CHAPTER 5: PAIRWISE SEQUENCE ALIGNMENT AND DATABASE SEARCHING

Dr. Garrett Dancik

Alignment algorithms (preview)

- Needleman-Wunsch (1970) and variations:
  - for aligning two sequences
  - uses dynamic programming to "consider" all possible alignments (10^{600} for two sequences of length 1000!)
- FASTA: uses a heuristic method for efficient searches (though not guaranteed to find the optimal solution)
  - Creates dictionary of k-tuples for the query sequence which is checked against sequences in the database
  - A local alignment algorithm is used to complete the alignment
- BLAST (Basic Local Alignment Search Tool): also fast and uses a heuristic
  - Finds short matches (which do not have to be perfect)
  - Then uses local alignment to complete the alignment

Needleman and Wunsch Dynamic Programming method

- Dynamic programming method
  - The problem can be divided into smaller parts
- Consider the following alignment of the sequences

x = THISLINE and y = ISALIGNED

```
T  H  I  S  -  L  I  -  N  E  -
-  -  I  S  A  L  I  G  N  E  D
```

Dynamic Programming Concept

The alignment below is the optimal global alignment, with BLOSUM-62 matrix, and linear gap penalty of 4:

A key observation is that the total alignment score is the sum of partial alignment scores:

```
Score:    -4  -4  +4  +4 | -4  +4  +4  -4 | +6  +5  -4

           T   H   I   S  |  -   L   I   -  |  N   E   -
           -   -   I   S  |  A   L   I   G  |  N   E   D

Score:            0       |        0        |       7
```

For any part of the alignment: The score of the optimal global alignment from positions 1➔ L is the sum of the partial alignment scores from positions 1➔ u, $u+1$ ➔ v, and $v+1$ ➔ L

Dynamic programming, key idea

The optimal alignment between sequences **x** and **y** can end in 3 possible ways (example for alignment between THIS and IS is shown):

1. A pair between the last character of **x** and last character of **y**

$$\boxed{\text{Optimal alignment of THI with I}} + \begin{array}{c} S \\ S \end{array}$$

2. A gap at the end of **x** aligned to the last character of **y**

$$\boxed{\text{Optimal alignment of THIS with I}} + \begin{array}{c} - \\ S \end{array}$$

3. A gap at the end of **y** aligned to the last character of **x**

$$\boxed{\text{Optimal alignment of THI with IS}} + \begin{array}{c} S \\ - \end{array}$$

Optimal alignment between THIS and IS has 3 possibilities:

Optimal alignment of THI with I

```
THI   S
--I + S
```

This possibility has the highest score, so it is the optimal alignment between THIS and IS

-4 + 4

0

Optimal alignment of THIS with I

```
THIS   -
--I- + S
```

-8 + -4

-12

Optimal alignment of THI with IS

```
THI   S
IS- + -
```

-6 + -4

-10

Dynamic Programming Algorithm

Let $S_{i,j}$ be the score of the optimal alignment of all characters up to x_i of sequence **x** and y_j of sequence **y**.

Then there are three possibilities for the end of this alignment

$$\cdots \; x_{i-1} \; x_i$$
$$\cdots \; y_{j-1} \; y_j$$
$$S_{i-1,j-1}$$

$$\cdots \; x_i \; -$$
$$\cdots \; y_{j-1} \; y_j$$
$$S_{i,j-1}$$

$$\cdots \; x_{i-1} \; x_i$$
$$\cdots \; y_j \; -$$
$$S_{i-1,j}$$

The optimal alignment score $S_{i,j}$ is the maximum of the following

$$S_{i-1,\,j-1} + s(x_i, y_j) \qquad S_{i,\,j-1} + g \qquad S_{i-1,\,j} + g$$

where $s(x_i, y_j)$ is the substitution (or match/mismatch score) for character x_i aligned with y_j, and g is a constant gap penalty

Example:

