Comparação e alinhamento de sequências

Comparar sequências

- A comparação de sequências de proteínas ou DNA/RNA é uma ferramenta essencial na procura da existência de **relações** de semelhança entre o todo ou parte dessas sequências, na avaliação da sua **proximidade evolutiva** e na identificação de regiões importantes na manutenção da **estrutura** e **função**

- Alinhamento e comparação são problemas que podem ser expressos de forma **matemática** e para os quais existem **algoritmos** robustos, mas:

- A **parametrização** do problema deverá reflectir o nosso conhecimento biológico (escolha das funções de **score**, **gap penalties** e outros parâmetros que determinam as soluções oferecidas pelos algoritmos).

Para quê comparar sequências?

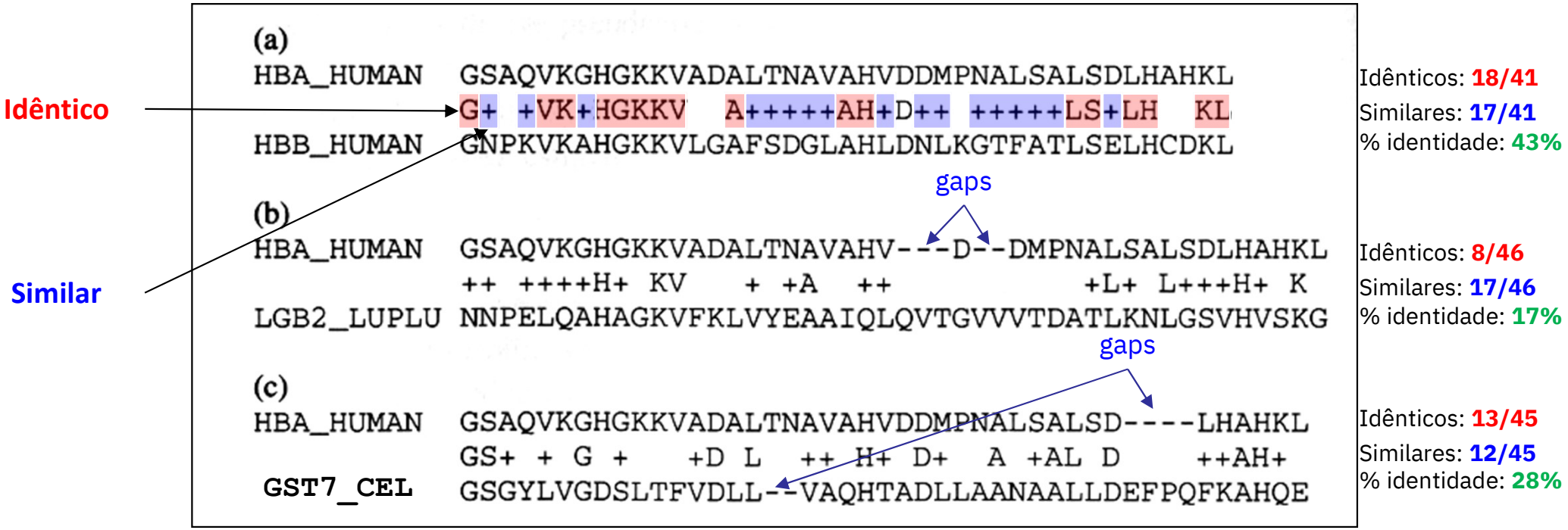
- Identificação de **regiões conservadas** entre duas ou mais sequências evidencia zonas importantes para a **estrutura** e/ou **função** das proteínas correspondentes.

- Estimativa da **distância evolutiva** entre os organismos dos quais provêm as sequências: maior disparidade das sequências geralmente reflecte uma maior divergência evolutiva

- Identificar, de entre as sequências presentes numa base de dados, aquelas que possuem semelhança **significativa** com uma determinada sequência de busca (identificação de **homólogos**)

- Identificação de uma sequência a partir de um **fragmento**

Comparar sequências não é trivial



(a) Sequências **muito aparentadas:** cadeias α e β-hemoglobina humanas

(b) Sequências **aparentadas:** α-hemoglobina humana e leghemoglobina vegetal

(c) Sequências **NÃO aparentadas:** α-hemoglobina humana e GST-7 de *C. Elegans*

Homologia vs. semelhança

Os termos **homologia**, **semelhança** e **identidade** têm significados distintos no contexto da análise de sequências biológicas:

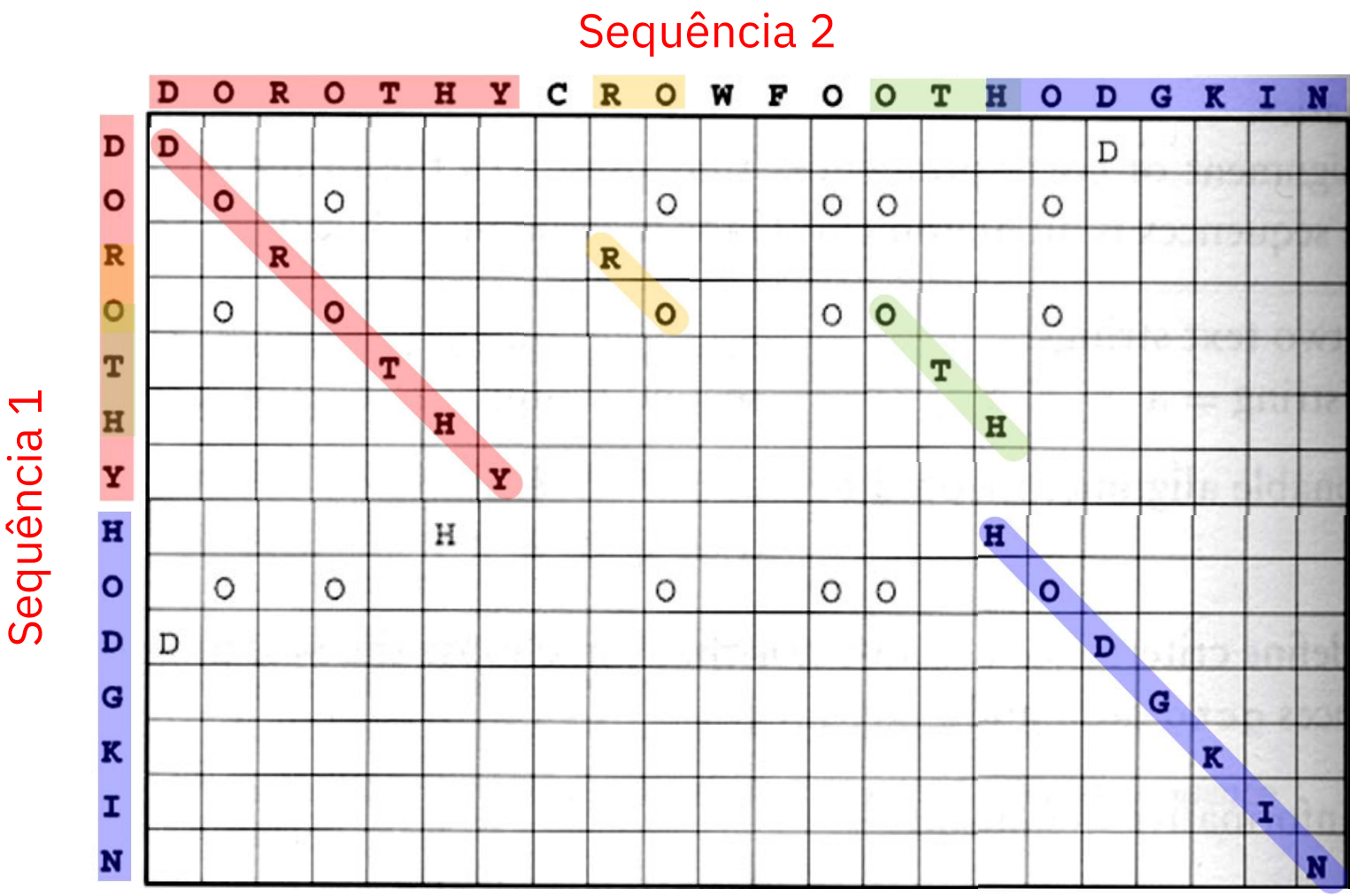
- **Homologia:** descreve um parentesco evolutivo entre duas sequências que poderão corresponder a proteínas de funções homologas em diferentes organismos (exemplo: **citocromo *c* humano** e **citocromo *c* bovino**).

- **Semelhança:** descreve o grau de parecença entre duas sequências, independentemente do seu contexto ou significado biológico. É quantificada através de um método matemático de alinhamento e depende da escolha do "scoring scheme" (**matriz de scoring**).

- **Identidade:** a **percentagem de identidade** entre duas sequências alinhadas é geralmente definida como sendo a razão entre o número total de resíduos idênticos e o número total de resíduos do alinhamento (incluindo gaps).

