

ALINEAMIENTO DE **SECUENCIAS**

Alineamiento de Secuencias

AGGCTATCACCTGACCTCCAGGCCGATGCCC
TAGCTATCACGACCGCGGGTCGATTTGCCCGAC

-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
TAG-CTATCAC--GACCGC--GGTCGATTTGCCCGAC

Definición:

Dadas dos cadenas: $x = x_1x_2\dots x_M$, $y = y_1y_2\dots y_N$

Un Alineamiento es una asignación de gaps a posiciones $0\dots N$ en x y $0\dots N$ en y de modo que se alienea cada letra en una secuencia con una letra o un espacio en la otra secuencia.

Alineamiento de Secuencias

Cual es un buen alienamiento?

AGGCTAGTT ,
AGCGAAGTTT

AGGCTAGTT-
AGCGAAGTTT

6 matches, 3 mismatches, 1 gap

AGGCTA-GTT-
AG-CGAAGTTT

7 matches, 1 mismatch, 3 gaps

AGGC-TA-GTT-
AG-CG-AAGTTT

7 matches, 0 mismatches, 5 gaps

Alineamiento de Secuencias

Las funciones de Scoring

- Sequence edits:

- Mutations
- Insertions
- Deletions

AGGCCTC

AGGACTC

AGGGCCTC

AGG . CTC

Scoring Function:

Match:	+m
Mismatch:	-s
Gap:	-d

Definicion

alternativa: La distancia mínima de edición:

“Dadas 2 cadenas x , y encontrar el minimo número de ediciones – inserciones , delecciones, o mutaciones – para transformar una cadena en la otra”

$$\text{Score } F = (\# \text{ matches}) \times m - (\# \text{ mismatches}) \times s - (\# \text{ gaps}) \times d$$

Alineamiento de Secuencias

Las funciones de Scoring – El Score es aditivo:

El score de alinear

$$X = x_1 x_2 \dots x_M$$

Es aditivo:

$$y = y_1 y_2 \dots y_N$$

En el alineamiento

Siguiente:

$$x_1 \dots x_i \quad x_{i+1} \dots x_M$$

$$y_1 \dots y_j \quad y_{j+1} \dots y_N$$

Los dos scores se
suman:

$$F(x[1:M], y[1:N]) = F(x[1:i], y[1:j]) + F(x[i+1:M], y[j+1:N])$$

Programación dinámica

En un problema de alineación:

Existe un número polinomial
De sub problemas de alineacion
Tipo:

Alinear Con:
 $x_1 \dots x_i$ $y_1 \dots y_j$

El problema de alineamiento
original es uno de los sub
problemas :

Alinear Con:
 $x_1 \dots x_M$ $y_1 \dots y_N$

Cada sub problema de puede resolver a partir de
sub problemas mas pequeños

Programación dinámica

En un problema de alineación:

Alinear

Con:

Sea $F(i,j)$ el score optimo de alinear:

$x_1 \dots x_i$

$y_1 \dots y_j$

Existen tres posibles casos:

1. x_i Alinea con y_j

$x_1 \dots x_{i-1} x_i$

$y_1 \dots y_{j-1} y_j$

$$F(i,j) = F(i-1,j-1) + \begin{cases} m & \text{si } x_i = y_j \\ -s & \text{si } x_i \neq y_j \end{cases}$$

Programación dinámica

En un problema de alineación:

Sea $F(i,j)$ el score optimo de alinear:

Alinear Con:
 $x_1 \dots x_i$ $y_1 \dots y_j$

Existen tres posibles casos:

2. x_i Alinea con —

$x_1 \dots x_{i-1}$ x_i

$y_1 \dots y_j$ —

$$F(i,j) = F(i-1,j) - d$$

3. y_j Alinea con —

$x_1 \dots x_i$ —

$y_1 \dots y_{j-1}$ y_j

$$F(i,j) = F(i,j-1) - d$$

Programación dinámica

En un problema de alineación:

Alinear Con:

Sea $F(i,j)$ el score optimo de alinear: $x_1 \dots x_i$ $y_1 \dots y_j$

Existen tres posibles casos.Cuál es el caso optimo?:

Si asumimos que $F(i,j-1)$, $F(i-1,j)$, $F(i-1,j-1)$ son optimos

Entonces:

$$F(i,j) = \max \begin{cases} F(i-1,j-1) + s(x_i, y_j) \\ F(i-1,j) - d \\ F(i,j-1) - d \end{cases}$$

Donde: $s(x_i, y_j) = m$ si $x_i = y_j$; y $s(x_i, y_j) = -s$ si $x_i \neq y_j$

Nota: Match = +m; Mismatch = -s; Gap = -d

Algoritmos dinámicos.

- **Alineamiento global (Needleman y Wunsch, 1970).**
 - Encontrar un alineamiento de dos secuencias completas (se introducen huecos para igualar sus longitudes) de tal manera que proporcione un valor numérico global máximo. Es Algoritmo relativamente lento que usa mucha memoria.
 - Ofrece buenos resultados para secuencias con alta similitud, tanto en la longitud como en el contenido de éstas.
- **Alineamiento local (Smith y Waterman, 1981).**
 - Encontrar fragmentos de dos secuencias que proporcionen alineamientos con puntuación máxima (se alinean las partes más parecidas).
 - Suele ser una combinación de muchos alineamientos globales de secuencias cortas.
 - Ofrece buenos resultados para secuencias con baja similitud que contengan regiones parecidas

BLAST (Basic Local Alignment Search Tool): principal programa de alineamiento local de secuencias (es “una evolución” del algoritmo de Smith-Waterman de alineamiento local).

Programación dinámica

Alineamiento Global

Needleman & Wunsch (1970)

Encuentra el mejor scoring para un alineamiento global de dos secuencias

- Inicialización de la matriz
- Llenado de Matriz con los score individuales
- Construcción del alinamiento que maximice el score total
- Recuperación de la solución (Backtracking)

Dadas las secuencias:

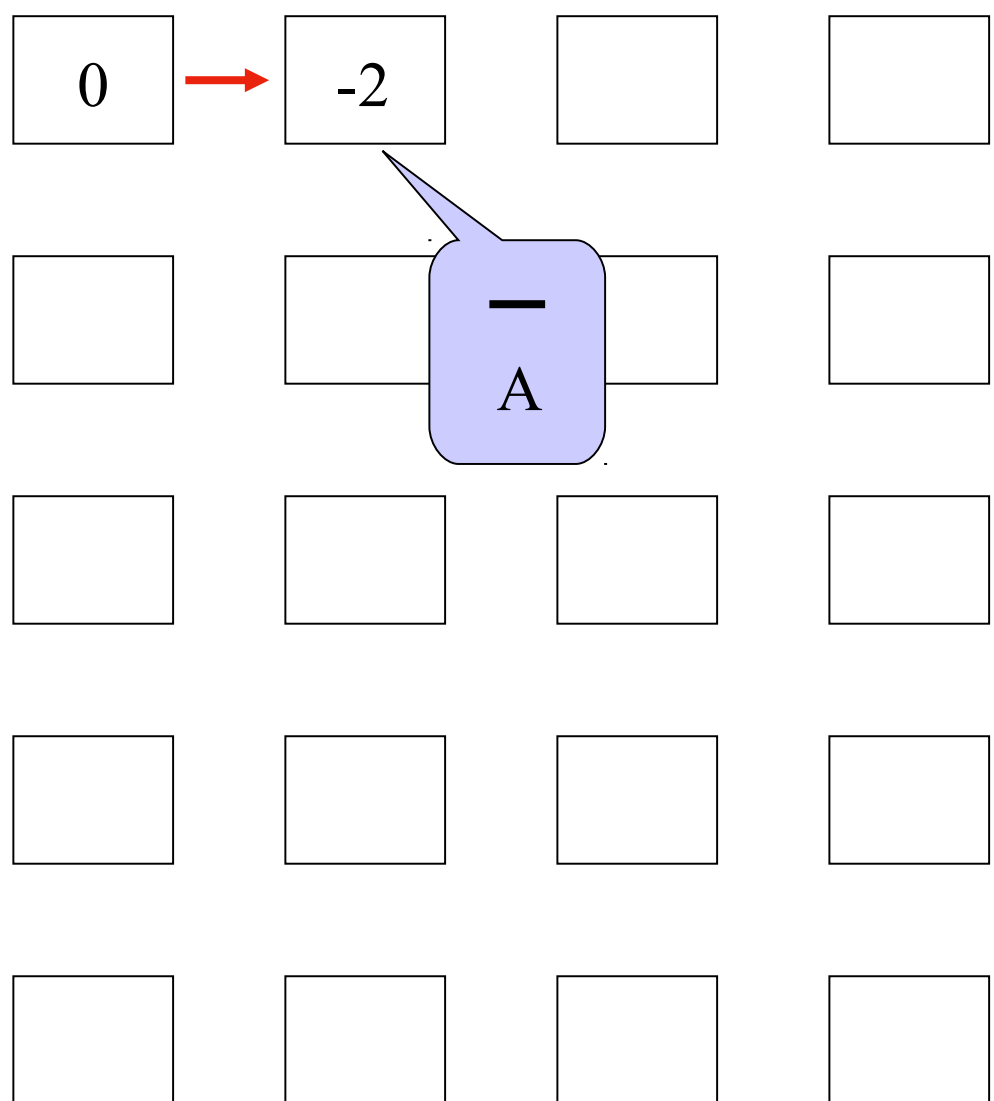
S = AAAC

T = AGC

		A	G	C
		<input type="text"/>	<input type="text"/>	<input type="text"/>
A		<input type="text"/>	<input type="text"/>	<input type="text"/>
A		<input type="text"/>	<input type="text"/>	<input type="text"/>
A		<input type="text"/>	<input type="text"/>	<input type="text"/>
C		<input type="text"/>	<input type="text"/>	<input type="text"/>

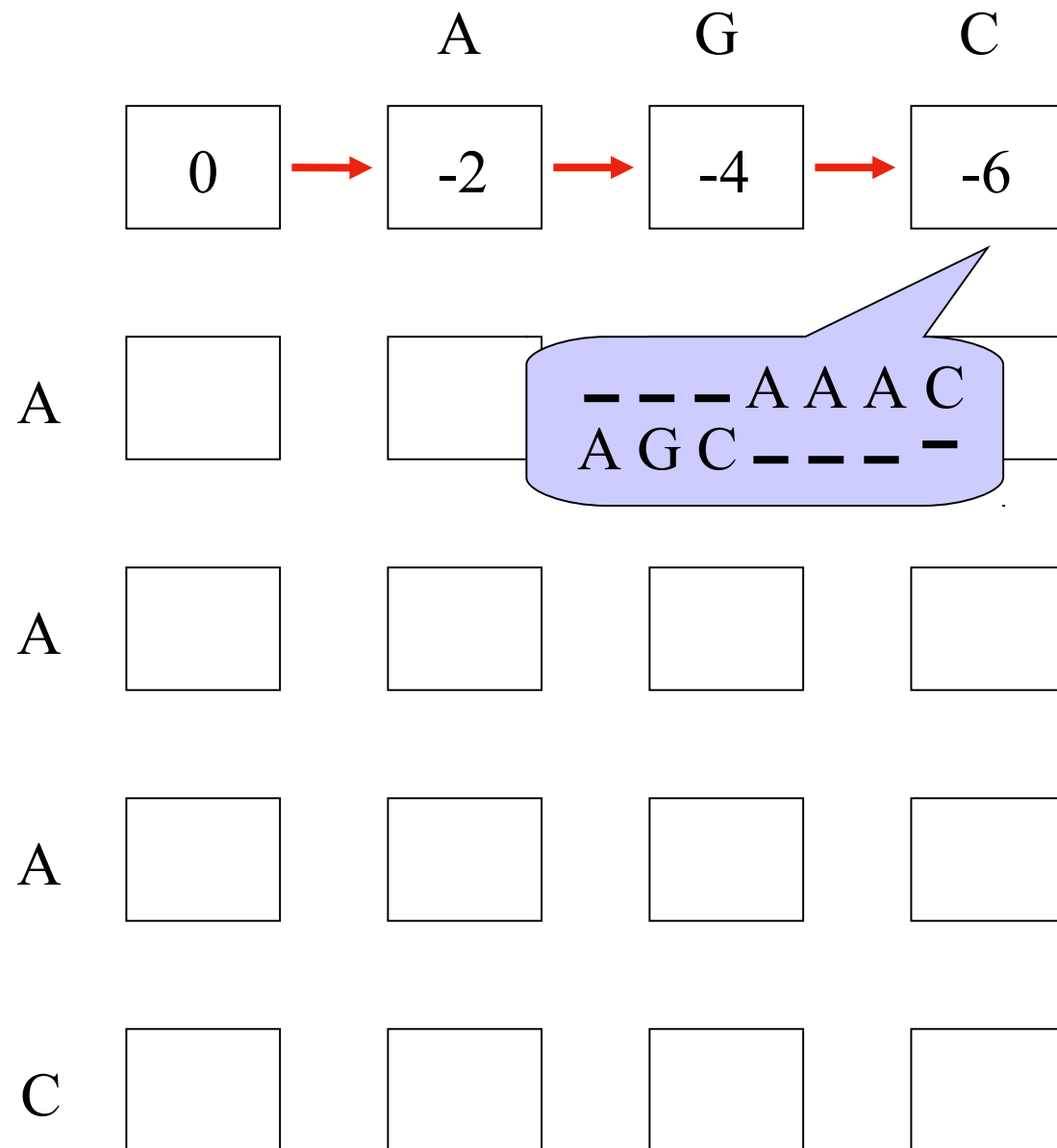
Inicio de la matriz:

