BIOINFORMATICS

Exam topics

2016. DECEMBER 22.

PPKE ITK
Kajtsa Dóra

Órai diasorok és egyéb jegyzetek felhasználásával.

Tartalomjegyzék

A

A1. What are the two branches of computer uses in biology? Give a (detailed) narrow definition and a broad definition for bioinformatics!

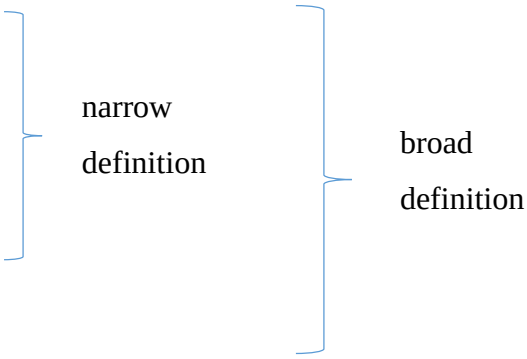
Two branches

Computational tools are fundamental in all branches of science. Computer uses in biology:

Data management

- acquisition
- storage, annotation
- interpretation, analysis, data-mining

Modelling, simulation



In data management, acquisition is when the scientist determines a gene sequence and uploads it into a database; storage and annotation is when the scientist adds notes to the sequence (e.g. species, sex etc.) and analysis and data-mining comes in when somebody compares sequences with databases and predicts functions of the gene based on comparison.

Modeling comes in when we model the movement of single molecules (single entities) or when we model large molecular assemblies or biological communities (bacteria, viruses) which we call large systems.

Single entities (molecular biology) – The molecular studies of simple systems (1-2 genes, 1-2 proteins etc.)

Large systems (system biology) – Measuring technique combined with specific computational approaches in order to model large systems.

Definitions

Narrow definition: Science of biological data. Mostly molecular biology. Description, management, interpretation.

Broad definition: Science of biological knowledge. All computer applications in molecular biology including modelling (simulation of behavior).

Bioinformatics in general

The objects in bioinformatics are molecular structures, metabolic pathways, regulatory networks and their databases. The methods are analysis and use of similarity. Bioinformatics is the intersection of molecular biology, knowledge representation and computer science.

A2. Give a generalized description of structures as entities and relationships - examples for entities and relationships in various systems!

Structure is a (constant space-time) arrangement of elements or properties, i.e. structure is a set of elements connected with relationships.

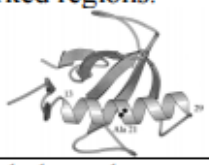
Ontology is a list of allowed elements and relations, and rules within a domain of knowledge; informally the controlled vocabulary and the rules.

The structure definitions are hierarchical (atom – amino-acid – protein – pathway – cell – tissue – etc.). For a given problem it is convenient to choose a standard descriptor or 'core structural level'. E.g. DNA sequences are the standard level for molecular biology problems.

For a standard or core description, we always have an underlying logical structure, plus various additional, simplified and annotated views.

Conceptual models of natural systems		
system	entities	relationships
molecules	atoms	atomic interactions (chemical bonds)
assemblies	proteins, DNA	molecular contacts
pathways	enzymes	chemical reactions (substrates/products)
genetic networks	genes	co-regulation
Structural descriptions		
system	entities	relationships
protein structure	atoms	atomic interactions (chemical bonds)
	secondary structures	sequential and topological vicinity
folds	C_α atoms	peptide bond
protein sequence	amino acids	sequential vicinity

A3. Describe the core data-types used in bioinformatics! Name the underlying models, the principal form of description and give examples of extended and simplified descriptions (annotations)!

Core data types in bioinformatics			
	underlying model	standard description	annotated description
Sequences	chemical formula: series of amino acids or nucleotides	character series: ATTGTCAT	marked regions: 
3D structures	3D chemical structure	3D coordinates with subunit description: A: X Y Z	marked regions: 
Networks	entities and relationships	graphs, graph matrices: A B C A 0 1 2 B 2 0 1 C 7 0 0	labeled graphs: 
Text*	human messages: – question – answer – conclusion	articles: – introduction – results, methods – discussion	articles with references and keywords

* Like other data, text has logical structure, standard, simplified and extended descriptions and databases. But: messages have an emitter (author) and an audience (reader), i.e. the articles are context dependent. They are loosely structured. There are ontologies for the language but not for the articles themselves.

A bit of cultural outlook

– Sequences resemble languages (language metaphor)

– 3D structures resemble real life objects (object metaphor)

– Networks resemble social assemblies (social metaphor)

– Scientific papers are messages (communication metaphor)

A4. Give a short overview of data representation types (structured, unstructured, mixed), and what is granularity!

Representation

Entities are described by the EAV scheme (entity–attribute–value). E.g. protein–mass–5 kDa Relations are described by the RAV scheme (relation–attribute–value). E.g. bond–length–2 Å.

Representation selection

Generally, a representation is selected in order to model something. Predictive models can use any property. For exact prediction, 'information-richness' is more important than the exact meaning of the variables (requires unstructured data) Interpretive models needs as simple content as it can be; the meaning of the variables are very important (requires structured data)

Representation types

We distinguish unstructured and structured data-types depending on if we know or want to use the internal structure.

Unstructured

We know nothing about internal structure, only the properties are known (global descriptors). These can be discreet or continuous. The unstructured data field is best described as vectors, where each dimension is an attribute and the vector's value describes the content.

The two vector types used in this model are binary (consist of 0 or 1) and non-binary vectors (consist of real or integer values, e.g. 0.950208,−3.5). Vector operations are fast.

E.g: catbot of animals: size, color, absence or presence of features (e.g. fur)

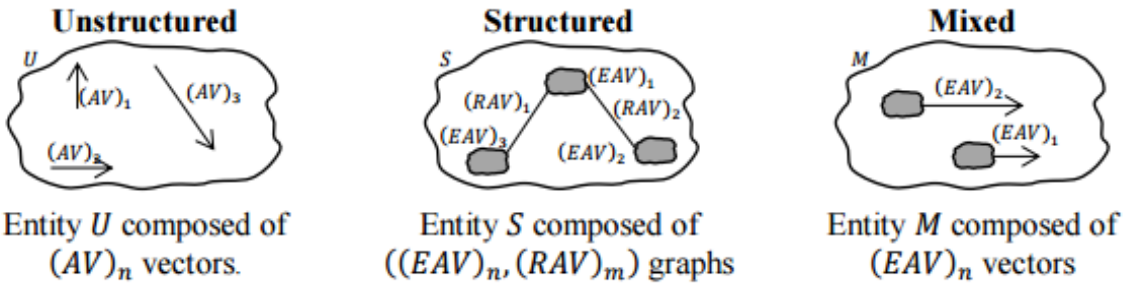
Structured

We know the internal structure in terms or entities and relationships (both described in terms of attributes and values, EAV and RAV). This representation is information-rich, allows detailed comparisons. The structured data field needs alignment (matching) for comparison. For example, a structured data representation is a character sequence or a graph.

Mixed

We decompose an object to parts of known structure, and count the parts (atomic composition, amino acid composition of proteins). The result is a vector describing the occurrences. Vector operations are fast, alignment is not necessary. The information content of the vector depends on the granularity of the parts.

E.g: clusters of genes
- → Clusters are structured inside: multiple alignment
- → No relation between clusters



Entity U composed of $(AV)_n$ vectors.

Entity S composed of $((EAV)_n, (RAV)_m)$ graphs

Entity M composed of $(EAV)_n$ vectors

Granularity

Granularity is the resolution of a description (e.g. 4 nucleotides or 10 nucleotides define the same class). Finding the right amount of detail is hard. This is part of 'the curse of dimensionality'. In case of too high resolution, all objects seem to be different, in case of too low resolution, all objects are equal, no difference between different groups: at low dimensions, all sequences are similar, at very high ones all are different.

For a given problem, optimal resolution can be found by calculating the intra- and inter-class variations, and by maximizing the latter.

A5. What are data aggregation and projection methods, and how can they help to understand biological data? (Give some examples!)

Aggregation of numbers and vectors

Numerical aggregation operations:

Sum, Average, Median, Minimum, Maximum, Stdev (When we calculate standard deviation, we automatically suppose that the distribution is normal (Gaussian, bell curve). This is often not true, but we still do this as a first approximation…)

Diagrammatic aggregation:

Histograms and distributions

Aggregation of sequences by functional and similarity links

Aggregation by links:

By similarity/distance links → distance matrix, heat map → similarity group → cladogram, tree

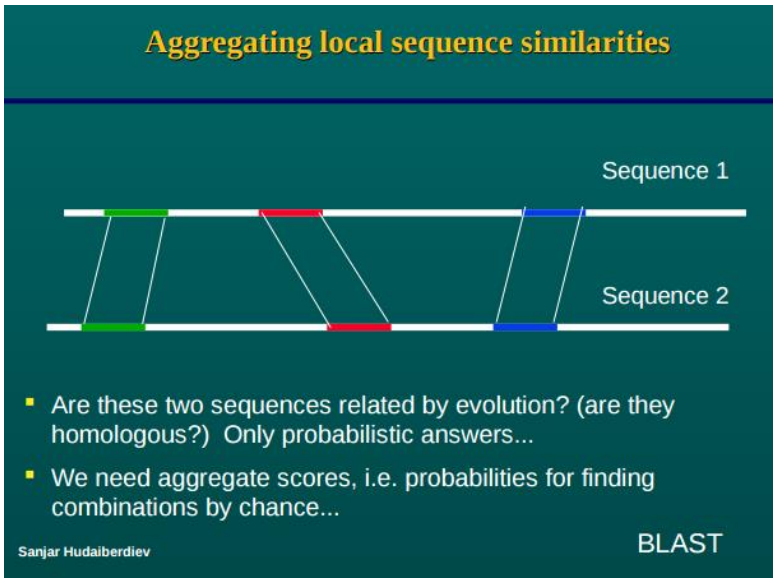
By functional (context) links → pathways

