Kernel Methods in Bioinformatics

Jean-Philippe Vert
Jean-Philippe.Vert@ensmp.fr

Centre for Computational Biology
Ecole des Mines de Paris, ParisTech

Machine Learning Summer School, Taipei, Taiwan, July 25-26, 2006.

Outline

Outline

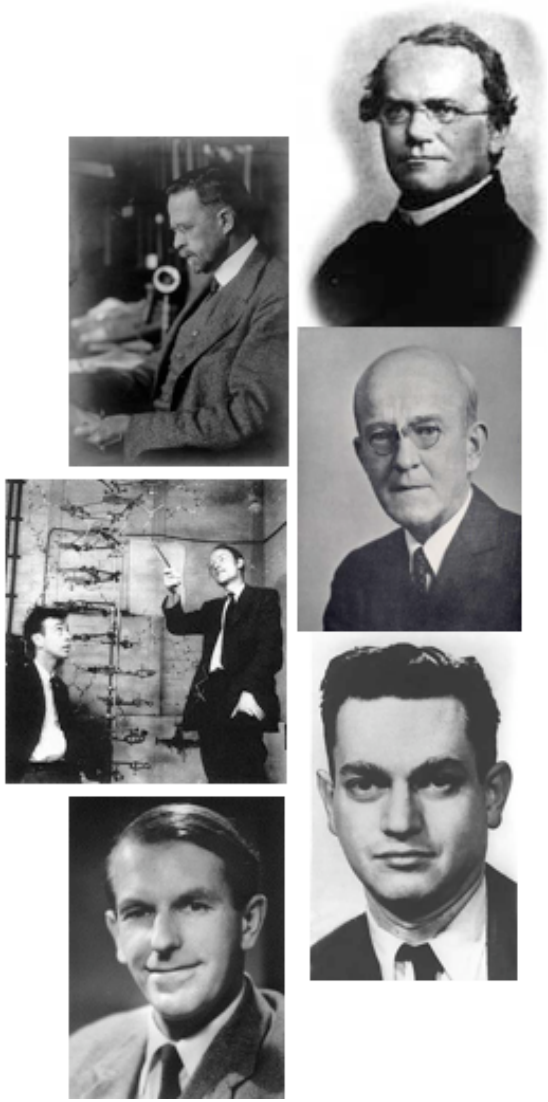
Outline

Outline

Part 1

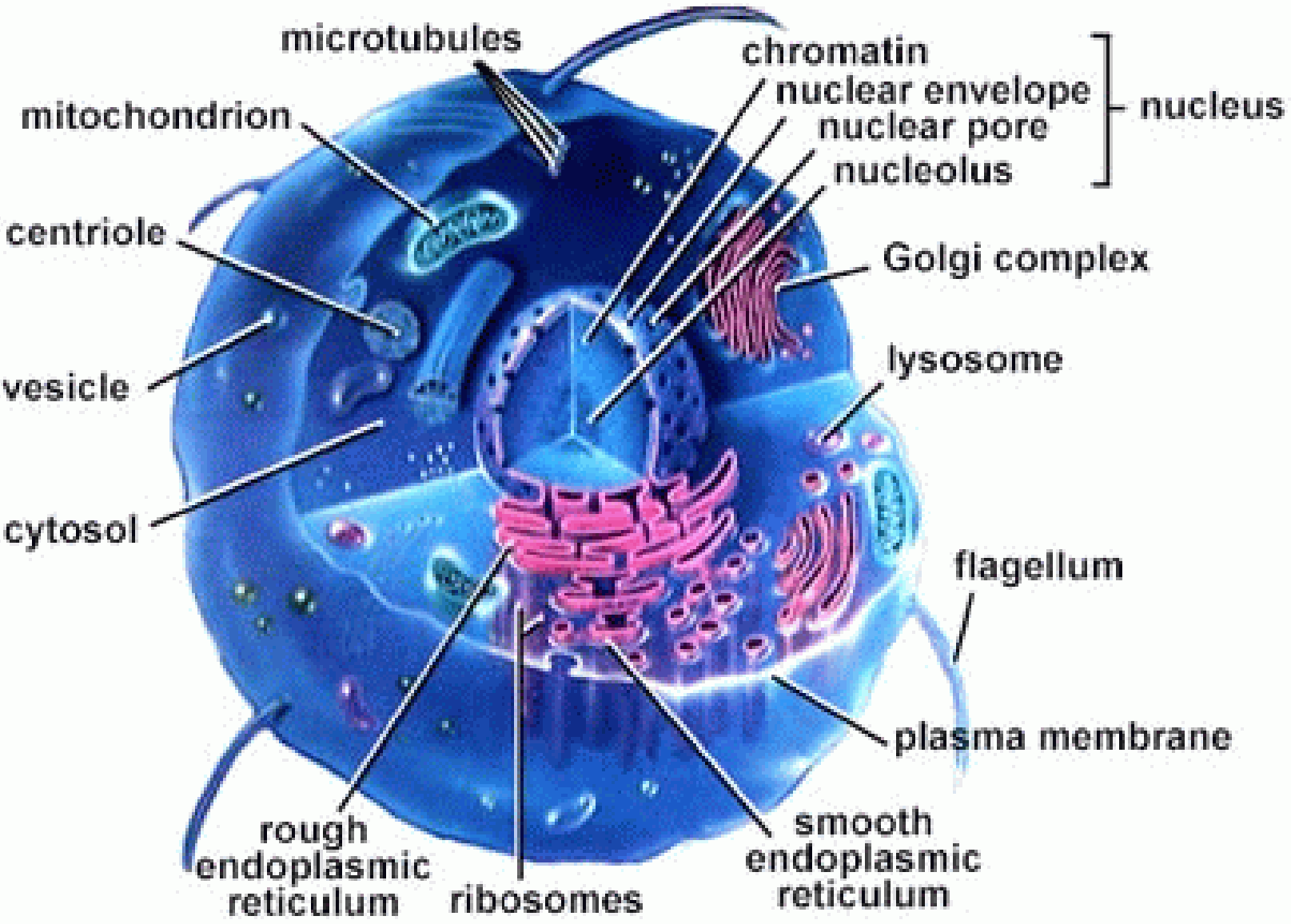
A short introduction to molecular biology

Short history of genomics

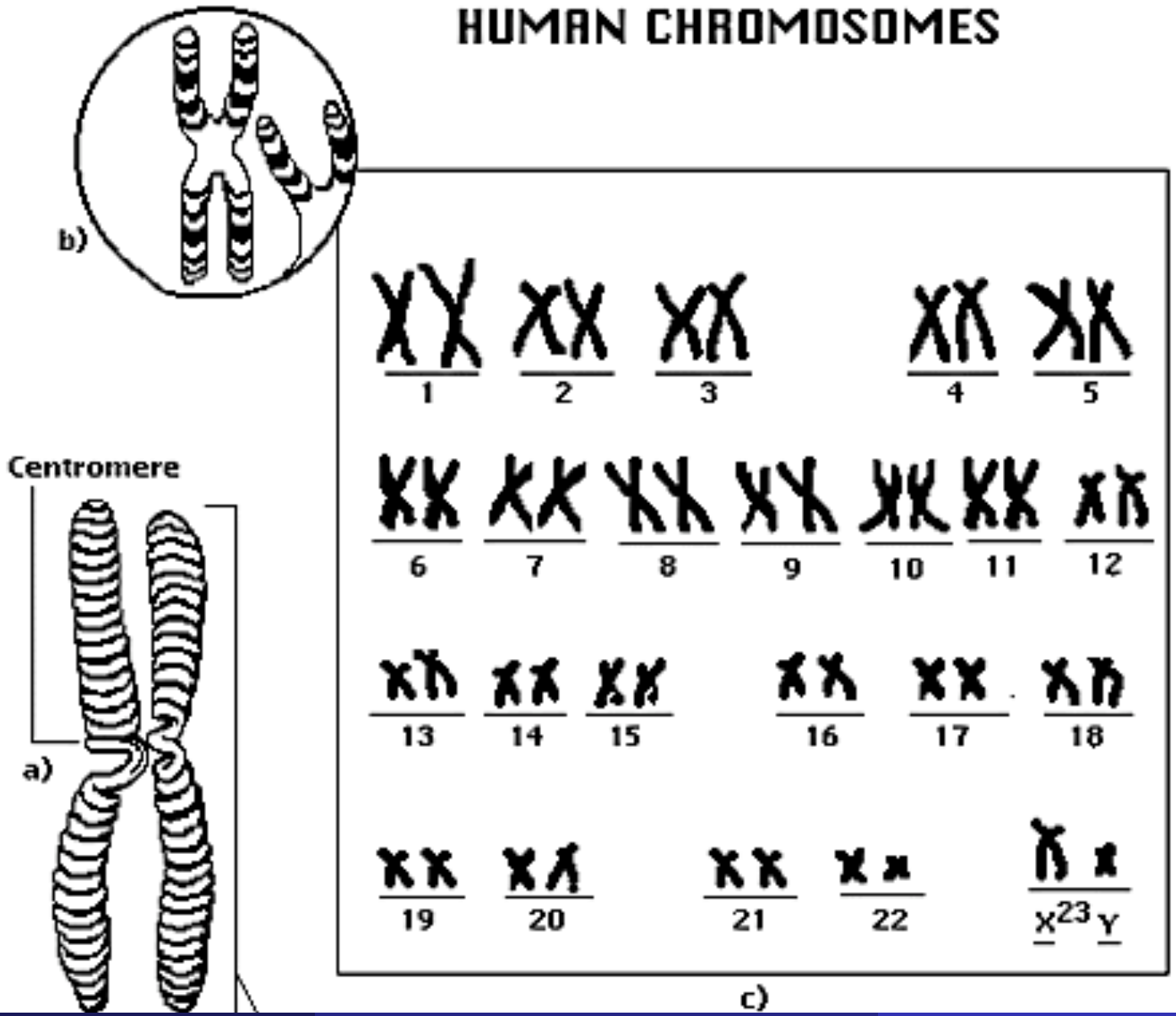


1866 : Laws of heredity (Mendel)
1909 : Morgan and the drosophilists
1944 : DNA supports heredity (Avery)
1953 : Structure of DNA (Crick and Watson)
1966 : Genetic code (Nirenberg)
1960-70 : Genetic engineering
1977 : Method for sequencing (Sanger)
1982 : Creation of Genbank
1990 : Human genome project launched
2003 : Human genome project completed

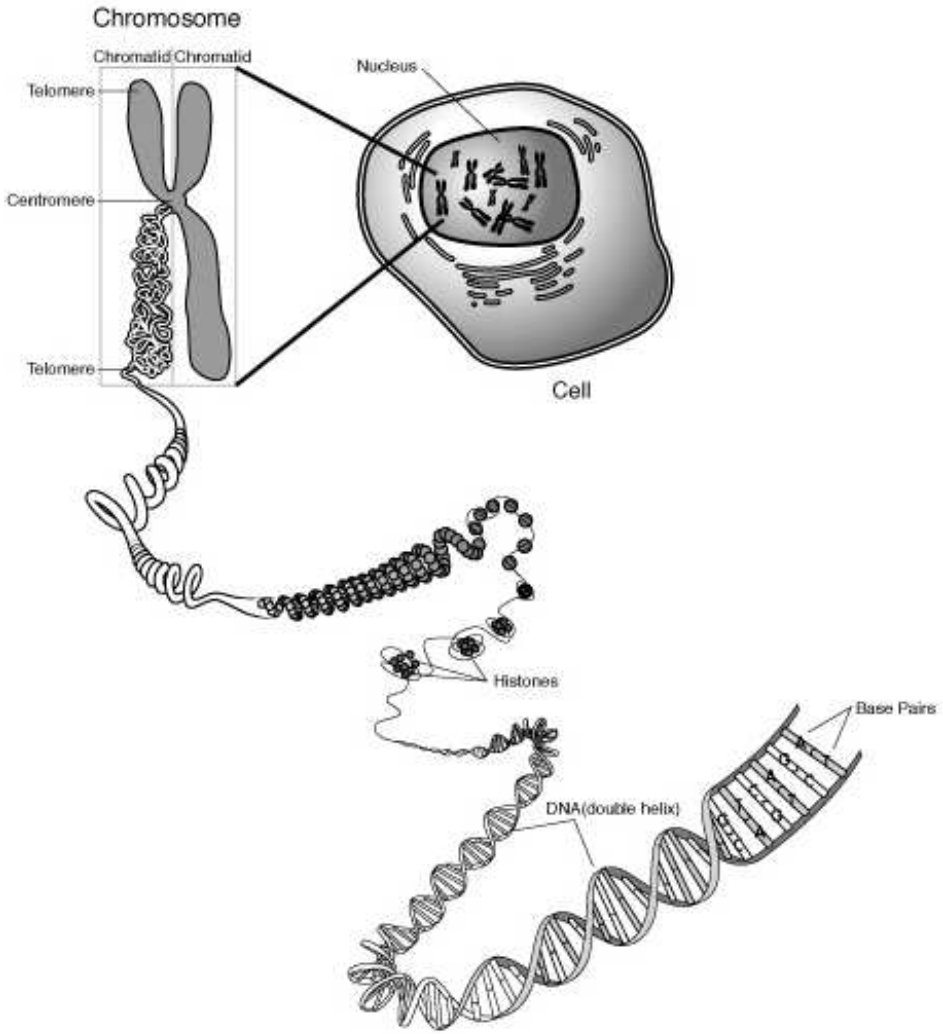
A cell


microtubules
mitochondrion
centriole
vesicle
cytosol
chromatin
nuclear envelope
nuclear pore
nucleolus
nucleus
Golgi complex
lysosome
flagellum
plasma membrane
rough endoplasmic reticulum
ribosomes
smooth endoplasmic reticulum

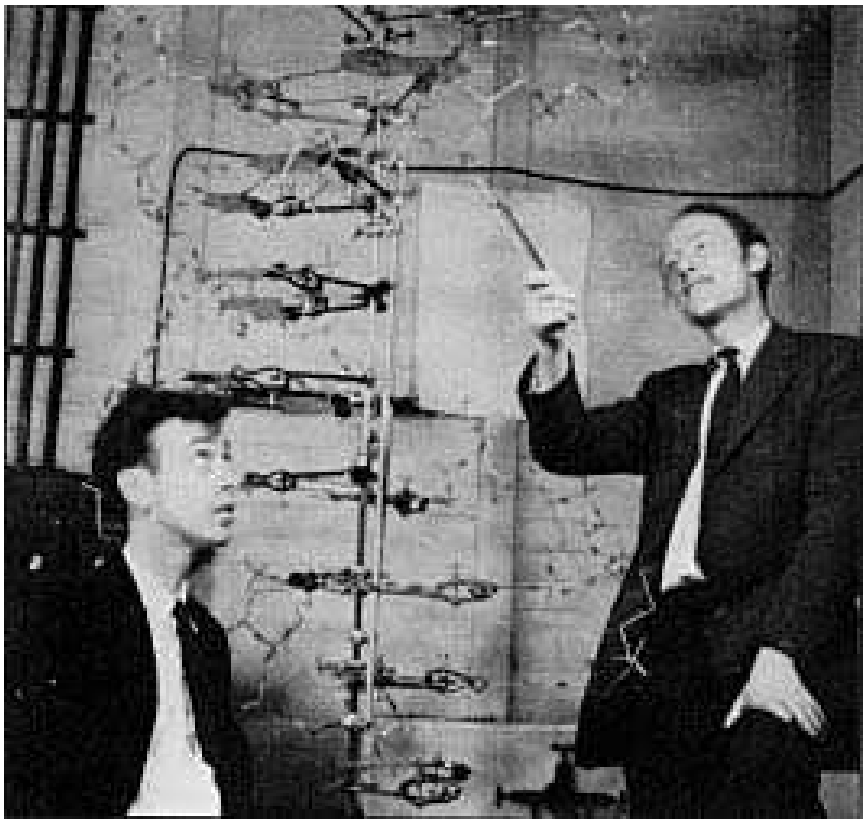
Chromosomes



Chromosomes and DNA

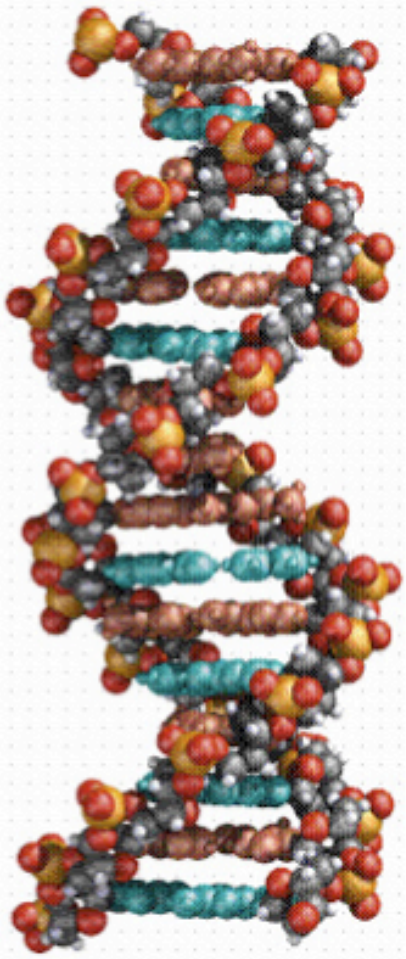


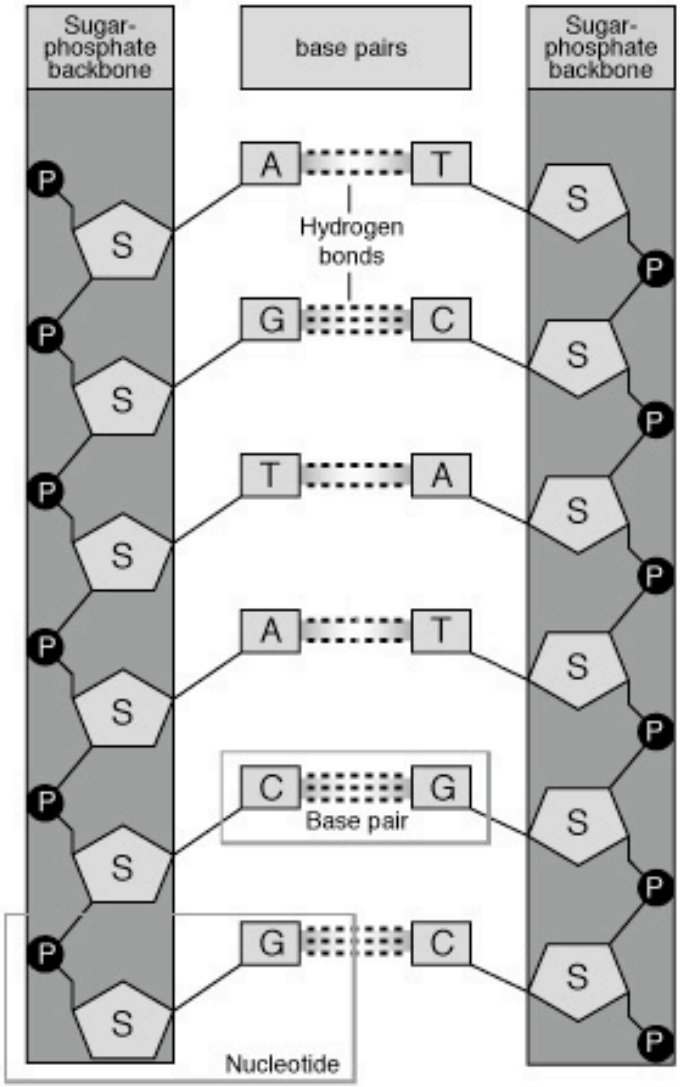
Structure of DNA



"We wish to suggest a structure for the salt of desoxyribose nucleic acid (D.N.A.). This structure have novel features which are of considerable biological interest" (Watson and Crick, 1953)

The double helix



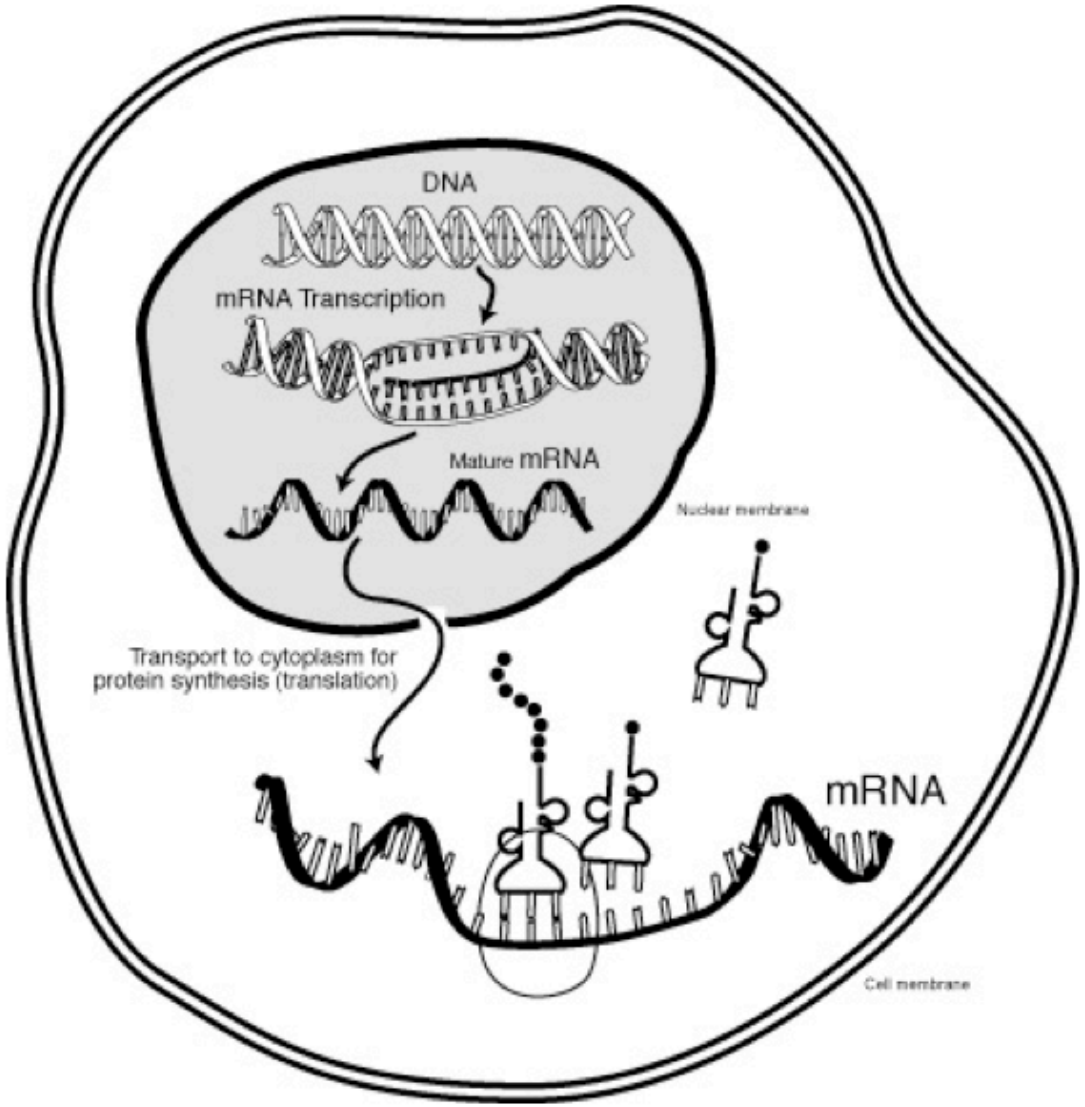


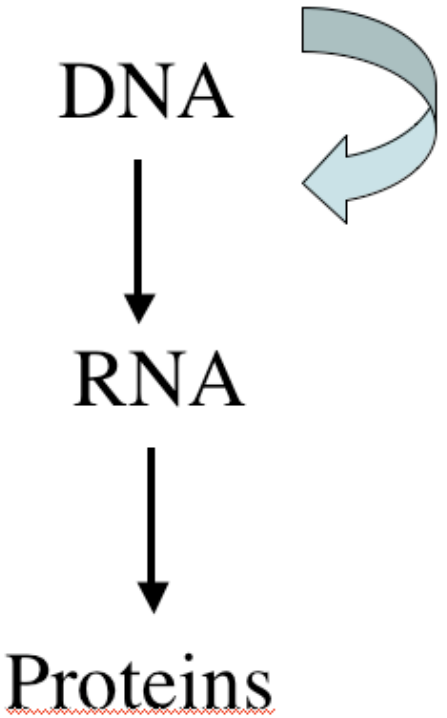
Genomes

(Almost) all cells in an organism share the same DNA, called genome.