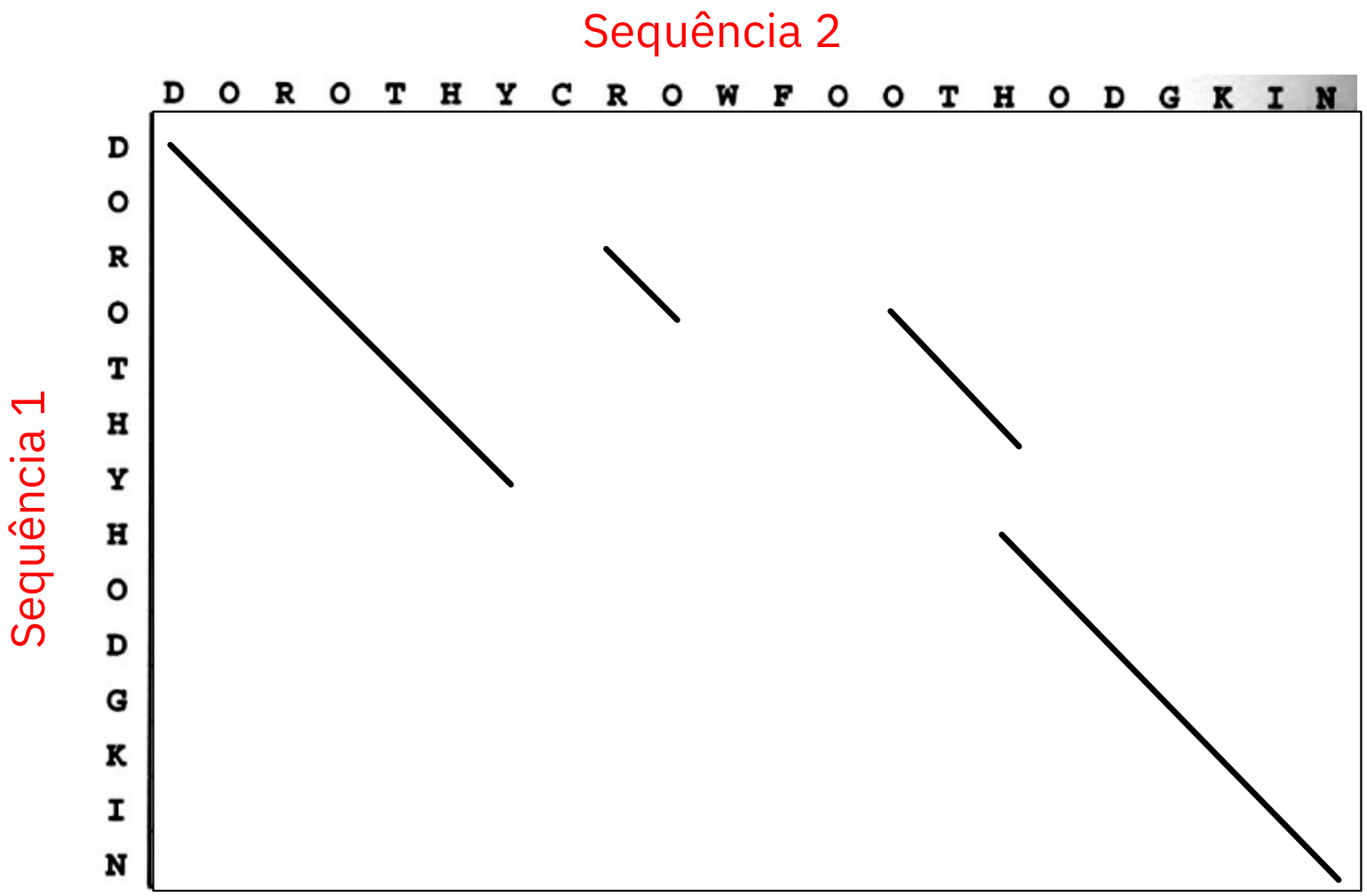
Comparar sequências: dot plots

Um “dot plot” é um modo de comparação de duas sequências baseado na construção de uma matriz de **N** linhas e **M** colunas, em que **N** e **M** são os comprimentos das duas sequências a comparar.

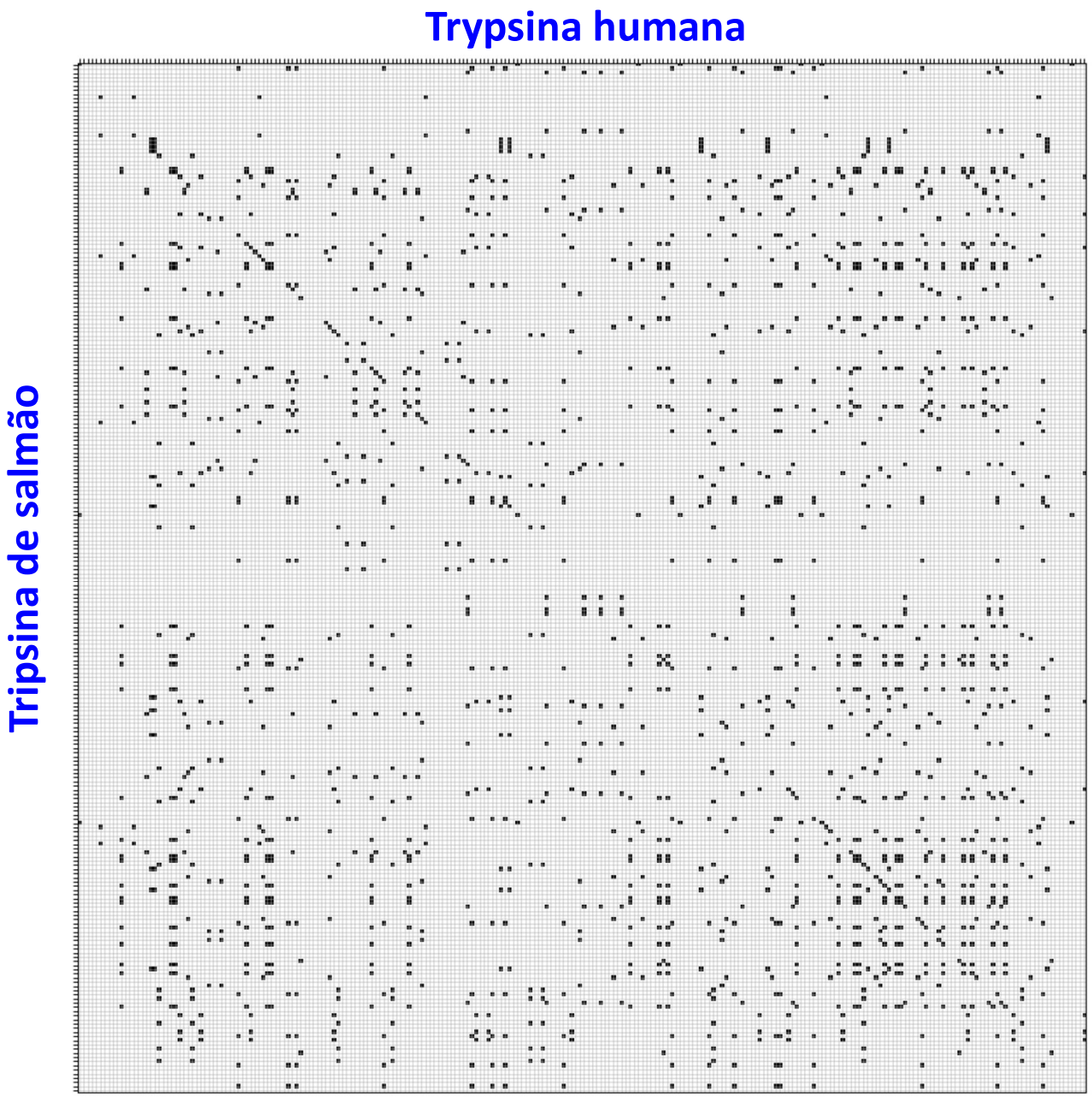


Comparar sequências: dot plots

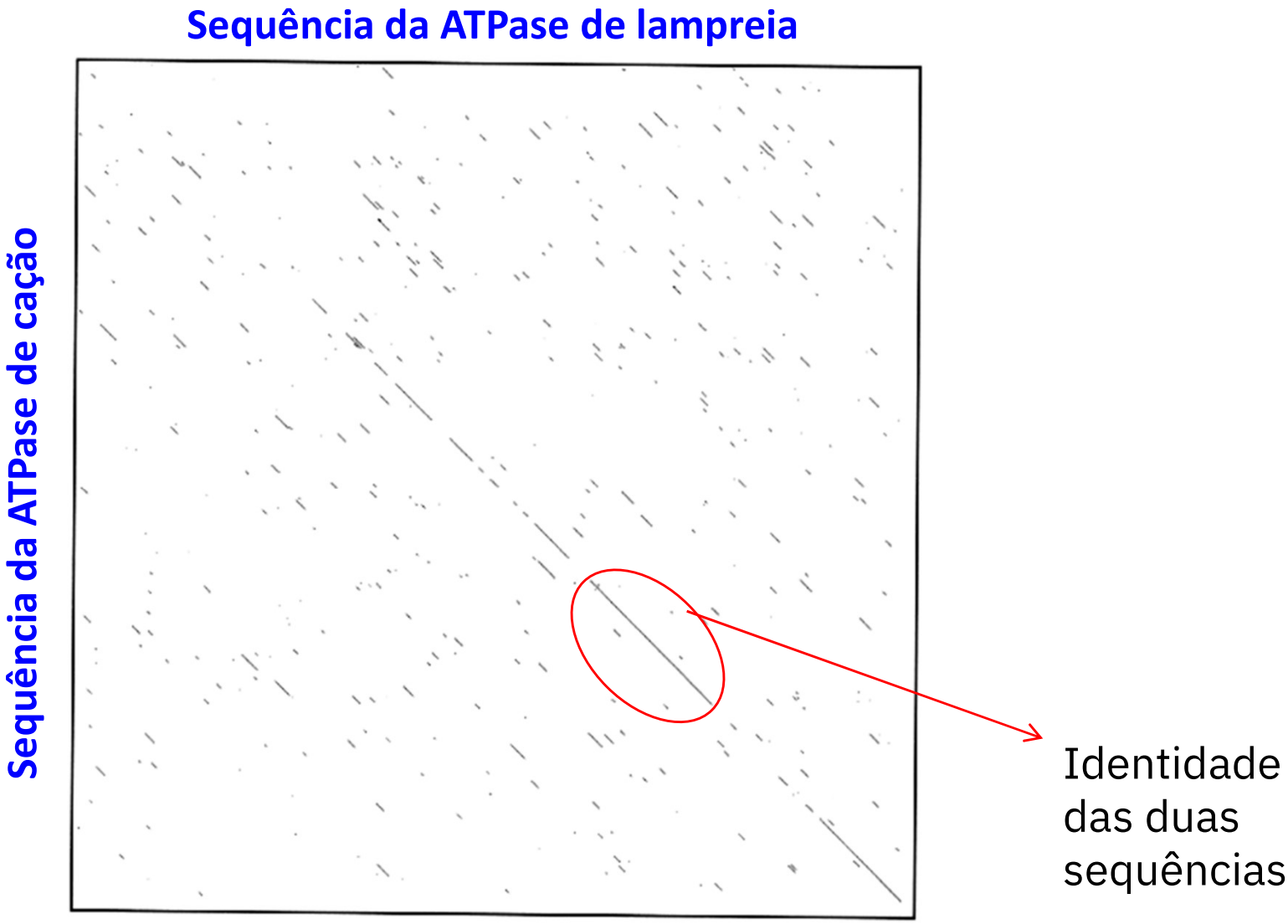
Um “dot plot” é um modo de comparação de duas sequências baseado na construção de uma matriz de **N** linhas e **M** colunas, em que **N** e **M** são os comprimentos das duas sequências a comparar.

