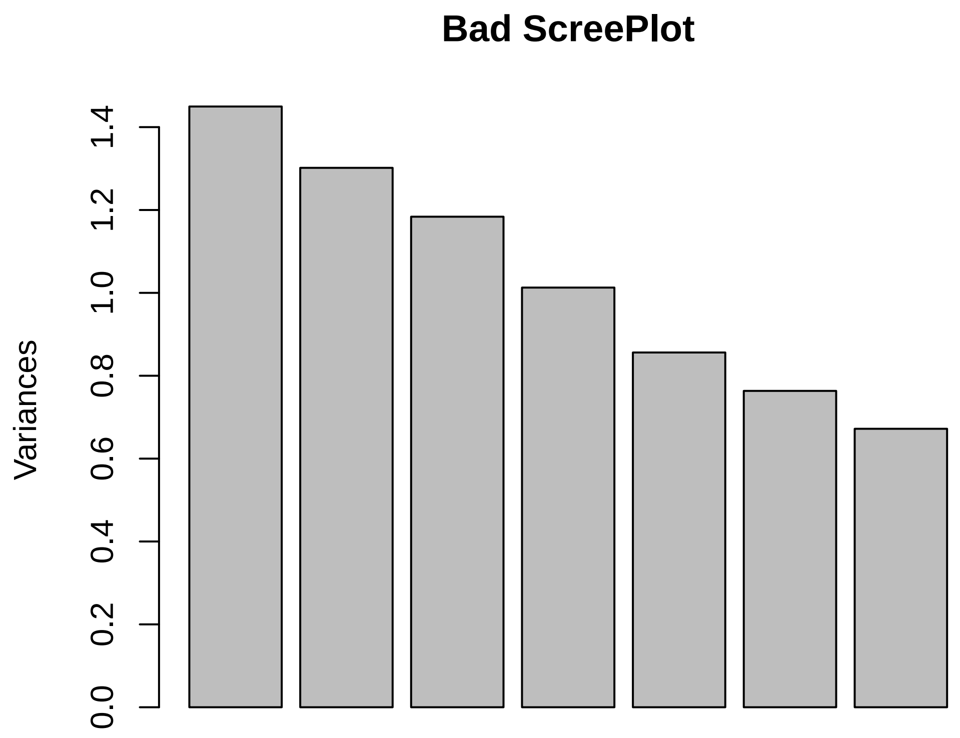
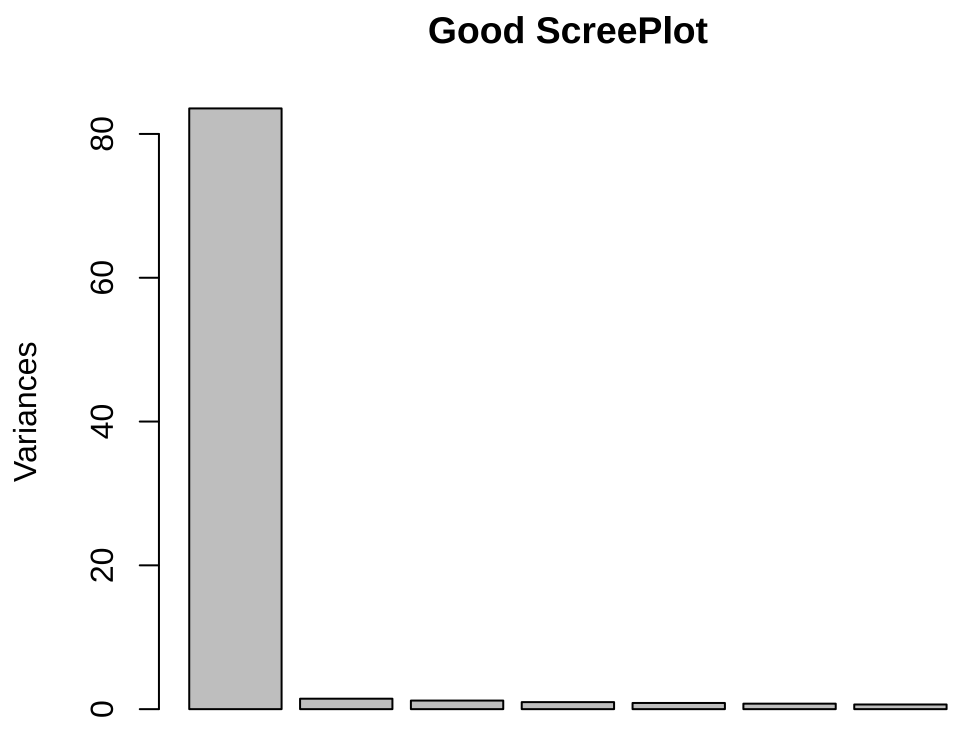
Problem 4

```r
library(MASS)
set.seed(20)
X <- data.frame(rnorm(100), rnorm(100), rnorm(100), rnorm(100), rnorm(100),
    rnorm(100), rnorm(100))
prin_X <- prcomp(X)
screeplot(prin_X, main = "Bad ScreePlot")
```



