Alignment methods

- Introduction to global and local sequence alignment methods
  - Global : Needleman-Wunch
  - Local : Smith-Waterman

Why search sequence databases?

- 1. I have just sequenced something. What is known about the thing I sequenced?
- 2. I have a unique sequence. Is there similarity to another gene that has a known function?
- 3. I found a new protein in a lower organism. Is it similar to a protein from another species?
- 4. I have decided to work on a new gene. The people in the field will not give me the plasmid. I need the complete cDNA sequence to perform RT-PCR.

Perfect Searches

- First “hit” should be an exact match.
- Next “hits” should contain all of the genes that are related to your gene (homologs)
- Next “hits” should be similar but are not homologs

How does one achieve the "perfect search"?

- Comparison Matrices (PAM vs. BLOSUM)
- Database Search Algorithms
- Databases
- Search Parameters
  - Expect Value-change threshold for score reporting
  - Translation-of DNA sequence into protein
  - Filtering-remove repeat sequences

Alignment Methods

- Learning objectives-Understand the principles behind the **Needleman-Wunsch** method of alignment. Understand how software operates to optimally align two sequences

Needleman-Wunsch Method (1970)

Output:
An alignment of two sequences is represented by three lines
The first line shows the first sequence
The third line shows the second sequence.
The second line has a row of symbols.
The symbol is a vertical bar wherever characters in
the two sequences match, and a space where ever they do not.
Dots may be inserted in either sequence to represent gaps.

Needleman-Wunsch Method (cont. 1)

For example, the two hypothetical sequences

```
 abcdefghajklm
    abbdhijk

could be aligned like this
    abcdefghajklm
    || |    | ||
    abbd...hijk
As shown, there are 6 matches,
2 mismatches, and one gap of length 3.
```

Needleman-Wunsch Method (cont. 2)

The alignment is scored according to a payoff matrix

```
$payoff = { match       => $match,
            mismatch    => $mismatch,
            gap_open    => $gap_open,
            gap_extend  => $gap_extend };
```

For correct operation, match must be positive,
and the other entries must be negative.

Needleman-Wunsch Method (cont. 3)

Example

Given the payoff matrix

```
$payoff = { match       =>  4,
            mismatch    => -3,
            gap_open    => -2,
            gap_extend  => -1 };
```

Needleman-Wunsch Method (cont. 4)

The sequences

```
    abcdefghajklm
    abbdhijk
are aligned and scored like this
                  a b c d e f g h a j k l m
                  | |   |       |   | |
                  a b b d . . . h i j k
    match         4 4   4       4   4 4
    mismatch         -3            -3
    gap_open             -2
    gap_extend           -1-1-1
```

`for a total score of 24-6-2-3 = 13.`

Needleman-Wunsch Method (cont. 5)

The algorithm guarantees that no other alignment of these two sequences has a higher score under this payoff matrix.

Needleman-Wunsch Method (cont. 6) Dynamic Programming

Potential difficulty. How does one come up with the optimal alignment in the first place? We now introduce the concept of dynamic programming (DP).

DP can be applied to a large search space that can be structured into a succession of stages such that:

1) the initial stage contains trivial solutions to sub-problems

2) each partial solution in a later stage can be calculated by recurring on only a fixed number of partial solutions in an earlier stage.

3) the final stage contains the overall solution.

Three steps in Dynamic Programming

1. Initialization

2 Matrix fill or scoring

3. Traceback and alignment

Two sequences will be aligned.

GAATTCAGTTA (sequence #1)
GGATCGA (sequence #2)

A simple scoring scheme will be used

$S_{i,j}$ = 1 if the residue at position I of sequence #1 is the same as the residue at position j of the sequence #2 (called match score)

$S_{i,j}$ = 0 for mismatch score

w = gap penalty

Initialization step: Create Matrix with M + 1 columns and N + 1 rows. First row and column filled with 0.