		A	G	C
		0	-2	
A				
A				
A				
C				

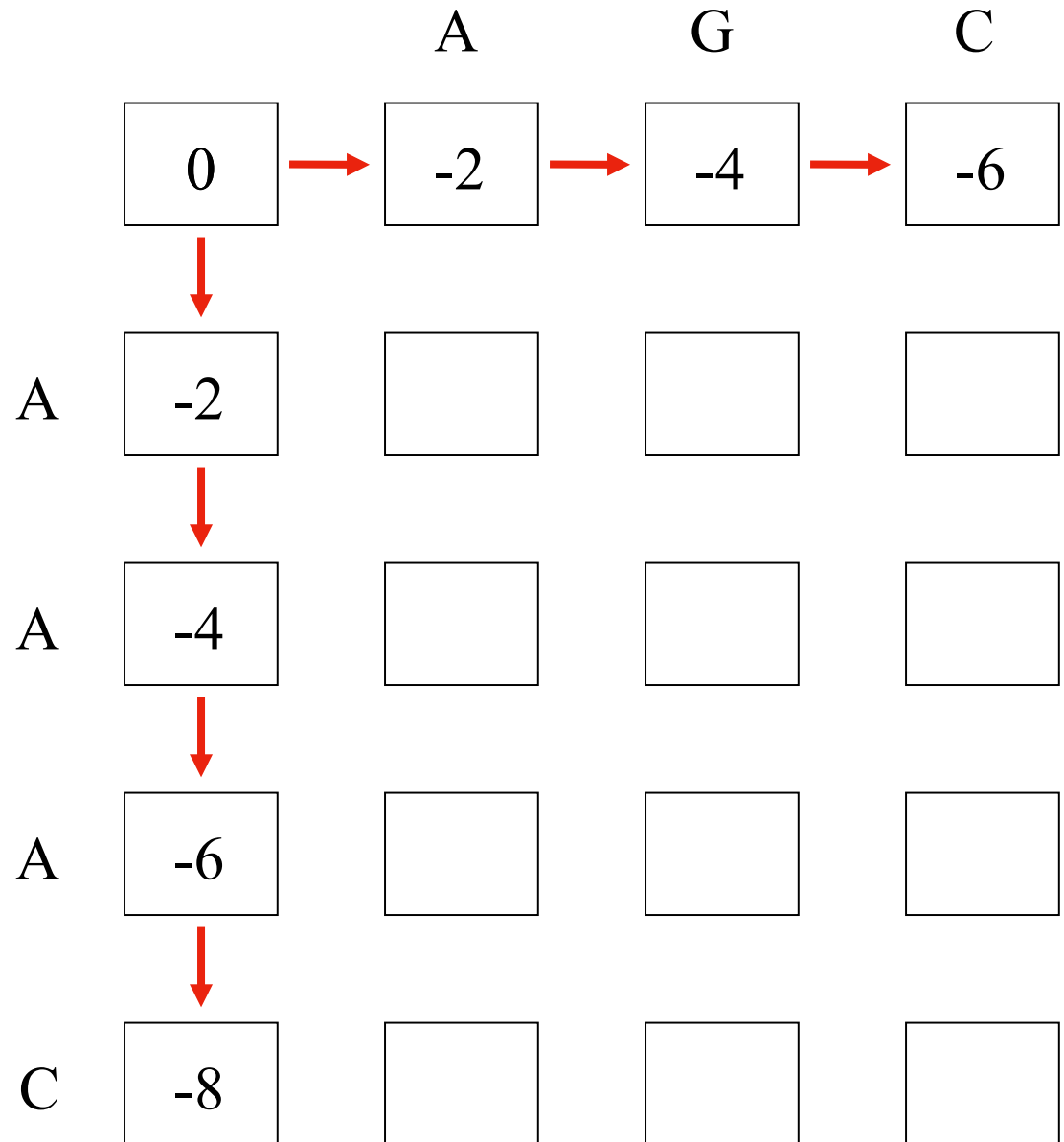


The diagram illustrates the initial state of a matrix. A red arrow points from the cell containing '0' to the cell containing '-2'. A callout box, which is light blue with a black border and a pointer, points to the '-2' cell and contains the text '- A'.

Inicio de la matriz:



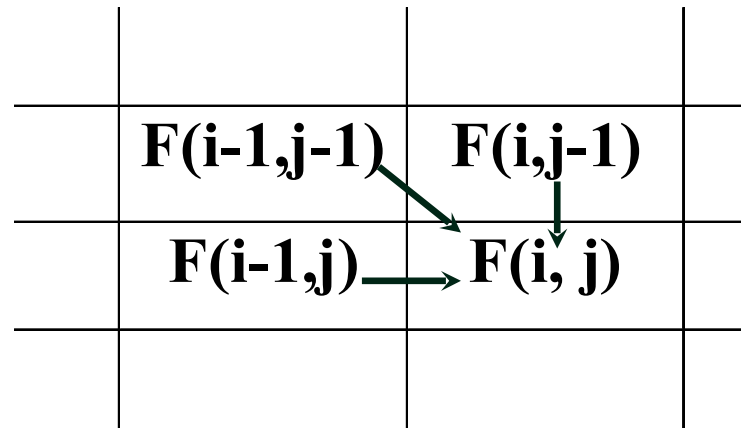
Llenado de la matriz con los score individuales



Llenado de la matriz con los score individuales

		A	G	C	
		0	-2	-4	-6
A	-2	1	-1	-1	
A	-4	1	-1	-1	
A	-6	1	-1	-1	
C	-8	-1	-1	1	

Construcción del alineamiento



$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(i, j) \\ F(i-1, j) - g \\ F(i, j-1) - g \end{cases}$$

$S(i, j)$ Función de similitud g = Penalización del Gap

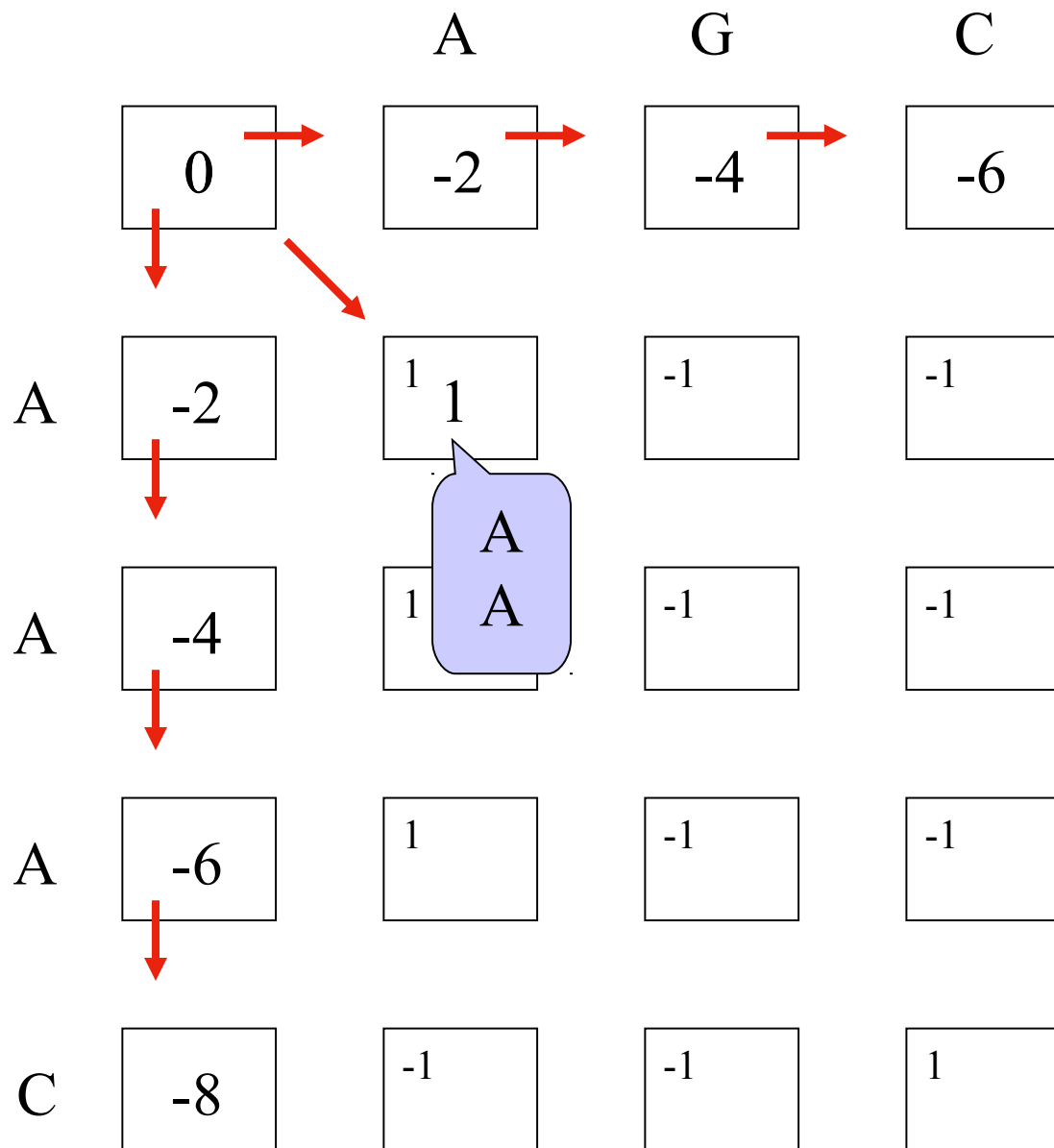
Construcción del alineamiento

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(i, j) \\ F(i-1, j) - g \\ F(i, j-1) - g \end{cases}$$

		A	G	C	
		<div>0</div> <div><div></div><div></div><div></div><div></div></div>	<div>-2</div> <div><div></div><div></div><div></div><div></div></div>	<div>-4</div> <div><div></div><div></div><div></div><div></div></div>	<div>-6</div> <div><div></div><div></div><div></div><div></div></div>
A	<div>-2</div> <div><div></div><div></div><div></div><div></div></div>	<div>1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	
A	<div>-4</div> <div><div></div><div></div><div></div><div></div></div>	<div>1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	
A	<div>-6</div> <div><div></div><div></div><div></div><div></div></div>	<div>1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	
C	<div>-8</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	<div>-1</div> <div><div></div><div></div><div></div><div></div></div>	<div>1</div> <div><div></div><div></div><div></div><div></div></div>	