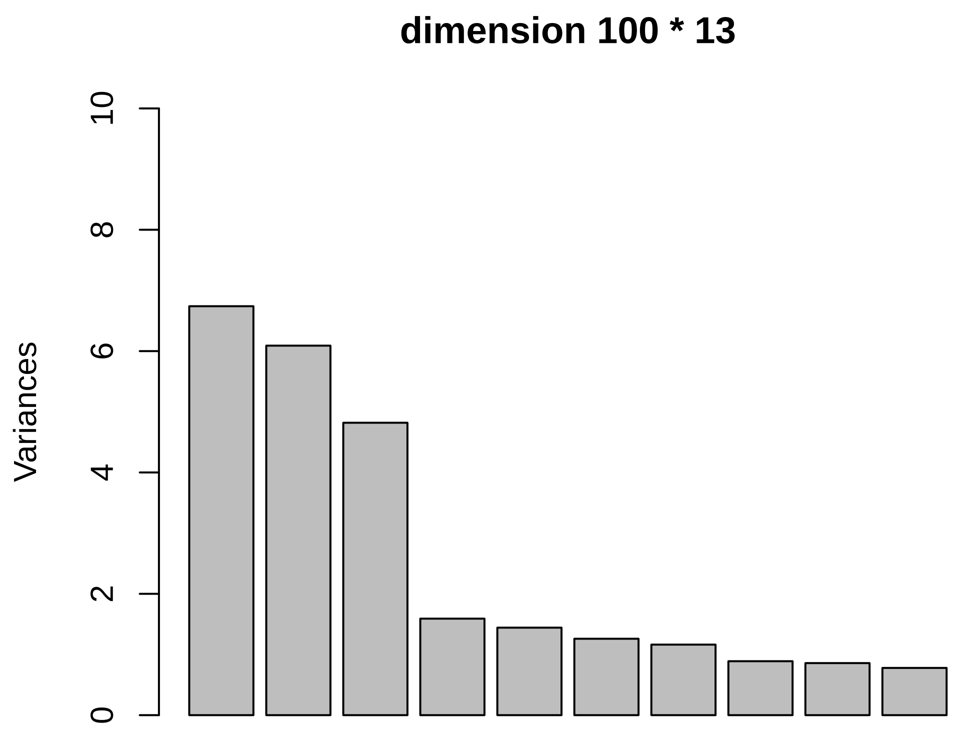
```r
# Comment: This is a bad screeplot since all of the principla components
# have similar variance. Thus, we can not use some of the principal
# compents to compress the data through projection.We need to use all of
# the principal components since each of them contain similar amount of
# information.
Y <- data.frame(rnorm(100), rnorm(100), rnorm(100), rnorm(100), rnorm(100),
    rnorm(100), rnorm(100, sd = 10))
prin_Y <- prcomp(Y)
screeplot(prin_Y, main = "Good ScreePlot")
```



