Part 1: Genome and Sequences

▼ *Contents*

1.0. DNA Sequences

- **Genome**: DNA molecules storing the information required to reproduce organisms
- 4 ATGC (**necleotides**) paired into triplets (**codon**) codes for 20 amino acids
  - Amino acids build proteins *(consisting of hundreds or thousands of amino acids on average)*
  - Start position and direction matter
- Genes are structured sequences, with non-coding and regulative regions (*exons*, *introns*, *promoters*)

- Models simplify and aid analysis: Genetic code and "gene" are *models*

  > *“Genes are responsible for proteins”*

- Genomic sequence string uses:
  - Collect and annotate genomes
  - Count bases/letters, find genes, compare sequences
  - Assemble fragments, identify differences
- *Objectives*:
  - Locate genes, patterns, functional sites in encoded proteins
  - Predict (3D) protein structure, ...

Note: While the genetic code is largely universal, there are a few instances of variations known as *codon reassignments* or *codon usage difference*. In some organisms/organelles, some codons may encode different amino acids with the same necleotide sequence than the standard one.

Human Genome Project

- Collected ~3 billion letters of the human genome sequence
- Identified and localized all 20.000-30.000 genes in the human genome
- Another key finding: The genome has significantly more *segmental duplications* than had been previously suspected.

Elementary String Algorithms

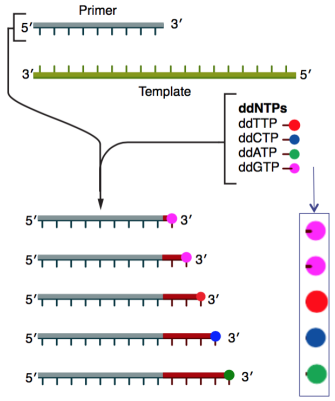
Assume two strings a, b with lengths N, M.

- Character-wise comparison ($M = N$) (→ Run-time: $O(N)$)
- Elementwise pattern-matching ($M < N$) (→ Run-time: $O(MN)$)
- *Boyer-Moore*: uses information gained by preprocessing the string (+ some rules) to skip as many alignments as possible

1.1. Sequence Alignment

How does **sequencing** work?

1. Primer sequence finds the starting point in the genome.
2. Primer is extended with complementary bases (→ *mirror-sequence* of DNA).
3. Results in sequences of different lengths with same primer, but different complementary bases: Couple (last) complementary base with a fluorescent molecule specific to the base.
4. Sort sequences by length.



Biological sequences are never exactly equal (due to errors, mutations, etc.):

- *Alignment problem*: find match with highest degree of similarities
- **Single-Nucleotide Polymorphism** (SNP; difference in one nucleotide) or gaps can have different severity on the encoded protein

1.1.1. Scoring

We utilize *scoring functions* in order to quantify the quality of an alignment.

- Scoring functions as a model of the biophysical situation derived from biological observations (impact of certain amino acids on the protein structure; similarity between amino acids)

Scoring Scheme:

- **Substitution Matrix**: Holds *alignment scores* for pairs of amino acids, where a positive entry indicates, that the substitution pair appears more often than random
  - $s(A, B) = \log(p_{AB}/q_A q_B)$, where p_{AB} is the observed pair frequency (given a good alignment), and q_A,q_B are the natural frequencies of amino acids A, B ($\sim 1/20$)
- **Score**: $\sum_i s(A_i, B_i)$, where $s(A_i, B_i)$ is an entry in the substitution matrix

- **Gap Penalty**: Additively penalize each gap position with a fixed, *positive* value d

Note: With $C_{min} = max\{M, N\}$ and $C_{max} = M + N$ *(everything matches to a gap)*, for the optimal alignment length it holds that $C_{min} \leq L \leq C_{max}$.

1.1.2. Algorithms for Finding the Optimal Alignment

Brute-force

Calculate the score for all possible alignments (→ number of alignments is exponential in length of sequence!)

Since the scores are computed *additively* and *independently* for each pair AND the true alignments are much more probable than random, we can utilize *Dynamic Programming* to compute best alignments of prefixes $F(i, j)$ between $a[:i]$ and $b[:j]$ using either letter matches or gap/letter to letter/gap matches.

Needleman-Wunsch Algorithm

Algorithm assumes *global* alignment, where sequences match end-to-end.

```
for (i in 0..M) F [i][0] = -i * d;
for (j in 0..N) F [0][j] = -j * d;
for (i in 1..M)
        for (j in 1..N)
                F [i][j] = max (
                        F [i-1][j-1] + s (a [i-1], b[j-1]),
                        F [i-1][ j ] - d,
                        F [i]  [j-1] - d);
```

a = HEAGAWGHEE
b = PAWHEAE

Optimal global alignment:

HEAGAWGHE -E
- - P -AW- HEAE

		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-41	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	-20	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	-5	-13	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	-3	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	3	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	-5	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	1

- First column and row are negative except for $F[0,0]$ to ensure, that the algorithm can distinguish between gaps introduces at the begining of the sequence and mismatches/gaps introduced during sequence alignment.
- Bottom-right corner of matrix is *end* of *optimal* global alignment with the corresponding score if the alignment
  - *Computation by hand*: All F-values needed for computing $F[i,j]$ lie within a 2×2 square with $F[i,j]$ being in the bottom-right corner
  - Recover the alignment via backtracking from $F(M,N)$ (multiple solutions are possible)

Smith-Waterman

Algorithm assumes *local* alignment of common subsequences

```
for (i in 0..M) F [i][0] = 0;
for (j in 0..N) F [0][j] = 0;
for (i = 1; i <= length (a); i++)
        for (j = 1; j <= length (b); j++)
                F (i, j) = max (0,
                        F (i-1, j-1) + s (a[i-1], b[j-1]),
                        F (i-1, j) – d),
                        F (i, j-1) – d));
```

Alignment ends in maximum F

Regard only local matches with positive scores (Random s(a,b) must be negative, at least one must be positive)

Optimal local alignment: AWGHE / AW -HE

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	5	0	0	0	0	0
W	0	0	0	0	2	0	20	12	4	0	0
H	0	10	2	0	0	0	12	18	22	14	6
E	0	2	16	8	0	0	4	10	18	28	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	6	13	18	12	4	0	4	16	26

- *Score interpretation*:
  - Introduce *no penalty* for gaps at begin/end
  - < 0 scores: clipped to 0, meaning no alignment is better than a bad one, since a negative score would imply that this alignment is worse than having no alignment at all

- > 0 scores: *possible* local alignment; highest possible score (from right and bottom border) is *end* of *optimal* local alignment

Repeated Local Alignment

Recover *multiple* local alignments by introducing a score threshold to be exceeded in order to get non-overlapping alignments.

```
F (0, j) = 0;
for (i = 1; i <= length (a); i++)
        F (i, 0) = max (F (i-1, 0), F (i-1, j) – T);
        for (j = 1; j <= length (b); j++)
                F (i, j) = max (F(i, 0),
                        F (i-1, j-1) + s (a[i], b[j]),
                        F (i-1, j) – d),
                        F (i, j-1) – d));
```

Positive threshold T (= 20)

Asymmetric

Going back from F(n+1,0)

HEAGAWGHEE
HEA AW -HE

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	1	1	1	1	1	3	9
P	0	0	0	0	1	1	1	1	1	3	9
A	0	0	0	5	1	6	1	1	1	3	9
W	0	0	0	0	2	1	21	13	5	3	9
H	0	10	2	0	1	1	13	19	23	15	9
E	0	2	16	8	1	1	5	11	19	29	21
A	0	0	8	21	13	6	1	5	11	21	28
E	0	0	6	13	18	12	4	1	5	17	27

Alternative Gap Penalty Formulation

Instead of an additive penalty for gaps, a more appropriate scoring mechanism would be to penalize each subsequent gap lower, resulting in lower penalties for longer gaps (*gap extension penalty*).

- **Affine Gap Scores**: Let d initial gap penalty, e penalty for each extension. The DP algorithm with $O(n^2)$ runtime would utilize three alignment score matrices:

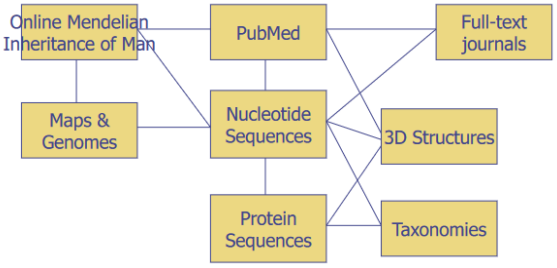
```
# best score given that a[i] and b[j] match
M(i,j) = max(M(i-1,j-1) + s(a[i], b[j]),
                        Ia(i-1,j-1) + s(a[i], b[j]),
             Ib(i-1,j-1) + s(a[i], b[j]))
```

```
# best score given that a[i] is matched to a gap
Ia(i,j) = max(M(i-1,j) - d,
              Ia(i-1,j) - e)

# best score given that b[j] is matched to a gap
Ib(i,j) = max(M(i,j-1) - d,
              Ib(i,j-1) - e)
```

1.2. Databases and Heuristic Algorithms

For the purpose of the Human Genome Project the acquired data had to be stored in *global databases*. Central databases are the "Nucleotide Sequences".



Databases at the NCBI

1.2.1 Sequence Databases

Sequence databases store any sequences and fragments, that have been found, uniquely identified with an **accession key** of the entry, while having attributes (e.g. source, species, *sequence string*).

It is more beneficial for the accession key to be unique, but *not specific*, since information about the protein and protein family might change over time.

```
XX
AC   X04751;
XX
SV   X04751.1
XX
DT   07-JUN-1987 (Rel. 12, Created)
DT   10-FEB-1999 (Rel. 58, Last updated, Version 5)
XX
DE   Rabbit alpha-1-globin gene to theta-1-globin pseudogene region
XX
KW   alpha-1-globin; alpha-globin; globin; pseudogene; repetitive sequence;
KW   tandem repeat; theta-1-globin; theta-globin.
XX
OS   Oryctolagus cuniculus (rabbit)
OC   Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia;
OC   Eutheria; Lagomorpha; Leporidae; Oryctolagus.
XX
RN   [1]
RP   1-4028
RA   Hardison R.C.;
RT   ;
RL   Submitted (02-FEB-1987) to the EMBL/GenBank/DDBJ databases.
RL   Hardison R.C., Pennsylvania State University, Althouse Laboratory,
RL   University Park, Pennsylvania 16802, USA.
```

The database entries have the (original) punch-card structure, where each row is identified with an abbreviation of some attribute (e.g. AC), followed by the attribute. Each line has at most 80 characters.

Having an unified database entry structure allows for the use of programs, that read those entries (e.g., for a graphical view).

One database application is to create a **Gene Ontology** to annotate and describe biological functions, in particular, to find genes related to some term.

1.2.2. Finding Related Sequences

Another typical use of the database is to find related sequences to a target sequence. While processing all entries in the database for that obviously computationally expensive, we fall back to faster alignment algorithms (**FAST**, **BLAST**), that don't find optimal alignment, but rather detect (sufficient) similarity in *ungapped* segments.

Considerations

- *Low-Complexity Regions*: only few different amino acids are repeated at high rates, which might produce high scoring hits; filter before query (since these are "assumed" to be non-functional)
- Substitution matrices might be different depending on how much change in the sequences we expect (e.g., between species)

  → PAM60 for dissimilar sequences, PAM120 for related ones

BLAST

Find all high-scoring segment pairs (aligned subsequences without gaps):

1. Find all words (with fixed length), that match somewhere in the query sequence with score > T.
2. Find 1:1 occurrences of these words (*seeds*) in the comparison string.
3. Extend seeds in both directions until score drops below a fraction of the maximum so far.
4. Report all segment pairs with score > S.

FAST

Find offset between query and comparison string resulting in high-scoring alignment:

1. Preprocess query by looking up tuples (length 1 or 2) and record offset (postion in sequence).
    - A (2, 6, 7), I (9), Q (8), V (10), ...
2. Scan database string and count offsets (*"diagonal method"*, since we find diagonal of DP matrix only). A good alignment would have a high offset count.
    - t = VDMAAQIA (target)
    - Pos 1 (V): [10-1] = [9]; Pos 4 (A): [-2, 2, 3] Pos 5 (A):[-3, 1, 2]; Pos 6 (Q): [2]; Pos 7 (I): [2]; Pos 8 (A): [-6, -2, -1]
    - Offset 2 occurs 4 times.
3. Join tuples with same possible offset into gapless regions, and rescore regions with substitution matrix. Recalculate using DP restricted to a *band* around the diagonal found.