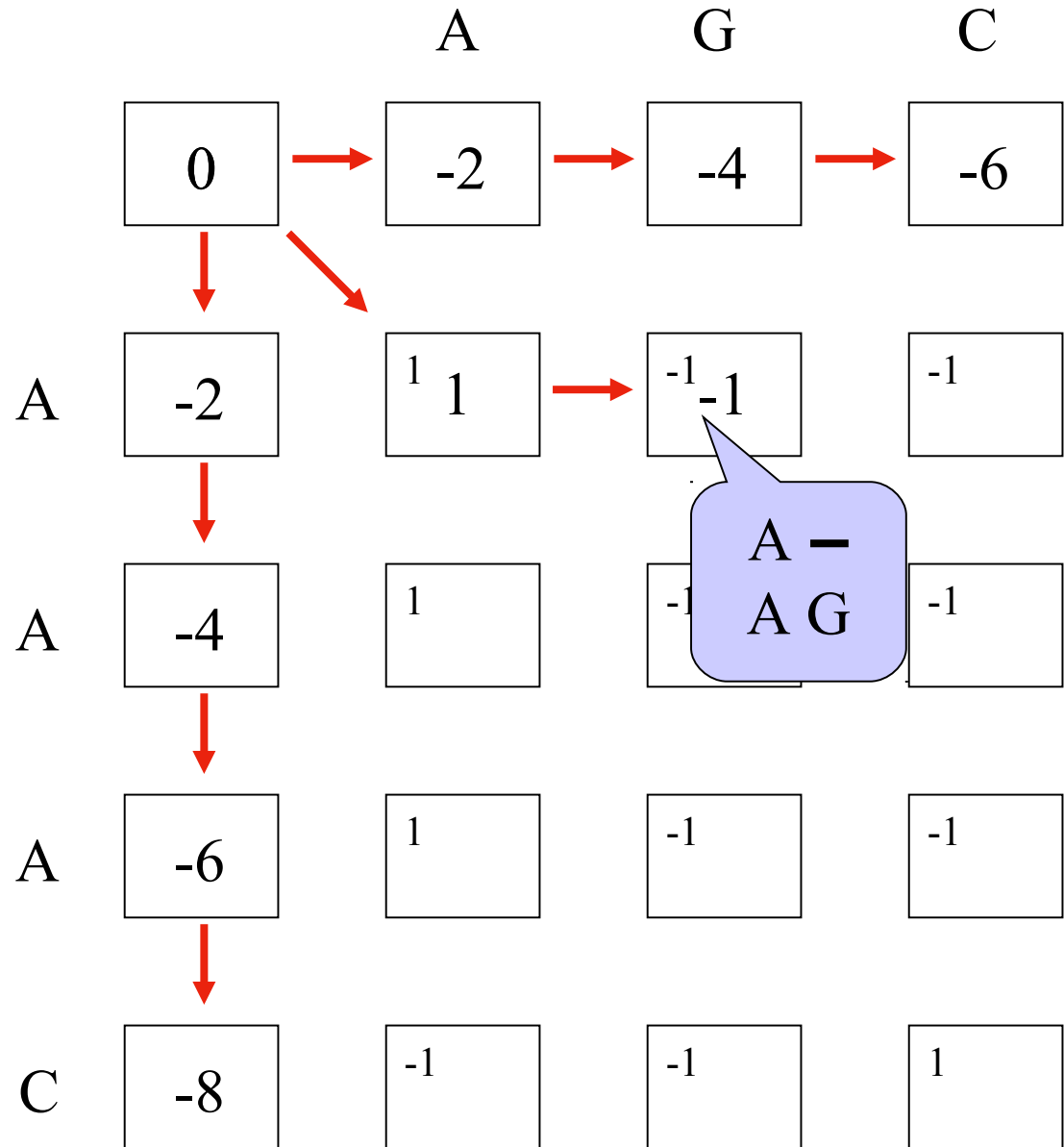
Construcción del alineamiento

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(i, j) \\ F(i-1, j) - g \\ F(i, j-1) - g \end{cases}$$



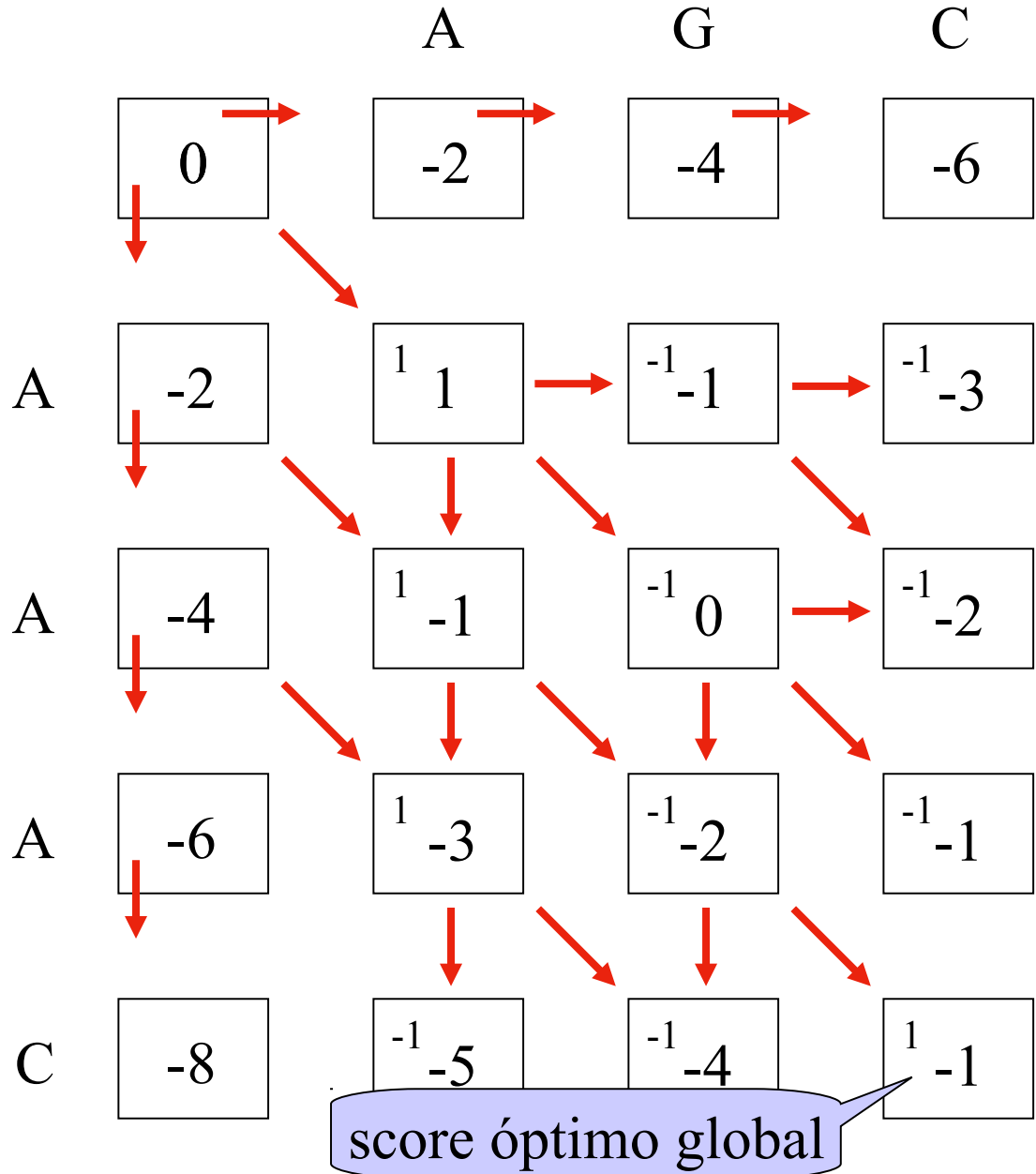
Construcción del alineamiento

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(i, j) \\ F(i-1, j) - g \\ F(i, j-1) - g \end{cases}$$

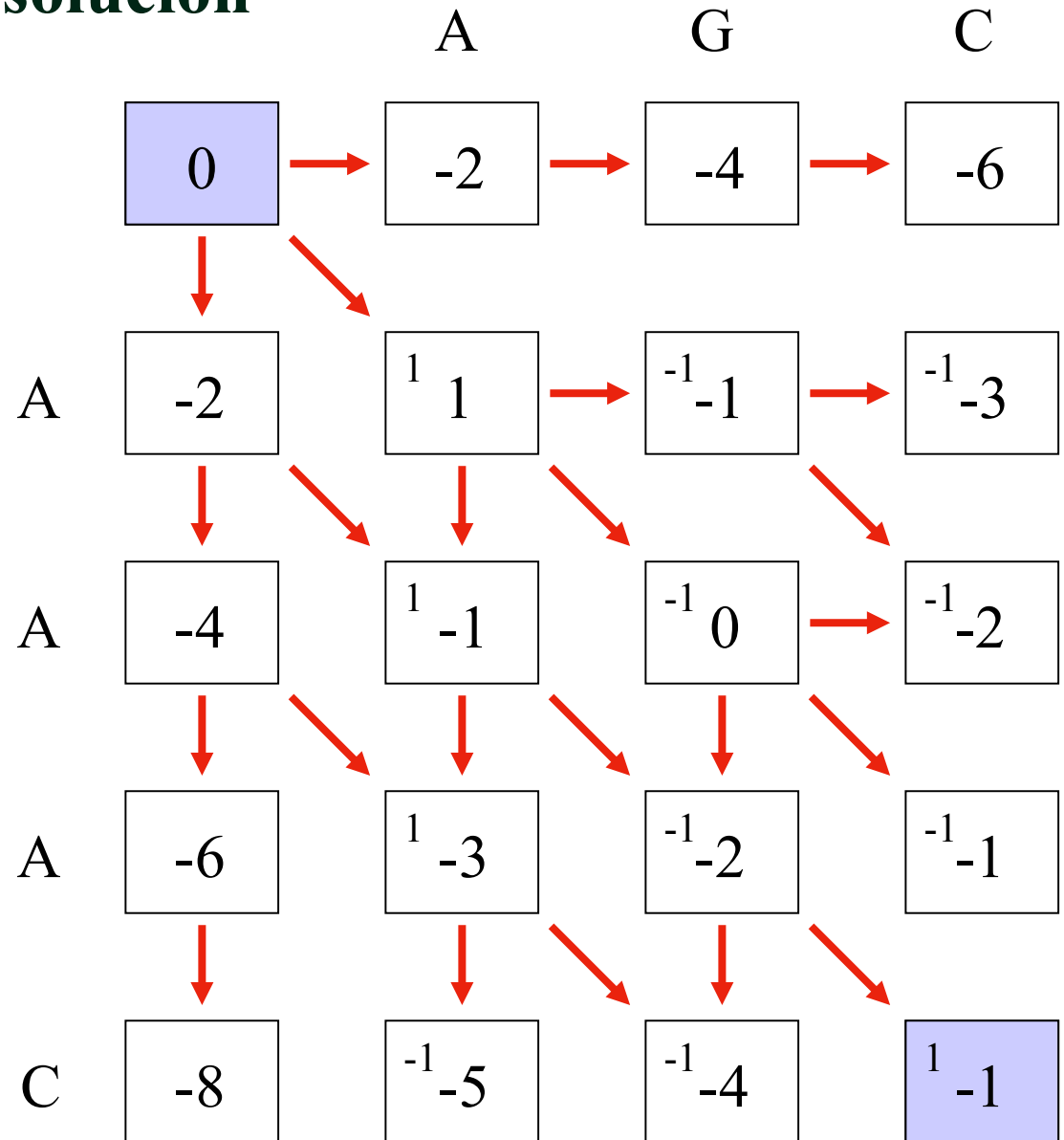


Construcción del alineamiento

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(i, j) \\ F(i-1, j) + g \\ F(i, j-1) + g \end{cases}$$



Recuperación de la solución

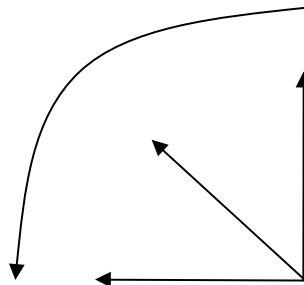


Recuperación de la solución (*Backtracking*)

Consiste en tomar la última coincidencia del alineamiento y comenzar a buscar el camino que maximice la función, es decir siguiendo los valores más altos