x = THISLINE and y = ISALIGNED

```
T  H  I  S  -  L  I  -  N  E  -
-  -  I  S  A  L  I  G  N  E  D
```

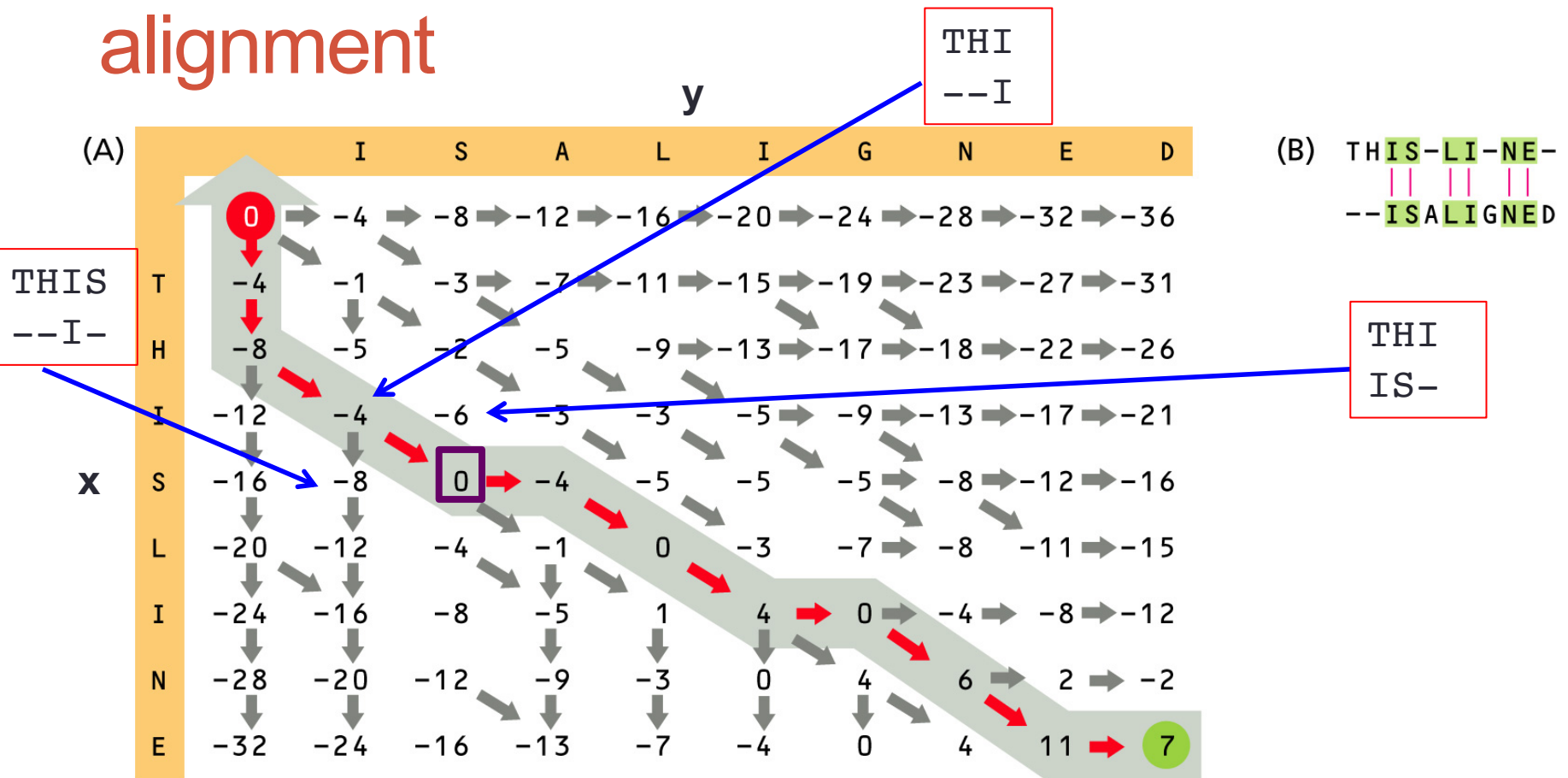Assume a constant (linear) gap penalty $g = 4$, and $s(x_i, y_j)$ calculated using the BLOSUM-62 scoring matrix.

$S_{4,2}$ is the optimal score for aligning THIS with IS and is the maximum of

```
THI  + S          THIS  + -          THI  + S
--I    S          --I-    S          IS-    -

-4   + 4          -8   + -4          -6   + -4
     0                 -12                -10
```
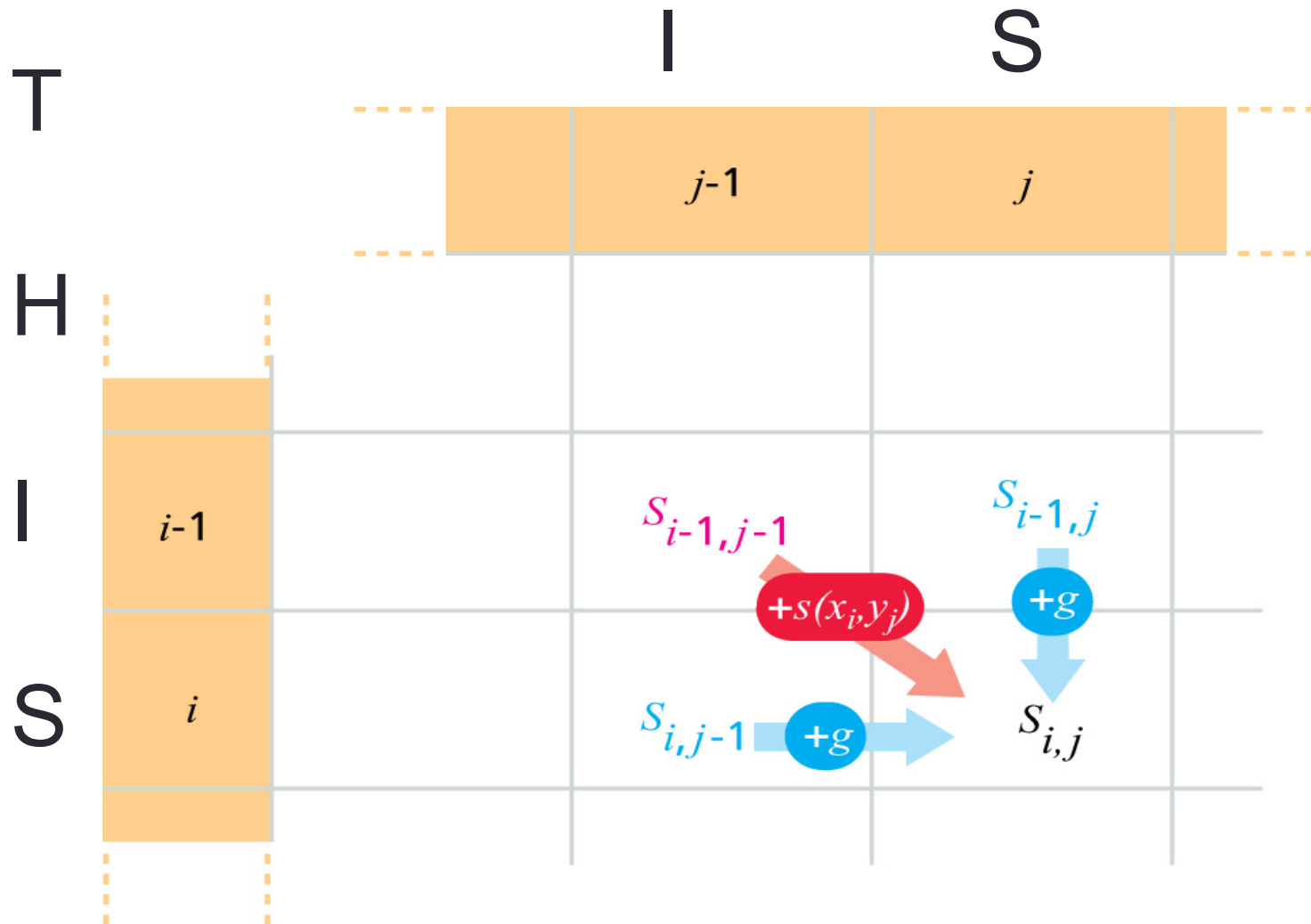
Dynamic programming matrix for global alignment