1.3. Recognizing Signals

Signals are *functional parts* in a sequence (start/stop, intron to extron etc.), that usually follow characteristic patterns, that can be described using regular expressions or *statistical models*. Patterns can be recognized and learned from examples.

Regular Expressions

Describe a set of patterns using regular expressions on letters ATGC.

Example: $G(C|G)A\{G|A|C|T\}^*CA$

- $\{\cdot\}^* : \geq 0$ repetitions

Transition Graphs

To recognize words of languages, we can use:

- **Automata** (*deterministic* decision) for regular expressions
- **Markov Chains** (*probabilistic* decision) to consider frequencies and ratios
    - *Example*: Probability of CG in genome is lower than random, but high in biologically interesting regions (*CpG islands*).

Sequence Families

Sequence families share substructures within associated sequences. Use *hidden Markov models* (**HMM**) to describe and check such “patterns” within one family. The model parameters are estimated from (carefully) manually aligned example sequences.

The **position-specific score matrix** (blocks) is a simpler model for short ungapped segments of fixed length. It gives position-specific probabilities $e_i(x)$, that amino acid x is observed at position i ($\hat{=}$ n-state HMM). Multiplying all probabilities over all positions results in the overall probability, that the segment matches the pattern.

1.4. Phylogenetic Trees

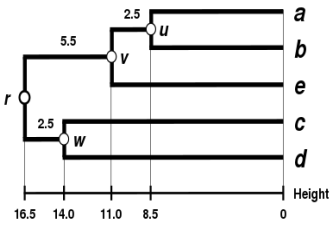
Genome sequences can be used to estimate *phylogeny*. However, genetic phylogeny does not coincide with species phylogeny, so we have to differentiate between

- *Orthologues*: genes diverged through speciation
- *Paralogues*: genes diverged through, e.g., gene duplication

A **phylogenetic tree** describes the derivation of an evolutionary relationship. Nodes are the species/sequences, edges indicate an evolutionary relationship and the amount of change ($\neq$ evolutionary time) between species/sequences is indicated by its length.

UPGMA Clustering

Built a binary tree (*dendrogram*) given species/sequences as nodes and pairwise distances.



- All nodes start of as leaf clusters at height 0.
- Join two clusters with minimal distance (as average over all possible pairs), and *place it at*

the height of half that cluster distance. Distance between two clusters A, B:

$$d(A,B) = \frac{1}{|A| \cdot |B|} \sum_{x \in A} \sum_{y \in B} d(x,y)$$

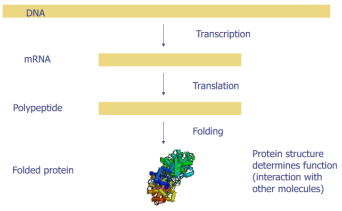
Distance Measures

- $f = \frac{1}{\#\text{different positions}}$; random alignment $f \approx 0.75$ (problem: not high enough)
- *Jukes-Cantor Distance*: $d_{JC} = -0.75 \cdot \log(1 - \frac{4f}{3})$ ($f \to 0.75 \Rightarrow d_{JC} \to \infty$); more realistic estimate
- *UPGMA assumes additive distance*: Distance between any leaves as the sum of paths connecting them is not necessarily the same as the given pairwise distance.

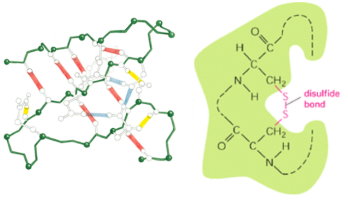
Part 2: Structures and Proteins

▼ *Contents*

DNA is transcripted into *mRNA*, which translates to the amino acid chain **polypeptide**, that folds into an unique 3D structure. The folded protein structure determines its *function* (= interaction with other molecules).



Every amino acid has a **main chain**, which has the same chemical structure over all amino acids, and an unique kind of **side chain** (*polar* vs. *nonpolar*). In water solution, the *hydrophilic* polar side chains form hydrogen bonds, and the *hydrophobic* nonpolar side chains built a hydrophobic core region.



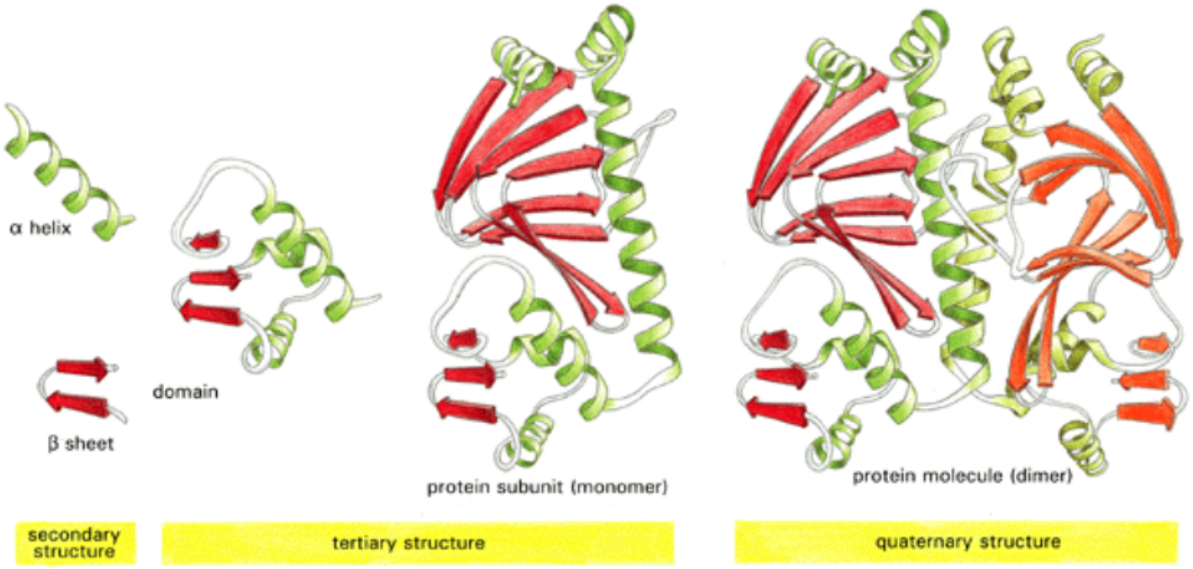
hydrogen bonds / disulfide bonds

Another bonds within the core region are **hydrogen bonds** and **disulfide bonds**, that create connections between remote places within the amino acid chain

Protein Structure Levels

The protein structure can be described at different levels as follows:

- *Primary structure*: 1D amino acid sequence
- *Secondary structure*: α *helix* or β *sheet*
- *Tertiary structure*: combination of secondary structures into protein subunit (**monomer**)
- *Quaternary structure*: combination of several proteins into the protein molecule (**dimer**)



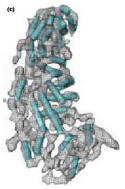
- α **helices** (right-handed) are energetically favourable configurations of hydrogen bond between (neighboring) amino acids
- β **sheets** are further energetically favourable configurations of hydrogen bonds between alternating strands of the amino acid chain (more stable loops)

Note: The current state of computer-based structure prediction of proteins is not reliable, as there are often local mispredictions, whose relevance is hard to assess without experimental evidence.

2.1. Protein Structure Determination

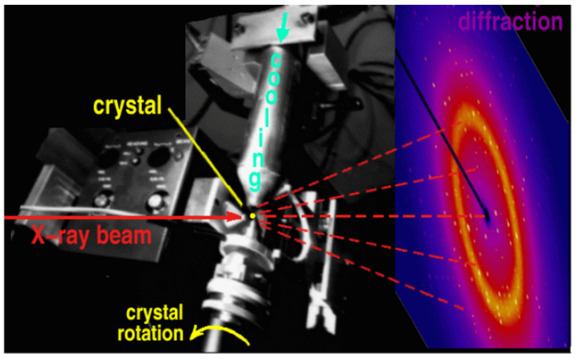
Different approaches can be used to determine the 3D structure of proteins. All of the approaches provide the *electron density*, not the actual protein structure, which is a density map determining *atom placement,* but not atom identification. Given the amino acid chain (along with expertise and examples), one can correctly identify, how the amino acid chain fits into the density map.

Transmission Electron Microscopy (TEM)



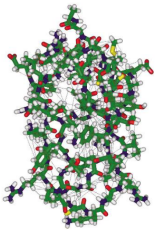
TEM uses a focused beam of electrons to interact with a protein sample and creates 2D images of *electron densities*. For a 3D representation, multiple images are required. However, the resolution is limited.

X-Ray Diffraction Analysis



When an x-ray beam hits the crystallized protein, it causes the x-rays to scatter on the detector, creating a pattern of dark and light spots. By analyzing the pattern, one can conclude the molecular structure. The method is the *most exact*, but the proteins have to be *crystallized* first (time-consuming and expensive).

Nuclear Magnetic Resonance (NMR)



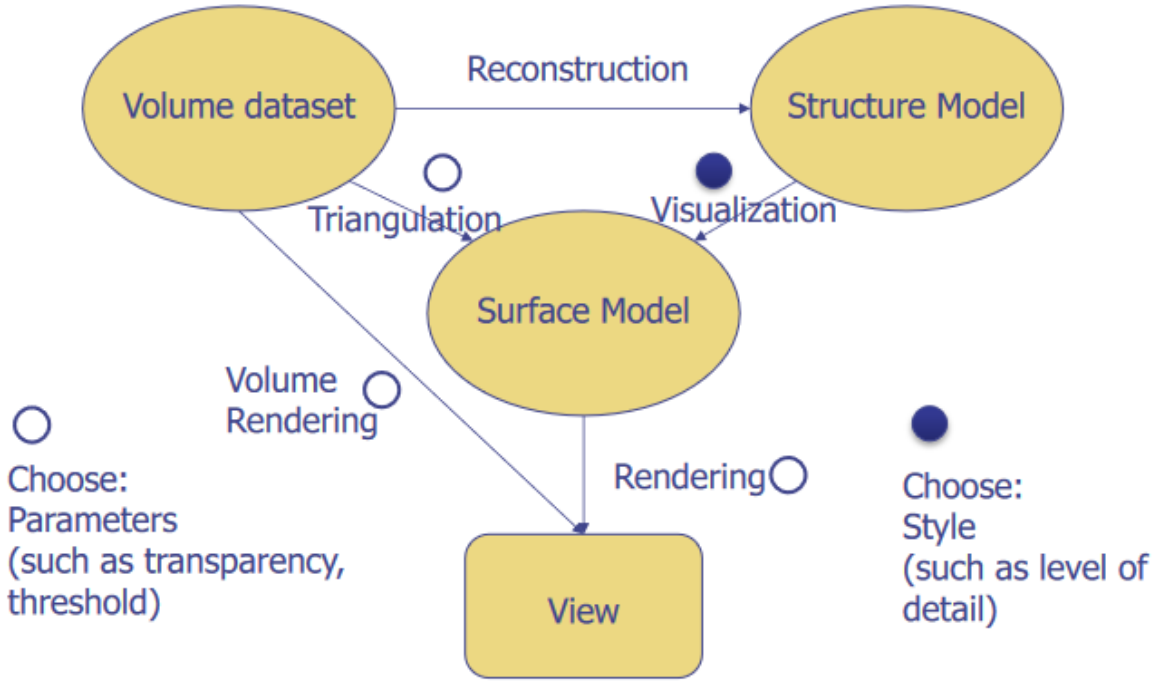
NMR uses strong magnets and radio waves to interact with the magnetic properties of certain atoms (hydrogen, carbon) in a protein. By analyzing the signals emitted, one can figure out how these atoms are arranged in the protein. The method is mainly used as an additional information source, since it is not sufficient to reliably identify the structure due to poor resolution.