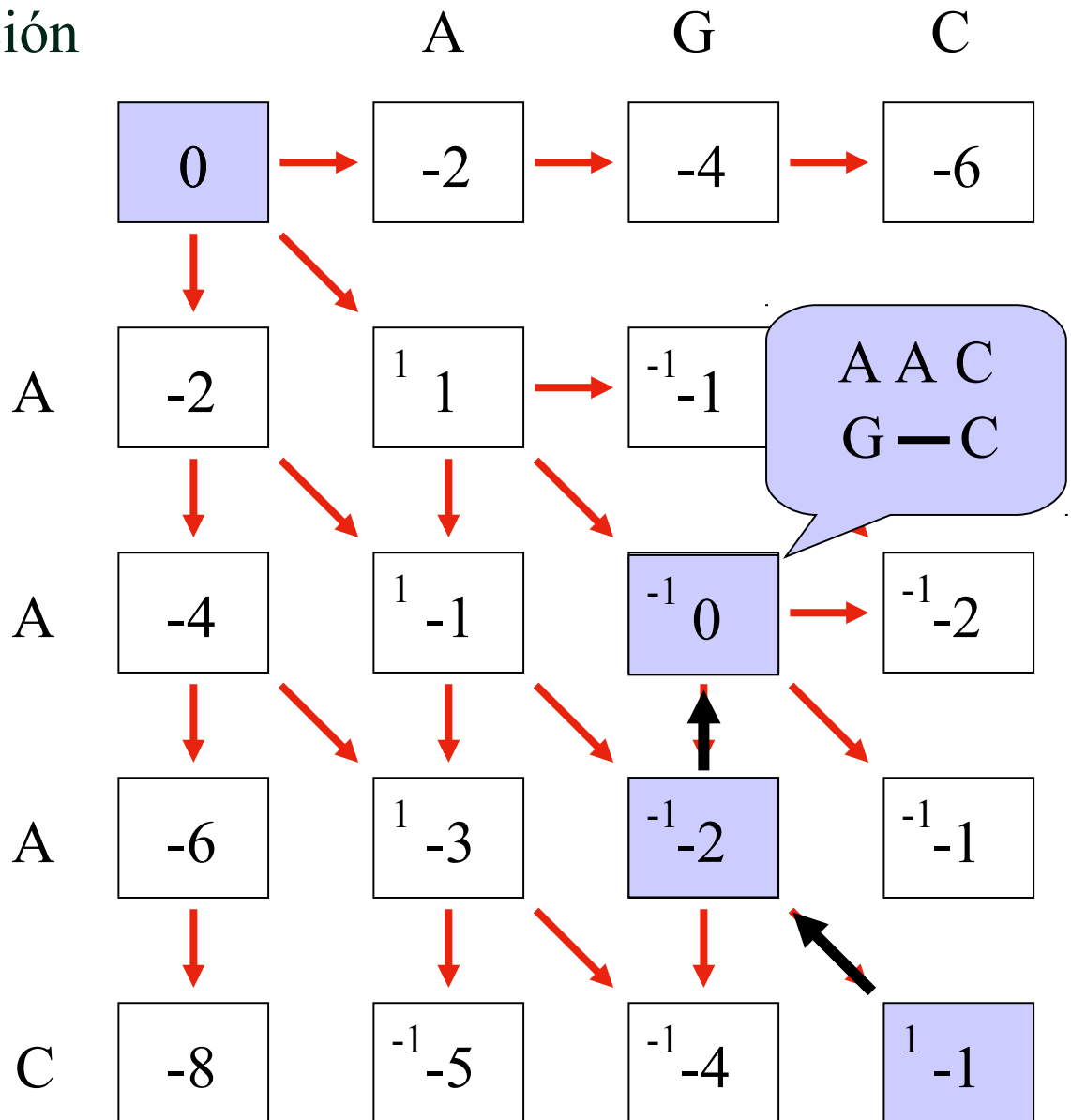
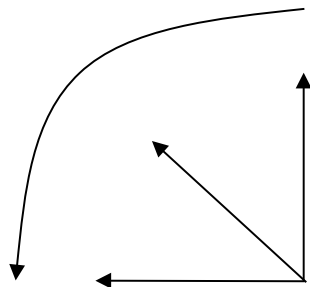
Cuando los valores coinciden, cualquier camino que se tome conducirá a la formación de un alineamiento de igual score. Sin embargo los algoritmos para resolverse deben dar preferencia a uno de los valores, en nuestro caso se toma preferencia en el sentido horario

Máxima preferencia



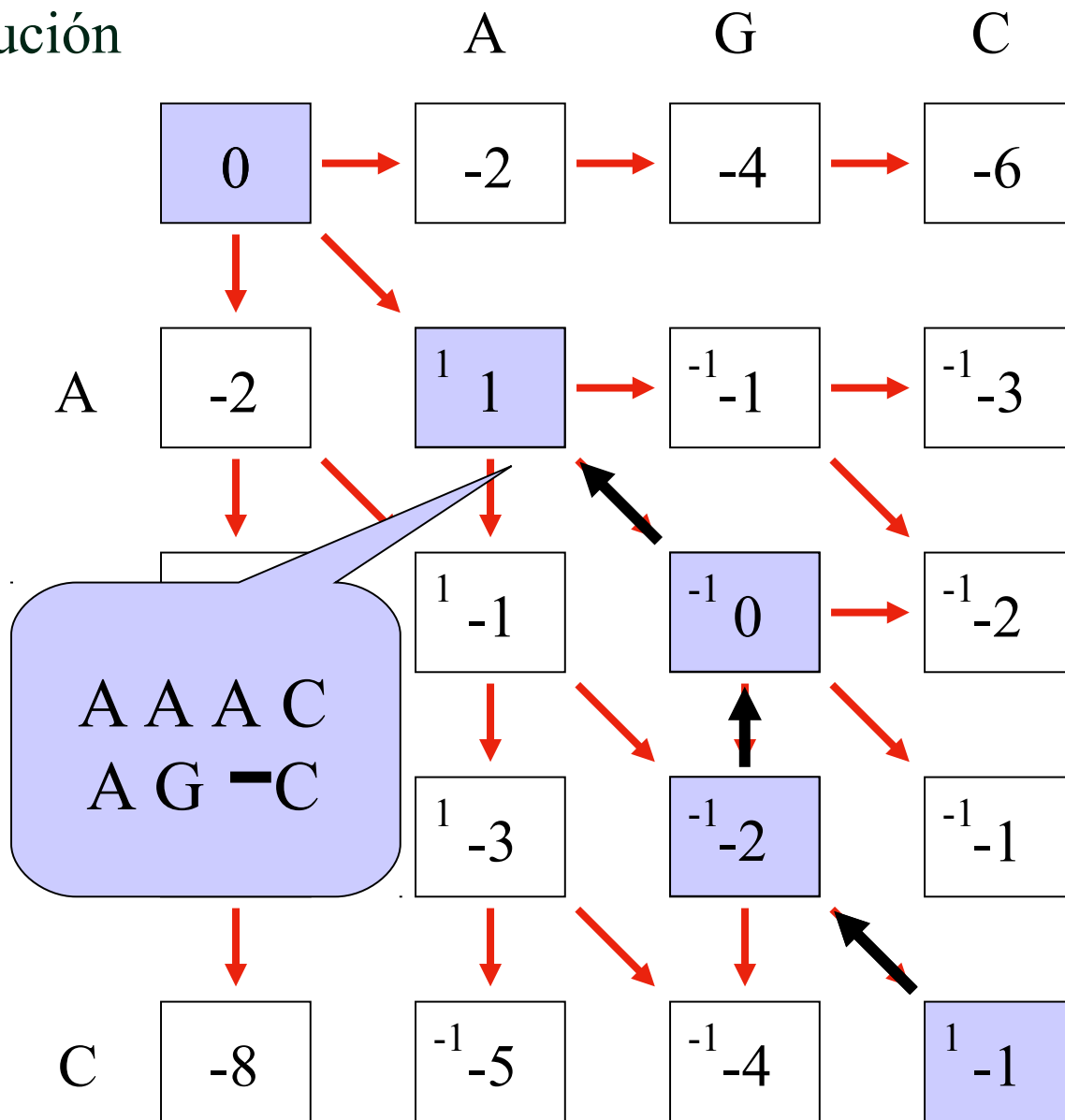
Recuperación de la solución

Máxima preferencia



Recuperación de la solución

Máxima preferencia



Algoritmo de Alineamiento Global

- Input: 2 secuencias: S y T.
- Problema:
 - Cuál es la máxima similitud entre las secuencias?
 - Encontrar el mejor alineamiento.
- Global: Que para cada String, implica alinear toda el String.

Parámetros

- Valores para:
 - Penalidades de los GAPS (valor g)
 - $P(i,j)$:
 - Valor de Matching. $S(i) = T(j)$
 - Valor de Mismatching. $S(i) \neq T(j)$
- De acuerdo a los parámetros la alineación puede variar.

Pasos del Algoritmo

1. Inicialización.
2. Llenado de la matriz.
3. Búsqueda de secuencias óptimas.

Inicialización

- Si se quiere alinear las secuencias S y T.
- Si n es el tamaño de S y m el de T.
- Hay $n+1$ posibles prefijos de S y $m+1$ posibles prefijos de T (incluyendo el string vacío).
- Se generará una matriz F de $(n+1) \times (m+1)$ elementos.
- El elemento $F(i, j)$ tendrá la mejor similitud entre las subsecuencias $S[1...i]$ y $T[1...j]$.

Llenado de la Matriz

- Se llena la matriz la primera fila y la primera columna con múltiplos de la penalidad por espacio.
- Se llenan los elementos interiores

$$\mathbf{F[i,j] = \max:} \quad \left\{ \begin{array}{l} F[i, j-1] + g \\ F[i-1, j-1] + p(i,j) \\ F[i-1, j] + g \end{array} \right.$$

- g = penalidad por espacio, $p(i,j)$ = valor del matching entre $S(i)$ y $T(j)$
- Sigue cualquier orden que permita tener los valores $F[i, j-1]$, $F[i-1, j-1]$, $F[i-1, j]$. Como fila por fila (de izquierda a derecha) o columna por columna (de arriba hacia abajo).

Con $g = -2$

Para i , desde 0 hasta m hacer

$$F[i, 0] = i \times g$$

Para i , desde 0 hasta n hacer

$$F[0, i] = i \times g$$

S

T

		T			
			A	G	C
S		0	-2	-4	-6
	A	-2			
	A	-4			
	A	-6			
	C	-8			

Con $g = -2$

Para i , desde 0 hasta n hacer

$$F[i, 0] = i \times g$$

Para i , desde 0 hasta n hacer

$$F[0, i] = i \times g$$

Para i , desde 1 hasta n hacer

Para j , desde 1 hasta m hacer

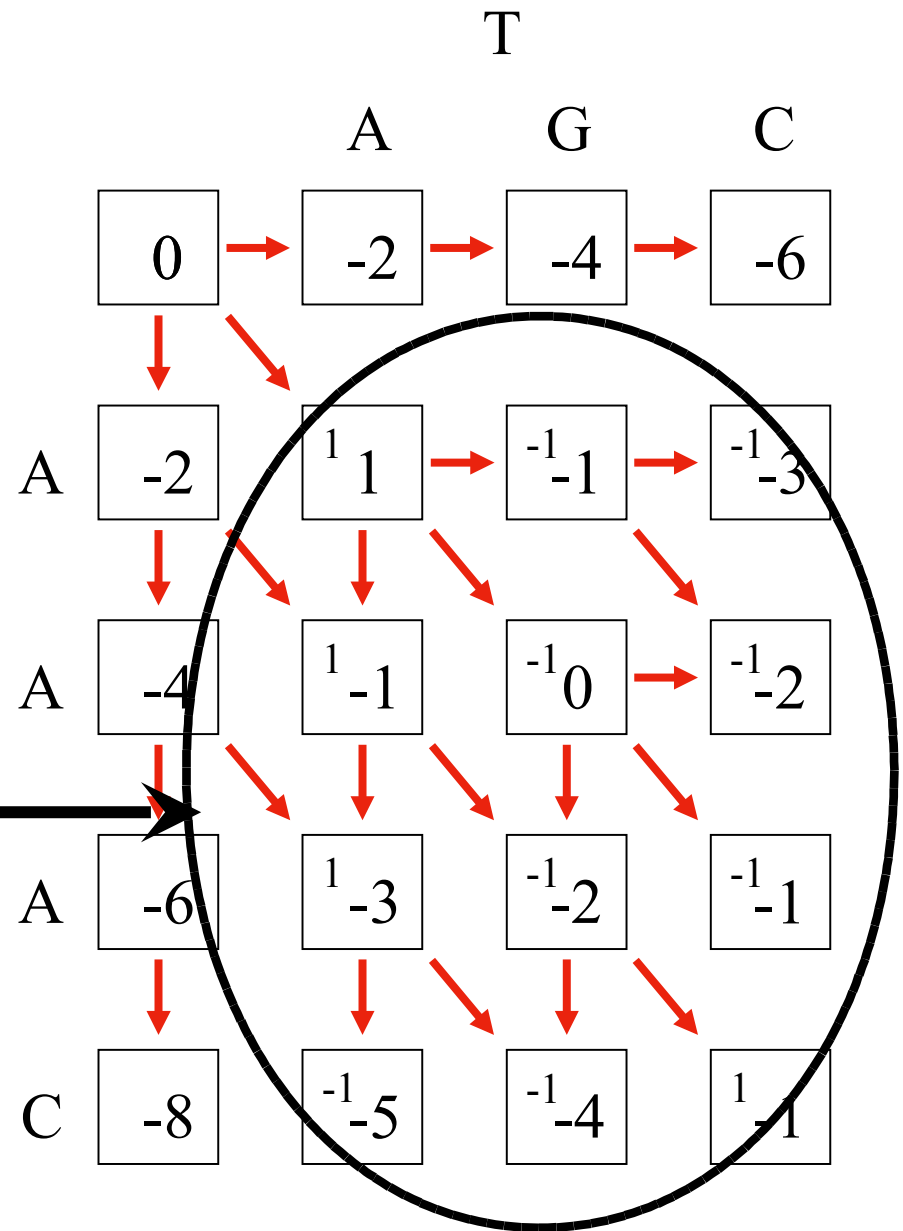
$$F[i, j] = \max \left(\begin{array}{l} F[i-1, j] + g, \\ F[i-1, j-1] + p(i, j) \\ F[i, j-1] + g \end{array} \right)$$

$$i = 1 \longrightarrow j = 1 \dots m \longrightarrow F(1, j)$$

$$i = 2 \longrightarrow j = 1 \dots m \longrightarrow F(2, j)$$

...

