Kernel Methods in Bioinformatics

Jean-Philippe Vert

Kanehisa Lab, Bioinformatics Center,
Kyoto University, Japan

Jean-Philippe.Vert@mines.org

Kanehisa Colloquium, Kyoto University, April 22, 2002.

Outline

1. What is a kernel?
2. What you can do with a kernel.
3. Making kernels.
4. Kernelizing the proteome.

Part 1

What is a kernel

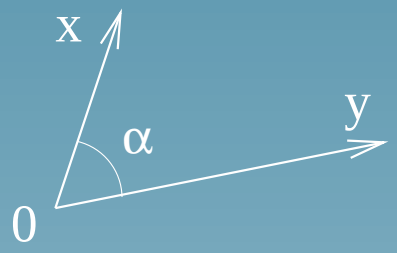
Remember the dot product?

For two vectors:

$$\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} \text{ and } \vec{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix},$$

the dot product is:

$$\vec{x}.\vec{y} = x_1.y_1 + \ldots + x_m.y_m = \|x\|.\|y\|.cos(\alpha)$$



A simple kernel for genes

- Consider a set of genes $g_1, g_2, \ldots, g_N$. Represent each gene g_i by its nucleotide composition (a vector $\vec{\Phi}(g_i)$ with 4 entries):

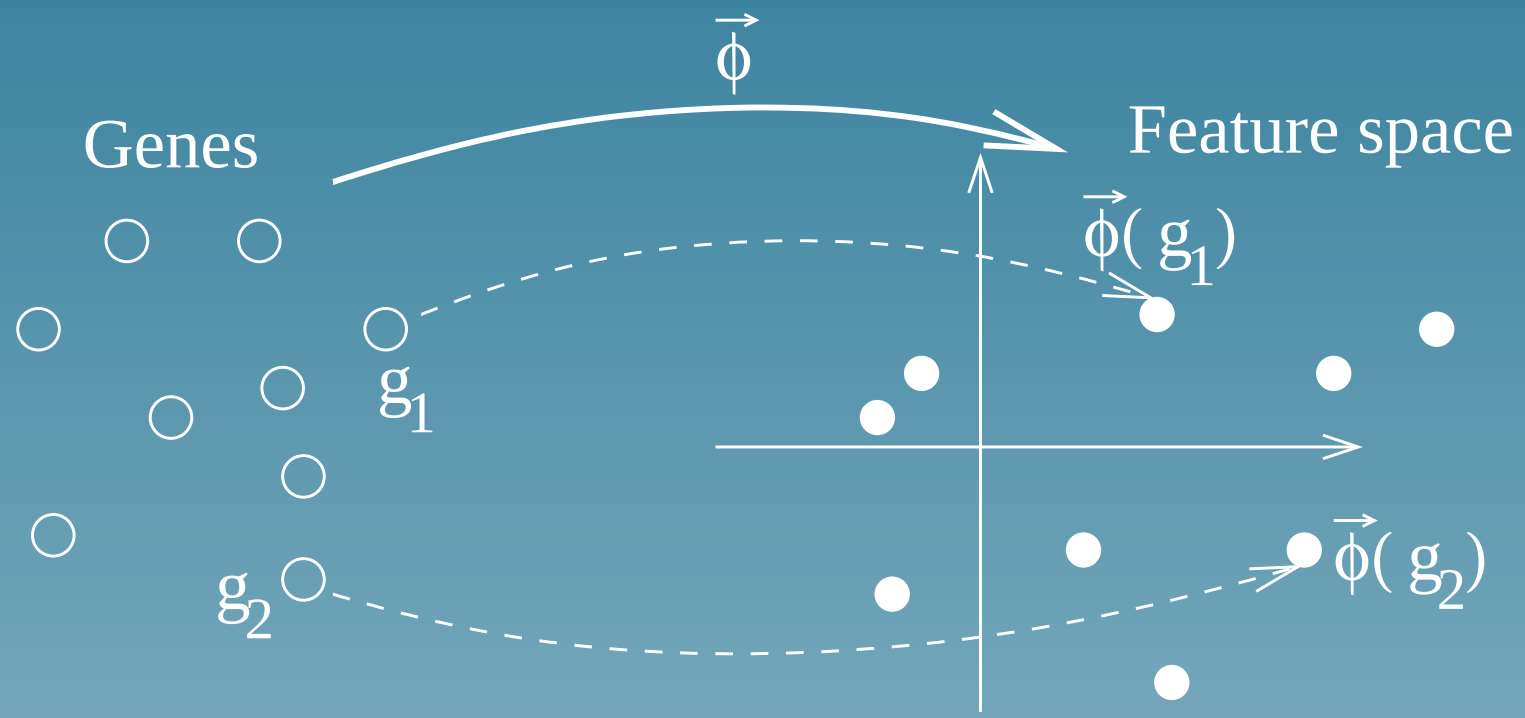
$$\vec{\Phi}(g_1) = \begin{pmatrix} 0.2 \\ 0.3 \\ 0.4 \\ 0.1 \end{pmatrix}, \vec{\Phi}(g_2) = \begin{pmatrix} 0.1 \\ 0.7 \\ 0.1 \\ 0.1 \end{pmatrix}, \ldots$$

- My first kernel:

$$K(g_1, g_2) = 0.2 \times 0.1 + 0.3 \times 0.7 + 0.4 \times 0.1 + 0.1 \times 0.1 = 0.28$$

General definition

$$K(g_i, g_j) \stackrel{def}{=} \vec{\Phi}(g_i).\vec{\Phi}(g_j)$$



You use kernels everyday!

Suppose you give me a function $K(g_i, g_j)$ which "measures" the similarity between genes in some sense (example: Smith-Waterman score). Is it a kernel?

Theorem 1. [Mercer] *It is a kernel if the following matrix is symetric positive definite (all eigenvalues are positive):*

$$K = \begin{pmatrix} K(g_1, g_1) & K(g_1, g_2) & \dots \\ K(g_2, g_1) & K(g_2, g_2) & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}$$

Summary

- A kernel is a similarity measure
- It defines the geometry of the feature space (lenghts and angles)
- 3 ways to make kernels:
  - ⋆ Define a set of features of interest, compute the feature vector of every gene, and compute the dot products.
  - ⋆ Define a large set of features and find tricks to compute the dot product implicitely (without computing the feature vectors)
  - ⋆ Start with a similarity measure you find pertinent (e.g., SW score) and check that it is a kernel.

Part 2

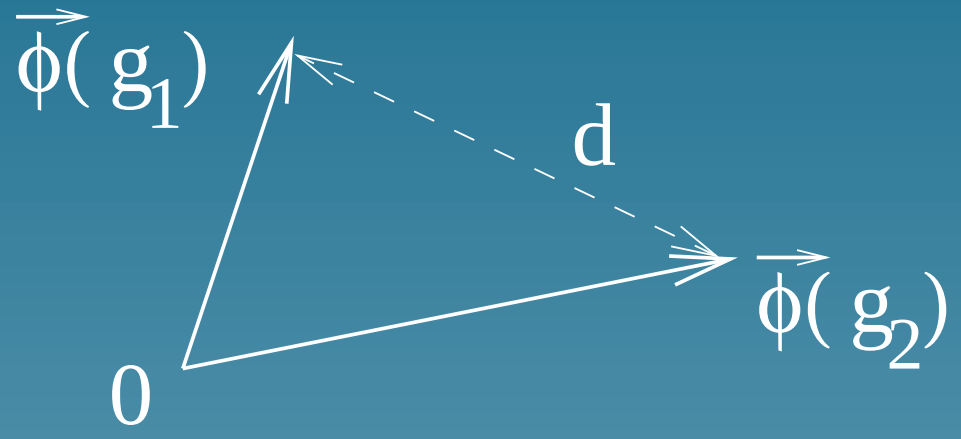
What you can do with a kernel

Overview

Suppose you are given a kernel $K(.,.)$. Then you can perform various operations in the feature space without computing the image $\vec{\Phi}(g)$ 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)