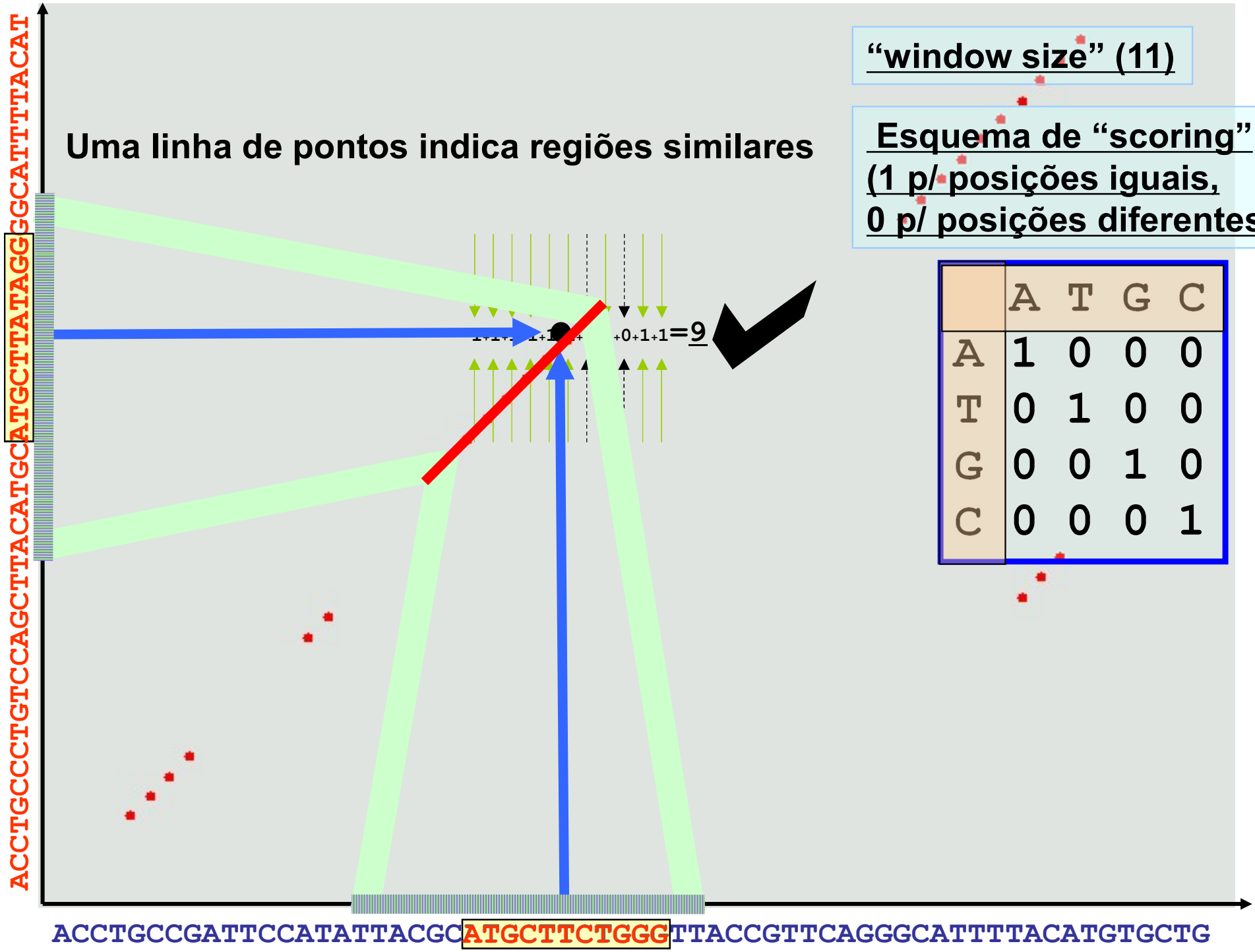


Exemplo de dot plot (1)



Exemplo de dot plot (2)





"window size" (11)

Esquema de "scoring"
(1 p/ posições iguais,
0 p/ posições diferentes)

	A	T	G	C
A	1	0	0	0
T	0	1	0	0
G	0	0	1	0
C	0	0	0	1

Comparações usando “dot plots”

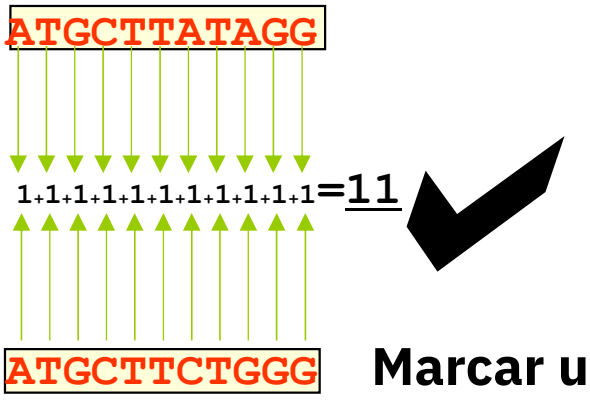
Detecção de correspondências exactas entre regiões

1) **Escolher um esquema de score**

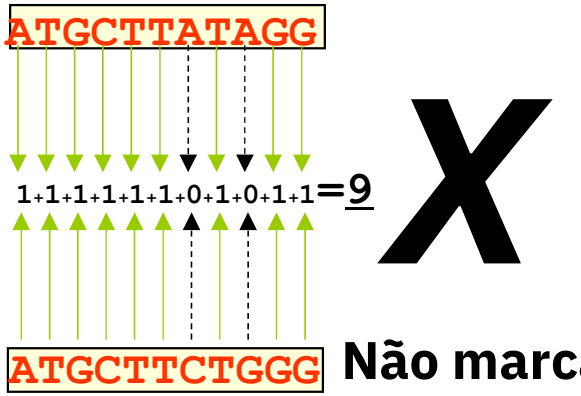
	A	T	G	C
A	1	0	0	0
T	0	1	0	0
G	0	0	1	0
C	0	0	0	1

e um tamanho de janela

Para cada par de janelas, calcular o score usando a matriz, e no caso do score máximo (11) ser atingido:



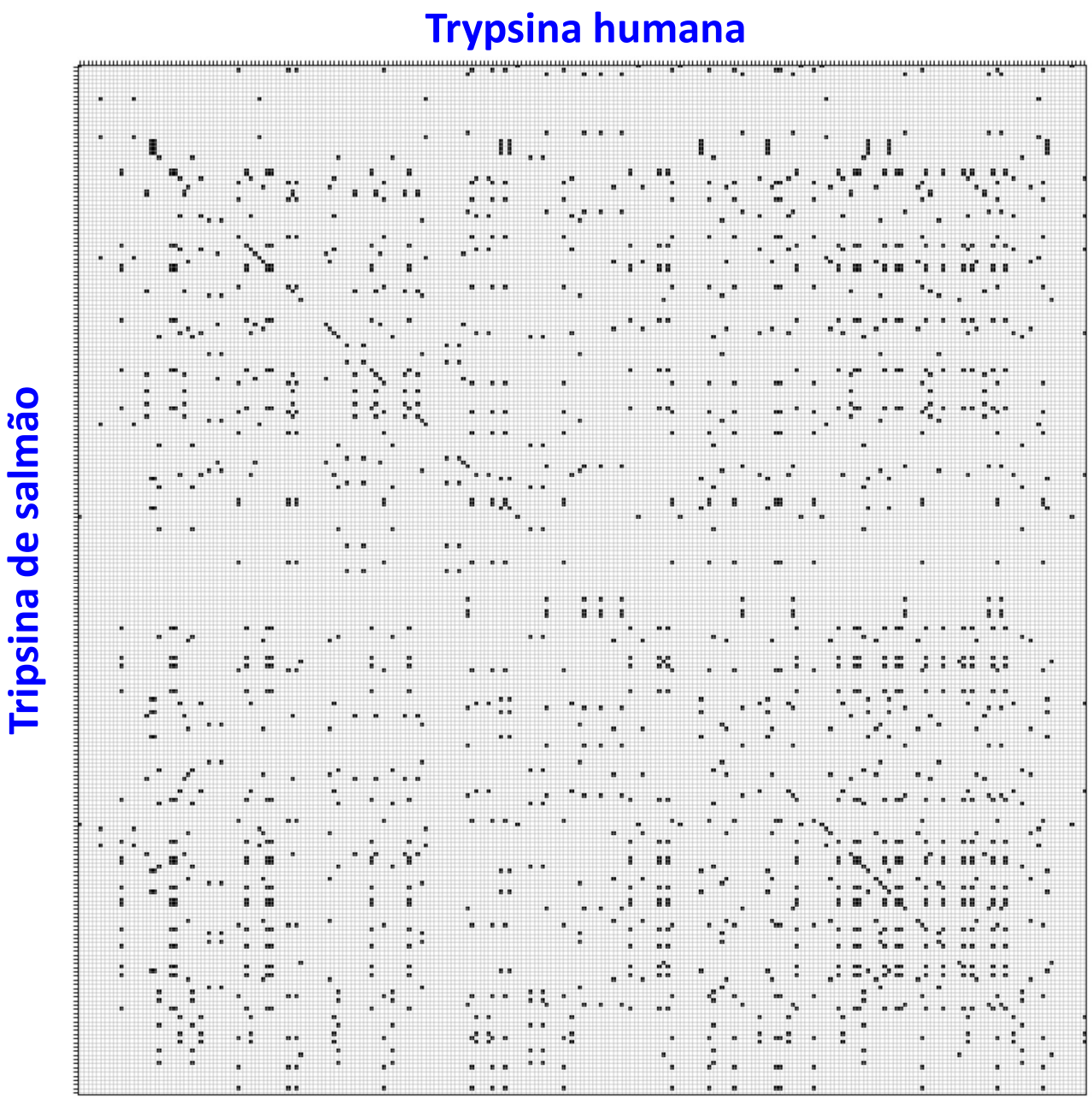
Marcar um ponto



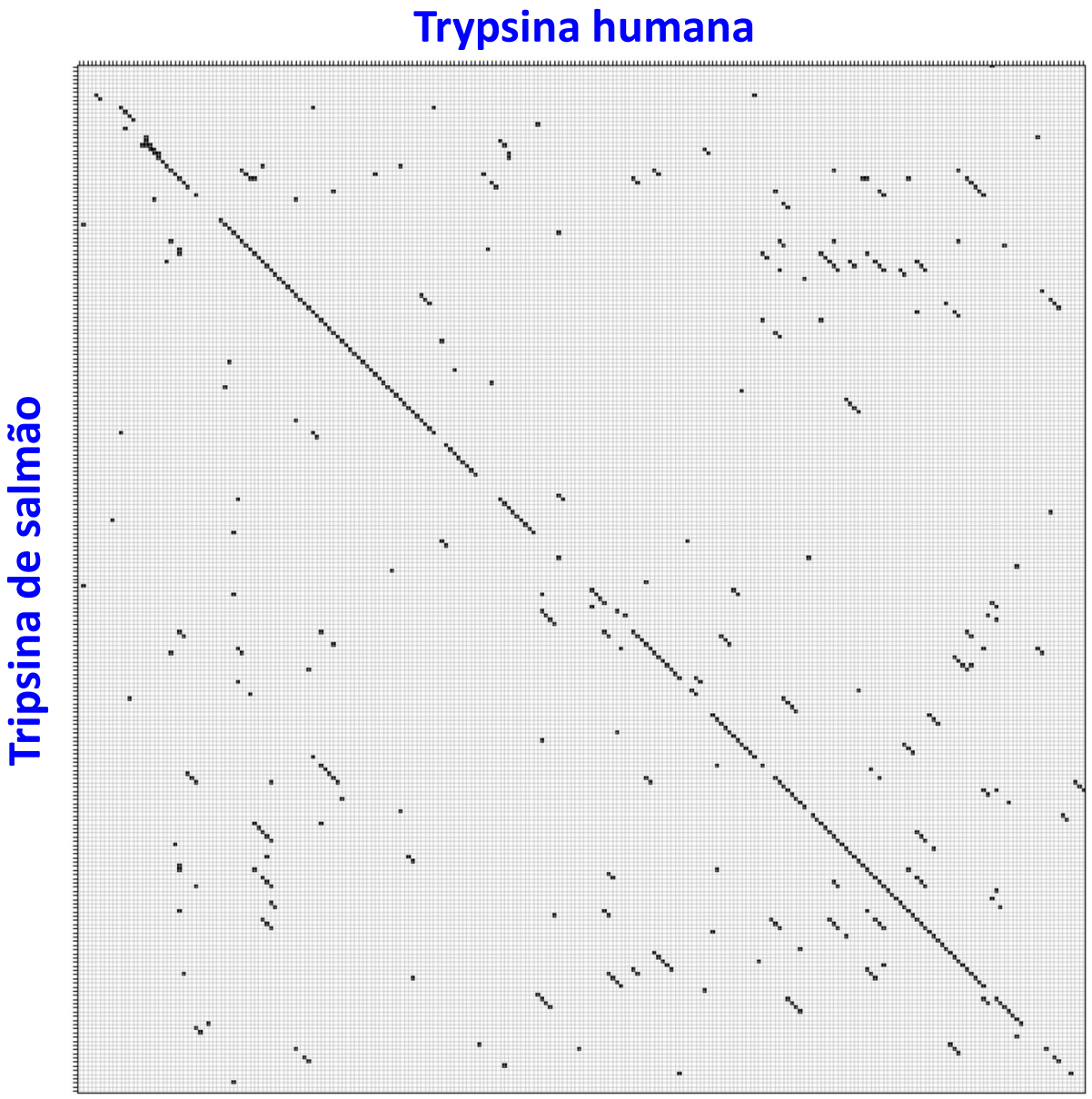
Não marcar um ponto

Neste caso o *score* de *cut-off* usado foi 11, mas podia ser um valor mais baixo

