Noyaux pour séquences biologiques

Cours Master 2005/06

Jean-Philippe Vert

Jean-Philippe.Vert@mines.org

Plan

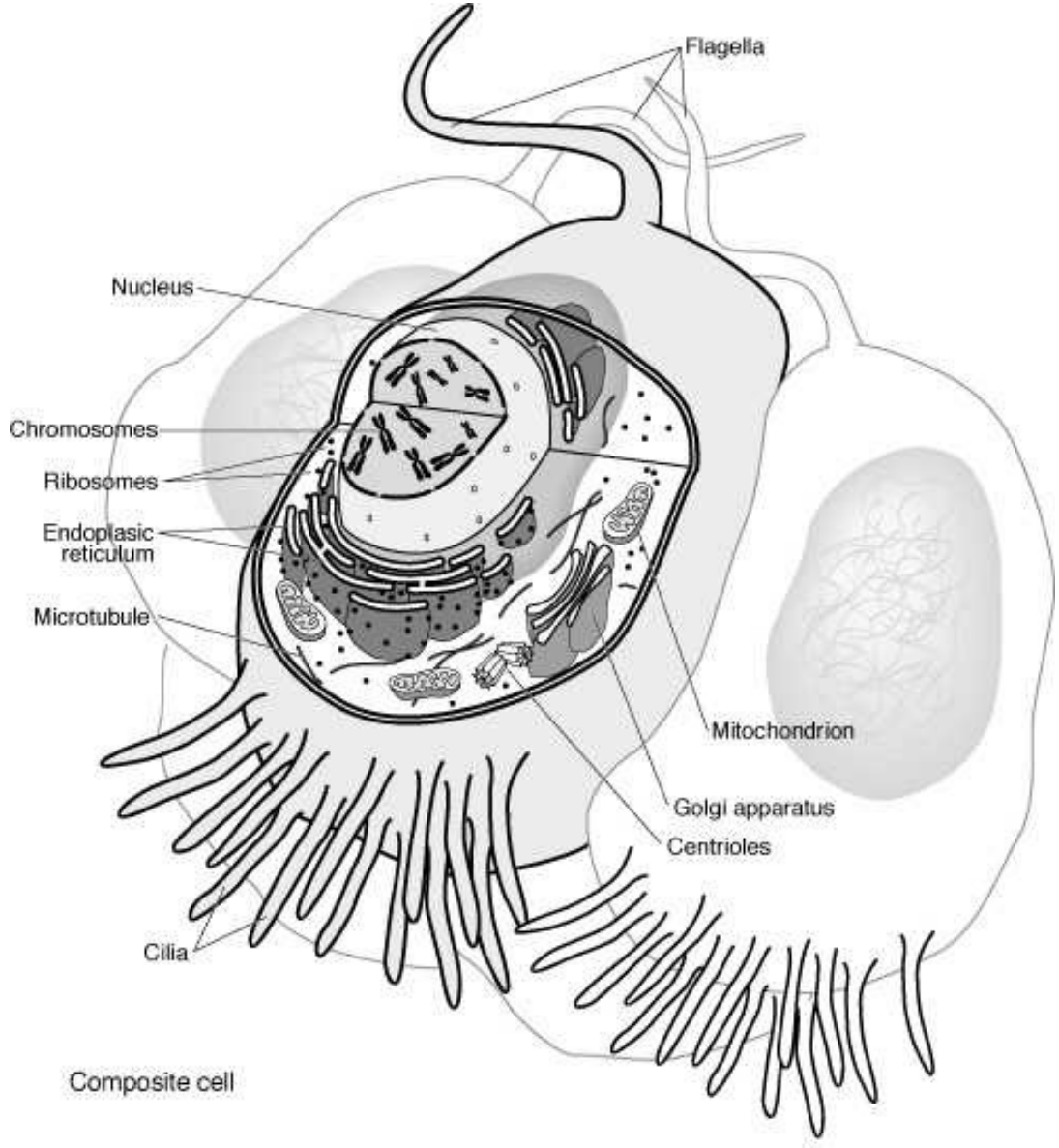
- Introduction à la biologie
- Espace des séquences protéiques
- Noyau spectral
- Noyau de sous-séquences
- Similarités par alignement
- Noyau de convolution
- Application: détection d'homologie lointaine

Introduction à la biologie

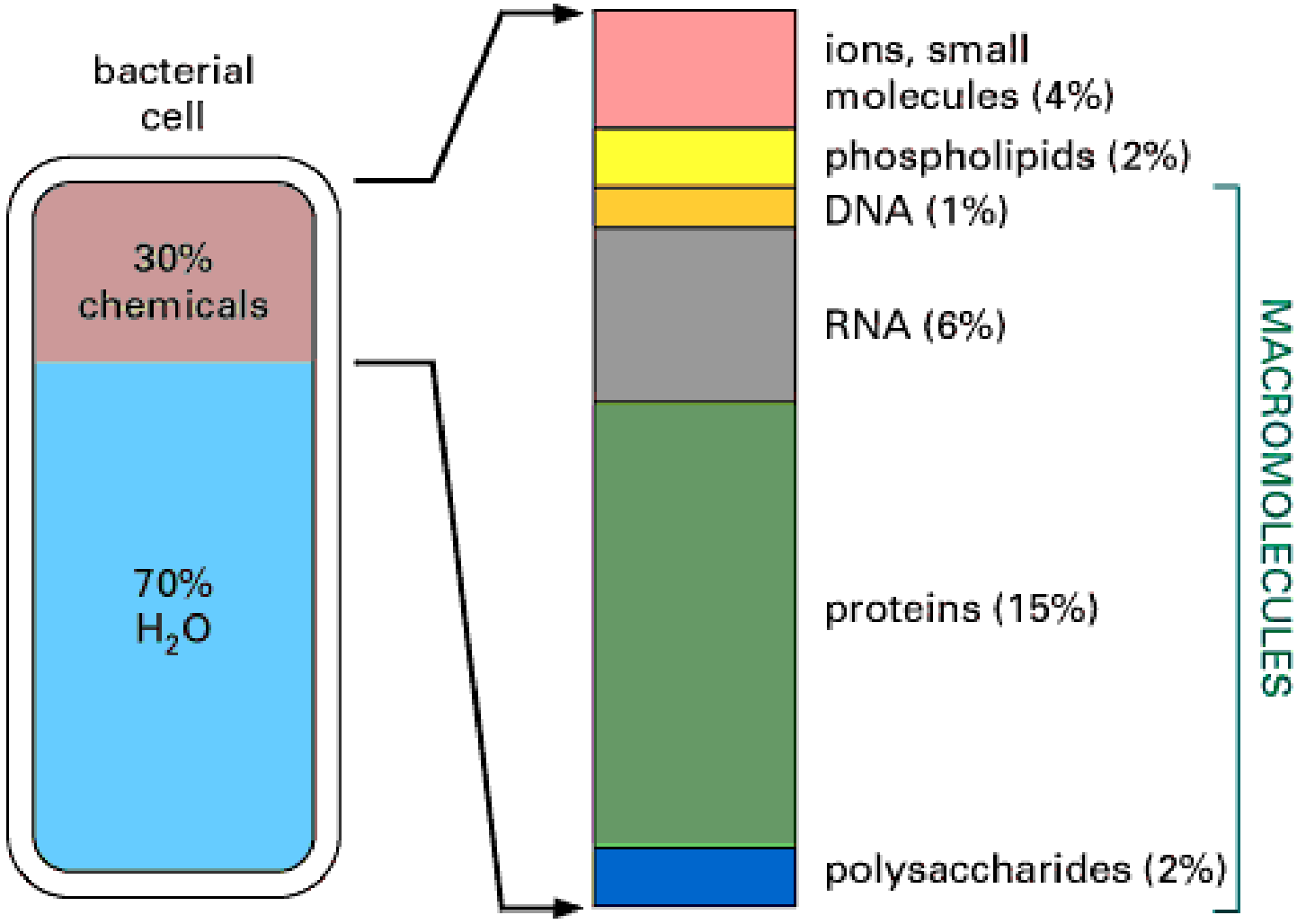
Cellules

- Tout organisme *vivant* est composé de *cellules*
- Une cellule est une solution contenant différentes molécules entourée d'une *membrane*
- Il y a des organismes *unicellulaires* (bactéries, levure...) ou *multicellulaires*.
- Exemple: il y a environ 6×10^{23} cellules dans un humain, de 320 types différents (peau, muscles, neurones...)

Cellules (eucaryote)