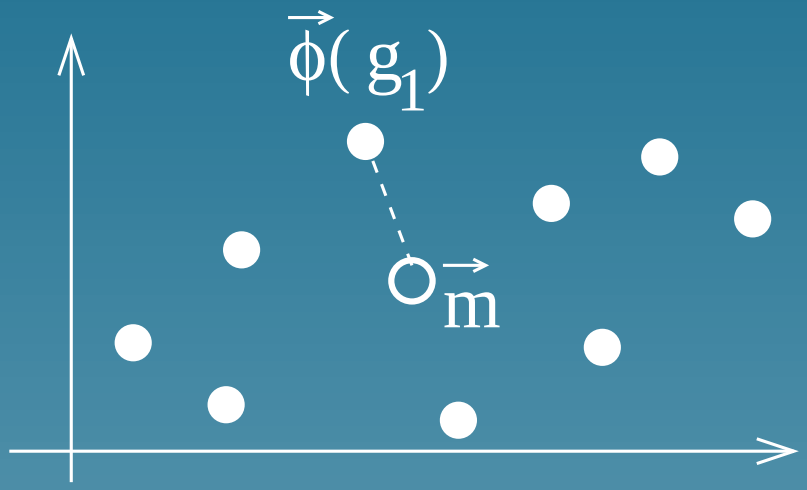
Distance between two genes



$$
\begin{aligned}
d(g_1, g_2)^2 &= \|\vec{\Phi}(g_1) - \vec{\Phi}(g_2)\|^2 \\
&= \left(\vec{\Phi}(g_1) - \vec{\Phi}(g_2)\right) . \left(\vec{\Phi}(g_1) - \vec{\Phi}(g_2)\right) \\
&= \vec{\Phi}(g_1).\vec{\Phi}(g_1) + \vec{\Phi}(g_2).\vec{\Phi}(g_2) - 2\vec{\Phi}(g_1).\vec{\Phi}(g_2)
\end{aligned}
$$

$$
d(g_1, g_2)^2 = K(g_1, g_1) + K(g_2, g_2) - 2K(g_1, g_2)
$$

Distance between a gene and the center of mass



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

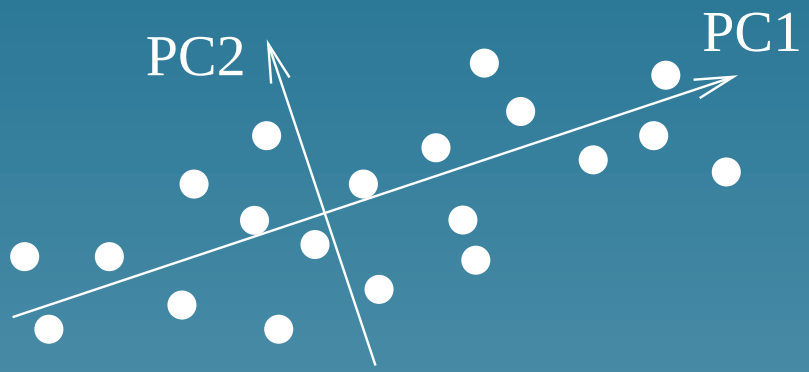
$$\|\vec{\Phi}(g_1) - \vec{m}\|^2 = \vec{\Phi}(g_1).\vec{\Phi}(g_1) - 2\vec{\Phi}(g_1).\vec{m} + \vec{m}.\vec{m}$$

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

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)

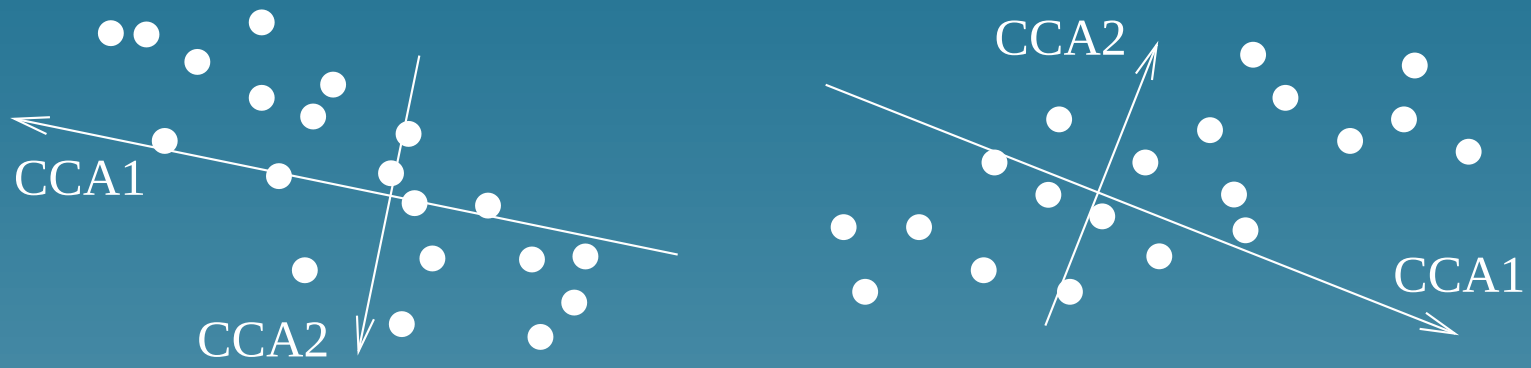


Find the eigenvectors of the matrix:

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

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

Canonical correlation analysis (CCA)

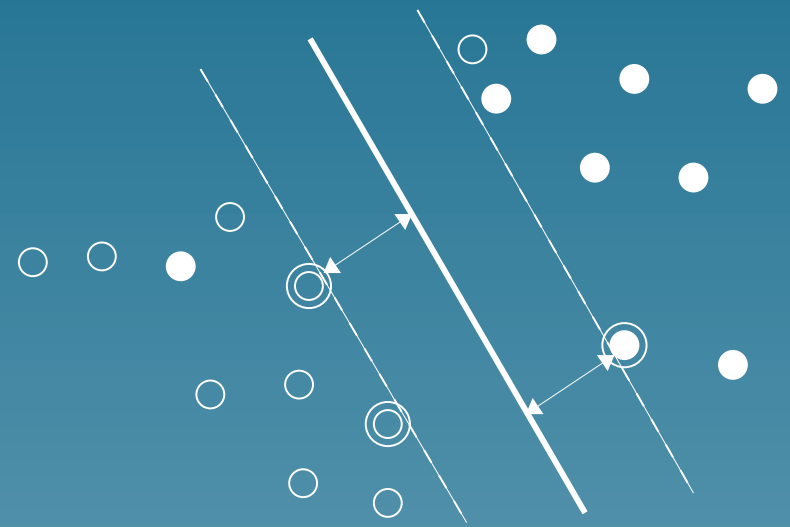


K_1 and K_2 are two different kernels for the same objects (genes). CCA is performed by solving the generalized eigenvalue problem:

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

Useful to find correlations between different representations of the same objects

Classification: support vector machines (SVM)



Find a linear boundary with maximum margin by solving:

$$\begin{cases} \max_{\vec{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(g_i, g_j) \\ \forall i = 1, \ldots, n \quad 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

Examples: SVM in bioinformatics

- Gene functional classification from microarry: Brown et al. (2000), Pavlidis et al. (2001)
- Tissue classification from microarray: Mukherje et al. (1999), Furey et al. (2000), Guyon et al. (2001)
- Protein family prediction from sequence: Jaakkoola et al. (1998)
- Protein secondary structure prediction: Hua et al. (2001)
- Protein subcellular localization prediction from sequence: Hua et al. (2001)

Summary

- If you have a kernel, you can do many things implicitely in the feature space.
- Methods such as SVMs are very efficient in real-world applications
- You can use any kernel with any method
- Gains popularity in bionformatics, but much remains to be done (up to now, limited to SVMs with classical kernels)