2.2. Protein Structure Visualization

Due to the inability of humans to directly see 3D structures, we rely on 3D representations with different purposes, and view them from various perspectives.

There are three different ways to represent a 3D structure of a molecule:

- **Volume Dataset (Density Model)**: Capture the scalar *electron density* field directly from imaging methods; direct visualization (volume rendering) is difficult
  - Due to limited storage, condense density field using a *voxel grid* (may also capture arbitrary measured attributes)
- **Surface Model**: Either *isosurfaces* of equal density or arbitrary surfaces; allows for better visualization, but some details are removed
  - Usually represented as *triangle meshes*.
  - Gives *extension* of molecule at a chosen density model, and an idea of the "form" of the molecule.
- **Structure Model**: S*tructure* as defined by atom positions and by secondary structures (e.g., α helices, β sheets)
  - For reconstruction of the atom composition we use the *PDB* (Protein Database File), storing a 3D coordinate and the binding length (to other atoms) for each atom. Rendered as surface model.
  - An *atomic* structural representation gives atom types and centers, but not their extent. The 3D structure of a protein can be described through a list of atom coordinates (not obvious) Better for algorithms, than surface representation.



Rendering (Mesh → View)

- Project 3D mesh onto image plane via *ray casting*. The color of one pixel is determined by an integral along the ray (I intensity, σ opacity):

$$C = \int \exp\left\{-\int_0^x \sigma(t)\,dt\right\} \cdot I(x)\,dx$$

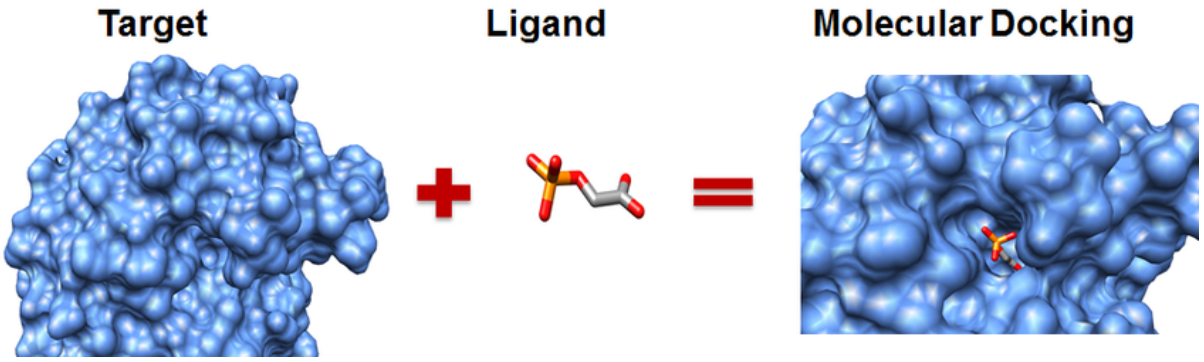
- But opacity is not given, so use an opacity model:
  - Choose opacity to be proportional to *density*.
  - Choose opacity in the basis of intensity: Every intensity above a certain threshold is opaque.
  - Choose maximum *intensity projection* (maximum along ray)

Segmentation (Volume Dataset → Surface)

- Assign a class (e.g. inside, outside) to each voxel depending on different factors, perform isosurface extraction (triangulation) via *Marching Cubes*.

2.3. Molecular Modeling

2.3.1. Docking



Docking refers to a computational process, that simulates and predicts how two static molecules, typically a protein and a small drug-like molecule, may fit together. One issue is that it is difficult to predict the potential side effects of the matching molecule, particularly in the context of drug development.

Principle:

1. Generate docking candidates.
2. Calculate fitness function (geometric alignment, number of van der Waals contacts) between protein and candidate.

Note: Looking for matches directly between their 3D structure is not straightforward, since assessing physical interactions is tricky with this method. Performing a chemical/biological experiment is obviously the gold standard, but it is tricky to isolate proteins and the reaction, and to ensure correct measurements.

2.3.2. Molecular Dynamics

In practice, however, molecules are not static (e.g. thermal movement at room temperature). **Molecular dynamics simulation** is a valuable tool, because limited equipment resolution prevents direct observation of molecular movement.

2.3.3. Protein Structure Prediction

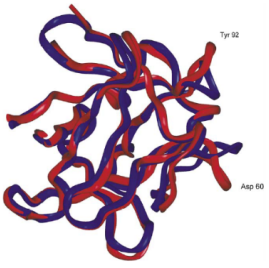
The 3D structure of protein molecules is determined by their amino acid sequence. Protein structure *prediction* methods are assuming some kind of starting point and optimize until a local optimum is reached:

- **Homology-Based**: Find aligned segments without gaps (*SCR*) and loops (*SVR*) to match other known segments from a database with least variation
- **Threading**: Generate fragments and check for alignments in sequence via *scoring* (e.g. with probability distribution of distances between amino acid pairs)
- **Conformational Energy**: Actual calculation of the quantum mechanics conformational energy; A correctly folded protein would have the lowest energy possible, but energy function has local minima.

Metropolis Algorithm

The *Metropolis algorithm* is a versatile *stochastic* optimization procedure, that aims to optimize particular arrangements, which have numerous possibilities. Can be used as a initial candiate for the three approaches before.

- Possible arrangements are expressed as a molecule *conformation* x (e.g. vector of all binding angles between animo acids) with energy $E(x)$.
- Minimize energy by finding a confirmation with a minimal energy:
  - Randomly disturb x into x'.
  - If $E(x') < E(x)$, then continue with x'.
  - Else, randomly choose whether to continue with x or x' (probability is *virtual temperature* T)



Red is prediction by SwissModel (RMS = 1,54Å)

- *Simulate an annealing* by starting with a high T ("exploration"), and lowering T to move into minimum. Guarantees global minimum, but reduction time for that is not known. Restart might give some inside about the quality of the minimum found.

2.4. Protein Databases

All the *additional* knowledge we gathered through simulation, prediction, and interaction with the proteins, needs to be stored in databases.

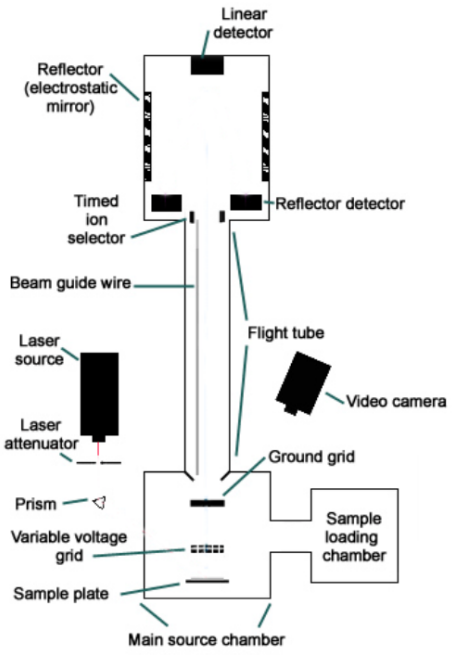
Protein databases include

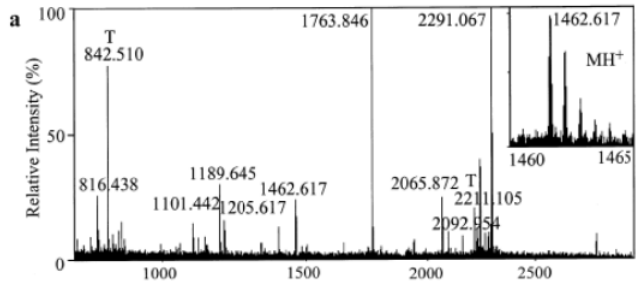
- **Sequence and Structure of Proteins** (e.g. **SwissProt**)
  - Sequence information is stored as in other sequence databases; *Structure information* is stored as PDB file
  - Includes patterns and families in context of other molecules
- **Protein Interactions** (e.g. **BIND**)
  - Information about *protein/protein* and *protein/molecule* interactions
- **Accession Keys**: Unique identification of entry for reference

---

2.5. Protein Indentification by Mass Spectrometry

Mass spectrometry (MS) is a method to analyze and measure the chemical composition of a sample by assessing the **mass-to-charge ratio** of charged particles. In a mass spectrometer, a sample is ionized to form charged particles, which are then accelerated in an electric field, and shoot at a detector, determinining its **Time of Flight** (**TOF**), which correlates with the mass-to-charge ratio.





x: time; time reveals the particle masses with high accuracy

Assume a purified protein (guaranteed one protein in sample). This protein is split into an unique set of peptides by sequence-specific proteases. Then, using a mass spectrometer, the mass-to-charge ratio of all peptides is measured.

Note: Measurements are not necessarly as you expect, since some peptide may be unstable and therefore don't appear in the spectrogram at all. There also may be inpurities/residues added to the spectrogram, that we don't expect from the set of peptides.

2.5.1. Indentification of Single Protein

Peptide Mass Fingerprinting

The identified peptide masses establish a "fingerprint" for that protein. To identify the protein, we can use a database search, where we compare the mass spectrum with known *theoretical* spectra for proteins, and obtain the best match.

The best match can be identified using the **shared peak count algorithm**: If for a shared mass spectrum peak of two proteins, the two masses are within the mass tolerance, it is a match. The highest number of matches is the best candidate.

This scoring function often sufficient, but is also flawed:

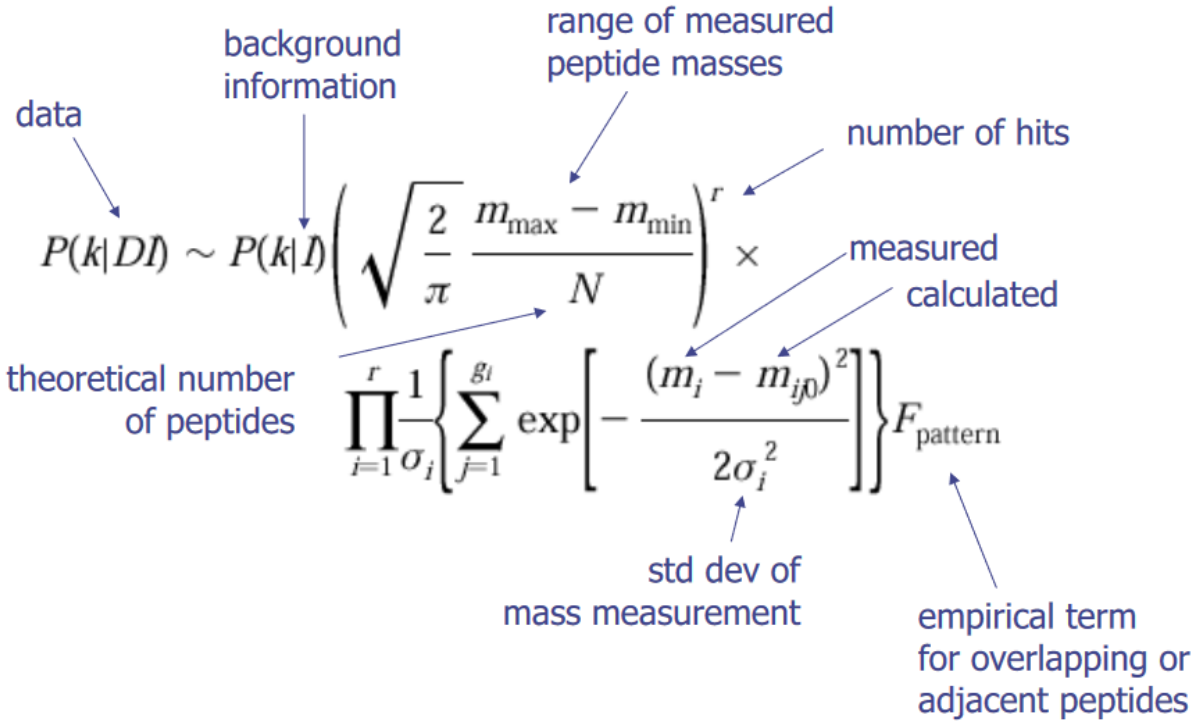
- Not all peptides occur in the mass spectrum, depending on physical properties and experiment conditions.
- Protein might not be even in the dataset.

