

# Introduction à la bioinformatique

6. Chercher les homologues dans des bases de données et vérifier la qualité du résultat

1

## Objectifs

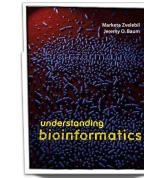
- Comprendre les différences entre l'alignement des séquences et chercher des homologues dans une base de données
- Être capable de mettre en place les systèmes FASTA et BLAST
- Comprendre comment on peut vérifier la qualité du résultat

3

© Tom Lenaerts, 2012

## Bibliographie

- Zvelebil et Baum, Understanding bioinformatics
- W. Pearson et D.J. Lipman (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci USA 85:2444-2448
- J.-P. Dumas et J. Ninio (1981) Efficient algorithms for folding and comparing nucleic acid sequences Nucleic Acid Research 10(1): 197-206
- D.J. Lipman et W.R. Pearson (1985) Rapid and sensitive protein similarity searches. Science 227:1435-1441
- S.F. Altschul et al (1990) Basic local alignment search tool. J Mol Biol 215:403-410
- M. Cameron et al (2006) A deterministic finite automaton for faster protein hit detection in BLAST. Jour Comp Biol 13(4):965-978
- Z. Zhang et al (2000) A greedy algorithm for aligning DNA sequences Jour Comp Biol 7(1/2):203-214



2

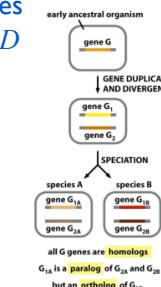
© Tom Lenaerts, 2012

## Déetecter les homologues

- Maintenant, nous savons comment faire des alignements (L3) et calculer leurs scores (L4)
- La question suivante est:  
*Etant donnée la séquence  $q$ , peut-on trouver des autres séquences  $d$  dans une base de données  $D$  qui sont homologues à  $q$ ?*

Par exemple, si la séquence  $q$  correspond au gène G2A, le processus de recherche devrait trouver tous les gènes similaires en mettant les gènes G1A, G1B et G2B au dessus de la liste parce qu'il sont des homologues de G2A

Les séquences qui sont trouvées par cette méthode pourraient donner une idée sur la structure et la fonction de la protéine  $q$



4

# Déetecter les homologues 2

- Approche simple :
 

Faire un alignement global entre la séquence  $q$  et chaque séquence  $d$  dans la base de données  $D$
- MAIS :
  - parfois seulement un segment de  $q$  est vraiment similaire à un segment dans la séquence  $d$
  - ou les deux séquences ont des motifs similaires et pas des segments d'acides aminés
  - ou l'ordre des segments ou domaines dans  $q$  et  $d$  n'est pas le même, mais il y a des similarités entre les domaines
- Pour ces raisons on a besoin d'un alignement local

5

5

# FASTA

*Proc. Natl. Acad. Sci. USA*  
Vol. 85, pp. 2444-2448, April 1988  
Biochemistry

## Improved tools for biological sequence comparison

(amino acid/nucleic acid/data base searches/local similarity)

WILLIAM R. PEARSON\* AND DAVID J. LIPMAN†

\*Department of Biochemistry, University of Virginia, Charlottesville, VA 22908, and †Mathematical Research Branch, National Institute of Diabetes and Digestive and Kidney Diseases, National Institutes of Health, Bethesda, MD 20205

Communicated by Gerald M. Rubin, December 2, 1987 (received for review September 17, 1987)

**ABSTRACT** We have developed three computer programs for the comparison of protein and DNA sequences. They can be used to search sequence data bases, evaluate similarity scores, and identify periodic structures based on local sequence similarity. The FASTA program is a more sensitive derivative of the FASTP program, which can be used to search protein or DNA sequence data bases and can compare a protein sequence to a DNA sequence data base by reading the DNA data base as it is searched. FASTA includes an additional step in the calculation of the initial pairwise similarity score that allows multiple regions of similarity to be joined to increase the score of related sequences. The RDF2 program can be used to evaluate the significance of similarity scores by comparing the observed scores to random sequence composition. The LFasta program can display all the regions of local similarity between two sequences with scores greater than a threshold, using the same scoring parameters and a similar alignment algorithm as a "graphic alignments." In addition, it can be used to allow comparison of DNA variety of alternative scoring

W. Pearson et D.J. Lipman (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci USA 85:2444-2448

7

# Déetecter les homologues 3

- Un alignement local → l'algorithme de Smith-Waterman (L3 programmation dynamique)
- MAIS, cette approche prend trop de temps si la base de données est trop grande
  - En décembre 2009, UniprotKB/TREMBL contient plus de  $10^7$  entrées
  - Dans les années 1980-1990, les ressources de calcul étaient limitées
  - il y avait un besoin de techniques performantes et précises

→FASTA et BLAST

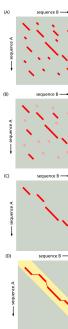
Les PD algorithmes garantissent de trouver les meilleurs alignements. Ceci n'est plus le cas pour les logiciels FASTA et BLAST puisqu'ils sont des heuristiques

6

6

# FASTA

Est un algorithme heuristique qui se compose de 4 étapes distinctes

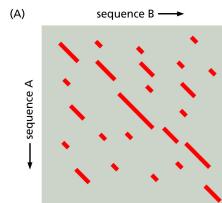


1. Chercher toutes les régions identiques entre deux séquences (des diagonals avec une longueur  $k$ )
2. Prendre les  $m$  meilleures diagonales et prendre les scores pour les tuples en utilisant une matrice de substitution et une pénalité pour les espaces entre les régions
3. Calculer l'alignement optimal en utilisant les régions initiales qui ont les plus hauts scores
4. Utiliser l'algorithme de Smith-Waterman pour l'alignement entre la séquence  $q$  et les séquences  $d$  de  $D$  qui sont au sommet du classement

8

# FASTA étape 1

Chercher toutes les régions identiques entre deux séquences



Un tableau de consultation est construit. Ce tableau contient des informations sur les positions des régions identiques (avec une longueur de  $k$ ) entre deux séquences

$k$  est 2 pour les protéines et 6 pour l'ADN

Le logiciel utilise une technique d'indexation (*hashing et chaining*) pour la construction de ce tableau

J.-P. Dumas et J. Ninio (1981) Efficient algorithms for folding and comparing nucleic acid sequences Nucleic Acid Research 10(1):197-206

9

9

## hashing et chaining 2

TC,CG,GG,GA,AT,TT,TC,CG,GT,TA,AC,CG,GG,GT,TA,AC,CG,GG,GA,AT,TC  
13, 6,10, 8, 3,15,13, 6,11,12, 1, 6,10,11,12, 1, 6,10, 8, 3,13

La séquence est traversée (une fois) remplitant les vecteurs T (tuple) et C (chaîne).