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

Matrix fill step: Each position $M_{i,j}$ is defined to be the MAXIMUM score at position i,j

$M_{i,j}$ = MAXIMUM **[**

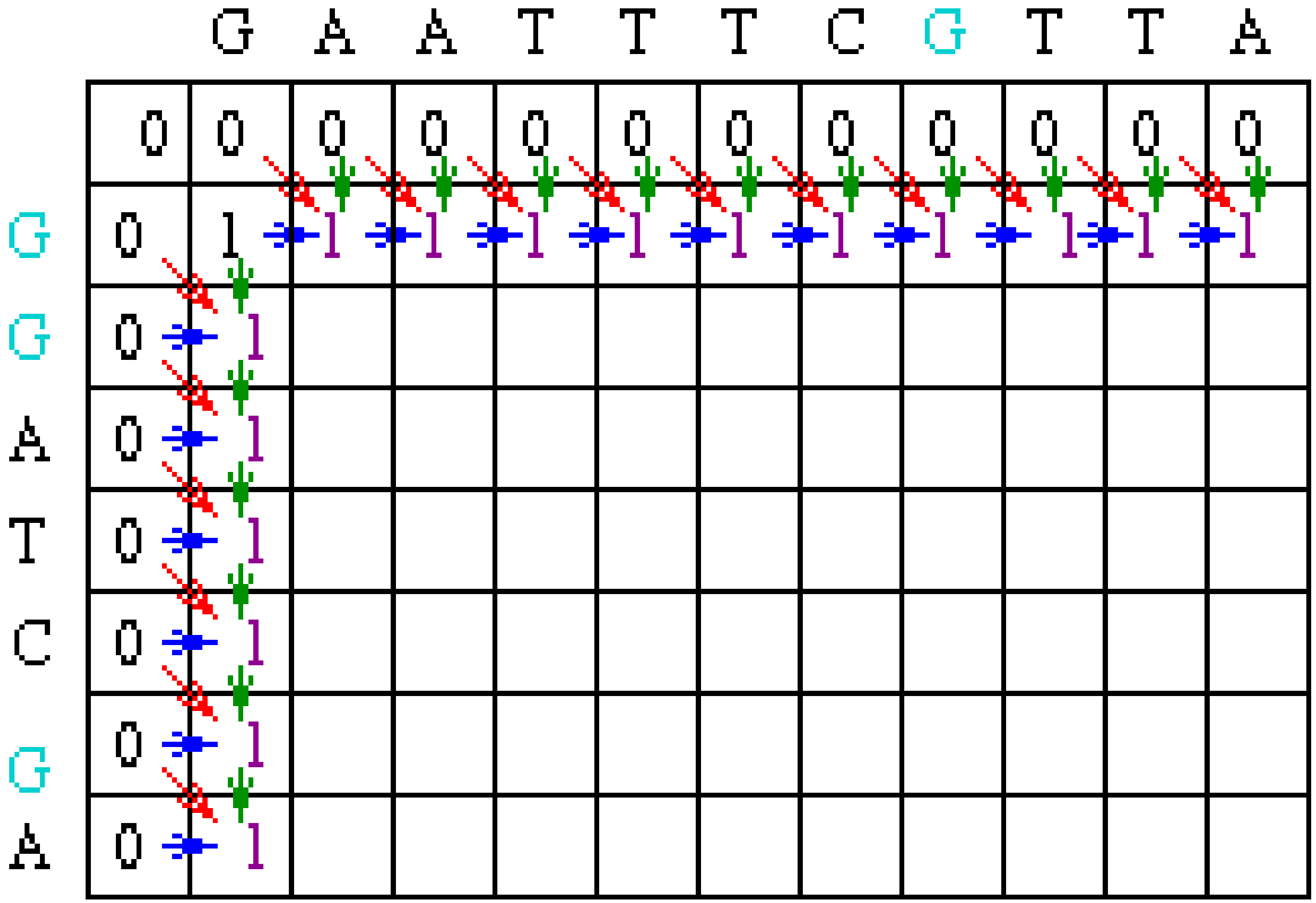
$M_{i-1,\ j-1}$ + $s_{i,,j}$ (match or mismatch in the diagonal)

$M_{i,\ j-1}$ + w (gap in sequence #1)