Aggregation by common motif:

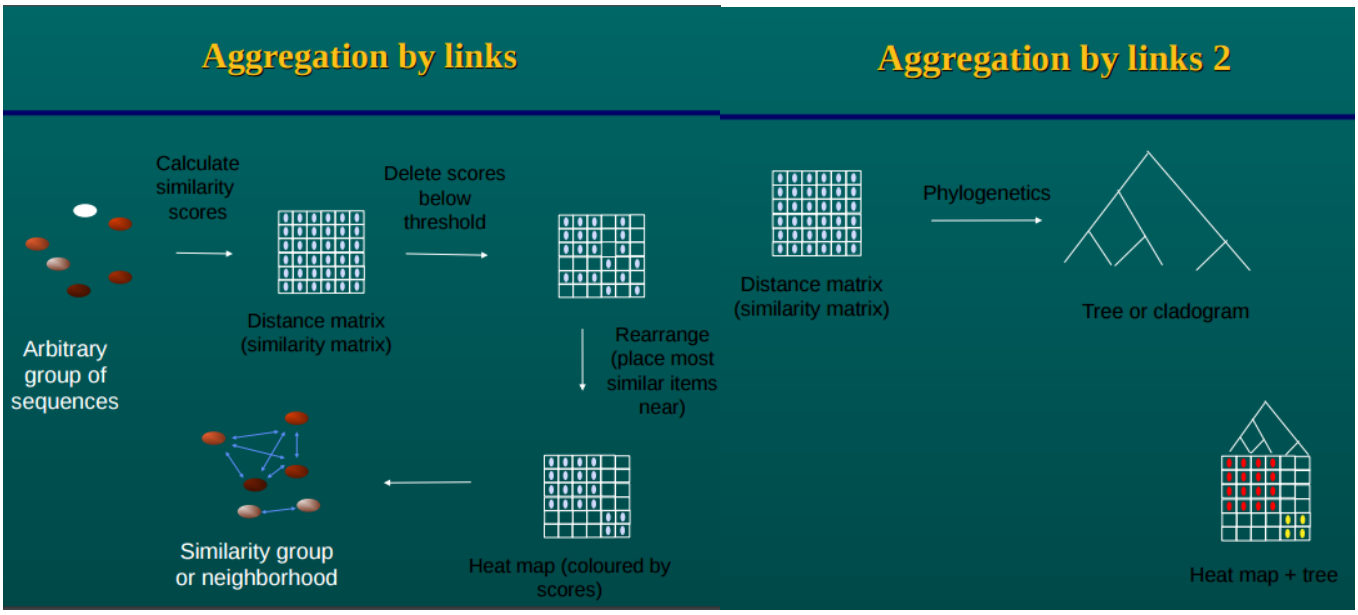
Multiple alignment

Consensus descriptions of multiple alignment

Agregation in analysis and identification: Biological objects are large and complex (genomes, proteomes, metagenomes, pathway data, etc.) Often, measuring instruments can only collect data on small pieces (next generation sequencing reads, peptide spectra in proteomics). Computational analysis of small fragments is accurate. There is one general trick: We divide a complex object into simple parts (like characteristic motifs), identify individual parts by simple numerical means, and then AGGREGATE the results. Not elegant, but works, even with very complex problems (forensic fingerprints).



Aggregation by similarity into distance matrices, heat maps, trees



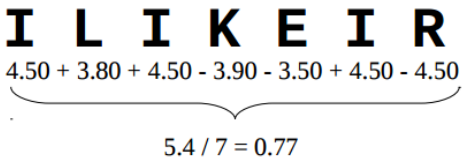
Projection: Numerical annotation of sequences, window sliding

What numbers do we plot:

- A property of an amino acid/nucleotide. I.e. a value stored in a lookup-table.
- A value calculated from the sequence or from the associated 3D structure (a „window")
- A value determined by experiment: The sequencing quality of the position. Number of reads "hitting" a position

Sliding Window Approach

- Calculate property for first sub-sequence
- Use the result (plot/print/store)
- Move to the next position in the sequence



Projection: Hydrophobicity plots

Prediction of hydrophobic and hydrophilic regions in a protein.

Hydrophobicity Plot:

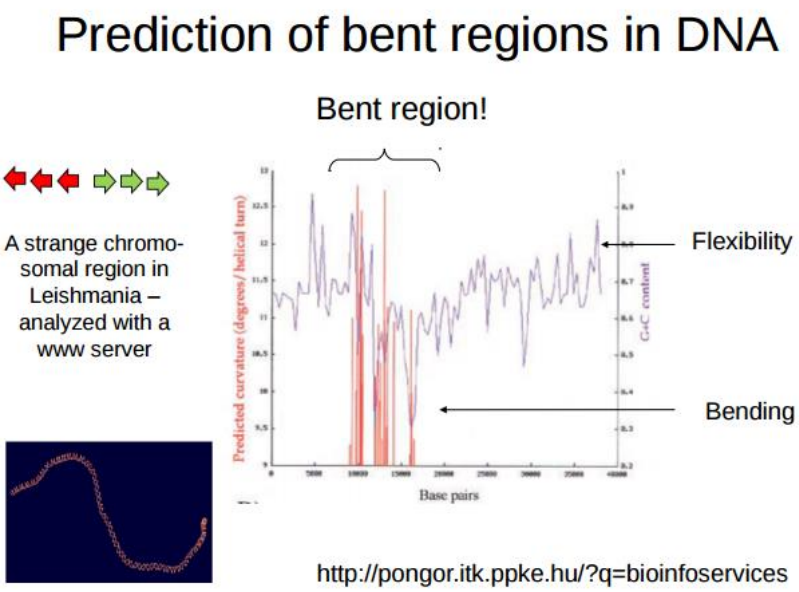
- Sum amino acid hydrophobicity values in a given window
- Plot the value in the middle of the window
- Shift the window one position

$$\langle H_i \rangle = \frac{1}{2k+1} \sum_{n=i-k}^{i+k} H_n$$

Large H → hydrophobic, e.g. membrane bound segments

Projection: DNA bending plots

Prediction bent regions in DNA.



A6. What are the substitution matrices? Describe the PAM and Blosum matrices!

The substitution matrix (also called scoring matrix) contains cost for amino acid or nucleotide identities and substitutions in an alignment. For amino acids it is a 20 × 20, for nucleotides it is a 4 × 4 matrix that can be constructed from pairwise alignments of related sequences. Related means either

– evolutionary relatedness described by an 'approved' evolutionary tree (PAM)

– any sequence similarity as described in the PROSITE database (BLOSUM) Groups of related sequences can be organized into a multiple alignment for calculation of the matrix elements.

Calculation of scoring matrices from multiple alignment

Matrix elements are calculated from the observed and expected frequencies using the 'log odds' principle. For A and B amino acids the matrix element in the intersection of row (column) A and column (row) B will be

$$M(A\ /\ B)\ = log\left(\frac{f(A\ /\ B)\ )}{f(A)\cdot\ f(B)}\right)$$

where $f(X)$ is the number of occurrences (frequency) of X, and $f(XY)$ is the number of occurrences where X is aligned with Y in the multiple alignment.

The log odds values in the matrix are normalized to a given range depending on the application (but the range does not matter much).

Nucleic acid matrices

The magnitudes of the elements are relative and can be scaled. Heuristic matrices can be easily constructed: the identity matrix contains '1'-s in the diagonal and '0'-s everywhere else. One can penalize certain associations assigning a large negative value to them, etc.

PAM matrix

PAM stands for Percent Accepted Mutation, which is a unit of evolutionary change for protein sequences. The PAM matrix is calculated from related sequences organized into 'accepted' evolutionary trees. This 20 × 20 matrix – where the columns add up to the number of cases observed – is converted into scoring matrix by log odds and scaling. In PAM 1 1% of amino acids mutate. One times the PAM matrix means 1 million years of divergence.

BLOSUM matrix

BLOSUM stands for Block Substitution Matrix. No evolutionary model is assumed; it is built from PROSITE derived sequence blocks and uses much larger, more diverse set of protein sequences. These matrices are sensitive to structural and functional substitution.

PAM			BLOSUM		
PAM X prepared by multiplying PAM 1 by itself a total of *X* times			BLOSUM X prepared from BLOCK sequences with *X* % sequence identity		
higher PAM detect more remote	PAM 40	short alignments with high similarity	BLOSUM 90	lower BLOSUM detect more remote	
	PAM 120	general alignment	BLOSUM 62		
	PAM 250	detecting weak local alignments	BLOSUM 30		
First useful scoring matrix for protein			Much later entry to matrix 'sweepstakes'		
Assumed Markov Model of evolution			No evolutionary model, built from PROSITE		
Derived from small, closely related proteins with ~15 % divergence			Uses much larger, more diverse set of protein sequences (30% – 90%)		
Errors are powered in higher PAM numbers			Errors arise from errors in alignment		

A7. Describe pairwise and multiple alignments! Give the relationship between pairwise and multiple alignment (difference in complexity and its consequences, construction of pairwise from multiple alignment and multiple from pairwise alignment)!