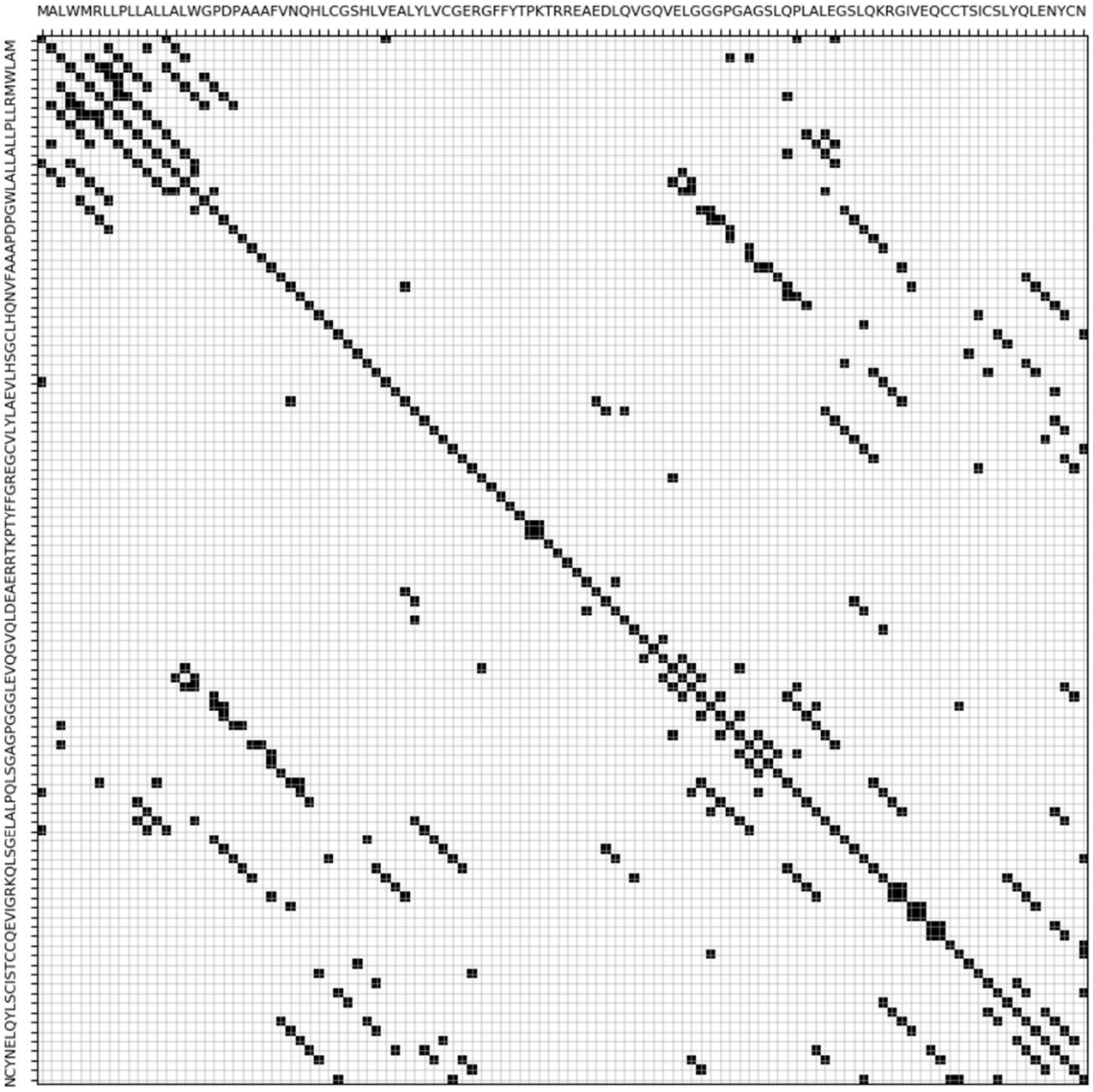
Dot plot com *window size* = 1



Dot plot com *window size* = 15

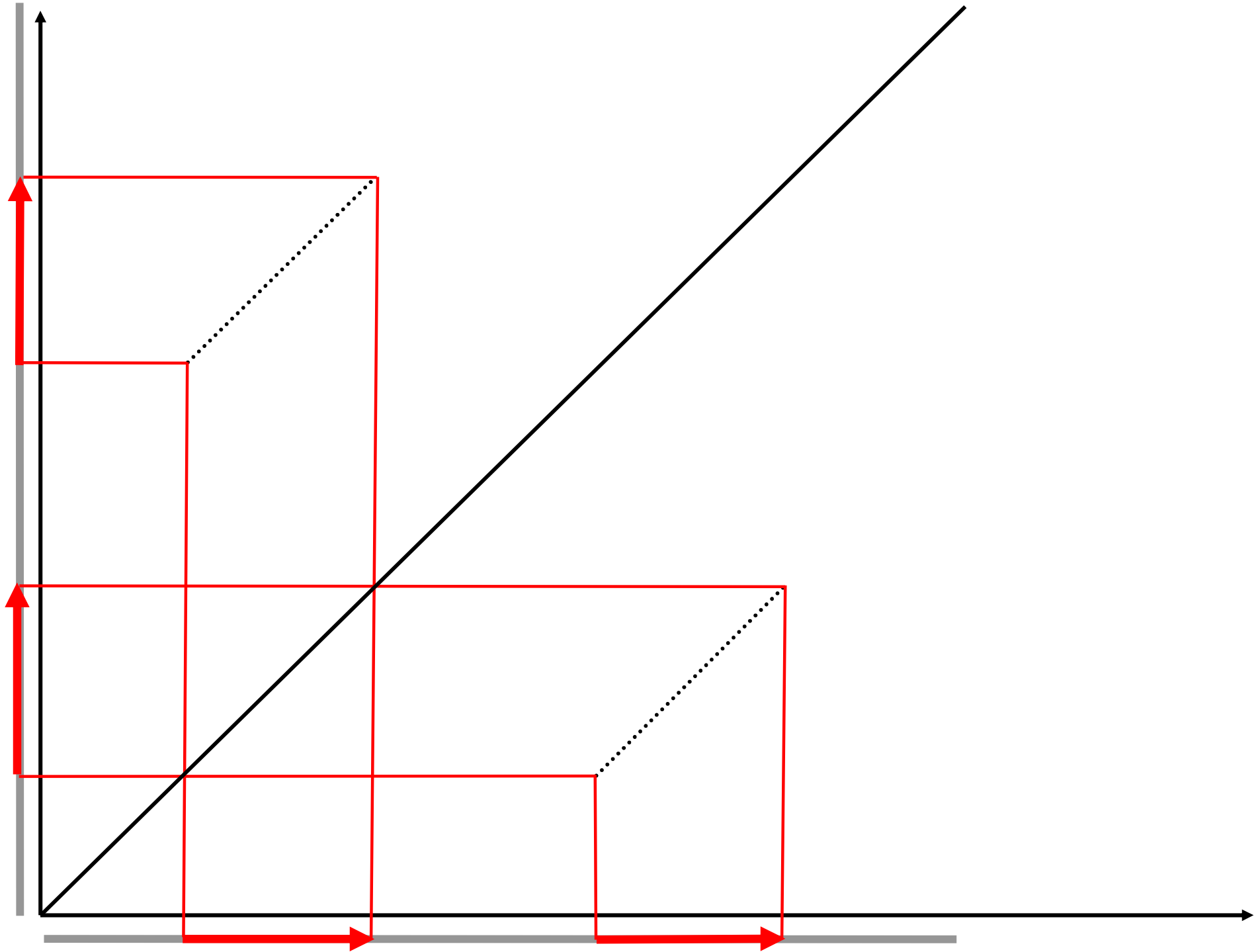


Insulina Humana: autocomparação

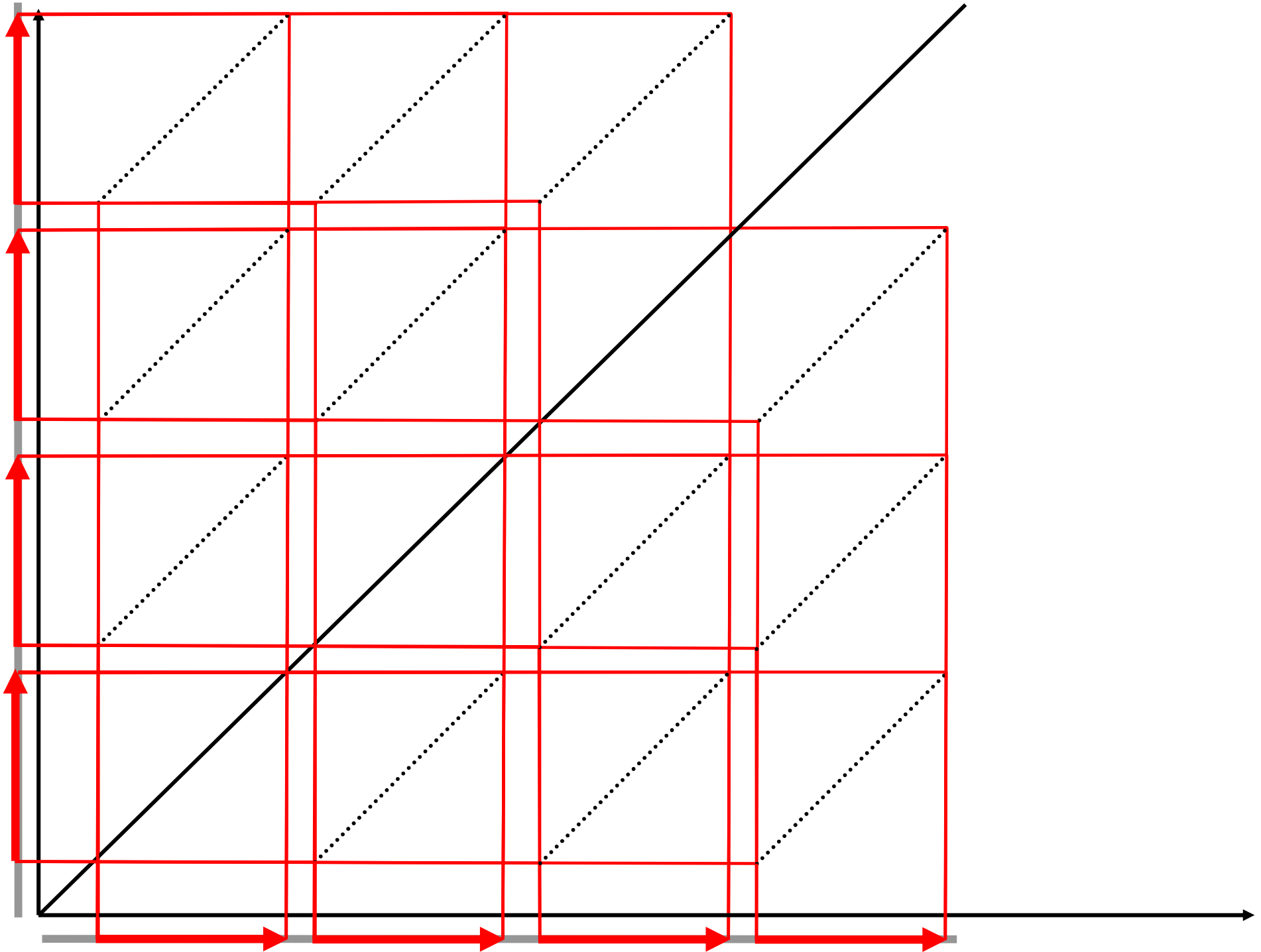


Window size = 15

Autocomparação: detecção de regiões repetidas numa sequência



Autocomparação: detecção de mútiplas regiões repetidas



Alinhamento de sequências

- Consideremos as duas sequências de caracteres:

GAATTCAGTTA

GGATCGA

- Pretendemos alinhar estas sequências de modo a obter um **score máximo** na sua comparação

O que se entende por “alinhar” ?

- Alinhar é estabelecer uma correspondência entre as duas sequências, o que pode ser feito inserindo espaços:

```
GAATTCAGTTA
GG-G--AT--CGA
```

O que se entende por “score” ?

- Um **score** é um número que é associado a cada um dos possíveis alinhamento e que pode ser definido de várias maneiras

Exemplo: associar um valor de 1 a cada posição **idêntica** nas duas sequências, e 0 a posições diferentes

```
GAATTCAGTTA
|  |  |  |  |
GGA-TC-G--A
```

Score=6

Alinhamento

	A	T	G	C
A	1	0	0	0
T	0	1	0	0
G	0	0	1	0
C	0	0	0	1

Matriz de score

Como achar o *score* máximo ?

- Podíamos tentar experimentar **TODOS** alinhamentos possíveis, e escolher aquele que produzisse o score máximo ?...

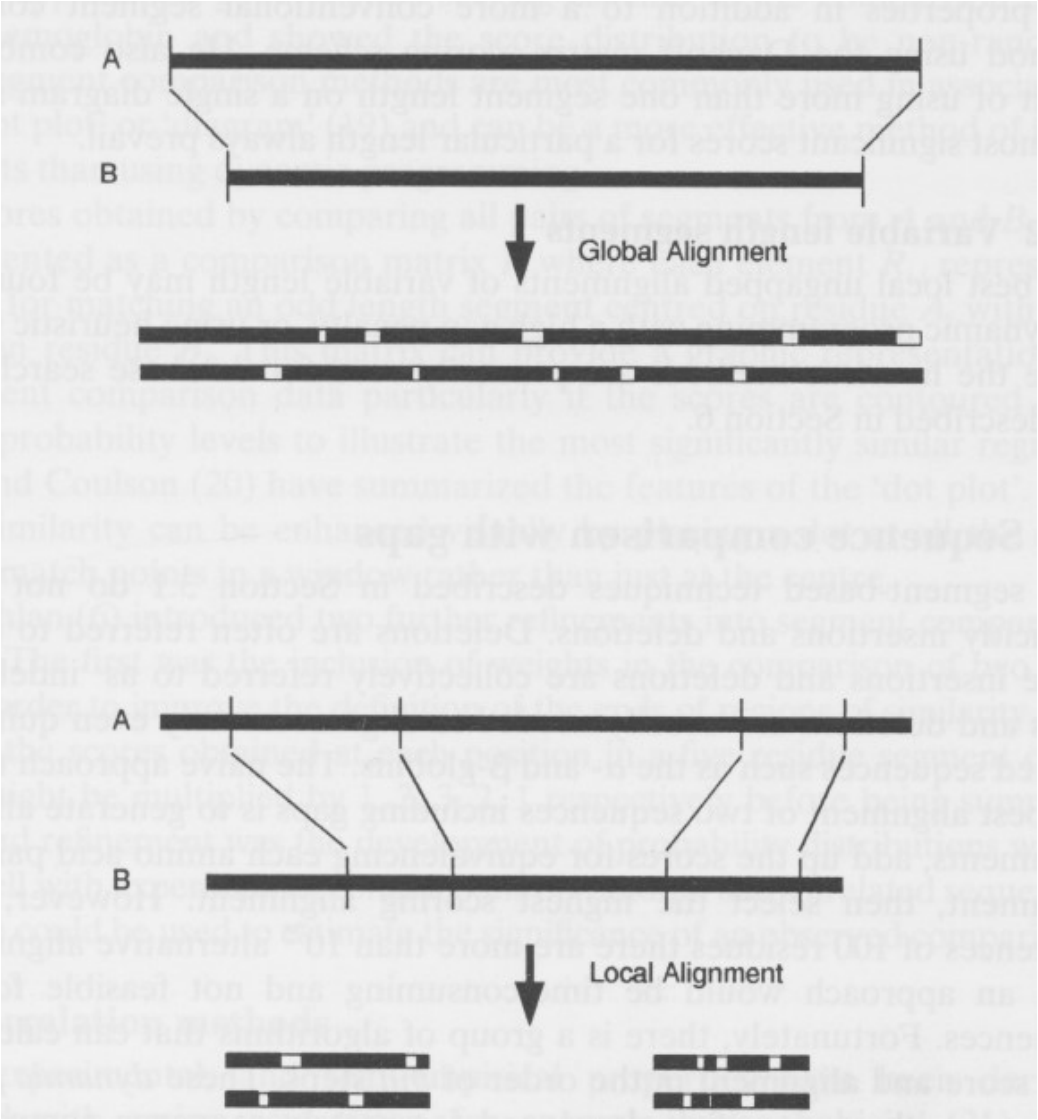
...em geral, a reposta é **não !**

- Para o caso apresentado, existem mais de 3000 alinhamentos possíveis... para duas sequências de 250 caracteres (comprimento médio de uma sequência de proteína) já existem mais de 10^{149} alinhamentos, um número que está muita para lá de toda a potência computacional disponível no planeta Terra!

Mas....

- O **alinhamento óptimo**, que é aquele que **maximiza** o score, pode ser encontrada sem ter que listar de forma exaustiva o conjunto de todos os alinhamentos possíveis. Para este efeito existem vários **algoritmos** computacionais de grande eficiência.

Alinhamento global vs. local



Alinhamento global: as sequências A e B são comparadas na totalidade do seu comprimento, sendo as diferenças de comprimento da sequência compensadas com "gaps" (inserções)

Alinhamento local: consiste na identificação de regiões isoladas de elevada similaridade entre as duas sequências, independentemente do seu contexto.