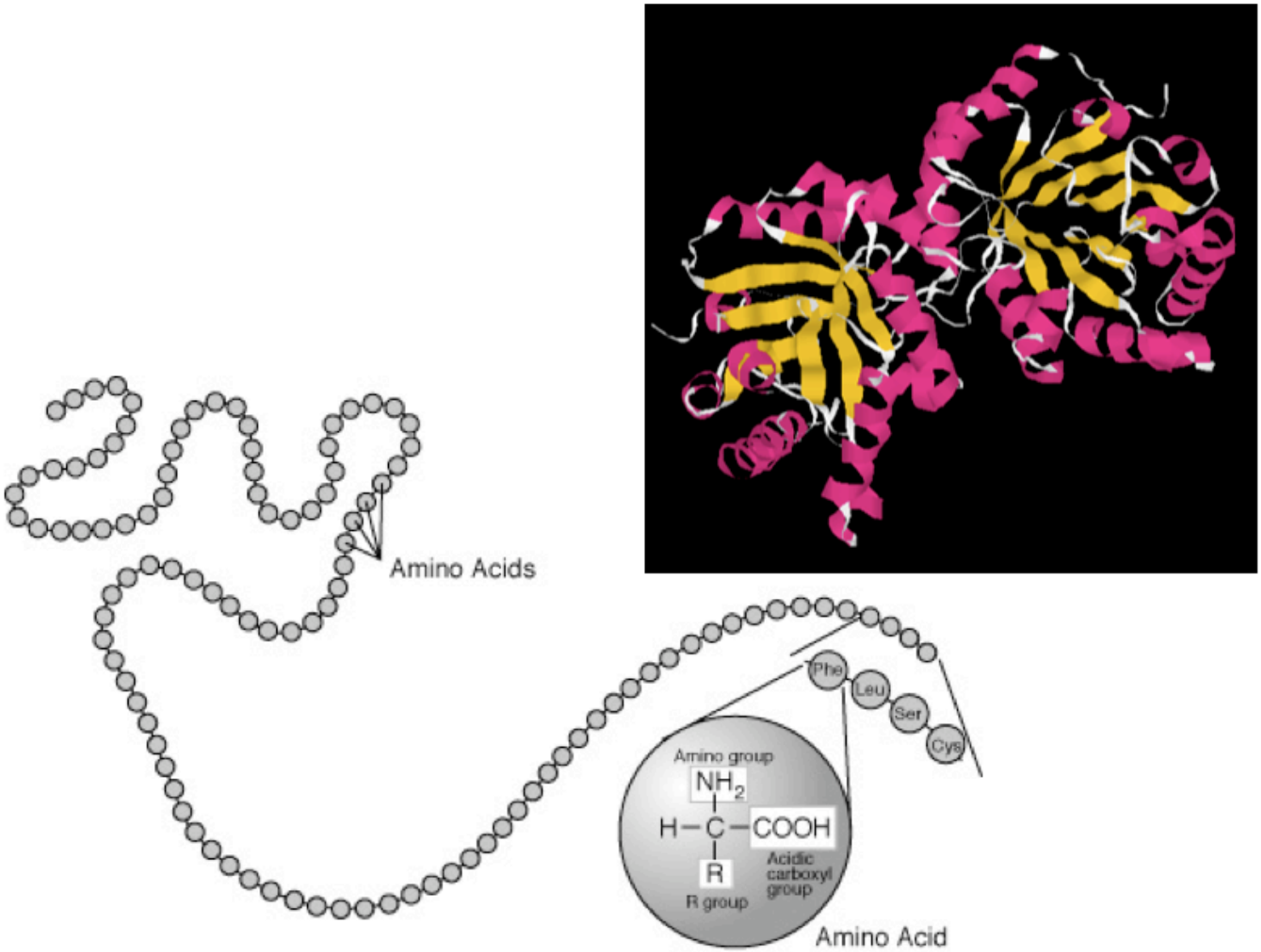
Organism	Chromosomes	Genome size (bp)
Bacteria	1	400,000 a 10,000,000
Yeast	12	14,000,000
Fly	4	300,000,000
Human	46	6,000,000,000

Central dogma

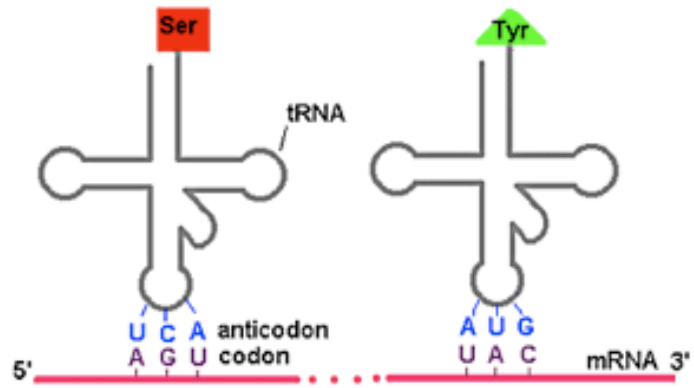




Proteins


Amino Acids
Phe
Leu
Ser
Cys
Amino group
NH_2
H—C—COOH
R
R group
Acidic carboxyl group
Amino Acid

Genetic code



2nd base in codon

1st base in codon	U	C	A	G	3rd base in codon
U	Phe Phe Leu Leu	Ser Ser Ser Ser	Tyr Tyr STOP STOP	Cys Cys STOP Trp	U C A G
C	Leu Leu Leu Leu	Pro Pro Pro Pro	His His Gln Gln	Arg Arg Arg Arg	U C A G
A	Ile Ile Ile Met	Thr Thr Thr Thr	Asn Asn Lys Lys	Ser Ser Arg Arg	U C A G
G	Val Val Val Val	Ala Ala Ala Ala	Asp Asp Glu Glu	Gly Gly Gly Gly	U C A G

The Genetic Code

DNA = 4 letters (ATCG)

↓

RNA = 4 letters (AUCG)

↓

Protein = 20 letters (amino acids)

1 amino acid

=

3 nucleotides

Human genome project

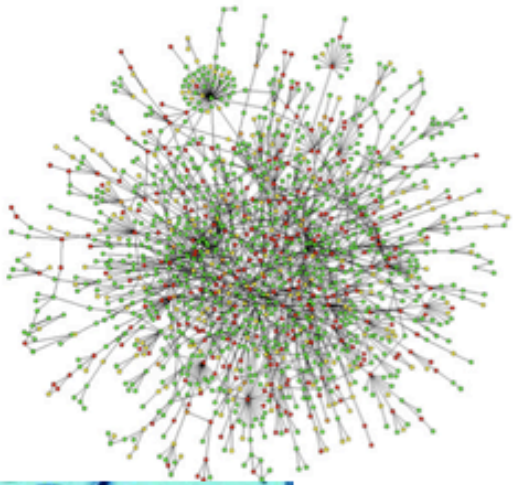
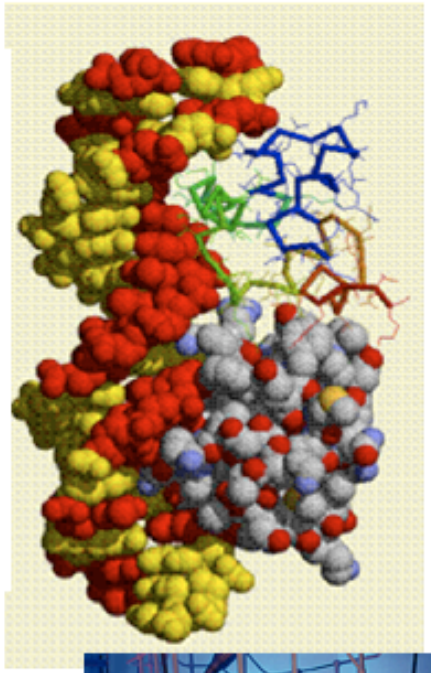
- Goal : sequence the 3,000,000,000 bases of the human genome
- Consortium with 20 labs, 6 countries
- Cost : about 3,000,000,000 USD

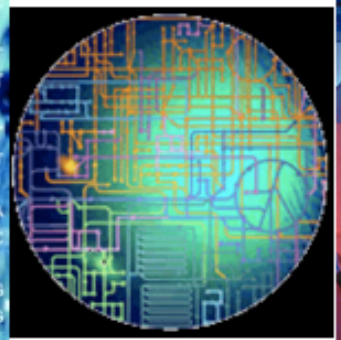
2003: End of genomics era

Findings

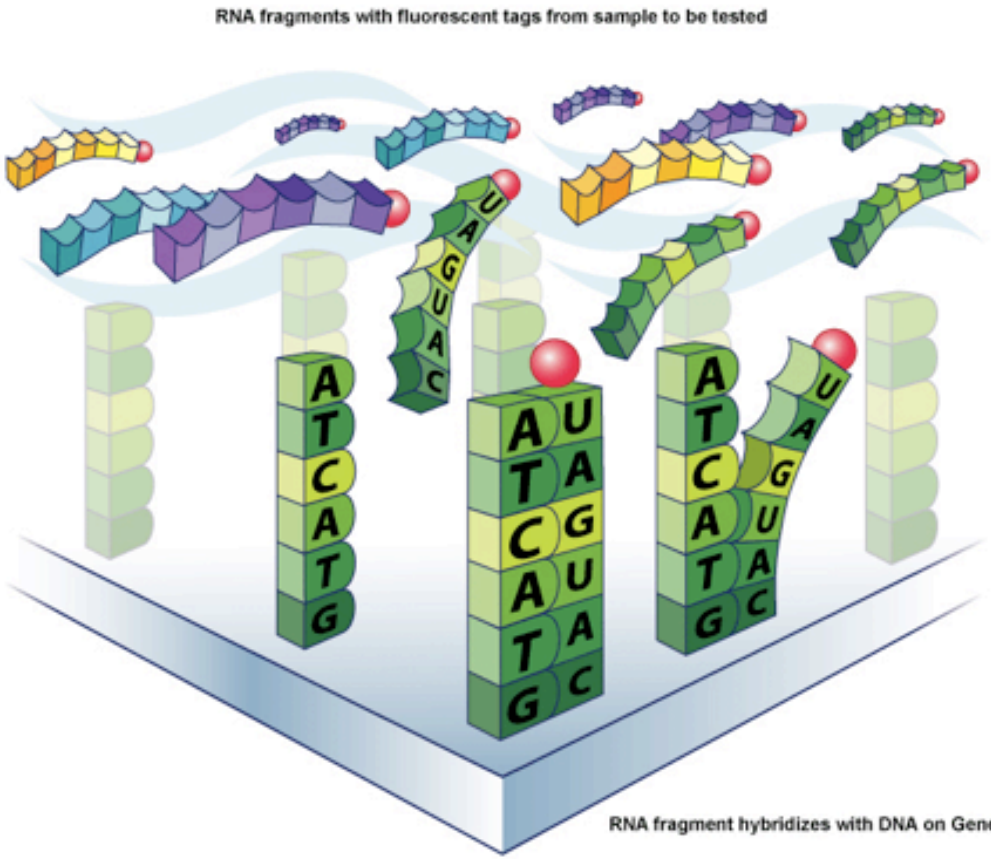
- About 25,000 genes only (representing 1.2% of the genome)
- Automatic gene finding with graphical models
- 97% of the genome is considered “junk DNA”
- Superposition of a variety of signals (many to be discovered)

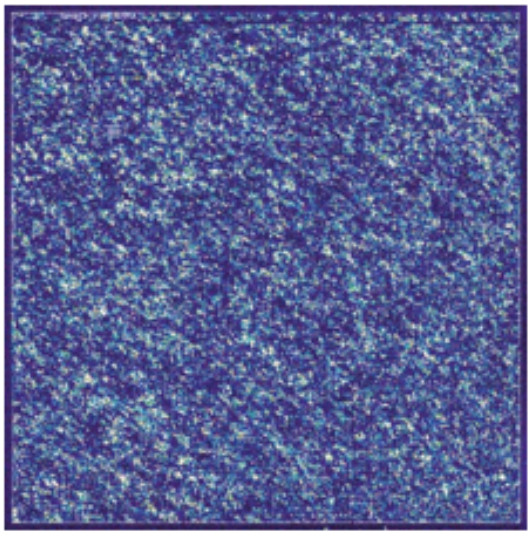
The post-genomic technological revolution


Example: DNA microarrays





Data available

- Sequences (genomes, genes, proteins, regulatory regions, peptides...)
- 3D structures (proteins, DNA, RNA...)
- Networks (interaction, regulation...)
- Time series (transcriptome, proteome, ...)
- Population data (SNPs, virus evolution...)

Expectations

Biology

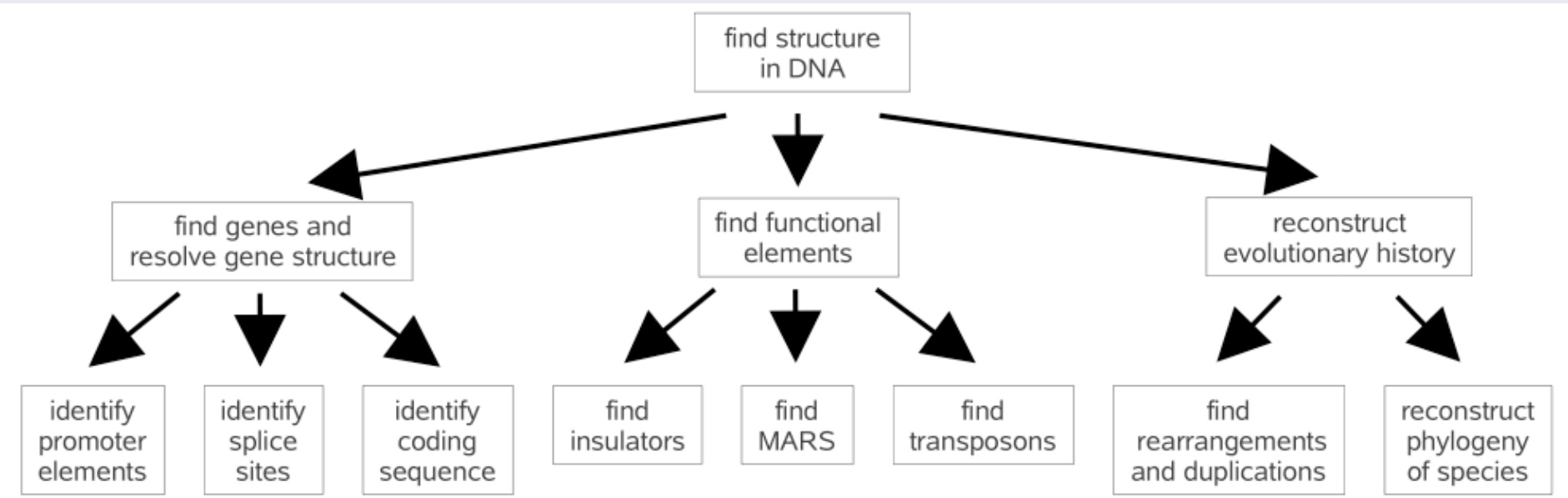
- Structure and functions of all molecules
- Interaction, regulation, systems biology
- Evolution, reverse engineering, synthetic biology..

Medicine

- Molecular basis of disease (cancer, virus infection...)
- Early diagnosis and prognosis
- New drug targets and drugs
- Personalized medicine (pharmagenomics)

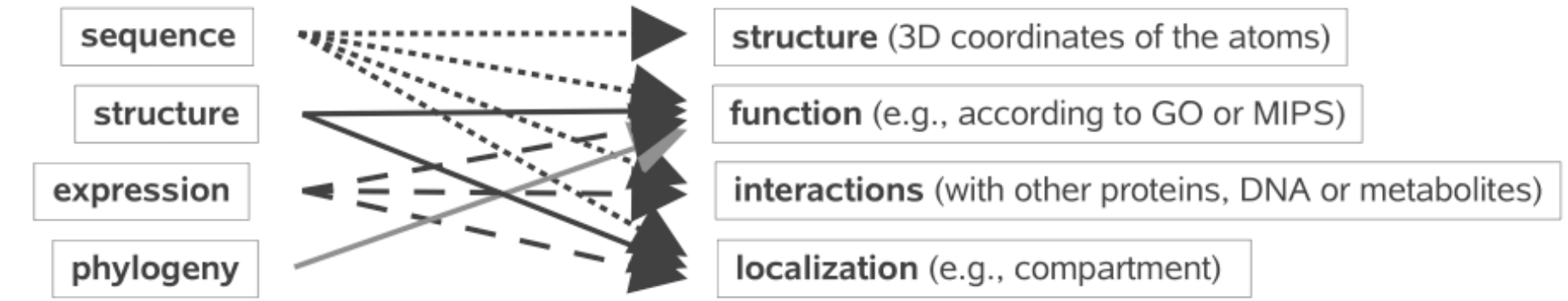
Some computational challenges

Genomics

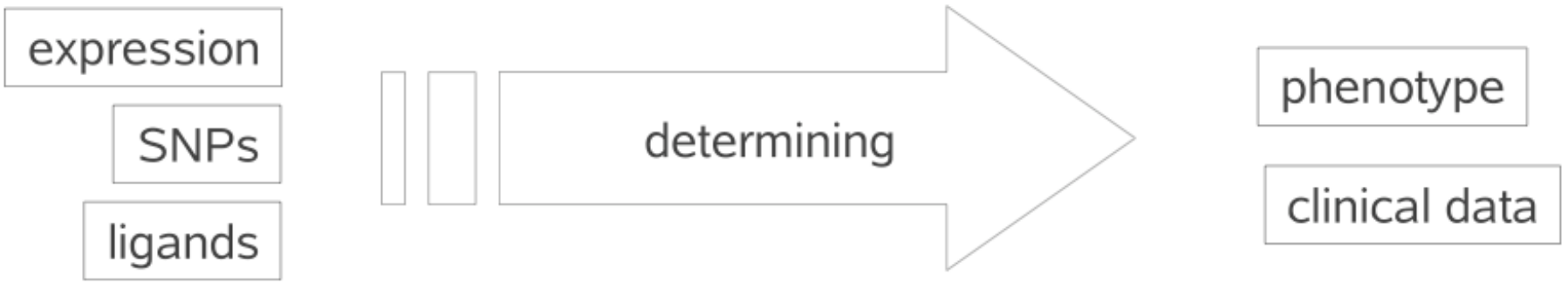


Some computational challenges

Proteomics

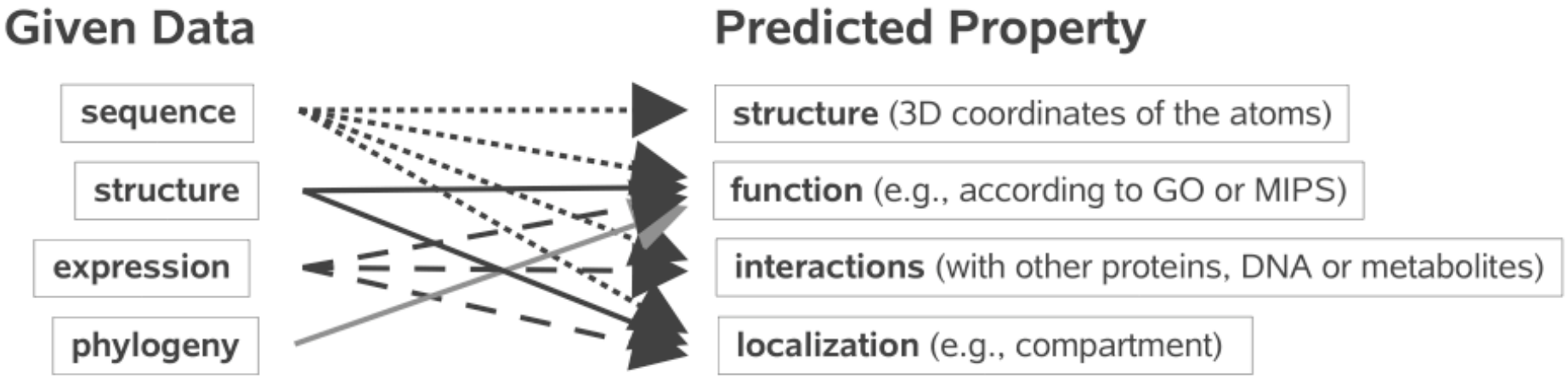


Pharmacogenomics

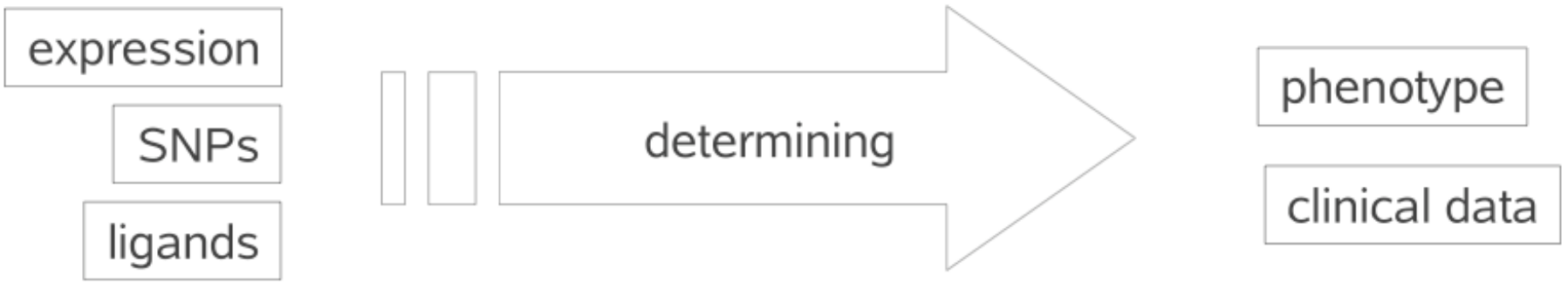


Some computational challenges

Proteomics



Pharmacogenomics



Some computational challenges

Systems biology

- Reconstruction of gene networks from large-scale heterogeneous data
- Simulation of complex biological systems (at the level of pathways, cell, tissues or whole organism)
- Modeling of systems-level phenomena

Summary

- Data revolution is occurring in biology, data-driven biology has started
- Despite the cultural gap math / computer science / physics are increasingly needed
- Machine learning is already playing a central role, and is likely to keep doing so
- Data are often noisy, structured, heterogeneous etc...
- Problems are usually not well defined

Part 2

Kernels and Kernel Methods

Outline

Biological data

Modern technologies provide data that are often:

- in large dimension (e.g., microarrays or proteomics data)
- structured (e.g., gene sequences, small molecules, interaction networks, phylogenetic trees...)
- heterogeneous (e.g., vectors, sequences, graphs to describe the same protein)
- in large quantities (e.g., $> 10^6$ protein sequences)

SVM and kernel methods lend themselves particularly well to these constraints (of course, there is much room for other approaches!)