$$i = n \longrightarrow j = 1 \dots m \longrightarrow F(n, j)$$



Algoritmo

$n = \text{long}(S)$

$m = \text{long}(t)$

Para i , desde 0 **hasta** n **hacer**

$F[i, 0] = i \times g$

Para j , desde 0 **hasta** m **hacer**

$F[0, j] = j \times g$

Para i , desde 1 **hasta** n **hacer**

Para j , desde 1 **hasta** m **hacer**

$F[i, j] = \text{máximo}(F[i-1, j] + g, F[i-1, j-1] + p(i, j), F[i, j-1] + g).$

Devolver F

Busqueda de secuencias óptimas

- Ahora que conocemos la máxima similitud entre 2 secuencias necesitamos construir el alineamiento óptimo entre ellas.
- Se comienza desde el valor de $F[n,m]$ y se realiza el camino inverso hasta llegar al valor $F[0,0]$.
- Por cada valor $F[i,j]$ se testea si provino de $F[i-1, j-1] + p(i,j)$ o $F[i-1, j] - g$ o $F[i, j-1] - g$.

Algoritmo de Backtracking

- Este algoritmo va a construir el alineamiento óptimo dado la matriz F y las secuencias originales S y T .
- La variable “len” contiene la longitud del alineamiento (que puede ser $n+m$)
- Los vectores (secuencia de valores) “aligS” y “aligT” contienen los caracteres alineados.

INPUT: indices i, j , array F .

Alinear(i, j)

Si $i=0$ y $j = 0$ **entonces**

len = 0

Sino Si $i>0$ y $F[i,j] = F[i-1, j] + g$ **entonces**

Alinear($i-1, j, \text{len}$)

len = len + 1

aligS[len] = s[i]

aligT[len] = - //(espacio)

Sino Si $i>0$ y $j>0$ y $F[i,j] = F[i-1, j-1] + p(i,j)$ **ent**

Alinear($i-1, j-1, \text{len}$)

len = len + 1

aligS[len] = s[i]

aligT[len] = t[j]

Sino // debe ser $F[i,j] = F[i, j-1] + g$

Alinear($i, j-1, \text{len}$)

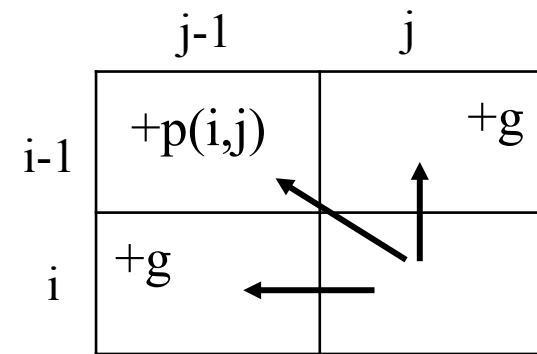
len = len + 1

aligS[len] = - //(espacio)

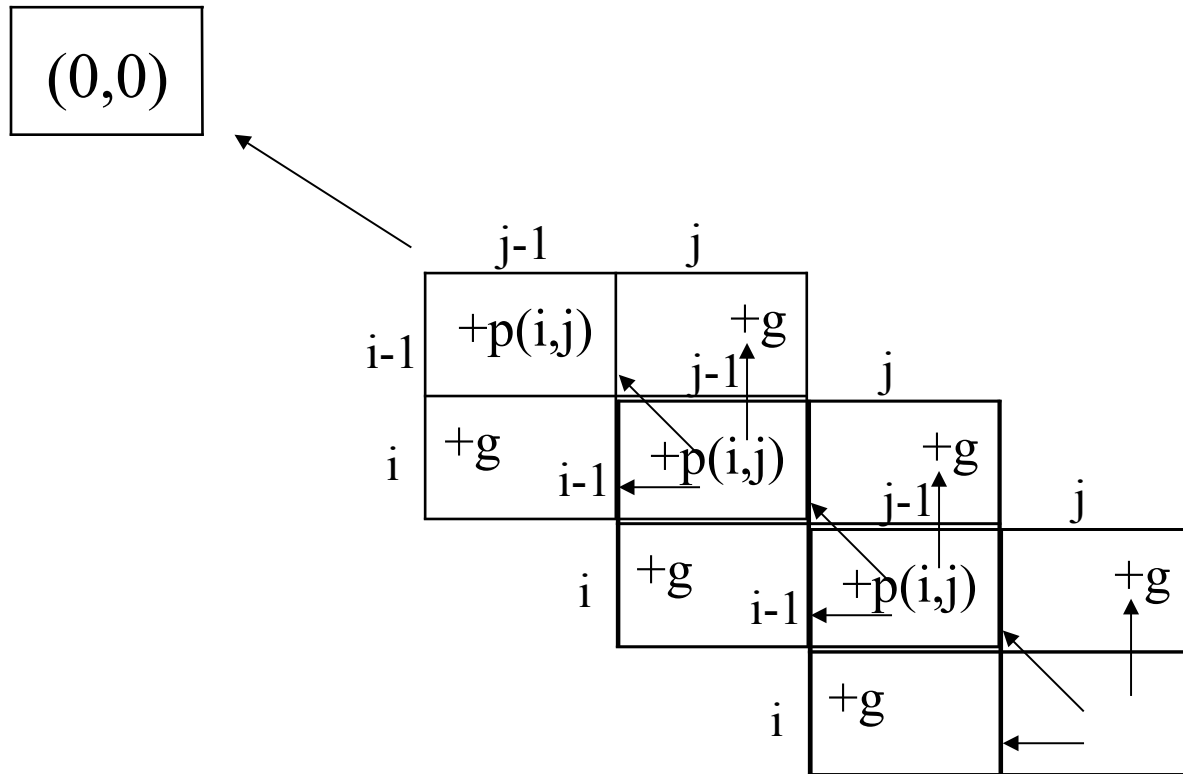
aligT[len] = t[j]

Fin

Devolver aligS, aligT y len.

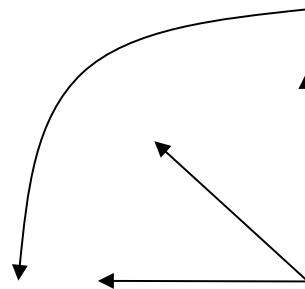


Recursión en el Algoritmo



- Se pueden generar varios caminos óptimos.
- El orden en el que las preguntas estan hechas dan la preferencia de elección de un camino óptimo sobre otro.

Máxima preferencia



Complejidad

$m = \text{long}(S)$

$n = \text{long}(t)$

Para i , desde 0 hasta n **hacer**

$F[i, 0] = i \times g$

} **n veces**

Para j , desde 0 hasta m **hacer**

$F[0, j] = j \times g$

} **m veces**

Para i , desde 1 hasta n **hacer**

m veces { **Para** j , desde 1 hasta m **hacer**

$F[i, j] = \text{máximo}(F[i-1, j] + g, F[i-1, j-1] + p(i, j), F[i, j-1] + g).$

} **n veces**

Devolver $F[n, m]$

Complejidad

- Complejidad Temporal:
 - $O(mn)$ en armar la matriz
 - $O(m + n)$ en buscar en la matriz (recorrer la secuencia resultante, que a lo sumo tiene $n+m$ elementos).
- Complejidad Espacial:
 - $O(mn)$ por el espacio necesario para la matriz

Alineamiento Semi-Global

- Ignoramos los espacios al final y al principio de las secuencias.
- Ahora los Strings se pueden solapar, no necesitan tener la misma longitud.

G C A G - C G T T A C G T T T C A
- - - G A C G A T A - - - - -

Tiene un score de 5 matches, 1 mismatch y 1 espacio.

Diferencias con el Global

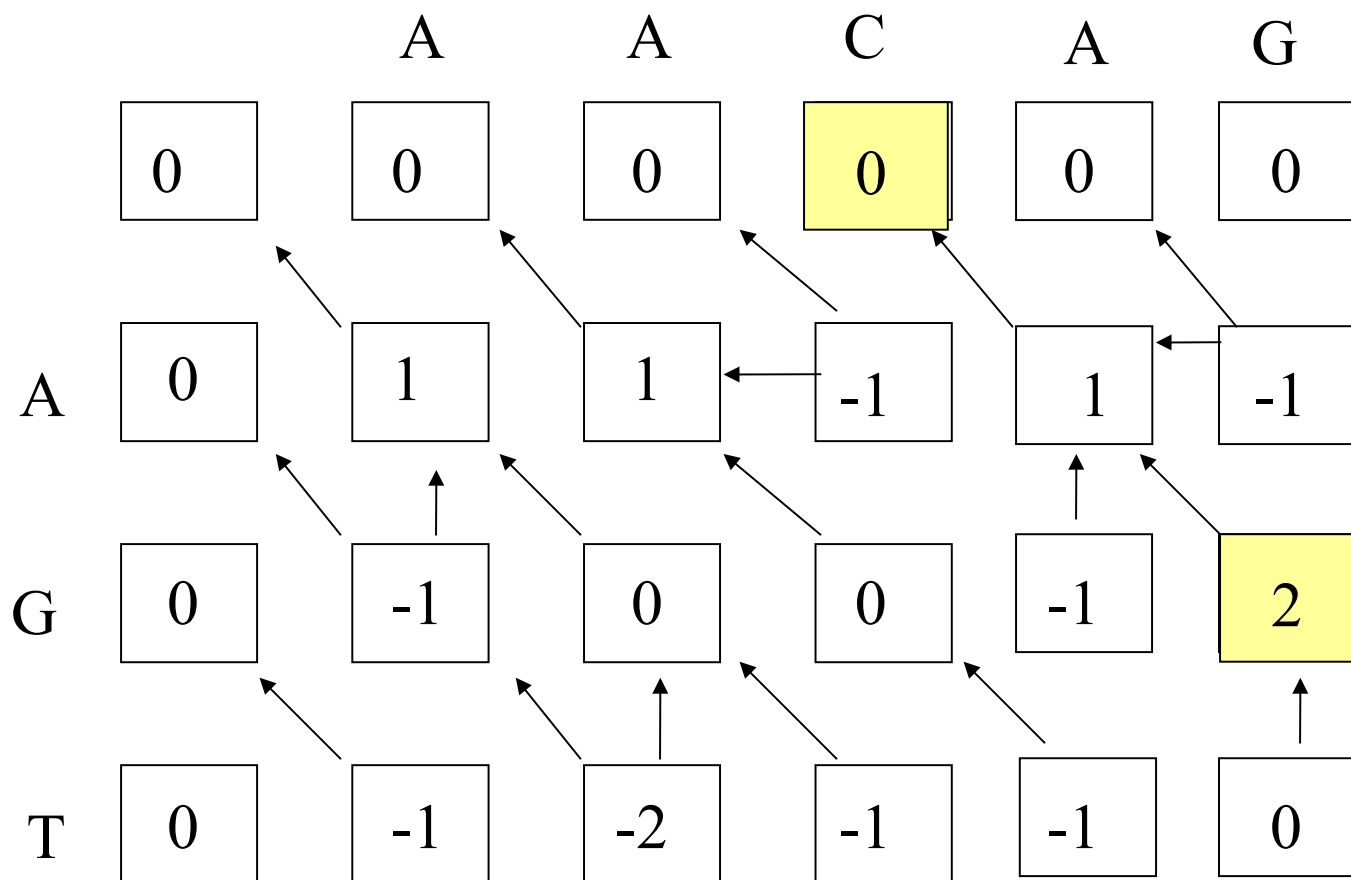
- Se inicializan la primera columna y fila en 0.
- Cuando se busca el mejor alineamiento en vez de tomar el valor $F[n,m]$ se busca el máximo de la última fila y de la última columna.
 - $\text{Similitud}(S,T) = \text{Máximo} \begin{cases} \text{Máximo } F(i, m) \\ \text{Máximo } F(n, i) \end{cases}$

Complejidad

- Tiene la misma complejidad que el algoritmo global.
- Solo que ahora también tarda $O(n+m)$ en buscar el mejor.

