Assignment1

Need to import libraries for the assignment.

```
library(matlib)
library(expm)

## Loading required package: Matrix

##
## Attaching package: 'expm'

## The following object is masked from 'package:Matrix':
##
##     expm
```

Question 1:

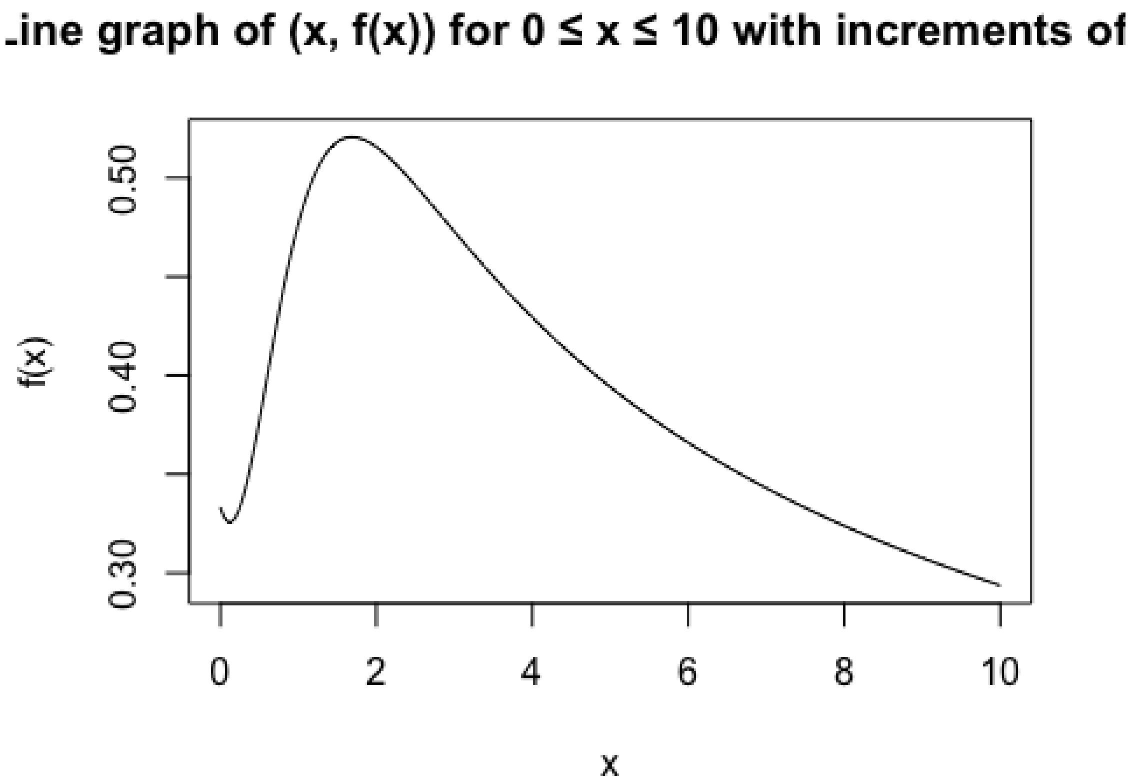
```
x <- seq(0, 10, by=.01)

f <- function(x)
{
  return(sqrt(((x^3)+(3*(x^2))+1)/((x^4)+(5*(x^3))+7*(x)+9)))
}

PlotF <- function(x)
{
  plot(
    x,f(x),
    main = "Line graph of (x, f(x)) for 0 ≤ x ≤ 10 with increments of 0.01",
    type="l")
}
```

(a) Draw a line graph of (x, f (x)) for $0 \leq x \leq 10$ with increments of 0.01.

```
PlotF(x)
```



