Functions Exercise

Nicole Mis

February 8, 2018

1.

(a) Write functions `tmpFn1` and `tmpFn2` such that if `xVec` is the vector $(x_1, x_2, ..., x_n)$, then `tmpFn1(xVec)` returns vector $(x_1, x_2^2, ..., x_n^n)$ and `tmpFn2(xVec)` returns the vector $(x_1, \frac{x_2^2}{2}, ..., \frac{x_n^n}{n})$.

Here is `tmpFn1`

```
tmpFn1 <- function(xVec){
  return(xVec^(1:length(xVec)))
}

## simple example
a <- c(2, 5, 3, 8, 2, 4)

b <- tmpFn1(a)
b
```

```
## [1]    2   25   27 4096   32 4096
```

and now `tmpFn2`

```
tmpFn2 <- function(xVec2){

  n = length(xVec2)

  return(xVec2^(1:n)/(1:n))
}




c <- tmpFn2(a)
c
```

```
## [1]    2.0000   12.5000    9.0000 1024.0000    6.4000  682.6667
```

(b) Now write a fuction `tmpFn3` which takes 2 arguments x and n where x is a single number and n is a strictly positive integer. The function should return the value of

$$1 + \frac{x}{1} + \frac{x^2}{2} + \frac{x^3}{3} + ... + \frac{x^n}{n}$$

```
tmpFn3 <-function(x, n){
  1+sum((x^(1:n))/(1:n))
}
```

2. Write a funciton `tmpFn(xVec)` such that if `xVec` is the vector $x = (x_1, ..., x_n)$ then `tmpFn(xVec)` returns the vector of moving averages:

$$\frac{x_1 + x_2 + x_3}{3}, \frac{x_2 + x_3 + x_4}{3}, ..., \frac{x_{n-2} + x_{n-1} + x_n}{3}$$

Try out your function. for example, try `tmpFn`(c(1:5,6:1))

```
tmpFn <- function(xVec){
   lenx <- length(xVec)
   (xVec[1:(lenx-2)]+xVec[2:(lenx-1)]+xVec[3:lenx])/3
}
tmpFn(c(1:5,6:1))
```

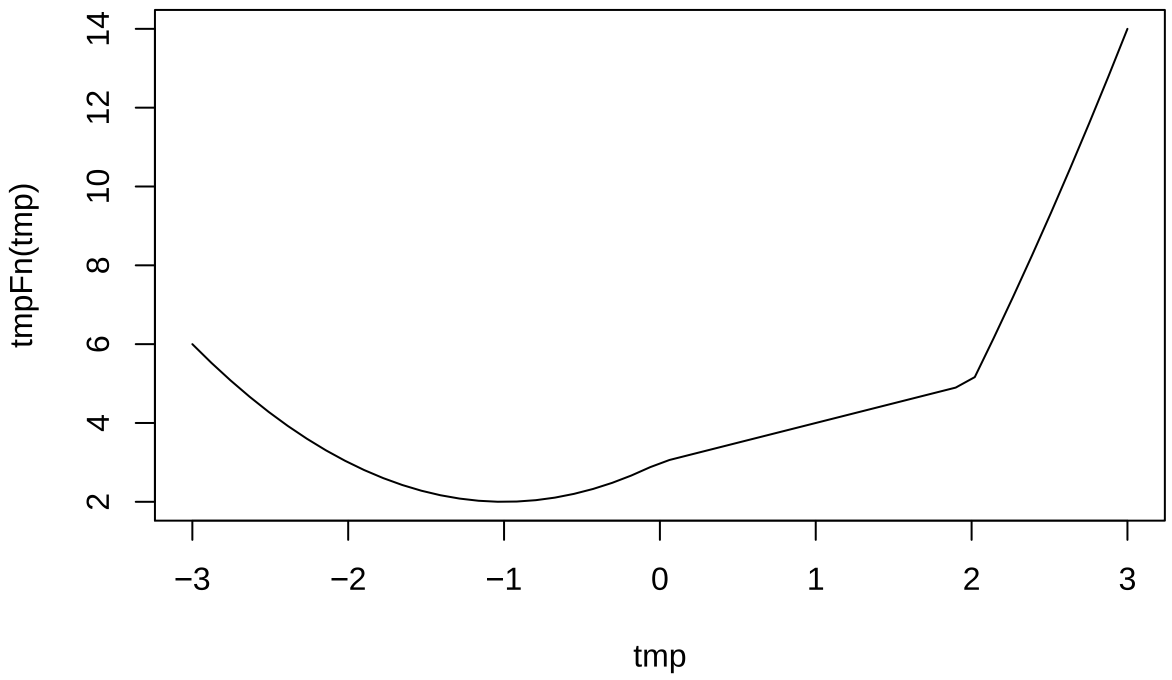
```
## [1] 2.000000 3.000000 4.000000 5.000000 5.333333 5.000000 4.000000 3.000000
## [9] 2.000000
```