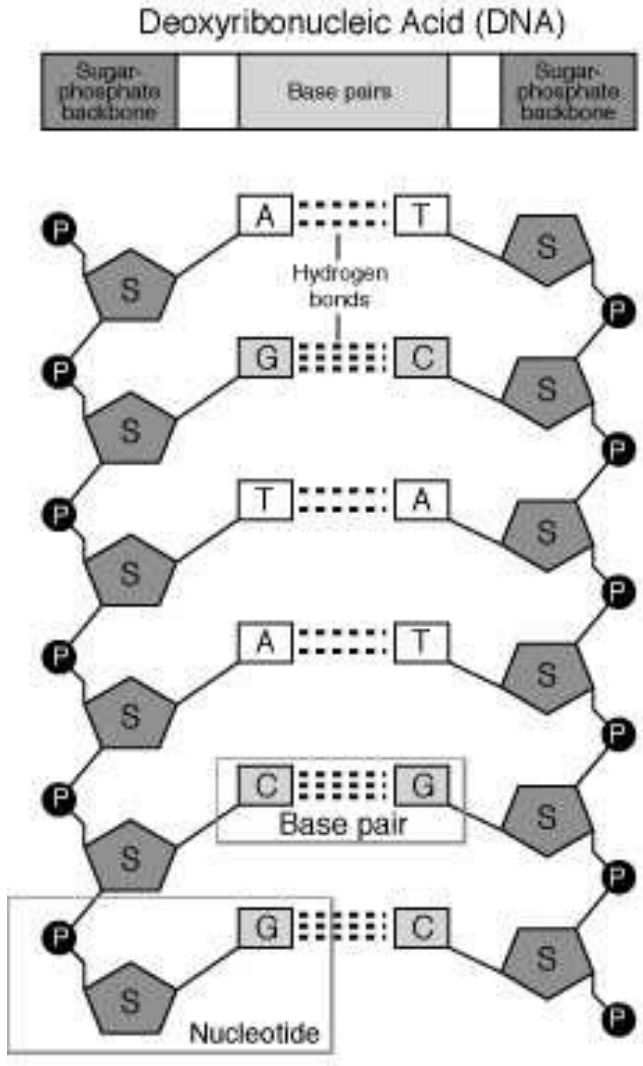
Dans la cellule

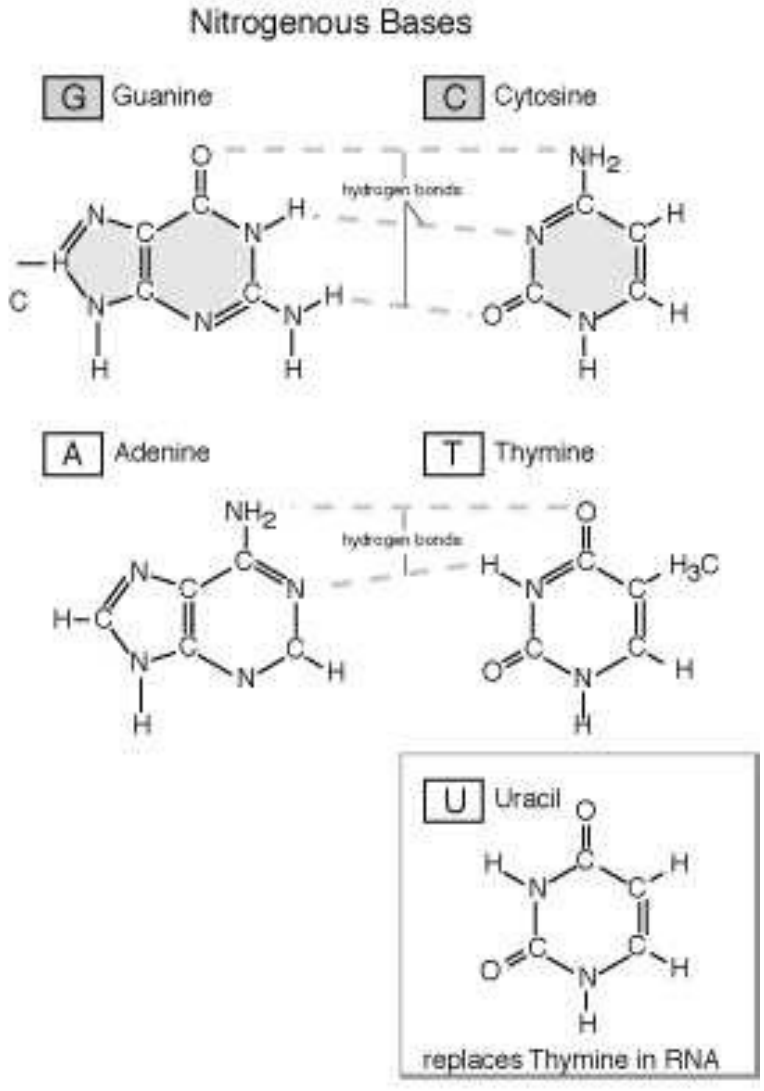


ADN

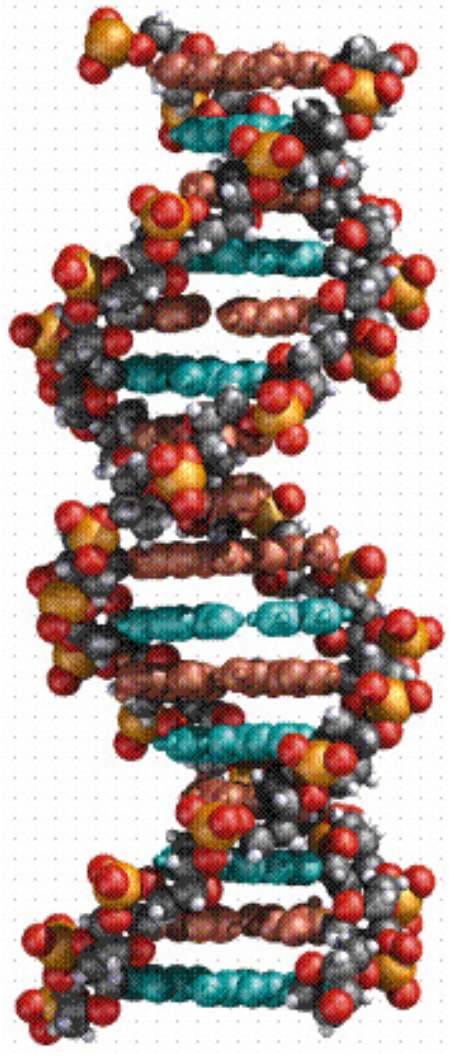
- L'*acide desoxyribonucléique* (ADN) est la molécule, présente dans toutes les cellules, qui contient l'information génétique transmise entre générations.
- L'ADN peut être en *simple brin* ou *double brin*.
- Un brin simple (aussi appelé polynucléotide) est un polymère linéaire composé de 4 *nucléotides*: adénosine (A), cytosine (C), guanine (G) et thymine (T)
- On représente un polynucléotide par une séquence orientée de lettres:
  `5′ -A-T-T-C-A-G-G-C-A-T-T-A-G-C- 3′`

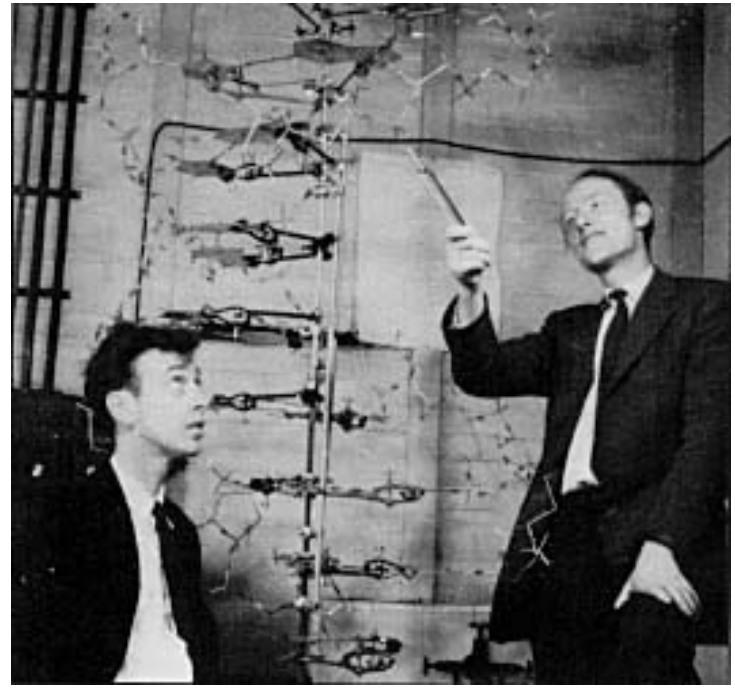
ADN double brin





Structure de l'ADN



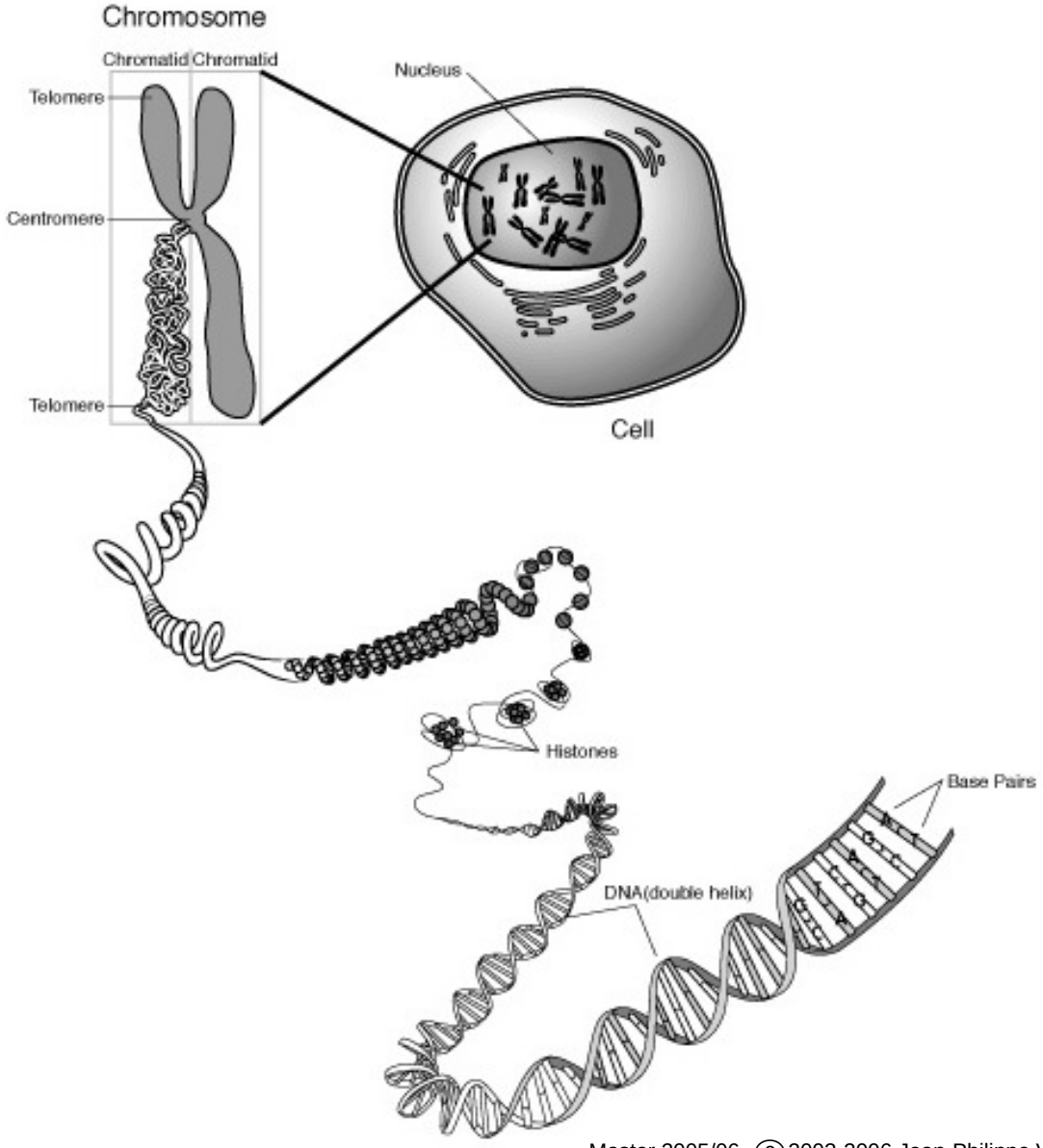


Watson et Crick, 1953.