(b) Find numerically the maximum value of f(x) and the maximizer x (report x to the second decimal place. For instance, x = 1.23).

```
fValues <- c(f(x))
maxFValue = max(fValues)
maximizer = x[which.max(fValues)]
print(paste("Max Value: ",maxFValue))

## [1] "Max Value:  0.520562546350245"

print(paste("Maximizer: ",round(maximizer,digits = 2)))

## [1] "Maximizer:  1.69"
```

Question 2:

```
set.seed(1)
x <- rnorm(1000)
```

(a) Calculate a standard deviation of the 1,000 numbers.

```
s <- sd(x)
print(paste("Standard Deviation: ",s))

## [1] "Standard Deviation:  1.0349158397994"
```

(b) Find the 100-th smallest number out of the 1,000 observations.

```
oneHundreth <- sort(x)[100]
print(paste("100-th smallest number: ",oneHundreth))

## [1] "100-th smallest number:  -1.34413012337621"
```

(c) The result in (b) is approximately the 10th percentile of the data. Is it roughly consistent with a normal probability table? See https://www.math.arizona.edu/~rsims/ma464/standardnormaltable.pdf, and explain briefly (no R-coding necessary).

```
z <- (oneHundreth-mean(x))/s
print("Yes, the z-score is -1.31 after rounding, which gives an area of
.09510 which falls in the 10th percentile")

## [1] "Yes, the z-score is -1.31 after rounding, which gives an area of
.09510 which falls in the 10th percentile"
```

Question 3:

```
A <- cbind(c(.979,.147),c(.144,-.999))
```

(a) Calculate A^2 (a matrix product).

```
print("A^2: ")

## [1] "A^2: "

print(A%^%2)

##           [,1]      [,2]
## [1,]  0.979609 -0.002880
## [2,] -0.002940  1.019169
```

(b) Calculate A^8, A^32 and A^1024 (Hint: A4 = A^2*A^2).

```
print("A^8: ")

## [1] "A^8: "

print(A%^%8)

##             [,1]       [,2]
## [1,]  0.92094709 -0.0115035
## [2,] -0.01174315  1.0789604

cat("\n")

print("A^32: ")

## [1] "A^32: "

print(A%^%32)
```

```
##               [,1]        [,2]
## [1,]  0.72011695 -0.04630104
## [2,] -0.04726564  1.35611313
```

```r
cat("\n")
```

```r
print("A^1024: ")
```

```
## [1] "A^1024: "
```

```r
print(A%^%1024)
```

```
##              [,1]      [,2]
## [1,]    98.7977 -1336.554
## [2,] -1364.3989 18457.853
```

(c) Calculate A^1000 (Hint: 1000 = 1024 – 32 + 8).

```r
print("A^1000: ")
```

```
## [1] "A^1000: "
```

```r
print(A%^%1000)
```

```
##               [,1]      [,2]
## [1,]    78.47005 -1061.558
## [2,] -1083.67339 14660.143
```

(d) Obtain the eigenvalues and eigenvectors of A.

```r
e <- eigen(A)
print("Eigenvalues:")
```

```
## [1] "Eigenvalues:"
```

```r
print(e$values)
```

```
## [1] -1.0096444  0.9896444
```

```r
cat("\n")
```

```r
print("Eigenvector:")
```

```
## [1] "Eigenvector:"
```

```r
print(e$vectors)
```

```
##               [,1]       [,2]
## [1,] -0.07222204 0.99727908
## [2,]  0.99738858 0.07371857
```

(e) Calculate A^1000 by using the result in (d).

```r
D <- cbind(c(e$values[1],0),c(0,e$values[2]))
P <- cbind(c(e$vectors[1,1],e$vectors[1,2]),c(e$vectors[2,1],e$vectors[2,2]))
inverseP <- inv(P)
```

```r
PDP <- P%*%(D^1000)%*%inverseP
print("A^1000: ")
```

```
## [1] "A^1000: "
```

```r
print(PDP)
```

```
##             [,1]      [,2]
## [1,]    78.47005 -1061.674
## [2,] -1083.55437 14660.143
```