$M_{i-1,\ j}$ + w (gap in sequence #2)**]**

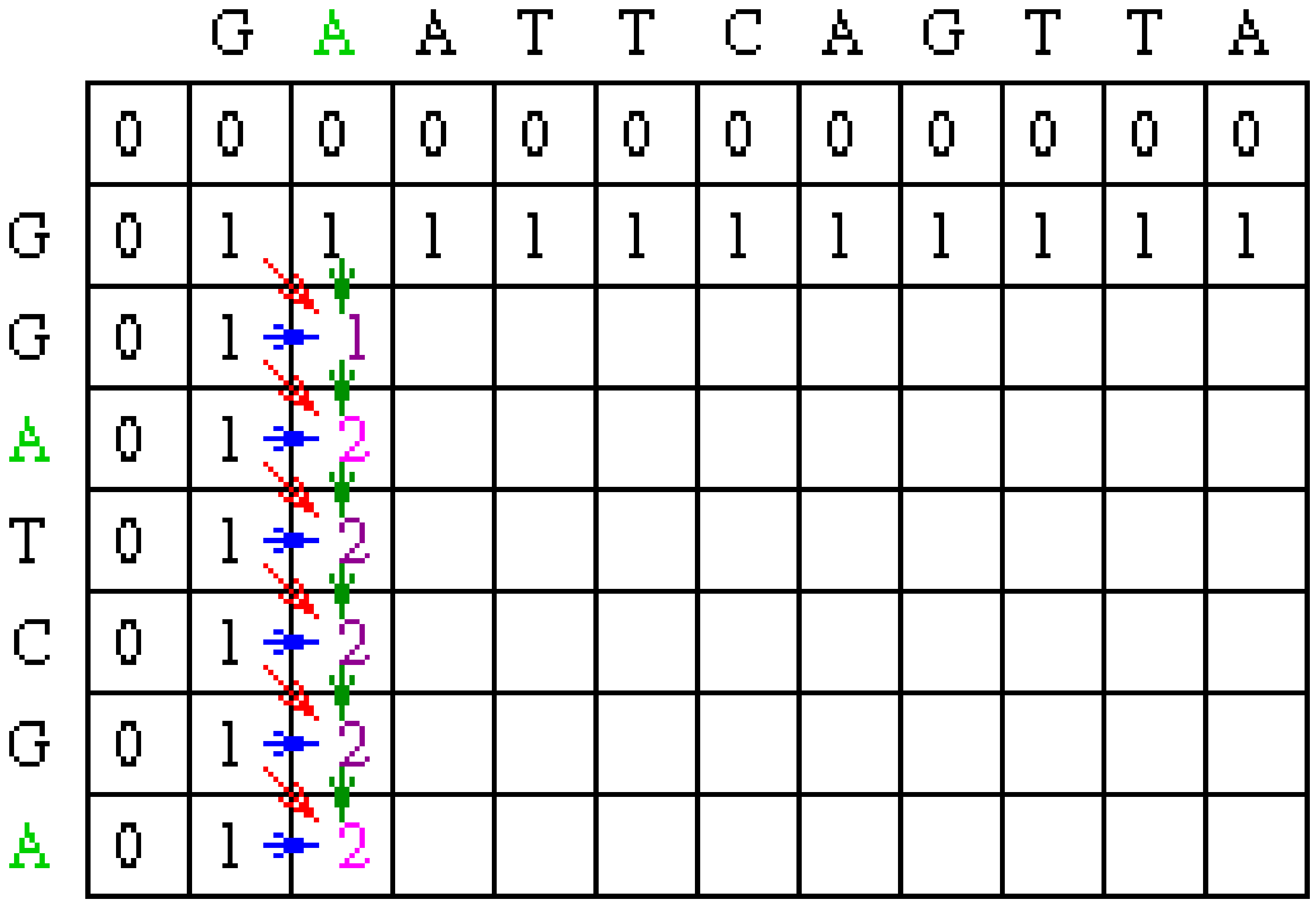
		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1										
G	0											
A	0											
T	0											
C	0											
G	0											
A	0											