Kernel methods for bioinformatics

Features

SVM and kernel method have in particular the following properties:

- statistical approaches to process large datasets
- kernels for structured objects
- multiple kernel learning for heterogeneous data

References

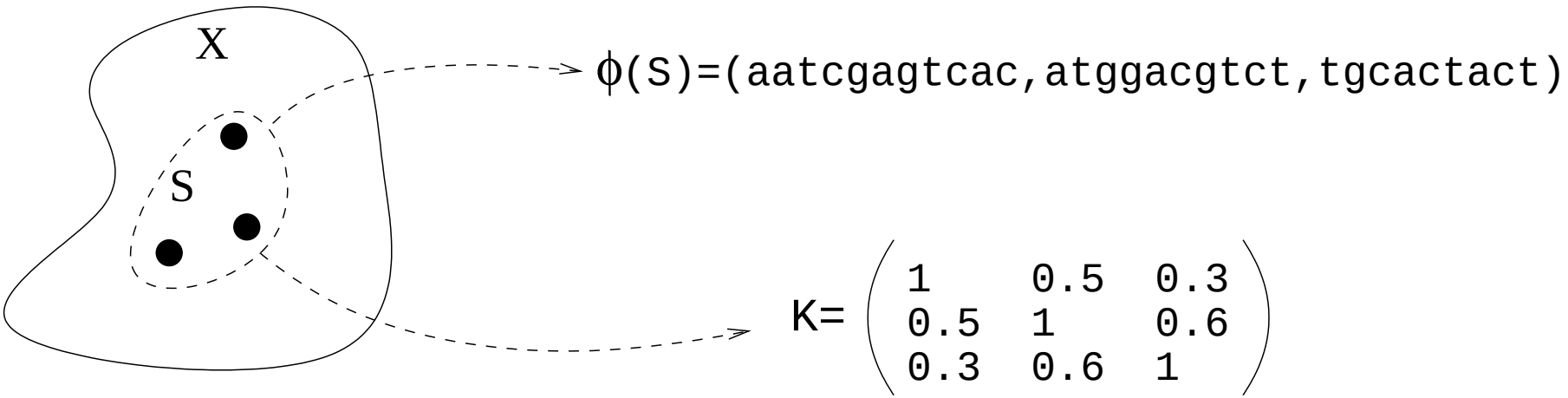
More than 500 references since 1998:

`http://cbio.ensmp.fr/~vert/svn/bibli/html/biosvm.html`

Outline

Motivations

- Develop versatile algorithms to process and analyze data
- No hypothesis made regarding the type of data (vectors, strings, graphs, images, ...)
- Instead we study methods based on pairwise comparisons.



Positive Definite Kernels

Definition

A positive definite (p.d.) kernel on the set $\mathcal{X}$ is a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ symmetric:

$$\forall \left(\mathbf{x}, \mathbf{x}'\right) \in \mathcal{X}^2, \quad K\left(\mathbf{x}, \mathbf{x}'\right) = K\left(\mathbf{x}', \mathbf{x}\right),$$

and which satisfies, for all $N \in \mathbb{N}$, $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \in \mathcal{X}^N$ et $(a_1, a_2, \ldots, a_N) \in \mathbb{R}^N$:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} a_i a_j K\left(\mathbf{x}_i, \mathbf{x}_j\right) \geq 0.$$

Examples

Classical kernels for vectors ($\mathcal{X} = \mathbb{R}^p$) include:

- The linear kernel

$$K_{lin}\left(\mathbf{x}, \mathbf{x}'\right) = \mathbf{x}^\top \mathbf{x}' \ .$$

- The polynomial kernel

$$K_{poly}\left(\mathbf{x}, \mathbf{x}'\right) = \left(\mathbf{x}^\top \mathbf{x}' + a\right)^d \ .$$

- The Gaussian RBF kernel:

$$K_{Gaussian}\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}' \|^2}{2\sigma^2}\right) \ .$$

Kernels as Inner Products

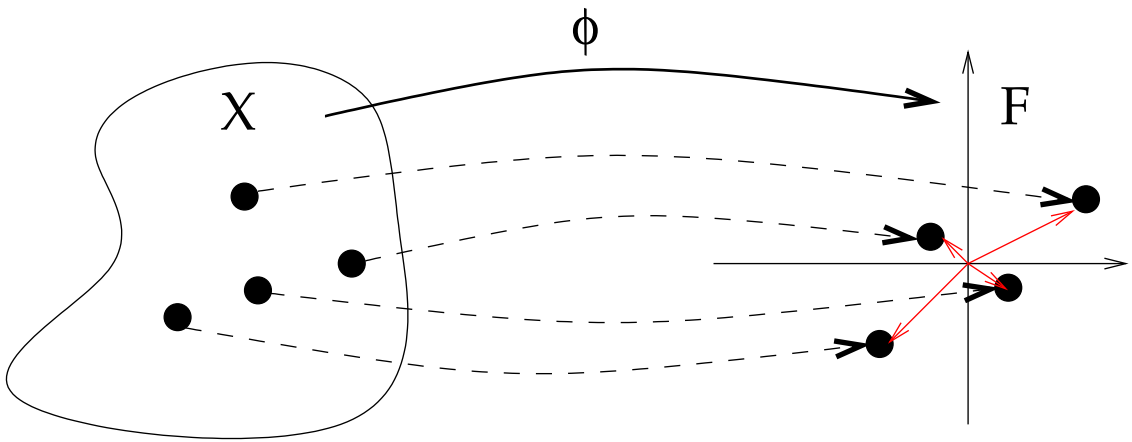
Theorem (Aronszajn, 1950)

K is a p.d. kernel on the set $\mathcal{X}$ if and only if there exists a Hilbert space $\mathcal{H}$ and a mapping

$$\Phi : \mathcal{X} \mapsto \mathcal{H} \ ,$$

such that, for any $\mathbf{x}, \mathbf{x}'$ in $\mathcal{X}$:

$$K\left(\mathbf{x}, \mathbf{x}'\right) = \left\langle \Phi\left(\mathbf{x}\right), \Phi\left(\mathbf{x}'\right) \right\rangle_{\mathcal{H}} \ .$$



Proof

If K can be written as:

$$K\left(\mathbf{x},\mathbf{x}'\right)=\left\langle\Phi\left(\mathbf{x}\right),\Phi\left(\mathbf{x}'\right)\right\rangle_{\mathcal{H}}\ ,$$

then it is p.d. because:

- $\left\langle\Phi\left(\mathbf{x}\right),\Phi\left(\mathbf{x}'\right)\right\rangle_{\mathcal{H}}=\left\langle\Phi\left(\mathbf{x}'\right),\Phi\left(\mathbf{x}\right)\right\rangle_{\mathcal{H}}$,
- $\sum_{i=1}^{N}\sum_{j=1}^{N}a_i a_j\left\langle\Phi\left(\mathbf{x}_i\right),\Phi\left(\mathbf{x}_j\right)\right\rangle_{\mathcal{H}}=\|\sum_{i=1}^{N}a_i\Phi\left(\mathbf{x}_i\right)\|^2\geq 0$.

The converse was proved by Mercer in 1905 for continuous K on compact $\mathcal{X}$ (called Mercer kernels), in 1941 by Kolmogorov for countable $\mathcal{X}$, and by Aronszajn (1950) for the general case. In order to prove it in full generality we must introduce the notion of *reproducing Hilbert space*.

Proof

If K can be written as:

$$K\left(\mathbf{x},\mathbf{x}'\right)=\left\langle \Phi\left(\mathbf{x}\right),\Phi\left(\mathbf{x}'\right)\right\rangle_{\mathcal{H}}\ ,$$

then it is p.d. because:

- $\left\langle \Phi\left(\mathbf{x}\right),\Phi\left(\mathbf{x}'\right)\right\rangle_{\mathcal{H}}=\left\langle \Phi\left(\mathbf{x}'\right),\Phi\left(\mathbf{x}\right)\right\rangle_{\mathcal{H}}$,
- $\sum_{i=1}^{N}\sum_{j=1}^{N}a_{i}a_{j}\left\langle \Phi\left(\mathbf{x}_{i}\right),\Phi\left(\mathbf{x}_{j}\right)\right\rangle_{\mathcal{H}}=\|\sum_{i=1}^{N}a_{i}\Phi\left(\mathbf{x}_{i}\right)\|^{2}\geq 0$.

The converse was proved by Mercer in 1905 for continuous K on compact $\mathcal{X}$ (called Mercer kernels), in 1941 by Kolmogorov for countable $\mathcal{X}$, and by Aronszajn (1950) for the general case. In order to prove it in full generality we must introduce the notion of *reproducing Hilbert space*.

Reproducing Kernel Hilbert Space

- To each p.d. kernel on $\mathcal{X}$ is associated a unique Hilbert space of function $\mathcal{X} \to \mathbb{R}$, called the reproducing kernel Hilbert space (RKHS) $\mathcal{H}$.
- Typical functions are:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}_i, \mathbf{x}) \ ,$$

with norm

$$\| f \|_{\mathcal{H}}^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \ .$$

Reproducing property

- For any $\mathbf{x} \in \mathcal{X}$ let $K_{\mathbf{x}} : \mathcal{X} \to \mathbb{R}$ be defined by:

$$K_{\mathbf{x}}\left(\mathbf{x}'\right) = K\left(\mathbf{x}, \mathbf{x}'\right), \quad \forall \mathbf{x}' \in \mathcal{X} .$$

- In the RKHS it holds that:

$$f\left(\mathbf{x}\right) = \left\langle f, K_{\mathbf{x}} \right\rangle_{\mathcal{H}}, \quad \forall f \in \mathcal{H}, \mathbf{x} \in \mathcal{X} .$$

- Reproducing property:

$$K\left(\mathbf{x}, \mathbf{x}'\right) = \left\langle K_{\mathbf{x}}, K_{\mathbf{x}'} \right\rangle_{\mathcal{H}}, \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X} .$$

This proves Aronsjazn's theorem by taking $\Phi : \mathcal{X} \to \mathcal{H}$ defined by

$$\Phi(\mathbf{x}) = K_{\mathbf{x}} . \quad \square$$

Characterization of RKHS

In fact the RKHS is completely characterized by the following properties:

Theorem

The RKHS $\mathcal{H}$ is the unique Hilbert space of functions that satisfies:

- *For any $\mathbf{x} \in \mathcal{X}$, $K_{\mathbf{x}} \in \mathcal{H}$,*
- *For any $\mathbf{x} \in \mathcal{X}$ and $f \in \mathcal{H}$,*

$$f(\mathbf{x}) = \langle f, K_{\mathbf{x}} \rangle_{\mathcal{H}} \ .$$

Smoothness functional

By Cauchy-Schwarz we have, for any function $f \in \mathcal{H}$ and any two points $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$:

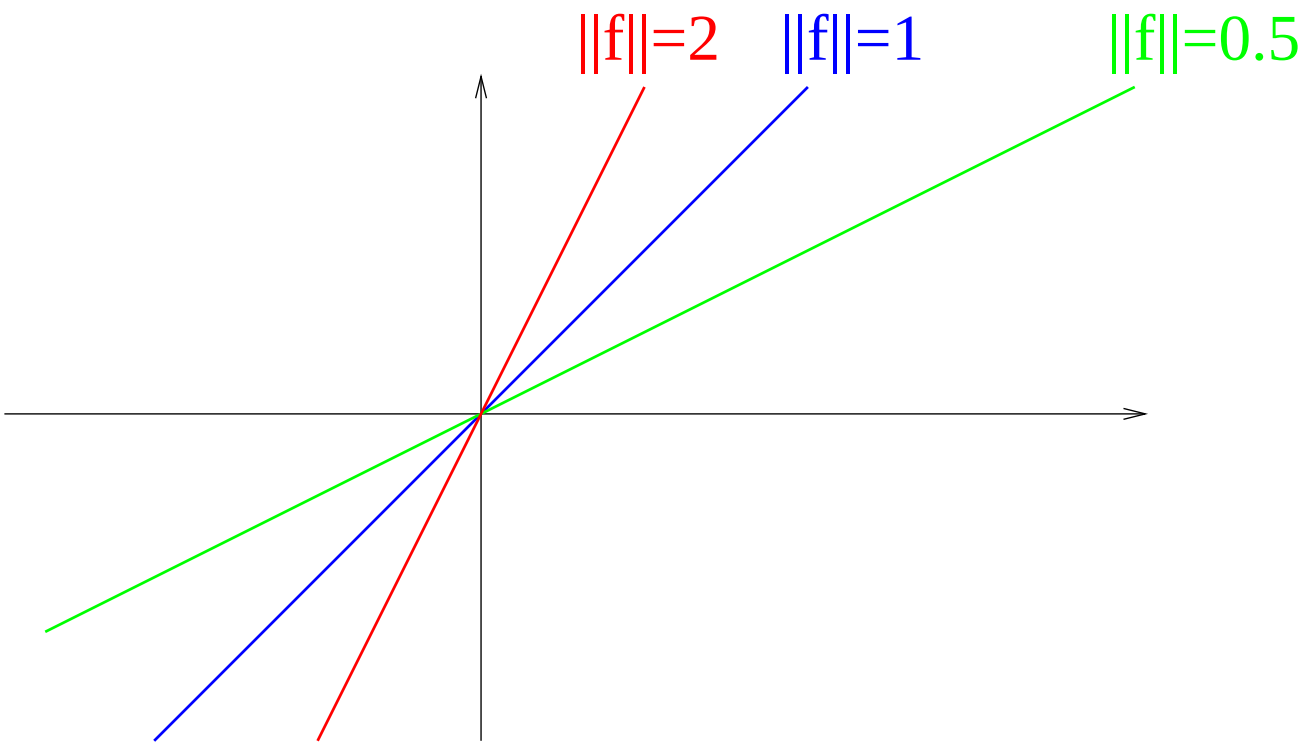
$$\begin{aligned} \left| f(\mathbf{x}) - f(\mathbf{x}') \right| &= \left| \langle f, K_{\mathbf{x}} - K_{\mathbf{x}'} \rangle_{\mathcal{H}} \right| \\ &\leq \| f \|_{\mathcal{H}} \times \| K_{\mathbf{x}} - K_{\mathbf{x}'} \|_{\mathcal{H}} \\ &= \| f \|_{\mathcal{H}} \times d_K(\mathbf{x}, \mathbf{x}') \ . \end{aligned}$$

The norm of a function in the RKHS controls how fast the function varies over $\mathcal{X}$ with respect to the geometry defined by the kernel.

Small norm $\Longrightarrow$ slow variations.

Example: Linear kernel

$$\begin{cases} K_{lin}(\mathbf{x}, \mathbf{x}') & = \mathbf{x}^\top \mathbf{x}' \ . \\ f(\mathbf{x}) & = w^\top x \ , \\ \| f \|_{\mathcal{H}} & = \| w \|_2 \ . \end{cases}$$



Examples: Gaussian RBF kernel

$$K_{Gaussian}\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}' \|^2}{2\sigma^2}\right) ,$$

$$f\left(\mathbf{x}\right) = \sum_{i=1}^{n} \alpha_i \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}_i \|^2}{2\sigma^2}\right) ,$$

$$\| f \|_{\mathcal{H}}^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}_i \|^2}{2\sigma^2}\right)$$

$$= \int \left| \hat{f}(\omega) \right|^2 e^{\frac{\sigma^2\omega^2}{2}} d\omega .$$

Outline

Pattern recognition and regression

Classical setting

- Input variables $\mathbf{x} \in \mathcal{X}$
- Output $y \in \mathcal{Y}$ with $\mathcal{Y} = \{-1, 1\}$ (pattern recognition) or $\mathcal{Y} = \mathbb{R}$ (regression)
- Training set $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$.
- Goal: learn the mapping $f : \mathcal{X} \to \mathcal{Y}$

Kernel methods

General formulation

1. Define a loss function $L(y, \hat{y})$
2. Solve the problem:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} L\left(y_i, f\left(\mathbf{x}_i\right)\right) + \lambda \| f \|_{\mathcal{H}}^2 \ .$$

λ controls the trade-off between fitting the data and being a smooth function.

Examples

Loss functions

- Support vector machines for classification:

$$L_{hinge}(y, \hat{y}) = \max(0, 1 - y\hat{y}) \ .$$

- Kernel logistic regression

$$L_{logit} = \log\left(1 + e^{-y\hat{y}}\right) \ .$$

- Kernel ridge regression

$$L_{square}(y, \hat{y}) = (y - \hat{y})^2 \ .$$

Examples

Loss functions

- Support vector machines for classification:

$$L_{hinge}(y, \hat{y}) = \max(0, 1 - y\hat{y}) \ .$$

- Kernel logistic regression

$$L_{logit} = \log\left(1 + e^{-y\hat{y}}\right) \ .$$

- Kernel ridge regression

$$L_{square}(y, \hat{y}) = (y - \hat{y})^2 \ .$$

Examples

Loss functions

- Support vector machines for classification:

$$L_{hinge}(y, \hat{y}) = \max(0, 1 - y\hat{y}) \ .$$

- Kernel logistic regression

$$L_{logit} = \log\left(1 + e^{-y\hat{y}}\right) \ .$$

- Kernel ridge regression

$$L_{square}(y, \hat{y}) = (y - \hat{y})^2 \ .$$

Kernel methods in practice

- Representer theorem: the solution of the optimization problem can in fact always be expanded as:

$$\tilde{f}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i K(\mathbf{x}_i, \mathbf{x})$$

- Plugging this into the optimization problem therefore boils down to a *n*-dimensional optimization problem (convex if L is convex)
- The complexity of the algorithms depend on n, the number of points

Summary

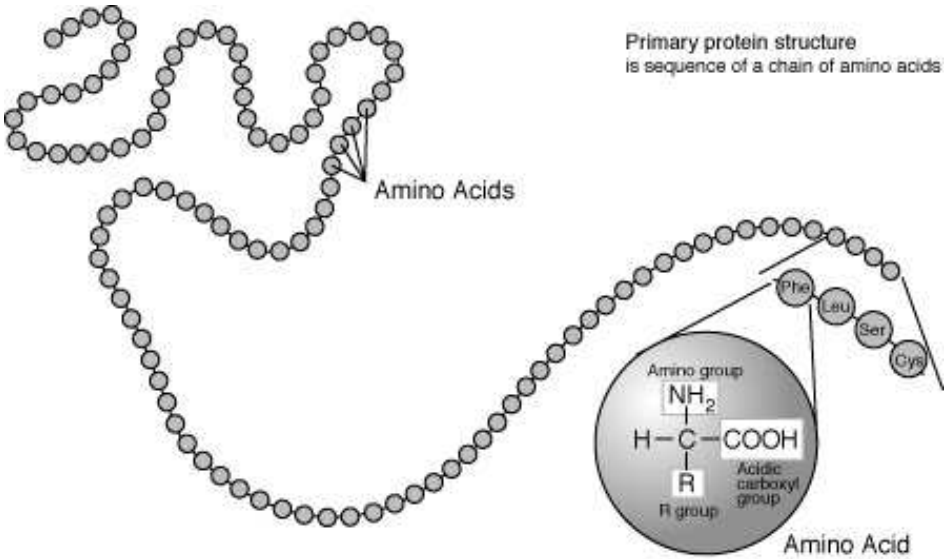
- A kernel defines an implicit geometry on the space of data, although data do not need to have any prior geometric/algebric structure
- Kernel methods learn functions that tend to be smooth with respect to this geometry
- Kernel engineering is the problem of designing specific kernel for specific data and specific tasks. Good place to put prior knowledge!
- We will now see on a practical examples different technical tricks to design kernels.

