Stat_computing_1

Yue Gu

9/17/2021

2 Length of Ones

Given an integer vector of 0 and 1's, write an R function named len_one that returns the length of its longest subsequence of consecutive 1's, among all possible subsequences of consecutive 1's.

```
# write function
len_one = function(seq){
  s = rle(seq)
  l = s$lengths[s$val == 1]
  if (length(l) > 0)
    max(l)
  else
    0
}
# test
len_one(c(0, 0, 1, 1, 1, 0, 1, 1))
```

```
## [1] 3
```

```
len_one(c(1, 1, 1, 1))
```

```
## [1] 4
```

```
len_one(c(0, 0, 0, 0))
```

```
## [1] 0
```

3 Hybrid Matrix

a. create a covariance matrix with 2's on the diagonal and 1's off the diagonal. The dimension of this matrix is 10 x 10.

b. convert this covariance matrix to a correlation matrix.

c. compute the Cholesky factorization of the correlation matrix.

d. create a hybrid matrix representation of both the correlation matrix and its Cholesky factorization, where its upper triangular part is the upper triangular part of the correlation matrix (excluding the diagonal) and its lower triangular part is the upper triangular part including the diagonal of the Cholesky factorization (by chol).

```
# (a) create cov matrix
cov_matrix = matrix(nrow = 10, ncol = 10)
diag(cov_matrix) = 2
cov_matrix[lower.tri(cov_matrix)]=1
cov_matrix[upper.tri(cov_matrix)]=1

cov_matrix
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    2    1    1    1    1    1    1    1    1     1
##  [2,]    1    2    1    1    1    1    1    1    1     1
##  [3,]    1    1    2    1    1    1    1    1    1     1
##  [4,]    1    1    1    2    1    1    1    1    1     1
##  [5,]    1    1    1    1    2    1    1    1    1     1
##  [6,]    1    1    1    1    1    2    1    1    1     1
##  [7,]    1    1    1    1    1    1    2    1    1     1
##  [8,]    1    1    1    1    1    1    1    2    1     1
##  [9,]    1    1    1    1    1    1    1    1    2     1
## [10,]    1    1    1    1    1    1    1    1    1     2
```

```
# (b) convert cov matrix to correlation matrix
cor_matrix = cov2cor(cov_matrix); cor_matrix
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5
##  [2,]  0.5  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5
##  [3,]  0.5  0.5  1.0  0.5  0.5  0.5  0.5  0.5  0.5   0.5
##  [4,]  0.5  0.5  0.5  1.0  0.5  0.5  0.5  0.5  0.5   0.5
##  [5,]  0.5  0.5  0.5  0.5  1.0  0.5  0.5  0.5  0.5   0.5
##  [6,]  0.5  0.5  0.5  0.5  0.5  1.0  0.5  0.5  0.5   0.5
##  [7,]  0.5  0.5  0.5  0.5  0.5  0.5  1.0  0.5  0.5   0.5
##  [8,]  0.5  0.5  0.5  0.5  0.5  0.5  0.5  1.0  0.5   0.5
##  [9,]  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  1.0   0.5
## [10,]  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0
```

```
# (c) compute the Cholesky factorization of the correlation matrix
cor_matrix_chol = chol(cor_matrix); cor_matrix_chol
```

```
##       [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,]    1 0.5000000 0.5000000 0.5000000 0.5000000 0.5000000 0.5000000
##  [2,]    0 0.8660254 0.2886751 0.2886751 0.2886751 0.2886751 0.2886751
##  [3,]    0 0.0000000 0.8164966 0.2041241 0.2041241 0.2041241 0.2041241
##  [4,]    0 0.0000000 0.0000000 0.7905694 0.1581139 0.1581139 0.1581139
##  [5,]    0 0.0000000 0.0000000 0.0000000 0.7745967 0.1290994 0.1290994
##  [6,]    0 0.0000000 0.0000000 0.0000000 0.0000000 0.7637626 0.1091089
##  [7,]    0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.7559289
##  [8,]    0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
```

```
##  [9,]    0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
## [10,]    0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000
##             [,8]       [,9]      [,10]
##  [1,] 0.50000000 0.50000000 0.50000000
##  [2,] 0.28867513 0.28867513 0.28867513
##  [3,] 0.20412415 0.20412415 0.20412415
##  [4,] 0.15811388 0.15811388 0.15811388
##  [5,] 0.12909944 0.12909944 0.12909944
##  [6,] 0.10910895 0.10910895 0.10910895
##  [7,] 0.09449112 0.09449112 0.09449112
##  [8,] 0.75000000 0.08333333 0.08333333
##  [9,] 0.00000000 0.74535599 0.07453560
## [10,] 0.00000000 0.00000000 0.74161985
```

```
t(cor_matrix_chol)
```

```