Fill in rest of row 1 and column 1



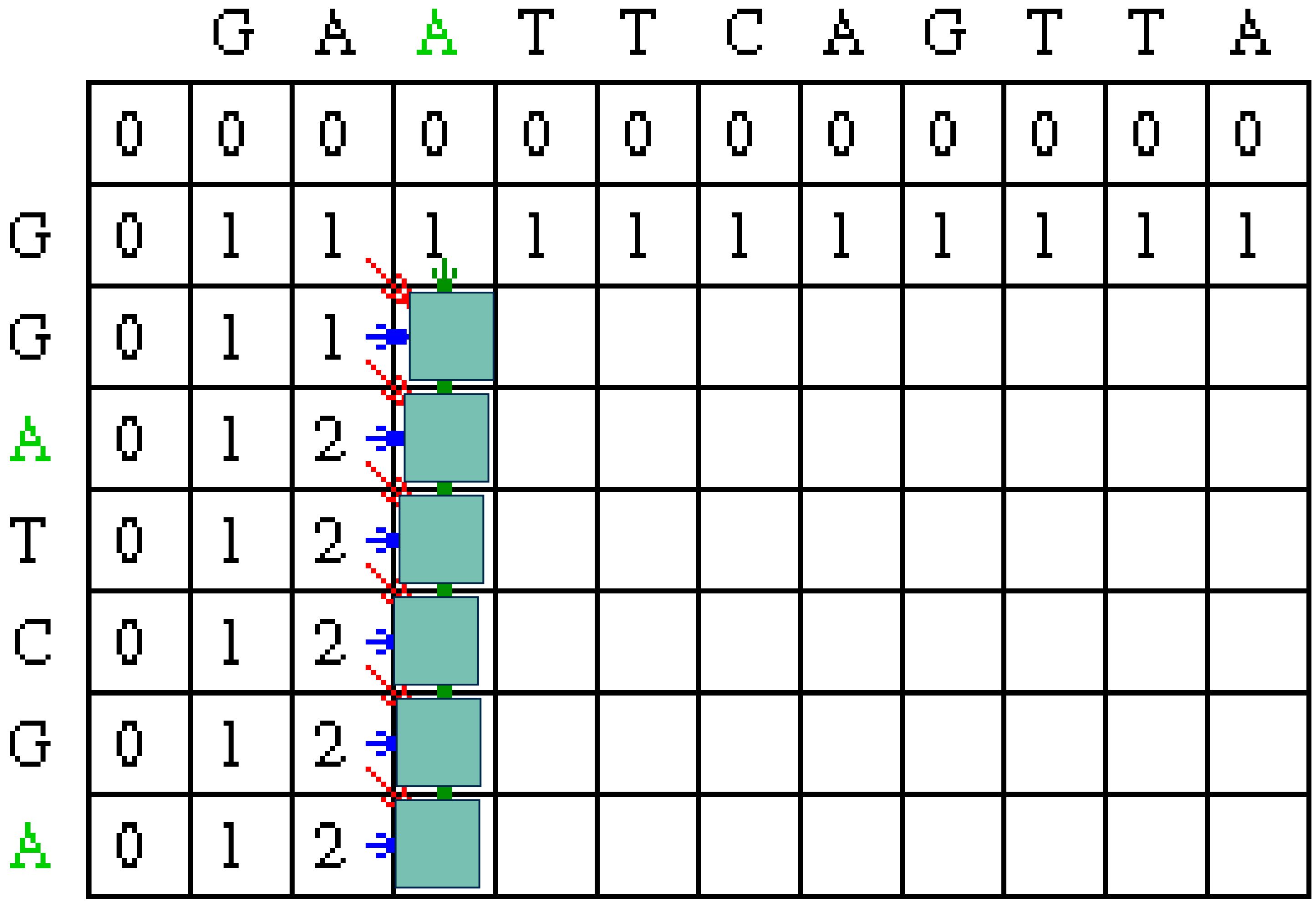
		G	A	A	T	T	T	C	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1										
A	0	1										
T	0	1										
C	0	1										
G	0	1										
A	0	1										

