Support Vector Machines (SVM) in bioinformatics

Day 3:
Advanced topics and current research

Jean-Philippe Vert

Bioinformatics Center, Kyoto University, Japan
Jean-Philippe.Vert@mines.org

Human Genome Center, University of Tokyo, Japan, July 17-19, 2002.

3 days outline

- Day 1: Introduction to SVM
- Day 2: Applications in bioinformatics
- Day 3: Advanced topics and current research

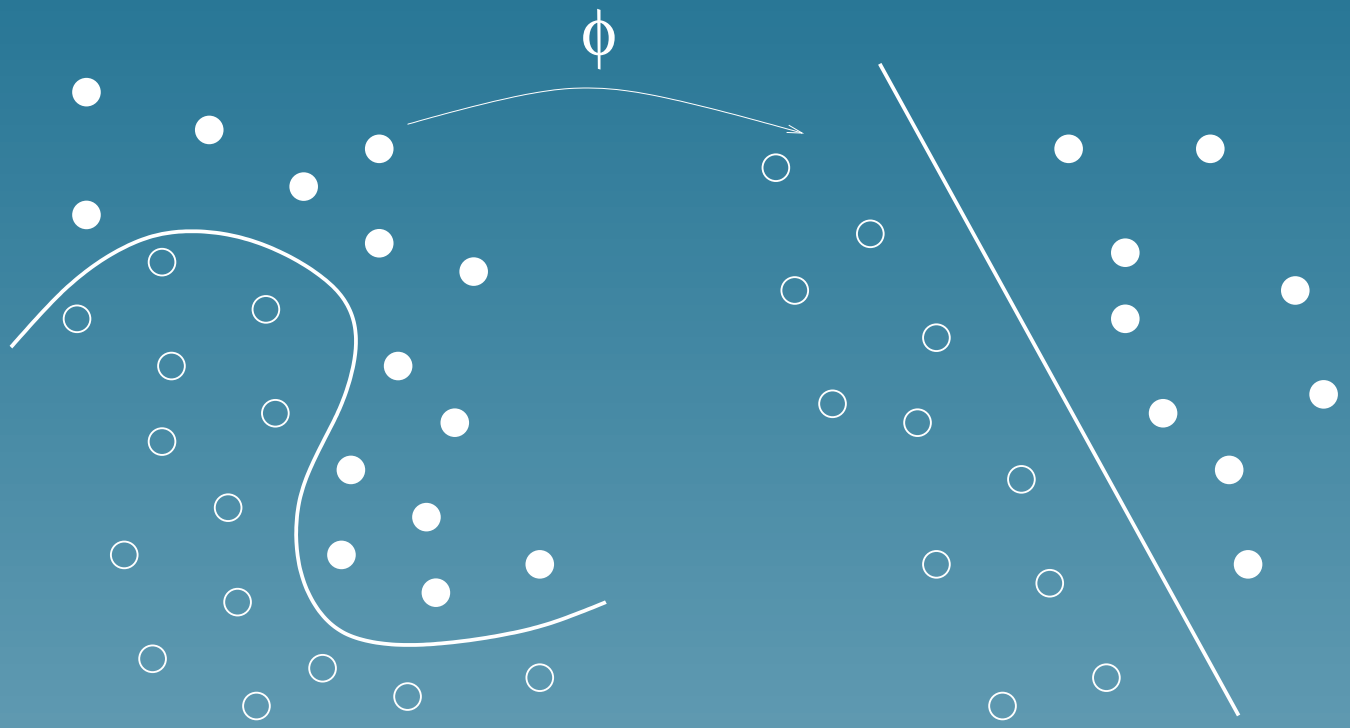
Today's outline

1. Kernel engineering
2. Other kernel methods
3. Example: graph-driven feature extraction from microarray data

Part 1

Kernel engineering

Remember the kernel



$$K(x, x') = \vec{\Phi}(x).\vec{\Phi}(x')$$

Properties of the kernel

- A kernel is a similarity measure
- It defines the geometry of the feature space (lengths and angles)
- A function $K(x, x')$ is a kernel if and only if the following matrix is symmetric positive definite (all eigenvalues are positive) for all choices of $(x_1, \ldots, x_n)$:

$$K = \begin{pmatrix} K(x_1, x_1) & K(x_1, x_2) & \ldots \\ K(x_2, x_1) & K(x_2, x_2) & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

3 ways to make kernels

- Define a set of features of interest, compute the feature vector of every gene, and compute the dot products (see examples in yesterday's talk).
- Define a large set of features and find tricks to compute the dot product implicitly (without computing the feature vectors)
- Start with a similarity measure you find pertinent (e.g., SW score) and check that it is a kernel.

Kernel engineering

Particular kernels can be imagined to include prior knowledge about:

- the types of data (vectors, sequences, graphs...)
- the problem at hand

into the geometry of the feature space.

This process is called kernel engineering

Examples of kernel engineering

- Kernels for sequences based on common subsequences
- Kernel to recognize translation initiation site
- Convolution kernels
- Kernels built from Bayesian tree models
- Diffusion kernels on graphs

Kernel engineering 1

Kernels for sequences based on common subsequences

Motivation

- Goal: define a kernel for variable-length sequences (useful to handle bio-polymers)
- Intuition: two sequences are related when they share common substrings or subsequences.

References

- H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini and C. Watkins. **Text classification using string kernels**. *Journal of Machine Learning Research*, 2:419-444, 2002.

- C. Leslie, E. Eskin and W.S. Noble. **The spectrum kernel: a string kernel for svm protein classification**. Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Kevin Lauerdale, Teri E. Klein, , *Proceedings of the Pacific Symposium on Biocomputing 2002*, 564-575. World Scientific, 2002.

Substrings

- A string $s = s_1, \ldots, s_p$ is a substring of a string $x = x_1, \ldots, x_n$ (with $n \geq p$) if the letters of s appear in the same order in x (gaps allowed).
- The length $l(s, x)$ of a substring s in a string x is the distance between the first and the last letter in x
- Example: $s =$ `ofot` is a substring of $x =$ `bioinformatics`, with length $l(s, x) = 9$.

String matching kernel (Lohdi et al., 2002)

- The string matching kernel is defined by:

$$K(x, x') = \sum_{s \text{ common substring}} \lambda^{l(s,x)+l(s,x')},$$

  where λ is a parameter.

- Two strings are similar when they share many common substrings
- The feature space is the space of all possible substrings

Computation of the string matching kernel

- The dimension of the feature space is very large (number of possible substrings), but...
- There exists a dynamic programming method to compute the kernel $K(x, x')$ between any two sequences in $O(|x||x'|n)$, where n is the length of the substrings considered.
- Promising results on text classification

Spectrum kernel (Leslie et al., 2002)

- Same idea, but gaps not allowed (common sub-blocks)
- Efficient implementation using a suffix tree
- Classification of a sequence x in $O(|x|)$ using a sliding window
- Encouraging results on remote homology detection (superfamily prediction): performs like PSI-Blast, a bit lower than SAM and SVM+Fisher kernel

Kernel engineering 2

Kernel to recognize translation initiation site

The problem

- Translation initiation sites (TIS) are the position in DNA where regions coding for proteins start
- All coding sequences start with the start codon `ATG`
- Given a `ATG` in a DNA sequence, is it a TIS?

References

- A. Zien, G. Ratsch, S. Mika, B. Schölkopf, T. Lengauer and K.-R. Muller. **Engineering support vector machine kernels that recognize translation initiation sites**. *Bioinformatics*, 16(9):799-807, 2000.

Formulation

- Pick up a window of 200 nucleotides centered around the candidate ATG
- Encode each nucleotide with a 5 bits word: 00001,. . . ,10000 for A,C,G,T and unknown.
- Use this 1000 long bit vectors to train a SVM to predict whether the central ATG corresponds to a TIS
- Which kernel to use?

Polynomial kernels

$$K(\vec{x}, \vec{x}') = (\vec{x}.\vec{x}')^d$$

The corresponding feature space is made of C_{n-1}^d monomials features of degree d

- $d = 1$: counts the number of common bits
- $d = 2$: counts the number of common pairs of bits (pairwise correlations)
- etc...

Locally improved kernels

- Intuition: while certain local correlations are typical for TIS, dependencies between distant positions are of minor importance or do not even exist. They only add noise to the feature space.

- At each sequence position, sequences can be compared locally using a small window of length $2l+1$ with inner correlations of up to d_1 positions:

$$win_p(x,x') = \left(\sum_{j=-l}^{+l} w_j \mathsf{match}_{p+j}(x,x') \right)^{d_1} .$$

Locally improved kernels (ctd.)

- Add the contributions of all windows, and of correlations between up to d_2 windows:

$$K(x, x') = \left(\sum_{p=1}^{n} win_p(x, y) \right)^{d_2}$$

Results

$d_2 > 1$ (long-range correlations) does not improve performance

Method	Overall error (%)
Neural network	15.4
Salzberg method	13.8
SVM, linear kernel	13.2
SVM, locally improved kernel ($d_1 = 4$, $l = 4$)	11.9

Kernel engineering 3

Convolution kernels

Intuition

- Many beautiful probabilistic models exist for biological sequences (HMM)
- They involve observed data (the sequence x) and hidden variable (the hidden states s)
- Intuition: two sequences x and x' are similar if they are likely to have the same hidden state sequence

References

- D. Haussler. **Convolution kernels on discrete structures**. , Technical report UC Santa Cruz, 1999.