Part 3

Making kernels

Overview

- Why make kernels?
- Kernel for strings based on rare common substrings
- Kernel for phylogenetic profiles
- Making a kernel from a graph

Why make kernels

- To include biological knowledge in the feature space, e.g.: "two genes should be close if...":
  - ⋆ their sequences are similar,
  - ⋆ their evolutions are similar,
  - ⋆ their expression patterns are similar...
- To be able to use powerful kernel methods based on these knowledges

Kernel for strings (PSB02)

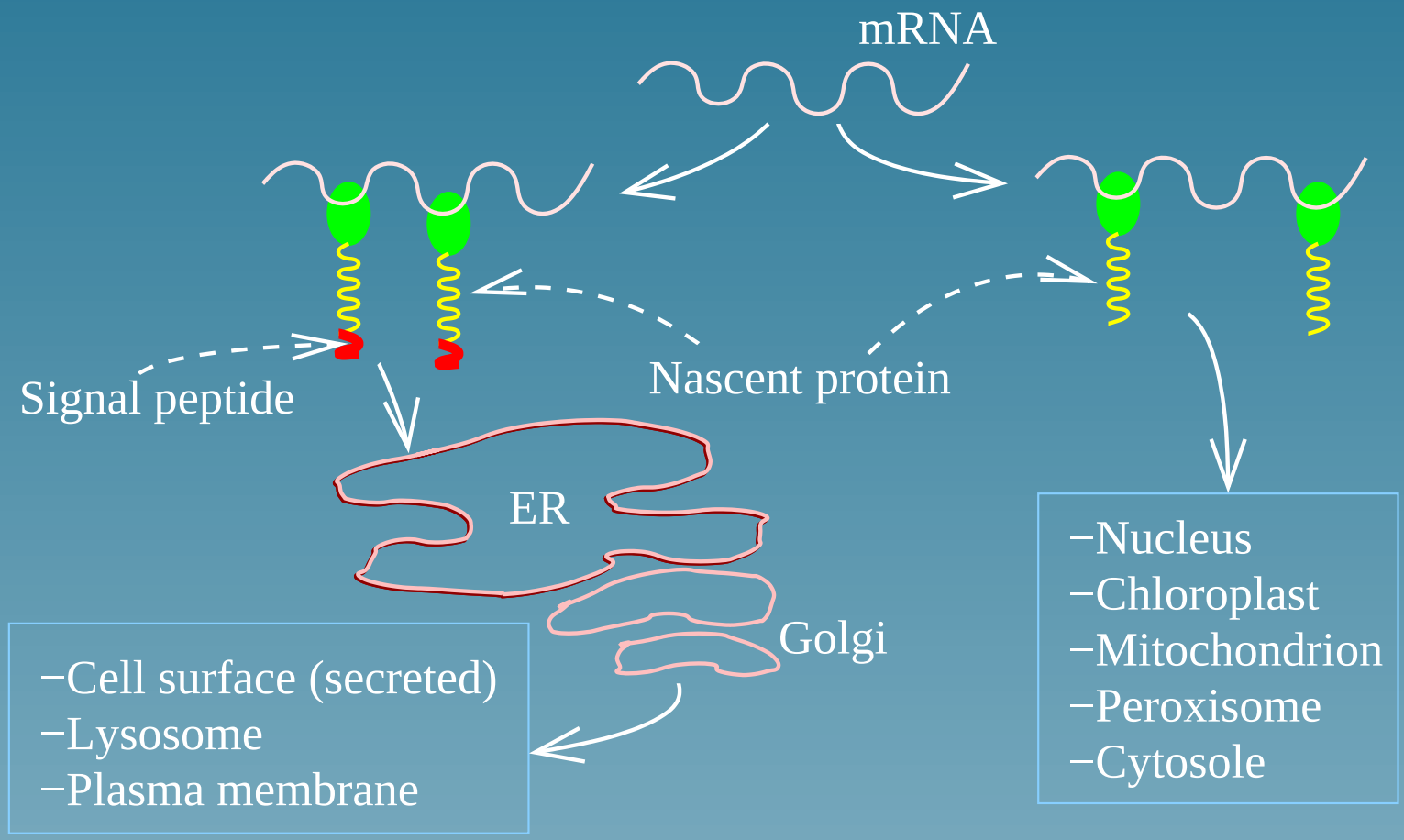
- Goal: a kernel for fixed-length strings (sequence windows...)
- Intuition: two strings should get closer in the feature space when they share rare common substrings
- Solution:
  - ⋆ Let p a probability distribution on the set of sequences of length m (e.g., a position specific weight matrix)
  - ⋆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