Description of alignments

Pairwise alignment

In the simplest case, we have two sequences originating from a common ancestor. The purpose of a pairwise alignment is to line up all positions that originate from the same position in the ancestral sequence i.e. to line up all residues that were derived from the same residue position in the ancestral gene or protein in two sequences. Pairwise alignments can only be used between two sequences at a time, but they are efficient to calculate and are often used for methods that do not require extreme precision (e.g. database searching).

Multiple alignment

Multiple sequence alignment is an extension of pairwise alignment to incorporate more than two sequences at a time. Multiple alignment methods try to align all of the sequences in a given query set. Multiple alignments are often used in identifying conserved sequence regions across a group of sequences hypothesized to be evolutionarily related. Multiple sequence alignments are computationally difficult to produce and most formulations of the problem lead to NP-complete combinatorial optimization problems.

Relationship between pairwise and multiple alignment

Difference in complexity

Alignments can be exhaustive or heuristic. Exhaustive, dynamic programming algorithms can be used to align two sequences. In principle we can include more sequences: instead of a 2D matrix we search a 3D cube or a 4D space etc., but this would require enormous memory and time resources (that we do not have yet). Instead of this, for multiple alignments we use heuristics that restrict the search space to a manageable size (at a price of losing some accuracy).

Multiple alignment form pairwise alignments

A multiple alignment cannot always be constructed from pairwise alignments (by simply writing them under each other). To construct a multiple alignment from pairwise alignments one can use additive or iterative methods.

Additive approach (once a gap, always a gap):

Do all pairwise alignments (e.g. Needleman–Wunsch). Select a simple, additive representation for the alignments: e.g. sequence *A* in alignment *X* can be represented as a series of gaps at given positions

(e.g.[0 2 0 0]). Such representations can be added: sum(x, y) ≔ max(x, y) After constructing the multiple alignment, we need to remove all empty columns.

Iterative approach (easy first, tricky last):
Do a pairwise comparison of all sequences. From this, calculate how sequences are related to each other (the more similar are easier to align). Construct a guide tree from the pairwise comparison values, using a clustering algorithm. Perform multiple alignment in order; the most similar are aligned first, the others are saved for later.

Pairwise alignments from multiple alignment
A multiple alignment can be decomposed into to pairwise alignments by deleting all lines but two and then delete all columns that are empty (contain only gaps). This pairwise alignment will still contain the 'traces' of the multiple alignment: A simple pairwise alignment algorithm may give a different result.

A8. Protein groups are heterogeneous from a number of points of view. Name a few group properties that make protein classification difficult! Explain the role of within-group and between-group similarities!

We call the proteins of all organisms the 'protein universe'. This is a cumulative result of evolution. The clustering in the protein universe is a graph where the nodes are proteins and the edges are the similarities between the proteins. A protein sequence similarity network contains many tightly connected similarity groups: these are those protein families that share structure and function (orthologs). The similarities between these groups are very low.

Difficulties of the classification

Most protein similarities are trivial so classification can be almost always done by alignment. But the rest is difficult due to the fact, that

– proteins consisting of a single domain from well-determined groups (families) but deeper analysis may reveal subgroups

– multi-domain proteins are connected to many different families and are difficult to deal with because of the shared domains between groups

Difficulties of working with groups

– Finding and validation of groups is possible only by multiple alignment

– Number of members from 1 to 3 · 105 (very imbalanced, highly redundant)

– Average sequence length from 5 to several thousand

– Relation of within group to between-group similarity varies…

– Besides of tight groups (where all members are similar with an $E < 10^{-4}$), there are loose groups where BLAST hits are usually very weak. There is no simple thresholding rule as $E < 10^{-4}$ on those groups.

Inerter-group and intra-group similarities

We call the relationship between two groups an easy case of similarity if the distance within the group D_w is less smaller that the distance between groups D_b. Therefore the similarity is high within the group and low between the groups: $Sw \gg Sb$

The difficult case of similarity is when the neighborhoods of positive members include negative members and therefore the within-group and between-group similarities are nearly equal $S_w \approx S_b$

A9. Describe how similarity search can help to understand the protein universe! What are the class annotated databases (COG, SBASE), what are the non-annotated cluster databases, how are they created and how can they help to get to know an unknown protein?

Proteins of all organisms is the protein universe. We can view the protein universe as a sequence similarity network which contains many tightly connected similarity groups. In these groups the proteins are belongs to the same protein family that share structure and function. Similarity search is important to find new members of a cluster, i.e. to predict the function of an unknown protein (based on its sequence).

Class annotated databases

Class annotated databases are databases of classes where the class members are labelled (i.e. we know the function of the group members). Class annotated databases are knowledge-base of meaningful similarities. Annotated clusters, such as COG, are manually curated, homogeneous, same function (orthologous). They can be used automatically, but difficult to maintain.

COG

The word COG stands for Clusters of Orthologous Groups. COG is a database of orthologous groups; it contains full sequences, best for prokaryotic groups. Sequences are in the same cluster carrying the same function. The clusters of proteins were generated by comparing the protein sequences of complete genomes. Each cluster contains proteins or groups of paralogs from at least three lineages. COG clustering method is as follows: Detect triangles of best hits between genomes. Merge triangles with a common side form clusters. Case-by-case manual analysis, examination of large clusters (might be split up).

SBASE

SBASE database is an online collection of protein domain sequences and related computational tools designed to facilitate detection of domain homologies based on simple database search. Protein sequence segments are annotated by structure, function, ligand-binding or cellular topology, clustered into over 6000 domain groups.

Domain identification and functional prediction are based on a comparison of BLAST search outputs with a knowledge base of biologically significant similarities extracted from known domain groups. The knowledge base is generated automatically for each domain group from the comparison of within-group (‘self’) and out-of-group (‘non-self’) similarities.

Non-annotated cluster databases (Uniref)

Uniref is a selection of Uniprot database, it contains automatically generated clusters. Separate collections for 90%, 50% identity clusters. It contains several million known sequences. This database is daily updated and highly redundant. Searches need to be checked manually.

Determine the function of an unknown protein

We usually use BLAST to find function of an unknown protein. If we found a very similar protein (e.g. $E < 10^{-4}$) with a known function in an annotated database, then we can assume that our unknown protein has the same function as the first hit. If the best hit has no function (e.g. 'hypothetical protein' or 'protein with unknown function') but the best hit is a member of a group in COG, then this strong similarity allows assignment of this function. If the best hit is not a member of a known protein group in COG but it is a member of an automated cluster in Uiref, we can look up the neighbors in some other clustering database and see if it has strong similarities of any protein with a known cluster. If yes, use that function. If the best hit is not in an automatically generated cluster or it is alone, the game is over.

A10. What is the maximum likelihood algorithm and which kind of data it should be used for? What is the Bayesian interference algorithm and which kind of data it should be used for? How can you determine the root of a phylogenetic tree? Describe consensus tree generating methods and data re-sampling!

Maximum likelihood

The maximum likelihood method uses standard statistical techniques for inferring probability distributions to assign probabilities to particular possible phylogenetic trees. The method requires a substitution model to assess the probability of particular mutations; roughly, a tree that requires more mutations at interior nodes to explain the observed phylogeny will be assessed as having a lower probability. In fact, the method requires that evolution at different sites and along different lineages must be statistically independent. What is the probability of seeing the observed data D given at model/theory T? $P(D|T)$

Bayesian inference

Bayesian inference can be used to produce phylogenetic trees in a manner closely related to the maximum likelihood methods. Bayesian methods assume a prior probability distribution of the possible trees, which may simply be the probability of any one tree among all the possible trees that could be generated from the data, or may be a more sophisticated estimate derived from the assumption that divergence events such as speciation occur as stochastic processes.

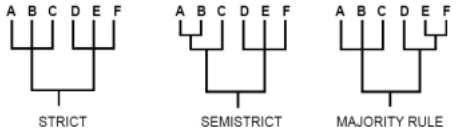
What is the probability that the model/theory T is correct given the observed data D? $P(T|D)$

Usage of different model based methods

Model based methods can be used if we have information on evolution models, we have time for the calculations and we have good statistics for the reliability of the results. For nucleotide sequences, if we have few of them, maximum likelihood is a good choice. If we have many of them, we should use Bayesian inference.

Rooting trees and consensus tree generation

A tree can be rooted by outgroup method. An outgroup must be unambiguously outside the clade of interest in the phylogenetic study. The DNA or protein sequences from the outgroup can be successfully aligned to sequences from the ingroup.



Data re-sampling

Data re-sampling generates several sub-samples – replications.

Two methods:

– Non parametric bootstrapping – random site selection with replacement

– Jackknife – delete half Calculate trees for all the sub-samples – The method is independent from sampling Generate consensus trees – 50% majority rule, only topology is evaluated

A11. What are structural and functional annotations in genomics? What is the typical workflow of an annotation process, and what kind of genomic regions are the most difficult to handle correctly?

Genome annotation is the process of attaching biological information to sequences.
It consists of two main steps:

- identifying elements in the genome, a process called gene prediction (syntax)
- attaching biological information to these elements (semantics)

Automatic annotation tools try to perform all this by computer analysis, as opposed to manual annotation (a.k.a. curation) which involves human expertise. Ideally, these approaches co-exist and complement each other in the same annotation pipeline.
The basic level of annotation is using BLAST for finding similarities, and then annotating genomes based on that. However, nowadays more and more additional information is added to the annotation platform. Other databases (e.g., ENSEMBL) rely on both curated data sources as well as a range of different software tools in their automated genome annotation pipeline.