3. Consider the continuous function

$$f(x) = \begin{cases} x^2 + 2x + 3 & if \quad x < 0 \\ x + 3 & if \quad 0 \leq x < 2 \\ x^2 + 4x - 7 & if \quad 2 \leq x \end{cases}$$

Write a function `tmpFn` which takes a single argument `xVec`. the function should return the vector of values of the function $f(x)$ evaluated at the values in `xVec`.
Hence plot the function $f(x)$ for $-3 < x < 3$.

```
tmpFn <- function(x)
{
ifelse(x < 0, x^2 + 2*x + 3, ifelse(x < 2, x+3, x^2 + 4*x - 7))
}
tmp <- seq(-3, 3, len=50)
plot(tmp, tmpFn(tmp), type="l")
```



4. Write a function which takes a single argument which is a matrix. The function should return a matrix which is the same as the function argument but every odd number is doubled.

Hence the result of using the function on the matrix

$$\begin{bmatrix} 1 & 1 & 3 \\ 5 & 2 & 6 \\ -2 & -1 & -3 \end{bmatrix}$$

should be:

$$\begin{bmatrix} 2 & 2 & 6 \\ 10 & 2 & 6 \\ -2 & -2 & -6 \end{bmatrix}$$

Hint : First try this for a specific matrix on the Command Line.

```
 tmpFn <- function(mat)
{
   mat[mat%%2==1] <- 2 * mat[mat%%2==1]
   mat
 }

# can test function with matA
```

```r
matA <- matrix(c(1, 5, -2, 1, 2, -1, 3, 6,-3), nrow = 3, ncol = 3)
matA
```

```
##      [,1] [,2] [,3]
## [1,]    1    1    3
## [2,]    5    2    6
## [3,]   -2   -1   -3
```

```r
tmpFn(matA)
```

```
##      [,1] [,2] [,3]
## [1,]    2    2    6
## [2,]   10    2    6
## [3,]   -2   -2   -6
```

5. Write a function which takes 2 arguements n and k which are positive integers. It should return the n x n matrix:

$$\begin{bmatrix} k & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & k & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & k & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & k & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & \cdots & k & 1 \\ 0 & 0 & 0 & 0 & \cdots & 1 & k \end{bmatrix}$$

Hint : First try to do it for a specific case such as $n = 5$ and $k = 2$ on the Command Line.

```r
tmpFn <- function(n, k)

  {
tmp <- diag(k, nrow = n, ncol = n)
tmp[abs(row(tmp) - col(tmp)) == 1] <- 1
tmp
}
tmpFn(5, 2)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    2    1    0    0    0
## [2,]    1    2    1    0    0
## [3,]    0    1    2    1    0
## [4,]    0    0    1    2    1
## [5,]    0    0    0    1    2
```

6. Suppose an angle α is given as a positive real number of degrees.

If $0 \leq \alpha < 90$ then it is quadrant 1. If $90 \leq \alpha < 180$ then it is quadrant 2.
if $180 \leq \alpha < 270$ then it is quadrant 3. if $270 \leq \alpha < 360$ then it is quadrant 4.
if $360 \leq \alpha < 450$ then it is quadrant 1. And so on . . .

Write a function `quadrant(alpha)` which returns the quadrant of the angle α.

```r
quadrant <- function(alpha)
{
 1+(alpha%%360)%/%90
}
```

7.

(a) Zeller's congruence is the formula:

$$f = ([2.6m - 0.2] + k + y + [y/4] + [c/4] - 2c) \ mod7$$

where $[x]$ denotes the integer part of x; for example $[7.5] = 7$.

Zeller's congruence returns the day of the week f given:

k = the day of the month
y = the year in the century
c = the first 2 digits of the year (the century number)
m = the month number (where January is month 11 of the preceding year, February is month 12 of the preceding year, March is month 1, etc.)
For example, the date 21/07/1963 has $m = 5, k = 21, c = 19, y = 63$; whilst the date 21/2/63 has $m = 12, k = 21, c = 19$, and $y = 62$.
Write a function `weekday(day,month,year)` which returns the day of the week when given the numerical inputs of the day, month and year.
Note that the value of 1 for f denotes Sunday, 2 denotes Monday, etc.

```
weekday <- function(day, month, year){
    if (month-2<=0){
    month <- month-2 +12
    year <- year - 1
    }
   else{
   month <- month -2
 }

  c <- year%/%100

  y <- year%%100

  result <- 1+((2.6*month-0.2)+day+y+(y%/%4)+(c%/%4)-2*c)%%7
  c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday")[result]

}
```

(b) Does your function work if the input parameters day, month, and year are vectors with the same length and valid entries?