S = AGT

T = AACAG



S' = - - - A G T

T' = A A C A G -

Algoritmos dinámicos.

- **Alineamiento global (Needleman y Wunsch, 1970).**

- Encontrar un alineamiento de dos secuencias completas (se introducen huecos para igualar sus longitudes) de tal manera que proporcione un valor numérico global máximo. Es Algoritmo relativamente lento que usa mucha memoria.
- Ofrece buenos resultados para secuencias con alta similitud, tanto en la longitud como en el contenido de éstas.

- **Alineamiento local (Smith y Waterman, 1981).**

- Encontrar fragmentos de dos secuencias que proporcionen alineamientos con puntuación máxima (se alinean las partes más parecidas).
- Suele ser una combinación de muchos alineamientos globales de secuencias cortas.
- Ofrece buenos resultados para secuencias con baja similitud que contengan regiones parecidas

BLAST (Basic Local Alignment Search Tool): principal programa de alineamiento local de secuencias (es “una evolución” del algoritmo de Smith-Waterman de alineamiento local).

Programación dinámica

Alineamiento Global

Needleman & Wunsch (1970)

Encuentra el mejor scoring para un alineamiento global de dos secuencias

- Inicialización de la matriz
- Llenado de Matriz con los score individuales
- Construcción del alinamiento que maximice el score total
- Recuperación de la solución (Backtracking)

Alineamiento Local

Smith-Waterman (1981)

En el alineamiento local, los algoritmos son básicamente iguales a los que vimos en alineamiento global.

Las diferencias son:

- Se agrega una posibilidad más al generar la matriz

$$\text{sim}(S[1..i], T[1..j]) = \max \begin{cases} \text{sim}(S[1..i], T[1..j-1]) + g \\ \text{sim}(S[1..i-1], T[1..j-1]) + p(i,j) \\ \text{sim}(S[1..i-1], T[1..j]) + g \\ 0 \end{cases}$$

- La alineación va desde el máximo de toda la matriz hasta encontrar el primer 0

Ejemplo

Consideremos las secuencias:

$S = \text{AAGG}$

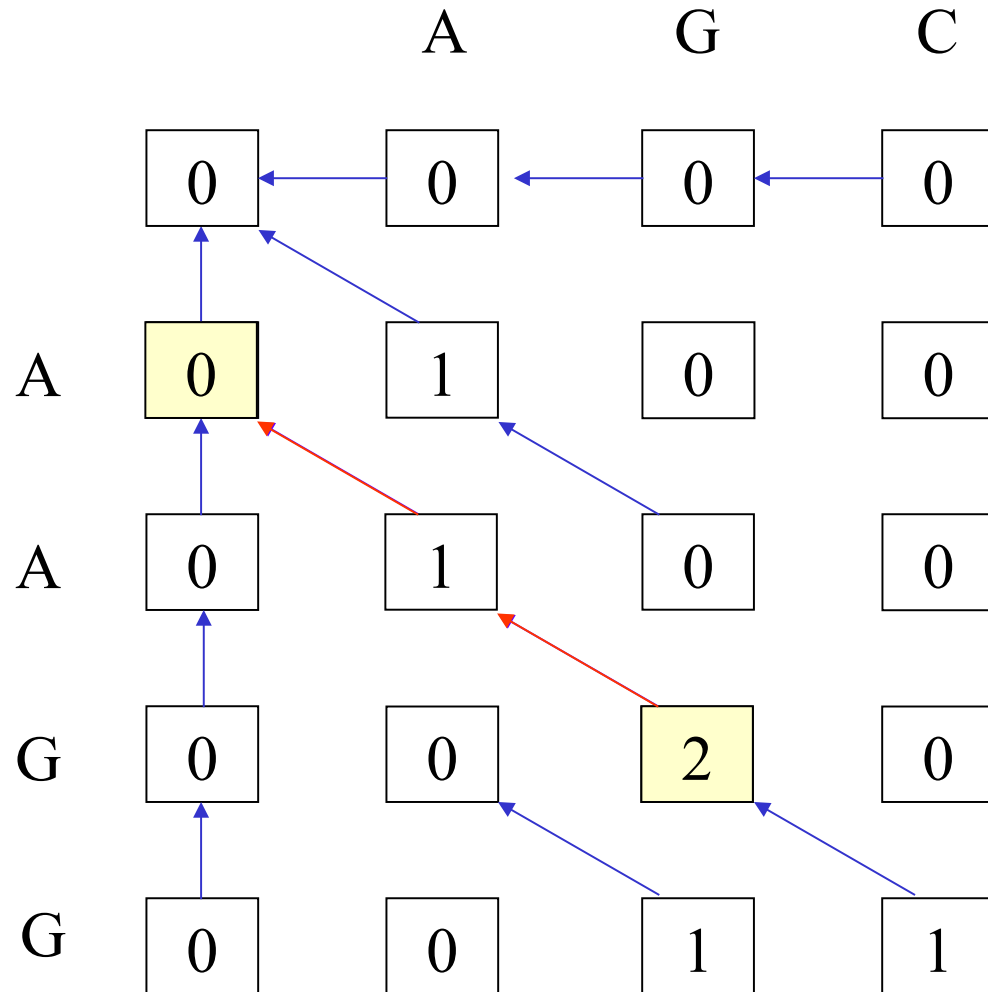
$T = \text{AGC}$

Aplicando el
algoritmo
podemos obtener
la siguiente
matriz:

$m=1$

$s=-1$

$g=-2$



Vamos a considerar otro ejemplo de ilustración del algoritmo de Smith-Waterman.

Tratemos de alinear localmente el par de secuencias

G G A T C G A

G A A T T C A G T T A

Consideremos la siguiente matriz de similitud era:










$$s(i, j) = \begin{cases} +5 & \text{si } A[i] = B[j] \\ -3 & \text{si } A[i] \neq B[j] \end{cases}$$

y la penalización por huecos es $g = -4$.

Procedamos a la fase de relleno de la matriz en el caso del algoritmo local de Smith-Waterman.

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	5	1	0	0	0	0	0	5	1	0	0
G	0	5	2	0	0	0	0	0	5	2	0	0
A	0	1	10	7	3	0	0	5	1	2	0	5
T	0	0	6	7	12	8	4	1	2	6	7	3
C	0	0	2	3	8	9	13	9	5	2	3	4
G	0	5	1	0	4	5	9	10	14	10	6	2
A	0	1	10	6	2	1	5	14	10	11	7	11

A continuación, vamos a determinar el correspondiente TRACE-BACK.

		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	 5	1	0	0	0	0	0	5	1	0	0
G	0	5	 2	0	0	0	0	0	5	2	0	0
A	0	1	10	 7	 3	0	0	5	1	2	0	5
T	0	0	6	7	 12	 8	4	1	2	6	7	3
C	0	0	2	3	8	9	 13	9	5	2	3	4
G	0	5	1	0	4	5	 9	10	14	10	6	2
A	0	1	10	6	2	1	5	 14	10	11	7	11

Finalmente decodificamos el TRACE-BACK: en este ejemplo hemos obtenido dos alineamientos locales óptimos

★ Primer alineamiento local óptimo:

↖	↖	↖	↖	←	↖	↑	↖
5	2	7	12	8	13	9	14
G	G	A	T	-	C	G	A
G	A	A	T	T	C	-	A

Valor óptimo del alineamiento local: $5 \cdot (+5) + 1 \cdot (-3) + 2 \cdot (-4) = +14$.

★ Segundo alineamiento local óptimo:

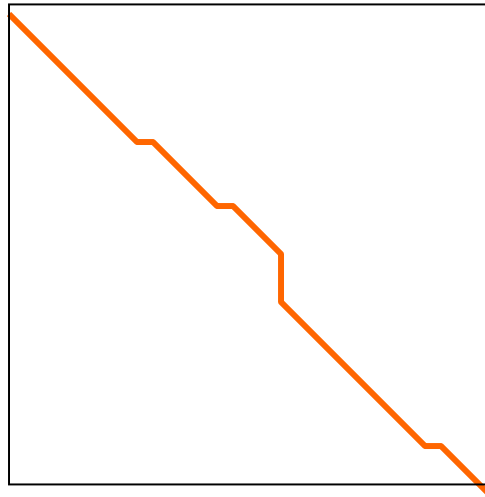
↖	↖	↖	←	↖	↖	↑	↖
5	2	7	3	8	13	9	14
G	G	A	-	T	C	G	A
G	A	A	T	T	C	-	A

Valor óptimo del alineamiento local: $5 \cdot (+5) + 1 \cdot (-3) + 2 \cdot (-4) = +14$.

Similitud global y local

Alineamiento global

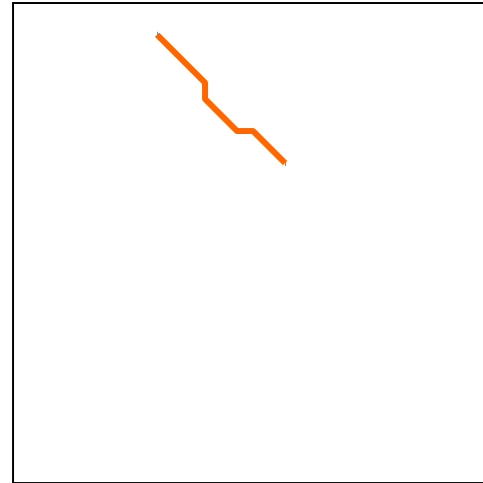
Needleman & Wunsch (1970)



Las secuencias se alinean esencialmente de un extremo a otro

Alineamiento local

Smith & Waterman (1981)



Las secuencias se alinean en regiones pequeñas y aisladas

Algoritmo de Alineamiento Local

- Input: Secuencias S y T.
- Cuál es la máxima similitud entre una Subsecuencia de S y una Subsecuencia de T?
- Encontrar las subsecuencias más similares

Diferencias con el global

- Los mismatch deben ser negativos.
- Se inicializan la primera columna y primera fila en 0.
- Cuando el score se vuelve negativo se debe poner 0.
- El alineamiento se produce al buscar el mayor valor en la matriz, y se sigue el camino hacia atrás hasta encontrar un 0.

$$F[i,j] = \text{máximo} \begin{cases} F[i,j-1] + g \\ F[i-1,j-1] + p(i,j) \\ F[i-1,j] + g \\ 0 \end{cases}$$

- El menor valor posible de score dentro de la matriz es 0.
- Complejidad Temporal: $O(mn)$
- Complejidad Espacial: $O(m+n)$

Esquemas de Puntuación

Capturar el significado biológico de las semejanzas

- Un sistema de scoring simple, penaliza por igual cualquier mismatch.
- Biológicamente tiene sentido penalizar ciertos cambios y ser más permisivo con otros. Por ejemplo:
 - En proteínas → residuos hidrofóbicos reemplazados entre sí.
 - En DNA → transversiones vs transiciones

Se crean matrices con un sistema de scoring que permite asignar puntajes individuales a cada una de las letras del alfabeto en uso.