Fill in column 2



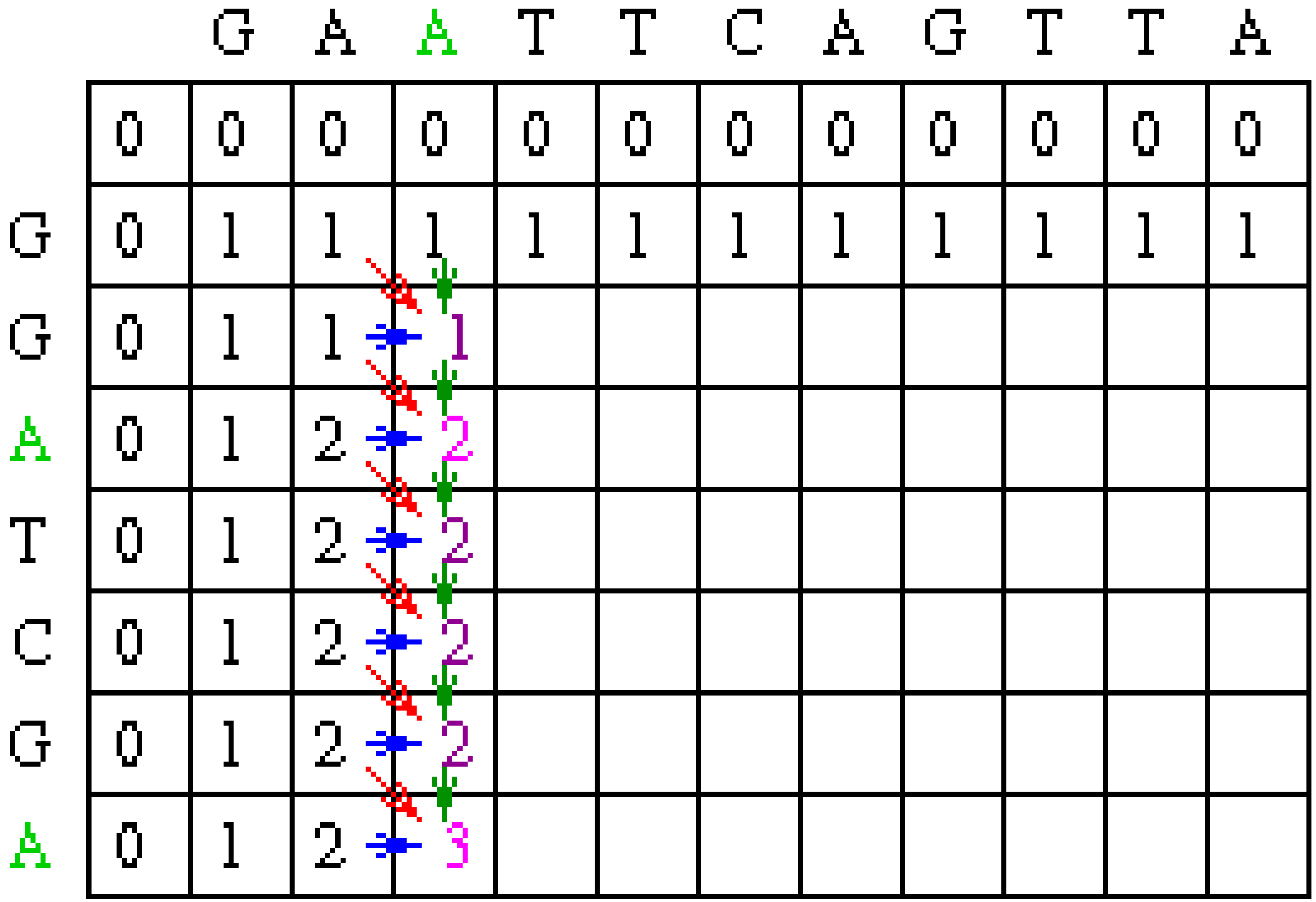
		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1									
A	0	1	2									
T	0	1	2									
C	0	1	2									
G	0	1	2									
A	0	1	2									

Fill in column 3



		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	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1									
A	0	1	2									
T	0	1	2									
C	0	1	2									
G	0	1	2									
A	0	1	2									