Algoritmo de Needleman-Wunsch

Needleman, S.B & Wunsch, C.D (1970) *J.Mol.Biol.* **48:443**

- É um algoritmo de programação dinâmica capaz de encontrar o **alinhamento óptimo** global de duas sequências.

- Como ponto de partida necessitamos apenas de uma matriz com o score de alinhamento para cada par de aminoácidos (ou bases nucleotídicas) e uma *gap penalty* (função que atribui um score de "penalização" para a criação de um espaço, ou "gap", na sequência).

- Este algoritmo produz **unicamente** o alinhamento óptimo, não permitindo identificar outros alinhamentos com scores próximos do óptimo e que poderão ser biologicamente relevantes (alinhamentos sub-óptimos).

Algoritmo de Needleman-Wunsch

Exemplo:

Pretende-se alinhar as sequências **GVTAH** e **AVTLI**

•A matriz de *score* usada vai ser a BLOSUM50

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	13	-1	-1	-4	-1	-3	-2	-4	-3	-3	-3	-4	-3	-2	-2	-2	-1	-2	-3	-5
S	-1	5	2	-1	1	0	1	0	-1	0	-1	-1	0	-2	-3	-3	-2	-3	-2	-4
T	-1	2	5	-1	0	-2	0	-1	-1	-1	-2	-1	-1	-1	-1	-1	0	-2	-2	-3
P	-4	-1	-1	10	-1	-2	-2	-1	-1	-1	-2	-3	-1	-3	-3	-4	-3	-4	-3	-4
A	-1	1	0	-1	5	0	-1	-2	-1	-1	-2	-2	-1	-1	-1	-2	0	-3	-2	-3
G	-3	0	-2	-2	0	8	0	-1	-3	-2	-2	-3	-2	-3	-4	-4	-4	-4	-3	-3
N	-2	1	0	-2	-1	0	7	2	0	0	1	-1	0	-2	-3	-4	-3	-4	-2	-4
D	-4	0	-1	-1	-2	-1	2	8	2	0	-1	-2	-1	-4	-4	-4	-4	-5	-3	-5
E	-3	-1	-1	-1	-1	-3	0	2	6	2	0	0	1	-2	-4	-3	-3	-3	-2	-3
Q	-3	0	-1	-1	-1	-2	0	0	2	7	1	1	2	0	-3	-2	-3	-4	-1	-1
H	-3	-1	-2	-2	-2	-2	1	-1	0	1	10	0	0	-1	-4	-3	-4	-1	2	-3
R	-4	-1	-1	-3	-2	-3	-1	-2	0	1	0	7	3	-2	-4	-3	-3	-3	-1	-3
K	-3	0	-1	-1	-1	-2	0	-1	1	2	0	3	6	-2	-3	-3	-3	-4	-2	-3
M	-2	-2	-1	-3	-1	-3	-2	-4	-2	0	-1	-2	-2	7	2	3	1	0	0	-1
I	-2	-3	-1	-3	-1	-4	-3	-4	-4	-3	-4	-4	-3	2	5	2	4	0	-1	-3
L	-2	-3	-1	-4	-2	-4	-4	-4	-3	-2	-3	-3	-3	3	2	5	1	1	-1	-2
V	-1	-2	0	-3	0	-4	-3	-4	-3	-3	-4	-3	-3	1	4	1	5	-1	-1	-3
F	-2	-3	-2	-4	-3	-4	-4	-5	-3	-4	-1	-3	-4	0	0	1	-1	8	4	1
Y	-3	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-1	-2	0	-1	-1	-1	4	8	2
W	-5	-4	-3	-4	-3	-3	-4	-5	-3	-1	-3	-3	-3	-1	-3	-2	-3	1	2	15
B	-3	0	0	-2	-2	-1	4	5	1	0	0	-1	0	-3	-4	-4	-4	-4	-3	-5
Z	-3	0	-1	-1	-1	-2	0	1	5	4	0	0	1	-1	-3	-3	-3	-4	-2	-2
X	-2	-1	0	-2	-1	-2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-2	-1	-3
*	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5