##       [,1]      [,2]      [,3]      [,4]      [,5]      [,6]       [,7]
##  [1,]  1.0 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.00000000
##  [2,]  0.5 0.8660254 0.0000000 0.0000000 0.0000000 0.0000000 0.00000000
##  [3,]  0.5 0.2886751 0.8164966 0.0000000 0.0000000 0.0000000 0.00000000
##  [4,]  0.5 0.2886751 0.2041241 0.7905694 0.0000000 0.0000000 0.00000000
##  [5,]  0.5 0.2886751 0.2041241 0.1581139 0.7745967 0.0000000 0.00000000
##  [6,]  0.5 0.2886751 0.2041241 0.1581139 0.1290994 0.7637626 0.00000000
##  [7,]  0.5 0.2886751 0.2041241 0.1581139 0.1290994 0.1091089 0.75592895
##  [8,]  0.5 0.2886751 0.2041241 0.1581139 0.1290994 0.1091089 0.09449112
##  [9,]  0.5 0.2886751 0.2041241 0.1581139 0.1290994 0.1091089 0.09449112
## [10,]  0.5 0.2886751 0.2041241 0.1581139 0.1290994 0.1091089 0.09449112
##             [,8]      [,9]     [,10]
##  [1,] 0.00000000 0.0000000 0.0000000
##  [2,] 0.00000000 0.0000000 0.0000000
##  [3,] 0.00000000 0.0000000 0.0000000
##  [4,] 0.00000000 0.0000000 0.0000000
##  [5,] 0.00000000 0.0000000 0.0000000
##  [6,] 0.00000000 0.0000000 0.0000000
##  [7,] 0.00000000 0.0000000 0.0000000
##  [8,] 0.75000000 0.0000000 0.0000000
##  [9,] 0.08333333 0.7453560 0.0000000
## [10,] 0.08333333 0.0745356 0.7416198
```

```
# check result
t(cor_matrix_chol) %*% cor_matrix_chol
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5
##  [2,]  0.5  1.0  0.5  0.5  0.5  0.5  0.5  0.5  0.5   0.5
##  [3,]  0.5  0.5  1.0  0.5  0.5  0.5  0.5  0.5  0.5   0.5
##  [4,]  0.5  0.5  0.5  1.0  0.5  0.5  0.5  0.5  0.5   0.5
##  [5,]  0.5  0.5  0.5  0.5  1.0  0.5  0.5  0.5  0.5   0.5
##  [6,]  0.5  0.5  0.5  0.5  0.5  1.0  0.5  0.5  0.5   0.5
##  [7,]  0.5  0.5  0.5  0.5  0.5  0.5  1.0  0.5  0.5   0.5
##  [8,]  0.5  0.5  0.5  0.5  0.5  0.5  0.5  1.0  0.5   0.5
##  [9,]  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  1.0   0.5
## [10,]  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5  0.5   1.0
```

```r
# (d) create a hybrid matrix representation
hybrid_matrix = matrix(nrow = 10, ncol = 10)
hybrid_matrix[upper.tri(hybrid_matrix)] = cor_matrix[upper.tri(cor_matrix)]
hybrid_matrix[lower.tri(hybrid_matrix)] = cor_matrix_chol[upper.tri(cor_matrix_chol)]
diag(hybrid_matrix) = diag(cor_matrix_chol)

hybrid_matrix
```

```
##            [,1]      [,2]      [,3]       [,4]       [,5]       [,6]      [,7]
##  [1,] 1.0000000 0.5000000 0.5000000 0.50000000 0.50000000 0.50000000 0.5000000
##  [2,] 0.5000000 0.8660254 0.5000000 0.50000000 0.50000000 0.50000000 0.5000000
##  [3,] 0.5000000 0.1581139 0.8164966 0.50000000 0.50000000 0.50000000 0.5000000
##  [4,] 0.2886751 0.5000000 0.2041241 0.79056942 0.50000000 0.50000000 0.5000000
##  [5,] 0.5000000 0.2886751 0.1581139 0.15811388 0.77459667 0.50000000 0.5000000
##  [6,] 0.2886751 0.2041241 0.1290994 0.12909944 0.20412415 0.76376262 0.5000000
##  [7,] 0.2041241 0.1581139 0.1091089 0.10910895 0.15811388 0.08333333 0.7559289
##  [8,] 0.5000000 0.1290994 0.5000000 0.09449112 0.12909944 0.50000000 0.1581139
##  [9,] 0.2886751 0.5000000 0.2886751 0.50000000 0.10910895 0.28867513 0.1290994
## [10,] 0.2041241 0.2886751 0.2041241 0.28867513 0.09449112 0.20412415 0.1091089
##             [,8]      [,9]     [,10]
##  [1,] 0.50000000 0.5000000 0.5000000
##  [2,] 0.50000000 0.5000000 0.5000000
##  [3,] 0.50000000 0.5000000 0.5000000
##  [4,] 0.50000000 0.5000000 0.5000000
##  [5,] 0.50000000 0.5000000 0.5000000
##  [6,] 0.50000000 0.5000000 0.5000000
##  [7,] 0.50000000 0.5000000 0.5000000
##  [8,] 0.75000000 0.5000000 0.5000000
##  [9,] 0.09449112 0.7453560 0.5000000
## [10,] 0.08333333 0.0745356 0.7416198
```

4 Translation

Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M, shown by the following table: Symbol Value I 1 V 5 X 10 L 50 C 100 D 500 M 1000 Write an R function named roman_trans using appropriate data structures that returns the value of an input roman symbol listed above.

```r
roman_trans = function(x){
  x = as.character(x)
  ifelse(x == "I", 1,
         ifelse(x == "V", 5,
                ifelse(x == "X", 10,
                       ifelse(x == "L", 50,
                              ifelse(x == "C", 100,
                                     ifelse(x == "D", 500,
                                            ifelse(x == "M", 1000, NA)))))))

}

# test
roman_trans("V")
```

```
## [1] 5
```

```
roman_trans("I")
```

```
## [1] 1
```

```
roman_trans("D")
```

```
## [1] 500
```

```
roman_trans("M")
```

```
## [1] 1000
```