Part 3

Kernels for Biological Sequences

Outline

Protein sequence



A : Alanine	V : Valine	L : Leucine
F : Phenylalanine	P : Proline	M : Méthionine
E : Acide glutamique	K : Lysine	R : Arginine
T : Threonine	C : Cysteine	N : Asparagine
H : Histidine	V : Thyrosine	W : Tryptophane
I : Isoleucine	S : Sérine	Q : Glutamine
D : Acide aspartique	G : Glycine	

Challenges with protein sequences

- A protein sequences can be seen as a variable-length sequence over the 20-letter alphabet of amino-acids, e.g., insuline: `FVNQHLCGSHLVEALYLVCGERGFFYTPKA`
- These sequences are produced at a fast rate (result of the sequencing programs)
- Need for algorithms to compare, classify, analyze these sequences
- Applications: classification into functional or structural classes, prediction of cellular localization and interactions, ...

Kernels for protein sequences

- Kernel methods have been widely investigated since Jaakkola et al.'s seminal paper (1998).
- What is a good kernel?
  - it should be mathematically valid (symmetric, p.d. or c.p.d.)
  - fast to compute
  - adapted to the problem (give good performances)

Kernel engineering for protein sequences

- Define a (possibly high-dimensional) feature space of interest
  - Physico-chemical kernels
  - Spectrum, mismatch, substring kernels
  - Pairwise, motif kernels
- Derive a kernel from a generative model
  - Fisher kernel
  - Mutual information kernel
  - Marginalized kernel
- Derive a kernel from a similarity measure
  - Local alignment kernel

Kernel engineering for protein sequences

- Define a (possibly high-dimensional) feature space of interest
  - Physico-chemical kernels
  - Spectrum, mismatch, substring kernels
  - Pairwise, motif kernels
- Derive a kernel from a generative model
  - Fisher kernel
  - Mutual information kernel
  - Marginalized kernel
- Derive a kernel from a similarity measure
  - Local alignment kernel

Kernel engineering for protein sequences

- Define a (possibly high-dimensional) feature space of interest
  - Physico-chemical kernels
  - Spectrum, mismatch, substring kernels
  - Pairwise, motif kernels
- Derive a kernel from a generative model
  - Fisher kernel
  - Mutual information kernel
  - Marginalized kernel
- Derive a kernel from a similarity measure
  - Local alignment kernel

Outline

Vector embedding for strings

The idea

Represent each sequence $\mathbf{x}$ by a fixed-length numerical vector $\Phi(\mathbf{x}) \in \mathbb{R}^n$. How to perform this embedding?

Physico-chemical kernel

Extract relevant features, such as:

- length of the sequence
- time series analysis of numerical physico-chemical properties of amino-acids along the sequence (e.g., polarity, hydrophobicity), using for example:
  - Fourier transforms (Wang et al., 2004)
  - Autocorrelation functions (Zhang et al., 2003)

$$r_j = \frac{1}{n-j} \sum_{i=1}^{n-j} h_i h_{i+j}$$

Vector embedding for strings

The idea

Represent each sequence $\mathbf{x}$ by a fixed-length numerical vector $\Phi(\mathbf{x}) \in \mathbb{R}^n$. How to perform this embedding?

Physico-chemical kernel

Extract relevant features, such as:

- length of the sequence
- time series analysis of numerical physico-chemical properties of amino-acids along the sequence (e.g., polarity, hydrophobicity), using for example:
  - Fourier transforms (Wang et al., 2004)
  - Autocorrelation functions (Zhang et al., 2003)

$$r_j = \frac{1}{n-j} \sum_{i=1}^{n-j} h_i h_{i+j}$$

Substring indexation

The approach

Alternatively, index the feature space by fixed-length strings, i.e.,

$$\Phi(\mathbf{x}) = (\Phi_u(\mathbf{x}))_{u \in \mathcal{A}^k}$$

where $\Phi_u(\mathbf{x})$ can be:

- the number of occurrences of u in $\mathbf{x}$ (without gaps) : spectrum kernel (Leslie et al., 2002)
- the number of occurrences of u in $\mathbf{x}$ up to m mismatches (without gaps) : mismatch kernel (Leslie et al., 2004)
- the number of occurrences of u in $\mathbf{x}$ allowing gaps, with a weight decaying exponentially with the number of gaps : substring kernel (Lohdi et al., 2002)

Example: spectrum kernel

- The 3-spectrum of

$$\mathbf{x} = \texttt{CGGSLIAMMWFGV}$$

is:

$$\texttt{(CGG,GGS,GSL,SLI,LIA,IAM,AMM,MMW,MWF,WFG,FGV)} \ .$$

- Let $\Phi_u(\mathbf{x})$ denote the number of occurrences of u in $\mathbf{x}$. The k-spectrum kernel is:

$$K(\mathbf{x}, \mathbf{x}') := \sum_{u \in \mathcal{A}^k} \Phi_u(\mathbf{x}) \Phi_u(\mathbf{x}') \ .$$

- This is formally a sum over $|\mathcal{A}|^k$ terms, but at most $|\mathbf{x}| - k + 1$ terms are non-zero in $\Phi(\mathbf{x})$

Substring indexation in practice

- Implementation in $O(|\mathbf{x}| + |\mathbf{x}'|)$ in memory and time for the spectrum and mismatch kernels (with suffix trees)
- Implementation in $O(|\mathbf{x}| \times |\mathbf{x}'|)$ in memory and time for the substring kernels
- The feature space has high dimension ($|\mathcal{A}|^k$), so learning requires regularized methods (such as SVM)

Dictionary-based indexation

The approach

- Chose a dictionary of sequences $\mathcal{D} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$
- Chose a measure of similarity $s(\mathbf{x}, \mathbf{x}')$
- Define the mapping $\Phi_{\mathcal{D}}(\mathbf{x}) = (s(\mathbf{x}, \mathbf{x}_i))_{\mathbf{x}_i \in \mathcal{D}}$

Examples

This includes:

- Motif kernels (Logan et al., 2001): the dictionary is a library of motifs, the similarity function is a matching function
- Pairwise kernel (Liao & Noble, 2003): the dictionary is the training set, the similarity is a classical measure of similarity between sequences.

Dictionary-based indexation

The approach

- Chose a dictionary of sequences $\mathcal{D} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$
- Chose a measure of similarity $s(\mathbf{x}, \mathbf{x}')$
- Define the mapping $\Phi_{\mathcal{D}}(\mathbf{x}) = (s(\mathbf{x}, \mathbf{x}_i))_{\mathbf{x}_i \in \mathcal{D}}$

Examples

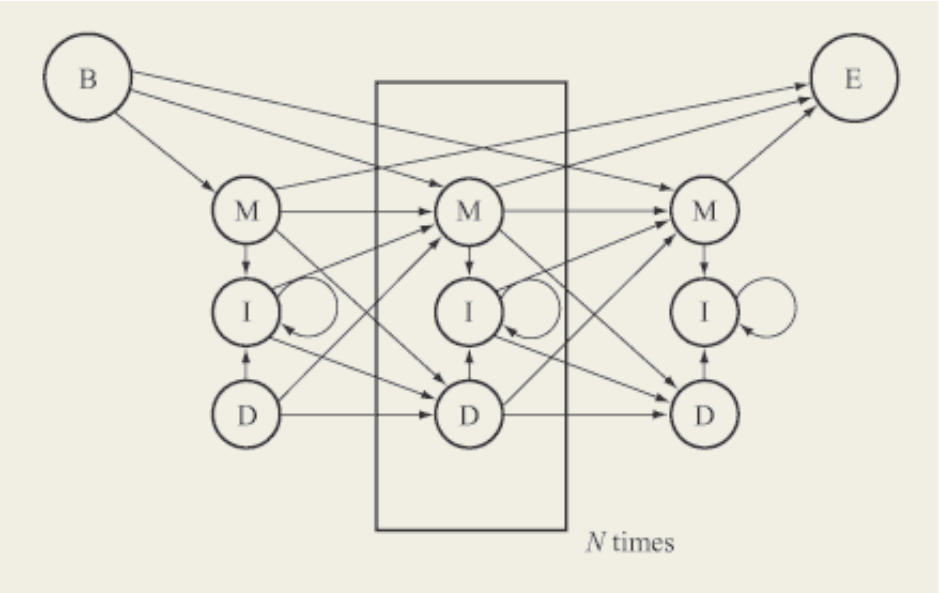
This includes:

- Motif kernels (Logan et al., 2001): the dictionary is a library of motifs, the similarity function is a matching function
- Pairwise kernel (Liao & Noble, 2003): the dictionary is the training set, the similarity is a classical measure of similarity between sequences.

Outline

Probabilistic models for sequences

Probabilistic modeling of biological sequences is older than kernel designs. Important models include HMM for protein sequences, SCFG for RNA sequences.



Parametric model

A model is a family of distribution

$$\{P_\theta, \theta \in \Theta \subset \mathbb{R}^m\} \subset \mathcal{M}_1^+(\mathcal{X})$$

Fisher kernel

Definition

- Fix a parameter $\theta_0 \in \Theta$ (e.g., by maximum likelihood over a training set of sequences)
- For each sequence $\mathbf{x}$, compute the Fisher score vector:

$$\Phi_{\theta_0}(\mathbf{x}) = \nabla_\theta \log P_\theta(\mathbf{x})|_{\theta=\theta_0} \ .$$

- Form the kernel (Jaakkola et al., 1998):

$$K\left(\mathbf{x}, \mathbf{x}'\right) = \Phi_{\theta_0}(\mathbf{x})^\top I(\theta_0)^{-1} \Phi_{\theta_0}(\mathbf{x}') \ ,$$

where $I(\theta_0) = E_{\theta_0}\left[\Phi_{\theta_0}(\mathbf{x})\Phi_{\theta_0}(\mathbf{x})^\top\right]$ is the Fisher information matrix.

Fisher kernel properties

- The Fisher score describes how each parameter contributes to the process of generating a particular example
- The Fisher kernel is invariant under change of parametrization of the model
- A kernel classifier employing the Fisher kernel derived from a model that contains the label as a latent variable is, asymptotically, at least as good a classifier as the MAP labelling based on the model (under several assumptions).

Fisher kernel in practice

- $\Phi_{\theta_0}(\mathbf{x})$ can be computed explicitly for many models (e.g., HMMs)
- $I(\theta_0)$ is often replaced by the identity matrix
- Several different models (i.e., different θ_0) can be trained and combined
- Feature vectors are explicitly computed

Mutual information kernels

Definition

- Chose a prior $w(d\theta)$ on the measurable set Θ
- Form the kernel (Seeger, 2002):

$$K\left(\mathbf{x},\mathbf{x}'\right)=\int_{\theta\in\Theta}P_\theta(\mathbf{x})P_\theta(\mathbf{x}')w(d\theta)\ .$$

- No explicit computation of a finite-dimensional feature vector
- $K\left(\mathbf{x},\mathbf{x}'\right)=<\phi\left(\mathbf{x}\right),\phi\left(\mathbf{x}'\right)>_{L_2(w)}$ with

$$\phi\left(\mathbf{x}\right)=\left(P_\theta\left(\mathbf{x}\right)\right)_{\theta\in\Theta}\ .$$

Example: coin toss

- Let $P_\theta(X=1) = \theta$ and $P_\theta(X=0) = 1-\theta$ a model for random coin toss, with $\theta \in [0,1]$.
- Let $d\theta$ be the Lebesgue measure on $[0,1]$
- The mutual information kernel between $\mathbf{x} = 001$ and $\mathbf{x}' = 1010$ is:

$$\begin{cases} P_\theta(\mathbf{x}) &= \theta(1-\theta)^2 \ , \\ P_\theta(\mathbf{x}') &= \theta^2(1-\theta)^2 \ , \end{cases}$$

$$K(\mathbf{x},\mathbf{x}') = \int_0^1 \theta^3(1-\theta)^4 \, d\theta = \frac{3!4!}{8!} = \frac{1}{280} \ .$$

Context-tree model

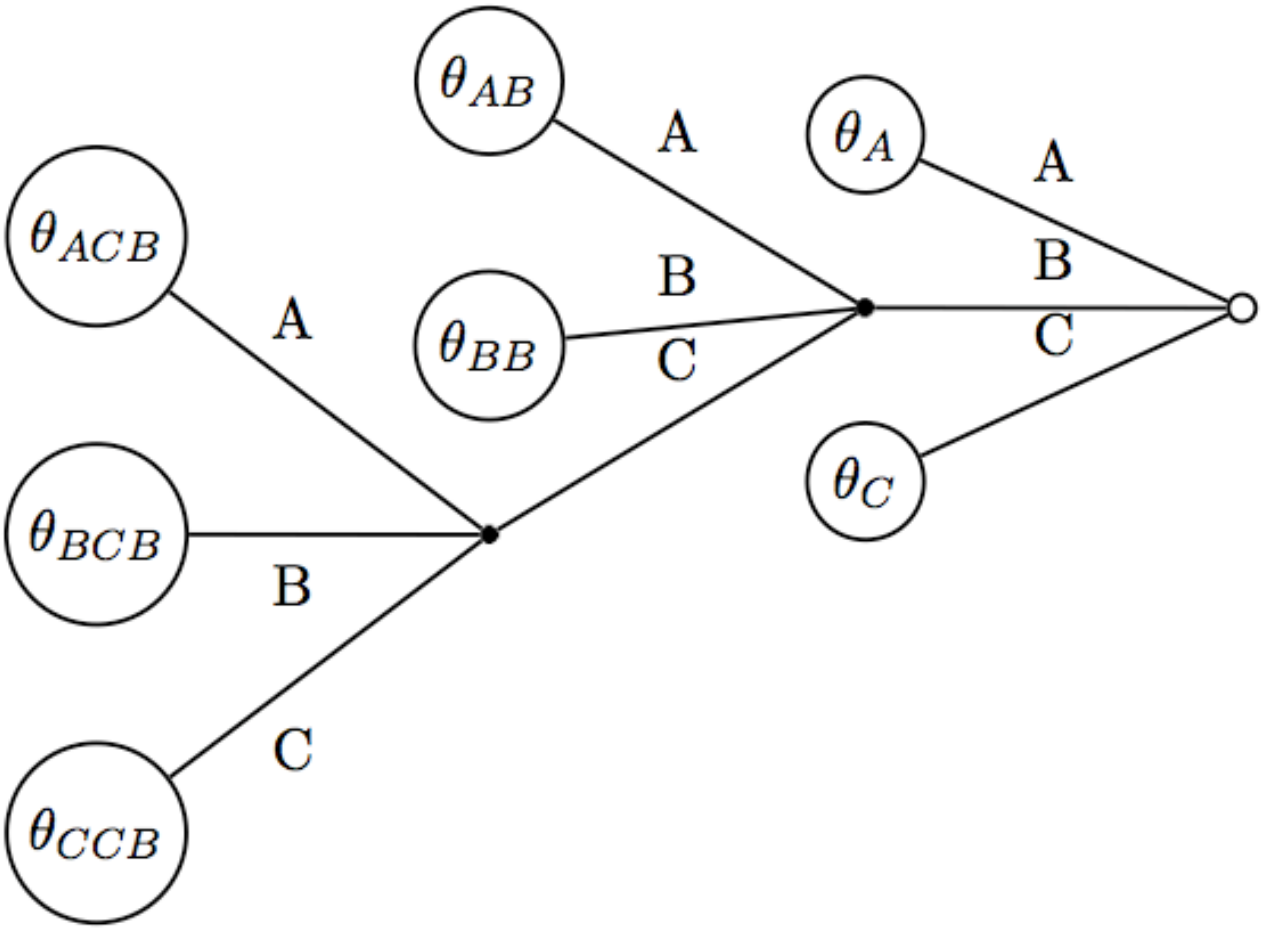
Definition

A context-tree model is a variable-memory Markov chain:

$$P_{\mathcal{D},\theta}(\mathbf{x}) = P_{\mathcal{D},\theta}(x_1 \dots x_D) \prod_{i=D+1}^{n} P_{\mathcal{D},\theta}(x_i \mid x_{i-D} \dots x_{i-1})$$

- $\mathcal{D}$ is a suffix tree
- $\theta \in \Sigma^{\mathcal{D}}$ is a set of conditional probabilities (multinomials)

Context-tree model: example



$$P(AABACBACC) = P(AAB)\theta_{AB}(A)\theta_A(C)\theta_C(B)\theta_{ACB}(A)\theta_A(C)\theta_C(A) \ .$$

The context-tree kernel

Theorem (Cuturi et al., 2004)

- *For particular choices of priors, the context-tree kernel:*

$$K(\mathbf{x}, \mathbf{x}') = \sum_{\mathcal{D}} \int_{\theta \in \Sigma^{\mathcal{D}}} P_{\mathcal{D},\theta}(\mathbf{x}) P_{\mathcal{D},\theta}(\mathbf{x}') w(d\theta|\mathcal{D}) \pi(\mathcal{D})$$

  can be computed in $O(|\mathbf{x}| + |\mathbf{x}'|)$ with a variant of the Context-Tree Weighting algorithm.
- *This is a valid mutual information kernel.*
- *The similarity is related to information-theoretical measure of mutual information between strings.*

Marginalized kernels

Definition

- For any observed data $\mathbf{x} \in \mathcal{X}$, let a latent variable $\mathbf{y} \in \mathcal{Y}$ be associated probabilistically through a conditional probability $P_{\mathbf{x}}(d\mathbf{y})$.
- Let $K_{\mathcal{Z}}$ be a kernel for the complete data $\mathbf{z} = (\mathbf{x}, \mathbf{y})$
- Then the following kernel is a valid kernel on $\mathcal{X}$, called a marginalized kernel (Kin et al., 2002):

$$K_{\mathcal{X}}(\mathbf{x}, \mathbf{x}') := E_{P_{\mathbf{x}}(d\mathbf{y}) \times P_{\mathbf{x}'}(d\mathbf{y}')} K_{\mathcal{Z}}(\mathbf{z}, \mathbf{z}')$$
$$= \int \int K_{\mathcal{Z}}((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) P_{\mathbf{x}}(d\mathbf{y}) P_{\mathbf{x}'}(d\mathbf{y}') \ .$$

Marginalized kernels: proof of positive definiteness

- $K_{\mathcal{Z}}$ is p.d. on $\mathcal{Z}$. Therefore there exists a Hilbert space $\mathcal{H}$ and $\Phi_{\mathcal{Z}} : \mathcal{Z} \to \mathcal{H}$ such that:

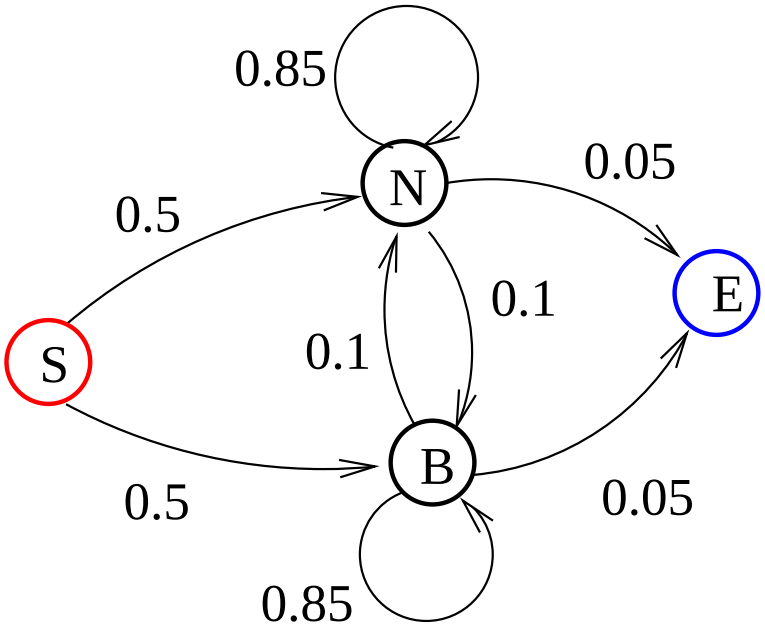
$$K_{\mathcal{Z}}\left(\mathbf{z}, \mathbf{z}'\right) = \left\langle \Phi_{\mathcal{Z}}\left(\mathbf{z}\right), \Phi_{\mathcal{Z}}\left(\mathbf{z}'\right)\right\rangle_{\mathcal{H}} .$$

- Marginalizing therefore gives:

$$\begin{aligned} K_{\mathcal{X}}\left(\mathbf{x}, \mathbf{x}'\right) &= E_{P_{\mathbf{x}}(d\mathbf{y}) \times P_{\mathbf{x}'}(d\mathbf{y}')} K_{\mathcal{Z}}\left(\mathbf{z}, \mathbf{z}'\right) \\ &= E_{P_{\mathbf{x}}(d\mathbf{y}) \times P_{\mathbf{x}'}(d\mathbf{y}')} \left\langle \Phi_{\mathcal{Z}}\left(\mathbf{z}\right), \Phi_{\mathcal{Z}}\left(\mathbf{z}'\right)\right\rangle_{\mathcal{H}} \\ &= \left\langle E_{P_{\mathbf{x}}(d\mathbf{y})} \Phi_{\mathcal{Z}}\left(\mathbf{z}\right), E_{P_{\mathbf{x}}(d\mathbf{y}')} \Phi_{\mathcal{Z}}\left(\mathbf{z}'\right)\right\rangle_{\mathcal{H}} , \end{aligned}$$

therefore $K_{\mathcal{X}}$ is p.d. on $\mathcal{X}$. $\square$

Example: HMM for normal/biased coin toss



- Normal (N) and biased (B) coins (not observed)

- Observed output are 0/1 with probabilities:

$$\begin{cases} \pi(0|N) = 1 - \pi(1|N) = 0.5, \\ \pi(0|B) = 1 - \pi(1|B) = 0.8. \end{cases}$$

- Example of realization (complete data):

NNNNNBBBBBBBBBNNNNNNNNNNNNBBBBBB

100101110111101001011001111011

1-spectrum kernel on complete data

- If both $\mathbf{x} \in \mathcal{A}^*$ and $\mathbf{y} \in \mathcal{S}^*$ were observed, we might rather use the 1-spectrum kernel on the complete data $\mathbf{z} = (\mathbf{x}, \mathbf{y})$:

$$K_{\mathcal{Z}}\left(\mathbf{z}, \mathbf{z}'\right) = \sum_{(a,s) \in \mathcal{A} \times \mathcal{S}} n_{a,s}\left(\mathbf{z}\right) n_{a,s}\left(\mathbf{z}\right),$$

  where $n_{a,s}\left(\mathbf{x}, \mathbf{y}\right)$ for $a = 0, 1$ and $s = N, B$ is the number of occurrences of s in $\mathbf{y}$ which emit a in $\mathbf{x}$.