10 20  
TCGGATTCTGT ACGGTACGGAA TC

Le vecteur C a la même taille que la séquence originale

Le vecteur T contient des positions pour chaque tuple de longueur  $k$  :  
pour les séquences d'acides nucléiques la longueur est  $4^k$

T	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21										

11

11

## hashing et chaining

Prenons par exemple la séquence  $q$  suivante (longueur =  $n=22$ ) et  $k=2$

10 20  
TCGGATTCTGT ACGGTACGGAA TC

Cette séquence peut être divisée en  $k$ -tuples superposants

10 20  
TC,CG,GG,GA,AT,TT,TC,CG,GT,TA,AC,CG,GG,GT,TA,AC,CG,GG,GA,AT,TC

Après on choisit un système de numération qui permettra d'enregistrer ces différentes paires dans un autre vecteur

$idx(l) = v(l)_1 4^l + v(l)_2 4^0 \quad v(A) = 0, v(C) = 1, v(G) = 2 \text{ et } v(T) = 3$

10 20  
13, 6, 10, 8, 3, 15, 13, 6, 11, 12, 1, 6, 10, 11, 12, 1, 6, 10, 8, 3, 13

10

10

## hashing et chaining 3

10 20  
TCGGATTCTGT ACGGTACGGAA TC

TC,CG,GG,GA,AT,TT,TC,CG,GT,TA,AC,CG,GG,GT,TA,AC,CG,GG,GA,AT,TC  
13, 6, 10, 8, 3, 15, 13, 6, 11, 12, 1, 6, 10, 11, 12, 1, 6, 10, 8, 3, 13

Le premier tuple (TC) commence à la position 0 dans la séquence et l'index pour ce tuple est 13

Mettez la valeur 0 dans la position 13 du vecteur T

T	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21										

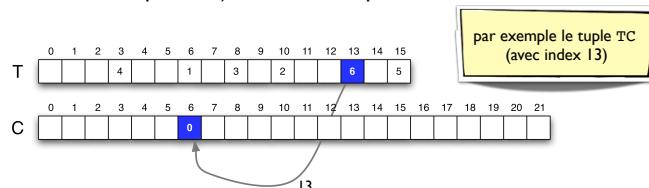
12

## hashing et chaining 4

10 20  
TCGGATTCTG ACAGTACCGA TC  
TC,CG,GG,GA,AT,TT,TC,C<sub>G</sub>,GT,TA,AC,CG,GG,GT,TA,AC,CG,GG,GA,AT,TC  
13, 6,10, 8, 3,15,13, 6,11,12, 1, 6,10,11,12, 1, 6,10, 8, 3,13

La même opération est effectuée pour tous les tuples

Quand on rencontre un tuple (avec index  $i$ ) qui est déjà enregistré dans le vecteur, on met l'ancienne position (qui était enregistrée dans  $T$ ) dans  $C$  (dans la nouvelle position), et la nouvelle position est mise dans  $T$



13

## Chercher les régions identiques

Quand le logiciel a construit le tableau de consultation pour la séquence  $q$ , on peut analyser chaque séquence  $d$  de  $D$  pour détecter les régions correspondantes

Prenons la séquence  $d$  : 10 20  
GCTGATTCCA TGGGTAACCT GA

Comparer chaque k-tuple dans la séquence  $d$  et indiquer s'ils existent aussi dans la séquence originale

10 20  
GC,CT,TG,GA,AT,TT,TC,CC,CA,AT,TG,GG,GG,GT,TA,AA,AC,CC,CT,TG,GA  
9, 7,14, 8, 3,15,13, 5, 4, 3,14,10,10,11,12, 0, 1, 5, 7,14, 8  
  
T 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15  
15 19 16 18 17 13 14 20 5  
  
C 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21  
0 1 0 1 7 2 8 9 10 11 12 3 4 6

15

## hashing et chaining 5

10 20  
TCGGATTCTG ACAGTACCGA TC  
TC,CG,GG,GA,AT,TT,TC,C<sub>G</sub>,GT,TA,AC,CG,GG,GT,TA,AC,CG,GG,GA,AT,TC  
13, 6,10, 8, 3,15,13, 6,11,12, 1, 6,10,11,12, 1, 6,10, 8, 3,13

T	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	15		19			16		18		17	13	14	20		5	

C	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
						0	1				7	2	8	9	10	11	12	3	4	6		

Ce tableau de consultation permet de calculer facilement la récurrence de certaines tuples

Mais il est aussi utilisé pour détecter des régions similaires entre deux séquences

14

## Chercher les régions identiques 2

Chaque fois qu'on trouve une correspondance entre deux tuples on peut calculer la diagonale de la matrice d'alignement sur laquelle on les trouvera

10 20  
GC,CT,TG,GA,AT,TT,TC,CC,CA,AT,TG,GG,GG,GT,TA,AA,AC,CC,CT,TG,GA  
9, 7,14, 8, 3,15,13, 5, 4, 3,14,10,10,11,12, 0, 1, 5, 7,14, 8

Par exemple, le tuple GA qui commence à la position 3 dans la séquence  $q$ , est également présent dans la séquence aux positions 18 et 3

T	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	15		19			16		18		17	13	14	20		5	

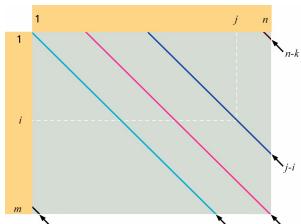
  

C	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
			3			0	1			7	2	8	9	10	11	12	3	4	6			

16

## Chercher les régions identiques 3

Par exemple, le tuple GA qui commence à la position 3 dans la séquence  $d$ , est également présent dans la séquence  $q$  aux positions 18 et 3



ATTENTION: ici le premier indice est 1 et dans l'exemple c'est 0

17

17

## Chercher les régions identiques 5

$$s=4 \text{ et } g(l) = -2*l$$

Par exemple au moment où on ajoute l'alignement GA, le score de la diagonale  $d_0$  devient  $s_0=4$

Au moment où on ajoute ...