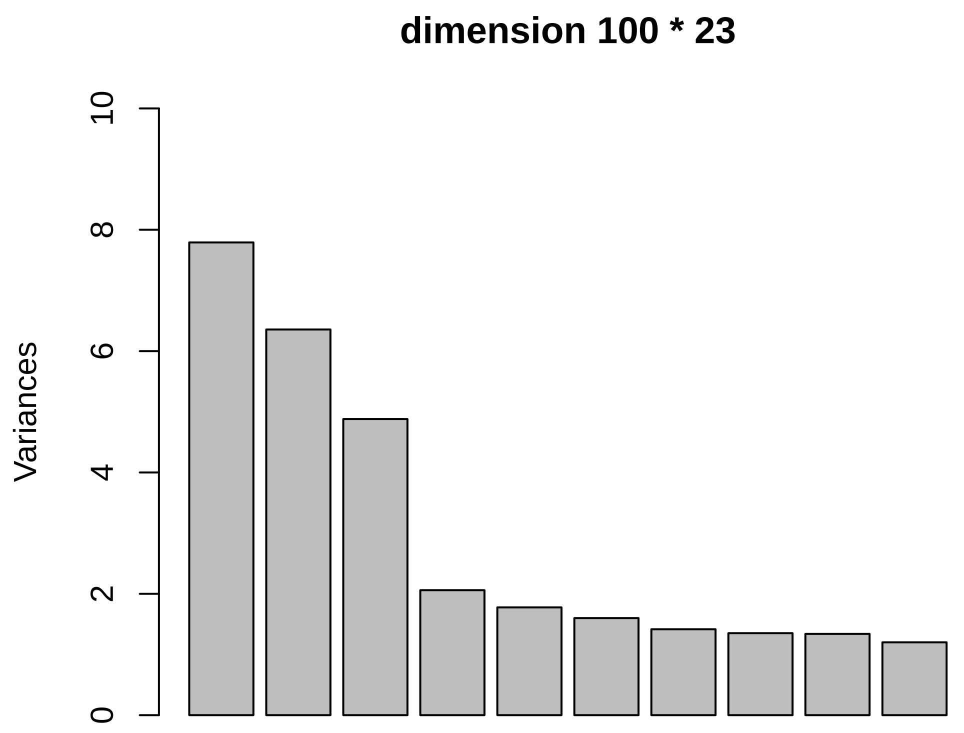
```
# Comment: This is a good screeplot since the first principal component
# containa a large amount of information. We can use it to compress data
# and project point based on first principal component.
```

Problem 5

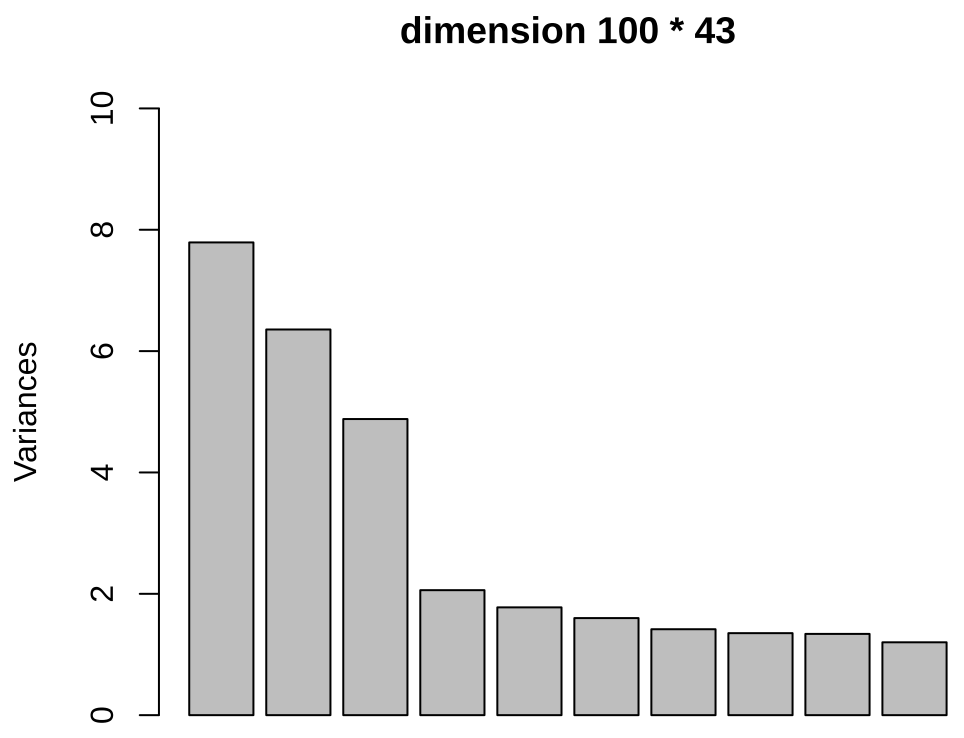
```
set.seed(250)
simulation <- cbind(matrix(rnorm(1000), nrow = 100), matrix(rnorm(300, sd = 2.5),
    nrow = 100))
screeplot(prcomp(simulation), main = "dimension 100 * 13", ylim = c(0, 10))
```