$S_{i,j}$ is the score of the optimal alignment of all characters up to x_i of sequence **x** and y_j of sequence **y**.

The score of the optimal alignment of THIS with IS is $S_{4,2} = 0$.

Dynamic programming matrix illustrating possible optimal alignments for subsequences $x_1, \ldots x_i$ and $y_1, \ldots y_j$

Algorithm for finding the optimal global alignment

- For two sequences, $x_1, \ldots x_m$ and $y_1, \ldots y_n$
- Construct an (m+1) by (n+1) matrix (starting with element $S_{0,0}$ at the top left)
- For each i, the element $S_{i,0}$ corresponds to the alignment

$$x_1 \quad x_2 \quad \ldots \quad x_i$$
$$- \quad - \quad \ldots \quad -$$

For each j, the element $S_{0,j}$ corresponds to the alignment

$$- \quad - \quad \ldots \quad -$$
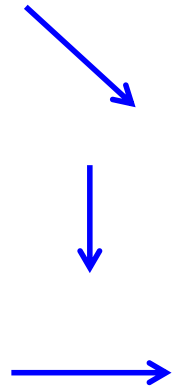$$y_1 \quad y_2 \quad \ldots \quad y_i$$

Fill in the matrix for all $S_{i,0}$ and $S_{0,j}$ appropriately

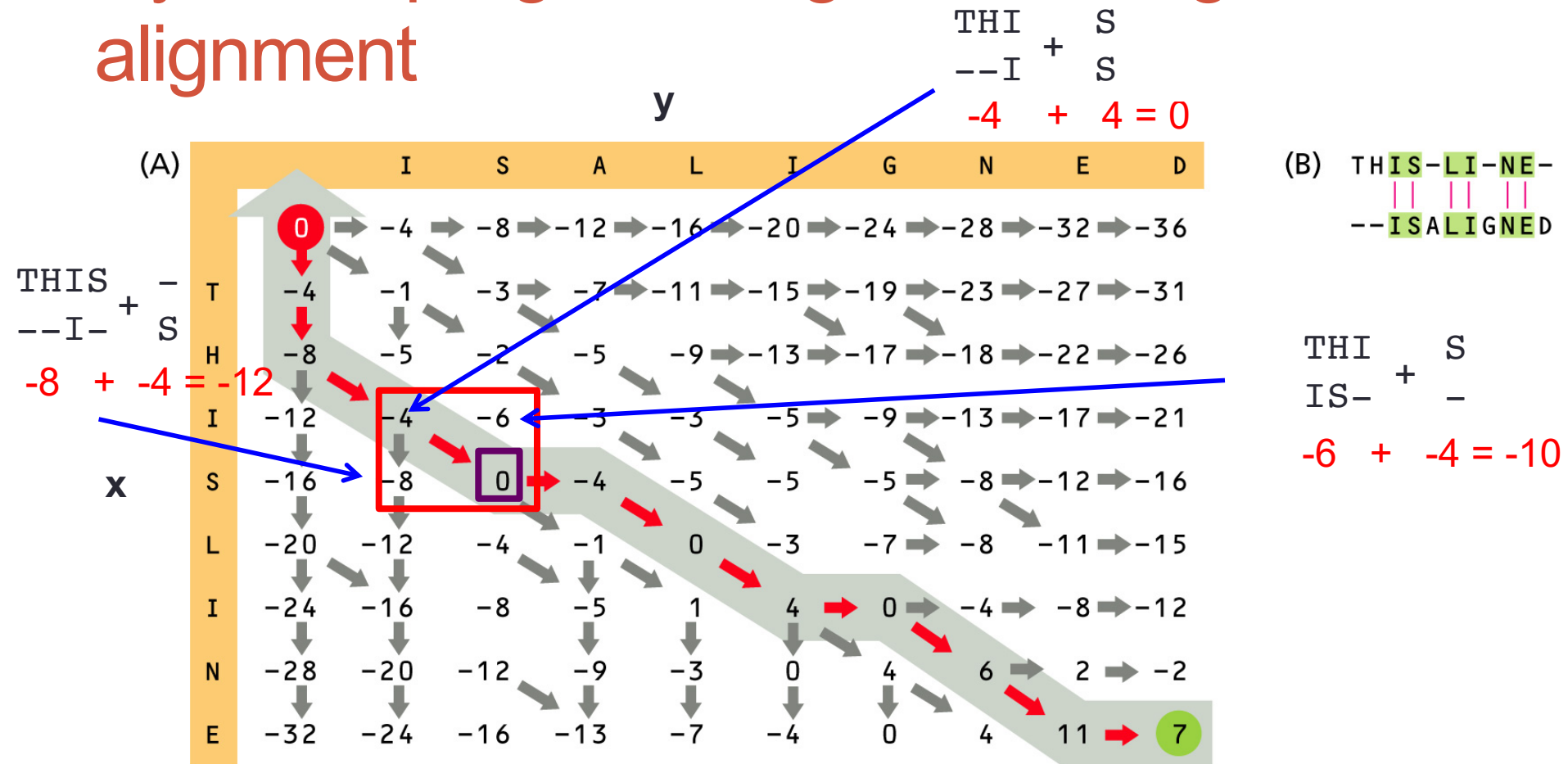# Assignment of $S_{i,0}$ and $S_{0,j}$ when a linear gap penalty of 4 is used
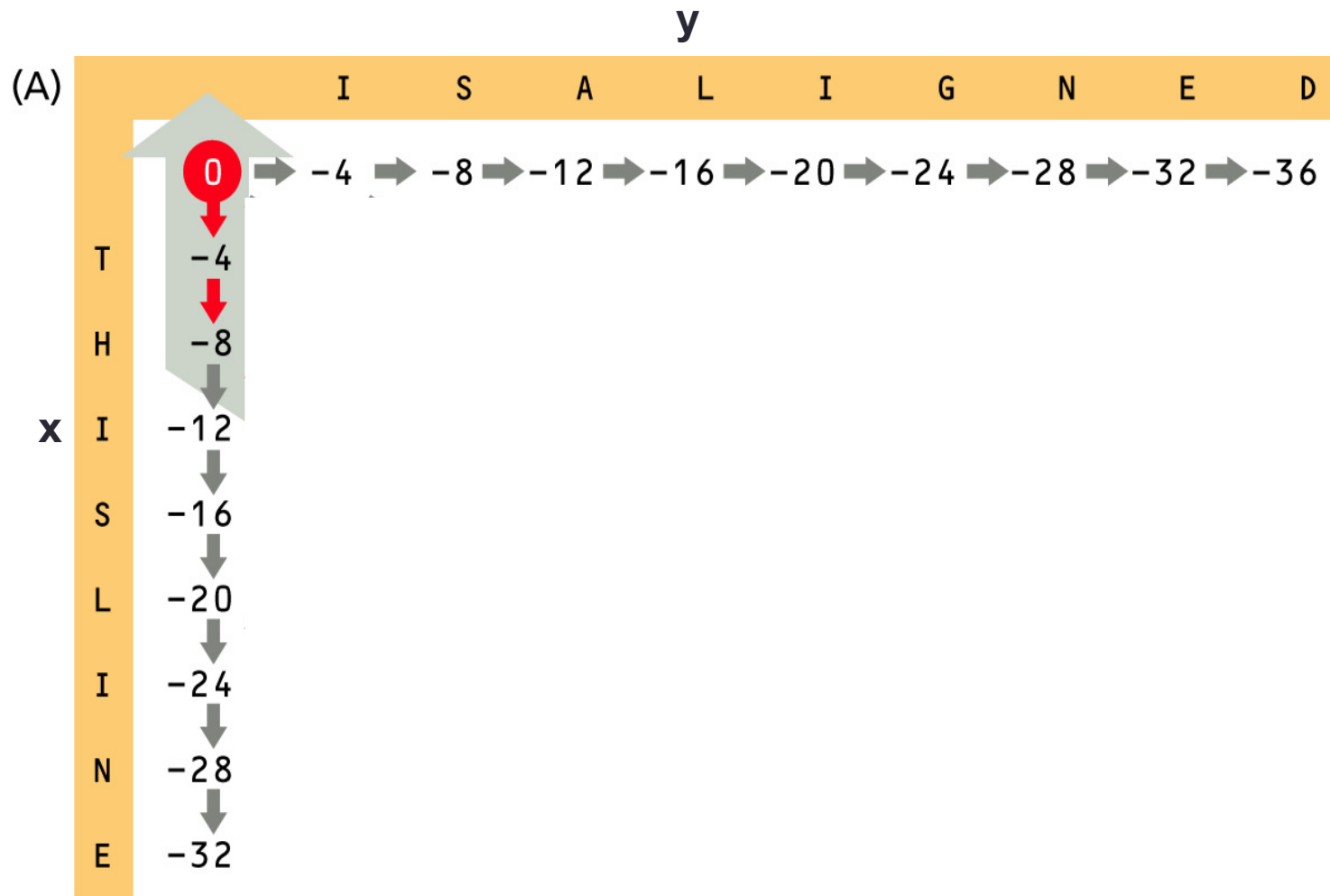
Algorithm for finding the optimal global alignment (con't)