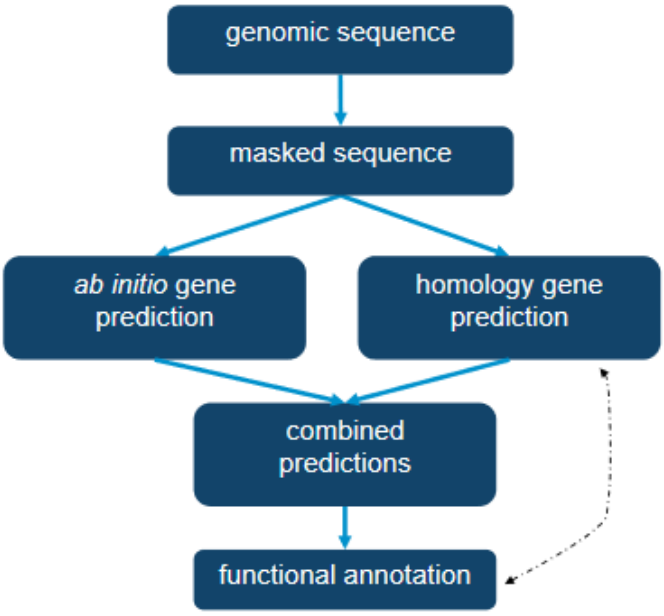
Structural annotation consists of the identification of genomic elements. (SYNTAX, using only the sequence)

- ORFs (open reading frames) and their localization
- gene structure (intron-exon)
- coding regions
- location of regulatory motifs

Functional annotation consists of attaching biological information to genomic elements.
(SEMANTICS, using dbases)

- biochemical function
- biological function
- regulation and interactions involved
- expression
- variants

These steps may involve both biological experiments and in silico analysis.



Mask repetitive regions (e.g., repeat masker)

Find genes ab initio and if possible by homology (many tools)

Combine both predictions (e.g., MAKER)

Repeats and low complexity regions are bad…

Repeats may confuse ab initio gene finders

- they may call exons or even complete genes in repeat regions.
- they may fragment gene predictions.

Repeats may confound sequence alignment

- especially in searches for synteny or segmental duplications.

A12. What is metagenomics? What are the main approaches and the main questions of a metagenomic analysis? Mention some databases and software tools of this field!

Metagenomics deals with samples taken directly from the environment:
Soil, water, hot spring, oil sands, human gut, stool;
Also called environmental genomics.

Necessarily more complex than genomics:
Mixture of multiple organisms
Many have never been looked at on the molecular level at all

Who's there? – Taxonomy analysis
How many/much of them are there? – Relative abundance of organisms
What do they do? – Functional analysis

Simple analysis using a reference gene 16SrRNA:

- Amplification of reference gene with general primers
- Overall microbial community composition (presence-absence)
- Consists of variable and conserved regions

Steps:

- DNA extraction
- From environment
- Samples
- From enrichment cultures
- PCR amplification
- Pyrosequencing
- Data analysis
- Microbial community analysis

Databases used: SILVA, GreenGenes

Whole genome sequencing (WGS)

- Sequencing + mapping to known genomes (or to specific marker database)
- Overall microbial community composition (quantitative)

- Dominant functions

Whole metagenome with „binning”:

- Map genes to (annotated) genomic sequences
- Count hits by taxa („bins”) – gives taxonomic composition incl. quantitation of taxa.
- You can use marker database instead of full genomes. Faster but less sensitive.

Whole metagenome with assembly:

- Assemble reads just like in genome assembly
- Complicated because of multiple unknown sources of metagenomic reads, lower coverage on individual genomes.
- Requires large computers.

Comparative metagenomic analysis

Across different environments:

- Individual taxon abundance
- Overall microbial community composition
- Dominant functions
- Environment-specific functions of same taxon

Within same environment:

- Changes over time

Metagenomic Tools (Local)

Qiime (Quantitative Insights Into Microbial Ecology):

- Consists of Python scripts
- Taxonomy and diversity statistics/visualization

Mothur:

- Commands written in C++ programs
- Taxonomy and diversity statics/visualization

MEGAN (Metagenome Analyzer):

- Provides functional as well as taxonomy analysis
- GUI with tree-based visualization

A13. Describe the main steps of preparing DNA for NGS data analysis! Summarise the workflow of analysis of mutations (give some examples for medical applications briefly)

Origin of the sample: tissue, cell line, salvia, blood, hair

Preparation

1. breaking up the cells

– Liquid nitrogen+hammer(tissues)

– Shredder + centrifuge (QIAshredder)(cells)

– Beads + homogenization(FastPrep)(everything)

2. isolation of DNA

3. DNA quality control (UV spectrometer)

4. DNA fragmentation

– Physical fragmentation: Acoustic shearing, Sonication

– Enzymatic method: DNase I or other restriction endonuclease

– Chemical fragmentation: heat digestion with a divalent (magnesium or zinc)

Analysis of mutation

Blood clotting is regulated by multiple pathways, in which Factor V has a critical role. Mutation in Factor V (Leiden mutation) leads to increased risk of thrombotic events. Factor V Leiden can be detected by snake venom, PCR-RFLP and sequencing.

Cancer is a genetic disease originating in a single cell. BRCA I/II are tumor suppressor genes (= loss of function). Germ line mutation in BRCA I/II leads to early breast cancer.

Oncogene-addiction: tumors depend on a single signal transduction pathway. The ERBB/KRAS signal transduction pathway has a critical role in solid tumors. Patients with a KRAS mutation do not respond to a therapy targeting a higher member in the pathway (e.g. panitumumab-EGFR). Methods for KRAS mutation status detection are either PCR or sequencing based.

A14. Describe the main difference between genomics and functional genomics! What is transcriptomics and what kinds of questions can be answered with it? What are microarray experiments and how can you analyze their results?

Genomics vs. functional genomics

Genomics

Sequenation, analyzation and annotation of the whole DNA (nuclear, mitochondrial and chloroplast DNA as well)

Functional genomics

Examination of genes or group of genes with high throughput methods which are designed based on the genome sequence (microarray, SNP) or uses sequenation (ChIP-seq, RNA-seq, etc.). Functional genomics is looking for the connections between genotype and phenotype at the level of the genome.

Microarray vs. NGS based methods

Microarrays are generally considered easier to use with less complicated and less laborintensive sample preparation than NGS. Microarray technique also require some prior knowledge of the genome. Microarray technique is also more accurate (nowadays). X-seq techniques In the above methods the DNA or the RNA is broken into small parts and from these parts ‘tags’ are sequened. These tags are the aligned with the reference genome. RNA-seq Examination of mRNA-s at the level of the genome

Genomic mapping

Map a gene in the genome.

Transcriptome mapping

Questions are the exact location of the promoters, position of the Transcription Start Site and the Transcription Factor Binding Site. The most important question is where, when, why and how many mRNA is produced by which gene? And how it is coded in the DNA?

DNA microarray

A DNA microarray (also commonly known as DNA chip or biochip) is a collection of microscopic DNA spots attached to a solid surface. Scientists use DNA microarrays to measure the expression levels of large numbers of genes simultaneously. Microarray data sets are commonly very large, and analytical precision is influenced by a number of variables. Normalization of the data is needed.

Algorithms that affect statistical analysis include:

- Image analysis
- Data processing
- Class discovery analysis
- Statistical analysis

De Novo assembly

De novo transcriptome assembly is the method of creating a transcriptome without the aid of a reference genome.

Gene expression analysis

RNA-seq can be used to determine the expression profile (abundance measurement of specific transcripts)

ChIP-seq

Examination of DNA-protein interactions at the level of the genome

GRO-seq

Examination of transcription at the level of the genome

B

B1. Describe data comparison - score, common alignment patterns!

The input of the comparison is two descriptions. For unstructured data the output is a score (similarity or distance). For structured data the output is a score (similarity or distance) and a common pattern (result of a matching, i.e. alignment).

Proximity measures (scores)

There are two types of scores: similarity measures (zero for different objects, large for ident i-cal objects) and distances (large for different objects, zero for identical objects). Scores are exists both for vectors and for structures. We say that a score is well behaved if it is bounded.

There is a very large and eve growing number of proximity measures. For easy problems, many of them work equally well, for difficult problems none of them do.

Popular distance measures

Vector distance

$$D_{ab} = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2}$$

This we call the Euclidean distance.

Generalized distances

$$D_{ab} = \left(\sum_{i=1}^{n} |a_i - b_i|^k \right)^{1/k}$$

This we call Minkowski metrics.

Popular similarity measures

Dot product

$$A \cdot B = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n = \sum_{i=1}^{n} a_i b_i$$

For binary vectors this is the number of matching nonzero attributes. Dot product is exactly 1 for identical vectors. Dot products of unit length vector is bounded in [0,1].

Association measures

$$J = \frac{a \cap b}{a \cup b}$$

The Jaccard (or Tanimoto) coefficient [0,1] expresses the similarity of two property sets a and b of non-zero attributes. J is 1 for iden-tical and zero for completely different sets.

Comparing structured descriptions

You can use proximity measures if you can turn the description into a vector. In addition, you can match (align) structures that gives a shared pattern.

Matching general structures

Matching graphs consists of finding the largest common subgraph. This matching task is a computationally hard problem: finding approximately identical subgraphs is NP complete.

