to summarize the ROC curve into a single number loses information about the pattern of tradeoffs of the particular discriminator algorithm.

ROC Curve



http://encyclopedia.thefreedictionary.com/Receiver-Operator%20Characteristic

Specificity / Selectivity

Specificity

$$Specificity = \frac{TN}{(TN + FP)}$$

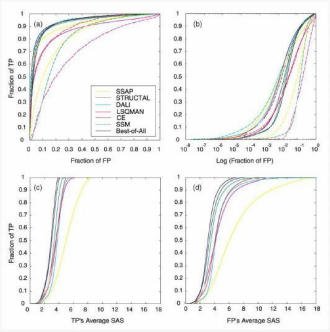
in benchmarks sometimes expressed as 'error per query'

Sensitivity

$$Sensitivity = \frac{TP}{(TP + FN)}$$

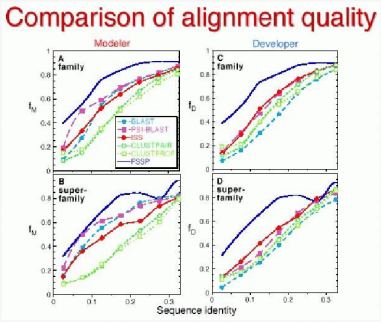
in benchmarks sometimes expressed to as 'percent coverage'

Example Benchmark Curves



ROC curves are not always the best presentation, because the true/false classification might be too coarse.

Example Benchmark Curve



In these plots both axes deviate from the standard true/false definition.

Important Aspects of Benchmarking

What is the 'standard of truth'?
Is the standard of truth independent of your own application?
For example, you can use a sequence alignment database derived from structural alignment as standard of truth for benchmarking a sequence alignment program, but you need to be careful when using a detabase derived from sequence alignme as standard of truth.

Is the training set random and independent of the test set?
When developing an application, you need to parametrise and test it on a standard of truth. The data set used for parametrisation is called 'training set' and the data set used for testing is called 'test set'.

1. It is absolutely mandatory that training and test data sets are independent (no overlap).
2. The data for the training set have to be drawn randomly from the standard of truth. If you pre-select specific data, the application will be biased towards these data and the benchmark as well.

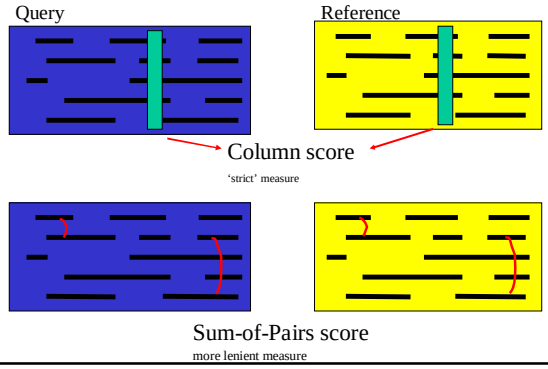
Evaluating multiple alignments (MSAs)

- Conflicting standards of truth
  - evolution
  - structure
  - function
- With orphan sequences no additional information
- Benchmarks depending on reference alignments
- Quality issue of available reference alignment databases
- Different ways to quantify agreement with reference alignment (sum-of-pairs, column score)
- "Charlie Chaplin" problem

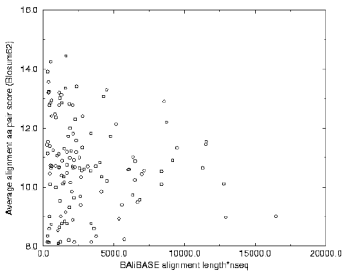
Charlie Chaplin once joined a Charlie-Chaplin competition in disguise and became third. What does this tell you about the 'objective function' with the jury?

Evaluation measures

What fraction of the matched amino acid pairs (or alignment columns) in the reference MSA are recreated in the query MSA?

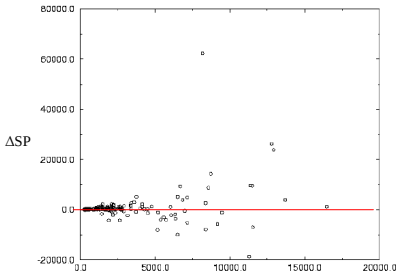


Evaluating multiple alignments



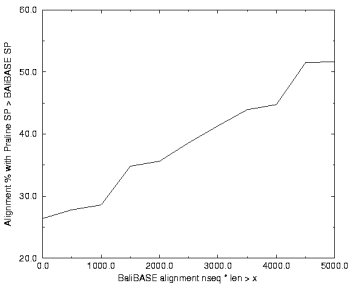
You can score a single MSA using the sum of all matched amino acid pairs score. This is also referred to as the Sum-of-Pairs (SP) score.. (a bit confusing with the use on the preceding slide..)

Evaluating multiple alignments



Many test alignments have a higher SP score than the reference alignment ("Charlie Chaplin problem')

Evaluating multiple alignments



Many test alignments have a higher SP score than the reference alignment ("Charlie Chaplin problem')

BAliBASE benchmark alignments

Thompson et al. (1999) NAR 27, 2682.

5 categories:

- *cat. 1 - equidistant*
- *cat. 2 - orphan sequence*
- *cat. 3 - 2 distant groups*
- *cat. 4 – long overhangs*
- *cat. 5 - long insertions/deletions*

141 Alignments in total

Comparing T-coffee with other methods

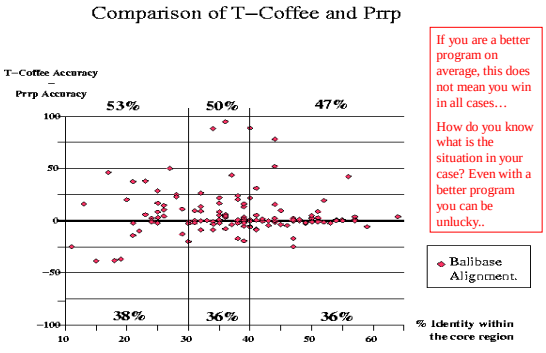
Table 2

Evaluation of four packages with BaliBase

Method Name	BaliBase Reference (Number of families)							T-Coffee Outperformed by method (%)
	Cat 1 (82)	Cat 2 (23)	Cat 3 (12)	Cat 4 (13)	Cat 5 (11)	Total 1 (141)	Total 2 (141)	
DiaIgn2	71.0	25.2	35.1	74.7	80.4	61.5	57.3	11.3**
ClustalW	78.5	32.2	42.5	65.7	74.3	66.4	58.6	26.2**
Prrp	78.6	32.5	50.2	51.1	82.7	66.4	59.0	36.9*
T-Coffee (CLE)	**80.7**	**37.3**	**52.9**	**83.2**	**88.7**	**72.1**	**68.7**	

Column scores are used here

BAliBASE benchmark alignments



If you are a better program on average, this does not mean you win in all cases…

How do you know what is the situation in your case? Even with a better program you can be unlucky..