$$\text{AT } s_0=4 + \max\{0,4+g(0)\} = 8$$

$$\text{TT } s_0=4 + \max\{0,8+g(0)\} = 12$$

$$\text{TC } s_0=4 + \max\{0,12+g(0)\} = 16$$

$$\text{GG } s_0=4 + \max\{0,16+g(4)\} = 12$$

$$\text{GT } s_0=4 + \max\{0,12+g(0)\} = 16$$

$$\text{TA } s_0=4 + \max\{0,16+g(0)\} = 20$$

$s_{j-i} = s + \max\{0,s_{j-i}+g(l)\}$

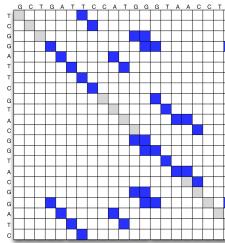
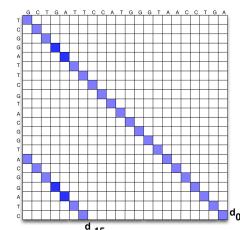
Les  $m$  meilleures diagonales sont utilisées dans la prochaine étape de FASTA

19

19

## Chercher les régions identiques 4

Dans l'exemple, les diagonales sont  $d_0$  et  $d_{-15}$



Tous ces  $k$ -tuples alignés ont le même score  $s$

Un score  $s_{j-i}$  est calculé pour chaque diagonale tout en ajoutant les couples alignées

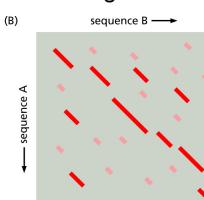
$$s_{j-i} = s + \max\{0,s_{j-i}+g(l)\}$$

18

18

## FASTA étape 2

Prendre les  $m$  meilleures diagonales et calculer les scores de tuples en utilisant une matrice de substitution et une pénalité pour les espaces entre les régions



les réalisations actuelles de FASTA utilise  $m=10$

Pour les protéines on utilise une matrice de substitution comme PAM250

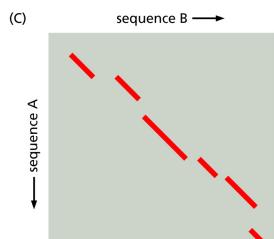
Pour les séquences ADN on utilise +5 pour des assortiments et -4 pour des erreurs d'assortiment

Pour chaque diagonale, le logiciel identifie également une sous-région avec un score maximal = les régions initiales

20

## FASTA étape 3

Calculer l'alignement optimal en utilisant les régions initiales qui ont les plus hauts scores



Dans cette étape, FASTA essaie de combiner les **régions initiales** en utilisant une approche simple de programmation dynamique

par exemple on peut utiliser Smith-Waterman dans la matrice entre la fin et le début de deux régions

On a besoin d'une pénalité de combinaison

Le score total de cet alignement (nommé **le score initial**) est utilisé pour la construction d'un classement préliminaire des séquences dans la base de données

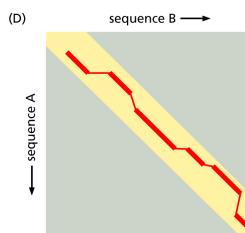
Les régions initiales avec un score au-dessus d'un certain seuil sont utilisées

21

21

## FASTA étape 4

Utiliser l'algorithme de Smith-Waterman (SW) pour l'alignement entre la séquence  $q$  et les séquences  $d$  de  $D$  qui sont au sommet du classement



Les anciennes version de FASTA appliquaient l'algorithme de SW qui calculait les scores pour une région limitée à une bande autour de la région initiale avec le plus haut score

La taille de cette bande est de 16 positions

Dans la nouvelle version (FASTA3) l'algorithme est appliquée à la matrice complète (pour les protéines)

Le score de cet alignement (nommé **le score optimal**) est utilisé pour faire un dernier classement des séquences

22

22

## Pertinence statistique 2

RDF (RDF2) est un logiciel de Pearson et Lipman qui fait la comparaison entre la séquence  $q$  et des permutations aléatoires de la séquence  $d$

D.J. Lipman et W.R. Pearson (1985) Rapid and sensitive protein similarity searches. Science 227:1435-1441

La pertinence est exprimée par un *z-score*

$$z = SS - (\langle SA \rangle) / \sigma^2 \quad \langle SA \rangle = \frac{1}{N} \sum_{i=1}^N SA_i$$

On pourrait aussi utiliser la moyenne ( $\langle SS \rangle$ ) et l'écart type des scores initiaux de tous les séquences  $d$  de la base de données  $D$  pour calculer  $z$

il y a seulement une différence quand la composition de la séquence est non-uniforme

24

24

## Pertinence statistique

Le logiciel trouvera toujours une séquence  $d$  au dessus de la liste des séquences de  $D$

**Mais il n'y a aucune garantie que la relation entre  $q$  et  $d$  est biologiquement significative !!!**

Comment déterminer la pertinence biologique?

Faire une estimation du score (score aléatoire ou  $SA$ ) entre la séquence  $q$  et des séquences aléatoires.

Si la différence entre le score de similarité ( $SS$ ), obtenu pour  $q$  et  $d$ , et le  $SA$  est grande, la relation statistique entre  $q$  et  $d$  est significative

23

23

# Pertinence statistique 3

De leur expérience, Pearson et Lipman proposent les directives suivantes

- Si  $z > 3$  la relation est peut-être significative
- Si  $z > 6$  la relation est probablement significative
- Si  $z > 10$  la relation est significative

Le logiciel RDF2 donne la possibilité de brouiller la séquence localement ou globalement

25

25

# BLAST

*J. Mol. Biol.* (1990) 215, 403–410

## Basic Local Alignment Search Tool

Stephen F. Altschul<sup>1</sup>, Warren Gish<sup>1</sup>, Webb Miller<sup>2</sup>  
Eugene W. Myers<sup>3</sup> and David J. Lipman<sup>1</sup>

<sup>1</sup>National Center for Biotechnology Information  
National Library of Medicine, National Institutes of Health  
Bethesda, MD 20894, U.S.A.

<sup>2</sup>Department of Computer Science  
The Pennsylvania State University, University Park, PA 16802, U.S.A.

<sup>3</sup>Department of Computer Science  
University of Arizona, Tucson, AZ 85721, U.S.A.

(Received 26 February 1990; accepted 15 May 1990)

A new approach to rapid sequence comparison, basic local alignment search tool (BLAST), directly approximates alignments that optimize a measure of local similarity, the maximal segment pair (MSP) score. Recent mathematical results on the stochastic properties of MSP scores allow an analysis of the performance of this method as well as the statistical significance of alignments it generates. The basic algorithm is simple and robust; it can be implemented in a number of ways and applied in a variety of contexts including straight-forward DNA and protein sequence database searches, motif searching, gene identification, secondary structure prediction and in the analysis of molecular evolution of similarities in long DNA sequences. In BLAST is an order of magnitude faster than FASTA.

