A Perl Program for Sequence Alignment

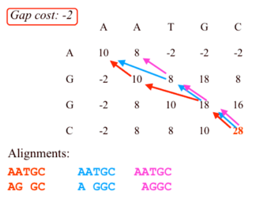
Sequence Alignment

The different steps of dynamic programming:

-build the DP matrix
-Trace-back
-Outputs the alignment

Parameters:

-The scoring matrix
-Gap penalty



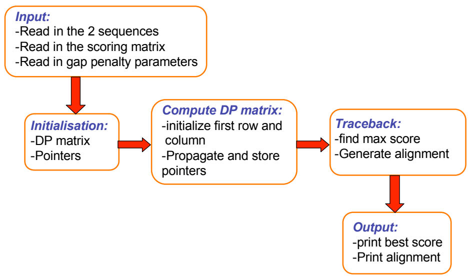
	A	A	T	G	C
A	10	8	-2	-2	-2
G	-2	10	8	18	8
G	-2	8	10	18	16
C	-2	8	8	10	28

Alignments:

```
AATGC      AATGC   AATGC
AG GC      A GGC    AGGC
```

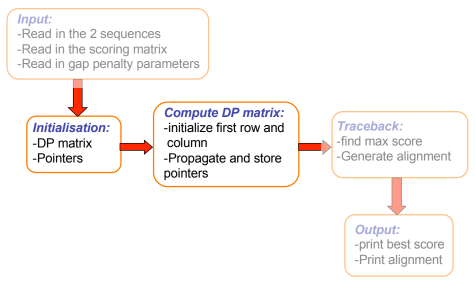
Sequence Alignment

The different steps of a Perl program for Sequence Alignment:



Sequence Alignment

The different steps of a Perl program for Sequence Alignment:



Sequence Alignment

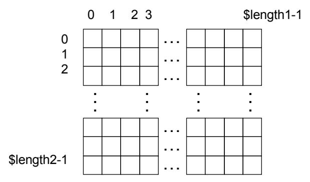
Step 2: Initialisation:

What we need:

- a 2D array to store the DP matrix: @MatAlign
- a 2D array to store row pointer: @PointerI
- a 2D array to store column pointer: @PointerJ

Size of the arrays:

$length2 row and $length1 column:



Sequence Alignment: initialization

```
# Initialize the three matrices we need:
# =======================================
#       - the alignment matrix MatAlign
#       - pointer along i : PointerI
#       - pointer along j : PointerJ
#
for ($i=0; $i<$length2; $i++)
{
        for ($j=0; $j<$length1; $j++)
        {
                $MatAlign[$i][$j]=0;
                $PointerI[$i][$j]=0;
                $PointerJ[$i][$j]=0;
        }
}
```

Sequence Alignment: Compute DP matrix

1. Initialize first row:

```
#
# first row
#
# First amino acid in sequence2, and its position in scoring matrix:
#
$AAi = $seq2[0];
$Pos_i = $position{$AAi};  ⟶  hash array that gives position
#                               position of AA in Score.
# Loop over all amino acids of sequence 1:
#
for ($j=0; $j<$length1; $j++)
{
     $AAj = $seq1[$j];
     $Pos_j = $position{$AAj};
     $MatAlign[0][$j] = $Score[$Pos_i][$Pos_j];
#     if($j > 0) {$MatAlign[0][$j] -=$Gop;}  ⟶  Would allow for gaps
}                                                 in first row
```

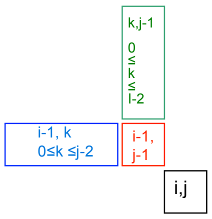
Sequence Alignment: Compute DP matrix

2. Initialize first column:

```
#
# first column
#
# First amino acid in sequence1, and its position in scoring matrix:
#
$AAj = $seq1[0];
$Pos_j = $position{$AAj};
#
# Loop over all amino acids of sequence 1:
#
for ($i=0; $i<$length2; $i++)
{
     $AAi = $seq2[$i];
     $Pos_i = $position{$AAi};
     $MatAlign[$i][0] = $Score[$Pos_i][$Pos_j];
#     if($i > 0) {$MatAlign[$i][0] -=$Gop;}
}
```