ADN et information

- La double hélice est stable, quelle que soit la séquence de nucléotides
- Parfait pour stocker 2 bits/base
- Distance entre 2 bases = 0.34nm, donc 6.10^7 bits/cm = *75ko/cm*
- Par repliement de l'ADN en 3D, on peut théoriquement monter à 2.10^{21} *bits/cm*^{3}

ADN et chromosomes



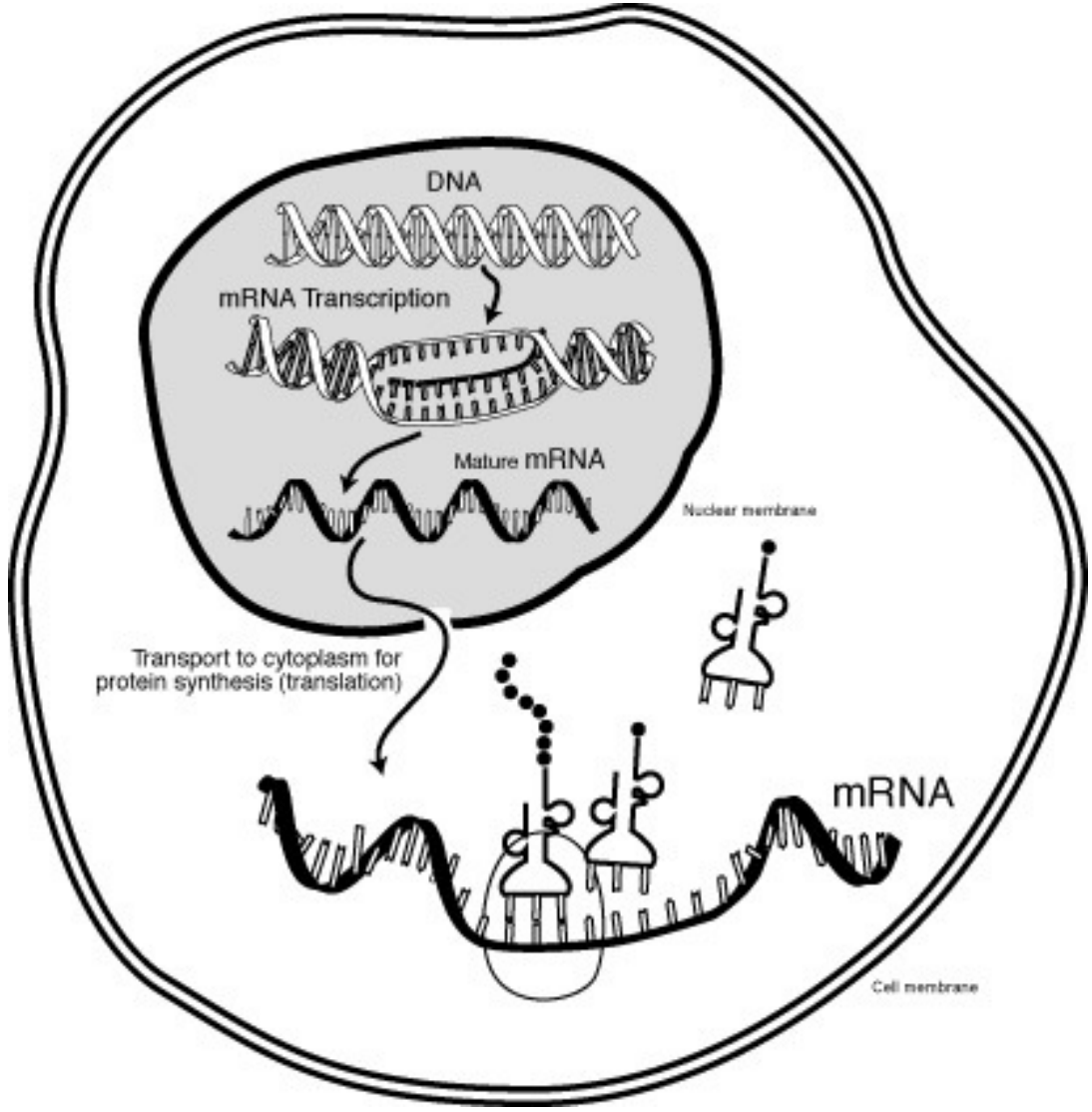
Génome

- Toutes les cellules d'un organisme ont (à peu près) le même ADN, appelé *génome*

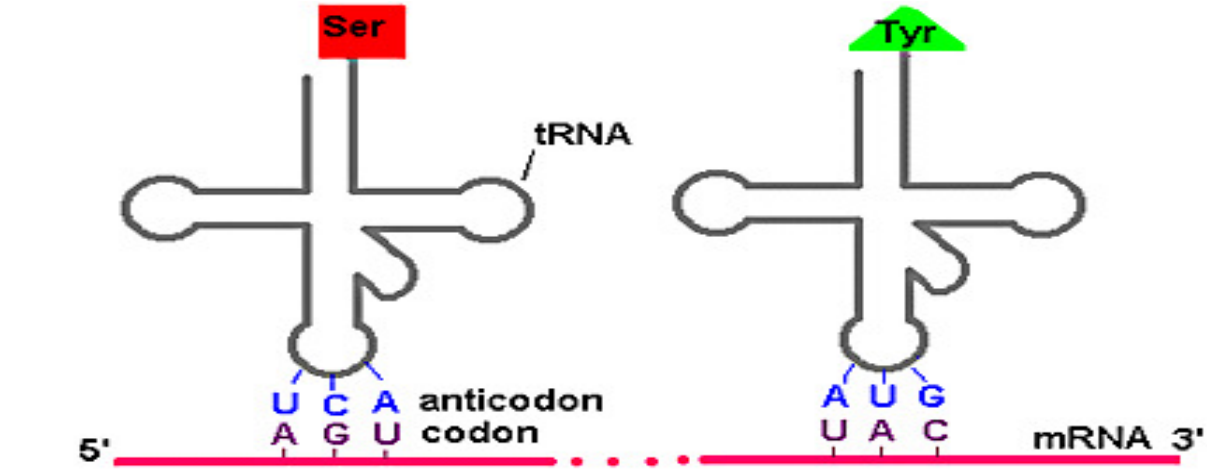
Organisme	Chromosomes	Taille du génome (bp)
Bactéries	1	400,000 a 10,000,000
Levure	12	14,000,000
Mouche	4	300,000,000
Homme	46	6,000,000,000

Séquenceur

De l'ADN aux protéines



Code génétique

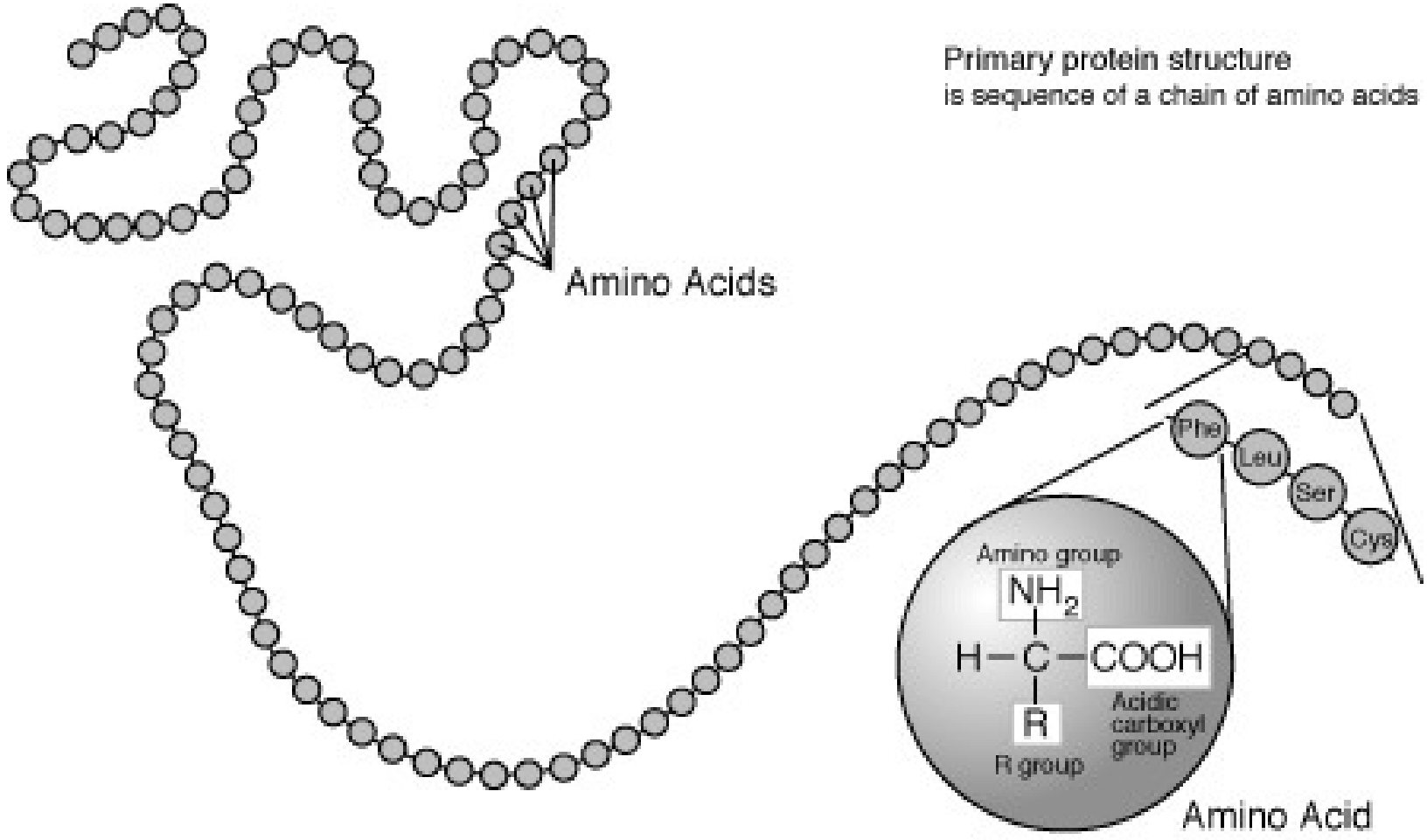


2nd base in codon

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

Protéine = polymère d'acides aminés



Structure de protéine

Protéines

Les protéines représentent 20% du poids de la cellule (eau=70%). Elles ont de *multiples fonctions*:

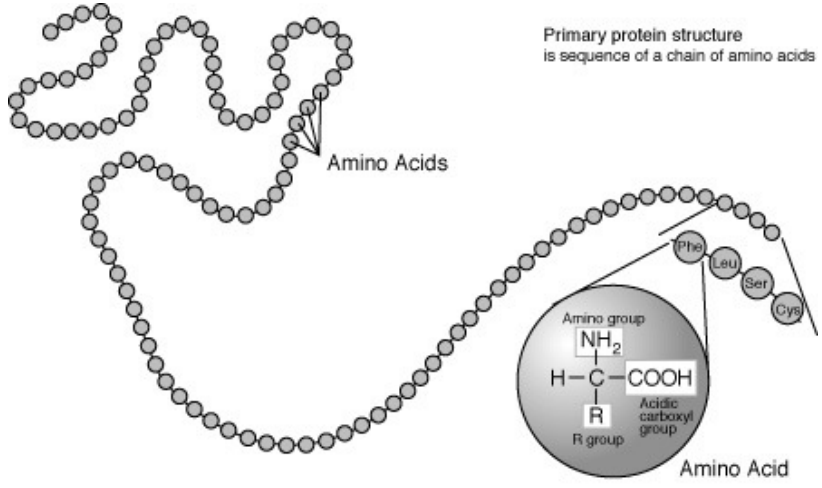
- *Structurale* : ex: le collagene relie les os et les tissus
- *Catalytique*: les *enzymes* catalysent une multitude de réactions biochimique (formant le métabolisme). Ex: la bexokinase permet la conversion du glucose au glucose-6-phosphate
- Les *protéines membranaires* maintiennent l'environnement cellulaire, régulent le volume de la cellule, créent des gradients ioniques pour les muscles et le systeme nerveux...

Espaces des séquences

Motivations

- *Croissance exponentielle* de la quantité de séquences disponible (effet direct des projets de séquencage): ADN, ARN, protéines.
- *Besoin d'algorithmes* pour comparer, classer, analyser ces séquences
- Applications: prédiction de structure/fonctions de protéines (*génomique fonctionnelle*), compréhension des signaux présents dans l'ADN, compréhension de processus tels l'epissage et la régulation génétique...

Séquences protéiques



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	

L'espace des protéines

- Soit $\mathcal{A}$ l'ensemble des 20 acides aminés
- L'espace des séquences possibles est:

$$\mathcal{X} = \bigcup_{n=1}^{\infty} \mathcal{A}^n$$

- On note $|\mathbf{x}|$ la longueur d'une séquence $\mathbf{x} \in \mathcal{X}$.
- On compte actuellement environ 10^6 séquences protéiques observées 'dans la nature'
- ... à comparer aux 20^{300} séquences possibles de 300 lettres de $\mathcal{A}$

Evolution des séquences

- Au sein d'un organisme (croissance) et entre génération, les génomes sont dupliqués.
- Des *erreurs* de duplication ont parfois lieu, notamment:
  - *Substitution* d'une lettre par une autre
  - *Déletion* d'une ou plusieurs lettres
  - *Insertion* d'une ou plusieurs lettres
- Ces erreurs (aléatoires) sont à la base de *l'évolution*

Exemple

```
1.  CGGSLIAMMWFGV
2.  CGGSLIVMMWFGV
3.  CGGSLIVMMNRLMWFGV
4.  CLIVMMNRLMWFGV
```

que l'on écrit généralement:

```
1.  CGGSLIAMM----WFGV
2.  CGGSLIVMM----WFGV
3.  CGGSLIVMMNRLMWFGV
4.  C---LIVMMNRLMWFGV
```

Objectif

- Pendant l'évolution, les *structures et les fonctions* sont beaucoup plus conservées que les séquences
- Pour prédire structure et fonction, il faut donc 'reconnaître' l'évolution à partir des séquences
- Objectif: faire un *noyau pour séquences protéiques qui reconnaisse l'évolution*: $K(\mathbf{x}, \mathbf{x}')$ est grand quand $\mathbf{x}$ et $\mathbf{x}'$ sont liées par l'évolution.

Noyau vectoriel

Motivations

- Nous connaissons plusieurs *noyaux génériques pour des vecteurs*: linéaire, polynomial, RBF Gaussien...
- Ils peuvent être appliqués à une séquence $\mathbf{x}$ de longueur variable si on peut la *vectoriser*, c'est-à-dire représentée par un vecteur $\Phi(\mathbf{x}) \in \mathbb{R}^d$
- *Problème: comment choisir la vectorisation*

$$\Phi : \mathcal{X} \to \mathbb{R}^d \quad ?$$

Vectorisation par caractéristiques globales

- Des caractèristiques quantitatives peuvent être extraites
- Examples:
  - longueur de la séquence
  - pourcentage de A, C, etc...
  - ??

Vectorisation de profiles

- Si à chaque lettre est attachée une caractéristique quantitative, il faut vectoriser une série temporelle de longueur variable, e.g.:
  - polarity
  - hydrophobicity
- Des méthodes puissantes existent pour extraire des caractéristiques de séries temporelles, par exemple:
  - Transformée de Fourier,
  - Fonction d'autocorrélation:

$$\forall j \geq 0, \quad r_j = \frac{1}{n-j} \sum_{i=1}^{n-j} h_i h_{i+j}$$

Noyau spectral

Motivations

- Nous connaissons plusieurs noyaux génériques pour des vecteurs: linéaire, polynomial, RBF Gaussien...
- Comment faire un noyau *simple* et *rapide à calculer* pour des *séquences de longeur variable*
- Référence: C. Leslie, E. Eskin, W.S. Noble,
  The spectrum kernel: a string kernel for SVM protein classif
  Proceedings of PSB 2002.

k-spectre d'une séquence

- Pour tout entier $k > 0$, le k-spectre d'une séquence de longueur finie est la *liste de k-mers (suites de k lettres) qu'elle contient*.
- Exemple: le 3-spectre de

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

est:

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

Noyau spectral

- Soit $k > 0$ fixé
- Pour tout $u \in \mathcal{A}^k$, soit $\Phi_u(\mathbf{x}) :=$ *le nombre de fois que le k-gram u apparaît dans $\mathbf{x}$*.
- On définit le *noyau spectral* par:

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

- C'est un noyau *défini positif*!
- Deux séquences sont comparées à travers leurs k-spectres.

Implémentation

- C'est un somme sur 20^k termes...
- mais la plupart sont nuls. Le k-spectre d'une séquence $\mathbf{x}$ a au plus $|\mathbf{x}| - k + 1$ composantes non nulles.
- On peut donc *calculer le noyau en* $O(|\mathbf{x}| + |\mathbf{x}'|)$
- La *classification par SVM* d'une séquence $\mathbf{x}$ prend $O(|\mathbf{x}|)$:

$$f(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) = \sum_u w_u \Phi_u(\mathbf{x}) = \sum_{i=1}^{|\mathbf{x}|-k+1} w_{x_i \ldots x_{i+k-1}}.$$

Remarques

- Le noyau spectral est *indépendant du type de séquences*: il fonctionne aussi bien sur des séquences biologiques que sur du texte en langage naturel, des programmes en C, ...
- Des *variantes* ont été proposées, notamment pour comparer les spectres à quelques mutations pres (l'implémentation est alors moins rapide)
- Pas toujours tres performant, mais *simple et rapide*

Noyau de sous-séquences

Motivations

- Le noyau spectral ne peut pas prendre en compte les *insertions* et *deletions*.
- Il existe peut-etre des features intéressantes qui sont des *suites non contigues* de lettres
- Références: H. Lodhi et al.,
  Text classification using string kernels, Journal of Machine Learning Research (2), 419-444, 2002.

Sous-séquences

- Pour $1 \leq k \leq n \in \mathbb{N}$, on note $\mathcal{I}(k,n)$ l'ensemble des *suite d'indices* $\mathbf{i} = (i_1, \ldots, i_k)$, avec $1 \leq i_1 < i_2 < \ldots < i_k \leq n$.
- Pour une séquence $\mathbf{x} = x_1 \ldots x_n \in \mathcal{X}$ de longueur n, et pour une suite d'indices $\mathbf{i} \in \mathcal{I}(k,n)$, on définit la *sous-séquence:*

$$\mathbf{x}(\mathbf{i}) := x_{i_1} x_{i_2} \ldots x_{i_k}$$

- La *longueur* de la sous-séquence est:

$$l(\mathbf{i}) = i_k - i_1 + 1.$$

Exemple de sous-séquence

ABRACADABRA

- $\mathbf{i} = (3, 4, 7, 8, 10)$
- $\mathbf{x}(\mathbf{i}) =$RADAR
- $l(\mathbf{i}) = 10 - 3 + 1 = 8$

Noyau de sous-séquences

- Réciproquement, soit $k \in \mathbb{N}$ et $\lambda \in \mathbb{R}^+$ fixés. Pour tout $\mathbf{u} \in \mathcal{A}^k$, on définit $\Phi_{\mathbf{u}} : \mathcal{X} \to \mathbb{R}$ par:

$$\forall \mathbf{x} \in \mathcal{X}, \quad \Phi_{\mathbf{u}}(\mathbf{x}) = \sum_{\mathbf{i} \in \mathcal{I}(k, |\mathbf{x}|): \quad \mathbf{x}(\mathbf{i}) = \mathbf{u}} \lambda^{l(\mathbf{i})}.$$

- Le *noyau de sous-séquences* est le n.d.p. défini par:

$$\forall (\mathbf{x}, \mathbf{x}') \in \mathcal{X}^2, \quad K_{k,\lambda}(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{u} \in \mathcal{A}^k} \Phi_{\mathbf{u}}(\mathbf{x}) \Phi_{\mathbf{u}}(\mathbf{x}')$$