- C. Watkins. **Dynamic alignment kernels**. Proceedings of NIPS 1999.

Convolution kernel for HMM

- Let $p(x, s)$ the probability for the complete variable.
- The convolution kernel between two sequences x and x' is defined by:

$$K(x, x') = \sum_s p(s)p(x|s)p(x|s).$$

- It can be computed using a dynamic programming algorithm (equivalent to pair HMM score, a variant of the Smith-Waterman algorithm)

Remarks

- It shows that natural ways to measure the similarity between sequences (pair HMM score) are in fact kernels.
- Uses only the distribution $p(x, s)$, and not the structure of the parametric model (unlike the Fisher kernel).

Kernel engineering 4

Kernel for strings based on rare common substrings

References

- J.-P. Vert. **Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings**. Russ B. Altman, A. Keith Dunker, Lawrence Hunter, Kevin Lauerdale, Teri E. Klein, , *Proceedings of the Pacific Symposium on Biocomputing 2002*, 649-660. World Scientific, 2002.

Motivations

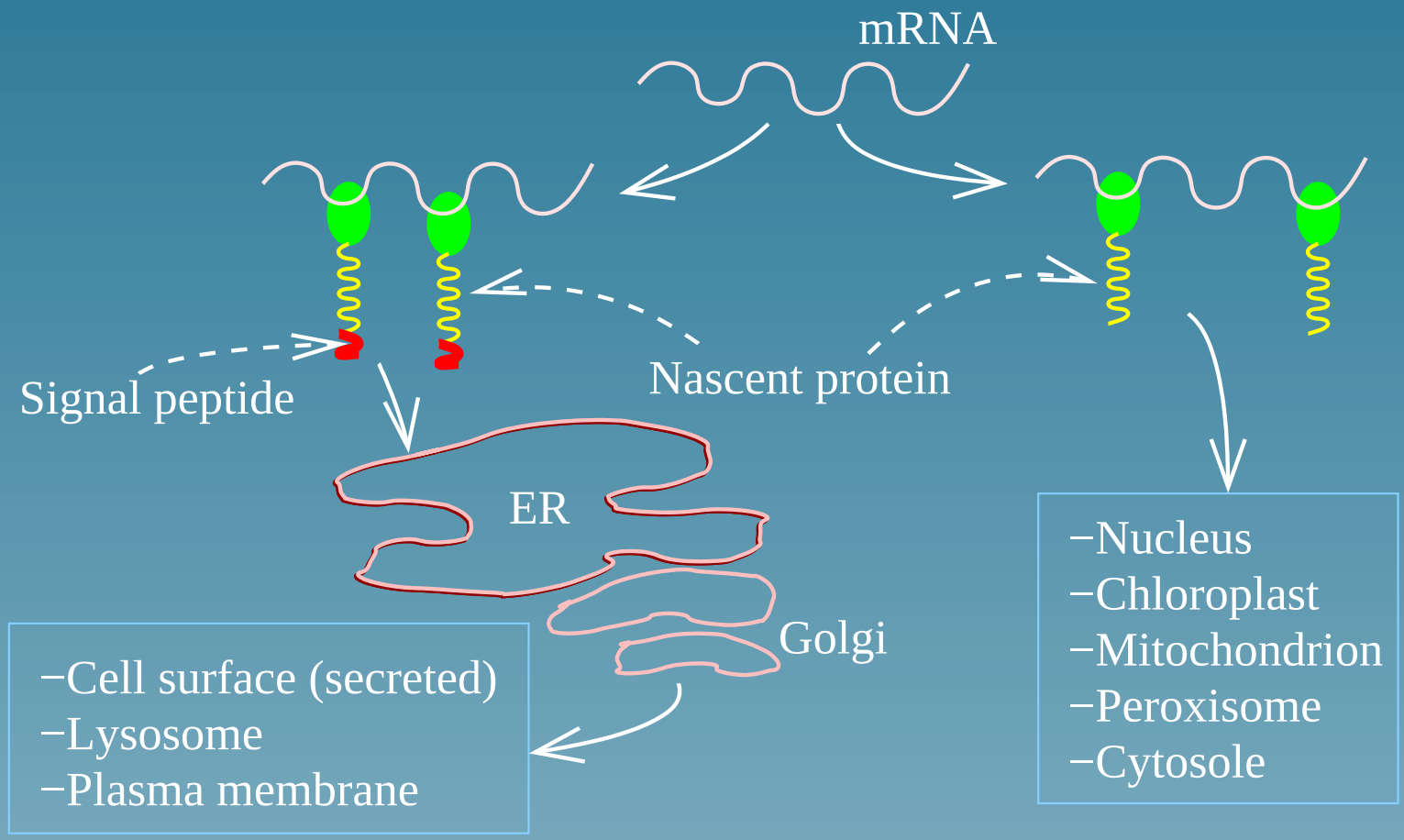
- Goal: a kernel for fixed-length strings (sequence windows...)
- Intuition: two strings should get closer in the feature space when they share rare common substrings
- Solution:
  - ⋆ Let p a probability distribution on the set of sequences of length m (e.g., a position specific weight matrix)
  - ⋆ The kernel between two strings x and y is:

$$K(x,y) = p(x)p(y) \sum_{s \text{ common substring}} \frac{1}{p(s)}$$

Properties of the string kernel

- $K(.,.)$ is a kernel
- Two strings get closer in the feature space when they share rare common subparts
- Efficient computation: For sequences of length m, there is an algorithm to compute the kernel with a complexity $O(m)$ (even though there are up to 2^m common substrings)

Application: SVM prediction of signal peptide cleavage site (1)



Signal peptides

Protein	-1 +1
(1)	MKANAKTIIAGMIALAISHTAMA EE...
(2)	MKQSTIALALLPLLFTPVTKA RT...
(3)	MKATKLVLGAVILGSTLLAG CS...

(1):Leucine-binding protein, (2):Pre-alkaline phosphatase, (3)Pre-lipoprotein

- 6-12 hydrophobic residues (in yellow)
- (-3,-1) : small uncharged residues

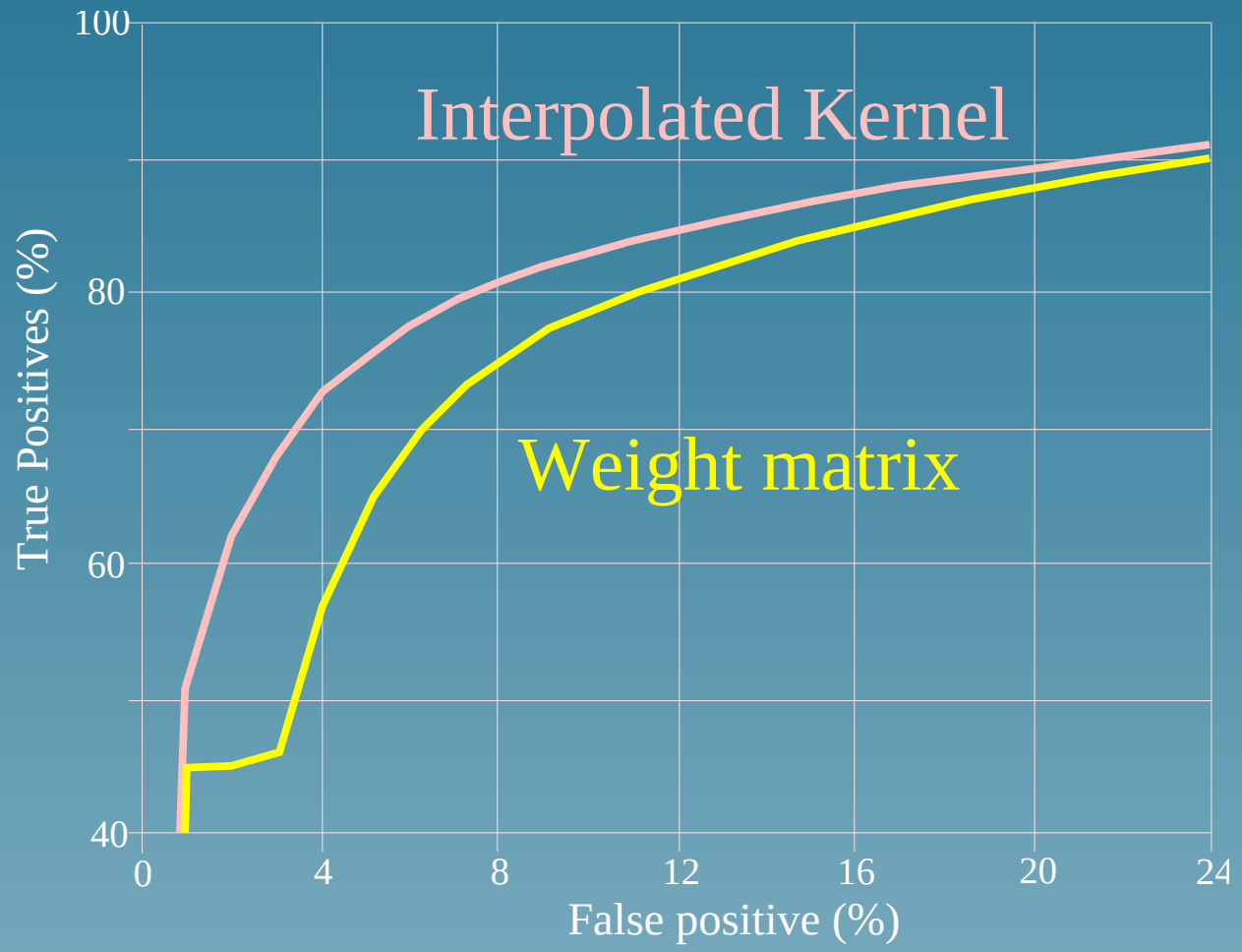
Experiment

- Challenge : classification of aminoacids windows, positive if cleavage occurs between -1 and $+1$:

$$[x_{-8}, x_{-7}, \ldots, x_{-1}, x_1, x_2]$$

- 1,418 positive examples, 65,216 negative examples
- Classification by a weight matrix;
- Classification by a SVM + string kernel