Matriz BLOSUM50

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	13	-1	-1	-4	-1	-3	-2	-4	-3	-3	-3	-4	-3	-2	-2	-2	-1	-2	-3	-5
S	-1	5	2	-1	1	0	1	0	-1	0	-1	-1	0	-2	-3	-3	-2	-3	-2	-4
T	-1	2	5	-1	0	-2	0	-1	-1	-1	-2	-1	-1	-1	-1	-1	0	-2	-2	-3
P	-4	-1	-1	10	-1	-2	-2	-1	-1	-1	-2	-3	-1	-3	-3	-4	-3	-4	-3	-4
A	-1	1	0	-1	5	0	-1	-2	-1	-1	-2	-2	-1	-1	-1	-2	0	-3	-2	-3
G	-3	0	-2	-2	0	8	0	-1	-3	-2	-2	-3	-2	-3	-4	-4	-4	-4	-3	-3
N	-2	1	0	-2	-1	0	7	2	0	0	1	-1	0	-2	-3	-4	-3	-4	-2	-4
D	-4	0	-1	-1	-2	-1	2	8	2	0	-1	-2	-1	-4	-4	-4	-4	-5	-3	-5
E	-3	-1	-1	-1	-1	-3	0	2	6	2	0	0	1	-2	-4	-3	-3	-3	-2	-3
Q	-3	0	-1	-1	-1	-2	0	0	2	7	1	1	2	0	-3	-2	-3	-4	-1	-1
H	-3	-1	-2	-2	-2	-2	1	-1	0	1	10	0	0	-1	-4	-3	-4	-1	2	-3
R	-4	-1	-1	-3	-2	-3	-1	-2	0	1	0	7	3	-2	-4	-3	-3	-3	-1	-3
K	-3	0	-1	-1	-1	-2	0	-1	1	2	0	3	6	-2	-3	-3	-3	-4	-2	-3
M	-2	-2	-1	-3	-1	-3	-2	-4	-2	0	-1	-2	-2	7	2	3	1	0	0	-1
I	-2	-3	-1	-3	-1	-4	-3	-4	-4	-3	-4	-4	-3	2	5	2	4	0	-1	-3
L	-2	-3	-1	-4	-2	-4	-4	-4	-3	-2	-3	-3	-3	3	2	5	1	1	-1	-2
V	-1	-2	0	-3	0	-4	-3	-4	-3	-3	-4	-3	-3	1	4	1	5	-1	-1	-3
F	-2	-3	-2	-4	-3	-4	-4	-5	-3	-4	-1	-3	-4	0	0	1	-1	8	4	1
Y	-3	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-1	-2	0	-1	-1	-1	4	8	2
W	-5	-4	-3	-4	-3	-3	-4	-5	-3	-1	-3	-3	-3	-1	-3	-2	-3	1	2	15
B	-3	0	0	-2	-2	-1	4	5	1	0	0	-1	0	-3	-4	-4	-4	-4	-3	-5
Z	-3	0	-1	-1	-1	-2	0	1	5	4	0	0	1	-1	-3	-3	-3	-4	-2	-2
X	-2	-1	0	-2	-1	-2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-2	-1	-3
*	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5	-5

Scores de *match* / *mismatch* que serão usados

Algoritmo de Needleman-Wunsch

Outro *score* que será usado:

- Inserção de *gap* no alinhamento tem um score **negativo** = -1

1) Construção da matriz de alinhamento

	d	G	V	T	A	H
d						
A						
V						
T						
L						
I						

Matriz de alinhamento

Todos possíveis alinhamentos são caminhos nesta matriz

	d	G	V	T	A	H
d						
A						
V						
T						
L						
I						

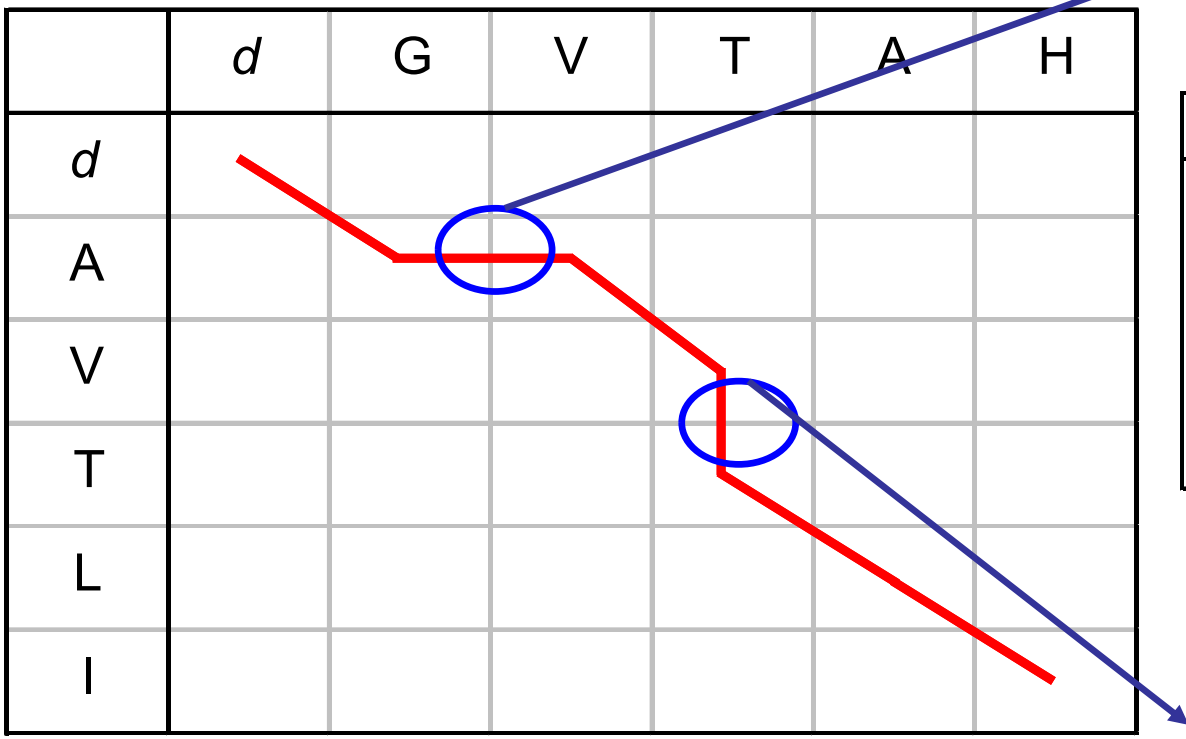
	G	V	T	A	H
A	0	0	0	5	-2
V	-4	5	0	0	-4
T	-2	0	5	0	-2
L	-4	1	-1	-2	-3
I	-4	4	-1	-1	-4

GVATH
AVTLI

Score= 0 + 5 + 5 + (-2) + (-4) = +4

Matriz de alinhamento

Todos possíveis alinhamentos são caminhos nesta matriz



	G	V	T	A	H
A	0	0	0	5	-2
V	-4	5	0	0	-4
T	-2	0	5	0	-2
L	-4	1	-1	-2	-3
I	-4	4	-1	-1	-4

GVT-AH
A-VTLI

Score= 0 + (-1) + 0 + (-1) + (-2) + (-4) = -8

Matriz de alinhamento

Todos possíveis alinhamentos são caminhos nesta matriz

	d	G	V	T	A	H
d						
A						
V						
T						
L						
I						

	G	V	T	A	H
A	0	0	0	5	-2
V	-4	5	0	0	-4
T	-2	0	5	0	-2
L	-4	1	-1	-2	-3
I	-4	4	-1	-1	-4

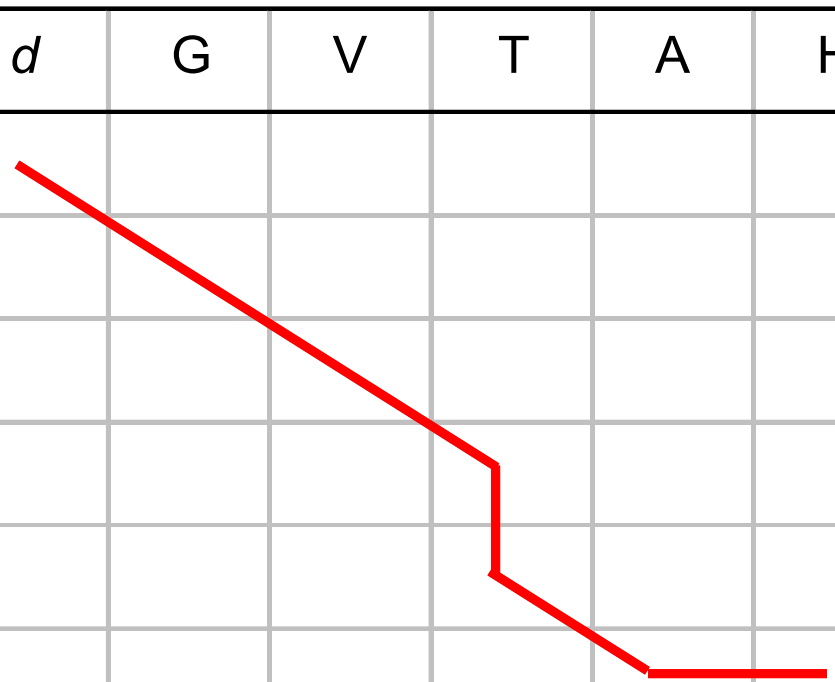
GVTAH-----
-----AVTLI

Score= (-1) + (-1) + (-1) + (-1) + (-1) + (-1) + (-1) + (-1) + (-1) + (-1) = -10

Matriz de alinhamento

Todos possíveis alinhamentos são caminhos nesta matriz

	d	G	V	T	A	H
d						
A						
V						
T						
L						
I						



	G	V	T	A	H
A	0	0	0	5	-2
V	-4	5	0	0	-4
T	-2	0	5	0	-2
L	-4	1	-1	-2	-3
I	-4	4	-1	-1	-4

GVT-AH
AVTLI-

Score= 0 + 5 + 5 + (-1) + (-1) + (-1) = +7

Algoritmo de Needleman-Wunsch

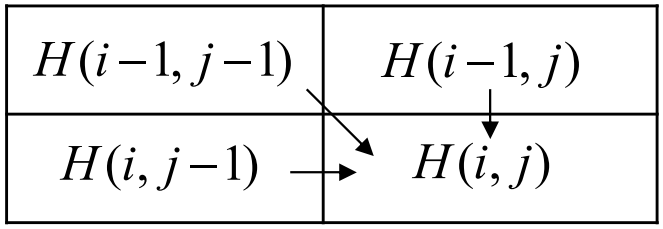
2) Inserção dos valores da *gap penalty*

	d	G	V	T	A	H
d	0	-1	-2	-3	-4	-5
A	-1					
V	-2					
T	-3					
L	-4					
I	-5					

Neste caso a *gap penalty* tem valor -1

Algoritmo de Needleman-Wunsch

3) Preenchimento da tabela, da esquerda para a direita e de cima para baixo, de acordo com seguinte regra:



$$H(i,j) = \max \begin{cases} H(i-1,j-1) + S(i,j) \\ H(i-1,j) + S(-,i) \\ H(i,j-1) + S(i,-) \end{cases}$$