Exemple

u	ca	ct	at	ba	bt	cr	ar	br
$\Phi_u(\text{cat})$	λ^2	λ^3	λ^2	0	0	0	0	0
$\Phi_u(\text{car})$	λ^2	0	0	0	0	λ^3	λ^2	0
$\Phi_u(\text{bat})$	0	0	λ^2	λ^2	λ^3	0	0	0
$\Phi_u(\text{bar})$	0	0	0	λ^2	0	0	λ^2	λ^3

$$\begin{cases} K\,(\text{cat,cat}) = K\,(\text{car,car}) = 2\lambda^4 + \lambda^6 \\ K\,(\text{cat,car}) = \lambda^4 \\ K\,(\text{cat,bar}) = 0 \end{cases}$$

Implémentation

On désire calculer, pour $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, le noyau:

$$K_{n,\lambda}\left(\mathbf{x}, \mathbf{x}'\right) = \sum_{\mathbf{u} \in \mathcal{A}^n} \Phi_{\mathbf{u}}\left(\mathbf{x}\right) \Phi_{\mathbf{u}}\left(\mathbf{x}'\right)$$

$$= \sum_{\mathbf{u} \in \mathcal{A}^n} \sum_{\mathbf{i}:\mathbf{x}(\mathbf{i})=\mathbf{u}} \sum_{\mathbf{i}':\mathbf{x}'(\mathbf{i}')=\mathbf{u}} \lambda^{l(\mathbf{i})+l(\mathbf{i}')}$$

L'énumération de toutes les sous-séquences est trop longue quand n augmente.

Implémentation

Pour $\mathbf{u} \in \mathcal{A}^n$ on a défini

$$\Phi_{\mathbf{u}}(\mathbf{x}) = \sum_{\mathbf{i}:\mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{i_n - i_1 + 1}.$$

Soit maintenant

$$\Psi_{\mathbf{u}}(\mathbf{x}) = \sum_{\mathbf{i}:\mathbf{x}(\mathbf{i})=\mathbf{u}} \lambda^{|\mathbf{x}| - i_1 + 1}.$$

Implémentation (cont.)

En notant $\mathbf{x}(1,j) = x_1 \dots x_j$, une simple ré-écriture montre que, si on note $a \in \mathcal{A}$ la derniere lettre de $\mathbf{u}$ ($\mathbf{u} = \mathbf{v}a$):

$$\Phi_{\mathbf{v}a}(\mathbf{x}) = \sum_{j \in [1,|\mathbf{x}|]: x_j = a} \Psi_{\mathbf{v}}(\mathbf{x}(1,j-1))\,\lambda,$$

et

$$\Psi_{\mathbf{v}a}(\mathbf{x}) = \sum_{j \in [1,|\mathbf{x}|]: x_j = a} \Psi_{\mathbf{v}}(\mathbf{x}(1,j-1))\,\lambda^{|\mathbf{x}|-j+1}.$$

Implémentation (cont.)

On remarque également que, si on s'intéresse à une séquence de la forme $\mathbf{x}a$ (i.e., dont la derniere lettre est $a \in \mathcal{A}$), alors

- Si la derniere lettre de $\mathbf{u}$ *n'est pas* a:

$$\begin{cases} \Phi_{\mathbf{u}}(\mathbf{x}a) & = \Phi_{\mathbf{u}}(\mathbf{x}), \\ \Psi_{\mathbf{u}}(\mathbf{x}a) & = \lambda \Psi_{\mathbf{u}}(\mathbf{x}). \end{cases}$$

- Si la derniere lettre de $\mathbf{u}$ est égale à a (i.e., $\mathbf{u} = \mathbf{v}a$ *avec* $\mathbf{v} \in \mathcal{A}^{n-1}$):

$$\begin{cases} \Phi_{\mathbf{v}a}(\mathbf{x}a) & = \Phi_{\mathbf{v}a}(\mathbf{x}) + \lambda \Psi_{\mathbf{v}}(\mathbf{x}), \\ \Psi_{\mathbf{v}a}(\mathbf{x}a) & = \lambda \Psi_{\mathbf{v}a}(\mathbf{x}) + \lambda \Psi_{\mathbf{v}}(\mathbf{x}). \end{cases}$$

Implémentation (cont.)

Montrons comment

$$B_n\left(\mathbf{x},\mathbf{x}'\right) := \sum_{\mathbf{u}\in\mathcal{A}^n} \Psi_{\mathbf{u}}\left(\mathbf{x}\right)\Psi_{\mathbf{u}}\left(\mathbf{x}'\right)$$

et le noyau qui nous intéresse:

$$K_n\left(\mathbf{x},\mathbf{x}'\right) := \sum_{\mathbf{u}\in\mathcal{A}^n} \Phi_{\mathbf{u}}\left(\mathbf{x}\right)\Phi_{\mathbf{u}}\left(\mathbf{x}'\right)$$

peuvent se calculer de manière récursive. On a tout d'abord:

$$\begin{cases} B_0\left(\mathbf{x},\mathbf{x}'\right) = K_0\left(\mathbf{x},\mathbf{x}'\right) = 0 & \text{pour tout } \mathbf{x},\mathbf{x}' \\ B_k\left(\mathbf{x},\mathbf{x}'\right) = K_k\left(\mathbf{x},\mathbf{x}'\right) = 0 & \text{si } \min\left(|\,\mathbf{x}\,|,|\,\mathbf{x}'\,|\right) < k \end{cases}$$

Calcul récursive de B_n

$$
\begin{aligned}
&B_n\left(\mathbf{x}a, \mathbf{x}'\right) \\
&= \sum_{\mathbf{u} \in \mathcal{A}^n} \Psi_{\mathbf{u}}\left(\mathbf{x}a\right) \Psi_{\mathbf{u}}\left(\mathbf{x}'\right) \\
&= \lambda \sum_{\mathbf{u} \in \mathcal{A}^n} \Psi_{\mathbf{u}}\left(\mathbf{x}\right) \Psi_{\mathbf{u}}\left(\mathbf{x}'\right) + \lambda \sum_{\mathbf{v} \in \mathcal{A}^{n-1}} \Psi_{\mathbf{v}}\left(\mathbf{x}\right) \Psi_{\mathbf{v}a}\left(\mathbf{x}'\right) \\
&= \lambda B_n\left(\mathbf{x}, \mathbf{x}'\right) + \\
&\quad \lambda \sum_{\mathbf{v} \in \mathcal{A}^{n-1}} \Psi_{\mathbf{v}}\left(\mathbf{x}\right) \left(\sum_{j \in [1, |\,\mathbf{x}'\,|] : x'_j = a} \Psi_{\mathbf{v}}\left(\mathbf{x}'\left(1, j-1\right)\right) \lambda^{|\,\mathbf{x}'\,| - j + 1} \right) \\
&= \lambda B_n\left(\mathbf{x}, \mathbf{x}'\right) + \sum_{j \in [1, |\,\mathbf{x}'\,|] : x'_j = a} B_{n-1}\left(\mathbf{x}, \mathbf{x}'\left(1, j-1\right)\right) \lambda^{|\,\mathbf{x}'\,| - j + 2}
\end{aligned}
$$

Calcul récursive de K_n

$$
\begin{aligned}
&K_n\left(\mathbf{x}a, \mathbf{x}'\right) \\
&= \sum_{\mathbf{u}\in\mathcal{A}^n} \Phi_{\mathbf{u}}\left(\mathbf{x}a\right)\Phi_{\mathbf{u}}\left(\mathbf{x}'\right) \\
&= \sum_{\mathbf{u}\in\mathcal{A}^n} \Phi_{\mathbf{u}}\left(\mathbf{x}\right)\Phi_{\mathbf{u}}\left(\mathbf{x}'\right) + \lambda \sum_{\mathbf{v}\in\mathcal{A}^{n-1}} \Psi_{\mathbf{v}}\left(\mathbf{x}\right)\Phi_{\mathbf{v}a}\left(\mathbf{x}'\right) \\
&= K_n\left(\mathbf{x}, \mathbf{x}'\right) + \\
&\quad \lambda \sum_{\mathbf{v}\in\mathcal{A}^{n-1}} \Psi_{\mathbf{v}}\left(\mathbf{x}\right) \left(\sum_{j\in[1,|\,\mathbf{x}'\,|]:x'_j=a} \Psi_{\mathbf{v}}\left(\mathbf{x}'\left(1,j-1\right)\right)\lambda \right) \\
&= \lambda K_n\left(\mathbf{x}, \mathbf{x}'\right) + \lambda^2 \sum_{j\in[1,|\,\mathbf{x}'\,|]:x'_j=a} B_{n-1}\left(\mathbf{x}, \mathbf{x}'\left(1,j-1\right)\right)
\end{aligned}
$$

Remarques

- L'algorithme est en $O(n\,|\,\mathbf{x}\,|\,|\,\mathbf{x}'\,|^2)$. On peut réutiliser les calculs pour avoir une complexité $O(n\,|\,\mathbf{x}\,|\,|\,\mathbf{x}'\,|)$
- Beaucoup plus long que le noyau spectral.
- A fourni des résultats intéressants pour la classification de textes.

Similarité par alignement

Le problème

Soient les deux séquences:

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

Comment *quantifier leur similarité* au sens de l'évolution?

Alignement des séquences

Aligner = *mettre en correspondance les lettres* des deux séquences, quitte à ajouter des gaps dans l'une ou dans l'autre

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

Alignement (cont.)

Mathématiquement, un alignement π de $p \geq 0$ lettres entre deux séquences $\mathbf{x}$ et $\mathbf{x}'$ de longueurs m et n est une paire de p-uples:

$$\pi = \left((\pi_1(1), \ldots, \pi_1(p)), (\pi_2(1), \ldots, \pi_2(p)) \right),$$

avec:

$$1 \leq \pi_1(1) < \pi_1(2) < \ldots < \pi_1(p) \leq |\mathbf{x}|,$$
$$1 \leq \pi_2(1) < \pi_2(2) < \ldots < \pi_2(p) \leq |\mathbf{y}|.$$

On note $\Pi(\mathbf{x}, \mathbf{x}')$ l'ensemble des alignements possibles entre les deux séquences $\mathbf{x}$ et $\mathbf{x}'$.