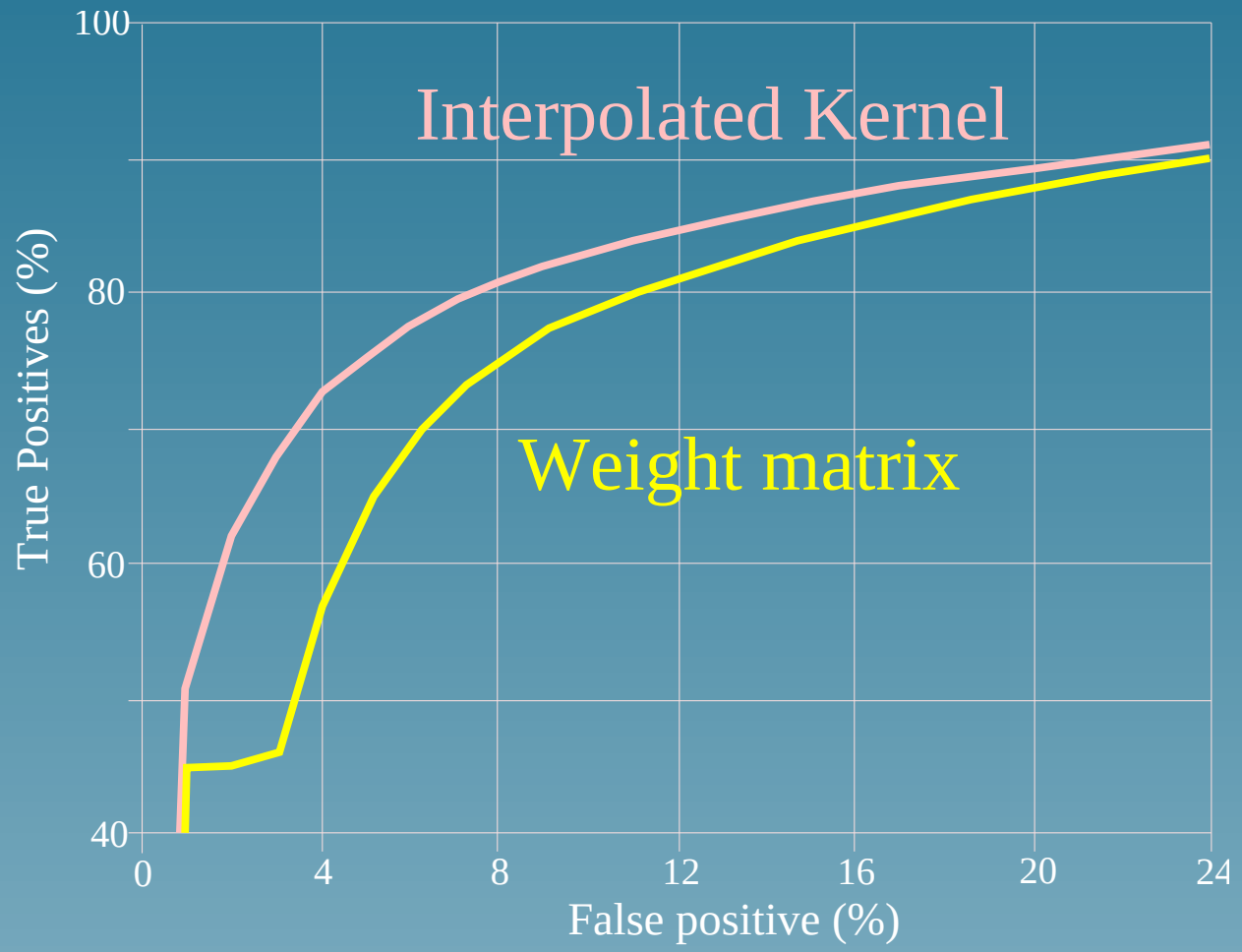
Experiment

- 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

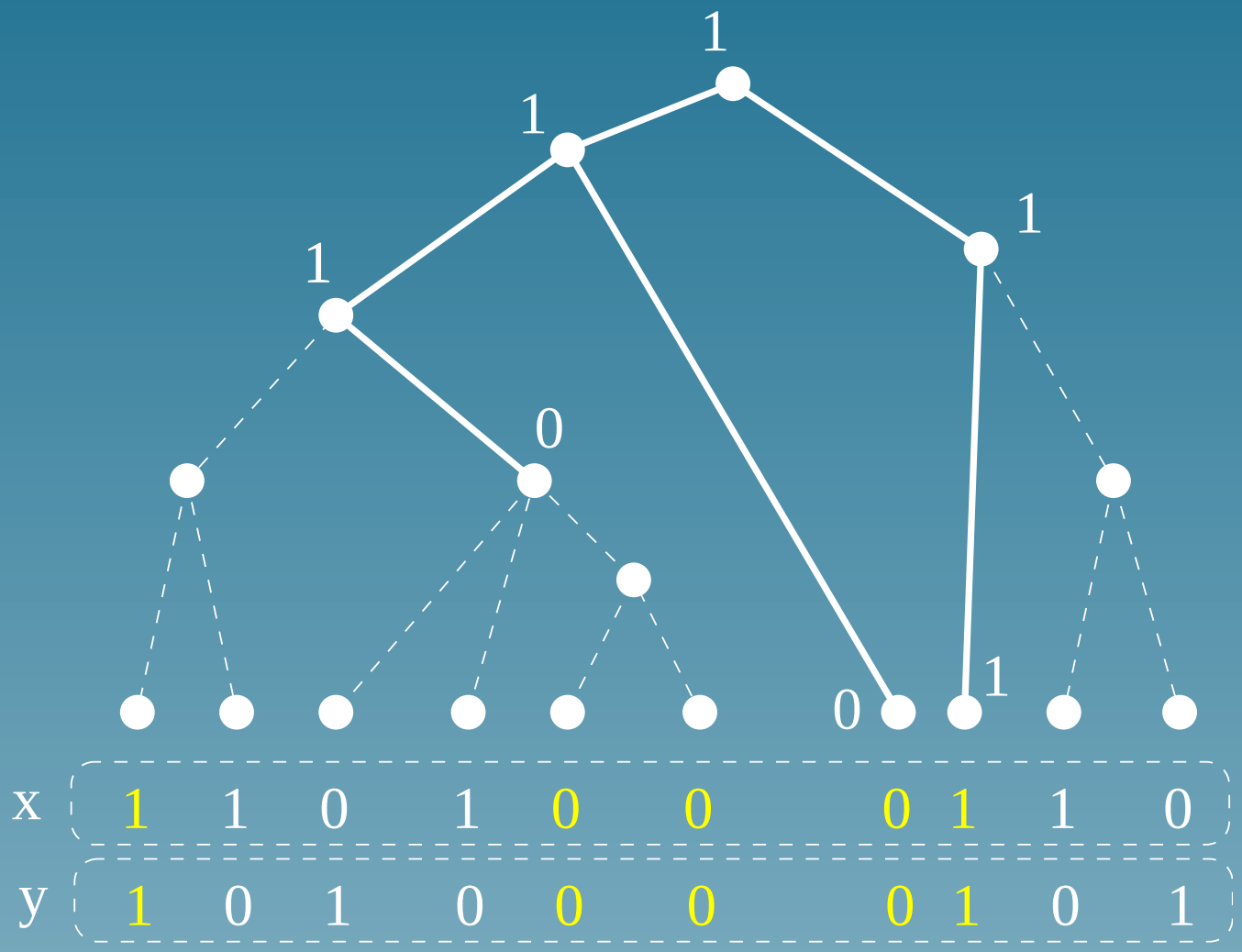


Kernel for phylogenetic profiles (ISMB02)