Result: ROC curves



Kernel engineering 5

Kernel for phylogenetic profiles

References

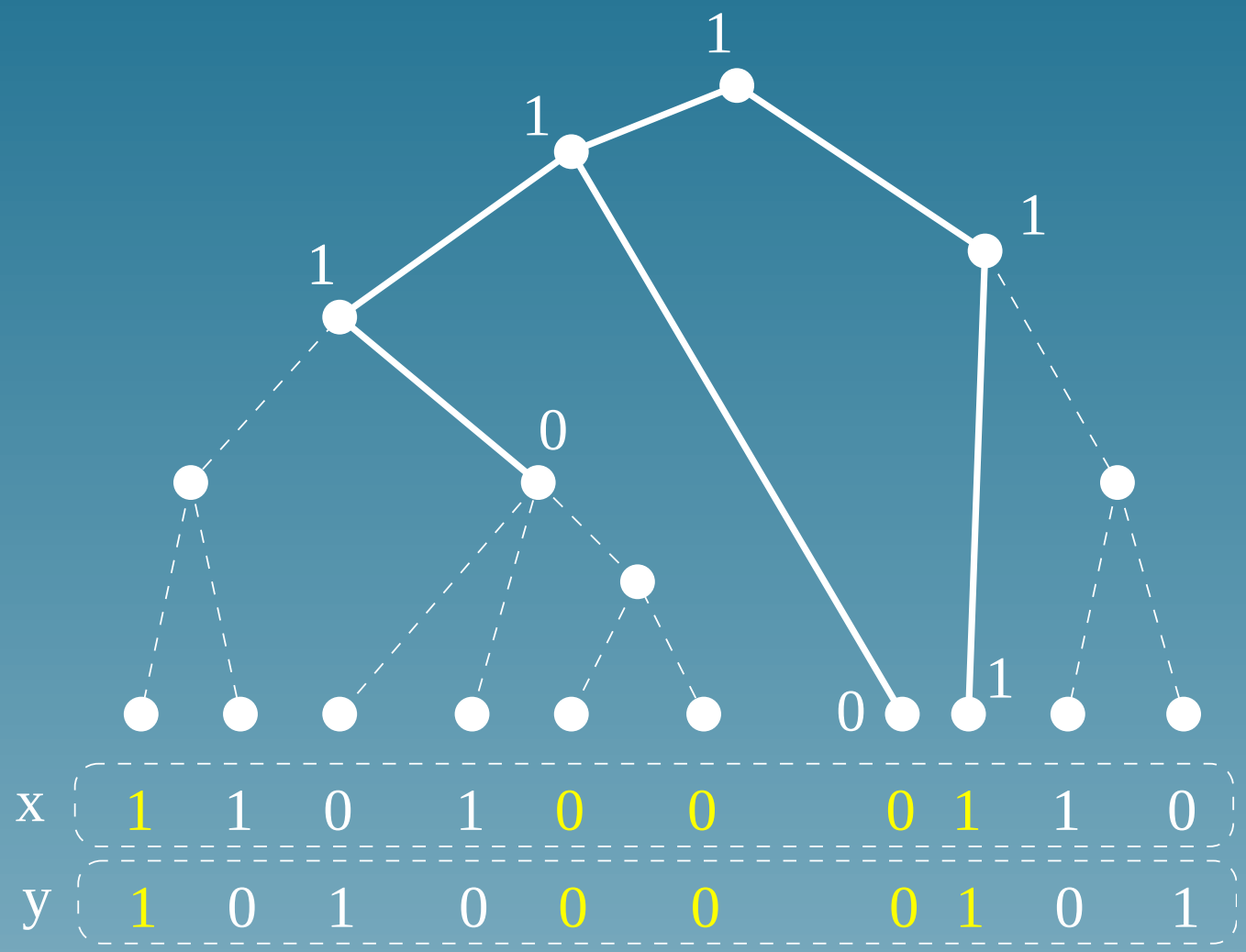
- J.-P. Vert. **A tree kernel to analyze phylogenetic profiles**. Proceedings of ISMB 2002, *Bioinformatics*, 2002. To appear.

Kernel for phylogenetic profiles

- Goal: a kernel for phylogenetic profiles (a string of bit which indicates the presence or absence of an homolog in every fully sequenced organism)
- Intuition: two genes should get closer in the feature space when they are likely to have shared common evolution patterns
- Solution Create a simple probabilistic model for the transmission of genes between species during evolution, and

$$K(x,y) = \sum_{e \text{ evolution pattern}} p(e)p(x|e)p(y|e)$$

Evolution patterns



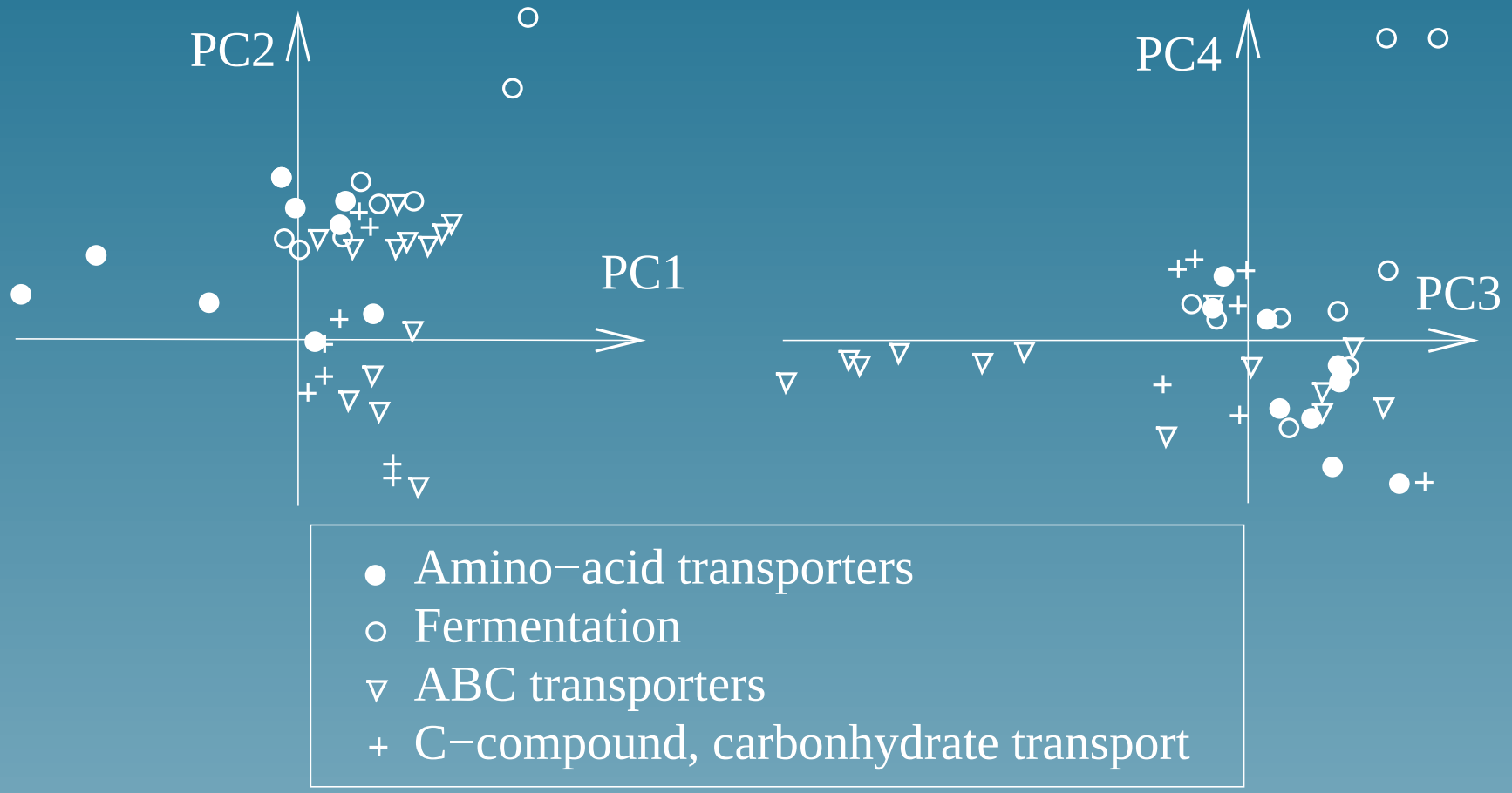
Properties of the tree kernel

- $K(.,.)$ is a kernel
- Two profiles get closer in the feature space when they have shared common evolution patterns with high probability
- Efficient computation: For profiles of length m, there is an algorithm to compute the kernel with a complexity $O(m)$ (even though there is an exponential number of evolution patterns)