S.F.Altschul et al (1990) Basic local alignment search tool. *J Mol Biol*  
215:403-410

27

# Sensibilité et sélectivité

**Sensibilité** : le système trouvera même les protéines qui sont des homologues éloignés

évitez les faux négatifs

**Sélectivité** : le système trouvera seulement les séquences qui sont apparentées

évitez les faux positifs

Il y a une dépendance entre la sensibilité et la sélectivité

En augmentant la sensibilité, on risque à diminuer la sélectivité (et vice versa)

26

26

# BLAST

La structure du logiciel BLAST est équivalente au logiciel FASTA

Il y a deux différences importantes entre les deux systèmes

Dans la première étape, FASTA cherche des k-tuples dans chaque séquence  $d$  qui sont identiques. BLAST cherche des des k-tuples qui ont un score au dessus d'un seuil  $T$

Dans la première étape, FASTA utilise un tableau de consultation. BLAST peut utiliser la même structure de données, mais ils ont découvert qu'un automate fini déterministe était plus efficace

28

# Quelques définitions

**Un segment** est une sous-séquence avec une certaine taille au sein d'une autre séquence

**Un mot** est un segment avec une taille  $w$

**Paire de segments avec un haut score** (PHS) est une paire de segments alignées pour lesquelles le score ne peut pas être augmenté en étendant l'alignement

29

29

## BLAST étape I

Pour chaque séquence  $d$  en  $D$ , cherchez tous les mots de la séquence  $d$  qui ont un score d'au moins  $T$  après l'alignement avec les mots de la séquence  $q$

On appelle chaque paire de mot remplissant cette condition un *hit*

Dans le logiciel initial, la même technique d'indexation (*hashing et chaining*) que FASTA était utilisée pour enregistrer les mots de  $q$  et chercher les paires de mot

Pour les protéines  $w=3$  et pour les séquences d'acides nucléiques  $w=11$

**MAIS** Ils ont trouvé que la performance des automates finis déterministes est supérieure

31

31

# BLAST

## Les 4 étapes de BLAST

1. Pour chaque séquence  $d$  dans  $D$ , **chercher tous les mots** de la séquence  $d$  qui ont un score d'au moins  $T$  après l'alignement avec les mots de la séquence  $q$
2. **Chercher les PSH:** prenez les paires de mots alignés et essayez de les étendre les deux extrémités
3. Utiliser la programmation dynamique pour calculer les alignements **avec des espaces** pour chaque PSH
4. Retracer les chemins pour tous les alignements (avec des espaces) trouvés à l'étape précédente

30

30

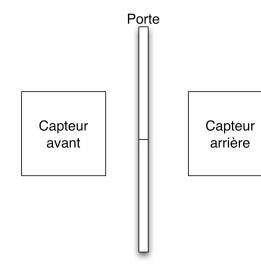
## Automate fini déterministe

Un **automate fini déterministe** (AFD) est un des modèles les plus simples pour représenter le calcul .

Ce sont des modèles pour des ordinateurs avec une mémoire limitée



portes automatiques



32

## Automate fini déterministe 2

33

33-1

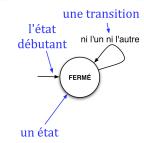
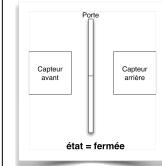
## Automate fini déterministe 2

33

33-3

## Automate fini déterministe 2

1

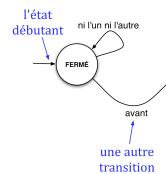
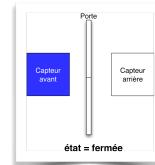


33

33-2

## Automate fini déterministe 2

2



33

33-4

## Automate fini déterministe 2

33

33-5

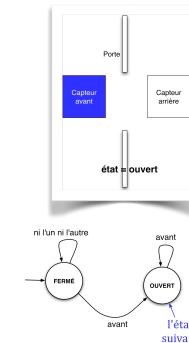
## Automate fini déterministe 2

33

33-7

## Automate fini déterministe 2

3

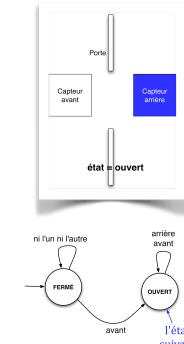


33

33-6

## Automate fini déterministe 2

4



33

33-8

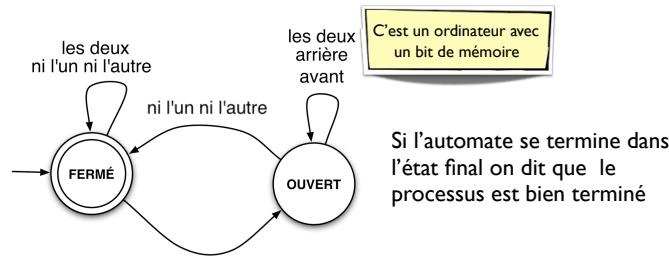
## Automate fini déterministe 2

33

33-9

## Automate fini déterministe 3

On appelle la figure un [diagramme d'état](#)



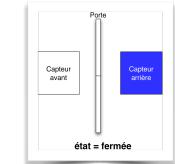
Les automates fini sont des outils utiles pour la [détection des motifs dans des données](#)

l'entrée du AFD est ici  
l'état des capteurs

34

## Automate fini déterministe 2

5



33

33-10

## Automate fini déterministe 4

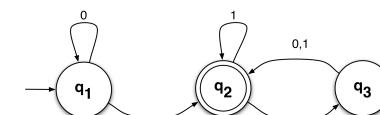
Pour l'analyse des séquences, l'entrée est une séquence de caractères

Prenons par exemple l'automate  $M1$  suivant:

Quand l'automate reçoit l'entrée 1101, il traite chaque caractère

Si l'automate se termine à  $q_2$ , la séquence est acceptée par l'automate

Autrement elle est rejetée



Est-ce que tu peux trouver des autres séquences qui sont acceptées par l'automate?

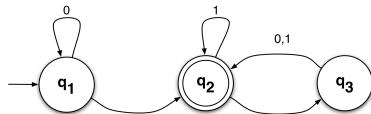
Quels est la langue qui contient toutes les chaînes de caractères qui sont acceptées par  $M1$ ?

35

35

## Automate fini déterministe 5

Une définition précise de l'ADF :



Un automate fini déterministe est un 5-tuple  $(Q, \Sigma, \delta, q_0, F)$

$Q$  est un ensemble fini d'états

$\Sigma$  est un ensemble fini de caractères, nommé l'alphabet

$\delta: Q \times \Sigma \rightarrow Q$  est la fonction de transition

$q_0 \in Q$  est l'état de départ

il y a un seul état de départ !