Exemple

$$\pi = ((1, 5, 6, 7, 8, 9, 10, 11, 12, 13), (1, 2, 3, 4, 5, 6, 7, 12, 13, 14))$$

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

Scoring d’alignement

Un alignement est bon si:

- les lettres alignées ’se ressemblent’;
- il y a peu de gaps.

On va construire une fonction (un score) qui est d’autant plus grand que les séquences se ressemblent

Scoring d'alignement (cont.)

Pour tout alignement $\pi \in \Pi(\mathbf{x}, \mathbf{y})$, on définit son score par:

$$s_{S,g}(\pi) := \sum_{i=1}^{|\pi|} S\left(x_{\pi_1(i)}, y_{\pi_2(i)}\right)$$
$$- \sum_{i=1}^{|\pi|-1} \left[g(\pi_1(i+1) - \pi_1(i) - 1) + g(\pi_2(i+1) - \pi_2(i) - 1)\right],$$

ou:

- S est une matrice 20×20 qui définit la similarité entre les paires de lettres
- $g : \mathbb{N} \to \mathbb{R}$ est une fonction qui définit le coût d'un gap de longueur donné (avec $g(0) = 0$).

Exemple

$$\pi = ((1,5,6,7,8,9,10,11,12,13),(1,2,3,4,5,6,7,12,13,14))$$

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

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

Score de Smith-Waterman

Définition 1 *On appelle score de Smith-Waterman, ou* score d'alignement local *entre deux séquences* $\mathbf{x}$ *et* $\mathbf{x}'$ *le score du meilleur alignement:*

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

Calcul de SW

- Soit le coût linéaire $g(n) = \gamma n$
- Pour tout $1 \leq a \leq b \leq |\,\mathbf{x}\,|$, soit $\mathbf{x}[a,b] := x_a \dots x_b$.
- Pour tout $1 \leq a \leq |\,\mathbf{x}\,|$ et $1 \leq a' \leq |\,\mathbf{x}'\,|$, on observe que:

$$SW_{S,g}(\mathbf{x}[1,a], \mathbf{x}'[1,a']) =$$

$$\max \begin{cases} 0 \\ SW_{S,g}(\mathbf{x}[1,a-1], \mathbf{x}'[1,a']) + \gamma \\ SW_{S,g}(\mathbf{x}[1,a], \mathbf{x}'[1,a'-1]) + \gamma \\ SW_{S,g}(\mathbf{x}[1,a-1], \mathbf{x}'[1,a'-1]) + S\left(x_a, x'_{a'}\right) \end{cases}$$

- Cela fournit une méthode de calcul en $O(|\,\mathbf{x}\,|\,|\,\mathbf{x}'\,|)$

Exercice

Trouver un algorithme en $O(|\mathbf{x}||\mathbf{x}'|)$ pour calculer le score d'alignement local dans le cas d'un cout affine:

$$\begin{cases} g(0) & = 0, \\ g(n) & = d + e(n-1) \text{ if } n \geq 1, \end{cases}$$

Noyau de convolution

Motivations

- Le score d'alignement local est censé être une bonne mesure de similarité entres séquences biologiques.
- Mais... ce n'est pas un noyau en général (dépend du choix des parametres).
- Comment faire un noyau qui imite ce score?
- Ref: Haussler 1999 (Convolution kernels), Vert et al. 2004 (Local alignment kernels)

Opérations de bases sur les n.d.p.

Lemme 2 *Si K_1 et K_2 sont des n.d.p., alors:*

$$K_1 + K_2,$$

$$K_1 K_2, \text{ et}$$

$$cK_1, \text{ pour } c \geq 0,$$

sont également des n.d.p. Si $(K_i)_{i \geq 1}$ est une famille de n.d.p. qui converge ponctuellement en une fonction K:

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

alors K est aussi un n.d.p.

Preuves

Soient A et B des matrices $n \times n$ semi-définies positives (s-d.p.). Alors, par diagonalisation, on peut écrire:

$$A_{i,j} = \sum_{p=1}^{n} f_p(i) f_p(j)$$

pour une famille de vecteurs $f_1, \ldots, f_n$. Donc, pour $\boldsymbol{\alpha} \in \mathbb{R}^n$ arbitraire:

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j A_{i,j} B_{i,j} = \sum_{p=1}^{n} \sum_{i,j=1}^{n} \alpha_i f_p(i) \alpha_j f_p(j) B_{i,j} \geq 0.$$

Preuves (cont.)

La matrice $C_{i,j} = A_{i,j} B_{i,j}$ est donc s-d.p, donc $K_1 K_2$ est un n.d.p. si K_1 et K_2 le sont. Les autres propriétés découlent directement de la définition. $\square$

Lemme: Noyaux produits

Lemme 3 *Soit $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2$. Soient K_1 un n.d.p. sur $\mathcal{X}_1$, et K_2 un n.d.p. sur $\mathcal{X}_2$. Alors les fonctions suivantes sont des n.d.p. sur $\mathcal{X}$:*

- *la* somme directe,

$$K\left((\mathbf{x}_1, \mathbf{x}_2), (\mathbf{y}_1, \mathbf{y}_2)\right) = K_1\left(\mathbf{x}_1, \mathbf{y}_1\right) + K_2\left(\mathbf{x}_2, \mathbf{y}_2\right),$$

- *le* produit direct

$$K\left((\mathbf{x}_1, \mathbf{x}_2), (\mathbf{y}_1, \mathbf{y}_2)\right) = K_1\left(\mathbf{x}_1, \mathbf{y}_1\right) K_2\left(\mathbf{x}_2, \mathbf{y}_2\right).$$

Preuves

Si K_1 est un noyau, soit $\Phi_1 : \mathcal{X}_1 \mapsto \mathcal{H}$ tel que

$$K_1(\mathbf{x}_1, \mathbf{y}_1) = \langle \Phi_1(\mathbf{x}_1), \Phi_1(\mathbf{y}_1) \rangle_{\mathcal{H}}.$$

Soit $\Phi : \mathcal{X}_1 \times \mathcal{X}_2 \to \mathcal{H}$ défini par:

$$\Phi((\mathbf{x}_1, \mathbf{x}_2)) = \Phi_1(\mathbf{x}_1).$$

Alors pour $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ et $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2) \in \mathcal{X}$, on a

$$\langle \Phi((\mathbf{x}_1, \mathbf{x}_2)), \Phi((\mathbf{y}_1, \mathbf{y}_2)) \rangle_{\mathcal{H}} = K_1(\mathbf{x}_1, \mathbf{x}_2),$$

ce qui montre que $K(\mathbf{x}, \mathbf{y}) := K_1(\mathbf{x}_1, \mathbf{y}_1)$ est un n..p. sur $\mathcal{X}_1 \times \mathcal{X}_2$. Le lemme découle des opérations d'addition et multiplication sur n.d.p. $\square$

Lemme: Noyaux pour parties

Lemme 4 *Soit K un n.d.p. sur un espace $\mathcal{X}$, et soit $\mathcal{P}(\mathcal{X})$ l'ensemble des parties finies d'éléments de $\mathcal{X}$. Alors la fonction K_P sur $\mathcal{P}(\mathcal{X}) \times \mathcal{P}(\mathcal{X})$ définie par:*

$$\forall A, B \in \mathcal{P}(\mathcal{X}), \quad K_P(A, B) := \sum_{\mathbf{x} \in A} \sum_{\mathbf{y} \in B} K(\mathbf{x}, \mathbf{y})$$

est un n.d.p. sur $\mathcal{P}(\mathcal{X})$.

Preuves

Soit $\Phi : \mathcal{X} \mapsto \mathcal{H}$ tel que

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

Alors, pour $A, B \in \mathcal{P}(\mathcal{X})$, on à:

$$\begin{aligned} K_P(A, B) &= \sum_{\mathbf{x} \in A} \sum_{\mathbf{y} \in B} \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle_{\mathcal{H}} \\ &= \left\langle \sum_{\mathbf{x} \in A} \Phi(\mathbf{x}), \sum_{\mathbf{y} \in B} \Phi(\mathbf{y}) \right\rangle_{\mathcal{H}} \\ &= \langle \Phi_P(A), \Phi_P(B) \rangle_{\mathcal{H}}, \end{aligned}$$

avec $\Phi_P(A) := \sum_{\mathbf{x} \in A} \Phi(\mathbf{x})$. $\square$

Convolution

Nous pouvons maintenant l'opération qui nous permettra de construire des noyaux pour séquences:

Définition 5 *Soient K_1 and K_2 deux n.d.p pour séquences.. La convolution de K_1 et K_2, denotée $K_1 \star K_2$, est la fonctions suivante, pour $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$:*

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

Théorème 6 *Si K_1 et K_2 sont des n.d.p., alors $K_1 \star K_2$ est aussi un n.d.p.*

Preuve

Soit $\mathcal{X}$ l'ensemble des séquences finies. Pour $\mathbf{x} \in \mathcal{X}$, soit

$$R(\mathbf{x}) = \{(\mathbf{x}_1, \mathbf{x}_2) \in \mathcal{X} \times \mathcal{X} : \mathbf{x} = \mathbf{x}_1 \mathbf{x}_2\} \subset \mathcal{X} \times \mathcal{X}.$$

On peut alors écrire

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

qui est un n.d.p. par les lemmes 3 et 4. □

3 noyaux de bases pour séquences

Pour toutes séquences $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, définissons:

- Un noyau trivial:

$$K_0(\mathbf{x}, \mathbf{y}) := 1.$$

- Un noyau pour lettres seules

$$K_a^{(\beta)}(\mathbf{x}, \mathbf{y}) := \begin{cases} 0 & \textit{si } |\mathbf{x}| \neq 1 \textit{ ou } |\mathbf{y}| \neq 1, \\ \exp(\beta S(\mathbf{x}, \mathbf{y})) & \textit{sinon} \end{cases}$$

- Un noyau pour gaps:

$$K_g^{(\beta)}(\mathbf{x}, \mathbf{y}) = \exp\left[\beta\left(g(|\mathbf{x}|) + g(|\mathbf{x}|)\right)\right]$$

Remarques

- $S : \mathcal{A}^2 \to \mathbb{R}$ est la fonction de similarité entre lettres utilisée dans le score d'alignement. $K_a^{(\beta)}$ est un n.d.p. seulement quand la matrice:

$$\left(\exp\left(\beta s(a,b)\right)\right)_{(a,b)\in\mathcal{A}^2}$$

  est semi-définie positive.