Matrices

→ Un ejemplo de matriz de scoring podría ser el clásico ejemplo de penalizar más los cambios que alteran las propiedades químicas de un residuo (aa)

Hidrofóbicos: Ile, Val, Leu, Ala

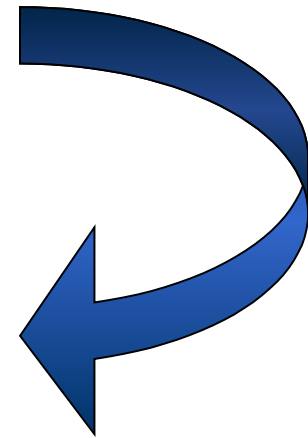
Polares (+): Lys, Arg

Polares (-): Glu, Asp

Aromáticos: Phe, Tyr, Trp

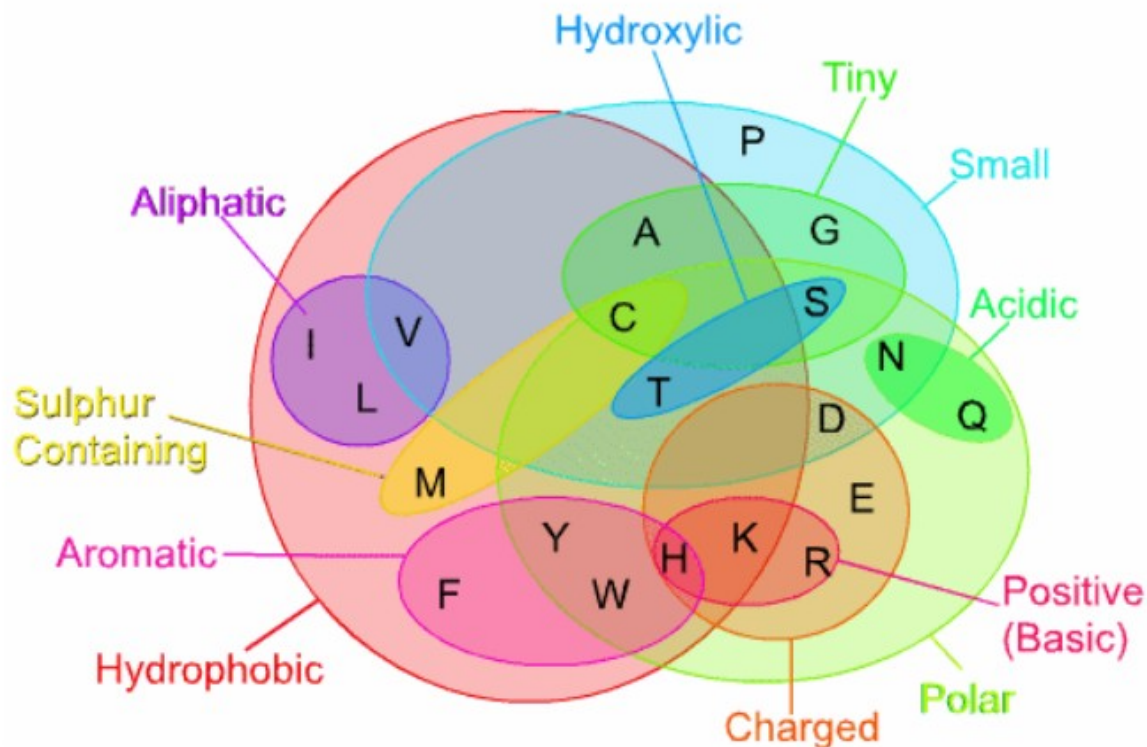
etc.

Ile x Val = -1
Ile x Asp = -5
Phe x Tyr = -1
Phe x Gly = -8



Matrices

alanine - ala - A
 arginine - arg - R
 asparagine - asn - N
 aspartic acid - asp - D
 cysteine - cys - C
 glutamine - gln - Q
 glutamic acid - glu - E
 glycine - gly - G
 histidine - his - H
 isoleucine - ile - I
 leucine - leu - L
 lysine - lys - K
 methionine - met - M
 phenylalanine - phe - F
 proline - pro - P
 serine - ser - S
 threonine - thr - T
 tryptophan - trp - W
 tyrosine - tyr - Y
 valine - val - V



Ile x Val = -1
 Ile x Asp = -5
 Phe x Tyr = -1
 Phe x Gly = -8

Si yo sé cuáles cambios son más o menos comunes en un gran número de proteínas puedo asistir a la predicción de alineamientos de un set de secuencias.

Matrices PAM

“Point Accepted Mutation”

- Dayhoff, 1978 - Implica modelo evolutivo.

- diseñado para detectar orígenes evolutivos de las proteínas.
- derivadas a partir de un grupo de secuencias > 85% similares.
- PAM = “point accepted mutations” unidad de divergencia evolutiva. 1 PAM = 1 aa sustituido cada 100 residuos.
- Mutaciones aceptadas porque no cambian funcionalidad.
- De los pares de secuencias altamente relacionadas se recolecta la información de 1 PAM... luego se extrapolan a períodos evolutivos más largos, (120, 250, 320 PAMs).

Matrices PAM

“Point Accepted Mutation”

1978: **Dayhoff** et al. estudiaron 1572 sustituciones en 71 grupos de proteínas muy parecidas entre sí

- Accepted Point Mutation (PAM): sustitución de un aminoácido por otro, que ha sido aceptada por la selección natural para una determinada proteína

1992: Henikof y Henikof mejoran la matriz de Dayhoff con una base de datos de 500 alineamientos (BLOCKS) entre proteínas poco parecidas entre sí



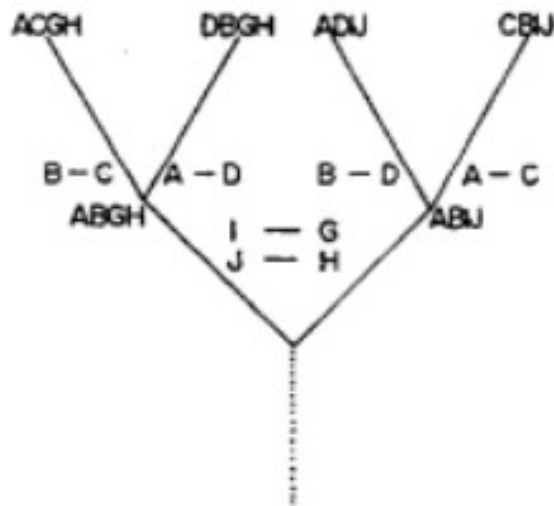
Margaret Dayhoff

Matrices PAM

“Point Accepted Mutation”

Se reconstruye un árbol filogenético para cada uno de los 71 grupos de proteínas (parecidas entre sí, $\geq 85\%$ de similitud)

- Y se determinan cuántos reemplazos (mutaciones aceptadas) hay respecto al ancestro común



	A	B	C	D	G	H	I	J
A			1	1				
B			1	1				
C	1	1						
D	1	1						
G							1	
H								1
I					1			
J						1		

Matrices PAM

“Point Accepted Mutation”

	A Ala	R Arg	N Asn	D Asp	C Cys	Q Gln	E Glu	G Gly	H His	I Ile	L Leu	K Lys	M Met	F Phe	P Pro	S Ser	T Thr	W Trp	Y Tyr	V Val
A																				
R	30																			
N	109	17																		
D	154	0	532																	
C	33	10	0	0																
Q	93	120	50	76	0															
E	266	0	94	831	0	422														
G	579	10	156	162	10	30	112													
H	21	103	226	43	10	243	23	10												
I	66	30	36	13	17	8	35	0	3											
L	95	17	37	0	0	75	15	17	40	253										
K	57	477	322	85	0	147	104	60	23	43	39									
M	29	17	0	0	0	20	7	7	0	57	207	90								
F	20	7	7	0	0	0	0	17	20	90	167	0	17							
P	345	67	27	10	10	93	40	49	50	7	43	43	4	7						
S	772	137	432	98	117	47	86	450	26	20	32	168	20	40	269					
T	590	20	169	57	10	37	31	50	14	129	52	200	28	10	73	696				
W	0	27	3	0	0	0	0	0	3	0	13	0	0	10	0	17	0			
Y	20	3	36	0	30	0	10	0	40	13	23	10	0	260	0	22	23	6		
V	365	20	13	17	33	27	37	97	30	661	303	17	77	10	50	43	186	0	17	
	A Ala	R Arg	N Asn	D Asp	C Cys	Q Gln	E Glu	G Gly	H His	I Ile	L Leu	K Lys	M Met	F Phe	P Pro	S Ser	T Thr	W Trp	Y Tyr	V Val

A_{LEU}

A_{ij}

1572 mutaciones aceptadas (PAMs) estudiadas entre los distintos aminoácidos, multiplicadas por 10
 Por ejemplo, 20.7 de las 1572 PAMs ocurren entre Leu y Met
 Leu tiene un total de 142.8 mutaciones relacionadas

Matrices PAM

"Point Accepted Mutation"

Para Ala se toma
un valor arbitrario
de 100

Asn	134	His	66
Ser	120	Arg	65
Asp	106	Lys	56
Glu	102	Pro	56
Ala	100	Gly	49
Thr	97	Tyr	41
Ile	96	Phe	41
Met	94	Leu	40
Gln	93	Cys	20
Val	74	Trp	18

m_j

Mutabilidad relativa
según Dayhoff et al. 1978

$$M_{ij} = \frac{\lambda m_j A_{ij}}{A_j}$$

- ♦ M_{ij} – probabilidad de que el aminoácido j cambie al aminoácido i en un intervalo evolutivo dado
- ♦ λ - constante de proporcionalidad (para Dayhoff, ~ 0.013)
- ♦ $M_{\text{MET-LEU}} = \lambda \cdot m_{\text{LEU}} \cdot A_{\text{MET-LEU}} / A_{\text{LEU}} = 0.013 \cdot 40 \cdot 207 / 1428 = 0.08\%$

Matrices PAM

“Point Accepted Mutation”

	A Ala	R Arg	N Asn	D Asp	C Cys	Q Gln	E Glu	G Gly	H His	I Ile	L Leu	K Lys	M Met	F Phe	P Pro	S Ser	T Thr	W Trp	Y Tyr	V Val
A	98.67	0.02	0.09	0.10	0.03	0.08	0.17	0.21	0.02	0.06	0.04	0.02	0.06	0.02	0.22	0.35	0.32	0.00	0.02	0.18
R	0.01	99.13	0.01	0.00	0.01	0.10	0.00	0.00	0.10	0.03	0.01	0.19	0.04	0.01	0.04	0.06	0.01	0.08	0.00	0.01
N	0.04	0.01	98.22	0.36	0.00	0.04	0.06	0.06	0.21	0.03	0.01	0.13	0.00	0.01	0.02	0.20	0.09	0.01	0.04	0.01
D	0.06	0.00	0.42	98.59	0.00	0.06	0.53	0.06	0.04	0.01	0.00	0.03	0.00	0.00	0.01	0.05	0.03	0.00	0.00	0.01
C	0.01	0.01	0.00	0.00	99.73	0.00	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.01	0.05	0.01	0.00	0.03	0.02
Q	0.03	0.09	0.04	0.05	0.00	98.76	0.27	0.01	0.23	0.01	0.03	0.06	0.04	0.00	0.06	0.02	0.02	0.00	0.00	0.01
E	0.10	0.00	0.07	0.56	0.00	0.35	98.65	0.04	0.02	0.03	0.01	0.04	0.01	0.00	0.03	0.04	0.02	0.00	0.01	0.02
G	0.21	0.01	0.12	0.11	0.01	0.03	0.07	99.35	0.01	0.00	0.01	0.02	0.01	0.01	0.03	0.21	0.03	0.00	0.00	0.05
H	0.01	0.08	0.18	0.03	0.01	0.20	0.01	0.00	99.12	0.00	0.01	0.01	0.00	0.02	0.03	0.01	0.01	0.01	0.04	0.01
I	0.02	0.02	0.03	0.01	0.02	0.01	0.02	0.00	0.00	98.72	0.09	0.02	0.21	0.07	0.00	0.01	0.07	0.00	0.01	0.33
L	0.03	0.01	0.03	0.00	0.00	0.06	0.01	0.01	0.04	0.22	99.47	0.02	0.45	0.13	0.03	0.01	0.03	0.04	0.02	0.15
K	0.02	0.37	0.25	0.06	0.00	0.12	0.07	0.02	0.02	0.04	0.01	99.26	0.20	0.00	0.03	0.08	0.11	0.00	0.01	0.01
M	0.01	0.01	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.05	0.08	0.04	98.74	0.01	0.00	0.01	0.02	0.00	0.00	0.04
F	0.01	0.01	0.01	0.00	0.00	0.00	0.00	0.01	0.02	0.08	0.06	0.00	0.04	99.46	0.00	0.02	0.01	0.03	0.28	0.00
P	0.13	0.05	0.02	0.01	0.01	0.08	0.03	0.02	[Sin título]	0.02	0.02	0.01	0.01	99.26	0.12	0.04	0.00	0.00	0.02	
S	0.28	0.11	0.34	0.07	0.11	0.04	0.06	0.16	0.02	0.02	0.01	0.07	0.04	0.03	0.17	98.40	0.38	0.05	0.02	0.02
T	0.22	0.02	0.13	0.04	0.01	0.03	0.02	0.02	0.01	0.11	0.02	0.08	0.06	0.01	0.05	0.32	98.71	0.00	0.02	0.09
W	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00	99.76	0.01	0.00
Y	0.01	0.00	0.03	0.00	0.03	0.00	0.01	0.00	0.04	0.01	0.01	0.00	0.00	0.21	0.00	0.01	0.01	0.02	99.45	0.01
V	0.13	0.02	0.01	0.01	0.03	0.02	0.02	0.03	0.03	0.57	0.11	0.01	0.17	0.01	0.03	0.02	0.10	0.00	0.02	99.01