No, the function doesn't work if the input parameters are vectors because my function has an if statement which can't be executed for vectors.

8.

(a) Suppose $x_0 = 1 \ and \ x_1 = 2$ and

$$x_j = x_{j-1} + \frac{2}{x_{j-1}} for \ j = 1, 2, ...$$

Write a function testLoop which takes the single argument n and returns the first $n-1$ values of the sequence $x_{j_{j\geq 0}}$ that means the values of $x_0, x_1, x_2, ..., x_{n-2}$.

```
testLoop <- function(n)
{
xVec <- rep(NA, n-1)
xVec[1] <- 1
xVec[2] <- 2
for( j in 3:(n-1) )
xVec[j] <- xVec[j-1] + 2/xVec[j-1]
xVec
}
```

(b) Now write a function testLoop2 which takes a single argument yVec which is a vector. The function should return

$$\sum_{j=1}^{n} e^j$$

where n is the length of yVec.

```
testLoop2 <- function(yVec)
{
n <- length(yVec)
sum( exp(seq(along=yVec)) )
}
```

9. Solution of the difference equation

$x_n = rx_{n-1}(1 - x_{n-1}),\ \textit{with starting value} x_1.$

(a) Write a function quadmap(start, rho, niter) which returns the vector $(X_1, ..., x_n)$ where $x_k = rx_{k-1}(1 - x_{k-1})$ and niter denotes n, start denotes x_1, and rho denotes r. Try out the function you have written: • for $r = 2$ and $0 < x_1 < 1$ you should get $x_n \to 0.5$ *as* $n \to \infty$. • try tmp <- quadmap(start=0.95, rho=2.99, niter=500) Now switch back to the Commands window and type: plot(tmp, type="l") Also try the plot plot(tmp[300:500], type="l")

```
quadmap <- function(start, rho, niter)
{
xVec <- rep(NA,niter)
xVec[1] <- start
for(i in 1:(niter-1)) {
xVec[i + 1] <- rho * xVec[i] * (1 - xVec[i])
}
x
}
```

(b) Now write a function which determines the number of iterations needed to get $\mid x_n - x_{n-1} \mid < 0.02$. So this function has only 2 arguments: start and rho. (For start=0.95 and rho=2.99, the answer is 84.)

```
quad2 <- function(start, rho, eps = 0.02)
{
x1 <- start
x2 <- rho*x1*(1 - x1)
niter <- 1
while(abs(x1 - x2) >= eps) {
x1 <- x2
x2 <- rho*x1*(1 - x1)
niter <- niter + 1
}
niter
}
```

10. (a) Given a vector $(x_1, ..., x_n)$, the sample autocorrelation of lag k is defined to be

$$r_k = \frac{\sum_{i=k+1}^{n}(x_i - \overline{x})(x_{i-k} - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

Thus

$$r_1 = \frac{\sum_{i=2}^{n}(x_i - \overline{x})(x_{i-1} - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2} = \frac{(x_2 - \overline{x})(x_1 - \overline{x}) + \cdots + (x_n - \overline{x})(x_{n-1} - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

Write a function tmpFn(xVec) which takes a single argument xVec which is a vector and returns a list of two values: $r_1 and\ r_2$. In particular, find $r_1 and r_2$ for the vector (2, 5, 8, . . . , 53, 56).

```
tmpFn <- function(xVec)
{
xc <- xVec - mean(xVec)
denom <- sum(xc^2)
n <- length(x)
r1 <- sum( xc[2:n] * xc[1:(n-1)] )/denom
r2 <- sum( xc[3:n] * xc[1:(n-2)] )/denom
list(r1 = r1, r2 = r2)
}
```

(b) (Harder.) Generalise the function so that it takes two arguments: the vector xVec and an integer k which lies between 1 and $n - 1$ where n is the length of $xVec$. The function should return a vector of the values ($r_0 = 1, r_1, ..., r_k$). If you used a loop to answer part (b), then you need to be aware that much, much better solutions are possible—see exercises 4. (Hint: *sapply*.)

```
tmpnFn <- function(x, k)
{
xc <- x - mean(x)
denom <- sum(xc^2)
n <- length(x)
tmpFn <- function(j){ sum( xc[(j+1):n] * xc[1:(n-j)] )/denom }
c(1, sapply(1:k, tmpFn))}
```

}