- For $i = 1, \ldots, m$ and $j = 1, \ldots, n$

$$\text{set } S_{i,j} = \max \begin{cases} S_{i-1,j-1} + s(x_i, \; x_j) \\ \\ S_{i-1,j} + g \\ \\ S_{i,j-1} + g \end{cases}$$

and draw an arrow to $S_{i,j}$ from the cell where the score was derived ($S_{i-1,j-1}$, $S_{i-1,j}$, or $S_{i,j-1}$)

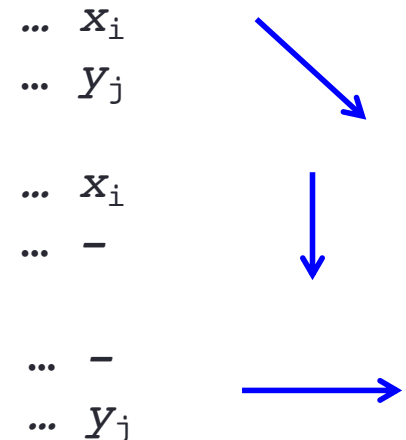# Dynamic programming matrix for global alignment

$S_{i,j}$ is the score of the optimal alignment of all characters up to x_i of sequence **x** and y_j of sequence **y**.

The score of the optimal alignment of THIS with IS is $S_{4,2} = 0$.

Algorithm for finding the optimal global alignment (con't)

- After the dynamic programming matrix is filled in, $S_{m,n}$ contains the score of the optimal global alignment
- A process called ***traceback*** is then used to recover the alignment itself, based on the path used to generate the optimal alignment score (based on the arrows), starting with $S_{m,n}$ and working backwards to $S_{0,0}$
- If the score $S_{i,j}$ was derived

  - diagonally from $S_{i-1,j-1}$, then align x_i with y_j

    ... x_i
    ... y_j

  - vertically from $S_{i-1,j}$, then align a gap with y_j

    ... x_i
    ... $-$

  - horizontally from $S_{i,j-1}$, then align x_i with a gap

    ... $-$
    ... y_j

Global alignment: Dynamic programming matrix using linear gap penalty of 4

Global alignment: Dynamic programming matrix using linear gap penalty of 4

Global alignment: Dynamic Programming Matrix using linear gap penalty of 8

Modification of algorithm for semiglobal alignments

- A **semiglobal alignment** does not penalize for gaps at the beginning and end of an alignment

- Appropriate for global alignments of sequences that are not the same length

- Modifications to algorithm for semiglobal alignment:

  - Set $S_{i,0}$ and $S_{0,j}$ to 0 for all i and j

  - Start the traceback from the highest scoring element in the bottom row or last column