$F \subseteq Q$  est l'ensemble des états finals

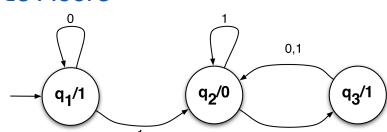
36

36

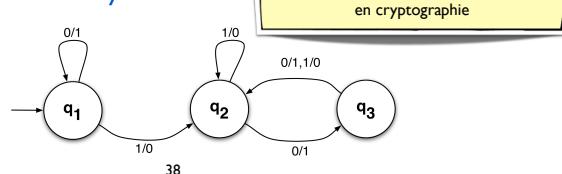
## Finite state transducers

Ce sont des machines qui produisent des symboles de sortie

Les machines de Moore



Les machines de Mealey



38

## Automate fini déterministe 6

Nous pouvons décrire l'automate  $M_I$  formellement comme

$M_I = (Q, \Sigma, \delta, q_1, F)$  dans lequel

$Q = \{q_1, q_2, q_3\}$

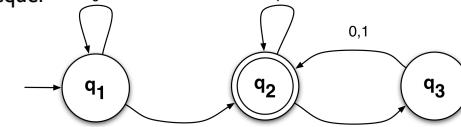
$\Sigma = \{0, 1\}$

$\delta$  est

	0	1
q1	q1 q2	q2 q3
q2	q3 q2	q2
q3	q2	q2

$q_1$  est l'état de départ

$F = \{q_2\}$



Si  $A$  est l'ensemble de toutes les chaînes de caractères acceptées par l'automate  $M$ , on dit que  $A$  est le langage de la machine  $M$

Si la machine n'accepte aucune chaîne de caractères, il connaît toujours le langage vide  $\emptyset$

37

37

## Finite state transducers 2

Ça change la définition de l'ADF !

il n'y a plus un ensemble d'états finaux

et on ajoute maintenant :

$\Lambda$  est un ensemble fini de caractères, nommé l'alphabet de sortie

$\lambda: Q \times \Sigma \rightarrow \Lambda$  est la fonction de sortie → pour les machines de Mealey

$\lambda: Q \rightarrow \Lambda$  est la fonction de sortie → pour les machines de Moore

Un finite state transducer est un 6-tuple  $(Q, \Sigma, \Lambda, \delta, \lambda, q_0)$

M. Cameron et al (2006) A deterministic finite automaton for faster protein hit detection in BLAST. Jour Comp Biol 13(4): 965-978

39

## BLAST étape I 2

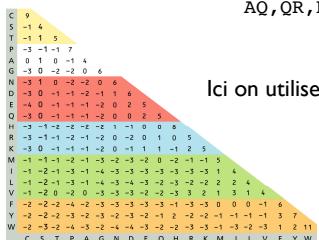
Prendre la séquence  $q$  suivante (longueur =  $n=10$ ) et  $k=2$

AQRQRQQARQ

L'alphabet est {A, Q, R} et la taille est  $\alpha=3$

Cette séquence est divisée dans les  $k$ -tuples suivants:

AQ, QR, RQ, QR, RR, RQ, QA, AR, RQ



Ici on utilise BLOSUM 62 pour calculer les scores

Chercher tous les mots de taille  $k$  (ici  $3^2=9$  possibilités) qui ont un score plus grand que  $T$  (ici  $T=4$ )

40

40

## BLAST étape I 4

Chaque préfixe avec une taille  $k-1$  est un état dans l'automate



Dans l'exemple il y a 3 états



42

42

## BLAST étape I 3

Tous les mots avec un score plus grand que 4 sont acceptés

	AQ	QR	RQ	RR	QA	AR
AA	3	-2	-2	-2	3	3
AQ	9	0	4	0	-2	5
AR	5	4	0	4	-2	9
QA	-2	4	0	0	9	-2
QQ	4	6	6	2	4	0
QR	0	10	2	6	4	4
RA	-2	0	4	4	5	-2
RQ	4	2	10	6	0	0
RR	0	6	6	10	0	4

Les éléments en bleu représentent les associations identiques

les éléments en rouge sont les associations similaires

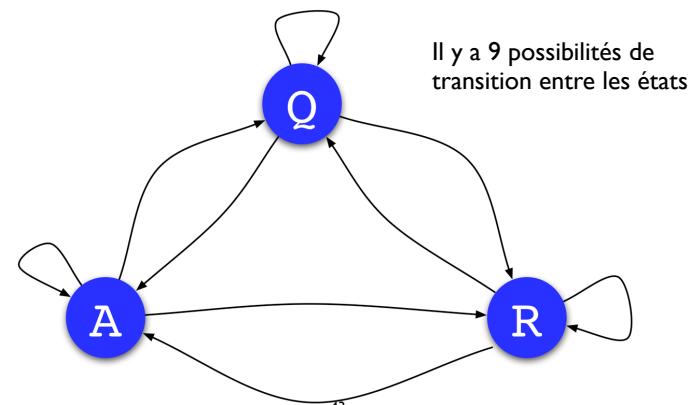
Comment construire la machine de Mealey?

41

41

## BLAST étape I 5

Chaque état a des transitions à  $\alpha$  autres états



Il y a 9 possibilités de transition entre les états

43

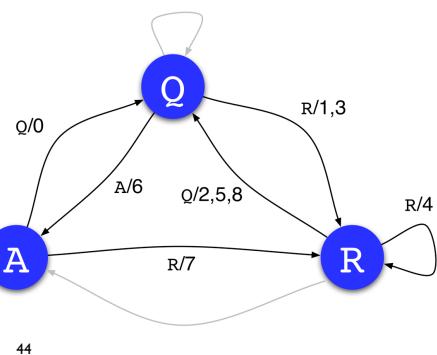
## BLAST étape I 6

L'alphabet de sortie contient les positions de départ pour les mots de taille  $k$  dans la séquence  $q$

0 . 2 . 4 . 6 . 8 .

$$q = \text{AQRQRRQARQ}$$

	AQ	QR	RQ	RR	QA	AR
AA	3	-2	-2	-2	3	3
AQ	9	0	4	0	-2	
AR	5	4	0	4	-2	9
QA	-2	4	0	0	9	-2
QQ	4	6	6	2	4	0
QR	0	10	2	6	4	4
RA	-2	0	4	4	-2	
RQ	4	2	10	6	0	0
RR	0	6	6	10	0	4



44

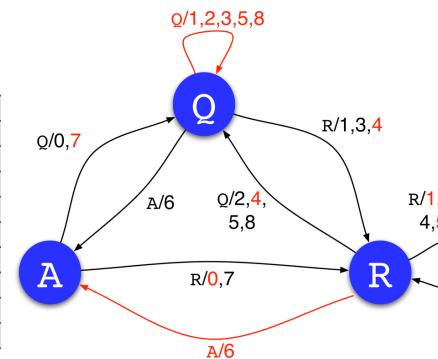
## BLAST étape I 7

L'alphabet de sortie contient les positions de départ pour les mots de taille  $k$  dans la séquence  $q$

