SCSB3213 BIOINFORMATICS DATABASE
ASSIGNMENT 1

Main Pairwise Sequence Alignment Function

The pairwiseAlignment function solves the pairwise sequence alignment problems mentioned above. It aligns one or more strings specified in the pattern argument with a single string specified in the subject argument.

```
> library(Biostrings)
> pairwiseAlignment(pattern = c("succeed", "precede"), subject =
"supersede")
Global PairwiseAlignmentsSingleSubject (1 of 2)
pattern: succ--eed
subject: supersede
score: -33.99738
```

The type of pairwise sequence alignment is set by specifying the type argument to be one of "global", “local", "overlap", "global-local", and "local-global".

```
> pairwiseAlignment(pattern = c("succeed", "precede"), subject =
"supersede", type = "local")
Local PairwiseAlignmentsSingleSubject (1 of 2)
pattern: [1] su
subject: [1] su
score: 5.578203
```

The gap penalties are regulated by the gapOpening and gapExtension arguments.

```
> pairwiseAlignment(pattern = c("succeed", "precede"), subject =
"supersede", gapOpening = 0, gapExtension = 1)
Global PairwiseAlignmentsSingleSubject (1 of 2)
pattern: su-cce--edsubject: sup--ersede
score: 7.945507
```

The substitution scoring scheme is set using three arguments, two of which are quality-based related (patternQuality, subjectQuality) and one is fixed substitution related (substitutionMatrix). When the substitution scores are fixed by character pairing, the substituionMatrix argument takes a matrix with the appropriate alphabets as dimension names. The nucleotide SubstitutionMatrix function tranlates simple match and mismatch scores to the full spectrum of IUPAC nucleotide codes.

```
> submat <- matrix(-1, nrow = 26, ncol = 26, dimnames = list(letters,
letters))
> diag(submat) <- 0
> pairwiseAlignment(pattern = c("succeed", "precede"), subject =
"supersede",substitutionMatrix = submat,gapOpening = 0, gapExtension = 1)
Global PairwiseAlignmentsSingleSubject (1 of 2)
pattern: succe-ed subject: supersede
```

```
score: -5
```
The final argument, scoreOnly, to the pairwiseAlignment function accepts a logical value to specify whether or not to return just the pairwise sequence alignment score. If scoreOnly is FALSE, the pairwise alignment with the maximum alignment score is returned. If more than one pairwise alignment has the maximum alignment score exists, the first alignment along the subject is returned. If there are multiple pairwise alignments with the maximum alignment score at the chosen subject location, then at each location along the alignment mismatches are given preference to insertions/deletions. For example, pattern: [1] ATTA; subject: [1] AT-A is chosen above pattern: [1] ATTA; subject: [1] A-TA if they both have the maximum alignment score.

```
> submat <- matrix(-1, nrow = 26, ncol = 26, dimnames = list(letters, letters))
> diag(submat) <- 0
> pairwiseAlignment(pattern = c("succeed", "precede"), subject = "supersede", substitutionMatrix = submat,gapOpening = 0, gapExtension = 1, scoreOnly = TRUE)
[1] -5 -5
```

Exercise 1

1. Using pairwiseAlignment, fit the global, local, and overlap pairwise sequence alignment of the strings "syzygy" and "zyzzyx" using the default settings.
2. Do any of the alignments change if the gapExtension argument is set to -Inf?

Pairwise Sequence Alignment Classes

Following the design principles of Bioconductor and R, the pairwise sequence alignment functionality in the Biostrings package keeps the end user close to their data through the use of five specialty classes: PairwiseAlignments, PairwiseAlignmentsSingleSubject, PairwiseAlignmentsSingleSubjectSummary, AlignedXStringSet, and QualityAlignedXStringSet. The PairwiseAlignmentsSingleSubject class inherits from the PairwiseAlignments class and they both hold the results of a fit from the pairwiseAlignment function, with the former class being used to represent all patterns aligning to a single subject and the latter being used to represent elementwise alignments between a set of patterns and a set of subjects.

```
> pa1 <- pairwiseAlignment(pattern = c("succeed", "precede"), subject = "supersede")
> class(pa1)

[1] "PairwiseAlignmentsSingleSubject"
attr(,"package")
[1] "Biostrings"
```

and the pairwiseAlignmentSummary function holds the results of a summarized pairwise sequence alignment.

```
> summary(pa1)
```

```
Global Single Subject Pairwise Alignments
Number of Alignments: 2
Scores:
Min. 1st Qu. Median Mean 3rd Qu. Max.
-34.00 -31.78 -29.56 -29.56 -27.34 -25.12
Number of matches:
Min. 1st Qu. Median Mean 3rd Qu. Max.
3.00 3.25 3.50 3.50 3.75 4.00
Top 7 Mismatch Counts:
SubjectPosition Subject Pattern Count Probability
1 3 p c 1 0.5
2 4 e c 1 0.5
3 4 e r 1 0.5
4 5 r e 1 0.5
5 6 s c 1 0.5
6 8 d e 1 0.5
7 9 e d 1 0.5
> class(summary(pa1))

[1] "PairwiseAlignmentsSingleSubjectSummary"
attr(,"package")
[1] "Biostrings"
```

