DaigleInClassLabWk9D2.R

2011home

Fri Mar 23 10:03:36 2018

```r
## Chris Daigle
## Wk9D2 Inclass Lab

# Exercise 1 #################################################################
# Calculate the PageRank using the following matrix.
m1 <- c(0,1/3,1/3,1/3)
m2 <- c(0,0,1/2,1/2)
m3 <- c(1,0,0,0)
m4 <- c(1/2,0,1/2,0)

M <- cbind(m1,m2,m3,m4)

eigenM <- eigen(M)
tempEigenVec <- round(eigenM$vectors[,1], digits = 3)
eigenVec <- Re(tempEigenVec)
normedEigenVec <- eigenVec / sum(eigenVec)
normedEigenVec
```

```
## [1] 0.3870102 0.1288245 0.2903918 0.1937735
```

```r
normedEigenVec[1]
```

```
## [1] 0.3870102
```

```r
cat("The page rank for this matrix is:")
```

```
## The page rank for this matrix is:
```

```r
cat("page 1 =", normedEigenVec[1])
```

```
## page 1 = 0.3870102
```

```r
cat("page 2 =" , normedEigenVec[2])
```

```
## page 2 = 0.1288245
```

```r
cat("page 3 = ", normedEigenVec[3])
```

```
## page 3 =  0.2903918
```

```r
cat("page 4 = ", normedEigenVec[4])
```

```
## page 4 =  0.1937735
```

```r
# Exercise 2 #################################################################
# Write functions to implement PageRank algorithm.
# Using these functions, calculate the PageRank using a much bigger matrix
# Which one is more efficient between the iteration method and eigenvalue approach?

# Power iteration
p_1 <- c(0,0,0,0)
p <- cbind(p_1, c(0,0,1,0)) # null vector, p1 and initial position, c(0,0,1,0)
```

```
T <- 2
while (sum(abs(p[,T]-p[,T-1]))>0.001) {
  p <- cbind(p, c(0,0,0,0))
  p[,T+1] <- M%*%p[,T]
  T <- T+1
}
p[,T]
```

```
## [1] 0.3871287 0.1290831 0.2902681 0.1935201
```

```
pr1 <- function(M) {
  p_1 <- rep(0,nrow(M))
  p <- cbind(p_1, c(1,rep(0,nrow(M)-1)))
  T <- 2
  while (sum(abs(p[,T]-p[,T-1]))>0.001) {
    p <- cbind(p, rep(0,nrow(M)))
    p[,T+1] <- M%*%p[,T]
    T <- T+1
  }
  plot(p[,T], type="h")
  return(p[,T])
}

# Eigenvalue
eigen.M <- eigen(M)

vec.temp <- eigen.M$vectors[,1]
vec.temp
```

```
## [1] 0.7210101+0i 0.2403367+0i 0.5407576+0i 0.3605051+0i
```

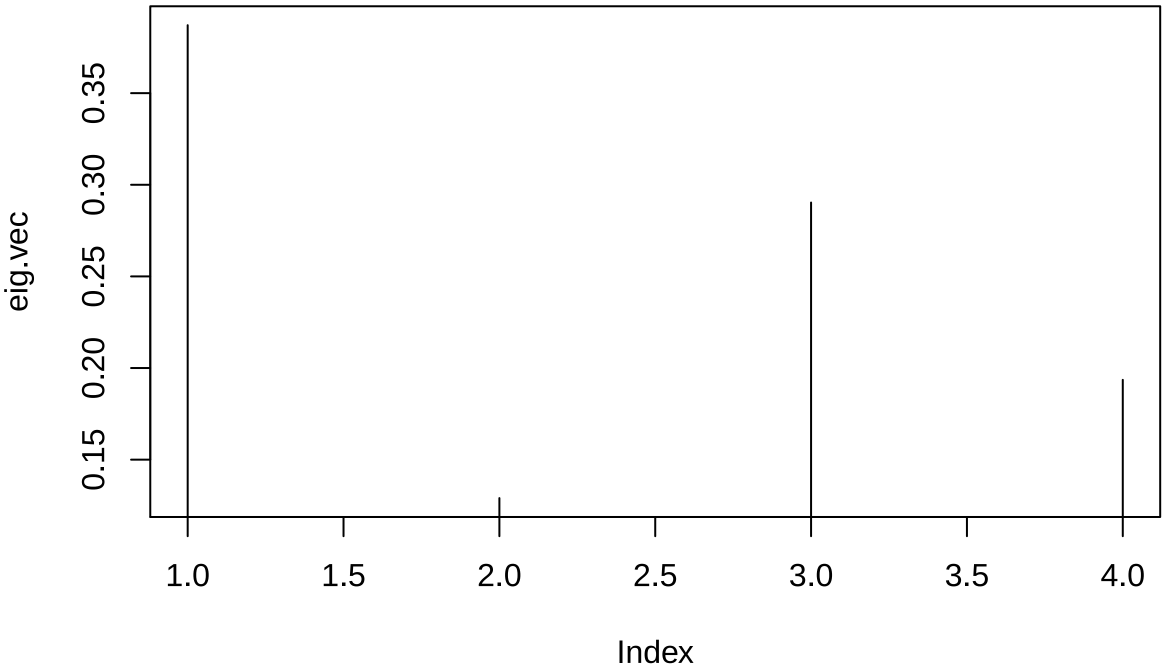
```
vec.temp <- Re(vec.temp)
vec.temp
```

```
## [1] 0.7210101 0.2403367 0.5407576 0.3605051
```

```
eig.vec <- vec.temp/sum(vec.temp)
eig.vec
```