0 . 2 . 4 . 6 . 8 .

$$q = \text{AQRQRRQARQ}$$

	AQ	QR	RQ	RR	QA	AR
AA	3	-2	-2	-2	3	3
AQ	9	0	4	0	-2	5
AR	5	4	0	4	-2	9
QA	-2	4	0	0	9	-2
QQ	4	6	6	2	4	0
QR	0	10	2	6	4	4
RA	-2	0	4	4	-2	
RQ	4	2	10	6	0	0
RR	0	6	6	10	0	4

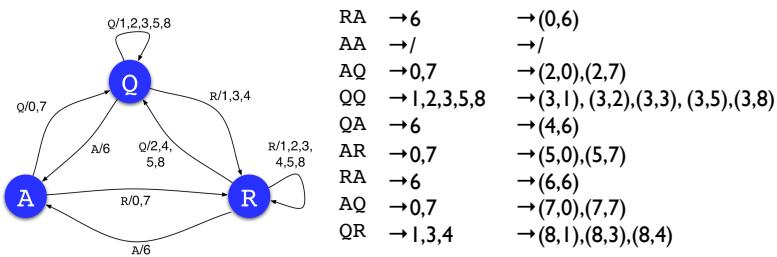


45

## BLAST étape I 8

En utilisant la machine de Mealey on peut trouver les “hits” dans chaque séquence  $d$  dans la base de données  $D$

$$d = \text{RAAQQARAQR}$$



46

46

## BLAST étape I 8

En utilisant la machine de Mealey on peut trouver les “hits” dans chaque séquence  $d$  dans la base de données  $D$

$$d = \text{RAAQQARAQR}$$

	R	A	A	Q	Q	A	R	A	Q	R
A	■	■	■	■	■	■	■	■	■	■
Q	■	■	■	■	■	■	■	■	■	■
R	■	■	■	■	■	■	■	■	■	■
Q	■	■	■	■	■	■	■	■	■	■
R	■	■	■	■	■	■	■	■	■	■
R	■	■	■	■	■	■	■	■	■	■
Q	■	■	■	■	■	■	■	■	■	■
A	■	■	■	■	■	■	■	■	■	■
R	■	■	■	■	■	■	■	■	■	■
Q	■	■	■	■	■	■	■	■	■	■

RA	→ 6	→ (0,6)
AA	→ /	→ /
AQ	→ 0,7	→ (2,0),(2,7)
QQ	→ 1,2,3,5,8	→ (3,1),(3,2),(3,3),(3,5),(3,8)
QA	→ 6	→ (4,6)
AR	→ 0,7	→ (5,0),(5,7)
RA	→ 6	→ (6,6)
AQ	→ 0,7	→ (7,0),(7,7)
QR	→ 1,3,4	→ (8,1),(8,3),(8,4)

les paires  $(j,i)$  sont utilisées dans la deuxième étape de BLAST

47

47

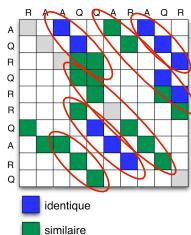
## BLAST étape 2

Chercher les PSH :

Comme dans le logiciel FASTA on calcule les diagonals

Prendre chaque *hit* et essayer d'étendre les deux extrémités  
Arrêter quand le score  $S$ , obtenu avec le segment devient moins que  $S-X$

Depuis l'article publié en 1997 dans NAR, on essaie avant d'étendre les segments, de combiner des *hits* qui sont sur la même diagonale



Par exemple, si la distance entre 2 *hits* est plus petite ou égale à 4, les deux *hits* sont fusionnés

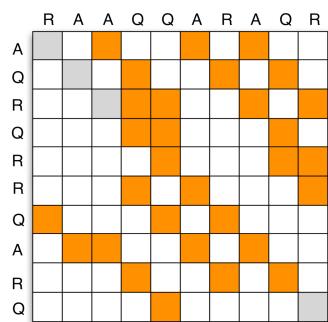
Tous les ensembles qui sont indiqués sur la figure sont des *hits* combinés

48

48

## BLAST étape 2 4

On peut aussi enregistrer le score ( $S$ ) et la taille ( $T$ ) en même temps



	I	P	S	T
A	-6	6	5	2
Q	-5	7	10	3
R	-2	5	15	4
Q	0	3	11	6
R	1	2	6	2
Q	2	0	10	3
A	4	4	6	2
R	5	0	15	5
Q	7	0	14	3

Tous les *hits* complets

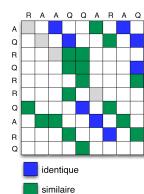
50

50

## BLAST étape 2 3

Comment combine-t-on les *hits*?

$RA \rightarrow (0,6)$   
 $AA \rightarrow /$   
 $AQ \rightarrow (2,0),(2,7)$   
 $QQ \rightarrow (3,1),(3,2),(3,3),(3,5),(3,8)$   
 $QA \rightarrow (4,6)$   
 $AR \rightarrow (5,0),(5,7)$   
 $RA \rightarrow (6,6)$   
 $AQ \rightarrow (7,0),(7,7)$   
 $QR \rightarrow (8,1),(8,3),(8,4)$



1. Calculer la diagonale pour chaque paire  $(j,i)$ :  
par exemple :  $diag(RA)=j-i=0-6=-6$
2. Enregistrer la position de départ (de  $q$ ) dans un tableau  $map$  qui est indexé par la diagonale  
 $RA : map(-6)=6$   
 $AQ : map(2)=0 ; map(-5)=7$   
 $QQ : map(2) = 1 !!!!$

3. Si l'index dans le tableau  $map$  est déjà occupé, calculer la distance entre les positions de départ dans  $q$  des deux hits:  
 $AQ$  commence à position 0 et  $QR$  commence à la position 1 :  $1-0 \leq 4$   
→ combiner les deux *hits* et recalculer le score en utilisant BLOSUM 62  
 $Score(AQQ,AQR) = 4+5+1=10$

49

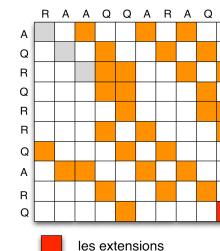
49

## BLAST étape 2 5

Prendre chaque *hit* et essayer d'étendre les deux extrémités  
Arrêter quand le score  $S$ , obtenu avec le segment devient moins que  $S-X$