Semiglobal alignment: Dynamic Programming Matrix using a linear gap penalty of 8

(A)
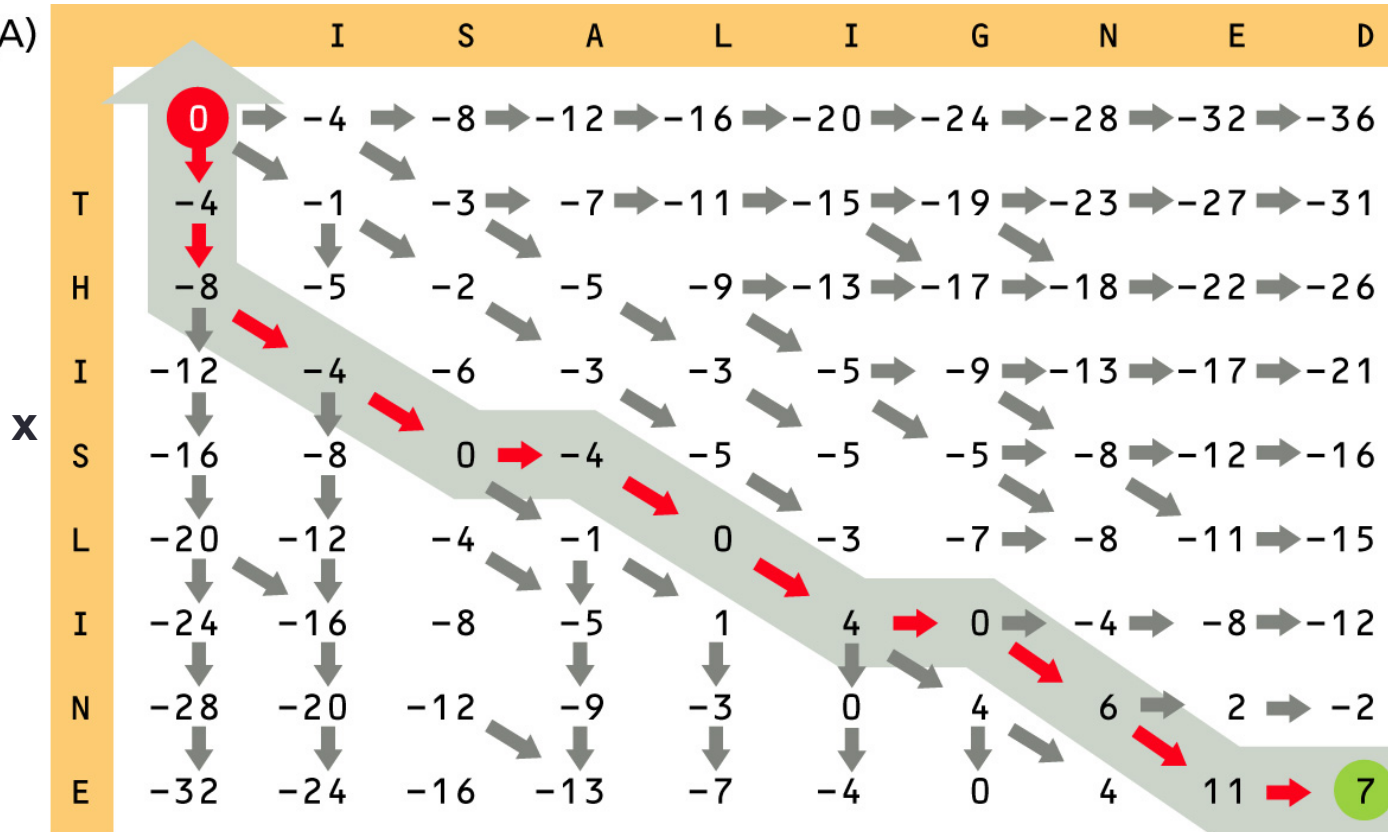
		I	S	A	L	I	G	N	E	D
	0	0	0	0	0	0	0	0	0	0
T	0	-1	1	0	-1	-1	-2	0	-1	-1