- 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



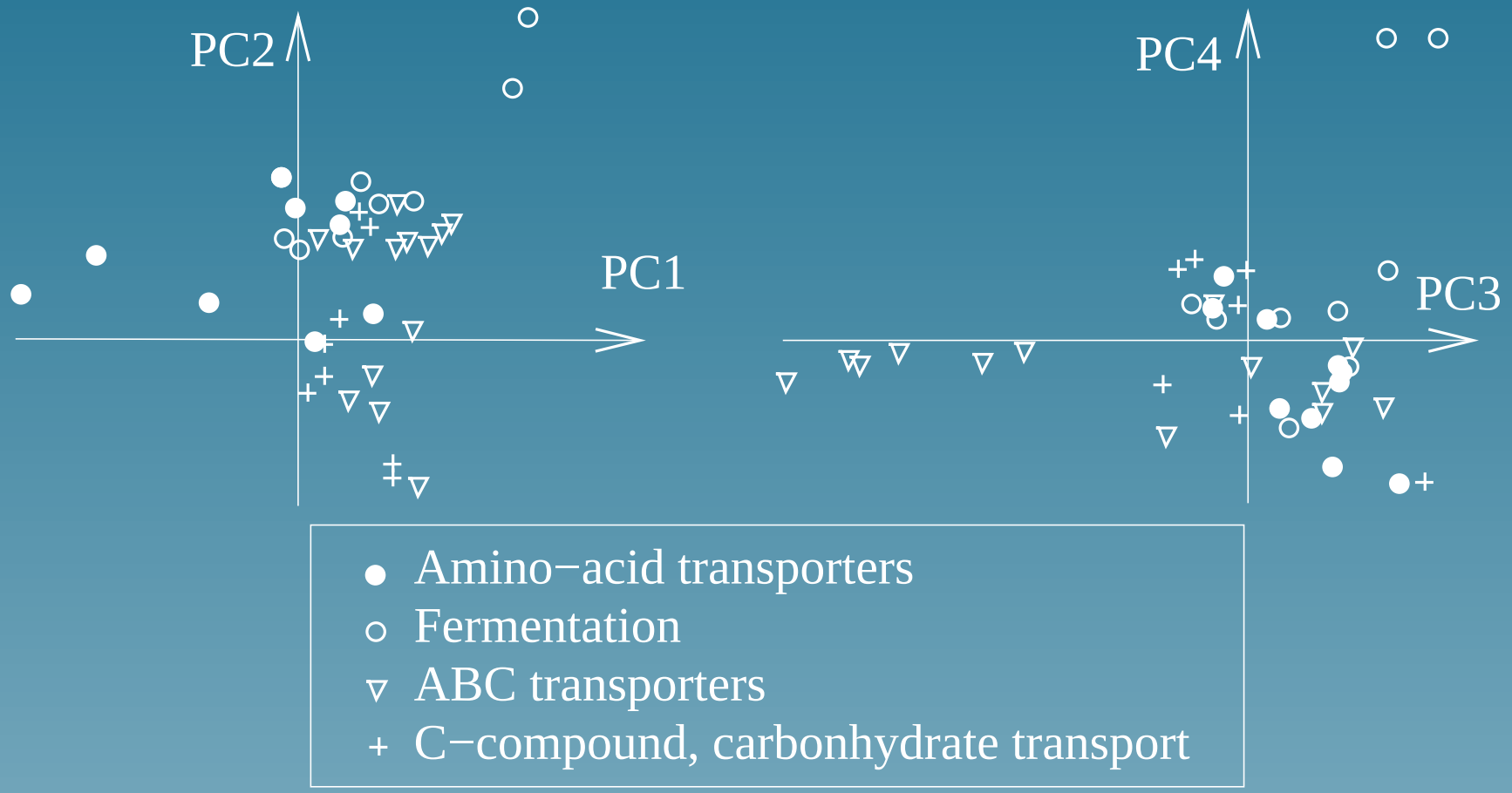
Properties of the tree kernel

- $K(.,.)$ 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

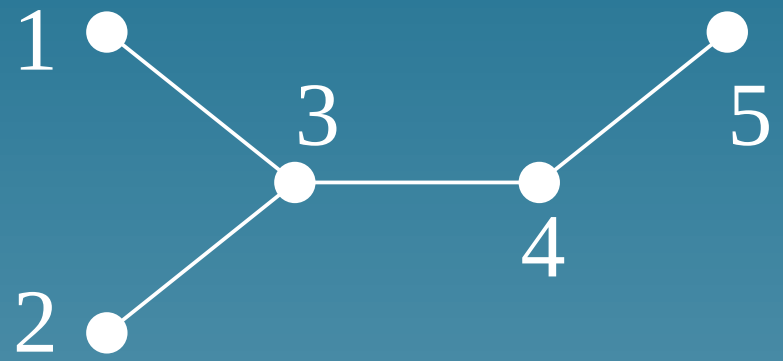


Making a kernel from a graph (Kandor, 2001)

- 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 A be the adjacency matrix (where diagonal terms are adjusted such that the sum of each row be nulle) For any $\lambda > 0$, the kernel matrix is:

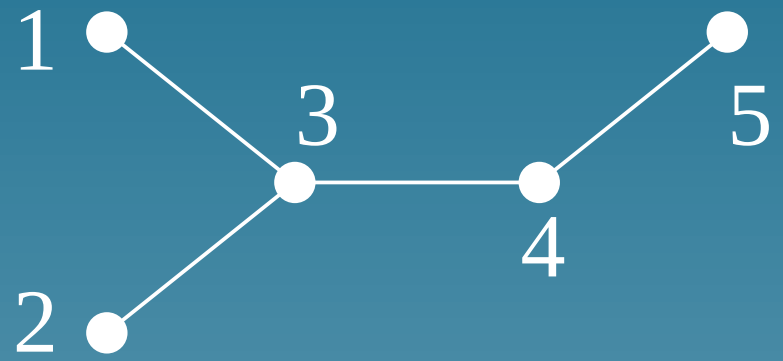
$$K = \exp(\lambda A)$$

Example of a graph kernel (1)



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

Example of a graph kernel (2)



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

Summary: making kernels

- 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 4

Kernelizing the proteome? (tentative)

Motivations

- There is no "universal kernel" for genes
- Different relationships (sequence similarity, function similarity, evolution similarity...) lead to different kernels
- The recent research on kernel engineering makes it possible to translate those relationships into kernels
- New analysis opportunities?

The kernel toolbox

Similarity based on...	Kernel
Aminoacid composition	linear, polynomial, Gaussian...
Sequence	SW, Fisher...
Evolution	Phylogenetic
Pathway (KEGG)	Graph kernel
Interaction (Y2H data...)	Graph kernel
Expression (microarray)	linear, polynomial, Gaussian...

"Any kernel works with any kernel method"

Comparison with KEGG's approach

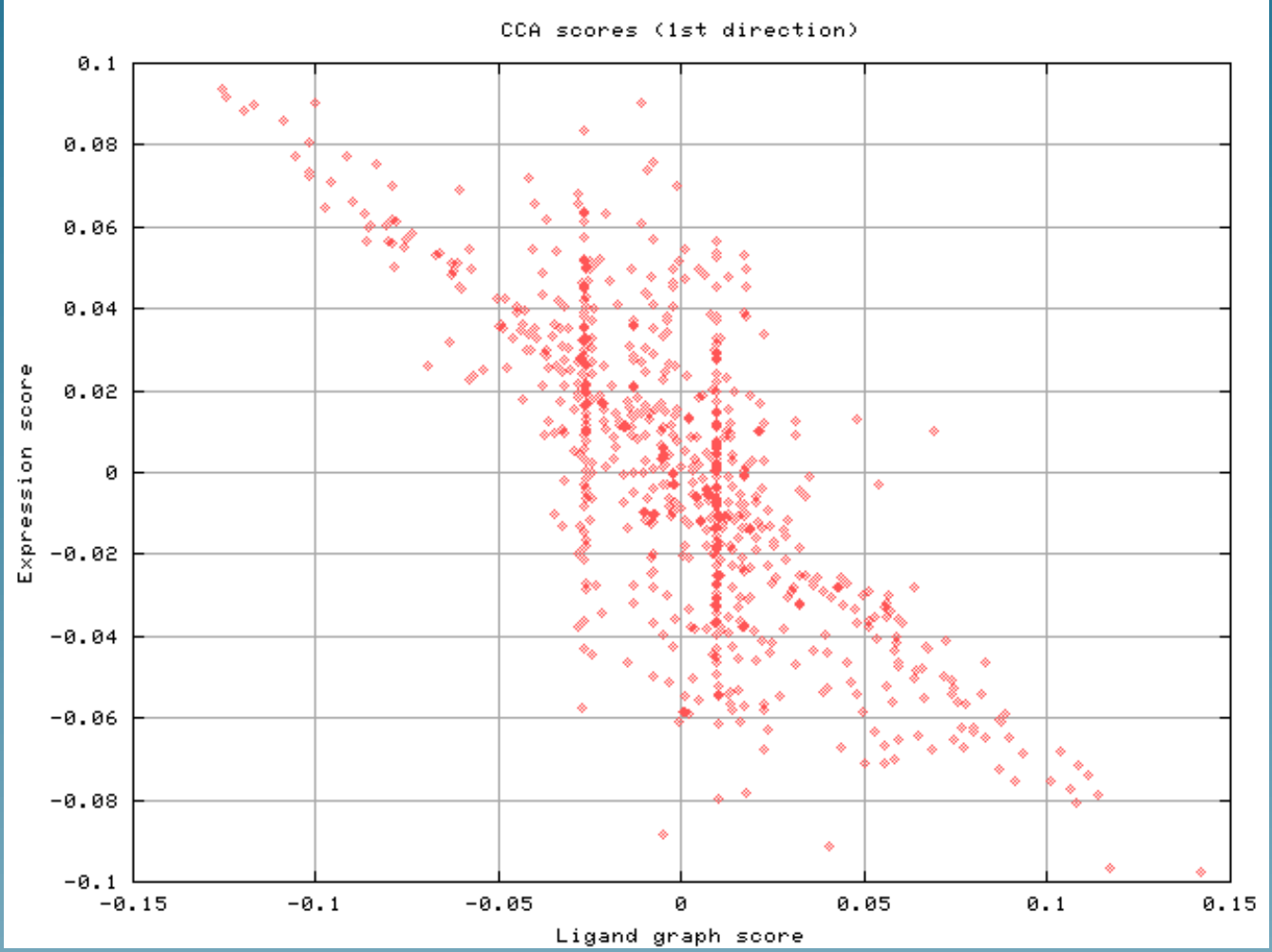
	Kernel approach	Graph theoretical approach
Gene representation	Points in a Euclidean space	Nodes in a graph
Similarity	Euclidean distance	Path length
Computation	Kernel methods	Graph algorithms
Global, noisy statistical analysis	PCA, CCA, SVM...	
Exact computation		CC, clusters...

Some sort of complementarity. Depends on the final goal.