Application: SVM function prediction from phylogenetic profiles (ROC_{50} performance)

Functional class	Dot kernel	Tree kernel	Difference
Amino-acid transporters	0.74	0.81	**+ 9%**
Fermentation	0.68	0.73	**+ 7%**
ABC transporters	0.64	0.87	**+ 36%**
C-compound transport	0.59	0.68	**+ 15%**
Amino-acid biosynthesis	0.37	0.46	**+ 24%**
Amino-acid metabolism	0.35	0.32	*- 9%*
Tricarboxylic-acid pathway	0.33	0.48	**+ 45%**
Transport Facilitation	0.33	0.28	*- 15%*

Application: kernel PCA of phylogenetic profiles



Kernel engineering 6

Diffusion kernels

References

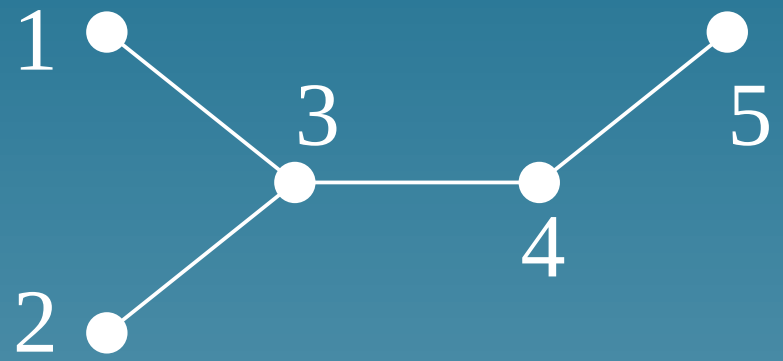
- R. I. Kondor and J. Lafferty. **Diffusion kernels on graphs and other discrete input**. *ICML 2002*. 2002.

Making a kernel from a graph

- Goal: Suppose you can define binary relations between genes (e.g., protein interaction). How to define a kernel for genes which reflects the topology of the graph?
- Intuition: Two nodes get closer in the feature space when there are many short paths between them in the graph
- Solution Let $L = D - A$ be the Laplacian matrix (D is the diagonal degree matrix, A the adjacency matrix) For any $\lambda > 0$, the kernel matrix is:

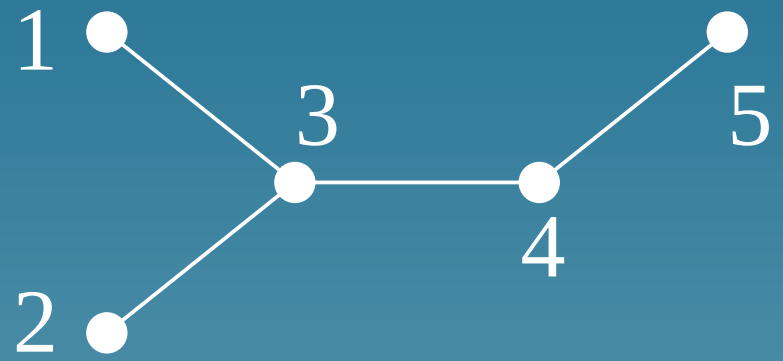
$$K = \exp(-\lambda L)$$

Example of a graph kernel (1)



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

Example of a graph kernel (2)



$$K = \exp(-L) = \begin{pmatrix} 0.49 & 0.12 & 0.23 & 0.10 & 0.03 \\ 0.12 & 0.49 & 0.23 & 0.10 & 0.03 \\ 0.23 & 0.23 & 0.24 & 0.17 & 0.10 \\ 0.10 & 0.10 & 0.17 & 0.31 & 0.30 \\ 0.03 & 0.03 & 0.10 & 0.30 & 0.52 \end{pmatrix}$$

Summary: Kernel engineering

- Kernels can be engineered to include some prior (biological) knowledge in the geometry of the feature space
- The biological knowledge is an intuition about "when two objects (genes) should be considered similar / close to each other".
- Once engineered, the kernel can be used by any kernel method for various purpose (sound mathematical framework)
- Kernel engineering is an active field of research currently

Part 2

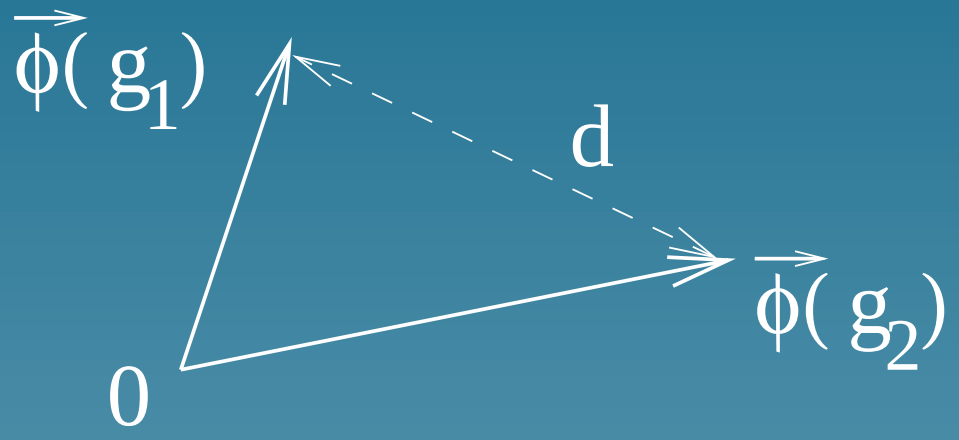
More kernel methods

Overview

Suppose you are given a kernel $K(.,.)$. Then you can perform various operations in the feature space without computing the image Φg of each gene g:

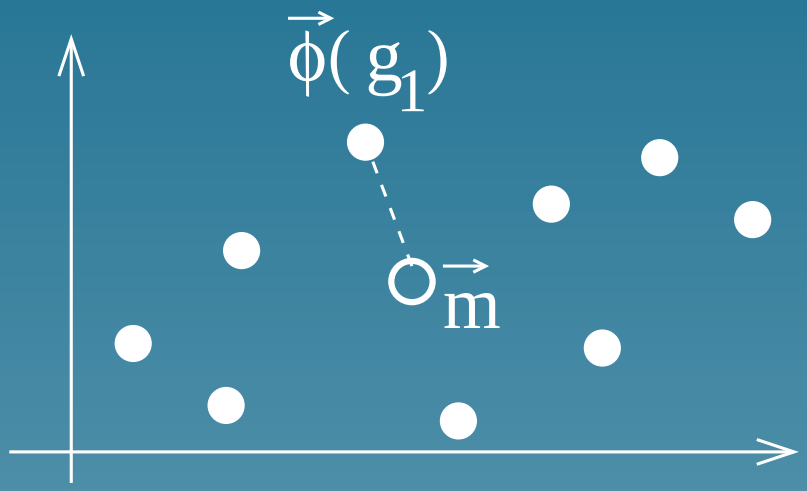
- Compute the distance between any two genes, or between any gene and the center of mass of the gene database
- Principal component analysis (PCA)
- Canonical correlation analysis (CCA)
- Classify the genes into classes (Support vector machines)

Distance between two genes



$$d(g_1, g_2)^2 = \|\Phi g_1 - \Phi g_2\|^2$$
$$= (\Phi g_1 - \Phi g_2) . (\Phi g_1 - \Phi g_2)$$
$$= \Phi g_1 . \Phi g_1 + \Phi g_2 . \Phi g_2 - 2\Phi g_1 . \Phi g_2$$
$$d(g_1, g_2)^2 = K(g_1, g_1) + K(g_2, g_2) - 2K(g_1, g_2)$$

Distance between a gene and the center of mass



Center of mass: $\vec{m} = \frac{1}{N}\sum_{i=1}^{N} \Phi g_i$, hence:

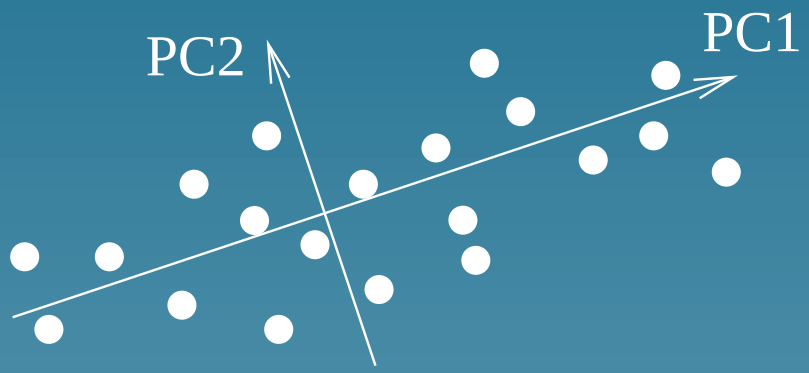
$$\|\Phi g_1 - \vec{m}\|^2 = \Phi g_1.\Phi g_1 - 2\Phi g_1.\vec{m} + \vec{m}.\vec{m}$$

$$= K(g_1, g_1) - \frac{2}{N}\sum_{i=1}^{N} K(g_1, g_i) + \frac{1}{N^2}\sum_{i,j=1}^{N} K(g_i, g_j)$$

Example: greedy multiple alignment (Gorodkin et al., GIW 2001)

- Use the SW score as a kernel for sequences (?)
- Compute the distance between each sequence and the center of mass
- First align the sequences near the center of mass
- Then add sequences one by one to the multiple alignment, by increasing distance from the center of mass

Principal component analysis (PCA)

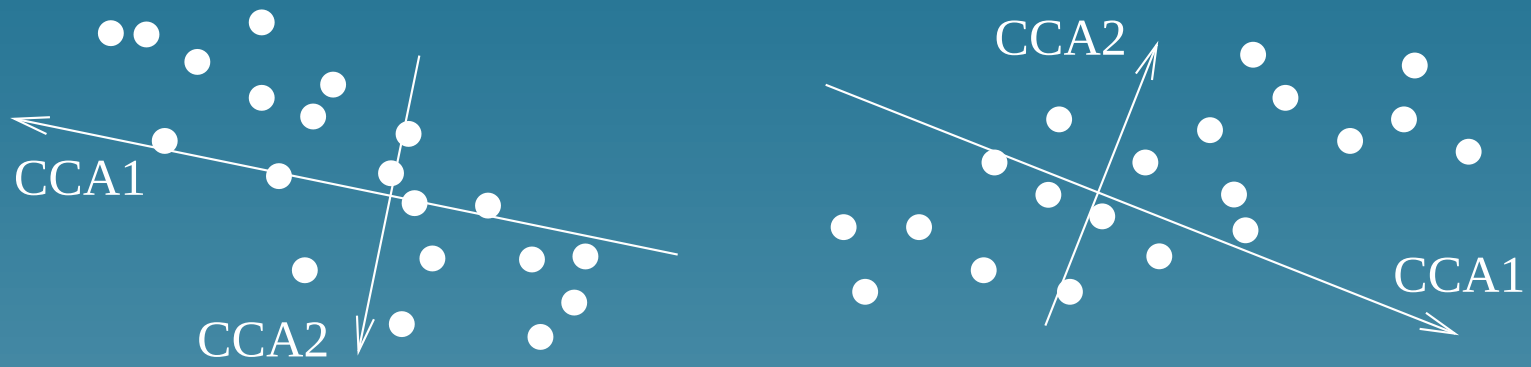


Find the eigenvectors of the matrix:

$$K = (\Phi g_i . \Phi g_j)_{i,j=1\ldots N}$$
$$= \big(K(g_i, g_j)\big)_{i,j=1\ldots N}$$

Useful to represent the objects as small vectors (feature extraction).

Canonical correlation analysis (CCA)



K_1 and K_2 are two different kernels for the same objects (genes). CCA is performed by solving the generalized eigenvalue problem:

$$\begin{pmatrix} 0 & K_1K_2 \\ K_2K_1 & 0 \end{pmatrix} \vec{\xi} = \rho \begin{pmatrix} K_1^2 & 0 \\ 0 & K_2^2 \end{pmatrix} \vec{\xi}$$

Useful to find correlations between different representations of the same objects

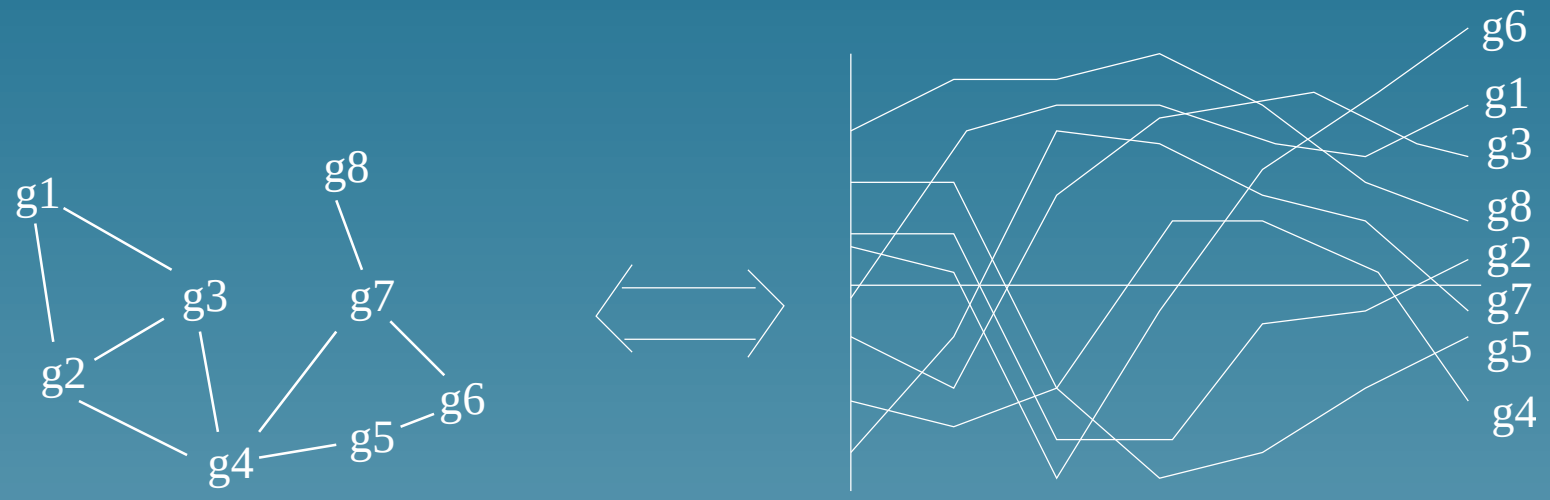
More kernel methods

- Any algorithm can be kernelized if it can be expressed in terms of inner product
- The library of kernel methods include SVM, kernel-PCA, kernel-CCA, kernel-Fisher discriminant, kernel-ICA, kernel-clustering, ...
- Modularity : any kernel can be used with any kernel method

Part 3

Example: graph-driven features extraction from microarray data

The problem



Gene network

Expression profiles

Are there "correlations"?

References

- J.-P. Vert and M. Kanehisa, **Graph-driven features extraction from microarray data**, Preprint, June 2002.