- g est la fontion utilisée pour pénaliser les gaps dans le score d'alignement. Le noyau pour gap est toujours d.p. car il s'écrit simplement comme un produit scalaire:

$$K_g^{(\beta)}\left(\mathbf{x},\mathbf{y}\right) = \exp\left(\beta g\left(|\,\mathbf{x}\,|\right)\right) \times \exp\left(\beta g\left(|\,\mathbf{y}\,|\right)\right)$$

Convolution des noyaux

- Pour chaque $n \in \mathbb{N}$, on définit un noyau pour alignement de n lettres:

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

- Le *noyau d'alignement local* est alors défini par:

$$K_{LA}^{(\beta)} = \sum_{i=0}^{\infty} K_{(i)}^{(\beta)}.$$

Remarques

- $K_{(n)}^{(\beta)}$ est un n.d.p. en tant que convolué de n.d.p. (théoreme 6).
- $K_{LA}^{(\beta)}$ est un n.d.p. en tant que limite ponctuelle de n.d.p. (lemme 2)

Lien avec le score d'alignement

Théorème 7 *Le n.d.p.* $K_{LA}^{(\beta)}$ *vérifie, pour tout* $\mathbf{x}, \mathbf{y} \in \mathcal{X}$*::*

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

et

$$\lim_{\beta \to +\infty} \frac{1}{\beta} \ln K_{LA}^{(\beta)}(\mathbf{x}, \mathbf{y}) = SW_{S,g}(\mathbf{x}, \mathbf{y}).$$

Remarques

- Le noyau d'alignement local est calculé à partir de scores d'alignements censés avoir un sens biologique
- Ce théoreme montre pourquoi le score de SW n'est pas un n.d.p.:
  - il ne considere que le score du meilleur alignement, au lieu de moyenner sur tous les alignements
  - il faudrait prendre son exponentielle pour se rapprocher du noyau d'alignement local.

Preuve du théoreme 7

Pour deux séquences $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, soit:

$$\Pi_n^0(\mathbf{x}, \mathbf{y}) := \{\pi \in \Pi_n(\mathbf{x}, \mathbf{y}) : \pi_1(n) = |\mathbf{x}|, \pi_2(n) = |\mathbf{y}|\}.$$

On va alors utiliser le

Lemme 8 *Pour $n \geq 1$, on a:*

$$\sum_{\pi \in \Pi_n^0(\mathbf{x},\mathbf{y})} \exp\left(\beta s(\mathbf{x}, \mathbf{y}, \pi)\right) = K_0 \star K_a \star \left(K_g \star K_a\right)^{n-1}(\mathbf{x}, \mathbf{y}),$$

et pour $n = 0$:

$$\sum_{\pi \in \Pi_0^0(\mathbf{x},\mathbf{y})} \exp\left(\beta s(\mathbf{x}, \mathbf{y}, \pi)\right) = K_0(\mathbf{x}, \mathbf{y})$$

Preuve du théoreme 7

On déduit le théoreme 7 du lemme 8 en écrivant:

$$\begin{aligned}
&\sum_{\pi \in \Pi_n(\mathbf{x},\mathbf{y})} \exp\left(\beta s(\mathbf{x},\mathbf{y},\pi)\right) \\
&\quad = \sum_{\mathbf{x}=\mathbf{x}_1\mathbf{x}_2,\mathbf{y}=\mathbf{y}_1\mathbf{y}_2} \sum_{\pi \in \Pi_n^0(\mathbf{x}_1,\mathbf{y}_1)} \exp\left(\beta s(\mathbf{x}_1,\mathbf{y}_1,\pi)\right) \\
&\quad = \sum_{\mathbf{x}=\mathbf{x}_1\mathbf{x}_2,\mathbf{y}=\mathbf{y}_1\mathbf{y}_2} K_0 \star K_a \star \left(K_g \star K_a\right)^{n-1} (\mathbf{x}_1,\mathbf{y}_1) \\
&\quad = K_0 \star K_a \star \left(K_g \star K_a\right)^{n-1} \star K_0(\mathbf{x},\mathbf{y}) \quad \square
\end{aligned}$$

Preuve du lemme 8

On prouve le lemme 8 par récurrence sur n.

- Il est trivial pour $n = 0$ (le score d'un alignement de 0 lettre étant nul).
- Pour $n = 1$, $\Pi_n^0(\mathbf{x}, \mathbf{y})$ est réduit au singleton $\{(|\mathbf{x}|, |\mathbf{y}|)\}$ (seules les dernieres lettres sont alignées), qui a un score $S(x_{|\mathbf{x}|}, y_{|\mathbf{y}|})$. D'autre part,

$$\begin{aligned} K_0 \star K_a(\mathbf{x}, \mathbf{y}) &= \sum_{\mathbf{x}_1\mathbf{x}_2=\mathbf{x}, \mathbf{y}_1\mathbf{y}_2=\mathbf{y}} K_0(\mathbf{x}_1, \mathbf{y}_1) K_a(\mathbf{x}_2, \mathbf{y}_2) \\ &= \exp\left[\beta S(x_{|\mathbf{x}|}, y_{|\mathbf{y}|})\right]. \end{aligned}$$

Preuve du lemme 8

Supposons le lemme valide pour $n-1$. Pour tout $\pi \in \Pi_n^0(\mathbf{x}, \mathbf{y})$, soit $f(\pi) \in \Pi_n(\mathbf{x}, \mathbf{y})$ l'alignement obtenu en retirant les dernieres lettres alignées dans π. On a alors:

$$\sum_{\pi \in \Pi_n^0(\mathbf{x},\mathbf{y})} \exp\left(\beta s(\mathbf{x}, \mathbf{y}, \pi)\right)$$

$$= \sum_{\mathbf{x}=\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3, \mathbf{y}=\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3} \sum_{\pi' \in \Pi_{n-1}^0(\mathbf{x}_1,\mathbf{y}_1)} \exp\left(\beta s(\mathbf{x}_1, \mathbf{y}_1, f^{-1}(\pi'))\right),$$

ou la somme porte porte sur toutes les décompositions telles que $|\mathbf{x}_3| = |\mathbf{y}_3| = 1$.

Preuve du lemme 8

Pour une telle décomposition $\mathbf{x} = \mathbf{x}_1\mathbf{x}_2\mathbf{x}_3$, le score d'alignement est par définition:

$$s(\mathbf{x}, \mathbf{y}, \pi) = s(\mathbf{x}_1, \mathbf{y}_1, f(\pi)) + g(|\mathbf{x}_2|) + g(|\mathbf{y}_2|) + S(\mathbf{x}_3, \mathbf{y}_3),$$

donc:

$$\exp\left(\beta s(\mathbf{x}, \mathbf{y}, \pi)\right) = \exp\left(\beta s(\mathbf{x}_1, \mathbf{y}_1, f(\pi))\right) K_g(\mathbf{x}_2, \mathbf{y}_2) K_a(\mathbf{x}_3, \mathbf{y}_3).$$

Preuve du lemme 8

Le noyau K_a étant nul pour les séquences non réduites à une lettre, on en déduit que:

$$\sum_{\pi \in \Pi_n^0(\mathbf{x},\mathbf{y})} \exp\left(\beta s(\mathbf{x},\mathbf{y},\pi)\right)$$

$$= \sum_{\mathbf{x}=\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3, \mathbf{y}=\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3} \sum_{\pi \in \Pi_{n-1}^0(\mathbf{x}_1,\mathbf{y}_1)} \exp\left(\beta s(\mathbf{x}_1,\mathbf{y}_1,f(\pi))\right)$$

$$\times K_g(\mathbf{x}_2,\mathbf{y}_2) K_a(\mathbf{x}_3,\mathbf{y}_3),$$

ou la somme porte sur toutes les décompositions possible de $\mathbf{x}$ et $\mathbf{y}$ en $\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3$ et $\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3$.

Preuve du lemme 8

En utilisant l'hypothese de récurrence on en déduit:

$$\sum_{\pi \in \Pi_n^0(\mathbf{x},\mathbf{y})} \exp\left(\beta s(\mathbf{x},\mathbf{y},\pi)\right)$$

$$= \sum_{\mathbf{x}=\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3, \mathbf{y}=\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3} \sum_{\pi \in \Pi_{n-1}^0(\mathbf{x}_1,\mathbf{y}_1)} \exp\left(\beta s(\mathbf{x}_1,\mathbf{y}_1,f(\pi))\right)$$

$$\times K_g(\mathbf{x}_2,\mathbf{y}_2) K_a(\mathbf{x}_3,\mathbf{y}_3)$$

$$= \sum_{\mathbf{x}=\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3, \mathbf{y}=\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3} K_0 \star K_a \star \left(K_g \star K_a\right)^{n-2}(\mathbf{x}_1,\mathbf{y}_1)$$

$$\times K_g(\mathbf{x}_2,\mathbf{y}_2) K_a(\mathbf{x}_3,\mathbf{y}_3)$$

$$= K_0 \star K_a \star \left(K_g \star K_a\right)^{n-1}(\mathbf{x},\mathbf{y}) \quad \square$$

Implémentation

- Le score d'alignement peut se calculer en $O\left(|\mathbf{x}||\mathbf{y}|\right)$ par programmation dynamique.
- Le noyau d'alignement aussi!, avec de tres légere modifications.