- Long proteins tend to be preferred, as there are more opportunities for a match.

Statistical Approach: ProFound

Developing a scoring function that accounts for all physical, chemical, and biological factors that affect the mass spectrum, while being both unbiased and precise, requires significant effort. One example can be seen below.

- *Background information*: include/exclude certain cases
- *Range of masses*: only use range that was actually measured; consider at the *theoretical number of peptides* within that range (creates a bias: more possible matches require more matches)
- Second term: distance measure between theoretical peak and measured peak
- *Empirical term*: take into account overlapping or adjacent peptides, that also create peaks



(only idea is required, don't need to learn that)

2.5.2. Identification of Multiple Proteins

Since obtaining a purified protein can be challenging, we aim to detect multiple proteins in a solution.

One simple approach assumes, that every protein in that mixture has at least one unique peptide. Another approach would be to set experimental constraints, such that database search results are unique, using e.g.

- high resolution/accuracy MS restricting hits
- (rare) Cysteine-containing peptides only
- limited number of genes/proteins in known organisms

2.5.3. Identification of Unknown Proteins

Using MS, we can break each peptide arriving at the detector into *ion fragments* by feeding it again into a MS. The peptide sequence is determined through comparison with a theoretical fragmentation of a peptide.

When dealing with unknown organism, we can additionally seach among data from all known organisms, or even using partial sequences only.

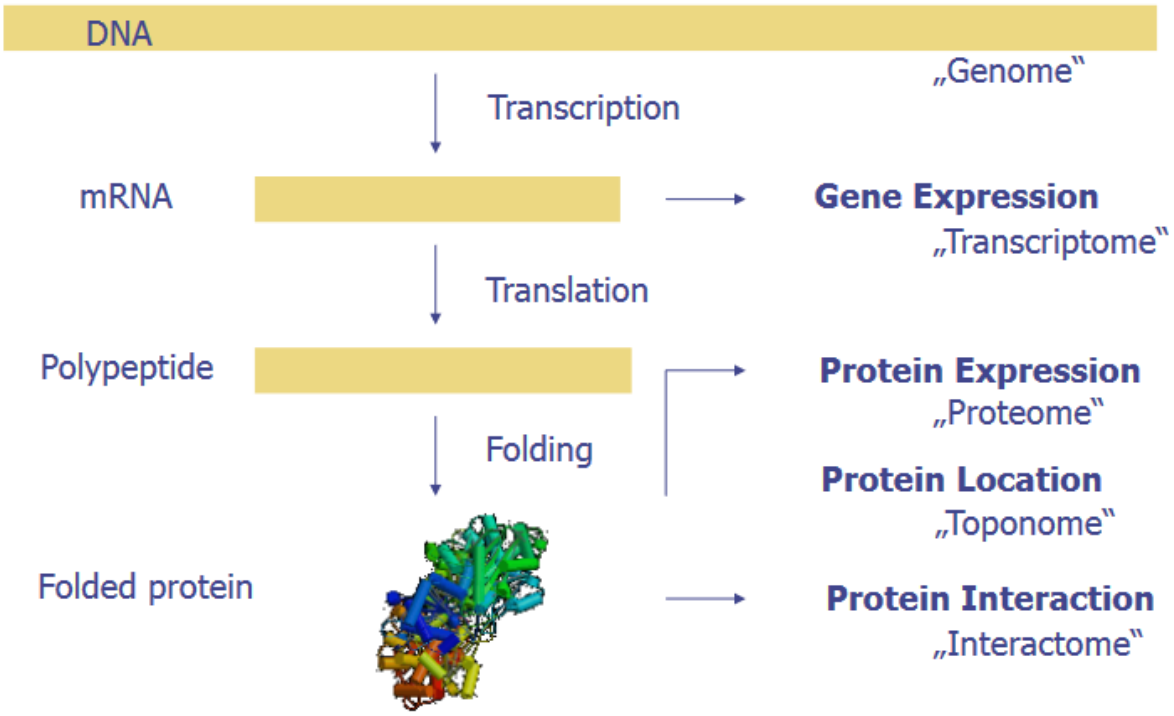
2.5.4. Formalization of the Peptide Identification Task

We examplary specify the problem for an solving algorithm of that task as follows:

- *Peptide* $P := p_1 \dots p_n$ with mass $m(P) = \sum_i m(p_i)$
- *Ion types* $\{d_1, \dots, d_k\}$ as weight differences; d-ion of partial peptide P has mass $m(P) - d$
- *Spectrum* $S := \{s_1, \dots, s_n\}$ is set of (experimental) masses
- Given S, ion types and mass m, find a peptide of mass m with maximal match to S. Scoring function can be shared peak count or more probabilistic.

Part 3: Protein Expression and Function

▼ *Contents*



Protein expression analysis, which denotes the amount of protein found, is distinct from *gene expression*. Gene expression refers to production process, whereas protein expression details the quantity of protein. Additionally, proteins require a specific location for their activity (*protein location*), and perhaps the co-location of other proteins (*protein interaction*).

3.1. Gene Expression Analysis

Although all cells share the same genome, different cell types exhibit varying genome expressions (e.g. skin cells, muscle cells): They only produce the proteins required for their functionality. Gene expression analysis aids in enhancing our understanding of diseases and can provide diagnostic value as well.

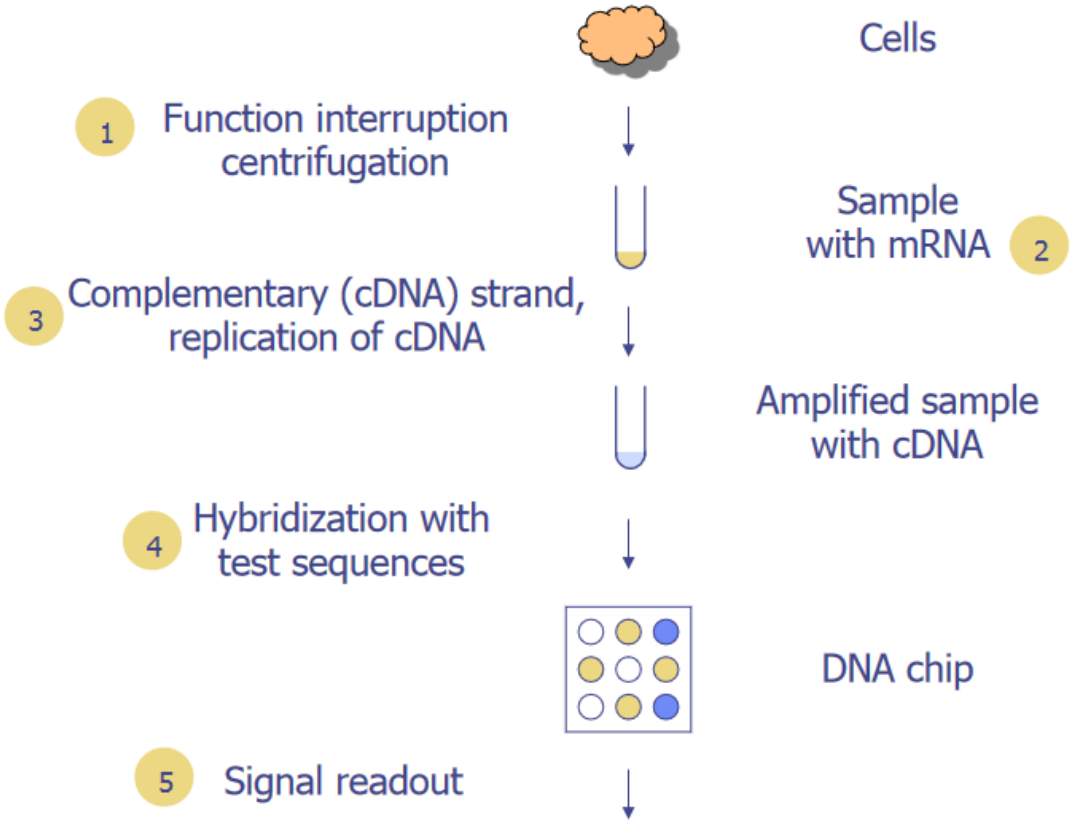
In the gene expression analysis, we measure *mRNA*, and since mRNA is a transcript of active genes, analyzing the corresponding gene expression can provide information about the specific genes that are turned on/off in different cell types.

Moreover, the expression is a dynamic process and its regulation is robust: It may vary over time (cell cycle; or minutes/hours) or over different external

conditions, and cell function can tolerate fluctuations in expression level.

Gene expression experiments have the following steps:

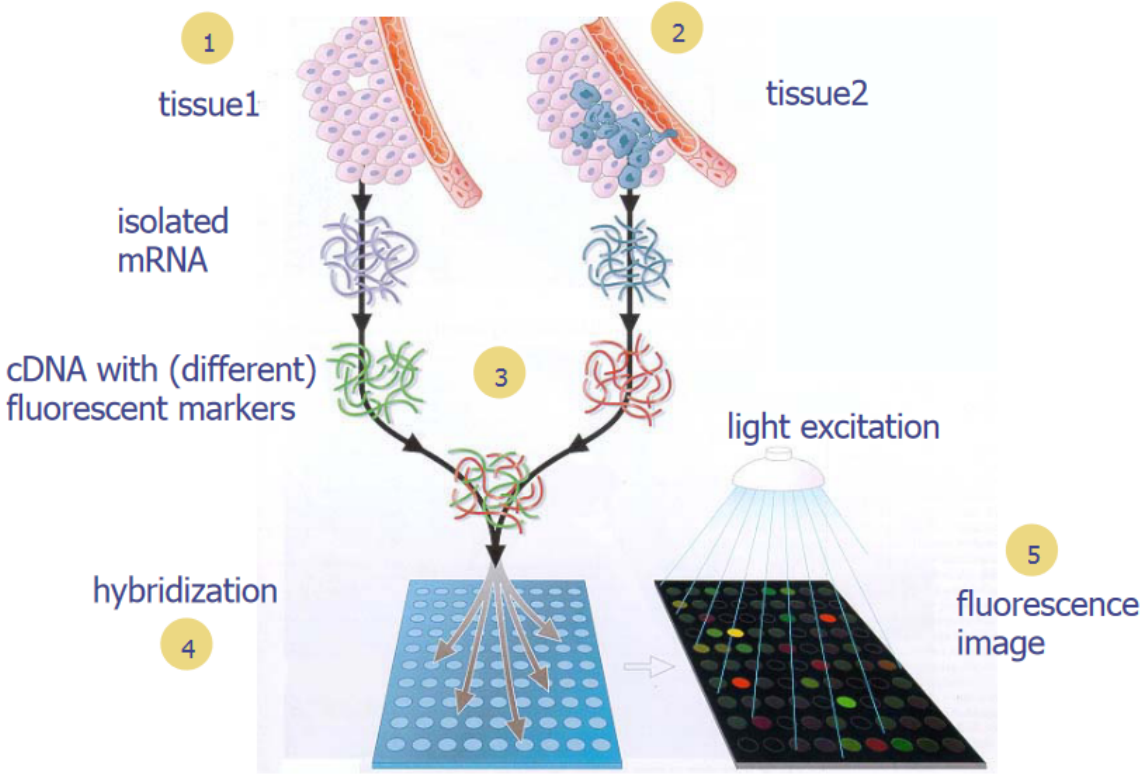
1. Interrupt cell function to avoid falsely experiment conclusions (since living cells might react to stress situations). (*mRNA Isolation*)
2. Create complementary DNA strand of extracted mRNA and amplify the cDNA strands (*cDNA Amplification*).
3. Label the cDNA strands with different fluorescent markers for different samples (*Labeling*).
4. **Hybridization**: Combine complementary DNA molecules from different sources, and fixed onto a DNA-chip, holding known sequences.
5. Assess the presence and abundance of specific genes by reading out the amount of DNA found in certain spots in the DNA-chip (*Optical readout*)



For measurements in a series of different external conditions (e.g. time), we obtain a so-called **expression vector**, which contains the mRNA concentration measured under these specific conditions.