Column 3 with answers



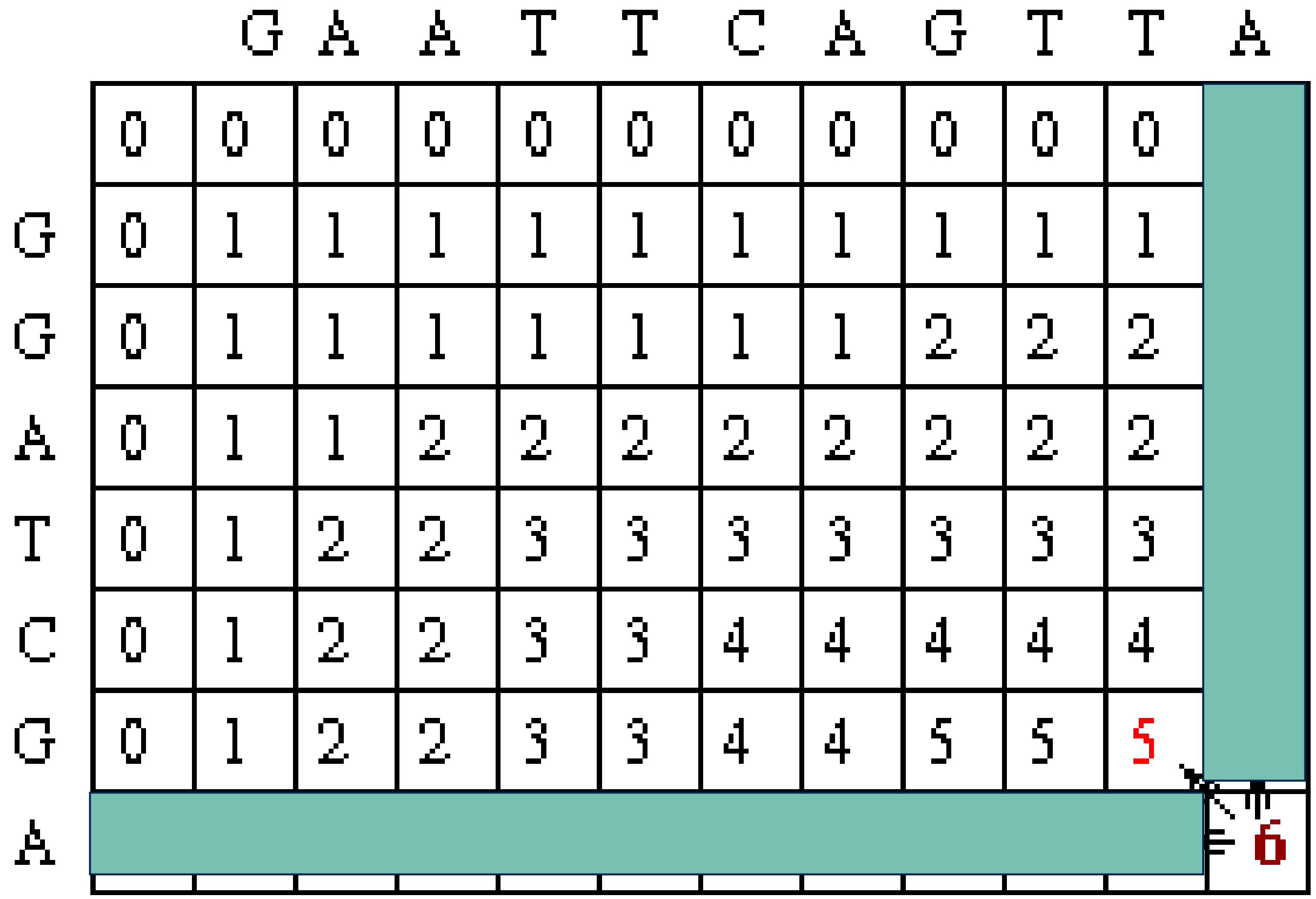
		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1	1								
A	0	1	2	2								
T	0	1	2	2								
C	0	1	2	2								
G	0	1	2	2								
A	0	1	2	3								

Fill in rest of matrix with answers

		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	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1	1	1	1	1	1	2	2	2	2
A	0	1	2	2	2	2	2	2	2	2	2	3
T	0	1	2	2	3	3	3	3	3	3	3	3
C	0	1	2	2	3	3	3	4	4	4	4	4
G	0	1	2	2	3	3	3	4	4	5	5	5
A	0	1	2	3	3	3	3	4	5	5	5	6

Traceback step:

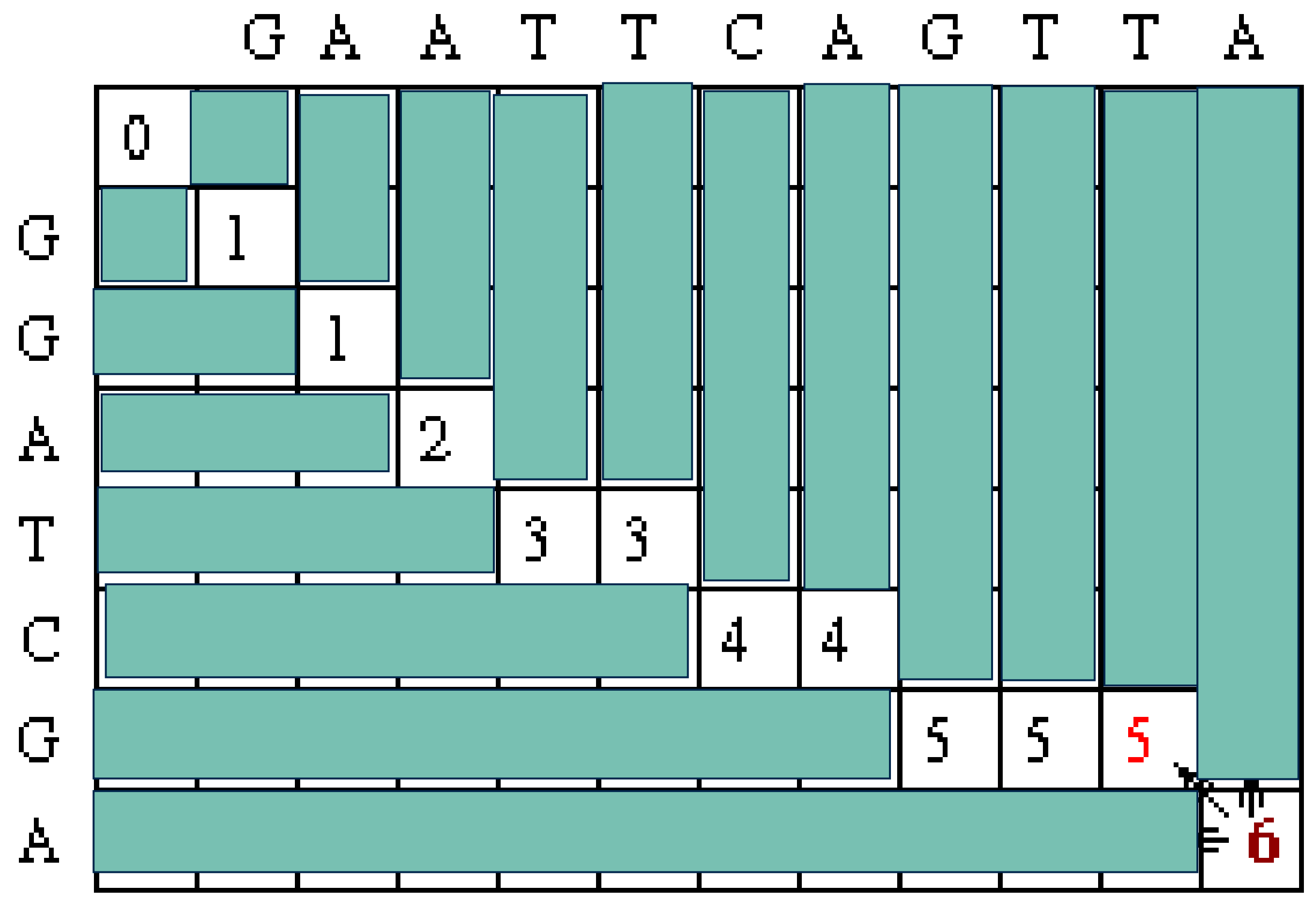
Position at current cell and look at direct predecessors



		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	
G	0	1	1	1	1	1	1	1	1	1	1	
G	0	1	1	1	1	1	1	1	2	2	2	
A	0	1	1	2	2	2	2	2	2	2	2	
T	0	1	2	2	3	3	3	3	3	3	3	
C	0	1	2	2	3	3	4	4	4	4	4	
G	0	1	2	2	3	3	4	4	5	5	5	
A												6

```
Seq#1  A
       |
Seq#2  A
```

Traceback step:
Position at current cell and look at direct predecessors



```
Seq#1   G A A T T C A G T T A
        |   | |   |   |     |
Seq#2   G G A T - C - G - - A
```

Needleman-Wunsch Method Dynamic Programming

The problem with Needleman-Wunsch is the amount of processor memory resources it requires. Because of this it is not favored for practical use, despite the guarantee of an optimal alignment. The other difficulty is that the concept of global alignment is not used in pairwise sequence comparison searches.