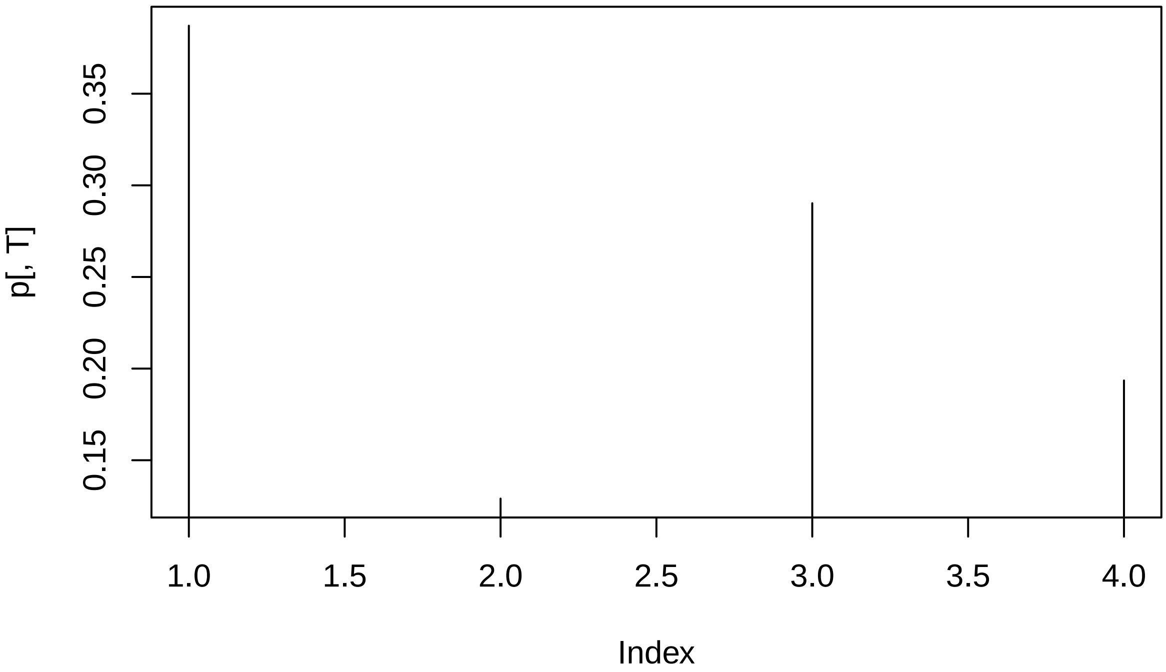
```
## [1] 0.3870968 0.1290323 0.2903226 0.1935484
```

```
pr2 <- function(M) {
  eigen.M <- eigen(M)

  vec.temp <- eigen.M$vectors[,1]
  vec.temp

  vec.temp <- Re(vec.temp)

  eig.vec <- vec.temp/sum(vec.temp)
  plot(eig.vec, type="h")
  return(eig.vec)
}
pr2(M)
```