Sequence Alignment: Compute DP matrix

3. Propagate

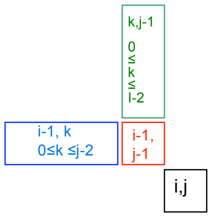


3.1 Find score of aligning i with j:

```
$AAi = $seq2[$i];
$AAj = $seq1[$j];
$Pos_i = $position{$AAi};
$Pos_j = $position{$AAj};
$score_ij = $Score[$Pos_i][$Pos_j];
```

Sequence Alignment: Compute DP matrix

3. Propagate



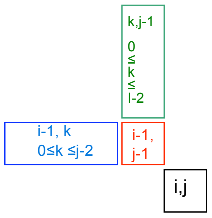
3.2 Score coming from (i-1,j-1):

```
#
#     i-1 aligned with j-1:
#
$score1 = $MatAlign[$i-1][$j-1];
$ipos1   = $i-1;
$jpos2   = $j-1;
```

Sequence Alignment: Compute DP matrix

3.3 Gap in sequence 1 (only for j>1):

3. Propagate

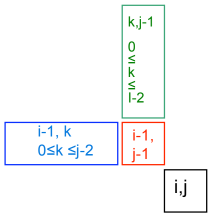


```
# For k = 0:
#
$gap = $Gop+($j-1)*$Gext;
$score2 = $MatAlign[$i-1][0] -$gap;
$ipos2 = i -1;
$jpos2 = 0;
#
# for remaining k values
#
for ($k = 1; $k<$j-1; $k++)
{
         $gap = $Gop + ($j-1-$k)*Gext;
         $score_test = $MatAlign[$i-1][$k]-$gap;
        if($score_test > $score2)
        {
              $score2 = $score_test;
              $jpos2 = $k;
         }
 }
```

Sequence Alignment: Compute DP matrix

3.3 Gap in sequence 2 (only for i>1):

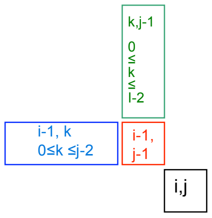
3. Propagate



```
# For k = 0:
#
$gap = $Gop+($i-1)*$Gext;
$score3 = $MatAlign[0][$j-1] -$gap;
$ipos3 = 0;
$jpos3 = j-1;
#
# for remaining k values
#
for ($k = 1; $k<$i-1; $k++)
{
         $gap = $Gop + ($i-1-$k)*Gext;
         $score_test = $MatAlign[$k][$j-1]-$gap;
        if($score_test > $score3)
        {
              $score3 = $score_test;
              $ipos3 = $k;
         }
 }
```

Sequence Alignment: Compute DP matrix

3. Propagate

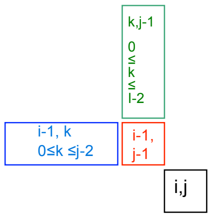


3.4 Combine the 3: find optimal:

```
$bestscore = $score1;
$ipos = $ipos1;
$jpos = $jpos1;
if($j > 1)
{
        if($score2 > $bestscore)
        {
                $bestscore = $score2;
                $ipos = $ipos2;
                $jpos = $jpos2;
        }
}
if($i > 1)
{
        if($score3 > $bestscore)
        {
                $bestscore = $score3;
                $ipos = $ipos3;
                $jpos = $jpos3;
        }
}
```

Sequence Alignment: Compute DP matrix

3. Propagate



3.5: Finally, update score + pointers:

```
$MatAlign[$i][$j]=$score_ij+$bestscore;
$PointerI[$i][$j]=ipos;
$PointerJ[$i][$j]=jpos;
```