S(i,j) é o score da matriz de score (BLOSUM50 neste caso), e
S(-,j) e S(i, -) scores para inserção de um *gap* horizonal ou vertical

	d	G	V	T	A	H
d	0	-1	-2	-3	-4	-5
A	-1	**0**	-1	-2	2	1
V	-2	-1	**5**	4	3	2
T	-3	-2	4	**10**	9	8
L	-4	-3	3	**9**	8	7
I	-5	-4	2	8	**8**	**7**

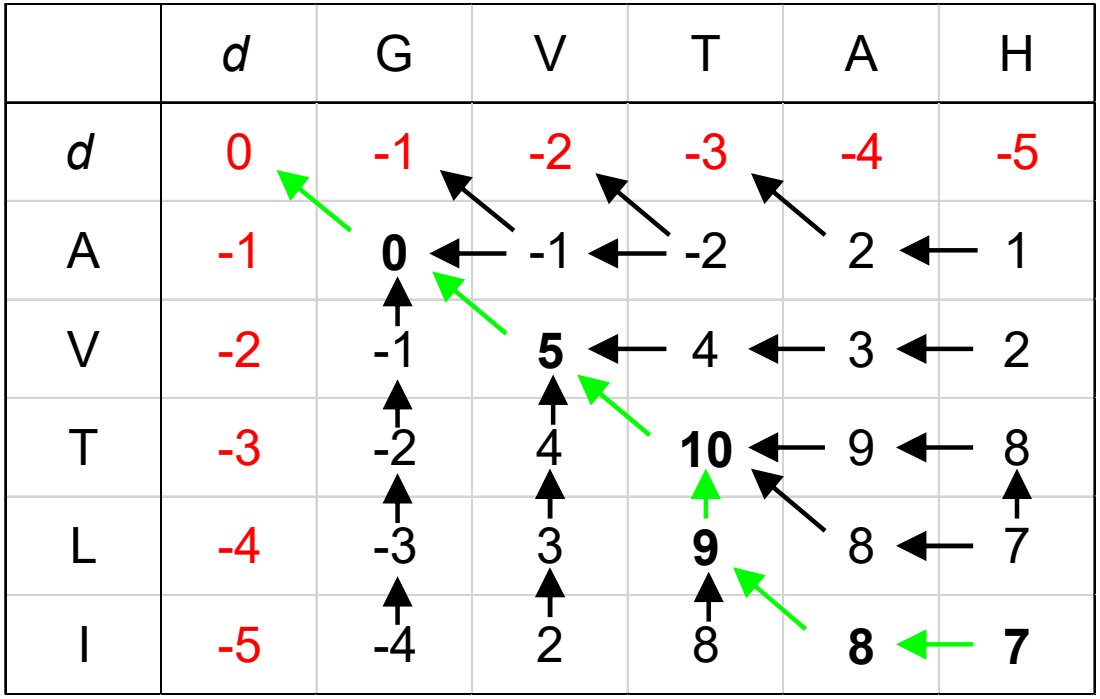
	G	V	T	A	H
A	0	0	0	5	-2
V	-4	5	0	0	-4
T	-2	0	5	0	-2
L	-4	1	-1	-2	-3
I	-4	4	-1	-1	-4

Scores da matriz BLOSUM50

Cada célula mantém a informação da proveniência do valor anterior (setas)

Algoritmo de Needleman-Wunsch

4) Traçar o caminho desde o canto inferior direito, seguindo as setas. Cada movimento horizontal ou vertical corresponde a uma gap na sequência respectiva.



G	V	T	—	A	H
A	V	T	L	I	—

Score: 7

Alinhamento óptimo

Gap penalty = -1

Match / Mismatch →

	G	V	T	A	H
A	0	0	0	5	-2
V	-4	5	0	0	-4
T	-2	0	5	0	-2
L	-4	1	-1	-2	-3
I	-4	4	-1	-1	-4

```
G V T - A H
A V T L I -
```

	gap	G	V	T	A	H
gap	0	-1	-2	-3	-4	-5
A	-1	0	-1	-2	+2	+1
V	-2	-1	+5	+4	+3	+2
T	-3	-2	+4	+10	+9	+8
L	-4	-3	+3	+9	+8	+7
I	-5	-4	+2	+8	+8	+7

Alinhamento local: algoritmo de Smith-Waterman

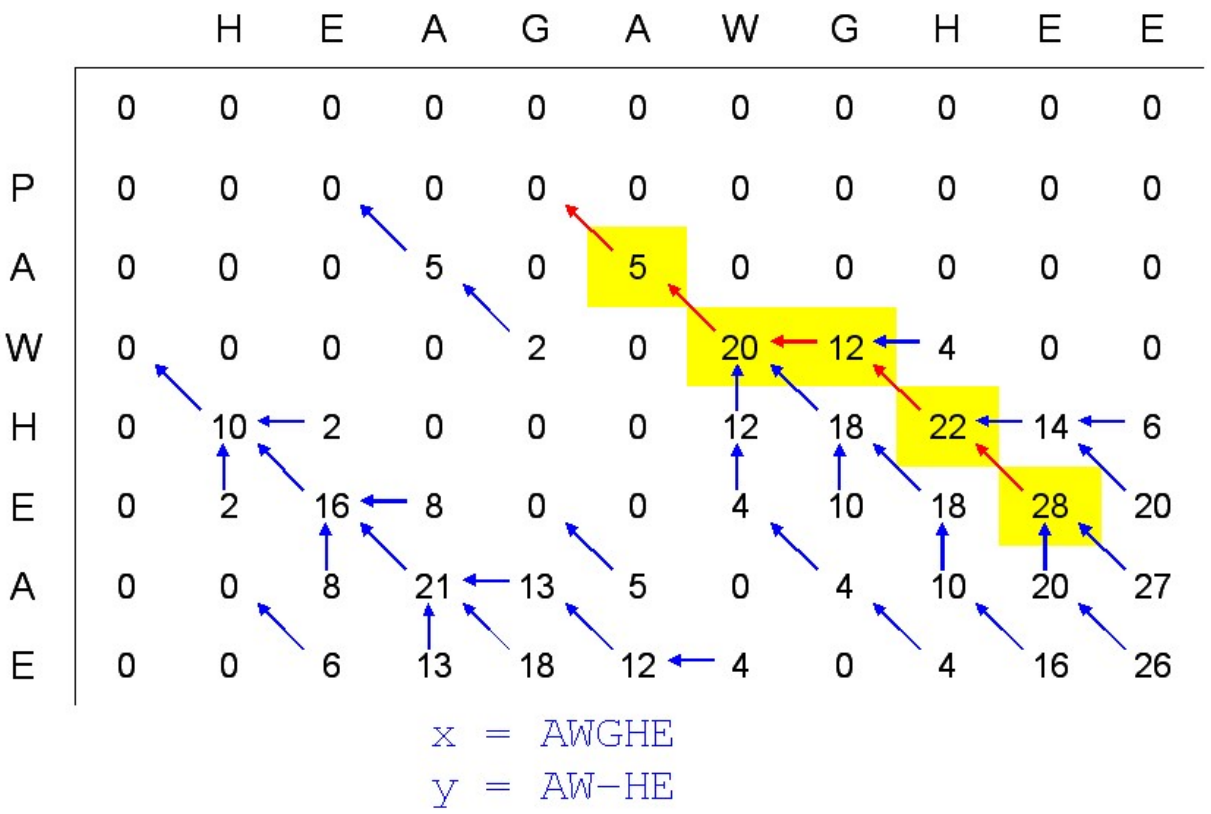
Smith, T.F. & Waterman, M.S (1981) *J.Mol.Biol.* **147:**195-197

O algoritmo de Smith-Waterman é uma versão modificada de N-W que permite encontrar o alinhamento local óptimo entre duas sequências.

$$H(i,j) = \max \begin{cases} 0 \\ H(i-1, j-1) + S(i,j) \\ H(i-1, j) + S(-,i) \\ H(i, j-1) + S(i,-) \end{cases}$$

Se o valor calculado partir das células anteriores for <0, é substituído pelo valor zero e o alinhamento termina nesse ponto. O alinhamento local inicia-se na célula de valor mais alto da matriz de alinhamento.

Alinhamento local: algoritmo de Smith-Waterman



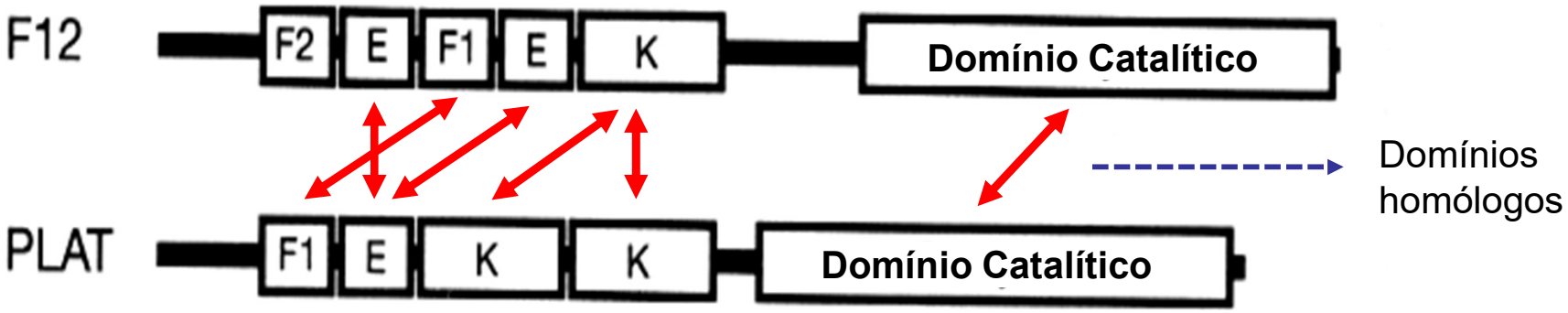
Para que este algoritmo funcione, é necessário que o *score* esperado para um alinhamento aleatório seja negativo, e que existam valores positivos na matriz de comparação

Importância do alinhamento local

Muitas proteínas apresentam uma estrutura **modular**, tendo regiões com proveniências evolutivas distintas e relacionadas com diferentes famílias.