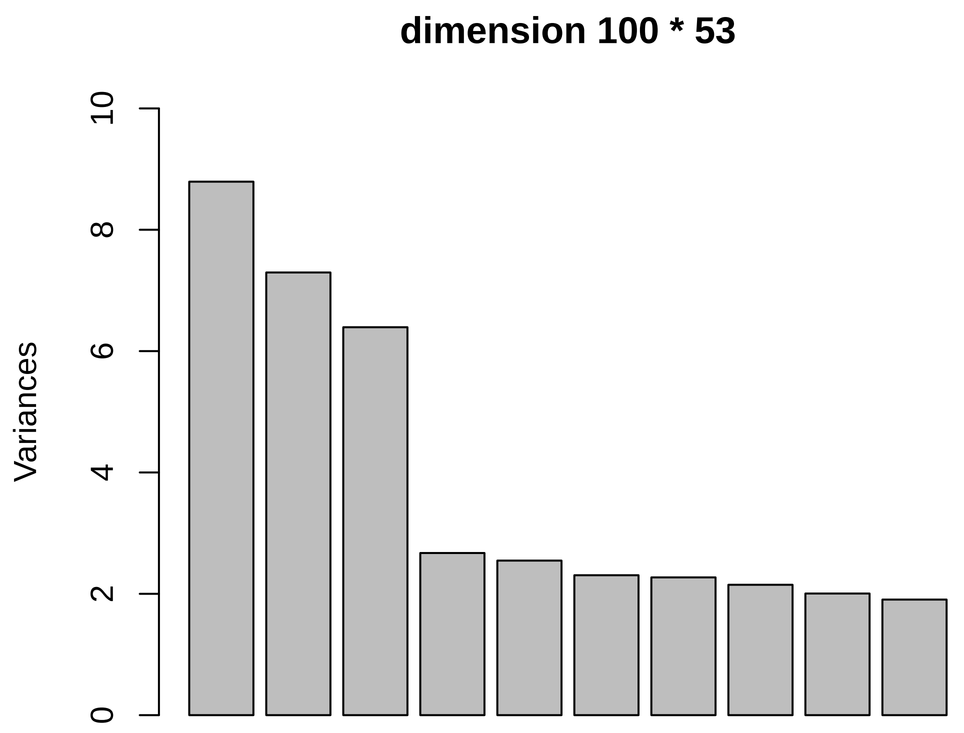
```r
simulation_1 <- cbind(matrix(rnorm(2000), nrow = 100), matrix(rnorm(300,
    sd = 2.5), nrow = 100))
screeplot(prcomp(simulation_1), main = "dimension 100 * 23", ylim = c(0,
    10))
```