- Example:

$$\begin{aligned} \mathbf{z} &= 1001011101111010010111001111011, \\ \mathbf{z}' &= 0011010110011111011010111101100101, \end{aligned}$$

$$\begin{aligned} K_{\mathcal{Z}}\left(\mathbf{z}, \mathbf{z}'\right) &= n_0\left(\mathbf{z}\right) n_0\left(\mathbf{z}'\right) + n_0\left(\mathbf{z}\right) n_0\left(\mathbf{z}'\right) + n_1\left(\mathbf{z}\right) n_1\left(\mathbf{z}'\right) + n_1\left(\mathbf{z}\right) n_1\left(\mathbf{z}\right. \\ &= 7 \times 15 + 9 \times 12 + 13 \times 6 + 2 \times 1 = 293. \end{aligned}$$

1-spectrum marginalized kernel on observed data

- The marginalized kernel for observed data is:

$$K_{\mathcal{X}}\left(\mathbf{x},\mathbf{x}'\right) = \sum_{\mathbf{y},\mathbf{y}'\in\mathcal{S}^*} K_{\mathcal{Z}}\left(\left(\mathbf{x},\mathbf{y}\right),\left(\mathbf{x},\mathbf{y}\right)\right) P\left(\mathbf{y}|\mathbf{x}\right) P\left(\mathbf{y}'|\mathbf{x}'\right)$$
$$= \sum_{(a,s)\in\mathcal{A}\times\mathcal{S}} \Phi_{a,s}\left(\mathbf{x}\right)\Phi_{a,s}\left(\mathbf{x}'\right),$$

with

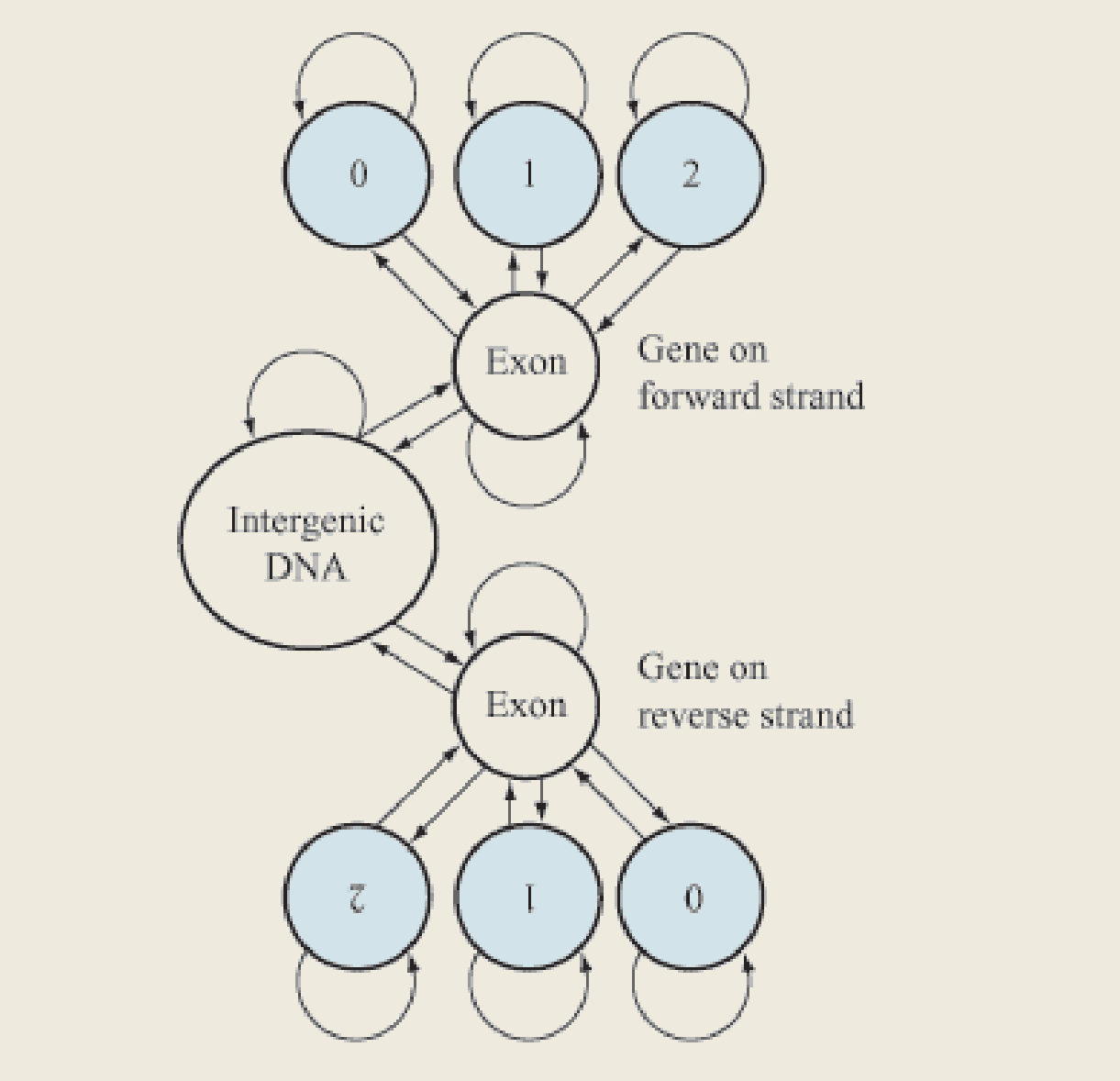
$$\Phi_{a,s}\left(\mathbf{x}\right) = \sum_{\mathbf{y}\in\mathcal{S}^*} P\left(\mathbf{y}|\mathbf{x}\right) n_{a,s}\left(\mathbf{x},\mathbf{y}\right)$$

Computation of the 1-spectrum marginalized kernel

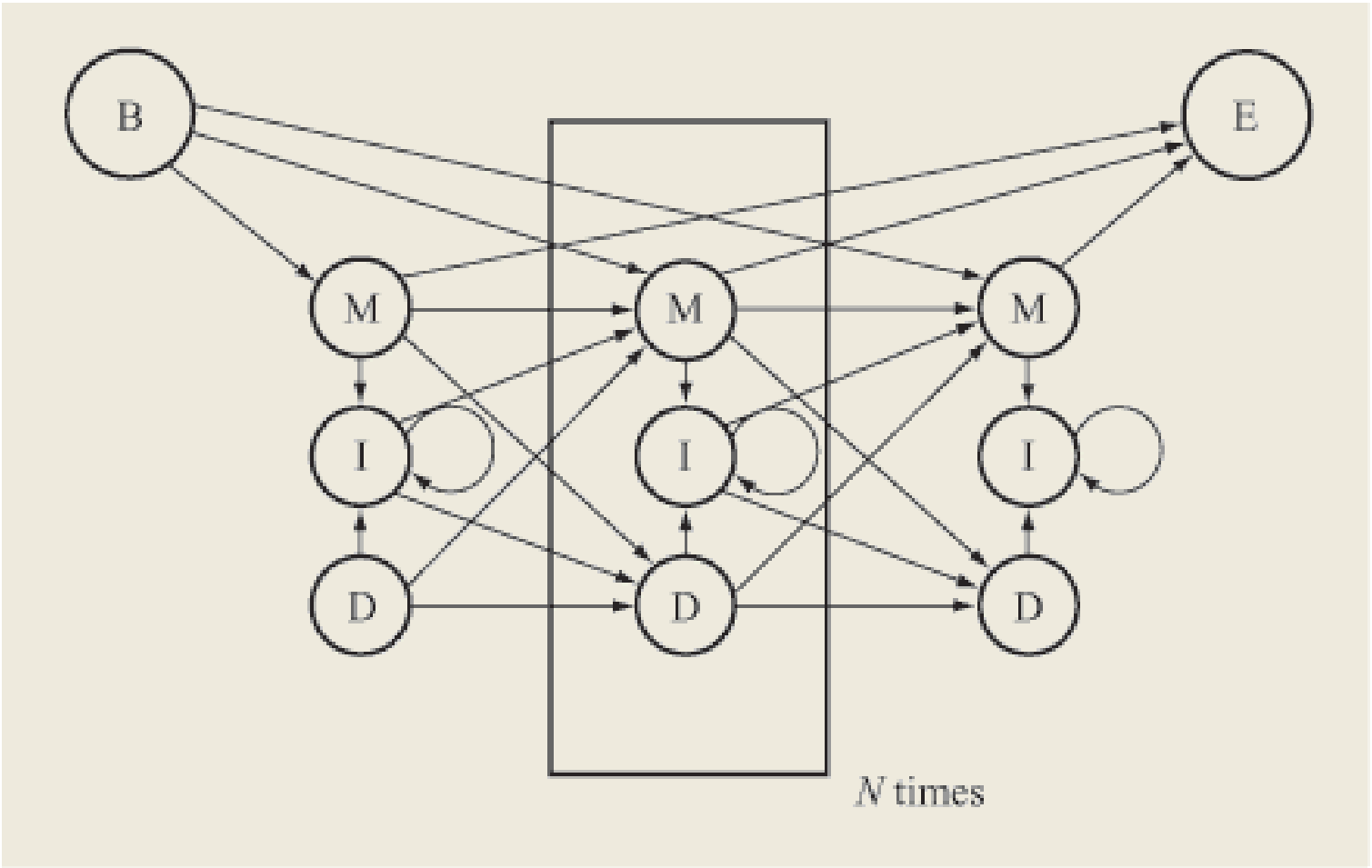
$$
\begin{aligned}
\Phi_{a,s}(\mathbf{x}) &= \sum_{\mathbf{y}\in\mathcal{S}^*} P(\mathbf{y}|\mathbf{x})\, n_{a,s}(\mathbf{x},\mathbf{y}) \\
&= \sum_{\mathbf{y}\in\mathcal{S}^*} P(\mathbf{y}|\mathbf{x}) \left\{ \sum_{i=1}^{n} \delta(x_i, a)\, \delta(y_i, s) \right\} \\
&= \sum_{i=1}^{n} \delta(x_i, a) \left\{ \sum_{\mathbf{y}\in\mathcal{S}^*} P(\mathbf{y}|\mathbf{x})\, \delta(y_i, s) \right\} \\
&= \sum_{i=1}^{n} \delta(x_i, a)\, P(y_i = s|\mathbf{x})\,.
\end{aligned}
$$

and $P(y_i = s|\mathbf{x})$ can be computed efficiently by forward-backward algorithm!

HMM example (DNA)



HMM example (protein)

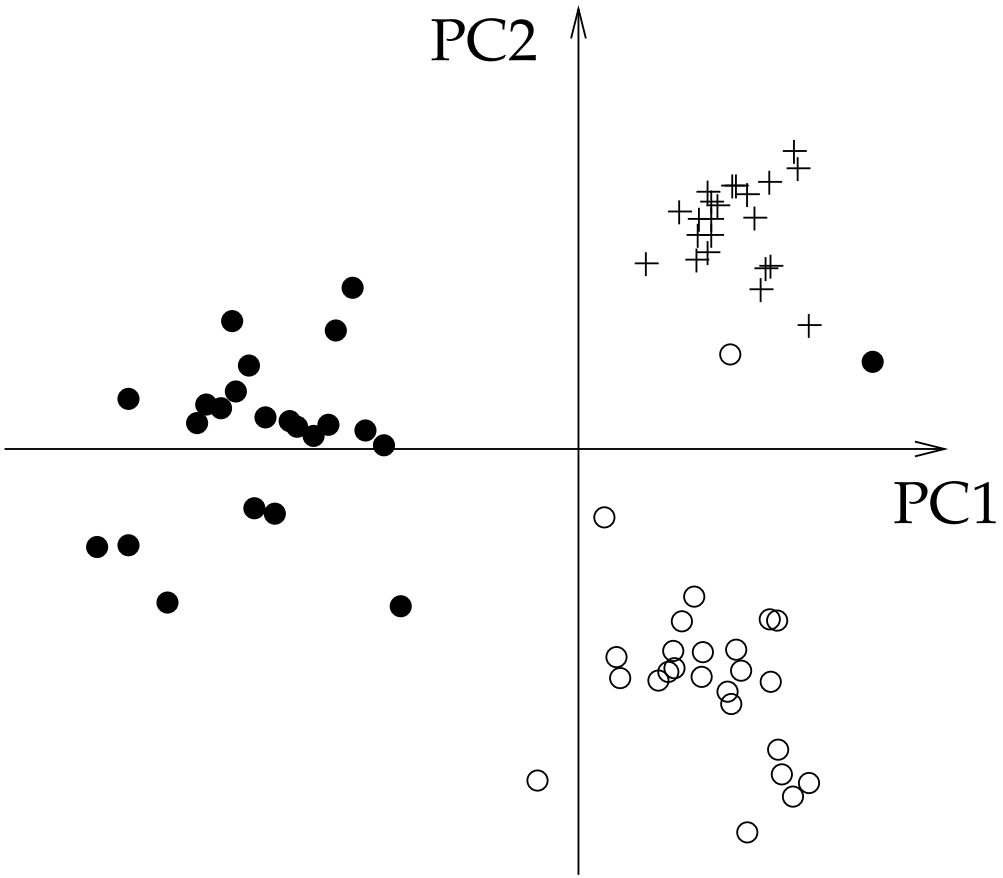


Marginalized kernels in practice

Examples

- Spectrum kernel on the hidden states of a HMM for protein sequences (Tsuda et al., 2002)
- Kernels for RNA sequences based on SCFG (Kin et al., 2002)
- Kernels for graphs based on random walks on graphs (Kashima et al., 2004)
- Kernels for multiple alignments based on phylogenetic models (Vert et al., 2005)

Marginalized kernels: example



A set of 74 human tRNA sequences is analyzed using a kernel for sequences (the second-order marginalized kernel based on SCFG). This set of tRNAs contains three classes, called Ala-AGC (*white circles*), Asn-GTT (*black circles*) and Cys-GCA (*plus symbols*) (from Tsuda et al., 2003).

Outline

Sequence alignment

Motivation

How to compare 2 sequences?

$$\mathbf{x}_1 = \texttt{CGGSLIAMMWFGV}$$
$$\mathbf{x}_2 = \texttt{CLIVMMNRLMWFGV}$$

Find a good alignment:

```
CGGSLIAMM----WFGV
|...|||||....||||
C---LIVMMNRLMWFGV
```

Alignment score

In order to quantify the relevance of an alignment π, define:

- a substitution matrix $S \in \mathbb{R}^{\mathcal{A} \times \mathcal{A}}$
- a gap penalty function $g : \mathbb{N} \longrightarrow \mathbb{R}$

Any alignment is then scored as follows

```
CGGSLIAMM----WFGV
|...|||||....||||
C---LIVMMNRLMWFGV
```

$$s_{S,g}(\pi) = S(C,C) + S(L,L) + S(I,I) + S(A,V) + 2S(M,M) + S(W,W) + S(F,F) + S(G,G) + S(V,V) - g(3) - g(4)$$

Local alignment kernel

Smith-Waterman score

- The widely-used Smith-Waterman local alignment score is defined by:

$$SW_{S,g}(\mathbf{x},\mathbf{y}) := \max_{\pi \in \Pi(\mathbf{x},\mathbf{y})} s_{S,g}(\pi).$$

- It is symmetric, but not positive definite...

LA kernel

The local alignment kernel:

$$K_{LA}^{(\beta)}(\mathbf{x},\mathbf{y}) = \sum_{\pi \in \Pi(\mathbf{x},\mathbf{y})} \exp\left(\beta s_{S,g}(\mathbf{x},\mathbf{y},\pi)\right),$$

is symmetric positive definite (Vert et al., 2004).

Local alignment kernel

Smith-Waterman score

- The widely-used Smith-Waterman local alignment score is defined by:

$$SW_{S,g}(\mathbf{x},\mathbf{y}) := \max_{\pi \in \Pi(\mathbf{x},\mathbf{y})} s_{S,g}(\pi).$$

- It is symmetric, but not positive definite...

LA kernel

The local alignment kernel:

$$K_{LA}^{(\beta)}(\mathbf{x},\mathbf{y}) = \sum_{\pi \in \Pi(\mathbf{x},\mathbf{y})} \exp\left(\beta s_{S,g}(\mathbf{x},\mathbf{y},\pi)\right),$$

is symmetric positive definite (Vert et al., 2004).

LA kernel is p.d.: proof

- If K_1 and K_2 are p.d. kernels for strings, then their convolution defined by:

$$K_1 \star K_2(\mathbf{x}, \mathbf{y}) := \sum_{\mathbf{x}_1\mathbf{x}_2=\mathbf{x}, \mathbf{y}_1\mathbf{y}_2=\mathbf{y}} K_1(\mathbf{x}_1, \mathbf{y}_1) K_2(\mathbf{x}_2, \mathbf{y}_2)$$

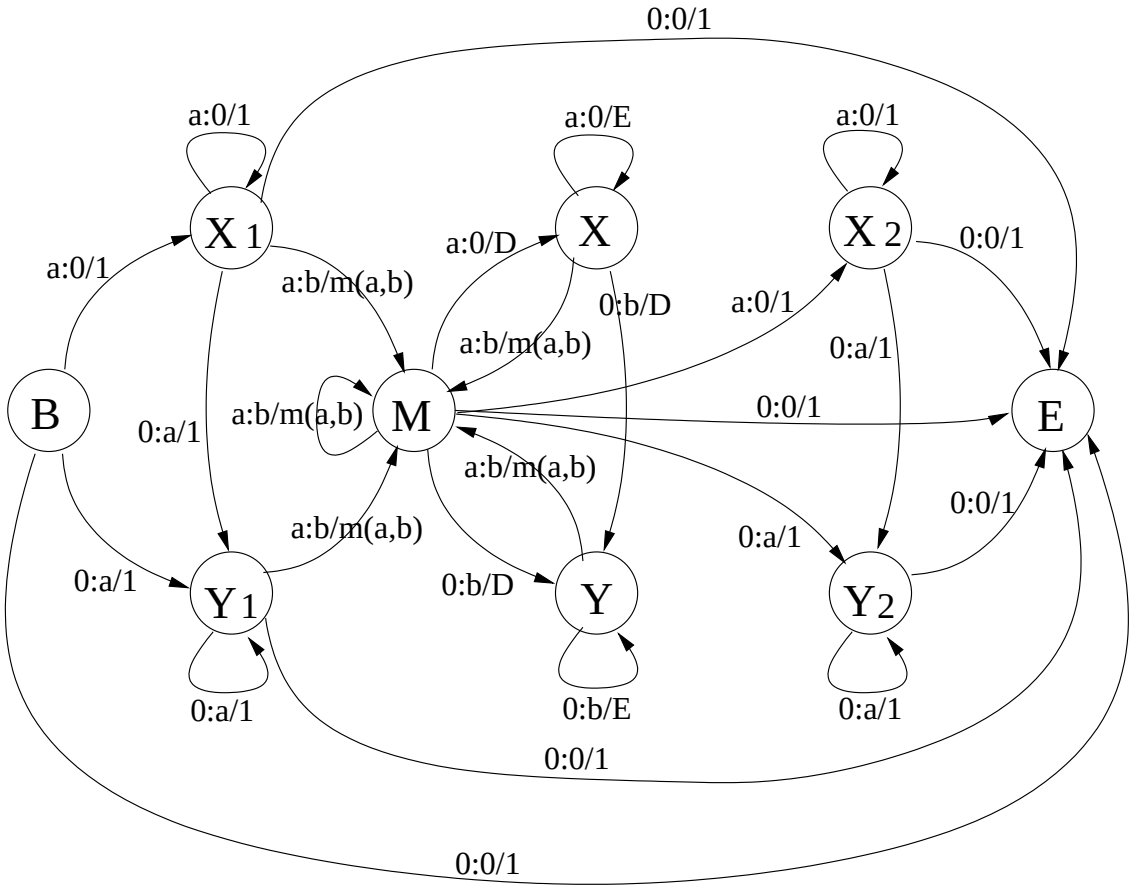
  is also p.d. (Haussler, 1999).
- LA kernel is p.d. because it is a convolution kernel (Haussler, 1999):

$$K_{LA}^{(\beta)} = \sum_{n=0}^{\infty} K_0 \star \left(K_a^{(\beta)} \star K_g^{(\beta)} \right)^{(n-1)} \star K_a^{(\beta)} \star K_0.$$

  where K_0, K_a and K_g are three basic p.d. kernels (Vert et al., 2004).