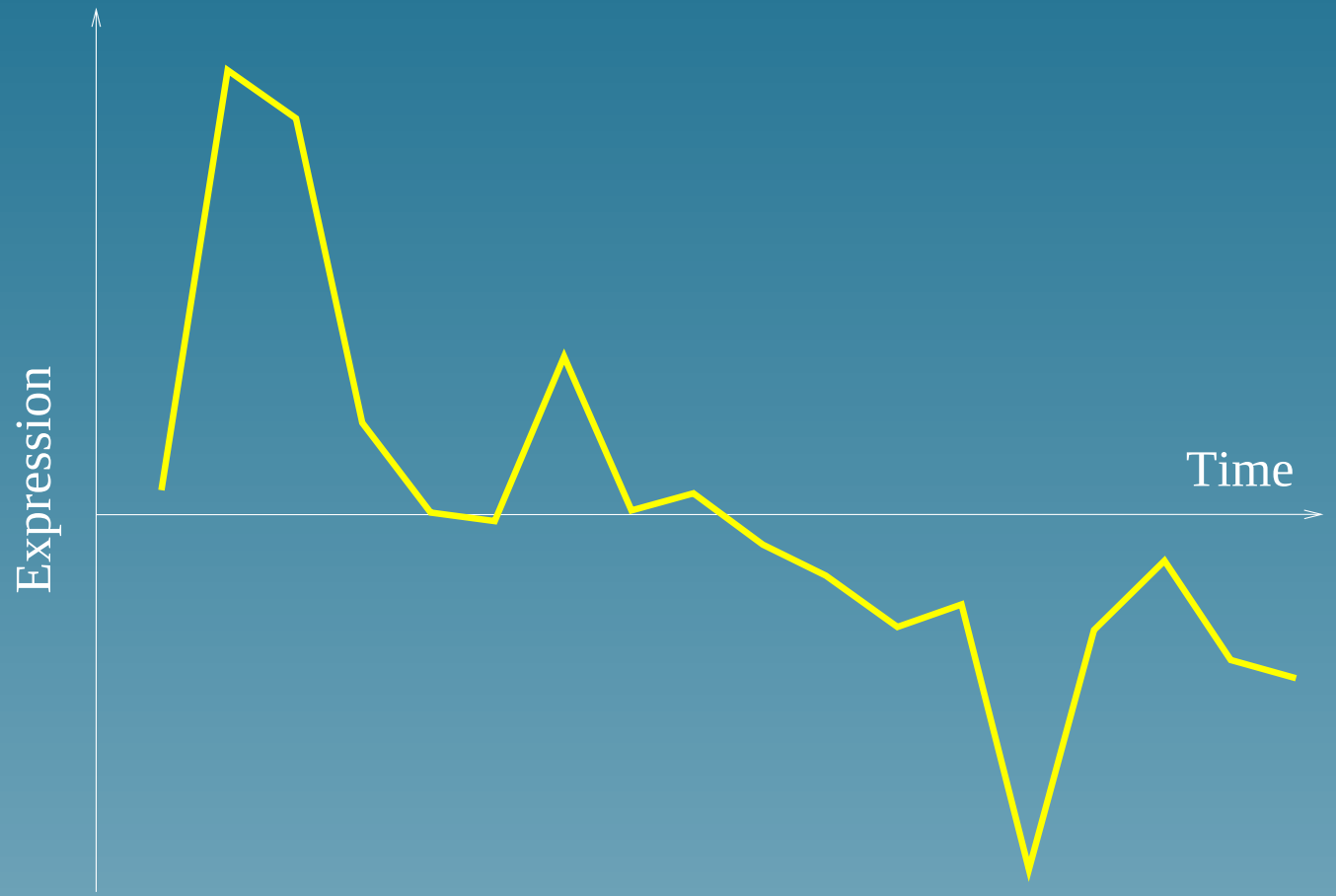
Example: correlations between expression and metabolic pathways

- Expression: Spellman's alpha factor arrest time series data (18 timepoints following removal of alpha factor added 120 minutes earlier). Use a linear kernel K_1 after normalization of the expression profiles.
- Metabolic pathways: KEGG's LIGAND database. Create a graph kernel K_2
- Perform a CCA analysis between K_1 and K_2 (742 common genes)

1st CCA scores



Upper left expression



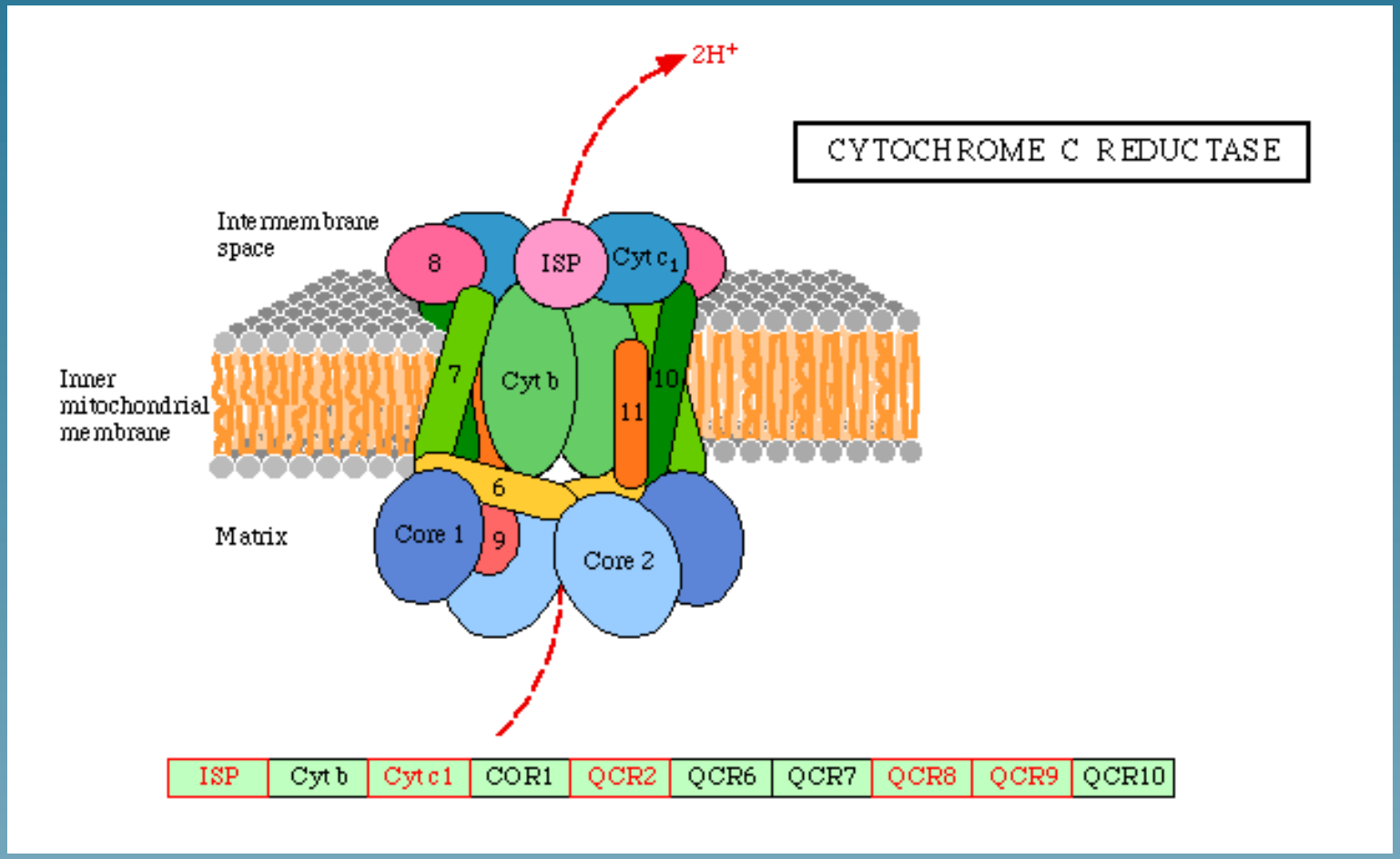
Average expression of the 50 genes with highest $s_2 - s_1$.