Matriz con probabilidades de mutación PAM1

1 PAM se define como la unidad de divergencia evolutiva en la que han ocurrido un 1% de mutaciones entre dos secuencias de proteínas

Matrices PAM

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
Ala A	13	6	9	9	5	8	9	12	6	8	6	7	7	4	11	11	11	2	4	9
Arg R	3	17	4	3	2	5	3	2	6	3	2	9	4	1	4	4	3	7	2	2
Asn N	4	4	6	7	2	5	6	4	6	3	2	5	3	2	4	5	4	2	3	3
Asp D	5	4	8	11	1	7	10	5	6	3	2	5	3	1	4	5	5	1	2	3
Cys C	2	1	1	1	52	1	1	2	2	2	1	1	1	1	2	3	2	1	4	2
Gln Q	3	5	5	6	1	10	7	3	7	2	3	5	3	1	4	3	3	1	2	3
Glu E	5	4	7	11	1	9	12	5	6	3	2	5	3	1	4	5	5	1	2	3
Gly G	12	5	10	10	4	7	9	27	5	5	4	6	5	3	8	11	9	2	3	7
His H	2	5	5	4	2	7	4	2	15	2	2	3	2	2	3	3	2	2	3	2
Ile I	3	2	2	2	2	2	2	2	2	10	6	2	6	5	2	3	4	1	3	9
Leu L	6	4	4	3	2	6	4	3	5	15	34	4	20	13	5	4	6	6	7	13
Lys K	6	18	10	8	2	10	8	5	8	5	4	24	9	2	6	8	8	4	3	5
Met M	1	1	1	1	0	1	1	1	1	2	3	2	6	2	1	1	1	1	1	2
Phe F	2	1	2	1	1	1	1	1	3	5	6	1	4	32	1	2	2	4	20	3
Pro P	7	5	5	4	3	5	4	5	5	3	3	4	3	2	20	6	5	1	2	4
Ser S	9	6	8	7	7	6	7	9	6	5	4	7	5	3	9	10	9	4	4	6
Thr T	8	5	6	6	4	5	5	6	4	6	4	6	5	3	6	8	11	2	3	6
Trp W	0	2	0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	55	1	0
Tyr Y	1	1	2	1	3	1	1	1	3	2	2	1	2	15	1	2	2	3	31	2
Val V	7	4	4	4	4	4	4	4	5	4	15	10	4	10	5	5	5	72	4	17

q_{ij}

Matriz con probabilidades de mutación PAM250

Para conseguir esta matriz, se multiplica la matriz PAM1 por sí misma 250 veces

Matrices PAM

“Point Accepted Mutation”

- Las matrices PAM calculadas no se utilizan directamente en los algoritmos de alineamiento, requieren de una normalización previa para facilitar los cálculos
- Lod (log odd) score:** puntuación del logaritmo de la frecuencia de sustitución de un aminoácido por otro
 - Las propiedades del logaritmo permiten sumar los lod scores, en vez de multiplicarlos, durante el alineamiento, lo cual es computacionalmente menos costoso
 - $\log(m \cdot n) = \log(m) + \log(n)$
 - Los lod scores son simétricos, simplificando las matrices.
 - Los lod scores suelen simplificarse a enteros (raw scores), simplificando los cálculos

Matrices PAM

“Point Accepted Mutation”

$$S_{ij} = 10 \times \log_{10} \frac{q_{ij}}{p_i}$$

- ◆ S_{ij} – lod score: logaritmo de la frecuencia con la que el aminoácido i se convierte en el aminoácido j
- ◆ p_i – frecuencia normalizada de aparición del aminoácido i
 - ◆ Es una normalización de la mutabilidad relativa
- ◆ q_{ij} – probabilidad de sustitución de i por j en PAMn

Gly	0.089	Arg	0.041
Ala	0.087	Asn	0.040
Leu	0.085	Phe	0.040
Lys	0.081	Gln	0.038
Ser	0.070	Ile	0.037
Val	0.065	His	0.034
Thr	0.058	Cys	0.033
Pro	0.051	Tyr	0.030
Glu	0.050	Met	0.015
Asp	0.047	Trp	0.010

Matrices PAM

“Point Accepted Mutation”

- El lod score de la sustitución de Alanina (A) en Arginina (R) en PAM250 es:
 - $S_{A,R} = 10 \times \log_{10}(q_{AR}/p_A) = 10 \times \log_{10}(0.06/0.087) = -1.61 \sim -2$
- Significado: la posibilidad de que haya una sustitución de A por R en dos cadenas (en un alineamiento regido por el estudio evolutivo de Dayhoff en PAM250) es $10^{-0.161} \sim$ un 70% de la posibilidad de que se dé esa correspondencia aleatoriamente
 - ¿Y en $S_{A,A}$, que es igual a 2? $\rightarrow 2 = 10 \times \log_{10}(x) \rightarrow x = 10^{0.2} = 1.58$

	Ala	Arg	Asn	Asp	Cys	Gln	Gly	0.089	Arg	0.041
	A	R	N	D	C	Q	Ala	0.087	Asn	0.040
Ala A	13	6	9	9	5	8	Leu	0.085	Phe	0.040
Arg R	3	17	4	3	2	5	Lys	0.081	Gln	0.038
Asn N	4	4	6	7	2	5	Ser	0.070	Ile	0.037
							Val	0.065	His	0.034
							Thr	0.058	Cys	0.033
							Pro	0.051	Tyr	0.030
							Glu	0.050	Met	0.015
							58 Asp	0.047	Trp	0.010

Matrices PAM

“Point Accepted Mutation”

A	2																			
R	-2	6																		
N	0	0	2																	
D	0	-1	2	4																
C	-2	-4	-4	-5	12															
Q	0	1	1	2	-5	4														
E	0	-1	1	3	-5	2	4													
G	1	-3	0	1	-3	-1	0	5												
H	-1	2	2	1	-3	3	1	-2	6											
I	-1	-2	-2	-2	-2	-2	-2	-3	-2	5										
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	-2	6									
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5								
M	-1	0	-2	-3	-5	-1	-2	-3	-2	2	4	0	6							
F	-3	-4	-3	-6	-4	-5	-5	-5	-2	1	2	-5	0	9						
P	1	0	0	-1	-3	0	-1	0	0	-2	-3	-1	-2	-5	6					
S	1	0	1	0	0	-1	0	1	-1	-1	-3	0	-2	-3	1	2				
T	1	-1	0	0	-2	-1	0	0	-1	0	-2	0	-1	-3	0	1	3			
W	-6	2	-4	-7	-8	-5	-7	-7	-3	-5	-2	-3	-4	0	-6	-2	-5	17		
Y	-3	-4	-2	-4	0	-4	-4	-5	0	-1	-1	-4	-2	7	-5	-3	-3	0	10	
V	0	-2	-2	-2	-2	-2	-2	-1	-2	4	2	-2	2	-1	-1	-1	0	-6	-2	4
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V

lod score PAM250

- Es una matriz simétrica
 - Debido a la normalización y redondeo al entero más cercano
 - Simplifica los cálculos de los algoritmos de alineamiento
- Se dan lod scores simplificados al entero más cercano (raw scores)
 - Simplifican los cálculos de los algoritmos de alineamiento
 - Puede introducir pequeños errores (asumibles)

Matrices PAM

“Point Accepted Mutation”

- El uso de una matriz PAM u otra depende de la similitud entre las secuencias
 - Para secuencias muy parecidas PAM10 puede dar buenos alineamientos
 - Para secuencias muy distintas PAM250 puede ser mejor
- A priori no siempre sabemos la similitud de las secuencias
 - Puede ser necesario repetir los alineamientos con varias matrices y quedarse con la que dé el mejor alineamiento

Ejemplo: Sistemas de scoring PAM250

[illegible]

El “score” de las intersecciones de la matriz reflejan qué tan seguido un aa se apareó con otro en un alineamiento de secuencias.

Matrices BLOSUM

“Blocks Amino Acid Substitution Matrices”

Las Matrices BLOSUM BLOck SUBstitution Matrix

- Estan basadas en comparaciones de Bloques de secuencias derivadas de una base de datos denominada Blocks.
- La Base de datos Blocks contiene segmentos sin gaps en alineacion multiple. Estos segmentos corresponden a las regiones más conservadas de las proteínas. (Alineamiento local versus Alieneamiento Global)
- Las matrices BLOSUM se derivan de bloques cuyo alineamiento corresponde al numero de la matriz BLOSUM (p.ej. BLOSUM 62 se deriva de Blocks que contienen secuencias mas de 62% identicas en un alineamiento sin gaps)
- BLOSUM 62 es la matriz por defecto para el programa BLAST de proteínas estándar

Matrices BLOSUM

“Blocks Amino Acid Substitution Matrices”

- Henikoff, 1991 – Sin modelo evolutivo, diseñadas para hallar dominios conservados. Función bioquímica similar.

- Representan más de 500 familias de proteínas (sin restricción).
- Se agrupan los blocks de acuerdo a su identidad y se generan matrices. Los scores provienen de la observación de los tipos y frecuencias de sustitución en distintas familias protéicas.

Conjuntos 80% idénticos -> BLOSUM80

Conjuntos 60% idénticos -> BLOSUM60

Etc (a mayor BLOSUM, menor distancia)

BLOSUM80

```
graph TD; A[BLOSUM80] --> B[Umbral de identidad utilizado para seleccionar los bloques de proteínas];
```

A diagram illustrating the BLOSUM80 matrix. The text "BLOSUM80" is shown in a dark blue serif font. A purple circle highlights the "80" portion of the text. A vertical purple arrow points from the bottom of this circle down to a purple rounded rectangular box. Inside the box, white text explains that the value represents an identity threshold used for selecting protein blocks.

Umbral de identidad
utilizado para seleccionar
los bloques de proteínas

Ejemplo: Sistemas de scoring **BLOSUM62**

Las identidades tienen scores positivos, pero algunas son más valoradas que otras.

scores positivos, pero algunas son más valoradas que otras.



A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V

**Algunas sustituciones
tienen scores positivos,
pero la mayoría son
negativos.**

tienen scores positivos,
pero la mayoría son
negativos.



A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4
A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	

Matrices BLOSUM

“Blocks Amino Acid Substitution Matrices”

A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	-1	2	0	-1	-1	1	1	-2	-1	-3	-2	5								
M	-1	-2	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V

Según Henikoff & Henikoff (1992)
BLOSUM62 es mejor alternativa que
PAM y otros BLOSUM para un
alineamiento genérico.
Es utilizada como matriz por defecto
por la mayoría de los programas de
alineamiento y de búsquedas en
bases de datos

Matrices BLOSUM

“Blocks of Amino Acid Substitution Matrices”

- ◆ PAM: basada en familias de proteínas muy relacionadas
 - ◆ Útil para alinear secuencias parecidas
- ◆ BLOSUM: basada en observaciones de alineamientos entre secuencias muy distintas
 - ◆ Útil para alinear secuencias distintas

BLOSUM90

BLOSUM62

BLOSUM45

PAM30

PAM120

PAM250

Less divergent



More divergent

Human versus
chimpanzee beta globin

65

Human versus
bacterial globins