(4) Hybridization

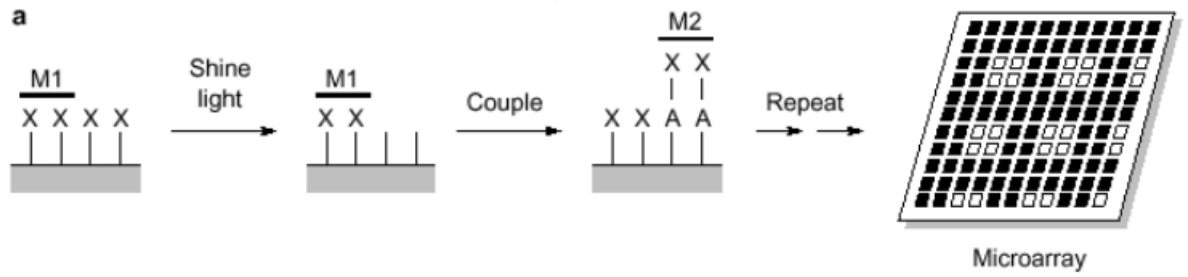
1. Sample preparation is conducted using multiple cell samples as it is not possible to extract cells of a specific type alone.
2. Remove *ribosomal RNA* (80% of total cell RNA; structural only).
3. Tagg cDNA with distinct fluorescent markers, which are visible through fluorescence light on the DNA chip after hybridization.
4. Sample cDNA attaches to probe sequence.
    - There may be also hybridization errors, where it attaches to *similar* sequences with some probability.
    - Use two probes for each sequence: *Perfect Match* (PM) with an identical sequence to the target and *Mismatch* (MM) with one different base. The actual signal is determined by the difference between the Perfect Match and Mismatch signals, helping filter out background noise and enhance data accuracy.



Photolithography

The DNA-chips for the hybridization are created using **photolithography**.

The **on-chip synthesis** of DNA refers to growing specific DNA sequences on each of the DNA-chip spots by selectively using an optical mask to expose certain areas to light. This light removes the chips coating, allowing the area to be flooded and coupled with one base at that exact position. As production time is long, short *oligonucleotides* (25 base pairs) are used per spot.



(5) Readout

- **Laser excitation** involves focusing on individual spots sequentially to minimize background noise from scattered light. Single pixel optical detector readout (*photo diode*) is used, but having many pixels per spot can result in slower readings.
- **Light excitation** simultaneously illuminates the entire chip using a high-resolution camera. This method is faster but may have a lower signal-to-noise ratio. Applied on lower density arrays.

Next-Generation Sequencing (NGS)

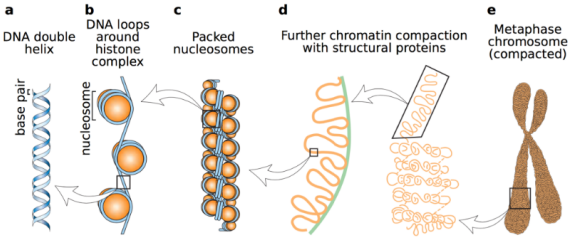
The previously introduced approach has a downside, that we have to make assumptions in form of DNA chips about what we may detect. We aim to understand the chain of events, that initiate a particular expression. Next-generation sequencing is a powerful technology, that not only analyzes gene expression, but also events happening at genome level.

Basics

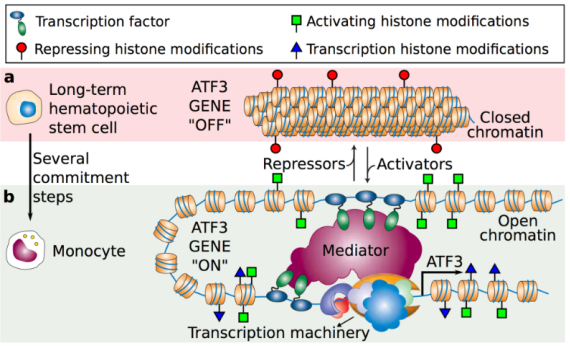
- DNA requires unpacking for transcription.
- *Transcription factors* regulate the transcription of genes. They bind to specific DNA sequences near the genes they regulate and influence the rate at which *RNA polymerase*, the enzyme responsible for transcription,

can initiate and proceed with transcription. We aim to identify which transcription factors are present at what time.

- An *open* chromatin is an partially unpacked DNA for a transcription. Transcription regions are identified either through the presents of the transcription factors or through nearby histone modifications.



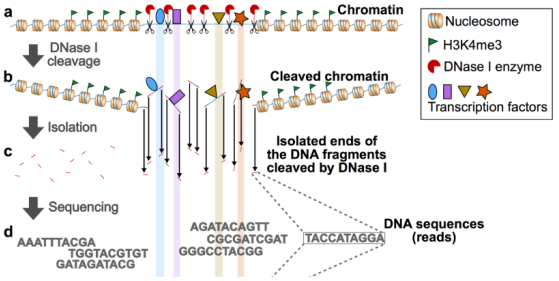
DNA structure



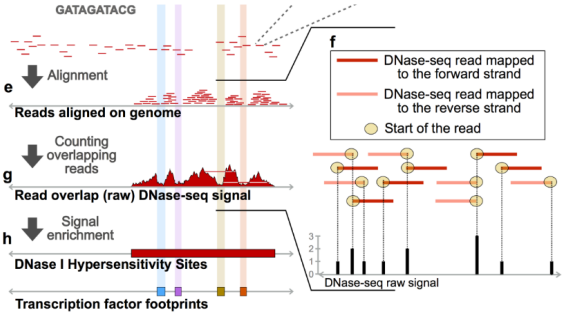
gene activation via transcription factors

Expression Analysis based on NGS

- Cut an *open* chromatin (during transcription) with its attached transcription factors. We not only obtain the gene itself, but also sequences associated with transcription factors.



- Sequence all pieces by realigning them to the non-genome sequence, which results in the identification of the *transcription factor binding regions* in the sequencing signal. They are characterized by a *transcription factor footprint*, where at the valley of two peaks there is a transcription factor binding region that is currently occupied.



3.2. Proteomics

Proteomics is the study of the complete set of proteins expressed by a cell, tissue, or organism at a specific time under particular conditions. It involves the identification, quantification, and characterization of proteins, including their structures and functions.

After protein translation, proteins can undergo

1) Degradation

- *Expression proteomics*: **gel electrophoresis**, **quantitative mass spectrometry**

2) Modification of individual amino acids

- Mass spectrometry

3) Transport

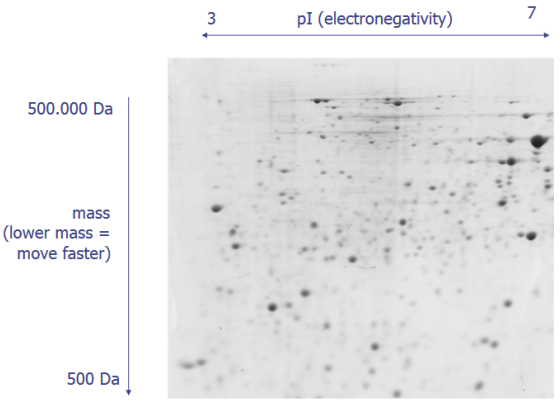
- *Topological proteomics:* selective purification of organelles, microscopy with fluorescent markers

4) Assembly

- *Interaction proteomics:* affinity purification of complexes through baits, bilateral interaction (similar to protein chips)

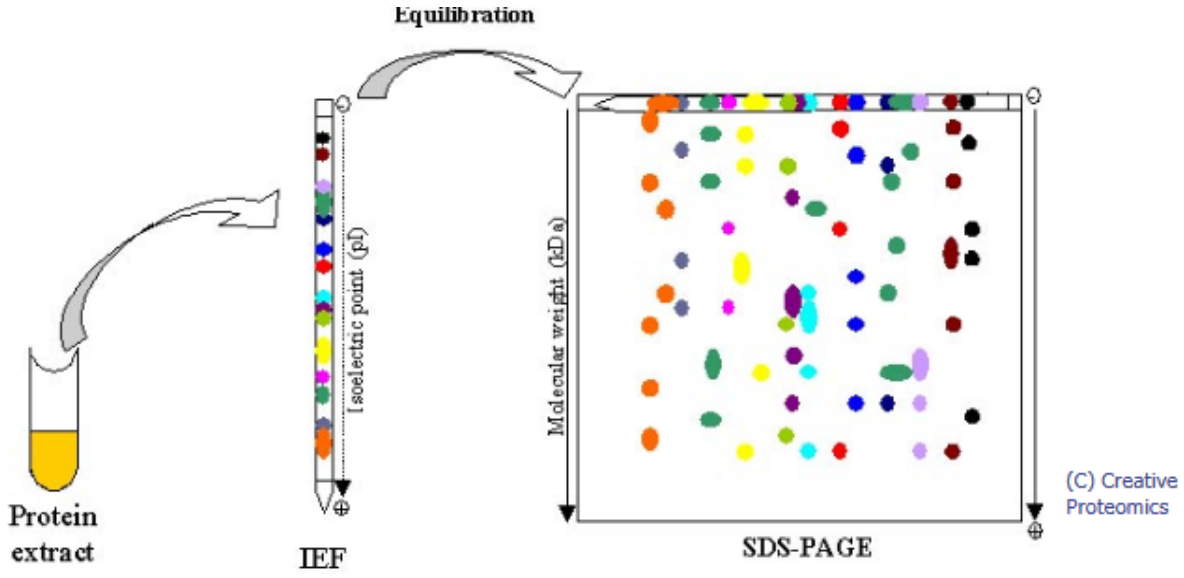
1.1) Degradation: Gel Electrophoresis

Gel electrophoresis is a laboratory technique used to separate proteins based on their size and charge. The method involves placing the biological samples in wells within a gel and applying an electric field. The molecules then migrate through the gel at different rates, with smaller and/or more negatively charged molecules moving faster.



Electrophoresis Preparation

1. Sample preparation via protein extraction.
2. *Electrostatic separation* (pI) on a strip within an electrical field.
3. Place strip vertically on a gel inside a fluid chamber, allowing the proteins to migrate from the strip to that gel through gravity.
4. Staining and scanning.



Electrophoresis Experimental Approach

We compare the gels of two *related* sample types (ideally, several gels per sample type for robustness) by comparing corresponding spots on the gels using a *differential view*, where we compare the *intensity change*. The spots can be extracted and analyzed using mass spectrometry for fingerprinting.

We can also compare several samples, when the proteins are colored beforehand.

Problem in Gel Electrophoresis

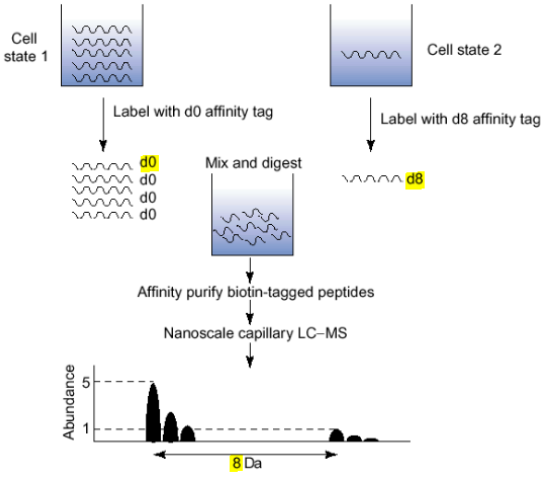
- *Reproducibility*: spot intensities/locations may vary; concentration of different proteins are not easily comparable; smearing of non-chemically stabilized proteins

- Further spot examination is difficult, since spots usually contain several proteins.
- But: *Electrophoresis is the only technique that allows quantification of a large number of proteins at the same time.*

1.2) Degradation: Quantitative Mass Spectrometry

Standard mass spectrometry is not quantitative: The intensity peaks do not correspond to the amount of proteins in a certain sample. Make the original mass spectrometry more quantitative:

1. Each protein in the mixture is identified by a *single tagged peptide* (isotope-coded affinity tagging **ICAT**).
2. Mixture is further separated by *liquid chromatography* (**LC**).
    - LC is similar to gel electrophoresis; Proteins arriving at the end of the column will be input into MS.
3. Relative abundance by peak difference between two different tags.