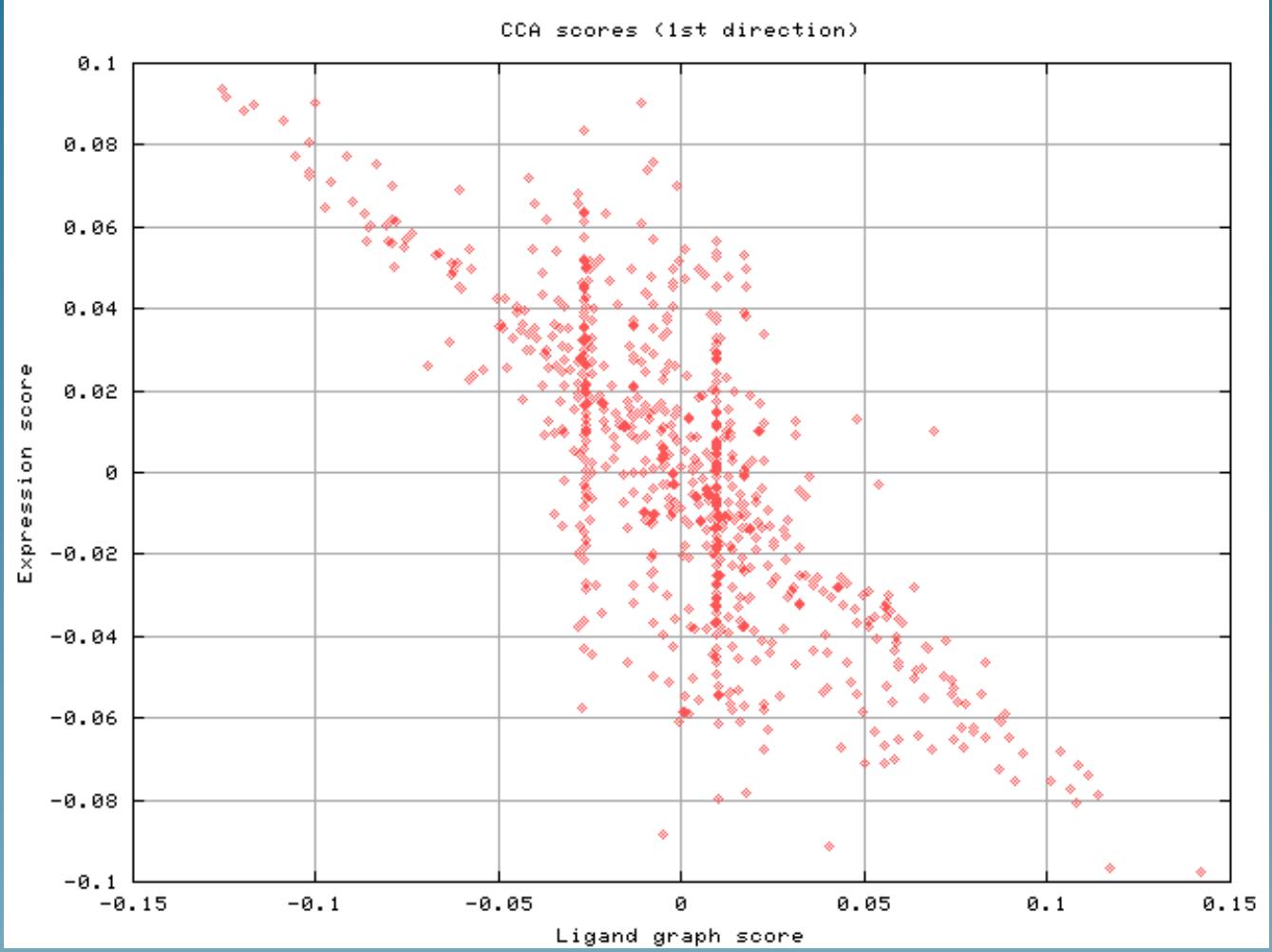
Approach

- From the microarray data build a kernel K_1 for genes using a linear kernel
- Use the gene network to build a kernel K_2 for genes using a diffusion kernel
- Perform a kernel CCA between K_1 and K_2 to extract correlations between the corresponding feature spaces

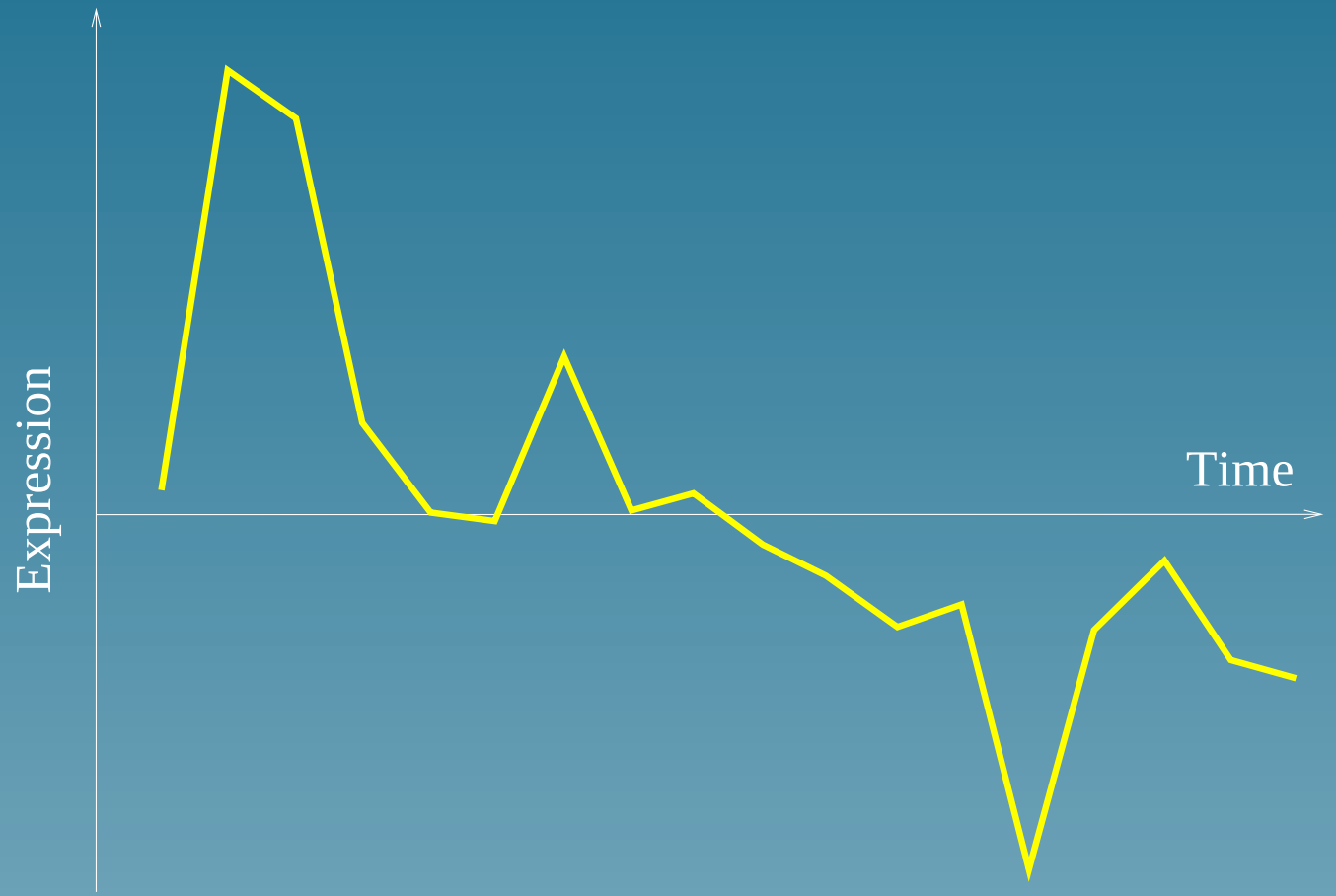
Data

- Gene network: genes are linked if they are known to catalyze two successive reactions (data available in Kyoto University's KEGG database, `www.genome.ad.jp`)
- Microarray data: 18 measures for all genes (6,000) of the budding yeast S. Cerevisiae by Spellman et al. (public data), corresponding to a cell cycle after release of alpha factor.

1st CCA scores



Upper left expression



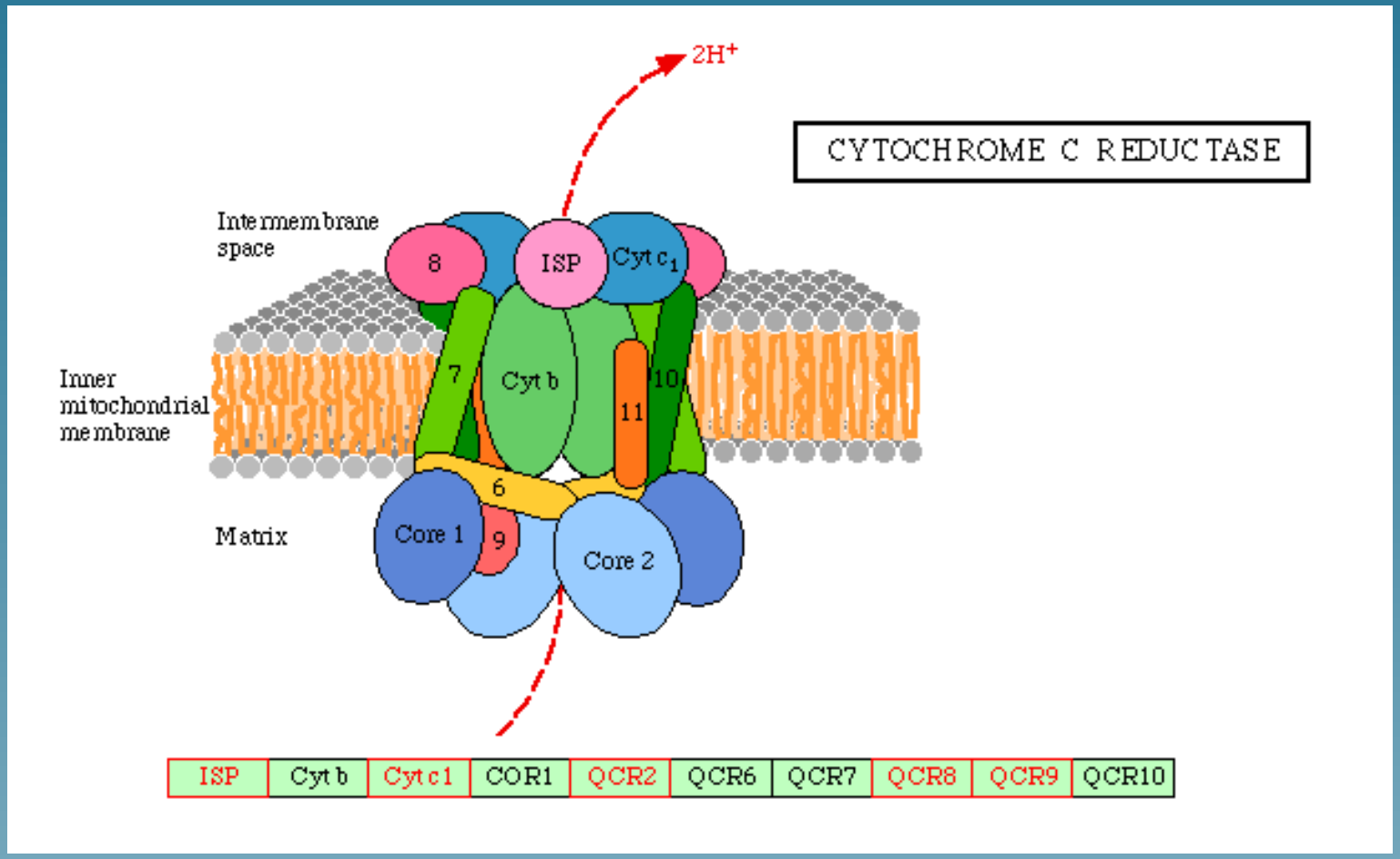
Average expression of the 50 genes with highest $s_2 - s_1$.