H	0	-3	-2	-1	-3	-4	-3	-1	0	-2
I	0	-4	-5	-3	1	1	-7	-6	-4	-3
S	0	-2	8	0	-5	-1	1	-6	-6	-4
L	0	2	0	7	4	-3	-5	-2	-9	-10
I	0	4	0	-1	9	8	0	-8	-5	-12
N	0	-3	5	-2	1	6	8	6	-2	-4
E	0	-3	-3	4	-4	-2	4	8	11	3
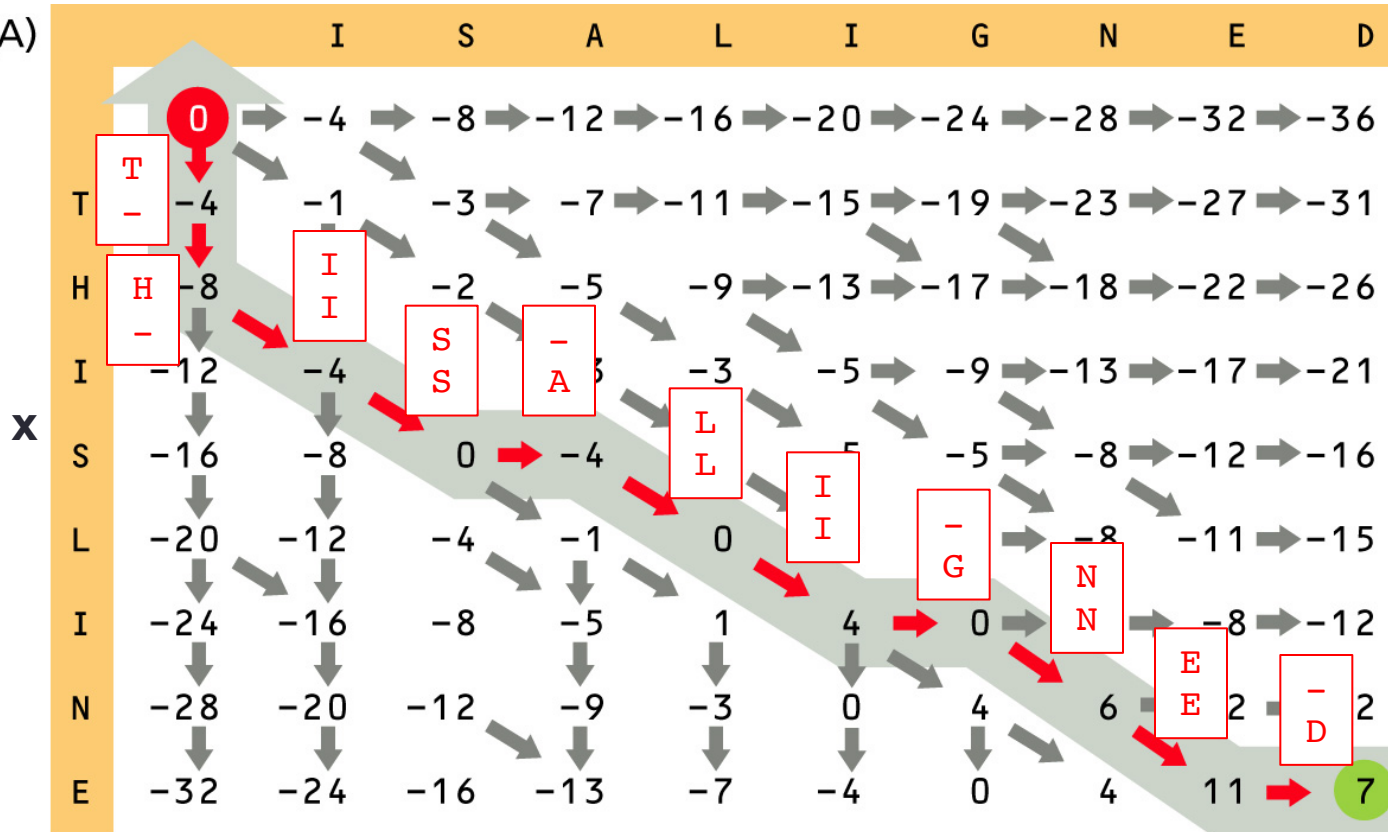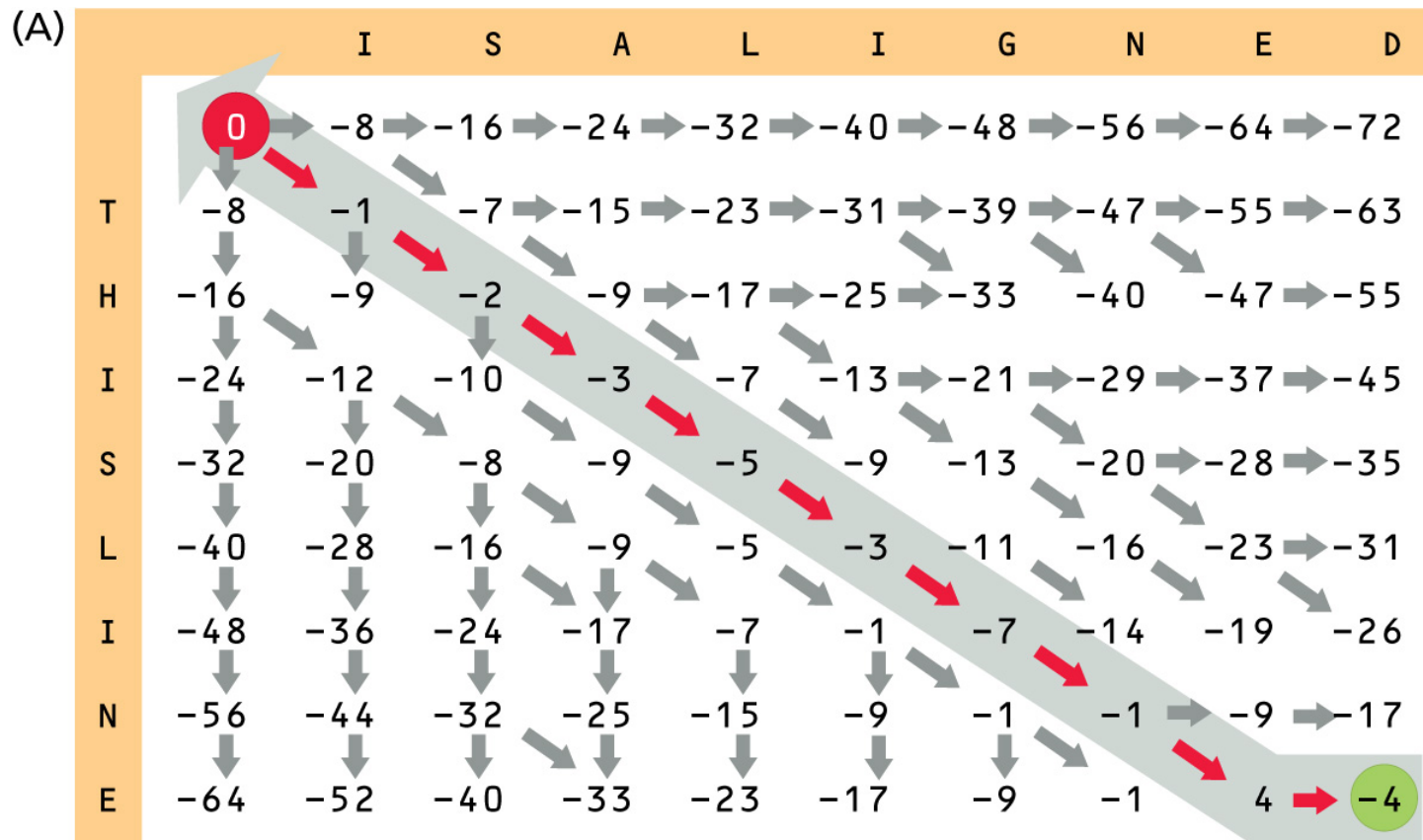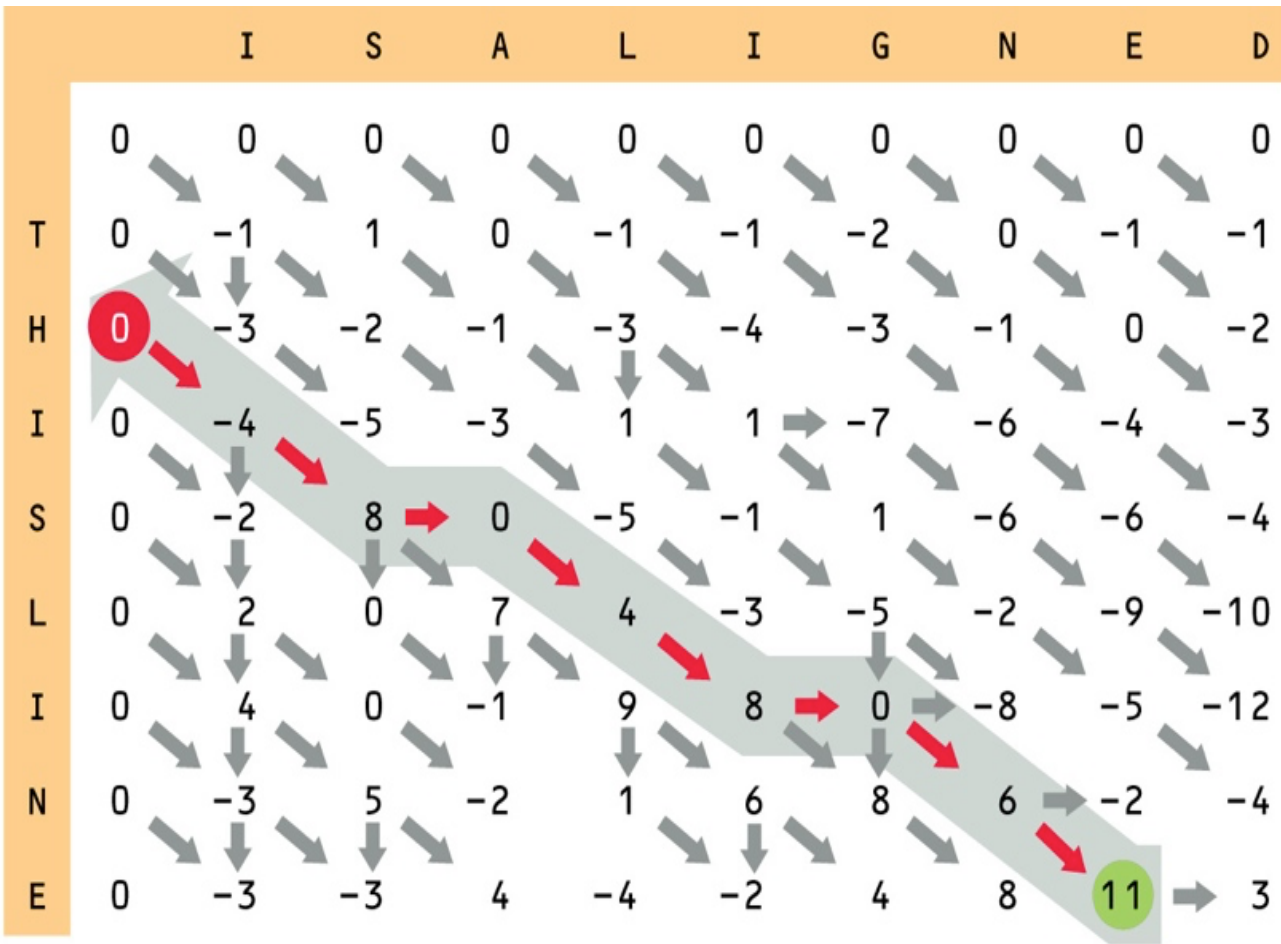
(B)
```
THIS-LI-NE-
  ||  ||  ||
--ISALIGNED
```