The AlignedXStringSet and QualityAlignedXStringSet classes hold the "gapped" S'i substrings with the former class holding the results when the pairwise sequence alignment is performed with a fixed substitution scoring scheme and the latter class a quality-based scoring scheme.

```
> class(pattern(pa1))
[1] "QualityAlignedXStringSet"
attr(,"package")
[1] "Biostrings"

> submat <- matrix(-1, nrow = 26, ncol = 26, dimnames = list(letters,
letters))
> diag(submat) <- 0
> pa2 <- pairwiseAlignment(pattern = c("succeed", "precede"), subject =
"supersede",substitutionMatrix = submat, gapOpening = 0, gapExtension = 1)
> class(pattern(pa2))

[1] "AlignedXStringSet"
attr(,"package")
[1] "Biostrings"
```

Pairwise Sequence Alignment Helper Functions

Table 1 show functions that interact with objects of class PairwiseAlignments, PairwiseAlignmentsSingleSubject, and AlignedXStringSet. These functions should be used in preference to direct slot extraction from the alignment objects. The score, nedit, nmatch, nmismatch, and nchar functions return numeric vectors containing information on the pairwise sequence alignment score, number of matches, number of mismatches, and number of
aligned characters respectively.

```
> submat <- matrix(-1, nrow = 26, ncol = 26, dimnames = list(letters, letters))
> diag(submat) <- 0
> pa2 <-pairwiseAlignment(pattern = c("succeed", "precede"), subject = "supersede",substitutionMatrix = submat, gapOpening = 0, gapExtension = 1)
> score(pa2)
[1] -5 -5
```

Table 1: Functions for PairwiseAlignments and PairwiseAlignmentsSingleSubject objects

Function	Description
[	Extracts the specified elements of the alignment object.
alphabet	Extracts the allowable characters in the original strings
compareStrings	Creates character string mashups of the alignments
deletion	Extracts the locations of the gaps inserted into the pattern for the alignments
length	Extracts the number of patterns aligned
mismatchTable	Creates a table for the mismatching positions
nchar	Computes the length of “gapped” substrings
nedit	Computes the Levenshtein edit distance of the alignments
indel	Extracts the locations of the insertion & deletion gaps in the alignments
insertion	Extracts the locations of the gaps inserted into the subject for the alignments
nindel	Computes the number of insertions & deletions in the alignments
nmatch	Computes the number of matching characters in the alignments
nmismatch	Computes the number of mismatching characters in the alignments
pattern, subject	Extracts the aligned pattern/subject

pid	Computes the percent sequence identity
rep	Replicates the elements of the alignment object
score	Extracts the pairwise sequence alignment scores
type	Extracts the type of pairwise sequence alignment

```
> nedit(pa2)
[1] 4 5

> nmatch(pa2)
[1] 4 4

> nmismatch(pa2)
[1] 3 3

> nchar(pa2)
[1] 8 9

> aligned(pa2)
BStringSet object of length 2:
width seq
[1] 9 succe-ed-
[2] 9 pr-ec-ede

> as.character(pa2)
[1] "succe-ed-" "pr-ec-ede"

> as.matrix(pa2)
[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[1,] "s" "u" "c" "c" "e" "-" "e" "d" "-"
[2,] "p" "r" "-" "e" "c" "-" "e" "d" "e"
```

Application: Using Evolutionary Models in Protein Alignments

When proteins are believed to descend from a common ancestor, evolutionary models can be used as a guide in pairwise sequence alignments. The two most common families evolutionary models of proteins used in pairwise sequence alignments are Point Accepted Mutation (PAM) matrices, which are based on explicit evolutionary models, and Block Substitution Matrix (BLOSUM) matrices, which are based on dataderived evolution models. The Biostrings package contains 5 PAM and 5 BLOSUM matrices (PAM30 PAM40, PAM70, PAM120, PAM250, BLOSUM45, BLOSUM50, BLOSUM62, BLOSUM80, and BLOSUM100) that can be used in the substitutionMatrix argument to the pairwiseAlignment function. Here is an example pairwise sequence alignment of amino acids from Durbin, Eddy et al being fit by the pairwiseAlignment function using the

BLOSUM50 matrix:

```
> data(BLOSUM50)
> BLOSUM50[1:4,1:4]
A R N D
A 5 -2 -1 -2
R -2 7 -1 -2
N -1 -1 7 2
D -2 -2 2 8
> nwdemo <-pairwiseAlignment(AAString("PAWHEAE"), AAString("HEAGAWGHEE"),
substitutionMatrix = BLOSUM50,gapOpening = 0, gapExtension = 8)
> nwdemo
Global PairwiseAlignmentsSingleSubject (1 of 1)
pattern: -PA--W-HEAE
subject: HEAGAWGHE-E
score: 1
> compareStrings(nwdemo)
[1] "?A--W-HE+E"
> pid(nwdemo)
[1] 50
```

Exercise 2

1. Repeat the alignment exercise above using BLOSUM62, a gap opening penalty of 12, and a gap extension penalty of 4.
2. Explore to find out what caused the alignment to change.