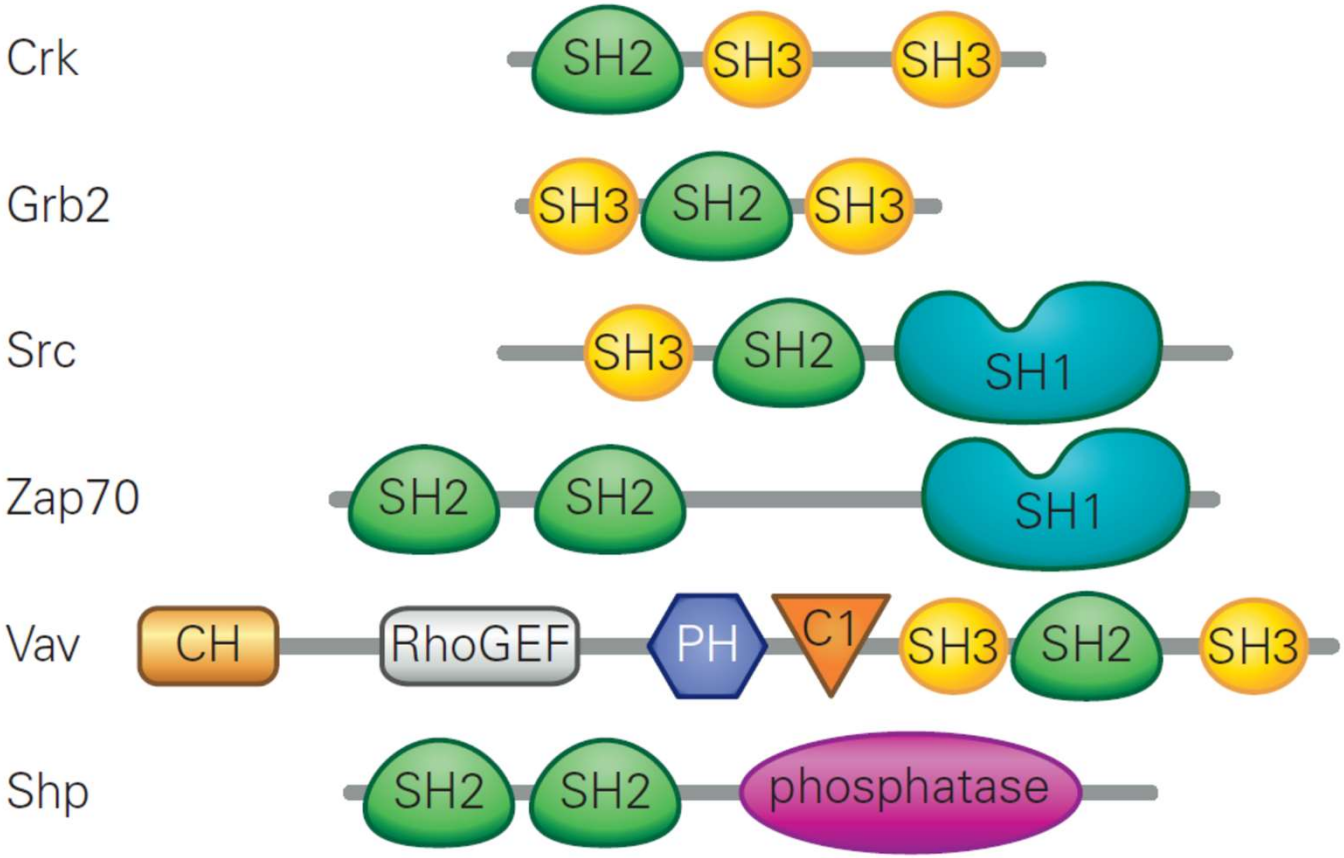
A comparação local de duas sequências permite mais facilmente reconhecer estas regiões, mesmo quando na sua globalidade as sequências são largamente discrepantes.

Exemplo:



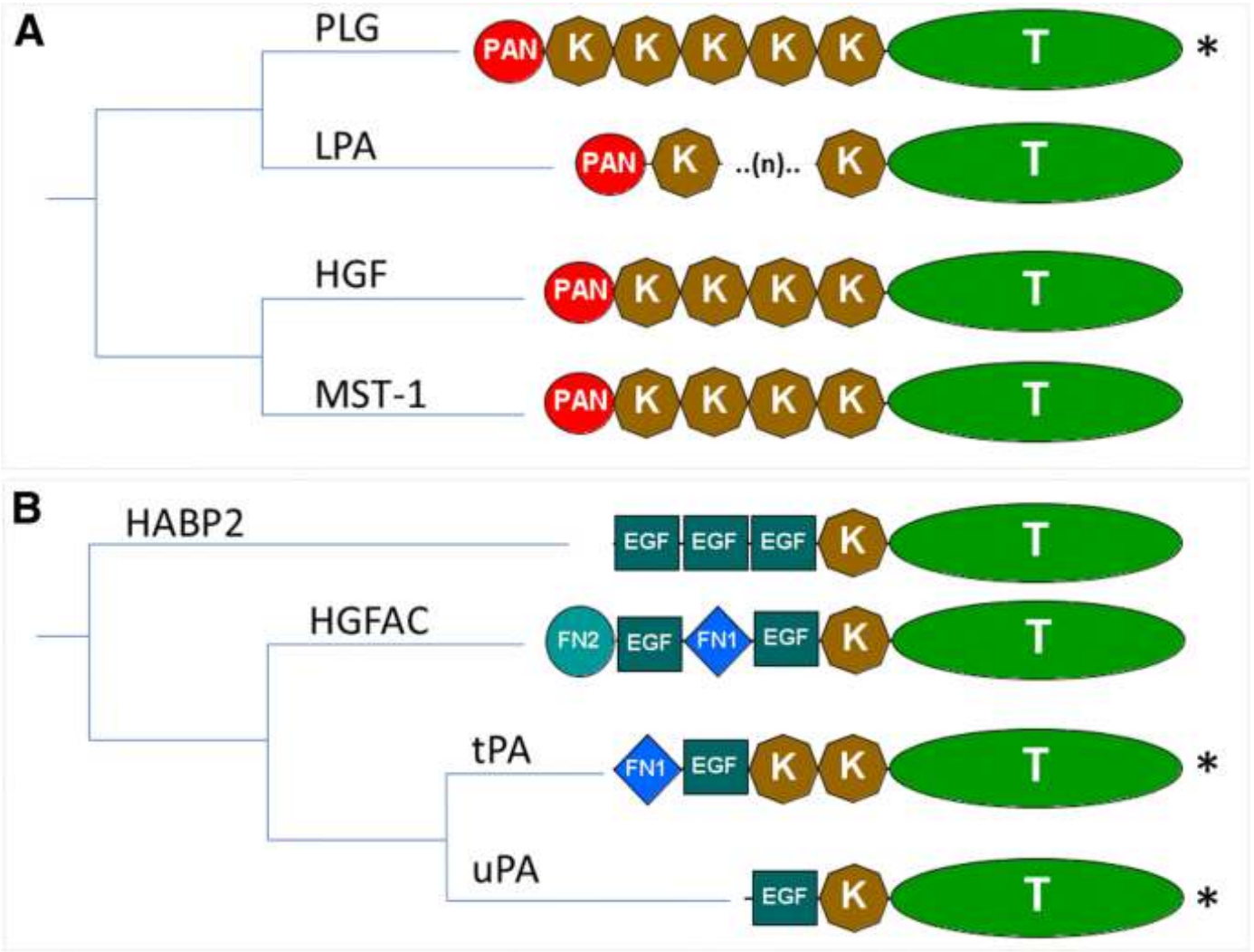
PLAT – plasminogen activator
F12 – coagulation factor XII

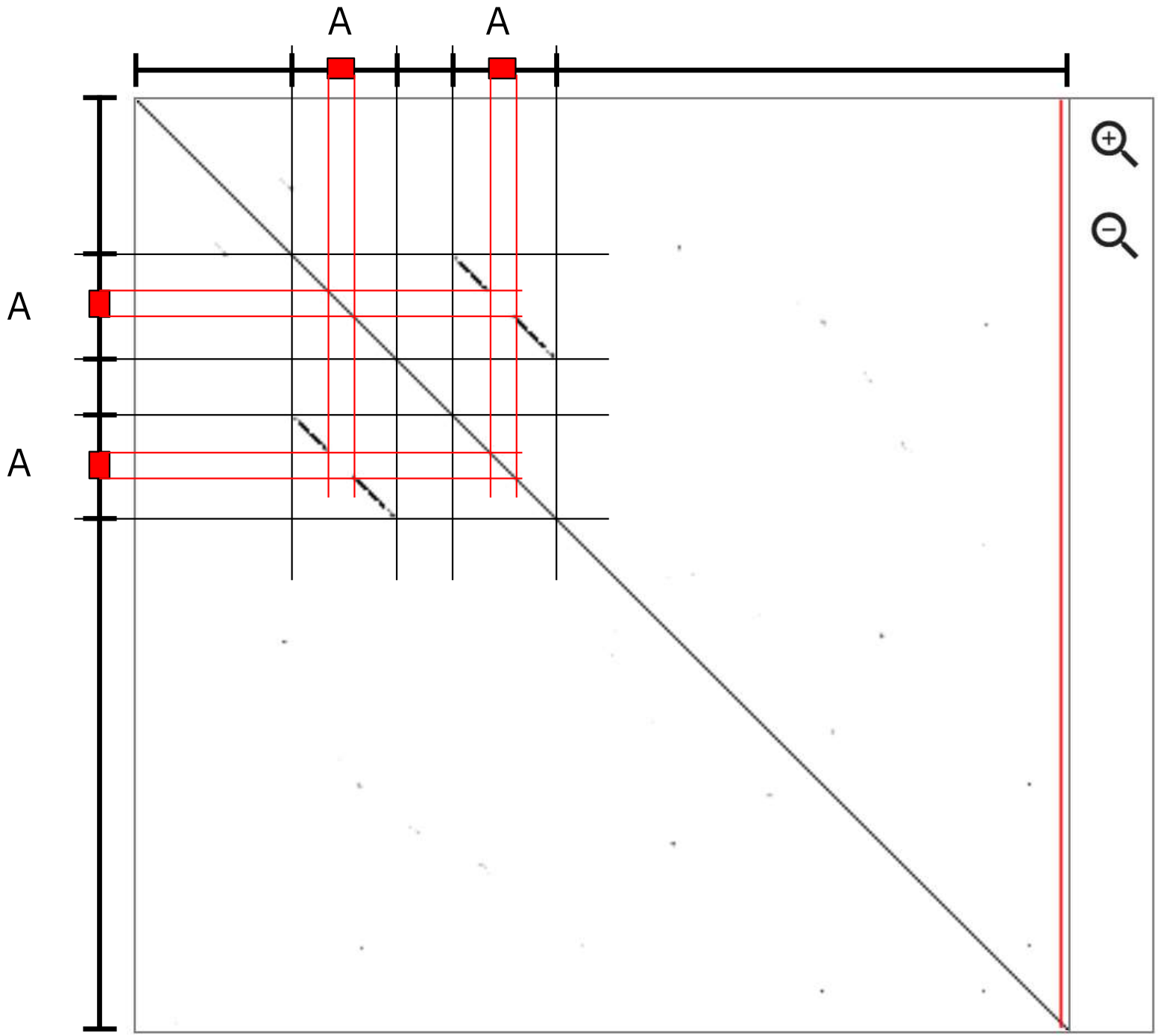
Domínios SH (**Src** Homology Domains)



Múltiplas proteínas com funções muito variadas são construídas a partir de diferentes combinações dos domínios SH1, SH2 e SH3. Os domínios Src tem activade de **tirosina cinase**.

Família do plasminogéneo




A
A
A
A