Modification of algorithm for local alignments (1st proposed by Smith-Watterman)

- A **local alignment** aligns *regions* of two sequences, and will not necessarily span the length of each sequence
- Appropriate for identifying functional domains of a protein
- Modifications to algorithm for local alignment:

  - set $S_{i,j}$ = max
  $$
  \begin{cases}
  S_{i-1,j-1} + s(x_i,\ x_j) \\
  S_{i-1,j} + g \\
  S_{i,j-1} + g \\
  0
  \end{cases}
  $$

- Assumes that the expected alignment score is negative for random sequences and is positive for similar sequences
- Traceback starts from highest scoring matrix element anywhere in the matrix and ends at 0

Local alignment: Dynamic Programming Matrix using a linear gap penalty of 8

(A)

	I	S	A	L	I	G	N	E	D
	0	0	0	0	0	0	0	0	0
T	0	0	1	0	0	0	0	0	0
H	0	0	0	0	0	0	0	1	0	0
I	0	4	0	0	2	4	0	0	0	0
S	0	0	8	1	0	0	4	1	0	0
L	0	2	0	7	5	2	0	1	0	0
I	0	4	0	0	9	9	1	0	0	0
N	0	0	5	0	1	6	9	7	0	1
E	0	0	0	4	0	0	4	9	12	4

(B)

SLI-NE
 || ||
ALIGNE