LA kernel in practice

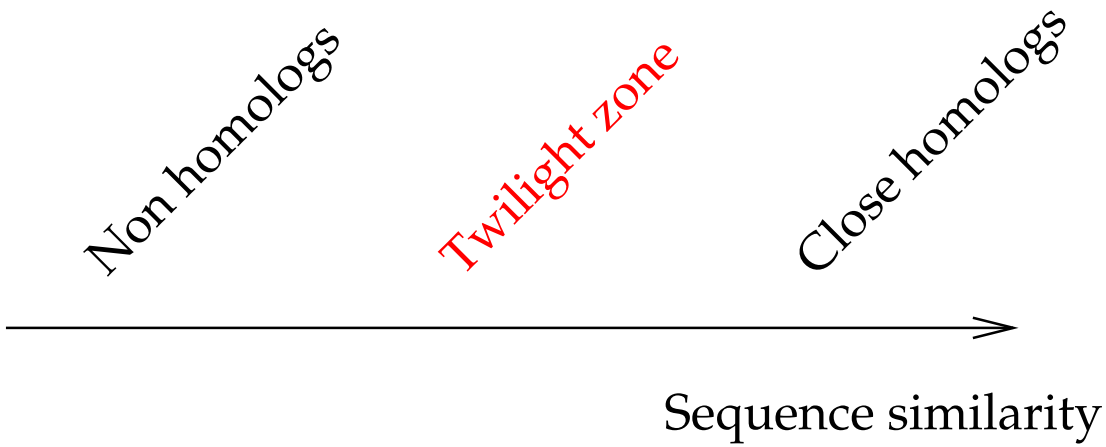
- Implementation by dynamic programming in $O(|\mathbf{x}| \times |\mathbf{x}'|)$



- In practice, values are too large (exponential scale) so taking its logarithm is a safer choice (but not p.d. anymore!)

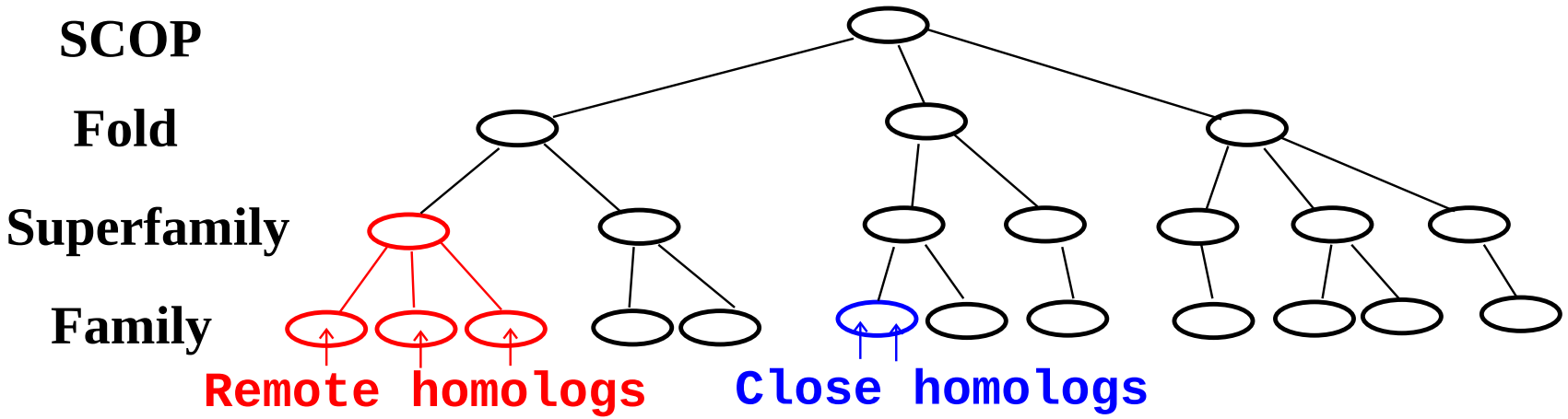
Outline

Remote homology



- Homologs have common ancestors
- Structures and functions are more conserved than sequences
- Remote homologs can not be detected by direct sequence comparison

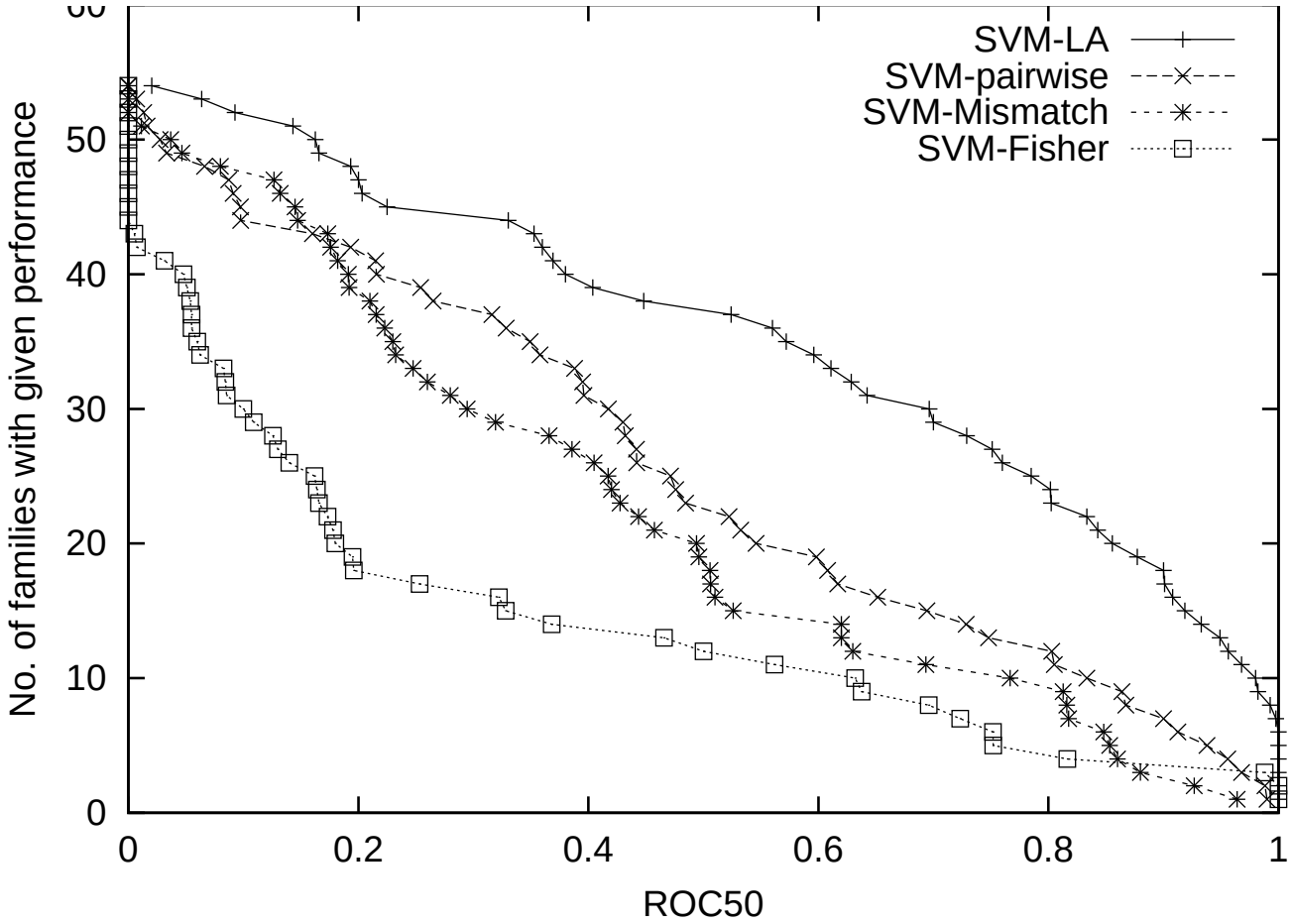
SCOP database



A benchmark experiment

- Goal: recognize directly the superfamily
- Training: for a sequence of interest, positive examples come from the same superfamily, but different families. Negative from other superfamilies.
- Test: predict the superfamily.

Difference in performance



Performance on the SCOP superfamily recognition benchmark (from Vert et al., 2004).

Summary

- A variety of principles for string kernel design have been proposed.
- Good kernel design is important for each data and each task. Performance is not the only criterion.
- Still an art, although principled ways have started to emerge.
- Their application goes beyond computational biology.

Part 4

Kernels on graphs

Outline

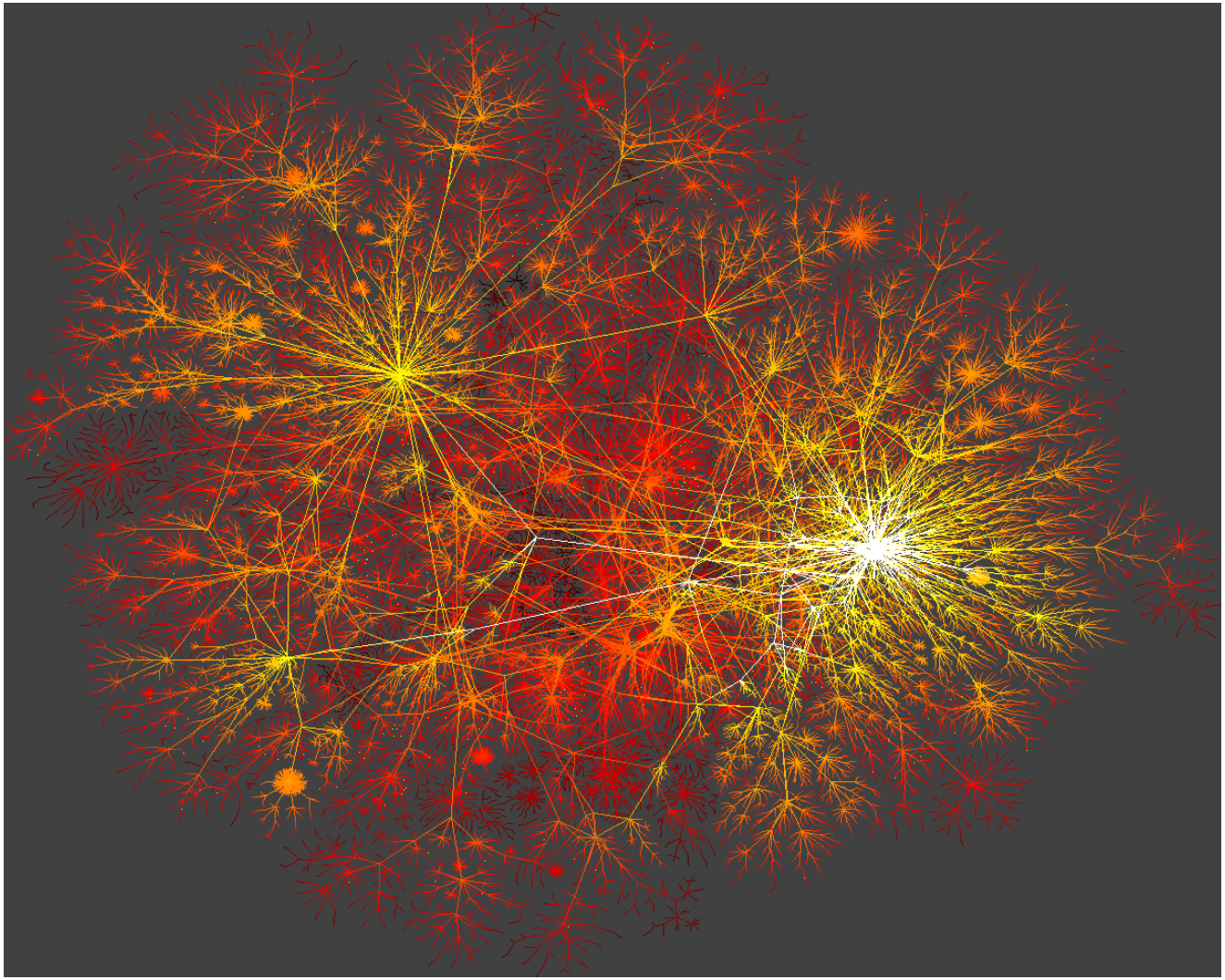
Graphs

Motivation

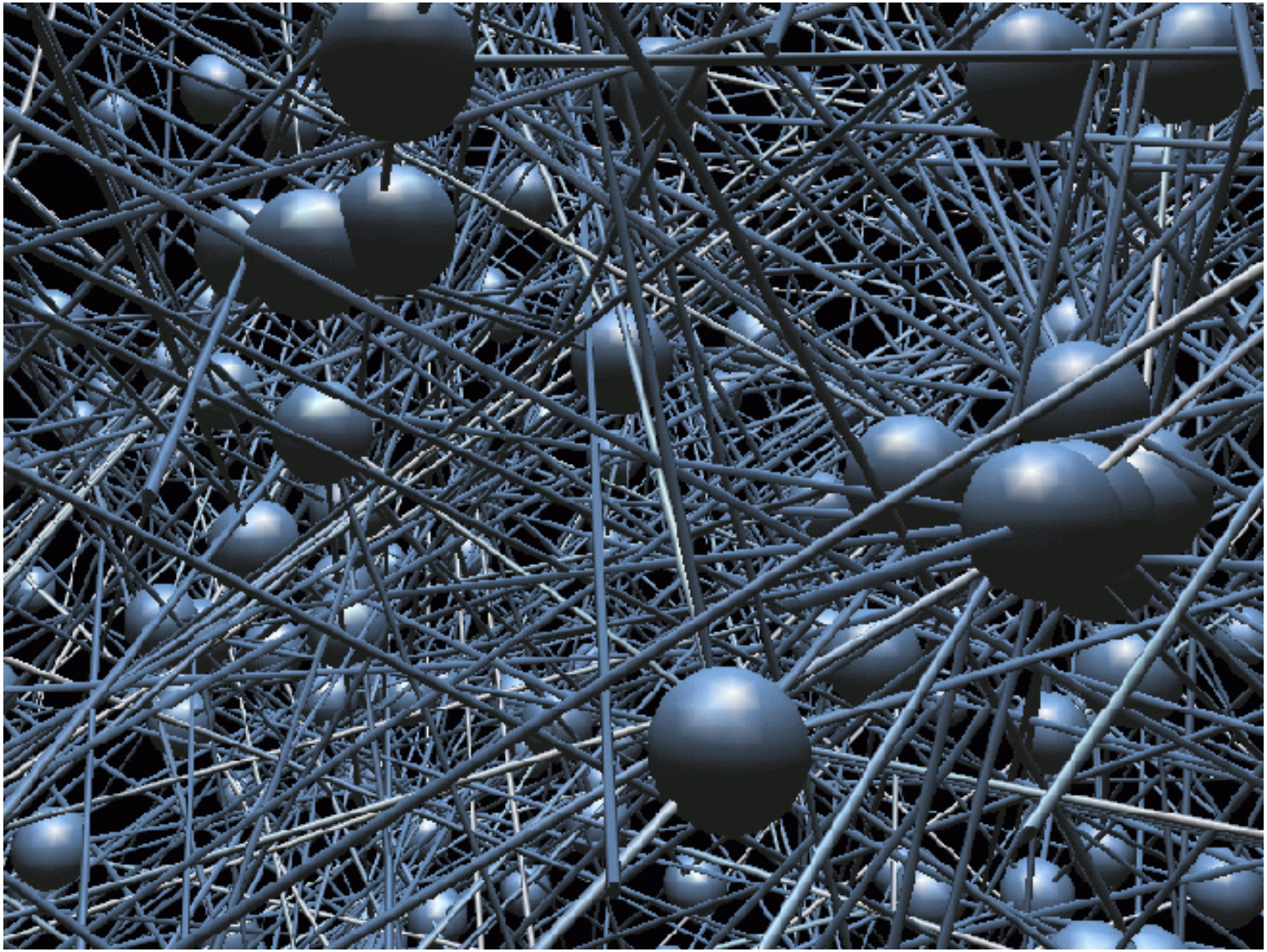
Many data come in the form of nodes in a graph for different reasons:

- by definition (interaction network, internet...)
- by discretization / sampling of a continuous domain
- by convenience (e.g., if only a similarity function if available)

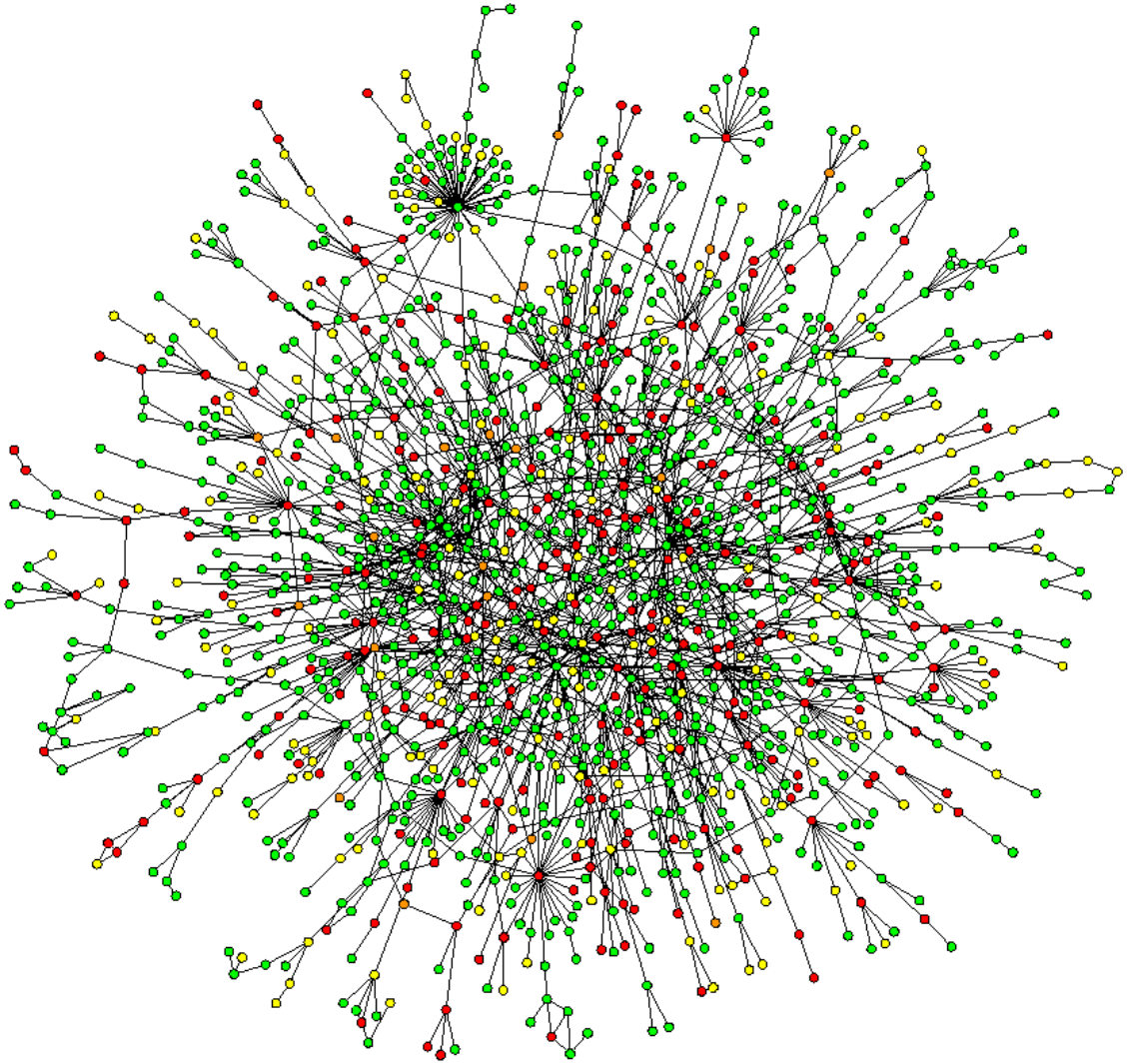
Example: web



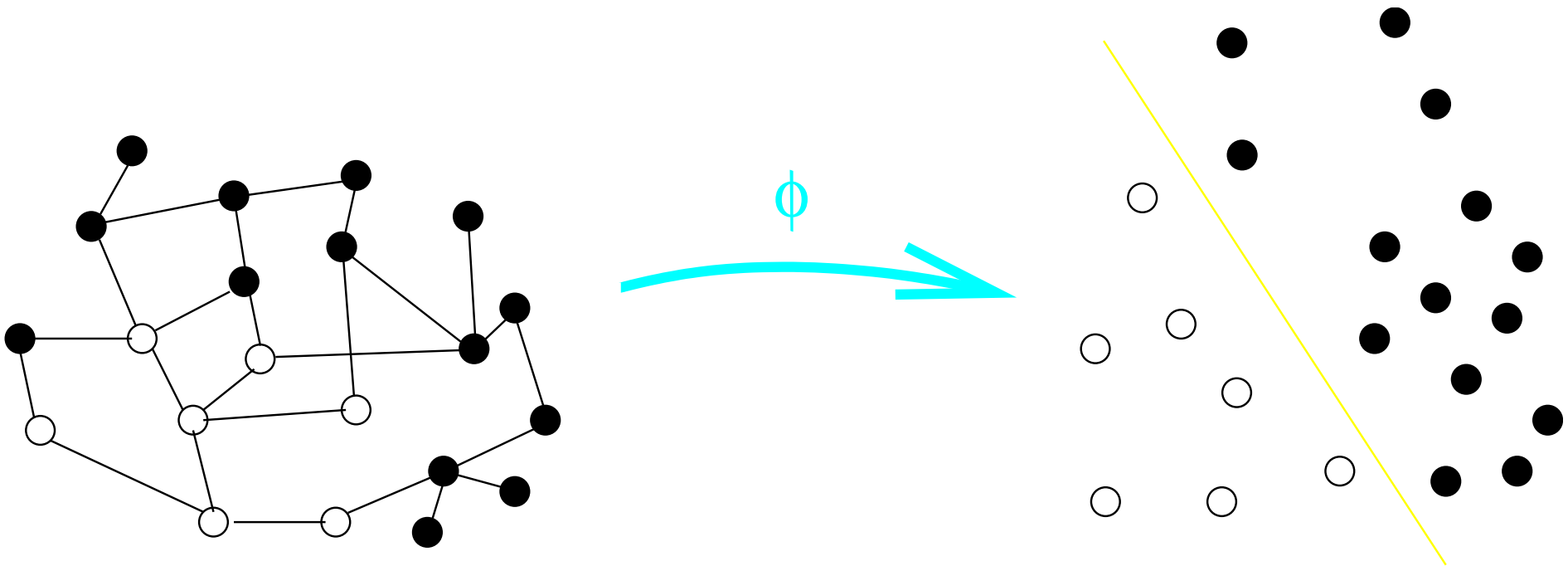
Example: social network



Example: protein-protein interaction



Kernel on a graph



- We need a kernel $K(\mathbf{x}, \mathbf{x}')$ between nodes of the graph.
- Example: predict gene protein functions from high-throughput protein-protein interaction data.