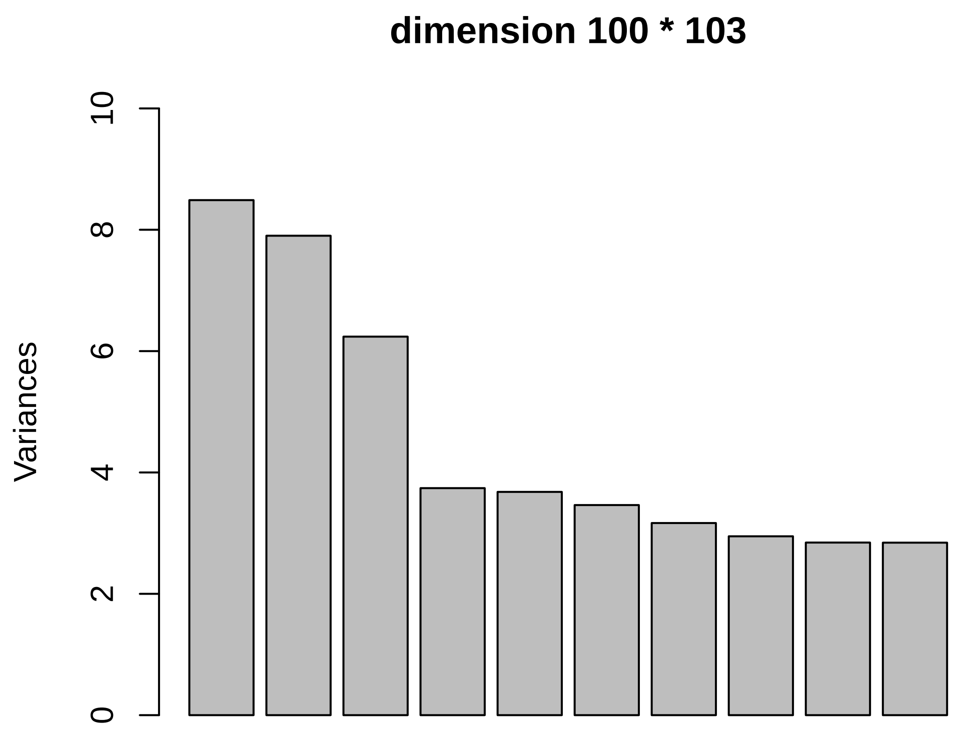
```
simulation_2 <- cbind(matrix(rnorm(4000), nrow = 100), matrix(rnorm(300,
    sd = 2.5), nrow = 100))
screeplot(prcomp(simulation_1), main = "dimension 100 * 43", ylim = c(0,
    10))
```



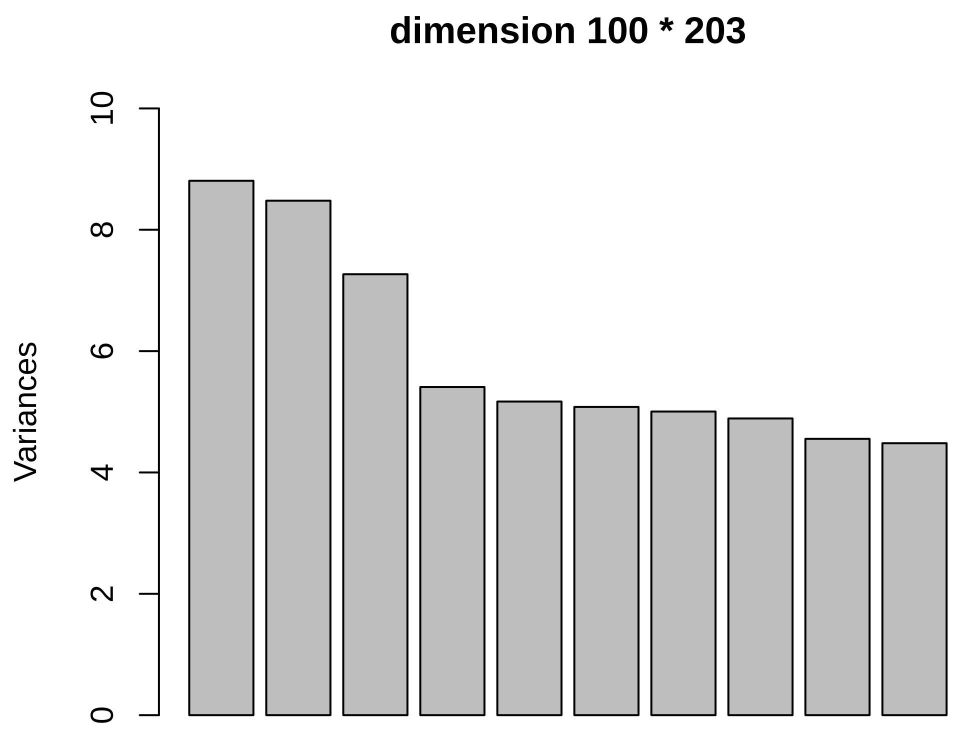
```
simulation_3 <- cbind(matrix(rnorm(5000), nrow = 100), matrix(rnorm(300,
    sd = 2.5), nrow = 100))
screeplot(prcomp(simulation_3), main = "dimension 100 * 53", ylim = c(0,
    10))
```



```
simulation_4 <- cbind(matrix(rnorm(10000), nrow = 100), matrix(rnorm(300,
    sd = 2.5), nrow = 100))
screeplot(prcomp(simulation_4), main = "dimension 100 * 103", ylim = c(0,
    10))
```



```
simulation_5 <- cbind(matrix(rnorm(20000), nrow = 100), matrix(rnorm(300,
    sd = 2.5), nrow = 100))
screeplot(prcomp(simulation_5), main = "dimension 100 * 203", ylim = c(0,
    10))
```