Matching bit or character-strings – **Hamming distance**

The Hamming distance is the number of exchanges necessary to turn one string of bits or characters into another one. The two strings are of identical length and no alignment is done. The exchanges in character strings can have different costs, stored in a lookup table.

Edit distance between character strings – **Levenshtein distance**

Defined as a sum of costs assigned to matches, replacements and gaps. The two strings do not need to be of the same length. A numerical similarity measure between biological sequences is a maximum value calculated within a range of alignment. The maximum depends on the scoring system that includes a lookup table of costs, and the costing of the gaps. The scores are often not metric, but closed to metricity.

Motif *between aligned sequences*

Shared motifs point to evolutionary conservation. They are more informative than simple se-quences. „What a sequence whispers, an alignment pattern shouts out loud”

B2. Explain the Dot Plot method! Write down the algorithm, describe the parameters and list some usages!

Dot plot method is to visualize sequence identities along two n-length nucleotide or amino acid sequences. It is based on an $n \times n$ matrix with the two sequences written on the two ax-es. If we put a dot on those positions where the i-th row and the j-th column has the same nucleotide(group) we get a diagonal line for identical sequences and some other pictures for other sequences.

Algorithm

1. Select a word size and a scoring scheme.
2. For every pair of words
    a. compute a word match score in the normal way
    b. if the score reaches the cut-off score, draw a dot in the intersection of the two lines containing the two words. Else, no dots are placed.

Parameters

Word size

Word size defines the 'windowing function' applied on the sequence. Large word size can miss small matches. Smaller words pick up smaller features. The drawback is that the smallest features are often just noise. For sequence with regions of small matching features- small words pick up small features individually. Larger words show matching regions more clearly. The lack of detail can be an advantage.

Scoring scheme

Scoring scheme is used in the calculation of the score of each word. There are different scor-ing schemes depending on the type of the sequences (nucleotide or amino acid).

DNA Usually, DNA Scoring schemes award a fixed reward for each matched pair of bases and a fixed penalty for each mismatched pair of bases. Choosing between such scor-ing schemes will affect only the choice of a sensible cut-off score and the way ambi-guity codes are treated.

Protein Protein scoring schemes differ in the evolution distance assumed between the pro-teins being compared. The choice is rarely crucial for dotplot programs.

Cut-off score

Cut-off score decides whether the dot at a given position is to be shown on the dot plot or not. The higher the cut-off score the fewer dots will be plotted but, each dot is more likely to be significant. The lower the cut-off score the more dots will be plotted, but dots are more likely to indicate a chance match (noise).

Usages

Detection of deletion, insertion

Plot two sequences retrieved from different sources but supposed to be identical. The dot plot will show insertions and deletions with the broken diagonal line.

Detection of repeats

Put the same sequence on both axes. The dot plot will show repeating sequences beside the main diagonal line.

Detection of stem loops

Put the same sequence on both axes, but in reverse direction on one of them. Stem loops will appear as perpendicular lines to the main diagonal.

B3. What is the difference between global and local alignment? Explain the main algorithms (Needleman&Wunsch and Smith&Waterman), and gap-handling!

Let us consider a pairwise alignment: we have two sequences. The purpose of an alignment is to line up all positions that originate from the same position in the ancestral sequence.

Global alignment (Needleman–Wunsch)

Align two sequences from 'head to toe', i.e. from 5' ends to 3' ends. For this problem an ex-haustive algorithm was published by Needleman and Wunsch in 1970. The rules of the algo-rithm are

- Match at position (i, j) is $M(i, j) = W(i, j)$ where W is a weighting matrix
- Gap is described by a gap function $G = G$initiation + Gelongation
- Score at position (i, j) is $S(i, j)$ which is a maximum of
  - $S(i - 1, j - 1) + M(i, j)$ (diagonal movement)
  - $S(i - 1, j) + M(i, j) - G$ (horizontal movement)
  - $S(i, j - 1) + M(i, j) - G$ (vertical movement)

To align two sequences we write them along two axis of a matrix. We initialize the matrix by adding a 'gap' character to the end of the sequences and calculate the scores in this last row (column). To align the sequences, we fill the matrix step-by step, decreasing i and j values. Also the score and the path of the calculation are stored in a list. To generate the alignment, we trace back the path choosing the maximum of the neighboring cells position-by-position.

Local alignment (Smith–Waterman)

Locate regions with sigh degree of similarity in two sequences. For this problem an exhaus-tive algorithm was published by Smith and Waterman in 1981. The rules of the algorithm are

- Match at position (i, j) is $M(i, j) = W(i, j)$ where W is a weighting matrix
- Gap is described by a gap function $G = G$initiation + Gelongation
- Score at position (i, j) is $S(i, j)$ which is a maximum of
  - $(i + 1, j + 1) + M(i, j)$ (diagonal movement)
  - $S(i + 1, j) + M(i, j) - G$ (horizontal movement)
  - $S(i, j + 1) + M(i, j) - G$ (vertical movement)
  - 0 (so we cannot have negative score values)

We align sequences like in global alignment, except that the score values will always be at least zero.

Gap handling

In the simplest case, every gap has a constant G value, e.g. 1. In this case the length of the gap does not matter.

In the linear case, the gap penalty is a linear function of the gap length, e.g. $G = G\text{length} \cdot K$ where K is a constant, e.g. 1.

In the affine case, gap penalty is a sum of a gap initiation cost and a gap elongation cost: $G = G\text{init} + G\text{extension penality} \cdot G\text{length}$

Also other gap functions are exists like convex gap penalty functions.

B4. Describe the iterative and additive approaches of multiple alignments!

Additive approach ("once a gap, always a gap")

- Do all pairwise alignments (e.g. Needleman-Wunsch)
- Select a simple, ADDITIVE representation for the alignments:
    - For instance: Sequence A in alignment X can be represented as a series of gaps at given positions (pos 4, 5; pos 5: 1). This is a sparse vector.
- Such representations can be added. Sequence A in the multiple alignment will contain the max amount of gaps at each position. This is a special case of vector addition.
- After constructing the multiple alignment in such a way, we need to remove all columns that contain only gaps.

Additive algorithm

1. **Pairwise Comparison**
    - Compare every single sequence to every other sequence, using pairwise sequence alignment

– seq_1 & seq_ 2 ⇒

A	.	A	T
A	C	A	T

– seq_ 1 & seq_ 3 ⇒

A	.	.	A	T
A	C	C	A	T

- Record the resulting gaps in vectors:

For 1 vs 2	0	1	0	0
For 1 vs 3	0	2	0	0

2. **Represent alignments as vectors**
    - Record the resulting gaps in vectors:

For 1 vs 2	0	1	0	0
For 1 vs 3	0	2	0	0

3. **Combine vectors for each pair**
    - Vectors to be added

For 1 vs 2	0	1	0	0
For 1 vs 3	0	2	0	0

- Rule of addition: $sum(x, y) = max(x, y)$

Sum for seq 1	0	2	0	0

4. **Construct multiple alignment**

A	.	.	A	T
A	C	.	A	T
A	C	C	A	T

If necessary, delete empty columns…
Problem: Wrong gaps will be preserved
("once a gap, always a gap…")

Iterative Approach (use "seeds" for avoiding wrong gaps)

- First do all the easy alignments (few wrong gaps). These are the seeds.
- Then gradually add all the difficult ones (distant sequences will form different groups)
- But how do we know what alignments are easy or difficult? Use a tree!

Iterative Algorithm (as applied in ClustalW and similar programs)

- Do a pairwise comparison of all sequences
- From this, calculate how sequences are related to each other (the more similar [pairs with high scores] are easier to align)
- Perform multiple alignment in growing order of scores; the most similar are aligned first, the others are saved for later

Steps

1. **Pairwise Comparison**
   - Compare every single sequence to every other sequence, using pairwise sequence alignment

     seq_1 & seq_2 → 0.91

     seq_1 & seq_3 → 0.23

     …

     seq_8 & seq_9 → 0.87
   - Record the resulting similarity scores
2. **Calculate The Guide Tree**
   - Construct a guide tree from the matrix containing the pairwise comparison values, using a clustering algorithm (Neighbor-Joining (Clustal W))

   Guide Tree - Neighbor-Joining
   - Start with a star tree with N nodes
   - Combine the pair with the smallest branch lengths
   - Continue until all N-3 interior branches are found
3. **Multiple Alignment**
   - Using the guide tree, we start aligning groups of sequences
   - The purpose of the guide tree is to know which sequences are most alike; so we can align the "easy" ones first, and postpone the tricky ones to later in the procedure!
   - The most similar sequences give the "seeds", CLUSTAL combines the seeds.

B5. Why do we need databases in biology, what are the main tasks of databases? What is their technical and logical structure? What are the main data-types (give examples to them) and data description formats (syntax, semantics)?

Databases in bioinformatics

The place of databases within bioinformatics

Algorithms and software are for gathering knowledge. Databases store and communicate this knowledge as a kind of 'message' (collection of messages). All messages put (map) knowledge items into a standardized form (database record) and use one or more standardized languages. So databases have syntax and semantics, often formalized as ontologies.

In brief, biological databases are searchable organized, regularly updated datasets which are specialized on certain data-types, contain textual info and often associated with special com-putational methods (e.g. similarity search).

Main tasks of databases

The main tasks in database construction are

1. Data collection and storage
2. Validation
3. Clustering (classification and redundancy filtering)
4. Annotation
5. Integration, visualization

Technical and logical structure

Technical structure

Basically the database contains units, which are records, containing fields. Fields can be mandatory or optional. Hierarchical order.