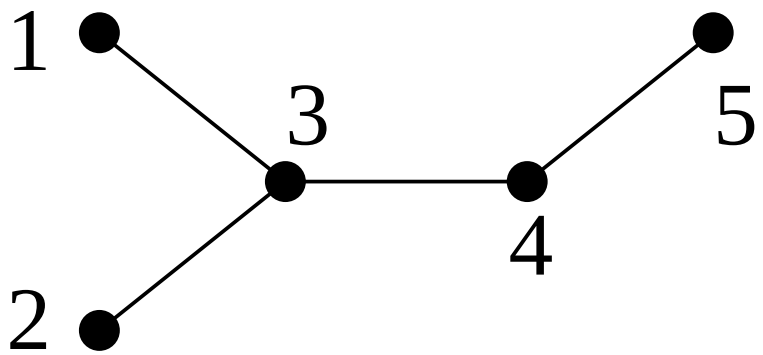
Notations

- $\mathcal{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ is finite.
- For $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, we note $\mathbf{x} \sim \mathbf{x}'$ to indicate the existence of an edge between $\mathbf{x}$ and $\mathbf{x}'$
- We assume that there is no self-loop $\mathbf{x} \sim \mathbf{x}$, and that there is a single connected component.
- The adjacency matrix is $A \in \mathbb{R}^{m \times m}$:

$$A_{i,j} = \begin{cases} 1 & \text{if } i \sim j, \\ 0 & \text{otherwise.} \end{cases}$$

- D is the diagonal matrix where $D_{i,i}$ is the number of neighbors of $\mathbf{x}_i$ ($D_{i,i} = \sum_{i=1}^{m} A_{i,j}$).

Example



$$A = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \qquad D = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

General remarks

- $\mathcal{X}$ being finite, any symmetric semi-definite matrix K defines a valid p.d. kernel on $\mathcal{X}$.
- How to "translate" the graph topology into the kernel?
  - Direct geometric approach: $K_{i,j}$ should be "large" when $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close" to each other on the graph?
  - Functional approach: $\| f \|_K$ should be "small" when f is "smooth" on the graph?
  - Link discrete/continuous: is there an equivalent to the continuous Gaussien kernel on the graph (e.g., limit by fine discretization)?

General remarks

- $\mathcal{X}$ being finite, any symmetric semi-definite matrix K defines a valid p.d. kernel on $\mathcal{X}$.
- How to "translate" the graph topology into the kernel?
  - Direct geometric approach: $K_{i,j}$ should be "large" when $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close" to each other on the graph?
  - Functional approach: $\| f \|_K$ should be "small" when f is "smooth" on the graph?
  - Link discrete/continuous: is there an equivalent to the continuous Gaussien kernel on the graph (e.g., limit by fine discretization)?

Geometric approach

- Remember : for $\mathcal{X} = \mathbb{R}^n$, the Gaussian RBF kernel is:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-d(\mathbf{x}, \mathbf{x}')^2 / 2\sigma^2\right),$$

  where $d(\mathbf{x}, \mathbf{x}')$ is the Euclidean distance.
- If $\mathcal{X}$ is a graph, let $d(\mathbf{x}, \mathbf{x}')$ be the shortest-path distance between $\mathbf{x}$ and $\mathbf{x}'$.
- Problem: $\exp\left(-d(\mathbf{x}, \mathbf{x}')^2 / 2\sigma^2\right)$ is not d.p. in general.
- Big problem: no simple criterion (to my knowledge) to check when $K(\mathbf{x}, \mathbf{x}') = \phi(d(\mathbf{x}, \mathbf{x}'))$ is p.d. or not...

Geometric approach

- Remember : for $\mathcal{X} = \mathbb{R}^n$, the Gaussian RBF kernel is:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-d(\mathbf{x}, \mathbf{x}')^2 / 2\sigma^2\right),$$

  where $d(\mathbf{x}, \mathbf{x}')$ is the Euclidean distance.
- If $\mathcal{X}$ is a graph, let $d(\mathbf{x}, \mathbf{x}')$ be the shortest-path distance between $\mathbf{x}$ and $\mathbf{x}'$.
- Problem: $\exp\left(-d(\mathbf{x}, \mathbf{x}')^2 / 2\sigma^2\right)$ is not d.p. in general.
- Big problem: no simple criterion (to my knowledge) to check when $K(\mathbf{x}, \mathbf{x}') = \phi(d(\mathbf{x}, \mathbf{x}'))$ is p.d. or not...

Outline

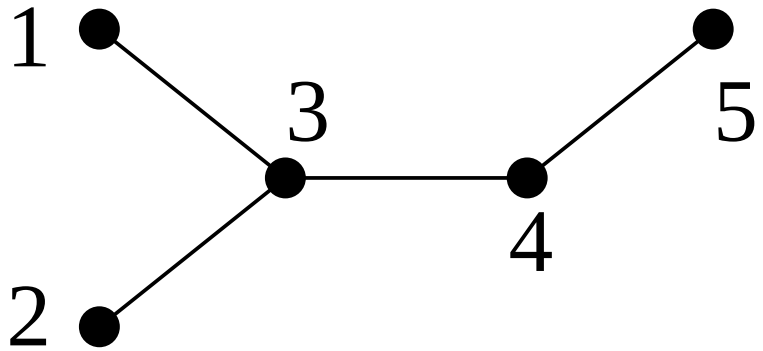
Motivation

- In this section we define a priori a smoothness functional on the functions $f : \mathcal{X} \to \mathbb{R}$.
- We then show that it defines a RKHS and identify the corresponding kernel
- As preliminaries we need to introduce the Laplacian of the graph.

Graph Laplacian

Definition

The Laplacian of the graph is the matrix $L = A - D$.



$$L = A - D = \begin{pmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 1 & 1 & -3 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix}$$

Properties of the Laplacian

Lemma

Let $L = A - D$ be the Laplacian of the graph:

- *For any $f : \mathcal{X} \to \mathbb{R}$,*

$$\Omega(f) := \sum_{i \sim j} \left(f\left(\mathbf{x}_i\right) - f\left(\mathbf{x}_j\right)\right)^2 = -f^\top L f$$

- *$-L$ is a symmetric positive semi-definite matrix*
- *0 is an eigenvalue with multiplicity 1 associated to the constant eigenvector $\mathbf{1} = (1, \ldots, 1)$*
- *The image of L is*

$$Im(L) = \left\{ f \in \mathbb{R}^m : \sum_{i=1}^m f_i = 0 \right\}$$

Proof: link between $\Omega(f)$ and L

$$
\begin{aligned}
\Omega(f) &= \sum_{i \sim j} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2 \\
&= \sum_{i \sim j} \left(f(\mathbf{x}_i)^2 + f(\mathbf{x}_j)^2 - 2 f(\mathbf{x}_i) f(\mathbf{x}_j)\right) \\
&= \sum_{i=1}^{m} D_{i,i} f(\mathbf{x}_i)^2 - 2 \sum_{i \sim j} f(\mathbf{x}_i) f(\mathbf{x}_j) \\
&= f^\top D f - f^\top A f \\
&= -f^\top L f
\end{aligned}
$$

Proof: eigenstructure of L

- L is symmetric because A and D are symmetric.
- For any $f \in \mathbb{R}^m$, $-f^\top L f = \Omega(f) \geq 0$, therefore the (real-valued) eigenvalues of $-L$ are ≥ 0 : $-L$ is therefore positive semi-definite.
- f is an eigenvector associated to eigenvalue 0
  iff $f^\top L f = 0$
  iff $\sum_{i \sim j} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2 = 0$,
  iff $f(\mathbf{x}_i) = f(\mathbf{x}_j)$ when $i \sim j$,
  iff f is constant (because the graph is connected).
- L being symmetric, $Im(L)$ is the orthogonal supplement of $Ker(L)$, that is, the set of functions orthogonal to $\mathbf{1}$. $\square$

Our first graph kernel

We are now ready to present a RKHS on the vertices of the graph and its associated kernel:

Theorem

The set $\mathcal{H} = \left\{ f \in \mathbb{R}^m : \sum_{i=1}^m f_i = 0 \right\}$ *endowed with the norm:*

$$\Omega(f) = \sum_{i \sim j} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j) \right)^2$$

is a RKHS whose reproducing kernel is $(-L)^*$*, the pseudo-inverse of the graph Laplacian.*

Pseudo-inverse of $-L$

Remember the pseudo-inverse $(-L)^*$ of $-L$ is the linear application that is equal to:

- 0 on $Ker(-L)$
- $(-L)^{-1}$ on $Im(-L)$, that is, if we write:

$$-L = \sum_{i=1}^{m} \lambda_i u_i u_i^\top$$

the eigendecomposition of $-L$:

$$(-L)^* = \sum_{\lambda_i \neq 0} (\lambda_i)^{-1} u_i u_i^\top .$$

- In particular it holds that $(-L)^*(-L) = (-L)(-L)^* = \Pi_{\mathcal{H}}$, the projection onto $Im(-L) = \mathcal{H}$.

Proof of Theorem 7

- Resticted to $\mathcal{H}$, the symmetric bilinear form:

$$\langle f, g \rangle = -f^\top L g$$

  is positive definite (because $-L$ is positive semi-definite, and $\mathcal{H} = Im(-L)$). It is therefore a scalar product, making of $\mathcal{H}$ a Hilbert space (in fact Euclidean).
- The norm in this Hilbert space $\mathcal{H}$ is:

$$\| f \|^2 = \langle f, f \rangle = -f^\top L f = \Omega(f) \ .$$

Proof of Theorem 7 (cont.)

To check that $\mathcal{H}$ is a RKHS with reproducing kernel $K = (-L)^*$, it suffices to show that:

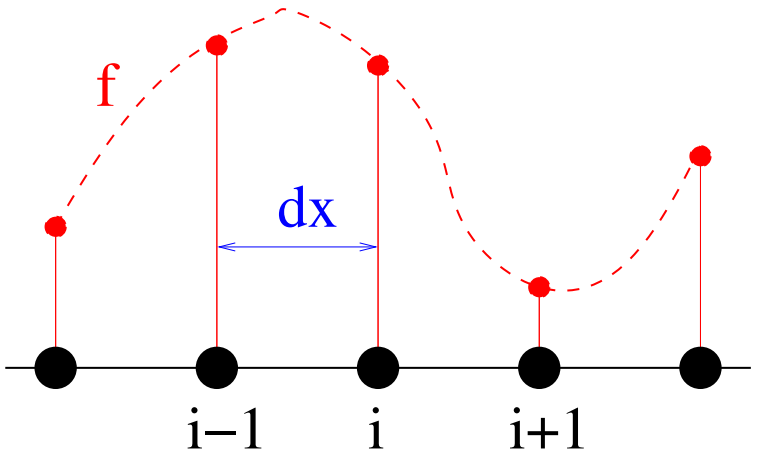
$$\begin{cases} \forall \mathbf{x} \in \mathcal{X}, & K_{\mathbf{x}} \in \mathcal{H} \ , \\ \forall \, (\mathbf{x}, f) \in \mathcal{X} \times \mathcal{H}, & \langle f, K_{\mathbf{x}} \rangle = f(\mathbf{x}) \ . \end{cases}$$

- $Ker(K) = Ker\left((-L)^*\right) = Ker\,(L)$, implying $K\mathbf{1} = 0$. Therefore, each row/column of K is in $\mathcal{H}$.
- Finally, for any $f \in \mathcal{H}$, if we denote by $g_i = \langle K(i, \cdot), f \rangle$ we get:

$$g = -KLf = -(-L)^* Lf = \Pi_{\mathcal{H}}(f) = f \ .$$

As a conclusion $K = (-L)^*$ is the reproducing kernel of $\mathcal{H}$. □

Interpretation of the Laplacian



$$
\begin{aligned}
\Delta f(x) &= f''(x) \\
&\sim \frac{f'(x + dx/2) - f'(x - dx/2)}{dx} \\
&\sim \frac{f(x + dx) - f(x) - f(x) + f(x - dx)}{dx^2} \\
&= \frac{f_{i-1} + f_{i+1} - 2f(x)}{dx^2} \\
&= \frac{Lf(i)}{dx^2}.
\end{aligned}
$$

Interpretation of regularization

For $f = [0,1] \to \mathbb{R}$ and $x_i = i/m$, we have:

$$\begin{aligned}
\Omega(f) &= \sum_{i=1}^{m} \left(f\left(\frac{i+1}{m}\right) - f\left(\frac{i}{m}\right) \right)^2 \\
&\sim \sum_{i=1}^{m} \left(\frac{1}{m} \times f'\left(\frac{i}{m}\right) \right)^2 \\
&= \frac{1}{m} \times \frac{1}{m} \sum_{i=1}^{m} f'\left(\frac{i}{m}\right)^2 \\
&\sim \frac{1}{m} \int_0^1 f'(t)^2 dt.
\end{aligned}$$

Outline

Motivation

- Consider the normalized Gaussian kernel on $\mathbb{R}^d$:

$$K_t(\mathbf{x}, \mathbf{x}') = \frac{1}{(4\pi t)^{\frac{d}{2}}} \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}' \|^2}{4t}\right).$$

- In order to transpose it to the graph, replacing the Euclidean distant by the shortest-path distance does not work.
- In this section we provide a characterization of the Gaussian kernel as the solution of a partial differential equation involving the Laplacian, which we can transpose to the graph: the diffusion equation.
- The solution of the discrete diffusion equation will be called the diffusion kernel or heat kernel.

The diffusion equation

Lemma

For any $\mathbf{x}_0 \in \mathbb{R}^d$*, the function:*

$$K_{\mathbf{x}_0}(\mathbf{x}, t) = K_t(\mathbf{x}_0, \mathbf{x}) = \frac{1}{(4\pi t)^{\frac{d}{2}}} \exp\left(-\frac{\| \mathbf{x} - \mathbf{x}_0 \|^2}{4t}\right).$$

is solution of the diffusion equation:

$$\frac{\partial}{\partial t} K_{\mathbf{x}_0}(\mathbf{x}, t) = \Delta K_{\mathbf{x}_0}(\mathbf{x}, t).$$

with initial condition $K_{\mathbf{x}_0}(\mathbf{x}, 0) = \delta_{\mathbf{x}_0}(\mathbf{x})$

(proof = direct computation).

Discrete diffusion equation

For finite-dimensional $f_t \in \mathbb{R}^m$, the diffusion equation becomes:

$$\frac{\partial}{\partial t} f_t = L f_t$$

which admits the following solution:

$$f_t = f_0 e^{tL}$$

with

$$e^{tL} = I + tL + \frac{t^2}{2!} L^2 + \frac{t^3}{3!} L^3 + \dots$$

Diffusion kernel (Kondor and Lafferty, 2002)

This suggest to consider:

$$K = e^{tL}$$

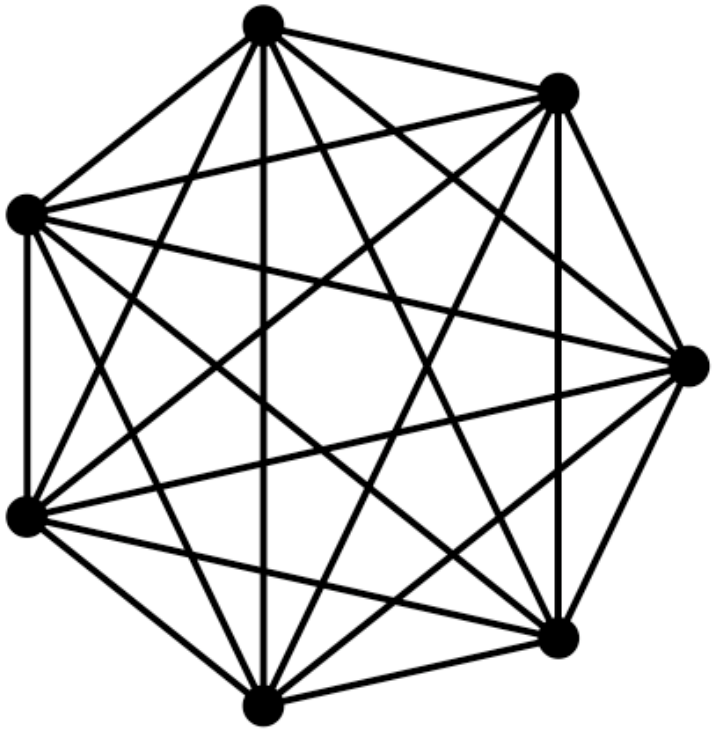
which is indeed symmetric positive semi-definite because if we write:

$$L = \sum_{i=1}^{m} (-\lambda_i) u_i u_i^\top \quad (\lambda_i \geq 0)$$

we obtain:

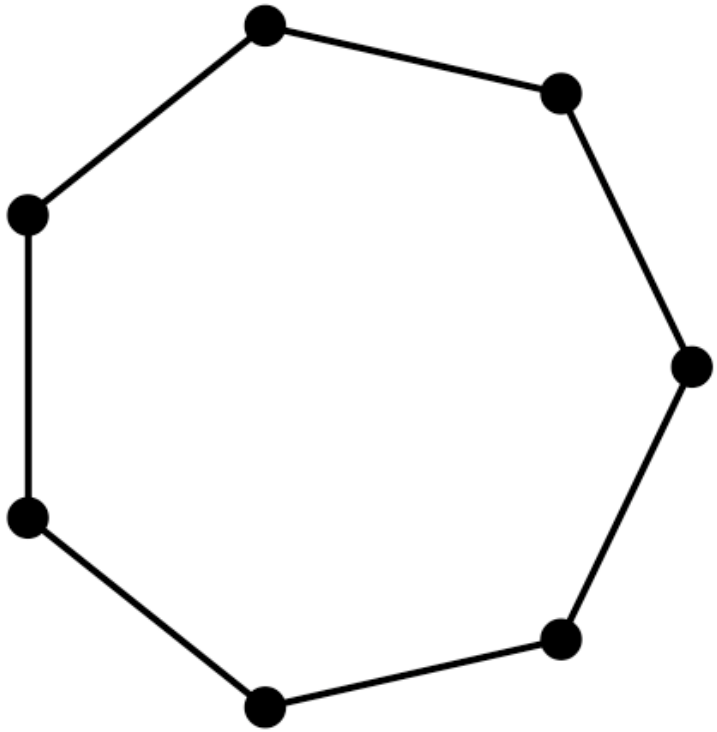
$$K = e^{tL} = \sum_{i=1}^{m} e^{-t\lambda_i} u_i u_i^\top$$

Example: complete graph



$$K_{i,j} = \begin{cases} \frac{1+(m-1)e^{-tm}}{m} & \text{for } i = j, \\ \frac{1-e^{-tm}}{m} & \text{for } i \neq j. \end{cases}$$

Example: closed chain



$$K_{i,j} = \frac{1}{m} \sum_{\nu=0}^{m-1} \exp\left[-2t\left(1 - \cos\frac{2\pi\nu}{m}\right)\right] \cos\frac{2\pi\nu(i-j)}{m}.$$

Outline

Motivation

- In this section we show that the diffusion and Laplace kernels can be interpreted in the frequency domain of functions
- This shows that our strategy to design kernels on graphs was based on (discrete) harmonic analysis on the graph
- In fact this powerful approach can be extended to many structures where harmonic analysis exist: graphs, differentiable manifolds, groups and semi-groups... but this is certainly beyond this tutorial!

Spectrum of the diffusion kernel

- Let $0 = \lambda_1 > -\lambda_2 \geq \ldots \geq -\lambda_m$ be the eigenvalues of the Laplacian:

$$L = \sum_{i=1}^{m} (-\lambda_i) u_i u_i^\top \quad (\lambda_i \geq 0)$$

- The diffusion kernel K_t is an invertible matrix because its eigenvalues are strictly positive:

$$K_t = \sum_{i=1}^{m} e^{-t\lambda_i} u_i u_i^\top$$

Norm in the diffusion RKHS

- Any function $f \in \mathbb{R}^m$ can be written as $f = K\left(K^{-1}f\right)$, therefore its norm in the diffusion RKHS is:

$$\| f \|_{K_t}^2 = \left(f^\top K^{-1}\right) K \left(K^{-1} f\right) = f^\top K^{-1} f$$

- For $i = 1, \ldots, m$, let:

$$\hat{f}_i = u_i^\top f$$

be the projection of f onto the eigenbasis of K.

- We then have:

$$\| f \|_{K_t}^2 = f^\top K^{-1} f = \sum_{i=1}^m e^{t\lambda_i} \hat{f}_i^2.$$

- This looks similar to $\int \left| \hat{f}(\omega) \right|^2 e^{\sigma^2 \omega^2} d\omega$ ...