Upper left genes

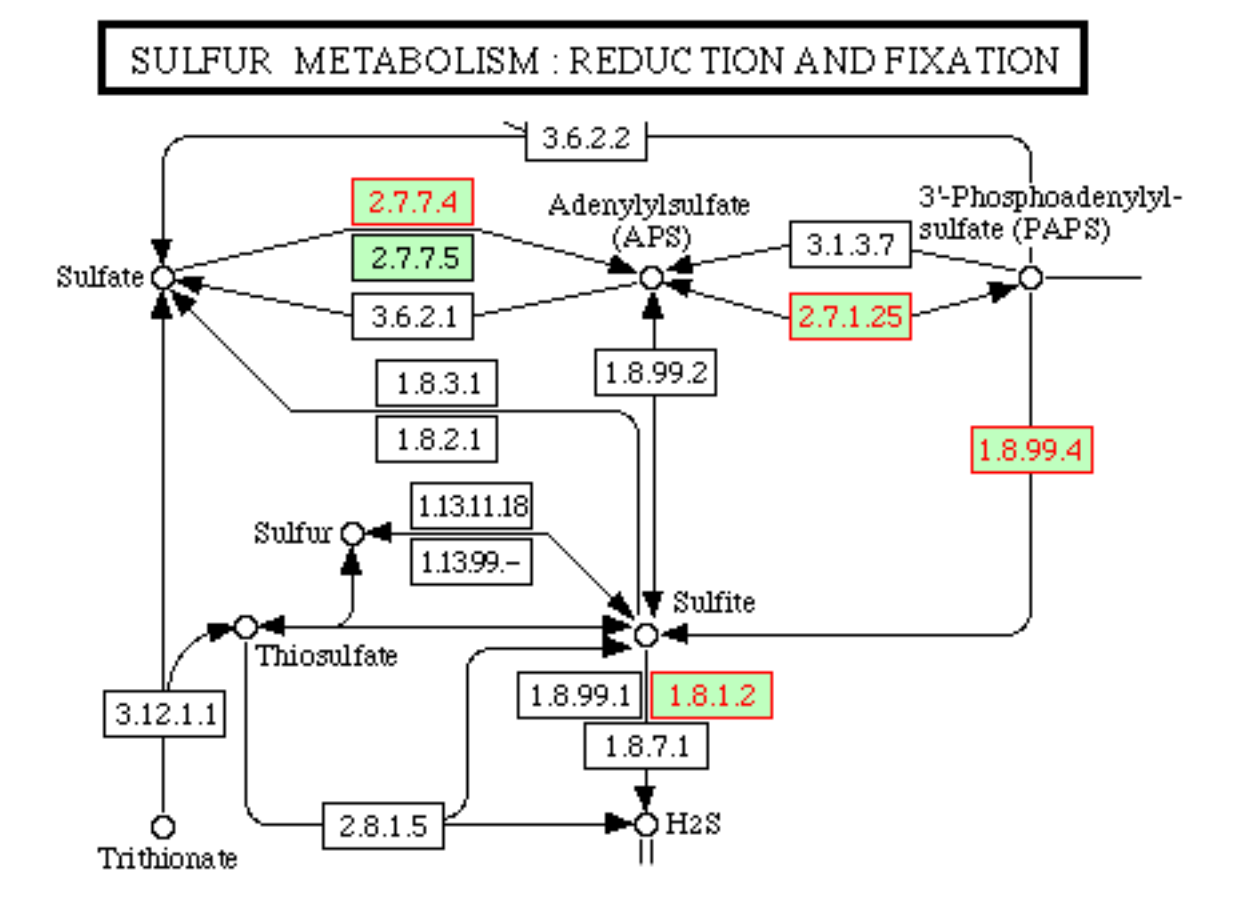
50 genes with highest $s_2 - s_1$ belong to:

- Oxidative phosphorylation (10 genes)
- Citrate cycle (7)
- Purine metabolism (6)
- Glycerolipid metabolism (6)
- Sulfur metabolism (5)
- Selenoaminoacid metabolism (4) , etc...

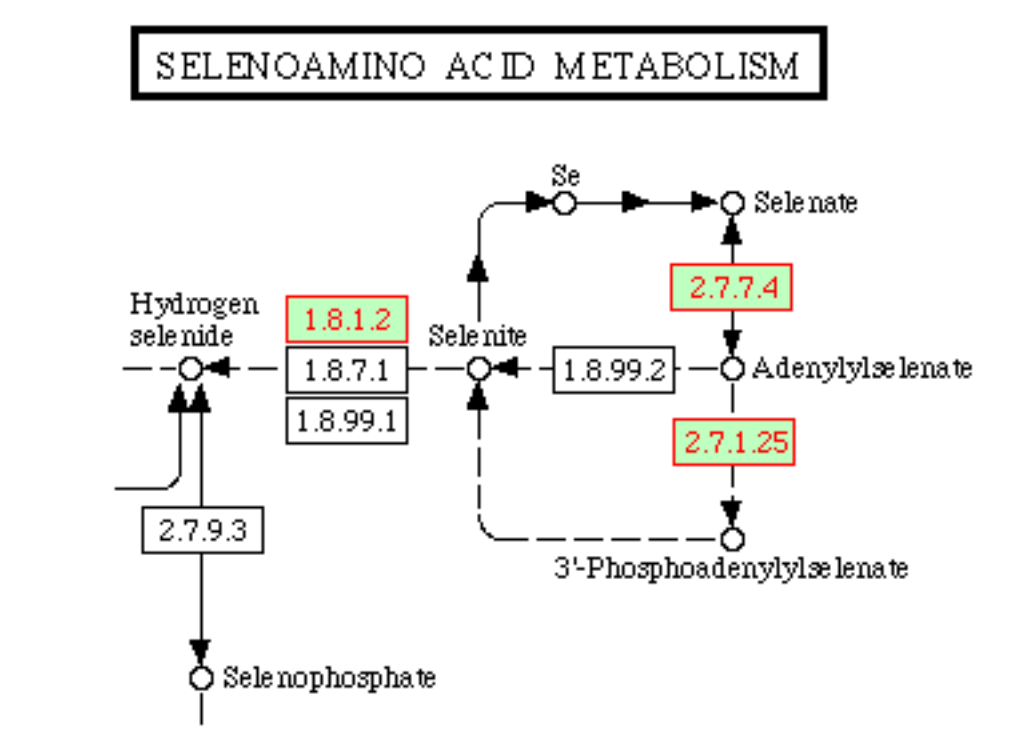
Upper left genes



Upper left genes



Upper left genes



Lower right expression

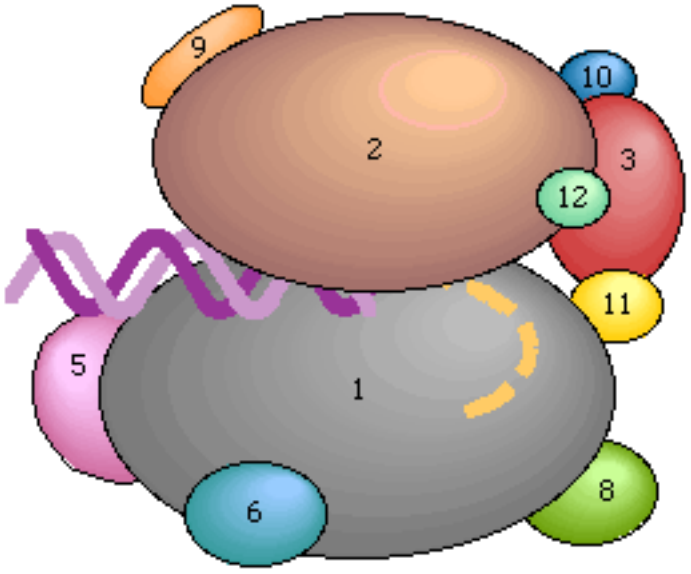
Average expression of the 50 genes with highest $s_2 - s_1$.

Lower right genes

- RNA polymerase (11 genes)
- Pyrimidine metabolism (10)
- Aminoacyl-tRNA biosynthesis (7)
- Urea cycle and metabolism of amino groups (3)
- Oxidative phosphorylation (3)
- ATP synthesis(3) , etc...