Logical structure

Database is built from metadata and data. Metadata is the 'data on data', e.g. cross ref-erences to other databases.

Main data types and description formats

Main data types of databases

- Nucleotide databases – *GenBank, DNA Data Bank of Japan*
- Protein sequence databases – *UniProt, Swiss-Prot*
- Genome databases – *MGI Mouse Genome, Wormbase*
- Others: proteomics, protein structure, protein model, RNA, carbohydrate, pathway etc.

Description formats

Syntax – 'grammar'

Basically, two types of alphabets: for DNA, RNA, nucleotide alphabets, with 4 main charac-ters, for protein, amino-acid alphabet, with 20 main characters. Wild cards can be added to the alphabets for residue groups and modifications. Today, databases use one letter codes. E.g. FASTA format (or other better readable formats mostly for visualization).

Metadata also have syntax; there are different languages for different purposes.

Semantics – 'sense'

Standardized concepts plus standardized language equals ontology. The simplest form is a standardized language plus added keywords.

The Gene Ontology (GO) project standardizes the names and functional description of genes and proteins, in the form of concept hierarchies.

B6. Give a short overview of the statistics of alignments (e-value, p-value, z-score, linearization, Karlin-Altschul statistics)!

Significant scores are different, some are bounded, some are not. We need to bring them to a common scale. The natural scale is probability, and this probability is called significance, well known from simple statistics. We have to tell how much different is the score that we found from scores obtained at random i.e. by chance. For this, we need to know the distribution of random similarities, or simulate them in some way.

p-value

The p-value is the probability that an alignment with this score occurs by chance in a database of this size. The closer the p-value is towards zero, the better the alignment. You can estimate p by making a histogram of random scores, linearizing it and reading p from the linear curve.

e-value

Number of matches with this score one can expect to find by chance in a database of size N. The e-value can be calculated from the p-value as $e = pN$. The closer the e-value is towards zero, the better the alignment.

Z-score

Calculating significance for comparing two sequences, A and B:

1. Calculate score comparing A and B. This will be the genuine score, S_{genuine}
2. Repeat $N > 100$ times. In every step,

    a. randomize sequence A by shuffling its residues in a random fashion.

    b. align randomized sequence A_{rand} to B and calculate alignment score S_{rand}

3. Calculate mean and standard deviation of the ransom scores: $\tilde{S}_{\text{rand}}$, σ_{rand}. The Z-score is calculated from the expression below:

$$Z = (S_{\text{genuine}} - \tilde{S}_{\text{rand}}) / \sigma_{\text{rand}}$$

In practice, Z-score is meaningless: it implies a normal distribution of scores which is not true, shuffling sequence A won't give us other compositions and poly-nucleotide sequences ('TTTT') remain the same at randomization so σ_{rand} will be close to zero. Furthermore, natu-ral sequences are not like random shuffled ones, some di- or trinucleotides are more frequent.

Linearization

Linearization is a method to estimate significance based on comparison with a histogram of unrelated similarity scores without knowing the distribution. To obtain a significance value, we draw a percentage histogram of chance similarities (probability vs. score). We have to linearize it (log-lin, lin-log, log-log transformations). After fitting a straight line onto the line-arized data points, for every score on the x-axis the significance value can be read from the y-axis. Usually one has to extrapolate quite far since large scores are rare.

Karlin–Altschul statistics

The Karlin–Altschul statistics is based on Extreme Value distribution. The expected number of high scoring segment pairs with score at least S is $E = Kmne^{-\lambda S}$

where m is the query length, n is the database length, K and λ are constants. Kmn is called the search space. The probability of finding at least one high scoring segment pair with the score S or greater is $p = 1 - e^{-E}$

where E is the expect value. Raw scores can be normalized to bit scores using the following formulas: $S' = (\lambda S - \ln K)/\ln 2$, $E = mne^{-S'}$

B7. What is similarity searching in a database, what are the main steps, the importance of it, what kind of heuristics do you know?

Given a query and a database, find the entry in the database that is most similar to the query in terms of a numerical similarity measure (distance, similarity score, etc.). In contrast: retrieval looks for an exact match to the query. Similarity searching is important in identification and automatized annotation of unknown proteins.

Steps of similarity searching

0. The query and database has to be in the same format and we need a similarity measure

1. *Compare* query with all entries in the database and register similarity score. Store results above some threshold (cut-off)

2. *Calculate significance* of the score (compared to chance similarities, e.g. Karlin–Altschul)

3. *Rank* entries according to similarity score or significance (top list)

4. *Report* the best hit (usually after some statistics, e.g. thresholding) add alignment pattern

5. *Assume function* of query, i.e. classify query into a class present in the database. Align-ment pattern is a confirmatory proof for classification.

Heuristics

Exhaustive algorithms are expensive on large sets of data. Therefore we use search heuristics in order to reduce the search space. There two main categories of the heuristics that we use:

- *Computational heuristics:* a) exact matching is fast b) search space can be easily reduced.
- *Biological heuristics:* a) local similarities are dense b) similar regions are near each other, c) low complexity sequences excluded, etc.

Computational heuristics

Word based methods

Exact word matching is fast, use it whenever possible. Word composition vectors are a possi-bility but word (n-gram) dimensionality grows fast with word size. Longer words are more informative but few of the possible words occur in a given sequence and most of them only once. We can make use of the longer words if we look only for shared words but there are much fewer of these. Word indexing methods, such as hash tables are extremely fast but need to build them previously.

Search space reduction

Most sequences in a database are not similar to a given query, we can filter the out. Threshold based filtering is important, but has drawbacks (you can throw away important hits). Two step workflows: Use

fast methods to filter out most of the dbase entries, and use increasingly ex-pensive methods only for the important ones. Example for two typical filters:

- Keep only sequences that contain n common words with the query
- The shared words should be in the right distance or in the right serial order.

Biological heuristics

Search space reduction

If you absolutely need dynamic programming, search only the vicinity of the diagonal.

Short conserved sequences

Sequences evolved from same roots always contain highly similar conserved regions. Keep the ones that have these regions and omit others. But it is difficult to define, what is a con-served region and what is a good threshold.

Repetitive or 'low complexity' regions

Biological sequences often contain repetitive of 'low complexity' regions. These can be ex-cluded from the query by masking them in advance with X-es in order to omit them from the entire calculation.

B8. Describe BLAST algorithms in details! What kind of heuristics does BLAST use? How can the BLAST specificity be improved? List 5 different BLAST programs (query type)!

The word BLAST stands for Basic Local Alignment Search Tool. The input of a BLAST search is a query sequence and a sequence database in preprocessed form or in 'normal' form for presenting the alignments using Smith–Waterman. The outputs of the algorithm are a top-list of most similar sequences, ranked according to and alignments (Smith–Waterman).

BLAST algorithm

1. Store database in a hash table with n-mert words and occurrences (preprocessing). The value of n is usually 11 for nucleotides and 2-4 for amino acids.
2. Record words in a query, includes similar words, based on BLOSUM similarity
3. Select database entries that share a given number of common words with the query (fast)
4. From here, select those where the shared words densely cluster in certain regions ('seed' of 'high scoring pair', HSP)
5. Elongate HSP-s and splices them together by some criterion
6. Calculate significance of the spliced regions using the BLAST formula
7. Rank the top scoring entries by significance and present the score and the motif.

Tricks and heuristics

- Two-pass search. Hash table preprocessing of database – allows a fast first-pass scan of the database.
- Extending query representation with 'similar words' (e.g. AG → AG, GA, AA, GG [sup-posing word size is 2 and A∼G is a similarity relationship) makes first pass sensitive.
- Looking for HSPs (conserved regions): fast, gapless search
- 'Seed and extend' strategy: approximate step for finding the alignment region and its score
- Good statistics (Karlin–Altschul) gives 'real' significance of alignment region.
- Smith–Waterman alignment: time consuming step restricted to only the alignment region

Improving specificity

Specificity can be improved by the removal of aspecific (biased composition) regions. Repeti-tive sequences will specifically match with many queries. Sequence complexity is an empiri-cal measure, proportional to the number of words (of arbitrary length) necessary to reproduce a sequence. Low complexity regions have a biased composition, they are often very repetitive. These low complexity

regions can be removed (above a threshold) replaced by X so that they will not take part in the alignment (SEG program). But, some interesting sequences are of low complexity.

BLAST query types

program	query		database
blastp	protein	vs.	protein
blastn	nucleotide	vs.	nucleotide
blastx	nucleotide → protein	vs.	protein
tblastn	protein	vs.	protein ← nucleotide
tbalstx	nucleotide → protein	vs.	protein ← nucleotide

B9. Describe the phylogenetic method Parsimony (character set, homology and homoplasy, character fit)! Define the exhausted and heuristic tree generation!

Parismony is an easily understandable phylogenetic method close to numerical taxonomy. It di-rectly deals with the characters and their states. Parsimony is one of the most popular methods.

Definitions

Characters

Characters are a set of homologous features. Character state is the manifestation of a feature. Character states are coded into table and they are mostly categorical data. E.g. a character can be '3 toed foot' and the character state describes if the species has or not has this property.

The best characters are unique (cannot arisen multiple times) and unreversed (cannot easily mutate back to the original state).

Presence of a character is a sign of common ancestry, absence of it means a divergence before the emergence of the character.