Dans cette étape les insertions et les deletions ne sont pas considérées

Si on prend  $X=I$ , les PSH suivants sont obtenu :



	I	P	S	T
A	-6	6	5	2
Q	-5	7	10	3
R	-2	5	15	4
Q	0	3	12	7
A	1	2	6	2
R	2	0	10	3
Q	4	4	6	2
A	5	0	15	5
R	7	0	14	3

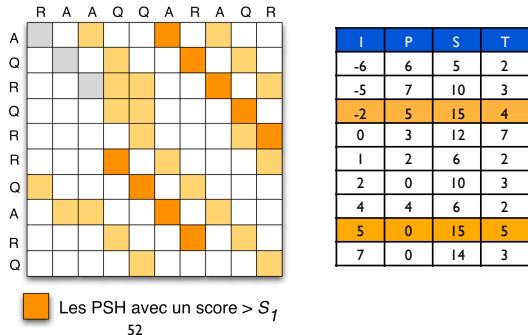
51

51

## BLAST étape 3

Tous les alignements sans espace qui ont un score plus grand que  $S_1$  sont traités dans cet étape

Ici on prend  $S_1 > 14$



52

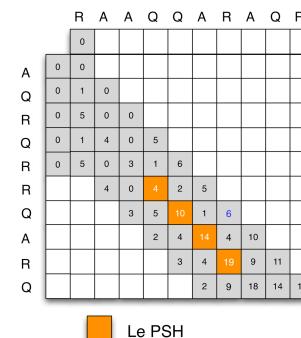
## BLAST étape 3 2

Utiliser la programmation dynamique (l'algorithme de Smith et Waterman) pour chercher un alignement [avec des espaces](#) qui contient ce PSH

Comme dans le logiciel FASTA on peut limiter la distance autour le PSH

l'idée est qu'on impose une limite sur le nombre d'insertions et suppressions

Smith-Waterman avec une pénalité linéaire pour les espaces (g) de -10



53

## BLAST étape 3 3

Ou la méthode X-drop peut être utilisée

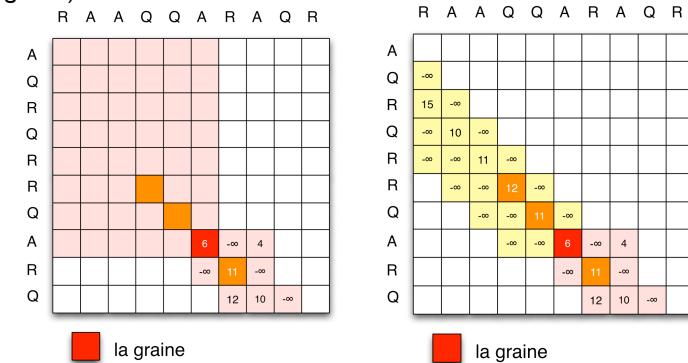
En calculant les éléments le long d'une ligne, la méthode X-drop s'arrête et recommence à la ligne suivante quand les valeurs du score tombent X au dessous du meilleur score

Z.Zhang et al (2000) A greedy algorithm for aligning DNA sequences Jour Comp Biol 7(1/2):203-214

54

## BLAST étape 3 5

Dans notre exemple on commence à un élément du PSH (la graine)



55

## BLAST étape 4

Tracer les chemins pour tous les alignements (avec des espaces) trouvés à l'étape précédente

	R	A	A	Q	Q	A	R	A	Q	R
A	0	0								
Q	0	1	0							
R	0	5	0	3	1	6				
R	0	5	0	3	1	6				
R		4	0	4	2	5				
Q		3	5	10	1	6				
A		2	4	14	4	10				
R		3	4	19	9	11				
Q		2	9	18	14	12				

56

Pour la première méthode on obtient l'alignement suivant:

QRQRR-QAR  
-RAA-QQAR

## Pertinence statistique

Est-ce que la séquence  $d$  avec un score  $S$  trouvé dans la base de données  $D$  est vraiment un homologue?

Si on veut comprendre la pertinence statistique du résultat Il y a deux questions qu'on doit répondre:

1. Quelle est la probabilité que le score d'au moins  $S$  se produise par hasard?
2. À combien d'associations par hasard peut-on s'attendre quand on lance une recherche dans une base de données?

58

58

## BLAST étape 4 2

Si on utilise X-drop, on doit tracer les chemins en deux directions

	R	A	A	Q	Q	A	R	A	Q	R
A	0	0								
Q	0	1	0							
R	0	5	0	3	1	6				
R	0	5	0	3	1	6				
R		4	0	4	2	5				
Q		3	5	10	1	6				
A		2	4	14	4	10				
R		3	4	19	9	11				
Q		2	9	18	14	12				

	R	A	A	Q	Q	A	R	A	Q	R
A	0	0								
Q	0	1	0							
R	0	5	0	3	1	6				
R	0	5	0	3	1	6				
R		4	0	4	2	5				
Q		3	5	10	1	6				
A		2	4	14	4	10				
R		3	4	19	9	11				
Q		2	9	18	14	12				

RAAQQARQ  
RQRQRQAR-

57

## Pertinence statistique 2

En utilisant BLOSUM62

$$S = \sum_u (S_{a,b})_u$$

R	L	A	S	V	E	T	D	M	P	L	T	L	R	Q	H
.		.		:	:		.	:	.	.	.		.	.	
T	L	T	S	L	Q	T	T	L	K	A	H	L	G	T	H

$-1+4+0+4+1+2+5-1+2-1-1-2+4-2-1+8=21$

Quelle est la pertinence statistique de ce score?

59

# Pertinence statistique 3

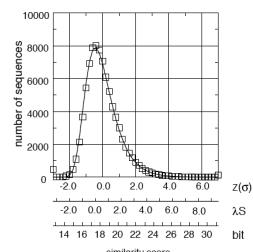
La distribution des scores est un *extreme value distribution* (une distribution de valeurs extrême ou DVE)

Quand on cherche des homologues dans des bases de données, on cherche toujours les meilleures séquences

Pour chaque alignement, on cherche toujours les alignements qui donnent le score le plus grand.

la distribution n'est pas gaussienne !  
C'est une DVE

Figure 14: The extreme value distribution



W.P.Pearson (2000) ISMB tutorial: Protein sequence comparison and protein evolution

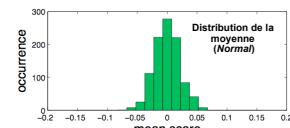
60

60

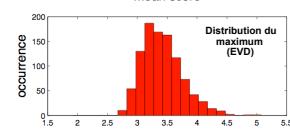
# Pertinence statistique 5

D'où vient cette DVE?