ICAT

3.1) Transport: High Content Screening

High content screening (**HCS**) is an automated cellular imaging technique used in biological and pharmaceutical research. It involves the systematic analysis of a large number of cellular samples for multiple parameters simultaneously *(e.g., cellular assays in an microplate in different conditions, measure drug response at different concentration)*. HCS combines automated *microscopy* with a **toponomics model** of the protein distribution, that derives a numerical descriptor for each well in the microplate.

Toponomics

Compared to *topological proteomics*, that measure spatial distribution, **toponomics** models spatial distribution based on measurement. The analysis has three models:

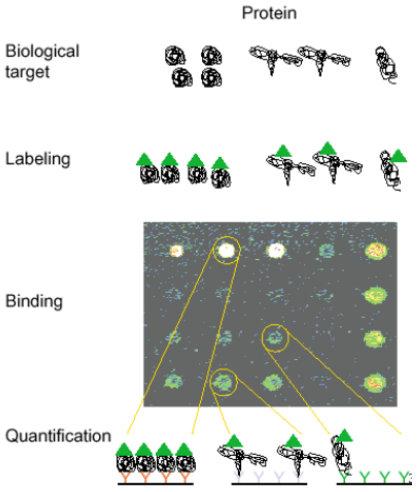
- *Co-location* (0D): identification of different subcellular conditions
- *Transport* (1D): information about regulation state
- *Inclusion/Exclusion* (2D): image analysis (e.g. nuclear region segmentation; quantification)

4.1) Assembly: Affinity Purification

Instead of labeling the protein of interest, we can tag the protein *genetically*, which are then (gently) isolated together with the proteins interacting with it.

4.2) Assembly: Protein Chips

As hybridization is a general process, we can also apply it to proteins in form of **protein chips**, though, unsuccessfully. In contrast to DNA chips, where hybridization is similar for each spot, protein interaction varies. So, we require specific knowledge about selectivity and specificity of antibodies on the chip, that we use to bind proteins onto *(e.g., antibodies can tag multiple proteins)*.



3.3. Image Processing

3.3.1. Single Image Processing

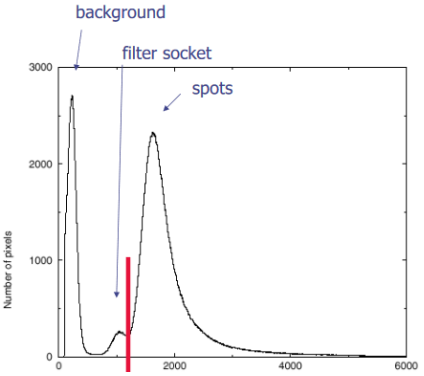
Earlier we saw how to create experimental 2D data (e.g. fluorescent images of spots on chip), from which we want to extract numbers. Quantitative single image processing can be separated into two steps:

1. **Segmentation**: signal vs. background/ vs. separate sources
2. **Quantification**: signal intensity estimation; background-signal estimation

Segmentation

Segmentation can be implemented as a simple thresholding algorithm, that assigns each pixel to a particular class, where we consider neighboring pixels for classification.

However, choosing a threshold is not trivial. We can utilize *histogram thresholding*: Determine a histogram (intensity over number of pixels), and set the threshold in the *valley* between peaks.



histogram thresholding

Quantification

We can perform **quantification** of the pixel belonging to the same class in various ways:

- **Average Peak**: average over all segmented pixels
  - size variations are not considered, non-uniform brackground will distort values (*background-sensitive*)
- **Total Intensity**: sum up intensities of all segmented pixels
  - *background-sensitive*, signals at the borders vary with segmentation threshold (*segmentation-sensitive*)
- **Curve Fitting**: fit curve with smallest distance from segmented data
  - outliers can skew fitting (*noise-sensitive*)

Curve Fitting: Gauss Fitting

- Set of segmented pixels $M_R = \{p(x,y)|(x,y) \text{ segmented}\}$
- Set of Gaussian values for segmented pixels $M_V[H, B, c_x, c_y, s_x, s_y] = \{G(x,y)|(x,y) \text{ segmented}\}$
- *Gaussian model*:

$$G(x,y) = H\exp\{-(x-c_x)^2/2s_x^2\}\exp\{-(y-c_y)^2/2s_y^2\} + B$$

- H height; B background; c_x, c_y center; s_x, s_y standard deviation
  - Least-Squares fit of M_R and M_V

Background Estimation

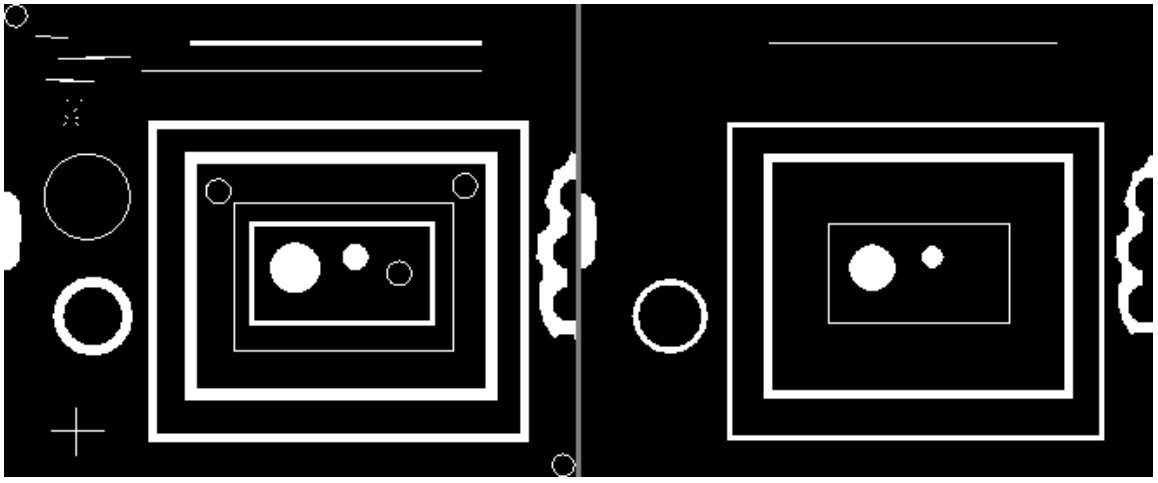
Background is not uniform on the entire chip due to possibly different lighting conditions and optical non-uniformities. We can mitigate the background effects using

- *Background Estimation* (after segmentation): average of surrounding non-spot areas, fitting
- *Background Subtraction* (before segmentation): **morphological operations**

Background Estimation: Morphological Operations

Morphological operations apply a *structuring element* (shape of neighborhood) to an input image, determining the smoothness of the operation. The value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors.

- *Erosion*: value of the output pixel is the minimum value of all pixels in the neighborhood; removes floating pixels and thin lines
  - underestimated background; more reproducible results than local background estimation, since it doesn't depend on the segmentation

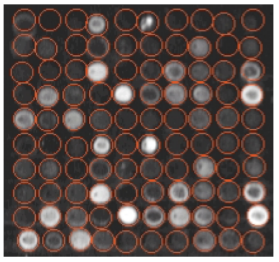


3.3.2. Registration

Instead of single images, determine a mapping between several different images to a common reference. We differentiate between two kinds of registration:

- **Rigid Registration**: translation + rotation, same scale
- **Elastic Registration**: eliminate distortion *(from e.g. optical systems, mechanical elasticity of carriers, spatial variations in biophysical processes)*

First, we need to register the spot grid on the chip by determining the orientation (manually/automatically) and the raster positions using the given dimensions of the raster.



- *Example registration*: set seed points at theoretical grid positions, calculate center of gravity and fit smallest circle above threshold

Rigid Registration: Gray Level Correlation

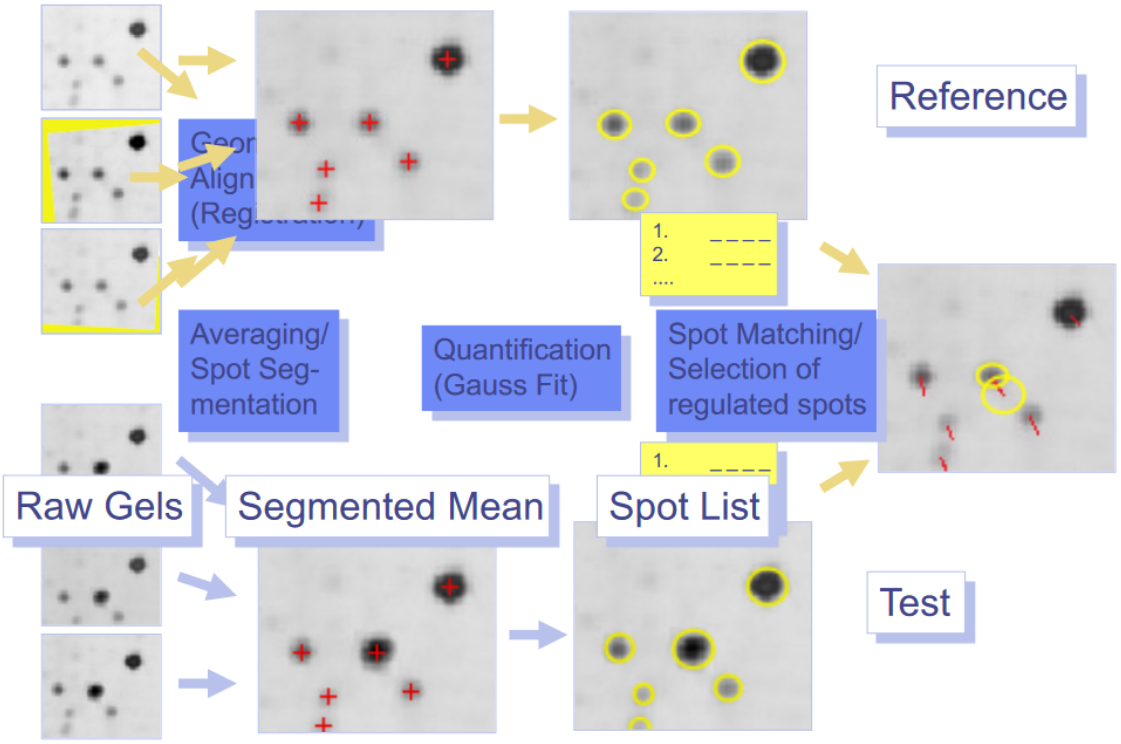
Assuming intensity correspondences between images, we can optimize a mapping between their colors for maximum correlation (*gray level correlation*)

- **Joint Histogram**: $p(a, b)$ is the number of pixels where intensity a in image A maps to intensity b in image B; maximize correlation coefficient in the histogram; requires same color scale
- **Joint Entropy**: $-\sum(p(a, b) \log p(a, b))$; high value means few pronounced entries; works with different color scales

Non-Rigid Registration: Optical Flow

In **optical flow**, a vector field defines an elastic mapping for each pixel. The mapping function is a grid of displacement vectors with lower resolution than the image, where the displacement vectors are interpolated.

- **Grid-based Elastic Matching:** Define contraints of displacement relative to neighbors using an *iterative refinement*. Starting with a rigid registration (one vector per pixel), subdivide image and adjust displacements.
    - Background areas causes problems, since they have no signal and thus no clear optimization direction.
- **Point Matching:** Instead of matching the entire space, we can match certain point sets *(e.g. segmented point sets)*.



elastic registration + Gauss fit

3.4. Data Analysis

After collecting all of the statistics, we use data analysis tools to examine the expression. Main questions here are:

- Which are the different patterns of expression? *(pattern = similar behavior under a given set of conditions)*
- What uncharacterized genes have similar expression pattern to well-characterized ones?

- Are there subtypes of disease X recognizable by tissue gene or protein expression? *(Which genes best differentiate between different tissue classes?)*

Main *data analysis ingredients* are:

- **Normalization** (quantitatively comparable across experiments)
- **Data mining** (**significant fold change**, **clustering**)
- **Visualization** (quality control, visual data mining)

3.4.1. Normalization

We can use one of the three approaches to determine the normalization factor:

- **Total intensity**: In the context of hybridized spot, we know, that the *sum of intensities* (= total DNA / protein content) is always the same across experiments. Requires careful control of DNA / protein content during the experiments.
- **Regression**: Establish a linear/non-linear correlation curve between two samples/datasets. Assumes a sufficient amount of similar genes in one dataset. Doesn't requires conditions of total intensity.
- **Set of known markers**: During experiments, use a collection of known markers (*housekeeping genes*: uniform concentration; known RNA) to generate a correction factor. Doesn't requires conditions of regression.