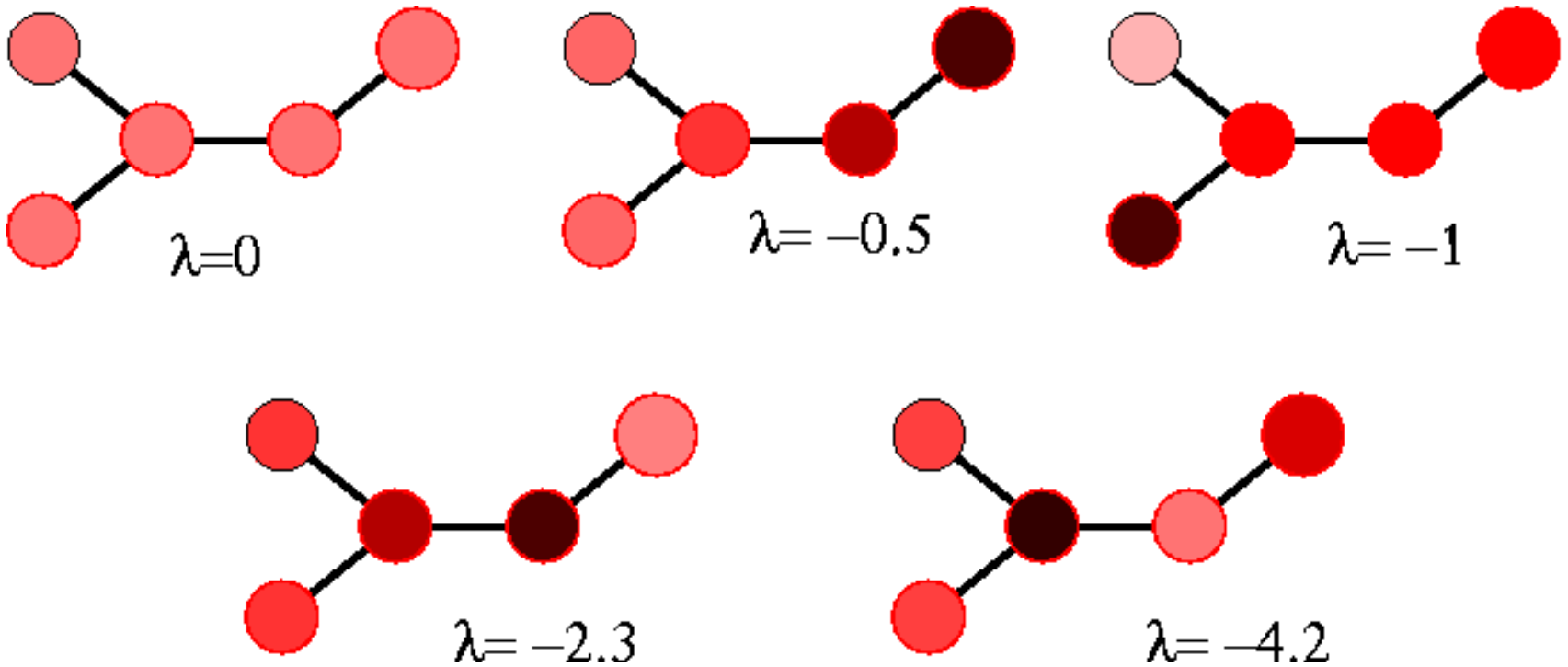
Discrete Fourier transform

Definition

The vector $\hat{f} = \left(\hat{f}_1, \ldots, \hat{f}_m\right)^\top$ is called the discrete Fourier transform of $f \in \mathbb{R}^n$

- The eigenvectors of the Laplacian are the discrete equivalent to the sine/cosine Fourier basis on $\mathbb{R}^n$.
- The eigenvalues λ_i are the equivalent to the frequencies $(i\omega)^2$
- Successive eigenvectors "oscillate" increasingly as eigenvalues get more and more negative.

Example: eigenvectors of the Laplacian



Generalization

This observation suggests to define a whole family of kernels:

$$K_r = \sum_{i=1}^{m} r(\lambda_i) u_i u_i^\top$$

associated with the following RKHS norms:

$$\| f \|_{K_r}^2 = \sum_{i=1}^{m} \frac{\hat{f}_i^2}{r(\lambda_i)}$$

where $r : \mathbb{R}^+ \to \mathbb{R}_*^+$ is a non-increasing function.

Example : regularized Laplacian

$$r(\lambda) = \frac{1}{\lambda + \epsilon}, \qquad \epsilon > 0$$

$$K = \sum_{i=1}^{m} \frac{1}{\lambda_i + \epsilon} u_i u_i^\top = (-L + \epsilon I)^{-1}$$

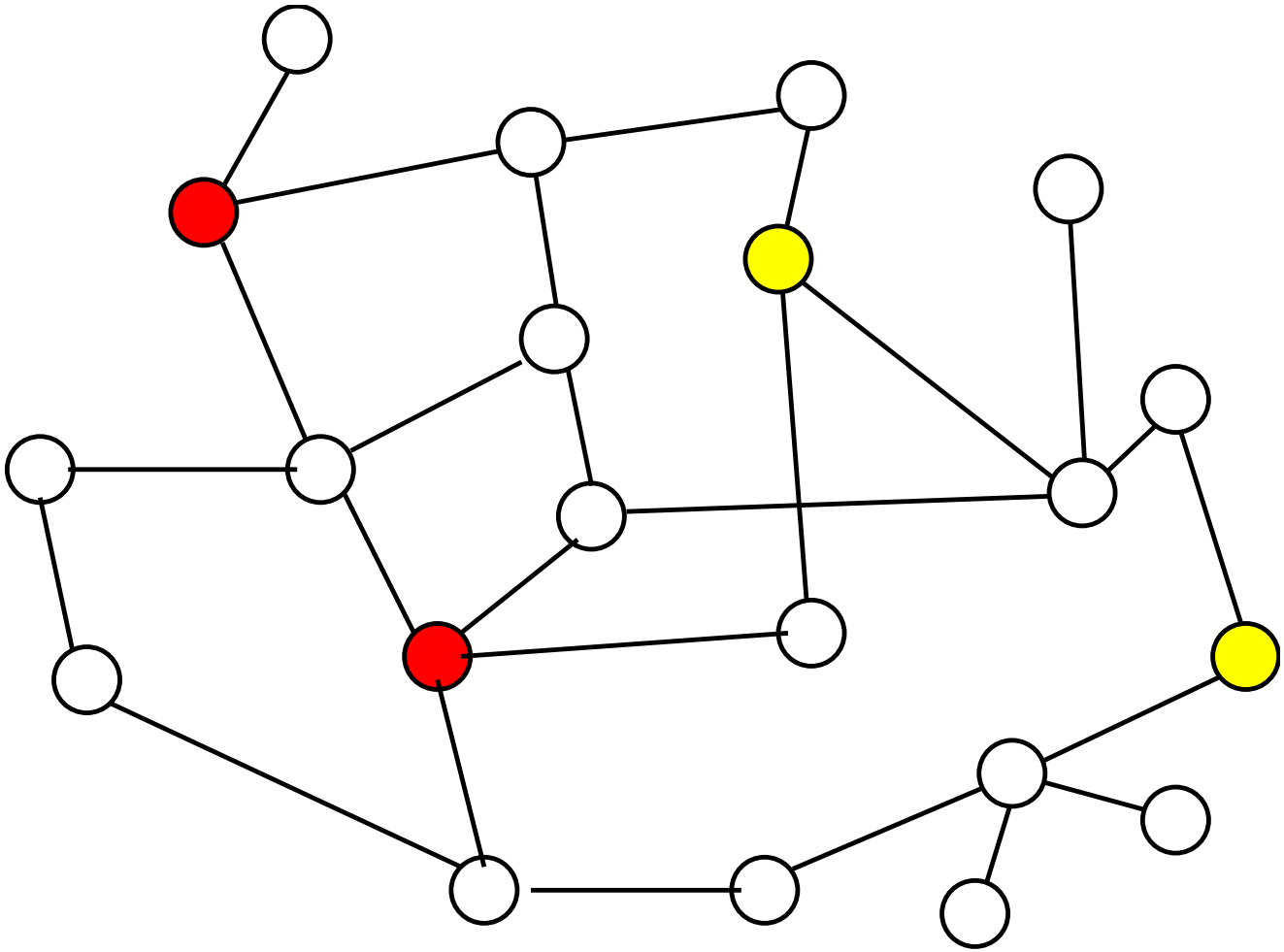
$$\| f \|_K^2 = f^\top K^{-1} f = \sum_{i \sim j} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j) \right)^2 + \epsilon \sum_{i=1}^{m} f(\mathbf{x}_i)^2 .$$

Outline

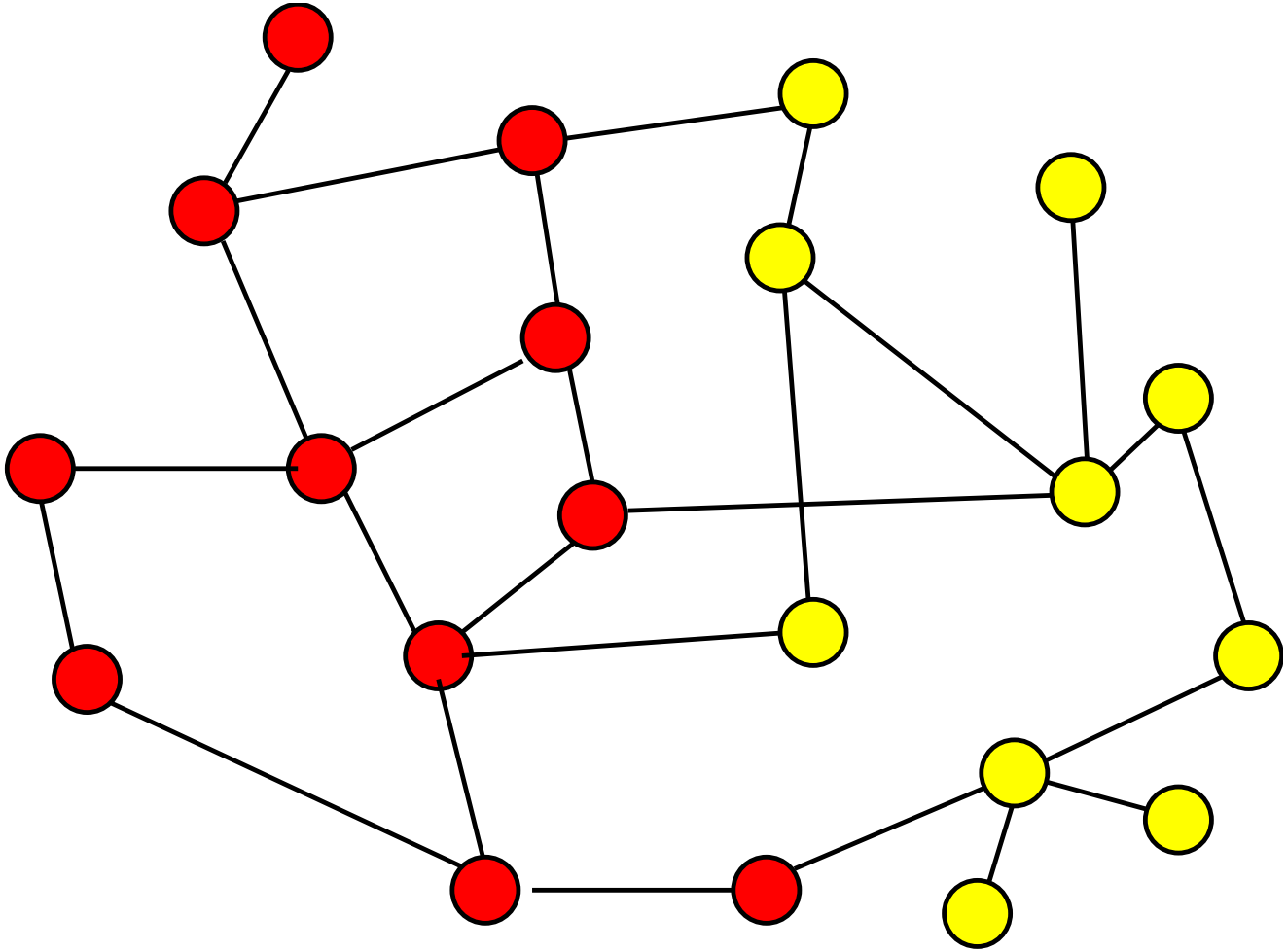
Motivations

- Learning on a graph can be useful by itself (e.g., predict protein functions from the protein-protein interaction network)
- This is a form of semi-supervised learning (unlabeled data can be used to create the kernel)
- The regularization functional can also be used as prior knowledge in high-dimensional microarray classification.

Semi-supervised learning



Semi-supervised learning



Tumor classification from microarray data

Data available

- Gene expression measures for more than $10k$ genes
- Measured on less than 100 samples of two (or more) different classes (e.g., different tumors)

Goal

- Design a classifier to automatically assign a class to future samples from their expression profile
- Interpret biologically the differences between the classes

Tumor classification from microarray data

Data available

- Gene expression measures for more than $10k$ genes
- Measured on less than 100 samples of two (or more) different classes (e.g., different tumors)

Goal

- Design a classifier to automatically assign a class to future samples from their expression profile
- Interpret biologically the differences between the classes

Linear classifiers

The approach

- Each sample is represented by a vector $x = (x_1, \ldots, x_p)$ where $p > 10^5$ is the number of probes
- Classification: given the set of labeled sample, learn a linear decision function:

$$f(x) = \sum_{i=1}^{p} \beta_i x_i + \beta_0 \ ,$$

  that is positive for one class, negative for the other
- Interpretation: the weight β_i quantifies the influence of gene i for the classification

Linear classifiers

Pitfalls

- No robust estimation procedure exist for 100 samples in 10^5 dimensions!
- It is necessary to reduce the complexity of the problem with prior knowledge.

Example : Norm Constraints

The approach

A common method in statistics to learn with few samples in high dimension is to constrain the norm of β, e.g.:

- Euclidean norm (support vector machines, ridge regression): $\| \beta \|_2 = \sum_{i=1}^{p} \beta_i^2$
- L_1-norm (lasso regression) : $\| \beta \|_1 = \sum_{i=1}^{p} | \beta_i |$

Pros

- Good performance in classification

Cons

- Limited interpretation (small weights)
- No prior biological knowledge

Example 2: Feature Selection

The approach

Constrain most weights to be 0, i.e., select a few genes (< 20) whose expression are enough for classification. Interpretation is then about the selected genes.

Pros

- Good performance in classification
- Useful for biomarker selection
- Apparently easy interpretation

Cons

- The gene selection process is usually not robust
- Wrong interpretation is the rule (too much correlation between genes)

Pathway interpretation

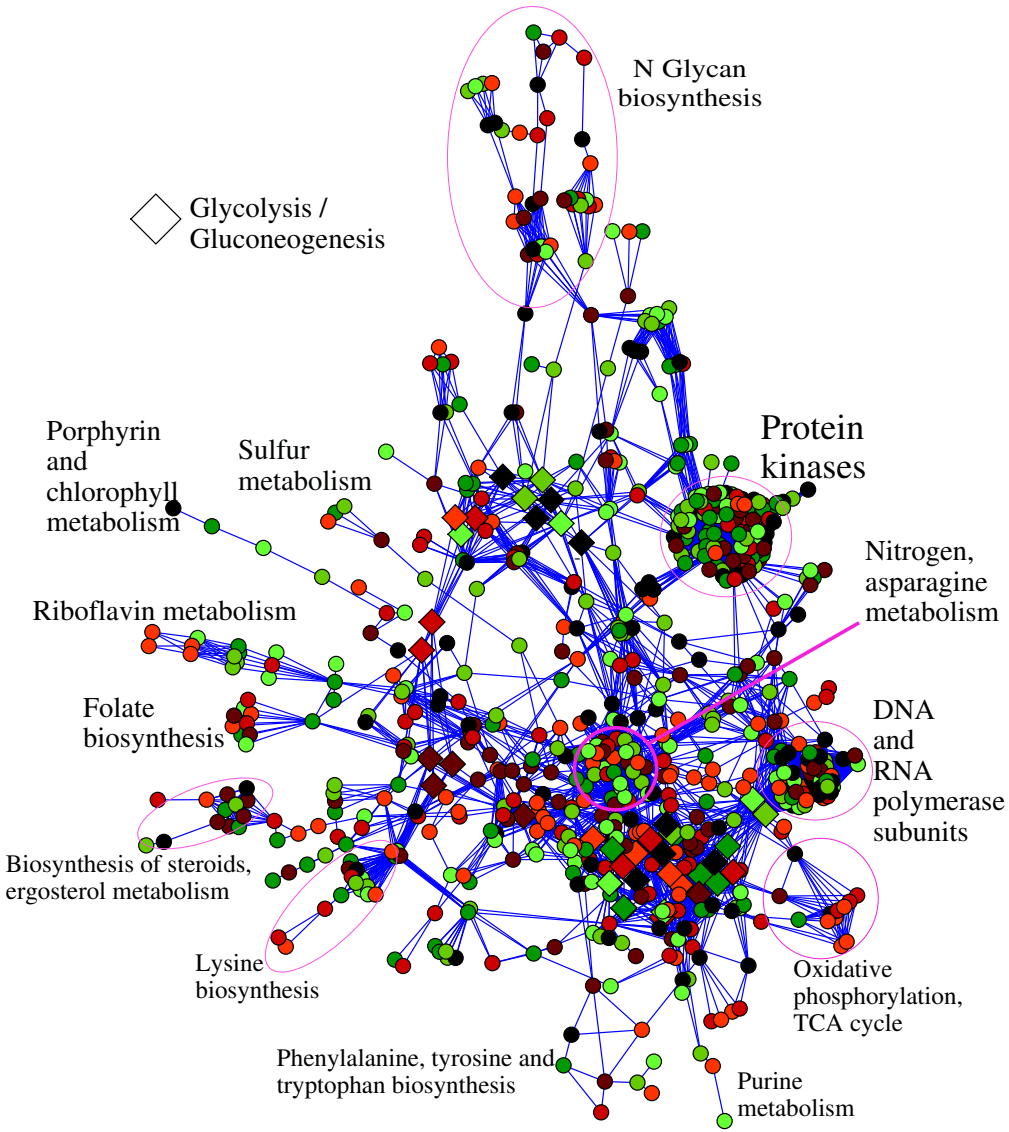
Motivation

- Basic biological functions are usually expressed in terms of pathways and not of single genes (metabolic, signaling, regulatory)
- Many pathways are already known
- How to use this prior knowledge to constrain the weights to have an interpretation at the level of pathways?

Solution (Rapaport et al., 2006)

- Constrain the diffusion RKHS norm of β
- Relevant if the true decision function is indeed smooth w.r.t. the biological network

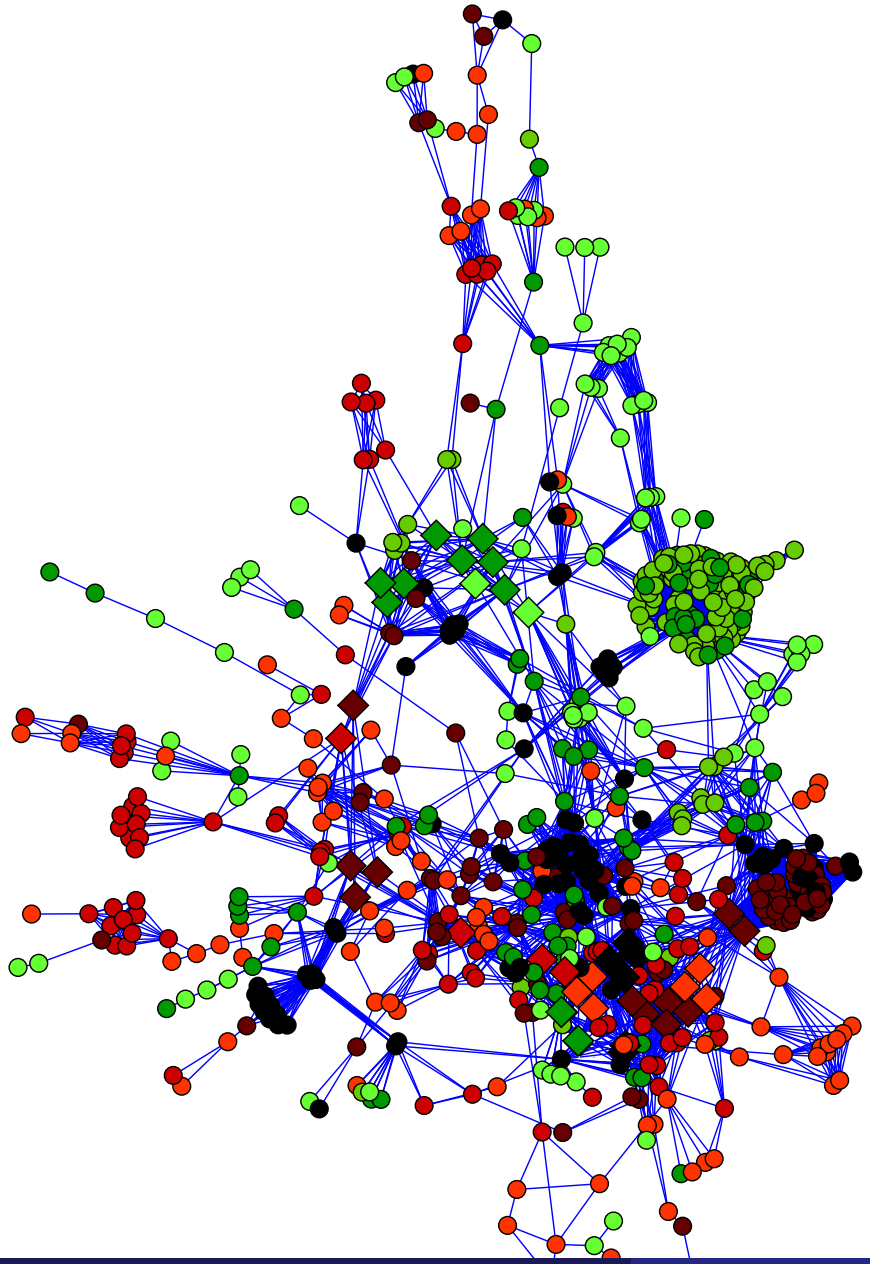
Pathway interpretation



Bad example

- The graph is the complete known metabolic network of the budding yeast (from KEGG database)
- We project the classifier weight learned by a SVM
- Good classification accuracy, but no possible interpretation!

Pathway interpretation



Good example

- The graph is the complete known metabolic network of the budding yeast (from KEGG database)
- We project the classifier weight learned by a spectral SVM
- Good classification accuracy, and good interpretation!

Conclusion

Conclusion

- Bioinformatics relies increasingly on machine learning
- Many things beyond this short tutorial (e.g., heterogeneous data integration by multiple kernel learning, graph inference, ...)
- The methods presented in this tutorial can be applied beyond bioinformatics
- Kernel methods are certainly not the end of the story, in particular more semantic is required to represent and manipulate biological systems.
- THANK YOU!