Comment: The matrix is constructed so that there are three columns with obvious variability and the other columns have less variability. So we will expect to have three principal components contributing more variability. However, as we increase the number of columns with less varied data, it turns out that the screeplot will explain less variability, which means that increasing dimension would make the variability more ambient data less interpretable using PCA.

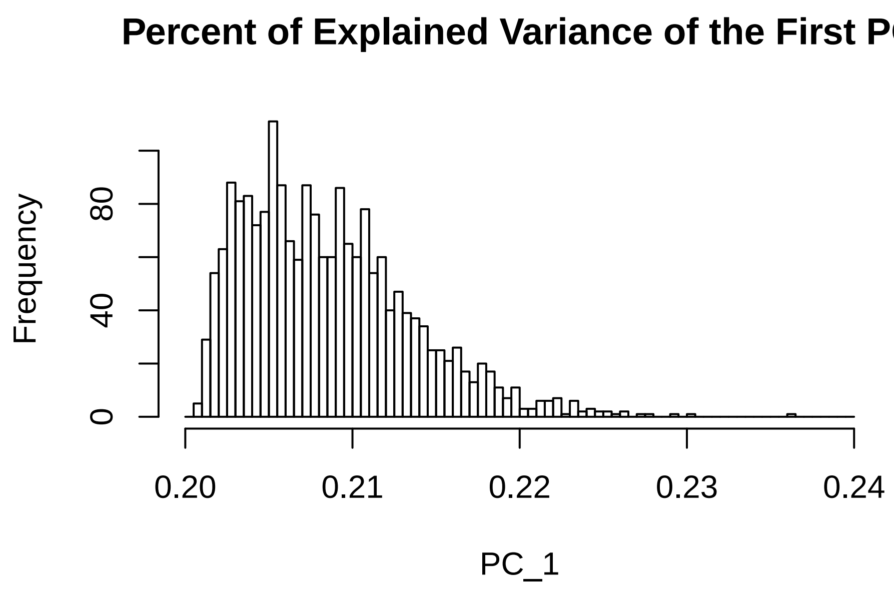
Problem 6

```
set.seed(100)
A <- matrix(0, nrow = 20, ncol = 20)
eigenvalue <- c(4, 4, 4, 4, rep(1.025, 16))
diag(A) <- eigenvalue
# Now we need to construct an orthogonal matrix, I did this by doing svd
# decomposition to a 8x8 random matrix a extract the U matrix.
f <- function() {
    U <- svd(matrix(rnorm(400), 20))$u
    # Then I can construct a Z matrix
    Z <- U %*% A %*% t(U)
    prin_Z <- prcomp(Z)
    explain_var_Z <- sapply(1:20, function(i) sum(prin_Z$sdev[1:i]^2)/sum(prin_Z$sdev^2))
    return(head(explain_var_Z, 4))
}
```