Upper left genes

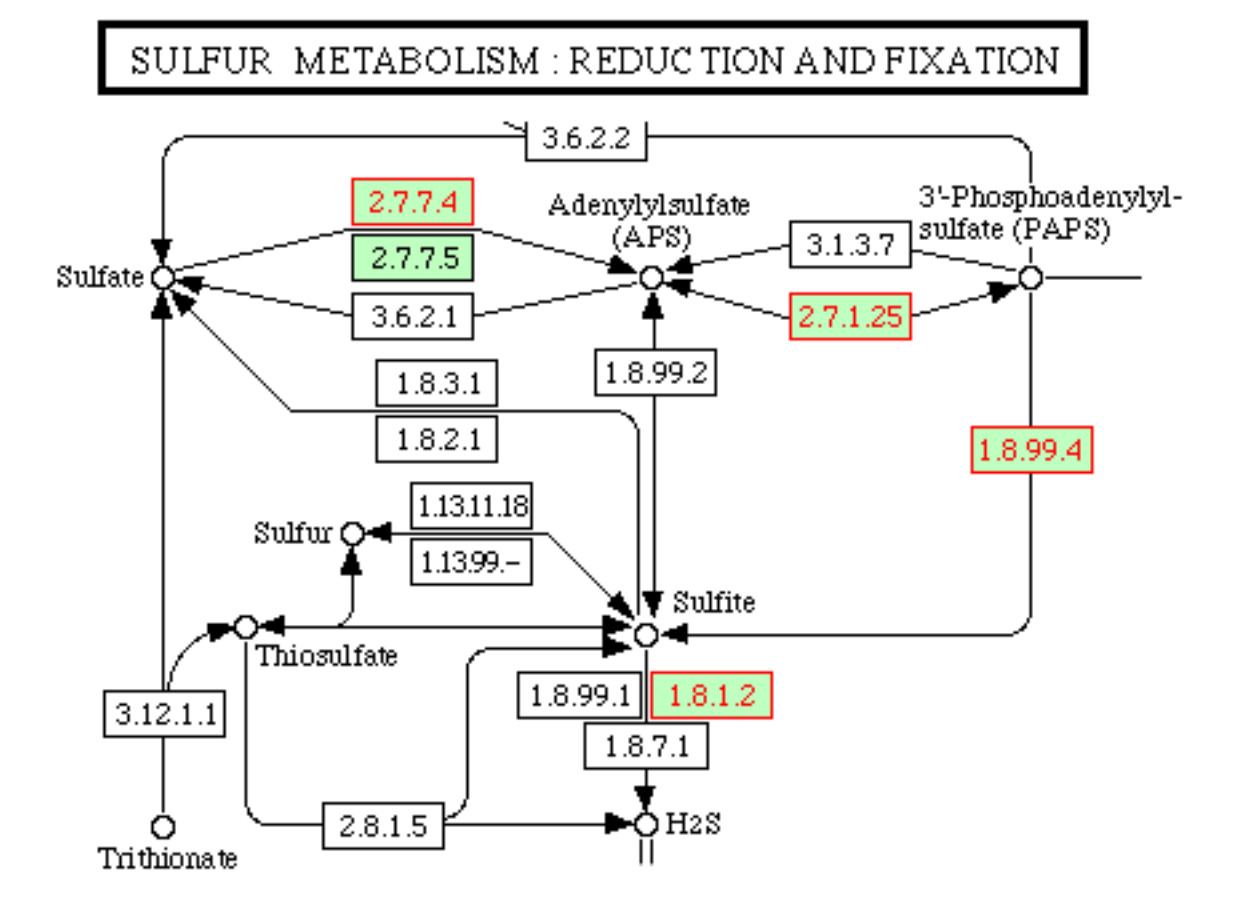
50 genes with highest $s_2 - s_1$ belong to:

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

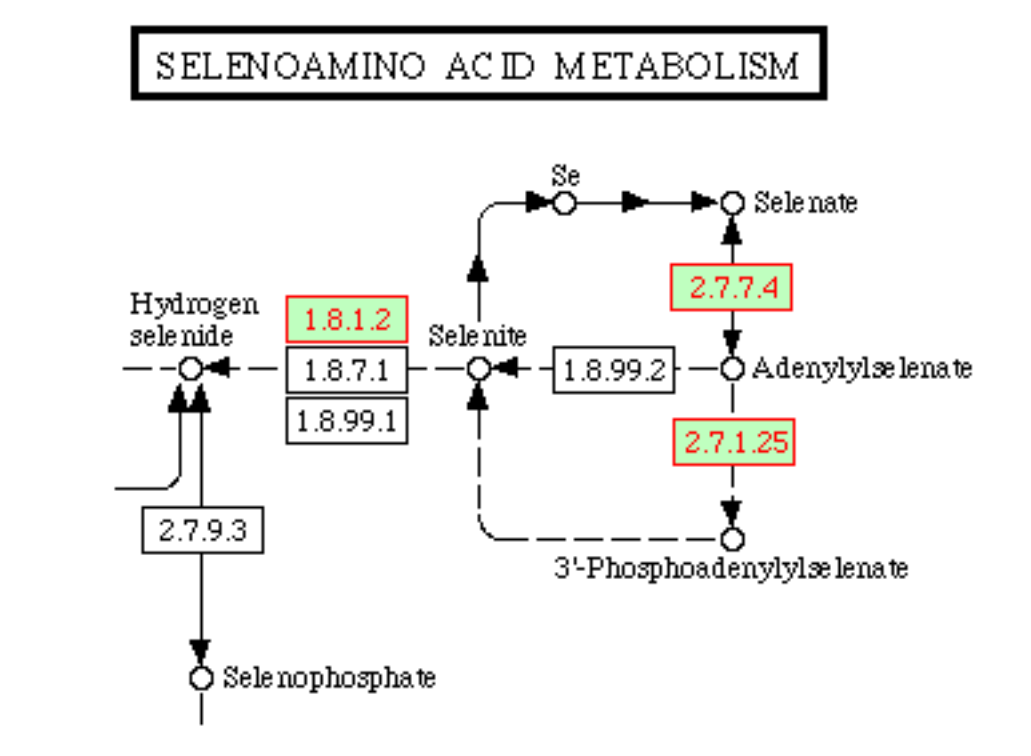
Upper left genes



Upper left genes



Upper left genes



Lower right expression

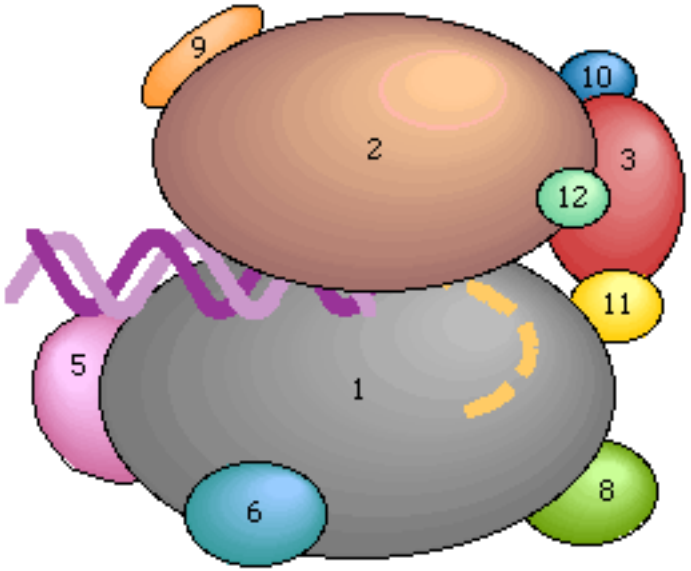
Average expression of the 50 genes with highest $s_2 - s_1$.