Homology

Homology is the existence of shared ancestry between a pair of structures, or genes, in differ-ent species

Homoplasy

Homoplasy is a similarity that is not homologous. It comes from independent evolution or back mutation.

Character fit

Initially, we can define the fit of a character to a tree as the minimum number of steps re-quired to explain the observed distribution of character states among taxa

The essence of parsimony

Parsimony is used to evaluate alternative trees (which was generated by some other method since parsimony does not create trees). Parsimony helps to find homoplastic characters. The most parsimonious trees have the minimum tree length and a weighted sum of the cost of each character is low. The result of parsimony analysis is a set of trees from which one or more best trees can be taken for further analysis.

Advantages

Simple method, easy to understand; does not depend on an explicit model of evolution; gives both trees and associated hypotheses of character evolution; reliable results if data is well-structured and homoplasy is either rare or widely distributed on the tree.

Disadvantages

Misleading if homoplasy is common or concentrated in particular parts of the tree; underest i-mates branch lengths.

Exhausting tree generation

Exhaustive (not exhausted as the topic title writes) tree generation means tree generation in all the possible ways. For 2 to 8 taxa it works well but above 12 taxa very high amount of re-sources are needed.

B10. Describe the distance based phylogenetics! List some nucleotide substitution models! What is the connection between them? Describe the UPGMA and Neighbour-joining algorithms at tree generation!

Distance base phylogenetics

The distance based methods are based on the calculated pairwise distances between sequences by selecting a model, eg. JC69 or K80. From the distances we create a distance matrix. This matrix will undergo some tree-building methods such as neighbor-joining.

Nucleotide substitution models

Nucleotide substitution models are models for DNA sequence evolution. There is solid math-ematics behind them. The most important differences between models are nucleotide frequen-cies and mutation rates.

Nucleotide frequencies can be ¼, measured from data or estimated with mathematical models.

Mutation rates can be uniform, constant or changing in time, etc.

Jukes-Cantor model (JC69)	TN93 model
Equal base frequencies Single overall mutation rate	Unique base frequencies Multiple mutation rates
HKY model	**T92 model**
Unique base frequencies	GC content
K80 model	**Generalized time-reversible model (GTR)**
Different mutation rates for different transi-tions and transversions	Unique base frequencies All mutation rates specified

Neighbor-joining

Neighbor-joining is an iterative algorithm that constructs the tree stepwise. It uses total branch length for evaluating the trees.

The algorithm starts with a star-shaped tree. In each step, join the two taxa which have the smallest branch lengths. Continue this iteration until all $N - 3$ interior branches are found.

The neighbor joining algorithm produces an unrooted tree. This method does not assume equal rates. Neighbor-joining is a quick and good guess of true phylogeny.

UPGMA

“The UPGMA (Unweighted Pair Group Method with Arithmetic mean) method produces rooted trees and require a constant-rate assumption - that is, it assumes an ultrametric tree in which the distances from the root to every branch tip are equal.” - *wikipedia*

B11. Explain the basics of next generation sequencing (definitions, file formats)! Compare the traditional (Sanger) sequencing and NGS approaches! Describe assembling strategies: hierarchical for long, de Brujin graph for short reads, scaffolding and the difficulties of assembling!

Definitions, file formats

Definitions

Read – A single piece of output by a sequencer (typically a 50-500bp long DNA sequence)

Coverage – The number of times a (genome) sequence is covered with reads. Sequence cov-erage is the fraction of the genome covered by reads.

Fragment library – A short insert (200bp) library of reads with overlapping ends

Long insert library – A 4-8 kb library of reads where only 100 bp on each end are sequenced. Aka CLIP, mate pair library. Contains unsequenced parts in the middle!

Contig – A contiguous sequence of DNA (assembled from single reads)

Scaffold – One or more contigs linked together by unknown sequence segments

Captured gap – A gap within a scaffold. The order and orientation of the contigs spanning the gap is known

File formats

The following file formats are widely used in sequencing, the reads are stored in the fotmats below:

FASTA – general sequence format, only the sequence, no sequencing quality information

Illumina qseq – includes all called sequences and a quality filtering flag (0 is bad, 1 is good)

Solexa/Illumina – Based on FASTQ with a different quality encoding

Traditional sequencing vs. NGS

Traditional sequencing	**Next Generation Sequencing**
- accurate	- less accurate (sometimes much less)
- also works on few samples	- shorter reads (in general)
- expensive for data	- economical only with many samples
- small capital investment	- 1000 times less expensive for data
- slow	- large capital investment
	- very fast

Essence

Assembling strategies

The problem of genome assembly is to recover the original sequence of bases of the genome. There is generally no other information available. The two main assembling strategies are

Traditional (hierarchical) way: long and accurate subsequences from subclones, assembly by exhaustive sequencing (Smith Waterman)

Next generation sequencing: Shotgun sequencing of the whole genome, assembly by graph-based methods.

Hierarchical strategies for long reads

1. Split the larger sequence into 40-200 kb segments.
2. Amplify using bacterial artificial chromosome (BAC)
3. Sequence and assemble each segment independently
4. Assemble segments

Short read strategies (De Brujin graph)

The sequencing is done by hybridization of short sequences. We use graph methods to assemble the genome. The reads are subsequences of the same sequence. Therefore the edges of the graph will be the overlaps (and the nodes will be the reads themselves). The genome sequence is a path in this graph.

The De Brujin graphs are built from sequences of the same length from a long text. Each k-mer is connected to the next window which gives a directed graph. The graph has one unique long path: this is the text (in our case the genome) itself. So we can find the genome string by finding an Euler-walk which is not an NP hard problem.

Scaffolding

In practice the sequence is broken up into k-mers. The graph is built from these k-mers. When we walk through a way in the graph from going along the edges connecting connecta-ble sequences we can get contig. These contigs can be combined into scaffolds by using dif-ferent k sizes and some libraries.

Difficulties

- Reads can contain errors
- Overlaps can be very short, at times even missing
- Reads can come from different strands
- There are repeats in eukaryote genomes
- Some sequences may be missing
- There is a huge (very huge) number of reads

B12. Describe the Hidden Markov Model! (What is the connection between protein profiles and Hidden Markov Model?)

Hidden Markov Model

The Hidden Markov Model (HMM) has the following components

Hidden states – states of the model describing what output matrix is used

Observable symbols – symbols that the HMM may emit

Output matrix – probability of emitting an observable symbol in the current hidden state

Initial distribution – probability of the model being in a certain hidden state at $t0$

State transition matrix – probability of moving from one hidden state to another hidden state

Just like the Markov Chain, the Hidden Markov Model is representable with a graph. The transitions are described by the transition matrices and the next state depends only on the current state (this is called the Markov property).

HMM analyzing tools

Viterbi algorithm

Viterbi algorithm calculates the most probable path $\pi*$ based on the transition probabilities and a series of emitted (observed) symbols. The algorithm uses iterative steps; it can be easily implemented by dynamic programing. The iterative step is $vl(i + 1) = el(x_{i+1})$ max k $(vk(i)$ akl)

where $vk(i)$ is the probability that the most probable path ends at state k, where it emits symbol xi. The el (x) is the emission probability of x in state l and akl is the transition probability from k to l.

Baum–Welch method

Usually we do not know the parameter of the HMM, so we want to estimate them. The Baum–Welch method is an iterative expectation maximization algorithm. It starts with random state series whose length matches the average length of the sequences, then after alignment, the parameter estimation is done. Iterated until the model stops changing.

Connection between protein profiles and HMM

Given a protein family and we want to find other members of this family in a given database. We can make a profile that represents the protein family and we compare the database elements to this profile. An HMM can describe such a profile:

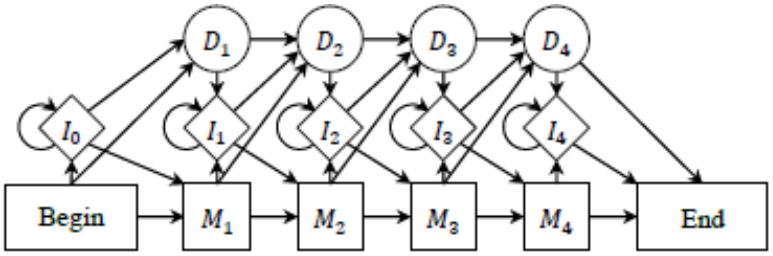
Hidden states – M (match), I (insertion) and D deletion.

Observable symbols – the amino acids

Output matrix – It describes the probability of the emitted or inserted symbols in state *M* and *I*. In state *D* there is no emission.

Initial state – probabilities according to the distribution function of the first amino acids

State transition matrix – problem-specific matrix containing the probability of match, insertion or deletion



B13. What are the main steps and difficulties and tricks of prokaryotic gene fining? What are the main differences in case of eukaryotic genes? Mention some software tools for these tasks!

Gene finding in prokaryotes

High gene density and simple gene structure (mostly ORF).

Short genes have little information.

Overlapping genes.

Example of tools suitable for gene prediction in prokaryotes:

Glimmer 3

GeneMark

MED 2.0

Steps

Finding open genes:

- We use a presumed syntax, the codon table.

Translation programs give you 6 reading frames

- But ORFs generally overlap

- Longest ORFs likely to be protein-coding genes

The Problem

Need to decide which ORFs are genes.

- Then figure out the coding start sites

Can do homology searches but that won't find novel genes

- Besides, there are errors in the databases

Generally can assume that there are some known genes to use as training set.