Normalization yields expression values, which are often expressed as a (absolute) *logarithmic expression ratio/level*.

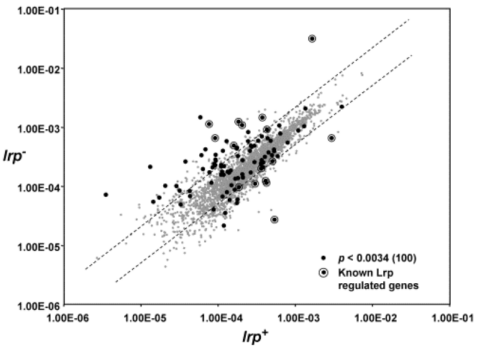
3.2.2. Data Mining

(N-)Fold Analysis

(N-)Fold analysis (*"folding analysis"*) involves comparing two states or experiments by examining expression ratios to identify significant changes.

The difficulty is determining the significance of the change, since the measurement is subject to *biological* (fluctuations in regulation) and *technical* (background noise) errors. Those errors can be estimated through repeated measurements.

Statistical estimates, often employing a *t-test* with a specified confidence level of significant change, depend on the error model of the array technology used.

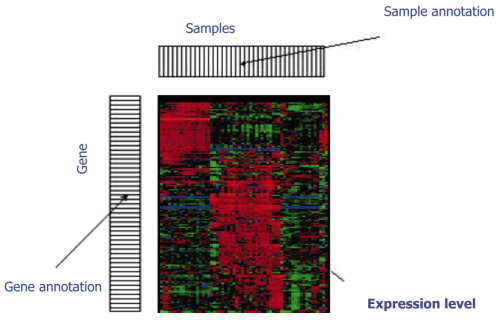


everything outside of the dashed lines may be considered suspocious (significant variation), but doesn't coincide within the placement of known regulated genes

Clustering

From genes and their expression under certain conditions / from certain samples we can create an **expression matrix** holding *expression level* values. To identify such groups as in the figure on the right, we have to perform *cluster analysis*.

Heat maps are commonly used to visualize relative expression levels.



heat map

Cluster analysis in gene expression data aims to identify groups with similar *expression patterns*, revealing shared responses to different conditions (e.g., states or treatments). The idea is to group similar samples/genes based on their expression levels by rearranging rows/columns to maximize similarity between adjacent samples/genes.

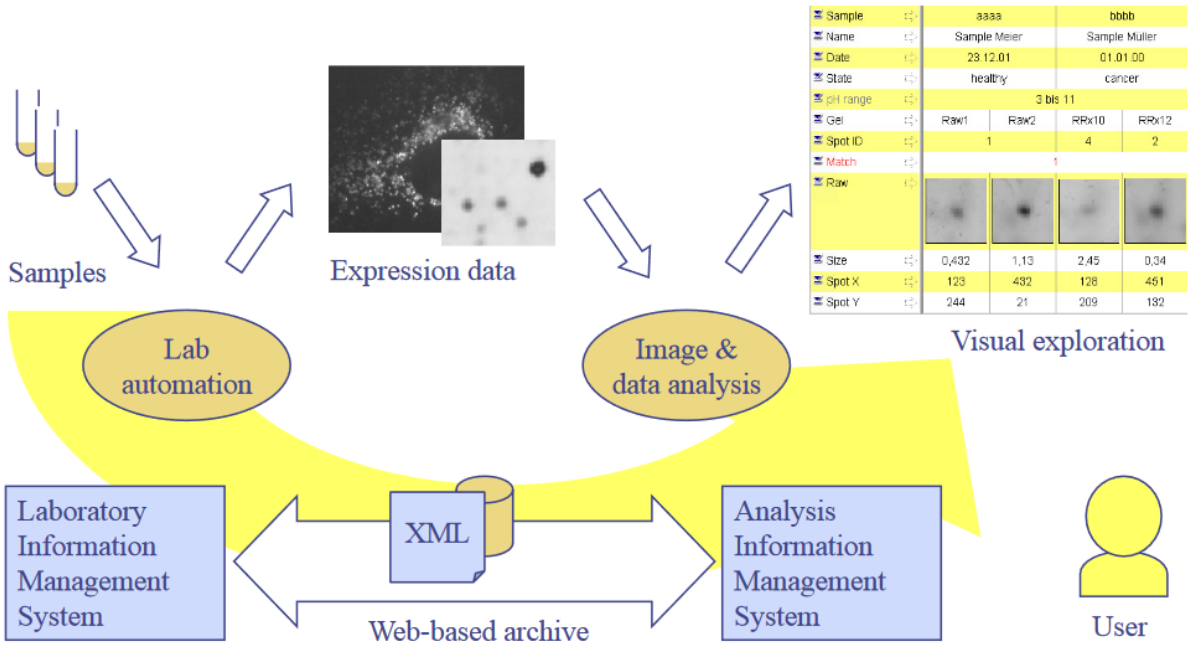
This *probabilistic* process relies on distance measures, and while clustering methods lack inherent prior knowledge. Incorporating error models enhances analysis accuracy.

Clustering algorithm examples: K-Means (*grouping*), UPGMA (*sorting*)

Distance Measures

Gene expression clustering utilizes distance measures, treating genes as *n*-dimensional vectors with various attributes. *Geometric metrics* like Euclidean distance (normalized dimensions assumed) and *similarity* measures such as correlation and mutual information (for incerse regulation) assess the relationships between gene expression profiles.

3.5. Data Management



To collect and archive the massive quantity of data gathered for later use, we require IT assistance:

1. **Laboratory Information Management System** (**LIMS**): tracks samples, experiments, and results
2. **Analysis Information Management System**
    a. **Archive**: long-term storage for documentation and approval

b. *Data Analysis Workspace*: interactive, statistical analysis, and visualization

3. [*Project Management*: resource control and accounting]

3.5.1. Archiving

Raw data archiving is crucial for instrument data in bioinformatics, requiring long-term storage in formats like ASCII (XML, PDF) for purposes like FDA approval and authenticity proof. A *LIMS*, preferably database-based, aids in tracking samples, linking to archives, and integrating with data analysis and project management.

3.5.2. Data Design

Designing data structures for individual studies depends on the experiment type, such as screening with a fixed protocol or variation experiments with experiment-specific input parameters. The variability in experimental designs and the introduction of new devices or protocols necessitate a *flexible* approach. It is essential to plan the data model concurrently with the experiment design to ensure an effective and adaptable framework.

Metadata

Metadata plays a crucial role by providing annotations and explanations for datasets. This additional information serves to contextualize and enhance the understanding of the data, contributing to effective management and analysis (e.g. time stamps, sample type).

Data Model

A **data model** describes the relationship between different metadata subgroups in form of an *entity relationship diagram*.

Databases offer a single schema for managing large amounts of similarly-structured data, making queries for specific records easy but overview challenging. They are often monolithic and expensive but allow for selective record updates. In contrast, **single files** (e.g., Tabular ASCII, Excel, XML, JSON) accommodate more complex structures with medium-sized data. While queries may be less efficient, browsing and overview are feasible. However, updating

selected records in single files often requires rewriting the entire file, providing individual flexibility.

Several modules (executable tools) can be used within one pipeline, but have to agree via input, output data, and metadata format.

File Formats

XML modeling and processing involve files with nested sections of different type and structure defined in a *Document Type Definition* (DTD). Processing can be specified for individual sections. Sections can be extended later (similar to *subclassing*), where old code ignores new data, and new code can handle both old and new sections. *JSON* is usually easier to work with, since for XML we require an individual XML parser.

Example usage of XML as a standardized interchange format: The **Gene Expression Markup Language** (**GEML**) includes the two XML document types: *pattern files* (for project, chip layout, and probes), as well as *profile files* (for signal, background, optical channel information, and log ratio).

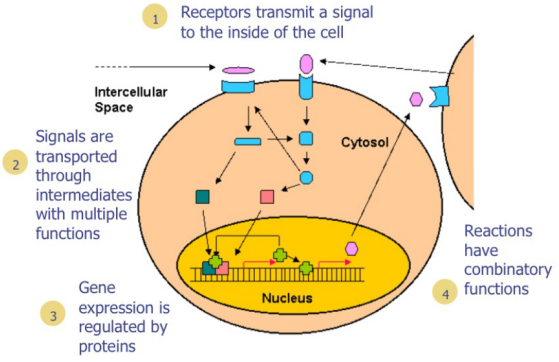
Part 4: Networks and Systems

▼ *Contents*

After examining specific components of molecular biology, we would like to investigate how those components interact inside cells.

The cell is a complex system with various simultaneous processes, including gene expression, protein synthesis, transport, metabolism, signaling, and regulation. The currently only way to model these processes is through **networks of reactions.**

A **reaction** has input and output, processes depend on a *quanititative relationship* between input and output.



Example of a network of reactions: Signal transduction in the cell (response to extracellular signal)

4.1. Biological Network

A **pathway** is a chain/sequence of individual steps, where input from one step is needed in the next one. The combination of several (possibly intersecting and overlapping) pathways creates a **network**.

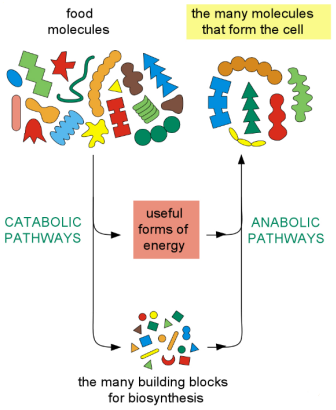
There are four types of biochemical networks that exist in a cell:

- **Metabolic Networks**: (basic) biochemincal cell function (e.g. energy production)
- **Gene Regulation Networks**: control of cell function depending on state and external influences
- **Signal Transduction Networks**: response to intracellular and extracellular signals
- **Transport**: between different compartments of the cell and to the outside

---

4.1.1. Metabolic Networks

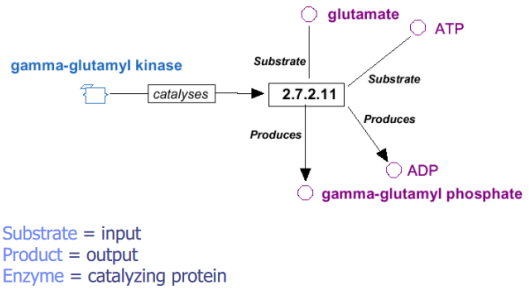
Metabolic networks describe the mechanism, that turn food molecules (e.g. glucose) into useful forms of energy and into building blocks for biosynthesis through *catabolic pathways*, both of which are used in the *anabolic pathway* to create molecules that form the cell itself.



▼ **Example: Individual Metabolic Reaction using Enzymes**

- *Mechanism of enzyme activity*: Enzyme has a binding pocket for substrates, that when entering this pocket are split into different products (in general, reversible, but unlikely)
- The mechanism is in general reversible; We quantitize the probability of the forward or backward process using the **equilibrium constant** K expressed by a concentration ratio of products divided by the concentration of substrate in a solution:

$$K = \frac{[C][D]}{[A][B]}$$



Substrate = input
Product = output
Enzyme = catalyzing protein

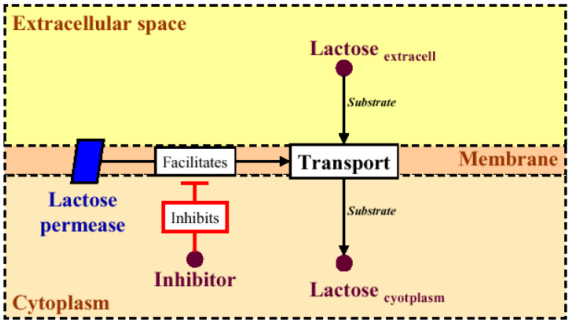
Example: gamma-glutamyl kinase catalyses (”starts”) the reaction, and using glutamate and energy (ATP), we get some other molecules.