Lower right genes

RNA POLYMERASE



RNA polymerase II (Saccharomyces cerevisiae)

Eukaryotic Pol II

B2	B3	B4	B5	B6	B7
B1	B8	B9	B10	B11	B12

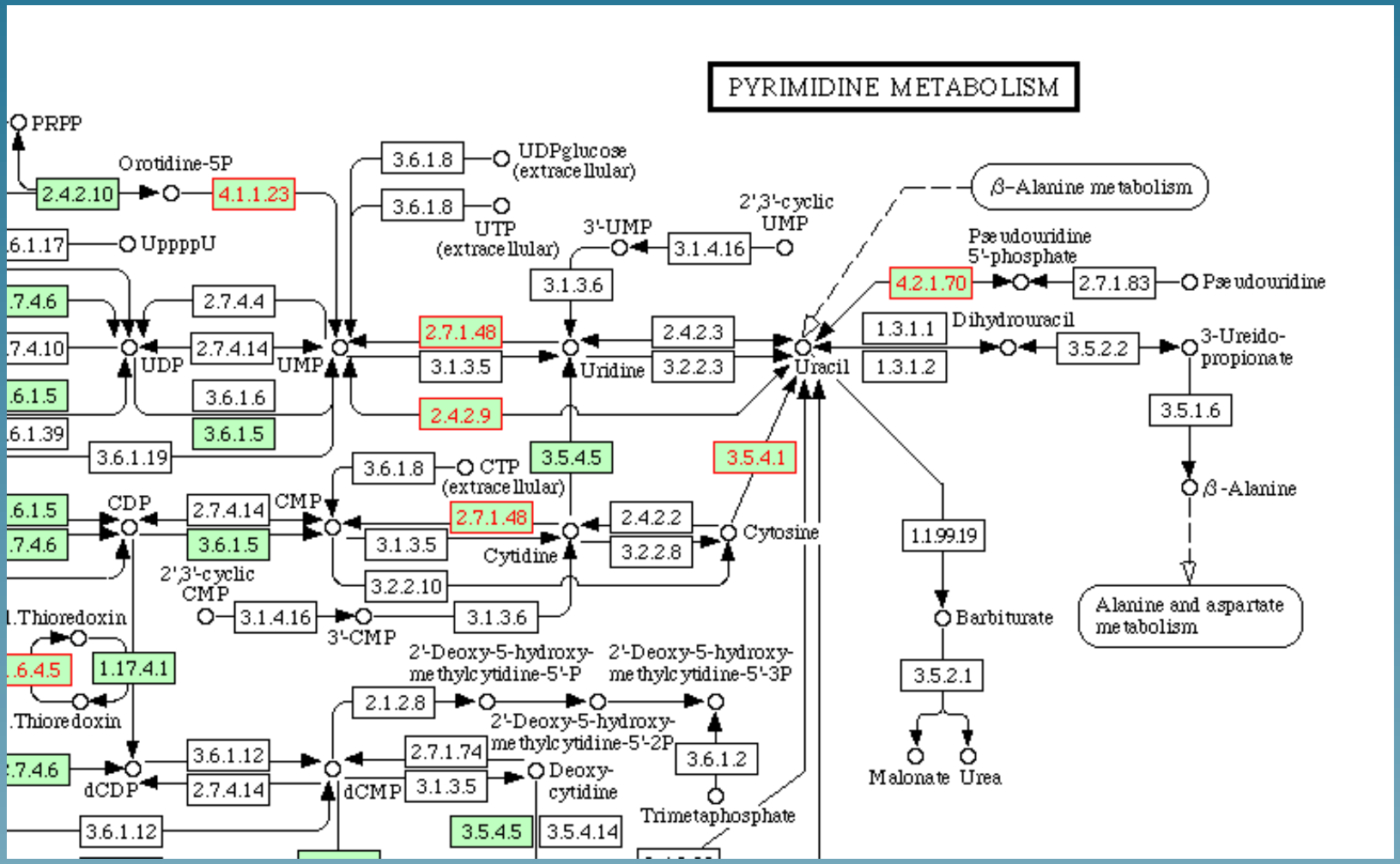
Eukaryotic Pol III

C2	C3	C4	C5	C11
C1	C19	C25	C31	C34

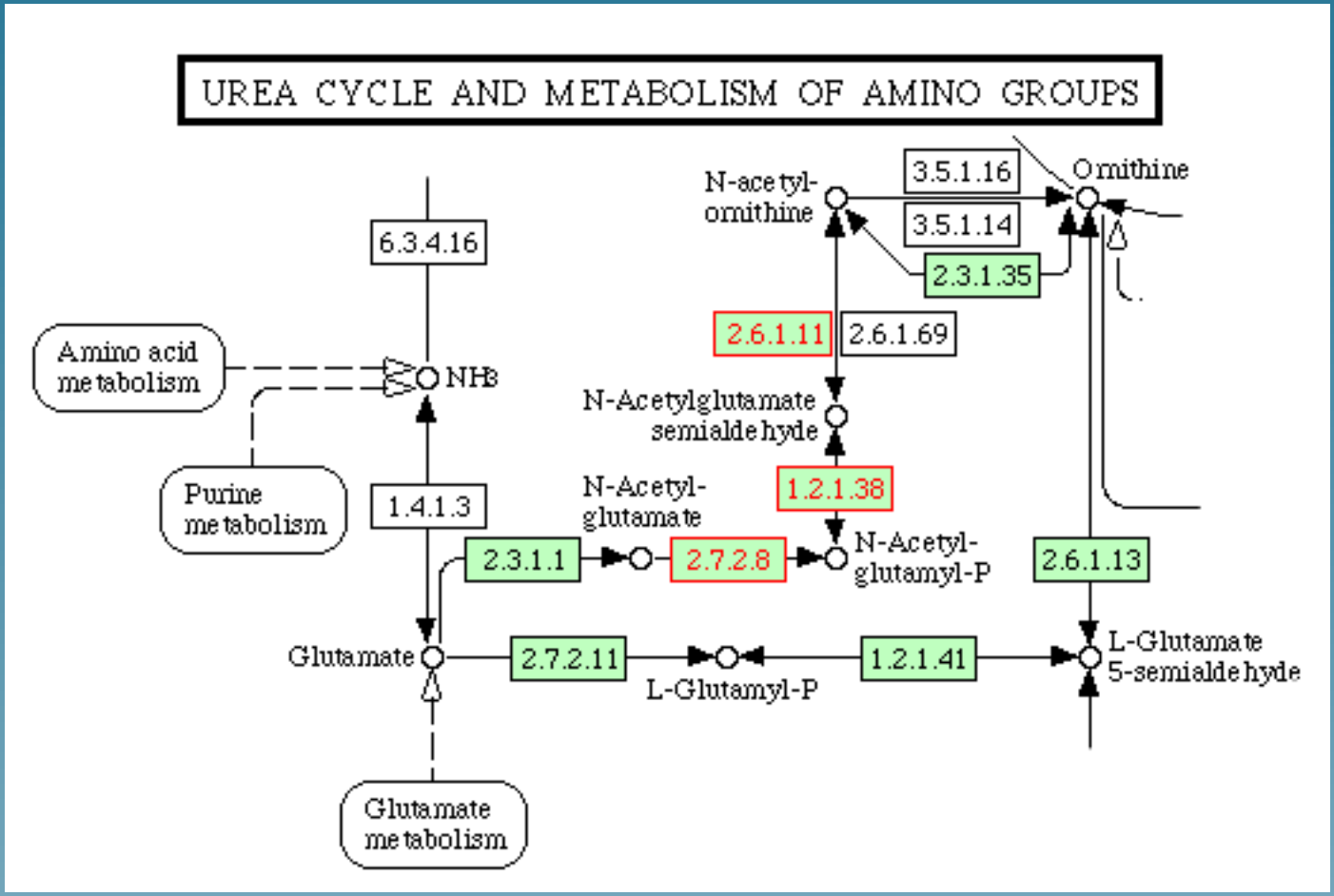
Eukaryotic Pol I

A2	A12	A14	A34	A43	A49
A1					

Lower right genes



Lower right genes



Why it works (*advanced*)

- The diffusion kernel K_1 induces a reproducible Kernel Hilbert space of real-valued functions on genes whose norm $||f||_{\mathcal{H}_1}$ is a smoothing functional
- The linear kernel K_2 induces a RKHS whose norm $||f||_{\mathcal{H}_2}$ is a relevance functional
- The CCA algorithm extract features f_1 and f_2 which maximize a trade-off between correlation and smoothness / relevance:

$$\max_{(f_1,f_2)\in\mathcal{H}_1\times\mathcal{H}_2} \frac{f_1' f_2}{\sqrt{f_1' f_1 + \delta||f_1||_{\mathcal{H}_1}}\sqrt{f_2' f_2 + \delta||f_2||_{\mathcal{H}_2}}}$$

Conclusion

Conclusion

- We saw yesterday that SVM can be used as replacement of other methods and give good results in real-world applications
- We saw today that SVM can be adapted much more general situations:
  - ⋆ by engineering ingenious kernels
  - ⋆ by using various kernel methods
- This research is still in its infancy!