On suppose que la fonction de gap est affine:

$$\begin{cases} g(0) & = 0, \\ g(n) & = d + e(n-1) \text{ si } n \geq 1, \end{cases}$$

Implémentation

Le noyau peut se calculer par:

$$K_{LA}^{(\beta)}(\mathbf{x},\mathbf{y}) = 1 + X_2(|\mathbf{x}|,|\mathbf{y}|) + Y_2(|\mathbf{x}|,|\mathbf{y}|) + M(|\mathbf{x}|,|\mathbf{y}|),$$

ou $M(i,j), X(i,j), Y(i,j), X_2(i,j)$, et $Y_2(i,j)$ pour $0 \leq i \leq |\mathbf{x}|$, et $0 \leq j \leq |\mathbf{y}|$ sont définis récursivement par:

$$\begin{cases} M(i,0) = M(0,j) = 0, \\ X(i,0) = X(0,j) = 0, \\ Y(i,0) = Y(0,j) = 0, \\ X_2(i,0) = X_2(0,j) = 0, \\ Y_2(i,0) = Y_2(0,j) = 0, \end{cases}$$

Implémentation

...et pour $i = 1, \ldots, |\mathbf{x}|$ et $j = 1, \ldots, |\mathbf{y}|$:

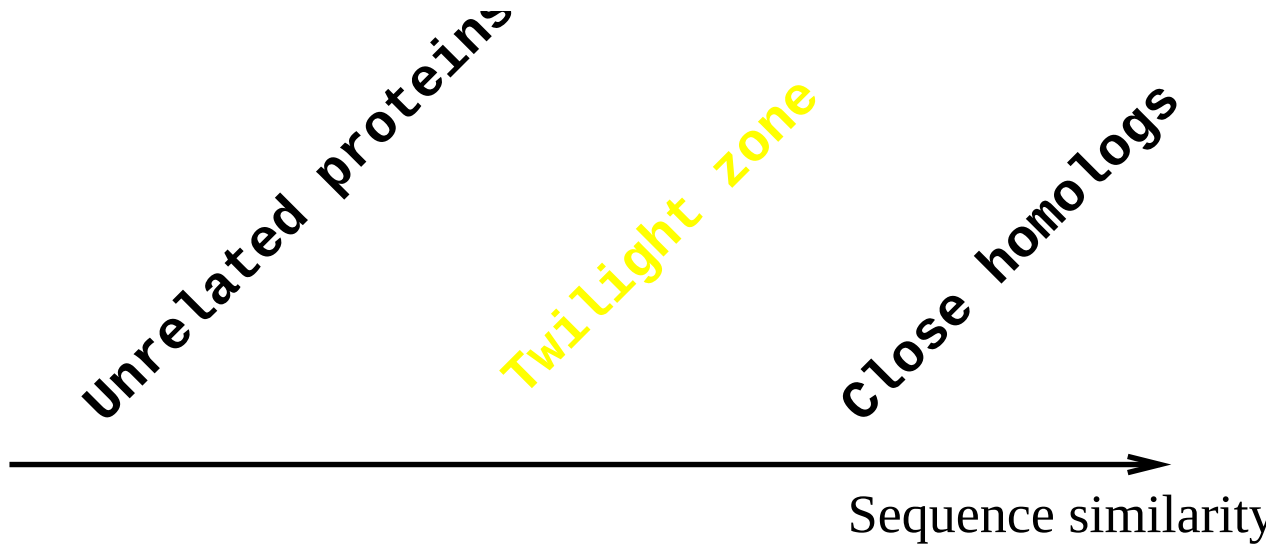
$$
\begin{cases}
M(i,j) & = \exp(\beta S(x_i, y_j)) \Big[1 + X(i-1, j-1) \\
& \qquad + Y(i-1, j-1) + M(i-1, j-1) \Big], \\
X(i,j) & = \exp(\beta d) M(i-1, j) + \exp(\beta e) X(i-1, j), \\
Y(i,j) & = \exp(\beta d) \left[M(i, j-1) + X(i, j-1) \right] \\
& \qquad + \exp(\beta e) Y(i, j-1), \\
X_2(i,j) & = M(i-1, j) + X_2(i-1, j), \\
Y_2(i,j) & = M(i, j-1) + X_2(i, j-1) + Y_2(i, j-1).
\end{cases}
$$

Implémentation: preuve

- La preuve de la validité de l'implémentation se fait par récurrence, voir Vert et al. (2004).
- Laissée en exercice...

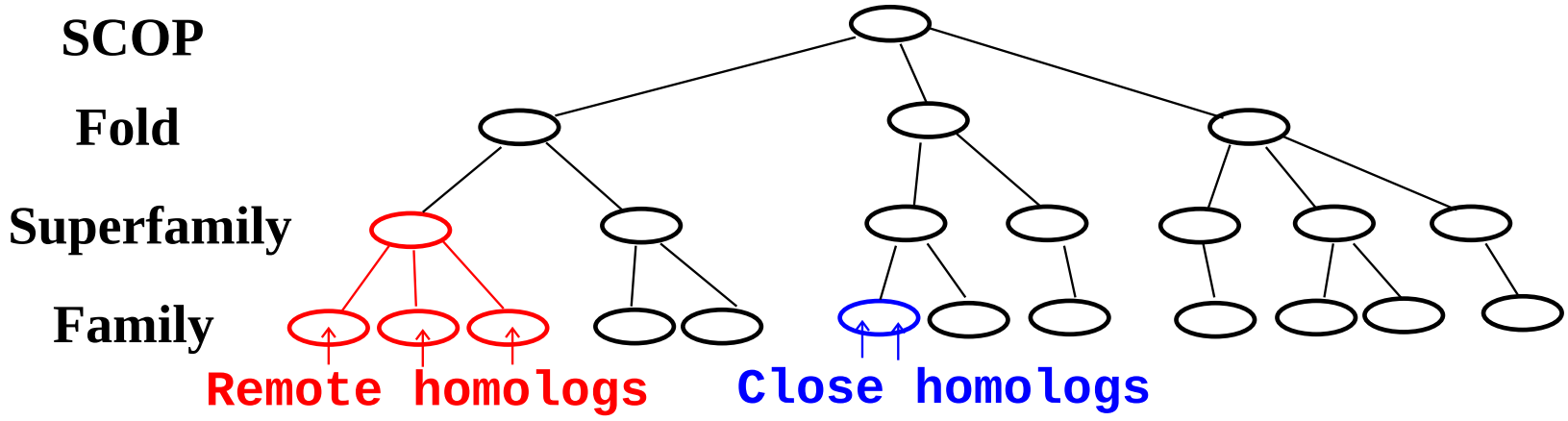
Application: détection d'homologie lointaine

Le probleme de l'homologie lointaine



- Same structure/function but sequence diverged
- *Remote homology can not be found by direct sequence similarity*

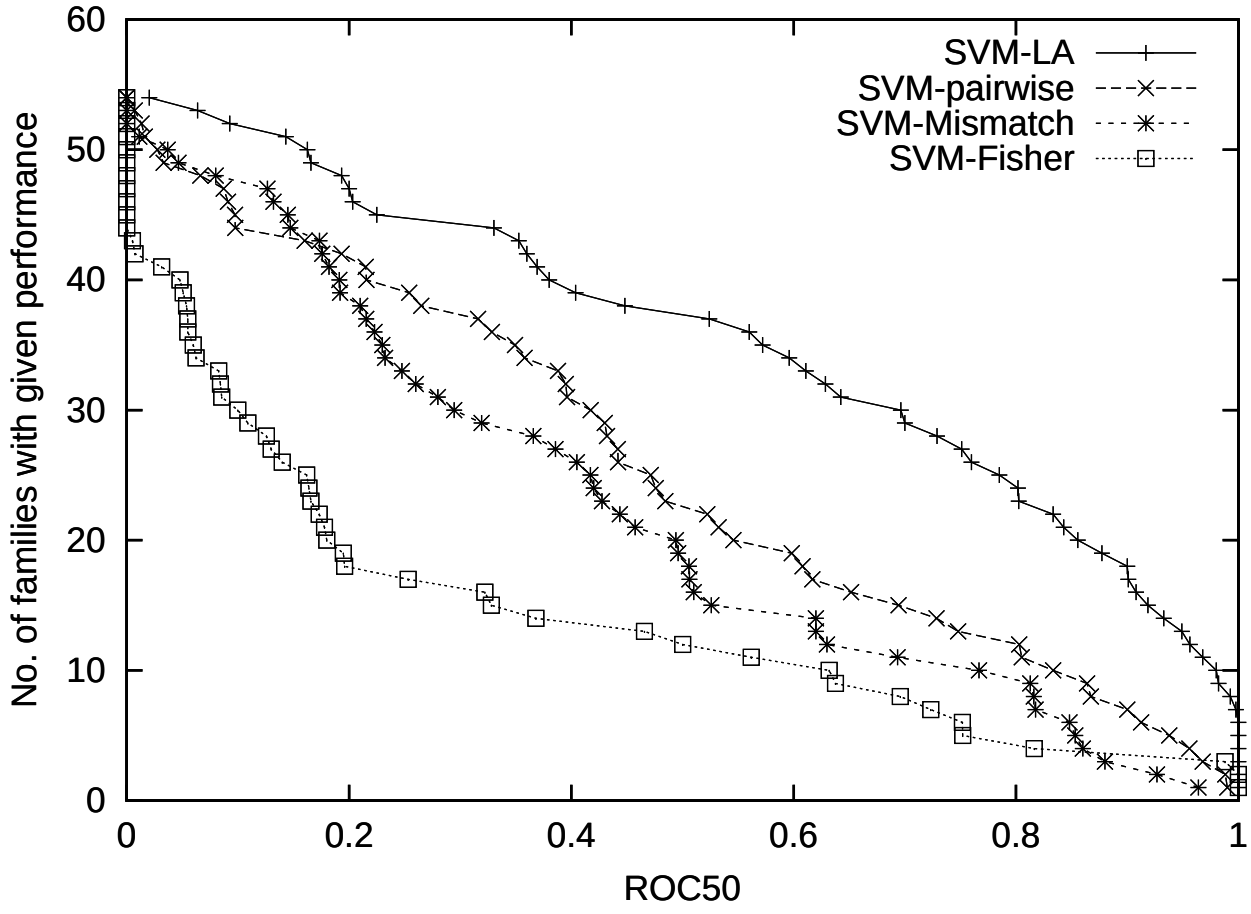
SCOP database



A benchmark experiment

- Can we predict the *superfamily* of a domain if we have not seen any member of its *family* before?
- During *learning*: remove a family and learn the difference between the superfamily and the rest
- Then, use the model to *test* each domain of the family removed

SCOP superfamily recognition benchmark



Remarque

- Les valeurs du noyau LA sont exponentiellement petite. En pratique on a du tricher et prendre leur logarithme. Le résultat n'est plus un n.d.p., mais on s'en est sorti avec des procédures ad hoc pour rendre la matrice de Gram symétrique définie positive.
- Le noyau d'alignement local est actuellement la méthode la plus performante pour cette expérience.