La distribution des valeurs moyennes est une *distribution normale*



MAIS: la distribution des valeurs maximales est une *DVE*



un *run* = une comparaison dans FASTA ou BLAST

62

62

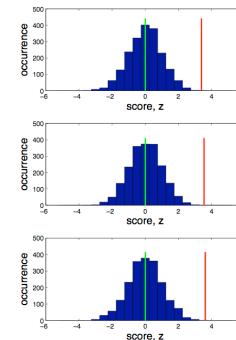
# Pertinence statistique 4

D'où vient cette DVE?

Répéter 1000 fois la simulation suivante (3 exemples sont montrées à droite):

Echantillonner 1000 valeurs ( $z$ ) d'une distribution normale

Dans chaque *run*, on calcule la *moyenne* et le *maximum*

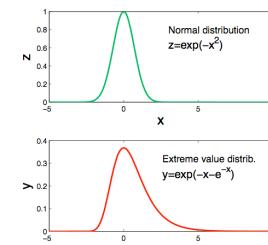


61

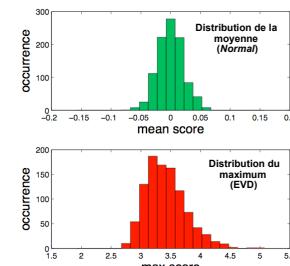
61

# Pertinence statistique 6

D'où vient cette DVE?



Les distributions théoriques

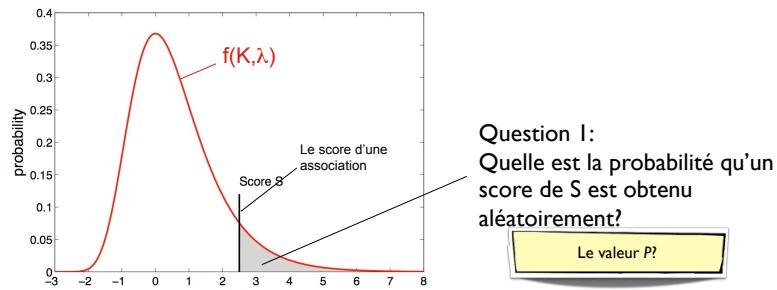


La DVE représente la distribution des scores qu'on peut attendre quand on fait l'alignement entre deux séquences qui ne sont pas apparentées.

63

## Pertinence statistique 7

La distribution montre la probabilité qu'on obtient un certain score par hasard



64

64

## Pertinence statistique 9

Comment déterminer la *valeur P* d'une paire de segments alignés (PSA)?

Karlin et Altschul ont aussi proposé d'utiliser un score de bit  $S'$

$$S' = \frac{\lambda S - \ln(K)}{\ln(2)}$$

$$\begin{aligned} Pval_S^{MSP} &= Ke^{-\lambda S} \\ &= Ke^{-\ln(2)S' + \ln(K)} \\ &= 2^{-S'} \end{aligned}$$

On peut maintenant calculer simplement la *valeur P* du bit-score

66

66

## Pertinence statistique 8

Comment déterminer la *valeur P* d'une paire de segments alignés (PSA)?

La *valeur P* d'une PSA avec score  $S$  est la probabilité qu'on obtient ce score d'au moins de  $S$  aléatoirement

$$\begin{aligned} Pval_S^{MSP} &= P(X \geq S) \\ &= 1 - \exp(-Ke^{-\lambda S}) \\ &\approx Ke^{-\lambda S} \end{aligned}$$

S. Karlin et F.L. Altschul (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proc Natl Acad Sci USA 87:2264-2268

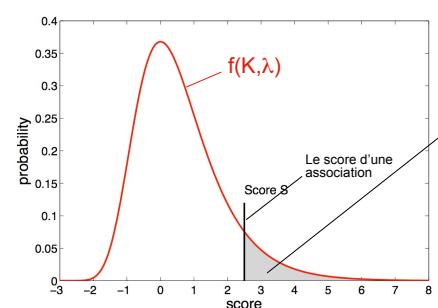
Les valeurs  $K$  et  $\lambda$  sont dépendantes de la matrice de substitution qu'on a utilisé pour l'alignement (elles étaient estimées indépendamment)

65

65

## Pertinence statistique 10

La distribution des probabilités qu'on ait un certain score par hasard



Question 1:  
Quelle est la probabilité qu'un score de  $S$  soit obtenu aléatoirement?

Réponse :  $P-val = 2^{-S'}$

Question 2:  
Combien associations aléatoires peut-t-on trouver quand on cherche dans une base de données?

67

# Pertinence statistique 11

La valeur  $E$  donne la réponse sur la question.

Lors de la recherche des homologues dans une base de données, on divise la séquence  $q$  et toutes les séquences  $d$  de  $D$  en mots (avec la taille  $k$ )

Chaque mot de la séquence  $q$  est comparée à tous les autres mots dans la base de données  $D$

Pour des séquences de taille  $m$  et une base de données qui contient  $n$  séquences, l'espace de recherche devient  $N = mn$

La valeur  $E$  est le nombre d'associations qu'on obtient aléatoirement dans une base de données avec un certain nombre de séquences

$$\begin{aligned} E &= mn \cdot P_{val} \\ &= Kmne^{-\lambda S} \\ &= NKe^{-\lambda S} \\ &= N/2^S \end{aligned}$$

68

68

# Pertinence statistique 12

Plus la valeur de  $E$  est inférieure, plus l'association entre la séquence  $q$  et la séquence  $d$  est significatif

Une valeur de  $E$  haute ( $> 1$ ) indique qu'on ne peut pas avoir confiance en cette association

FASTA et BLAST introduit un seuil sur la valeur  $E$

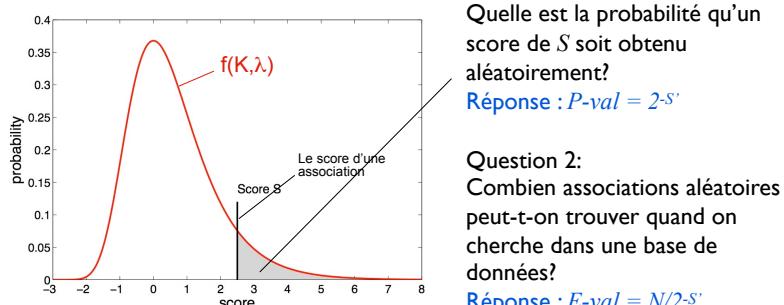
Ce seuil est  $10 =$  chaque requête renvoie 10 associations aléatoirement = 10 faux positifs

69

69

# Pertinence statistique 13

La distribution des probabilités qu'on ait un certain score par hasard



70

70