-Or just find the obvious ones

Gene finding in eukaryotes

Low gene density and complex gene structure.

Alternative splicing.

Alternative start and stop.

Pseudo-genes

Ab initio methods:

Based on statistical signals within the DNA:

- Signals: short DNA motifs (promoters, start/stop codons, splice sites, ...)
- Coding statistics: nucleotide compositional bias in coding and non-coding regions

Strengths:

- easy to run and fast execution time
- only require the DNA sequence as input

Weaknesses:

- prior knowledge is required (training sets)
- high number of mispredicted gene structures

Ab initio methods classification and examples

Generalized HMM (Hidden Markov Models)

- GenScan
- HMMgene
- Augustus
- SNAP

Linear and Quadratic discriminant analysis

- FGENES and derived versions
- MZEF

Decision trees

- MORGAN

Neural Networks (NN)

- GRAIL

Support Vector Machines (SVM)

- CONTRAST
- mGene

Main differences:

Prokaryotes are simple, classification of ORFs with HMM-style methods is sufficient (GLIMMER).

In eukaryotes, short signals are more important, exon/intron structure given problems.

Eukaryote methods are similar to those in prokaryotes but there are no single best methods.

The use of several gene finders and combination of the results is recommended.

B14. Briefly describe RNA-seq, ChIP-seq, GRO-seq and ChIA-PET techniques (applications, input, output data, and steps of the typical workflow)!

ChIP-seq (ChIP-chip) – DNS-fehérje kölcsönhatások genomszintű vizsgálata

RNA-seq – mRNS-ek genomszintű vizsgálata

GRO-seq – transzkripció genomszintű vizsgálata

ChIA-PET – DNS hurkok vizsgálata

A DNS-t vagy az RNS-t feldarabolják, majd a kis darabokból „tag”-eket szekvenálnak, és azokat illesztik a referencia genomhoz

Mire lehet az RNA-seq-et használni

Teljes transzkriptom analízis

- Új transzkriptek (átíródó régiók) meghatározása
- Splice variánsok meghatározása

Expressziós profil meghatározása (egyes transzkriptek abundanciájának a mérése)

Genetikai polimorfizmusok meghatározása:

- SNPs, micro-indels
- CNVs

Az RNA-seq-et lehet használni referencia genom nélkül is, akkor a transzkriptomot rakjuk össze de novo

Trancriptome analysis with RNA-seq

- Total RNA, rRNA depleted or mRNA isolation
- Fragmentation
- cDNA synthesis
- Library preparation
- Sequencing one or both ends of the fragments

RNA sequencing

- Samples of interest
- solate RNAs
- Generate cDNA, fragment, size select, add linkers
- Sequence ends

- Map to genome, transcriptome, and predicted exon junctions
- Downstream analysis

RNA-seq analízis pipeline

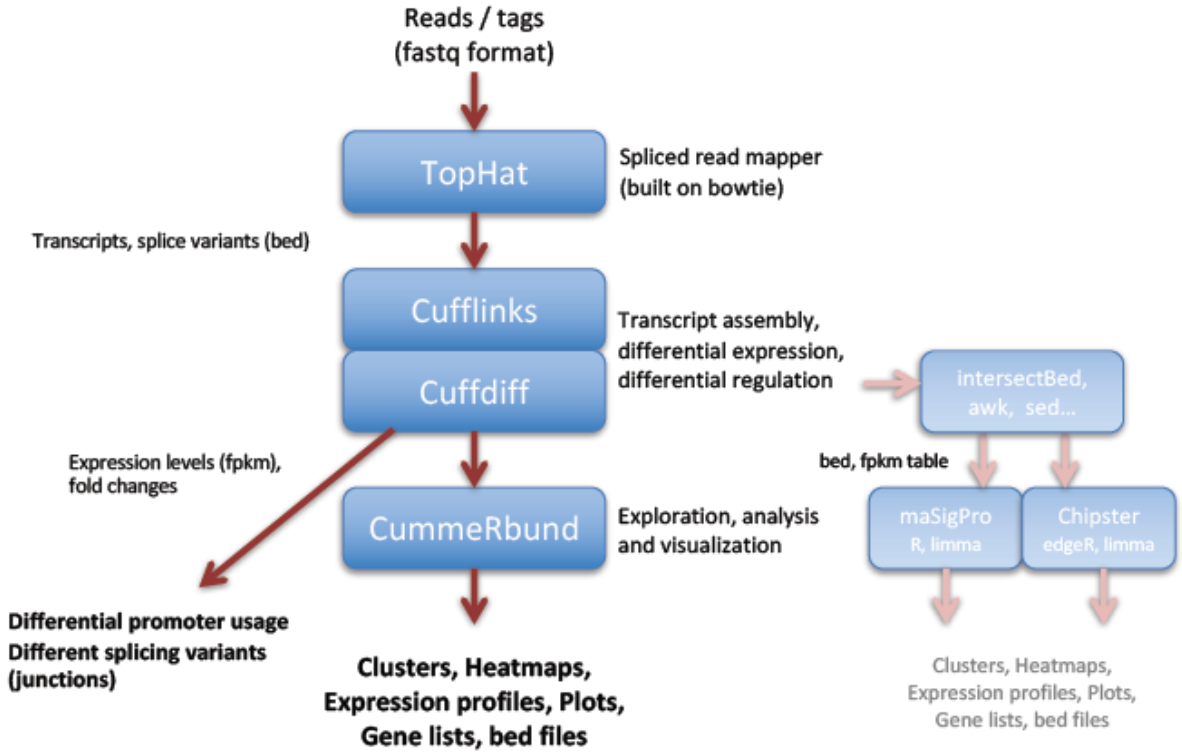
Bowtie illesztő (tophat) + cufflinks + CummeRbound

- Ez a leggyakrabban használt csomag

TRINITYRNASEQ de novo transzkriptóm összerakáshoz

- Nagy a memóriaigénye (90GB<: HPC a Pécsi
- Akkor hasznos, ha nincs referencia genom (és/vagy transzkriptóm)
- Normalizált számlálása a transzkriptekre eső read-eknek génexpresszió (probléma: egy génhez több transzkript is tartozhat)
- A kivágódási (splice) helyeket átfogó read-ek exon junctions alternatív splicing
- Alternatív promóterek

Eredmény: A differenciáltan kifejeződő gének listája expressziós értékekkel, új transzkriptek listája (új splice formák)



Mi az a ChIP-szekvenálás?

- DNS-fehérje kölcsönhatások vizsgálata kromatin immunoprecipitációval és NGS

szekvenálással

- A ChIP-szekvenálás egy új forradalmi technológia, amellyel fehérje-DNS interakciókat lehet térképezni genomszinten.

ChIP-Seq:

A kromatin immunoprecipitáció (ChIP) kombinálása az újgenerációs szekvenálással

- A génregulációban kulcsszerepe van a kromatin szerkezetének, a transzkripciós faktorok kötődésének

-A módszer a kromatin vagy a transzkripciós faktorok DNS-hez kötésén, majd specifikus antitesttel történő kicsapásán alapul.

GRO-seq – Global Run-On (Massive) Sequencing

- GRO-Seq detects all transcribing RNAs (coding RNAs, ncRNAs, enhancer/e-transcripts) in the nuclei from the 5’ to the 3’ end (strand specific)

- Theoretically, one tag (read) in the GRO-seq data represents one RNA polymerase molecule

RNA polymerase I synthesizes pre-rRNA

RNA polymerase II synthesizes precursors of mRNAs most snRNA and miRNA

RNA polymerase III synthesizes tRNAs, rRNA5S and other small RNAs

RNA plymerases IV and V are in plants, but no GRO-seq from plants so far

GRO-seq analízis pipeline

Alapvetően hasonló megközelítés mind az RNA-seq analízisnél

1. A read-ek számolása transzkripteken

Az intronokon is lesznek read-ek!

A génen lévő enhenszerek (ott külön transzkripció van) torzíthatják az eredményt

Hosszú géneknél befolyásolhatja az eredményt a transzkripció sebessége

2. Új transzkriptek keresése (mivel minden transzkripciót mutat)

Lehetnek kódoló gének, génváltozatok

Lehetnek nem-kódoló RNS-ek (ncRNA, lncRNA)

Enhenszer transzkripció

Mi a közös a három funkcionális genomikai módszerben bioinformatikai szempontból

- Mind a genomi DNS-t (ChIP-seq), mind az RNS-t (RNA-seq), mind a naszcens RNS-t (GRO-seq) 100-300 bp-os darabokra fragmentáljuk

- A könyvtárkészítés után hasonló DNS fragmenteket kapunk, amelyeket szekvenáló linkerek, primerek határolnak

- Egy (ChIP-seq, GRO-seq) vagy mindkét végét (RNA-seq) ezeknek a fragmenteknek megszekvenáljuk

- A következő lépés mindig a referencia genomra illesztés (kivéve RNA-seq de novo assembly)

- A ChIP-seq és GRO-seq esetében a cél csak a megszekvenált rövid read-ek pontos genomi localizációjának a meghatározása (elég 30-50 bp-t meghatározni)

- Az RNA-seq esetében kíváncsiak lehetünk a különböző splicing variánsokra, ezért célszerű hosszabb szekvenciákat leolvasni a fragmentumok mindkét oldaláról

ChIA-PET Analysis pipeline

Download data (SRA/ ENCODE)

Filter data (300kbp < interactions)

Assign CTCF motifs to interactions

Interaction analysis: Visualization, Meta analysis, Heatmap analysis, Annotation