4.1.2. Gene Regulation Networks

Gene regulation networks manage the control of cellular functions in response to internal states and external influences. These networks involve the intricate interplay of genes, transcription factors, and other regulatory elements to modulate gene expression, ultimately influencing cellular processes and behaviors.

4.1.3. Transport

Transport networks are responsible for the movement of molecules between various compartments within the cell and to the external environment. These networks manage the intricate processes of importing essential substances,



exporting waste products, and maintaining the cellular balance necessary for proper functioning and responsiveness to external stimuli.

4.2. Network Databases

Before we look at how to model networks and analyze them, we should first look at where the information for them comes from.

The most prominent databases are:

- **KEGG**: metabolic and regulatory data with emphasis on sequenced genes
- **EcoCyc**: genome and biochemical information on *Escherichia coli*
- **BRENDA**: comprehensive non-restricted enzyme information
- **TRANSPATH**: molecular pathways and cellular network modelling

4.2.1. KEGG

KEGG (Kyoto Encyclopedia of Genes and Genomes) is a repository hosted by the Institute for Chemical Research at Kyoto University. It serves as a collection of metabolic pathways for organisms with fully sequenced genomes, encompassing both *metabolic* and *regulatory information*. In cases where experimental data is limited, protein functions are inferred based on sequence similarity with proteins characterized in other organisms. The pathways are visually represented as manually created diagrams stored as static GIF files, with organism-specific enzyme information highlighted in color upon selection.

4.2.2. EcoCyc

EcoCyc was initially established as a metabolic pathway database for *Escherichia coli*. It is currently undergoing expansion to encompass other microbial organisms, including *MetaCyc*, which lacks genomic data. Pathway diagrams, created using a graph layout algorithm, are stored as static images for web browsing. Unlike KEGG, EcoCyc includes information derived from published experimental data, providing insights into genes whose function has been characterized through genetic and biochemical approaches, even if they have not yet been cloned.

4.2.3. BRENDA

BRENDA is a comprehensive enzyme information system that originated with a primary focus on metabolic pathways.

In BRENDA, enzymes are identified with an specific **EC number** (e.g. EC 1.18.1.2) as an accession key:

- *First digit*: enzyme overvall reaction
- *Second digit*: substrate acted on by enzyme
- *Third digit*: acceptor type
- *Fourth digit*: enzyme's arbitrarily assigned serial number in its subsubclass

---

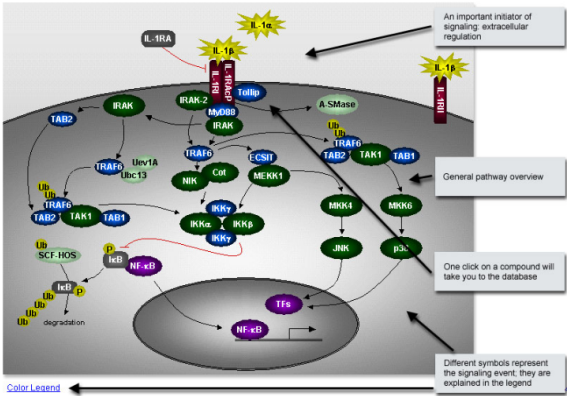
4.2.3. TRANSPATH

TRANSPATH is a database focused on molecular pathways and cellular network modeling, covering various aspects of molecular interactions within the cell cytoplasm. It allows *on-the-fly pathway generation* from extracellular agents down to affected transcription factors and triggered genes. The database includes elements such as ligands, receptors, enzymes, and transcription factors, providing information on molecular interactions within relevant signal transduction pathways.

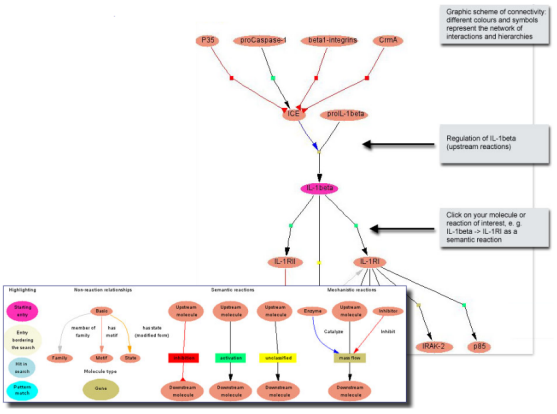
Selected topics included in TRANSPATH are:

- Cell cycle control
- Cell adhesion and motility
- Degradation
- G-protein pathways
- Apoptosis
- Growth and Differentiation
- Immune response

TRANSPATH includes not only molecule entries, but also *reaction entries*, that are used to create *maps of signaling networks* in a compact shape. These differ significantly from hand-drawn networks in components like arrangement and subsystems.



Hand-drawn map



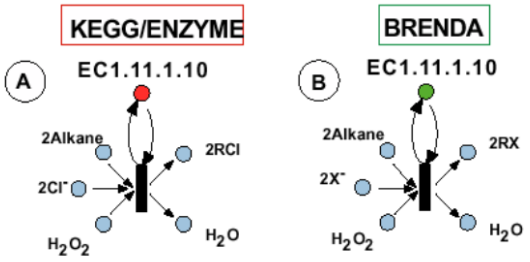
Map created by TRANSPATH

4.3. Network Analysis

Exploit information from network databases for biochemical network analysis.

Reactions as Petri Nets

Reactions can be modeled as *transitions* of **Petri Nets** $N = (P, T, E)$ (places, transitions, edges) with $E \subseteq (P \times T) \cup (T \times P)$, with a binary representation of quantity as a form of *activity* in metabolic reactions, that is usually dynamic.



Extension Through Other DBs

For a good linkage of different reactions in term of regulation we need not only signaling reactions, but also *gene regulation*. Gene regulation information is not originally stored in interaction databases directly, but indirectly in e.g. keywords of protein database entries. A network can be populated with gene regulation information by applying *text mining* on protein databases on keywords indicating activation relationships.

Pathways

Due to *biological robustness*, a network usually has several thousand of pathways for the same reaction in a network.

To better interpret a gene expression experiment using networks, we ulitize additional path scoring methods. **Pathway scoring** involves the *selection* of pertinent pathways within a closed set or specific organism. For that, we relate the acquired gene expression data to the enzymes in the pathway and score whole sequences of reactions according to number of reaction occurrences.

Gene Scores

- **Conspicousness score:** Assume, that a gene g is *conspicuous* when its is changing more than normal according to a near zero mean *normal distribution*. The conspicousness score then evaluates the distance from the mean. For a time series, it is the sum of the individual scores. (Similar to N-fold analysis)

$$score_t(g) = -\log P_t^0(g) = -\log 2\Phi\left(-\left|\frac{m_{g,t} - 0}{s_{err}}\right|\right)$$
$$score_c(g) = \frac{1}{|T| - 1} \sum_{t \in T - \{t_0\}} score_t(g)$$

- **Synchrony score**: Sum up correlation coefficients to all other genes in the pathway, exclusing the gene itself, if it is a member of the pathway:

$$score_p(g) = \frac{1}{|p_g|} \sum_{h \in p_g} cc(g, h)$$

Path Scores

The **pathway score** is then the average of all gene scores (conspicousness/synchrony) over the path p:

$$score(p) = \frac{1}{|p|} \sum_{g \in p} score_{c/g}(g)$$

Path scores are calculated over the entire path and highlight relevant paths.

(In general, we might also miss information when analyzing pathways only, simply by not having the information for connections, so usually we utilize pathway assisted analysis methods.)

4.4. Simulation

Biological networks can also be *simulated*. This involves considering a comprehensive set of reactions with specified constants, *conditions* such as molecule numbers, states, and locations, and then observing the dynamic behavior and temporal functions of the network *over time*, including individual molecule interactions and global patterns.

Simulation methods can be categorized into two categories:

- **System dynamics**, involving the formulation of *differential equations* with iterative solutions to capture reaction kinetics and diffusion transport.
- **Stochastic methods**, which utilize *probabilities* for individual changes, incorporating smaller elements like individual molecules and steps, allowing for the representation of diverse state transitions, including individual modifications and complex formation, although with non-reproducible *randomness*.

4.4.1. System Dynamics

One example for system dynamics is *E-Cell*, which is a (reduced) dynamics model of the bacterium Mycoplasma Genitalium.

People working on this simulation created a special language called the **Systems Biology Markup Language** (**SBML**) to describe all the details of these reactions, which are initially stored in the pathway database. SBML, an *XML extension* for Systems Biology, defines compartments, species, reactions, and optionally includes parameters, units, and rules (in a not-human-readable way). Supported by various simulation tools like E-Cell, SBML accommodates *kinetic rate law equations*, including 34 reaction types and various inhibitions.

Not the simulation itself is challenging, but the *choice of parameters* in kinetic laws is:

- *Obtaining parameters:* Often measured under artificial conditions, with many unknown values

- *Characterizing reactions:* Uncertain exact mechanisms, leading to difficulty in determining the best rate law approximation

- *Complexity*: We require a considerable number of parameters (millions of experiments) for more complex reaction types

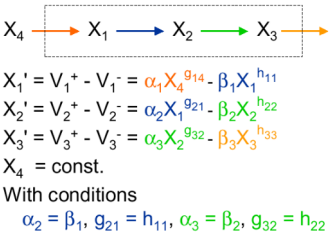
S-Systems

S-Systems offer a simpler approximation to the laws with a *general* structure. By transforming most differential equations into a nonlinear canonical form known as the *S-System*, efficient computation of higher-order derivatives becomes possible. This transformation, especially when conducted in logarithmic coordinates, enables the construction of a higher-order Taylor solver, providing a faster solution (10-100 times) compared to standard ODE solvers. This approach is advantageous when estimating or inferring parameters and equation types.

The reaction rate change depends on all of the substrates and products, for which we just have exponents to identify the nature of change (unmodulated reaction [0.5,1], activations [0,0.5], inhibitions [-0.5,0]). The S-Systems general equation is then given as the subtraction of the product rate and the substrate rate, consisting of a product of (unknown?) individual rate constants X_i:

$$X_i' = \underbrace{\alpha_i X_1^{g_{i1}} X_2^{g_{i2}} \cdots X_N^{g_{iN}}}_{product} - \underbrace{\beta_i X_1^{f_{i1}} X_2^{f_{i2}} \cdots X_N^{f_{iN}}}_{substrate}$$

For a linear pathway, we simple use the nodes' input as the product and its output as the substrate, leading to a system of equations given certain conditions on the exponents.



4.4.2. Stochastic Simulation

Stochastic simulation offers an alternative to the deterministic approach, which relies on *assumptions often not valid for biological systems*, such as infinite reaction volume and systems close to equilibrium. The stochastic approach is applicable for scenarios involving small reaction volumes, short timescales, and situations far from equilibrium

It involves a *discrete* state space, representing integer amounts of each species in the system. It uses *reaction probabilities* instead of reaction rates, essentially

simulating a *Markov process* with one biochemical reaction per iteration. The simulation generates *trajectories* through the state space, and different runs produce varied results due to the inherent stochastic nature of the process.

StochSim Algorithm

1. At time t (small increments), randomly choose two molecules.
    a. Second choice can come from *dummy* molecules to simulate unimolecular reactions.
2. *Choosing a probable reaction*: Scan list of all possible reactions and their probabilities. Choose one randomly accounting for that probability, or choose none with the remaining probability.
3. *Execute reactions*: Produce new molecules or modify state of molecules.

Gillespie Algorithm

1. Choose next reaction: Increment time by d, execute effects of reaction u.
2. Compute probabilty density, that next reaction will occur at time increment d and will be of type u.
    a. *"First reaction"* method: Calculate time expectation for every reaction type, and choose the minimum (expensive)
    b. *"Next reaction"* method: Keep "next time" for all reactions in a sorted list. Update only those that are affected by the current reaction.

Comparison

- Gillespie method skips to the next "active" time interval, treating all molecules of a kind together and assuming homogeneous concentration.
- StochSim, although slower due to individual molecule consideration, can better account for different protein states (e.g., accounting for a protein with N different modification sites as 2^N distinct molecule classes by Gillespie) and accommodates complex formation and geometric distribution.