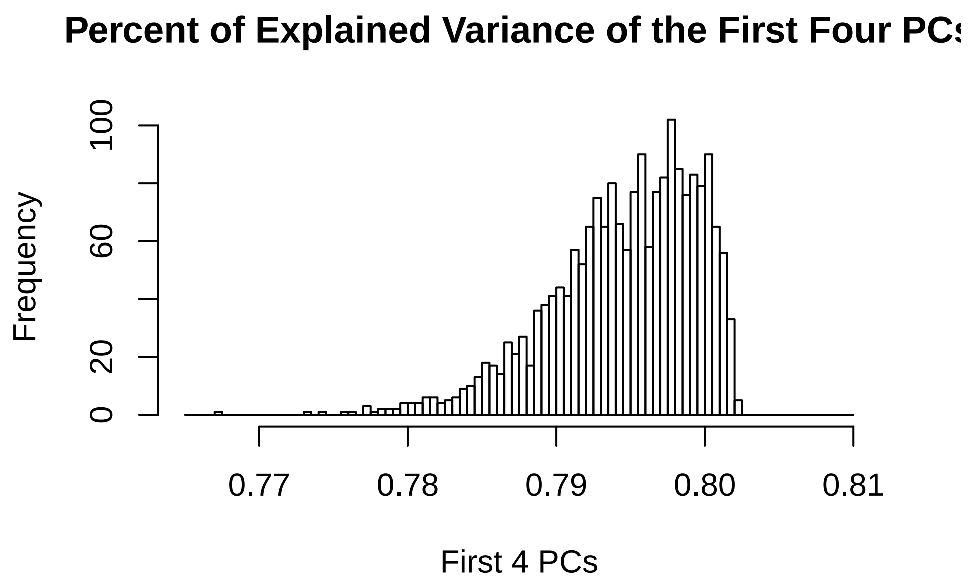
```
Rep <- replicate(2000, f())
rowMeans(Rep)

## [1] 0.2085 0.4171 0.6256 0.7946

hist(Rep[1, ], breaks = seq(0.2, 0.24, by = 5e-04), main = "Percent of Explained Variance of the First P
    xlab = "PC_1")
```



```
hist(Rep[4, ], breaks = seq(0.765, 0.81, by = 5e-04), main = "Percent of Explained Variance of the First
    xlab = "First 4 PCs")
```



Comment: I did this problem by constructing a diagnal matrix A with first four diagnal elements as 2 and the rest are smaller, then pretend it is the eigen matrix of some decomposed matrix using svd. Then, I construct the "original" matrix by multiplying orthogonal matrix to A. The method is the same as doing SVD decomposition backwards. Then, I replicate the function 2000 times to get the average of the explanation first four PCs. By the output, we have four three eigen vector explaining raughly 20 percent variability each, and the rest explain 20 percent variability.
In terms of stability; by first histogram, it seems that the the first principal component tends to explain more than 20 percent of variance(heavy righ tail) although it should explain exactly the same amount of variance as the other three by construction; however, the total variance explained by the top four PCs will not exceed 80 percent but it tends to explain less than 80 percent(heavy left tail in the histogram).

Problem 7

```
# For N(0,1) distribution
iqr <- qnorm(0.75) - qnorm(0.25)
# For N(miu, sd) distribution, IQR = sd*(qnorm(0.75) - qnorm(0.25)), I
# use the following code to verify.
mu <- 10
sigma <- 5
iqr_new <- qnorm(0.75, mean = mu, sd = sigma) - qnorm(0.25, mean = 10, sd = sigma)
all.equal(iqr_new, sigma * iqr)

## [1] TRUE

# For N(0,1)
fraction_outlier <- 1 - pnorm(qnorm(0.75) + 1.5 * iqr) + pnorm(qnorm(0.25) -
    1.5 * iqr)
```

(1)

For$N$(0,1) distribution, the IQR of is 1.349

(2)

For $N(\mu,\sigma^2)$ distribution, the IQR is going to be σ*(Q_3-Q_1). In order to check it, we have use N(10,5) to verify. It turns out its IQR is 6.7449, which is the raughly the same as 5 * 1.349

(3)

The fraction of outlier for N(0,1) is roughly 0.007

The rest of problem is attached as a hand written page