Needleman-Wunsch Method Typical output file

```
Global: HBA_HUMAN vs HBB_HUMAN
Score: 290.50

HBA_HUMAN       1         VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFP 44
                          |:| :|: | | ||||  :  | | ||| |: : :| |: :|
HBB_HUMAN       1        VHLTPEEKSAVTALWGKV..NVDEVGGEALGRLLVVYPWTQRFFE 43

HBA_HUMAN       45       HF.DLS.....HGSAQVKGHGKKVADALTNAVAHVDDMPNALSAL 83
                          | |||      |: :|| |||||  | :: :||:|::    : |
HBB_HUMAN       44       SFGDLSTPDAVMGNPKVKAHGKKVLGAFSDGLAHLDNLKGTFATL 88

HBA_HUMAN       84       SDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKF 128
                         |:||  || ||| ||:|| : |:  || |   |||| | |:  |
HBB_HUMAN       89       SELHCDKLHVDPENFRLLGNVLVCVLAHHFGKEFTPPVQAAYQKV 133

HBA_HUMAN       129      LASVSTVLTSKYR                                 141
                         :| |:  |  ||
HBB_HUMAN       134      VAGVANALAHKYH                                 146

%id = 45.32             %similarity = 63.31
Overall %id = 43.15; Overall %similarity = 60.27
```

Smith-Waterman Algorithm

Advances in
Applied Mathematics, 2:482-489 (1981)

The Smith-Waterman algorithm is a local alignment tool used to obtain sensitive pairwise similarity alignments. Smith-Waterman algorithm uses dynamic programming. Operating via a matrix, the algorithm uses backtracing and tests alternative paths to the highest scoring alignments, and selects the optimal path as the highest ranked alignment. The sensitivity of the Smith-Waterman algorithm makes it useful for finding local areas of similarity between sequences that are too dissimilar for alignment. The S-W algorithm uses a lot of computer memory. BLAST and FASTA are other search algorithms that use some aspects of S-W.

Smith-Waterman (cont. 1)

a. It searches for both full and partial sequence matches .

b. Assigns a score to each pair of amino acids

- -uses similarity scores
- -uses positive scores for related residues
- -uses negative scores for substitutions and gaps

c. Initializes edges of the matrix with zeros

d. As the scores are summed in the matrix, any sum below 0 is recorded as a zero.

e. Begins backtracing at the maximum value found anywhere in the matrix.

f. Continues the backtrace until the score falls to 0.

Smith-Waterman (cont. 2)

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	5	0	0	0	0	0
W	0	0	0	0	3	0	20	12	4	0	0
H	0	10	2	0	0	0	12	18	22	14	6
E	0	2	16	8	0	0	4	10	18	28	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	6	13	18	12	4	0	4	16	26

Put zeros on borders. Assign initial scores based on a scoring matrix. Calculate new scores based on adjacent cell scores. If sum is less than zero or equal to zero begin new scoring with next cell.

This example uses the BLOSUM45 Scoring Matrix with a gap extension penalty of -3

Smith-Waterman (cont. 3)

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	5	0	0	0	0	0
W	0	0	0	0	3	0	20	12	4	0	0
H	0	10	2	0	0	0	12	18	22	14	6
E	0	2	16	8	0	0	4	10	18	28	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	6	13	18	12	4	0	4	16	26

```
AWGHE
|| ||
AW-HE
```

Path Score=28

Begin backtrace at the **maximum** value found anywhere on the matrix.
Continue the backtrace until score falls to zero

Calculation of percent similarity

```
A   W   G   H   E
A   W   -   H   E
```

```
5   15   -5   10   6        Blosum45 SCORES

          -3                GAP EXT. PENALTY
```

% SIMILARITY =
NUMBER OF POS. SCORES
DIVIDED BY NUMBER OF AAs
IN REGION x 100

% SIMILARITY = 4/5 x 100
= 80%