Lower right genes

- RNA polymerase (11 genes)
- Pyrimidine metabolism (10)
- Aminoacyl-tRNA biosynthesis (7)
- Urea cycle and metabolism of amino groups (3)
- Oxidative phosphorlation (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

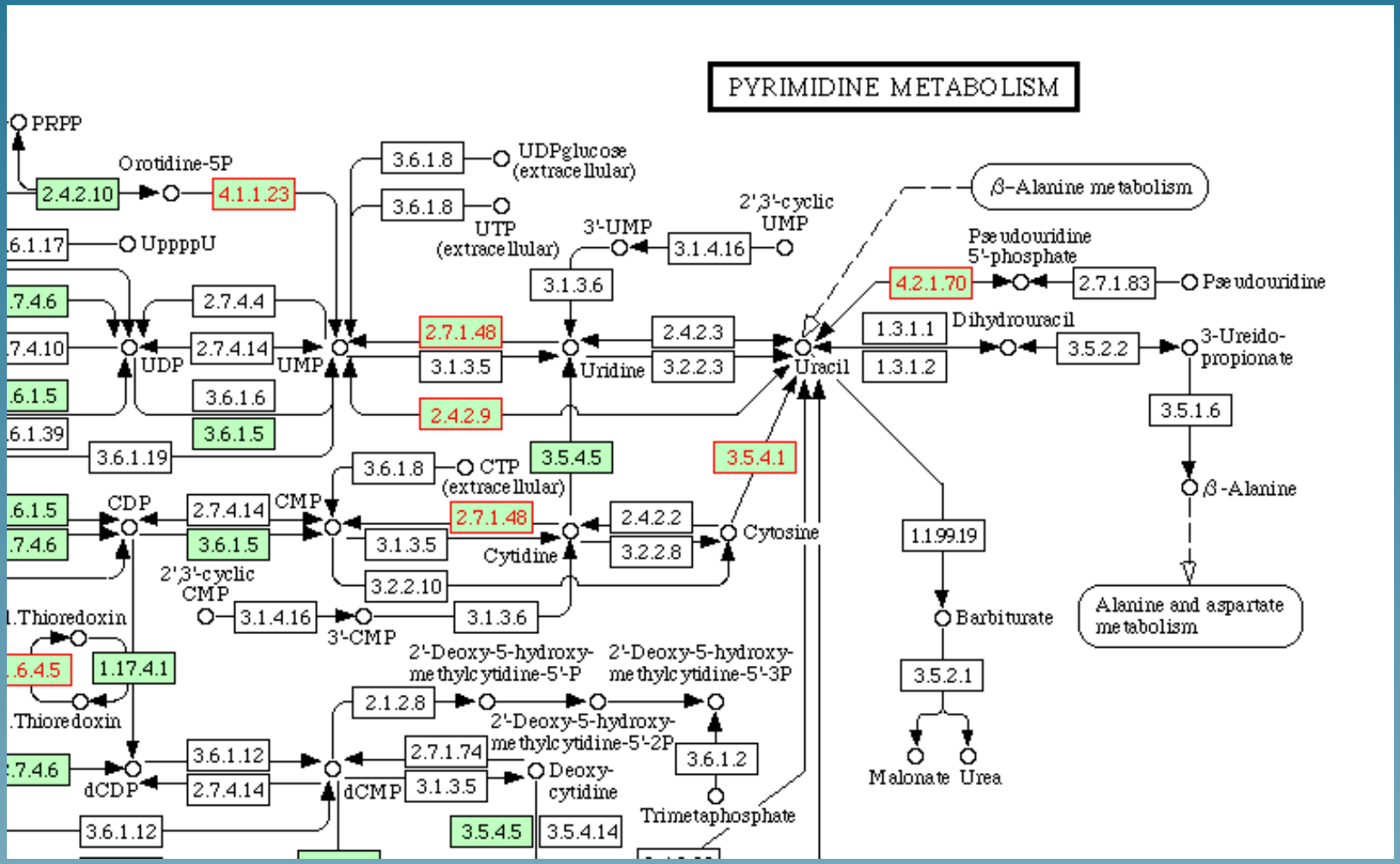
Eukaryotic Pol III

C2	C3	C4	C5	C11
C1	C19	C25	C31	C34

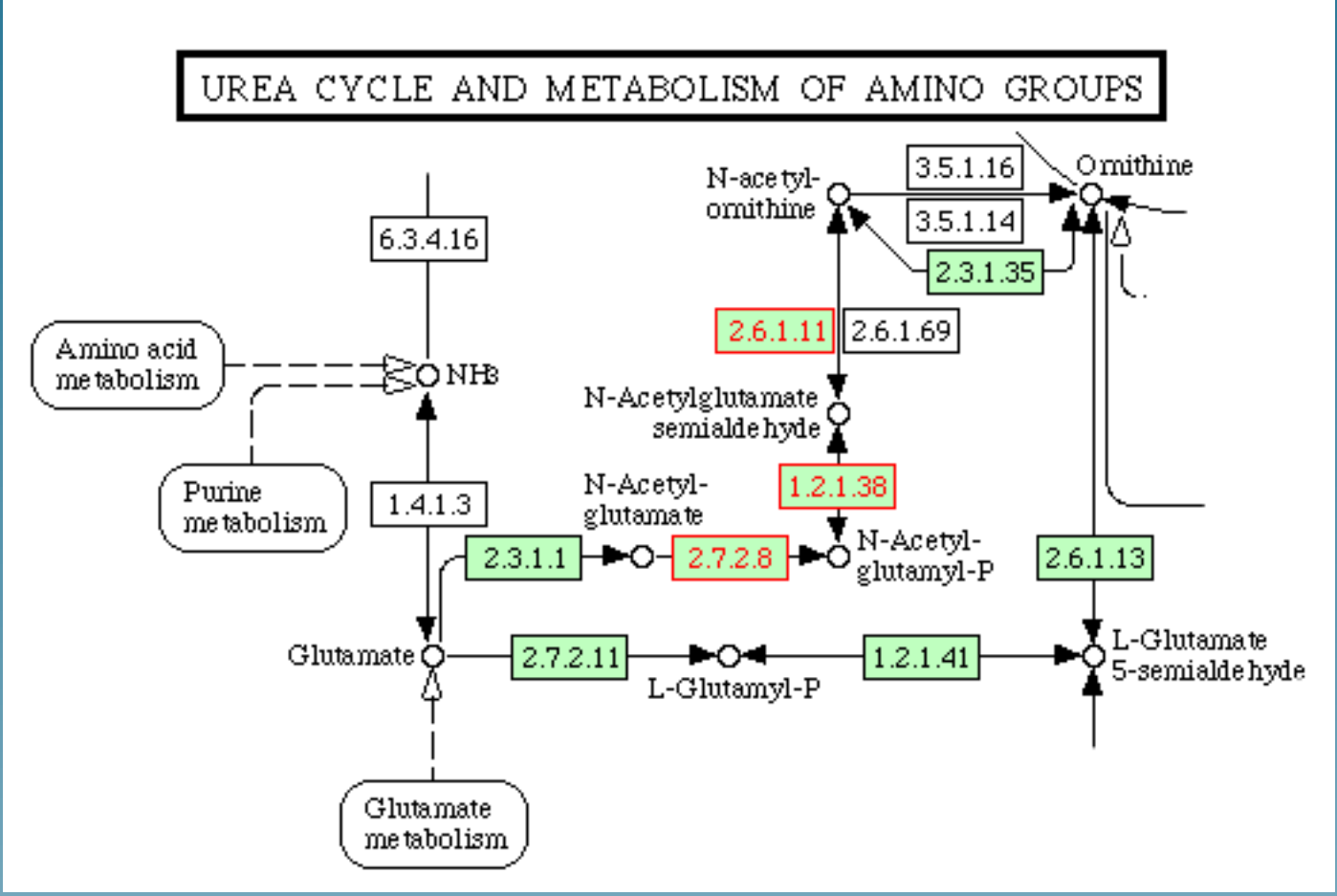
Eukaryotic Pol I

A2	A12	A14	A34	A43	A49
A1					

Lower right genes



Lower right genes



Conclusion

Conclusion

- The kernel approach is one way to represent and handle genes
- Biological knowledge can be included through kernel engineering
- Each kernel can be used by each kernel method (SVM, PCA, CCA,...)
- These methods usually perform well (SVM...), and provide new analysis opportunities (PCA of the metabolic pathways...)
- Much remains to be done!