[Re] Local alignment statistics - Simulate full set of data

Nathan Brouwer

11/18/2019

Simulate data for 1 sub-experiment as for() loop

In this sections the basic computations of the experiment are iterated in order to generate the date for one row of the table. This requires the Robinson and Robinson amino acid table to be loaded from the appropriate script.

Run experiment for n = m = 191 amino acids

The first row of Table 1, page 465 indicates that simulated sequences of 191 amino acids were investigated. **In the original simulation 10000 iterations were carried out; we'll do just 1000.**

for() loop for 191 amino acids

```
# load libraries
library(Biostrings)
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter,
##     Find, get, grep, grepl, intersect, is.unsorted, lapply, Map,
##     mapply, match, mget, order, paste, pmax, pmax.int, pmin,
##     pmin.int, Position, rank, rbind, Reduce, rownames, sapply,
##     setdiff, sort, table, tapply, union, unique, unsplit, which,
##     which.max, which.min
```

```
## Loading required package: S4Vectors
```

```
## Loading required package: stats4
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following object is masked from 'package:base':
##
##     expand.grid
## Loading required package: IRanges
##
## Attaching package: 'IRanges'
## The following object is masked from 'package:grDevices':
##
##     windows
## Loading required package: XVector
##
## Attaching package: 'Biostrings'
## The following object is masked from 'package:base':
##
##     strsplit
```

```r
# load amino acid frequencies
robinson.aafreq <- read.csv(file = "robinson_aafreq.csv")

# create dataframe to store scores and lengths
random.scores.191 <- data.frame(interation = 1:1000,
                          score.i = NA,
                          length.i = NA,
                          seq.length1 = 191,
                          seq.length2 = 191)

# for loop

for(i in 1:1000){
 seq1 <- sample(x = robinson.aafreq$aa1,
                  size = 191,
                  replace = TRUE,
                  prob = robinson.aafreq$aa.freq)
 seq1 <- paste(seq1, sep = "",
                  collapse = "")

 seq2 <- sample(x = robinson.aafreq$aa1,
                  size = 191,
                  replace = TRUE,
                  prob = robinson.aafreq$aa.freq)
 seq2 <- paste(seq2, sep = "",
                  collapse = "")

 alignment.i <- pairwiseAlignment(pattern = seq1,
                                  subject = seq2,
                    type               = "local",
                    substitutionMatrix = "BLOSUM62",
                    gapOpening         = 12,
                    gapExtension       = 1)
 score.i         <- score(alignment.i)
```

```
 alignment.seq.i <- toString(alignment.i)

 length.i        <- nchar(alignment.seq.i)


 random.scores.191$score.i[i]  <- score.i
 random.scores.191$length.i[i] <- length.i

}
```

The results of one full set of simulations look like this:

```
head(random.scores.191)
```

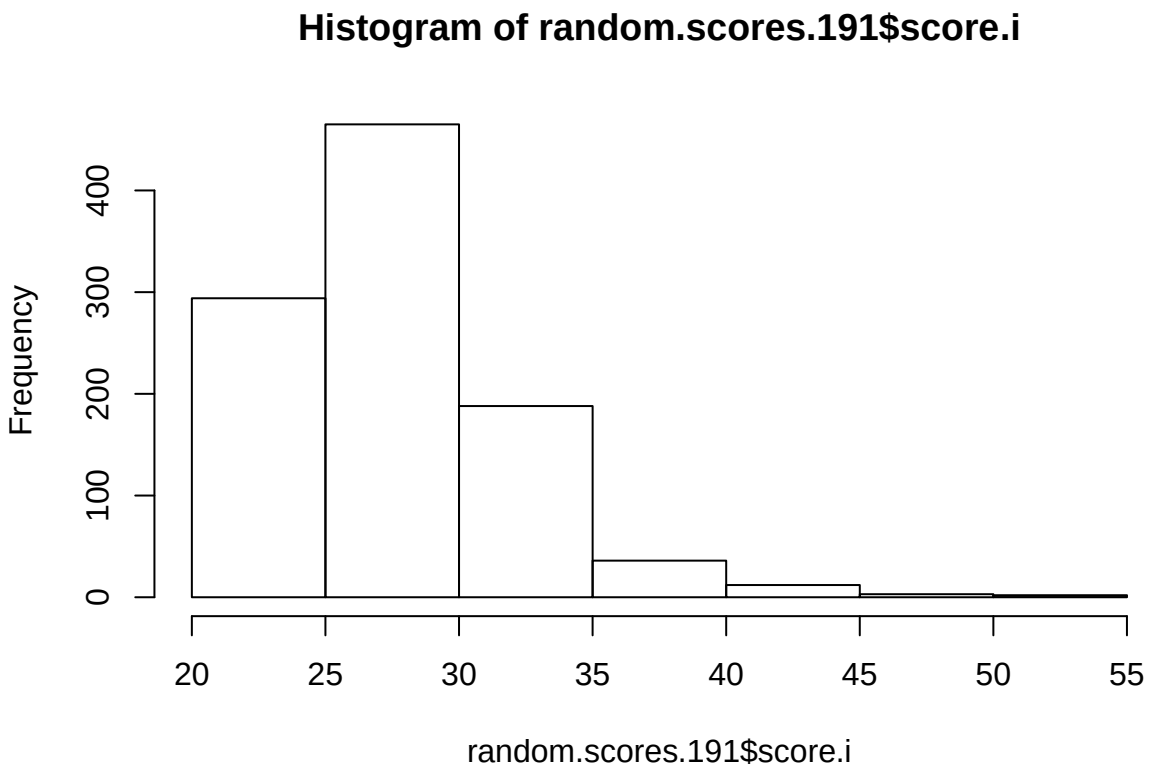
```
##   interation score.i length.i seq.length1 seq.length2
## 1          1      23       10         191         191
## 2          2      32       15         191         191
## 3          3      35       14         191         191
## 4          4      29       12         191         191
## 5          5      29       15         191         191
## 6          6      31       41         191         191
```

```
summary(random.scores.191)
```

```
##    interation        score.i         length.i       seq.length1
##  Min.   :   1.0   Min.   :20.00   Min.   :  4.00   Min.   :191
##  1st Qu.: 250.8   1st Qu.:25.00   1st Qu.: 12.00   1st Qu.:191
##  Median : 500.5   Median :27.00   Median : 18.50   Median :191
##  Mean   : 500.5   Mean   :28.09   Mean   : 21.83   Mean   :191
##  3rd Qu.: 750.2   3rd Qu.:30.00   3rd Qu.: 28.00   3rd Qu.:191
##  Max.   :1000.0   Max.   :55.00   Max.   :115.00   Max.   :191
##   seq.length2
##  Min.   :191
##  1st Qu.:191
##  Median :191
##  Mean   :191
##  3rd Qu.:191
##  Max.   :191
```

The distribution of scores can be plotted with hist()

```
hist(random.scores.191$score.i)
```



Save the output

```
write.csv(random.scores.191, file = "random_scores_191.csv")
```