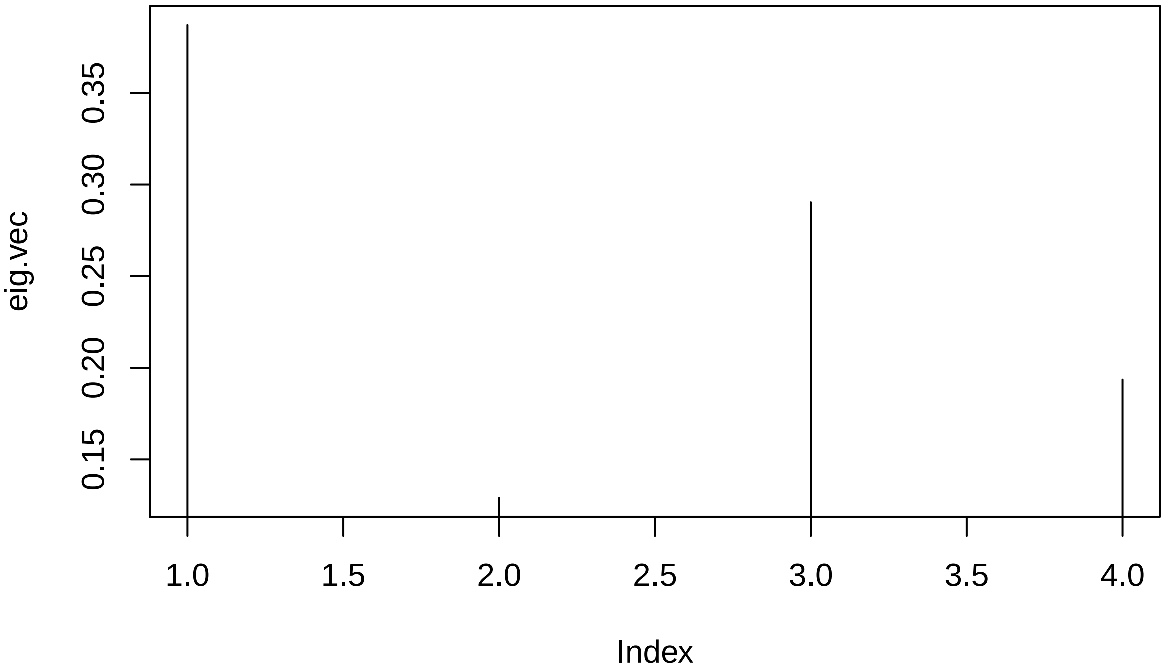


```
## [1] 0.3870968 0.1290323 0.2903226 0.1935484
```

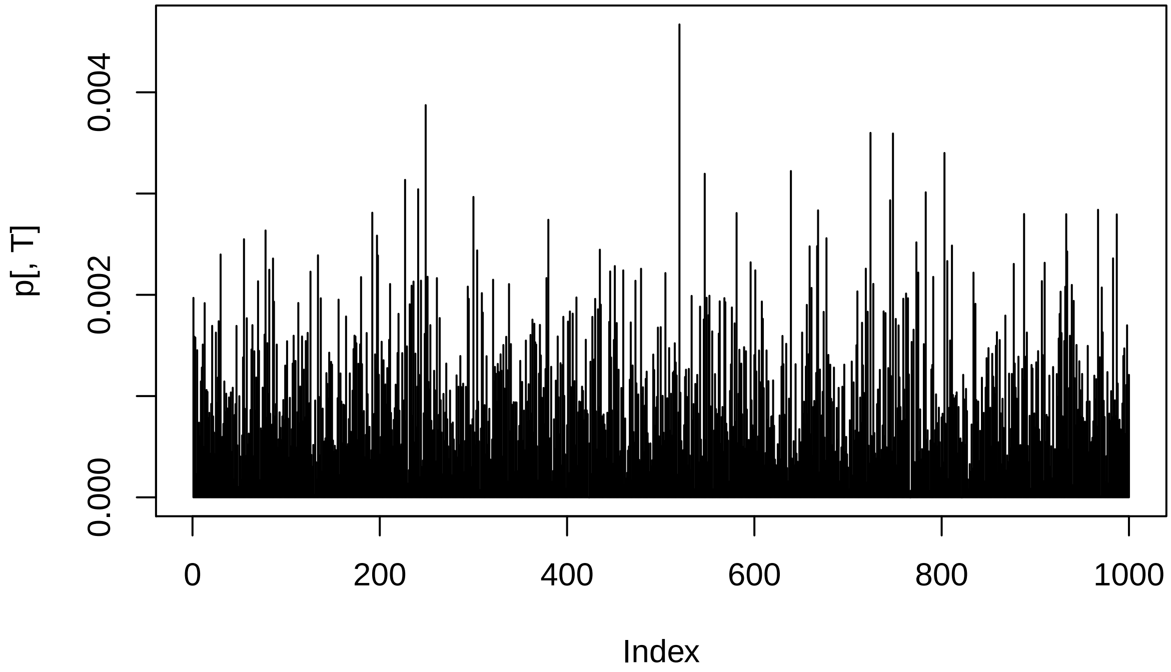
```
pr.1 <- pr1(M)
```



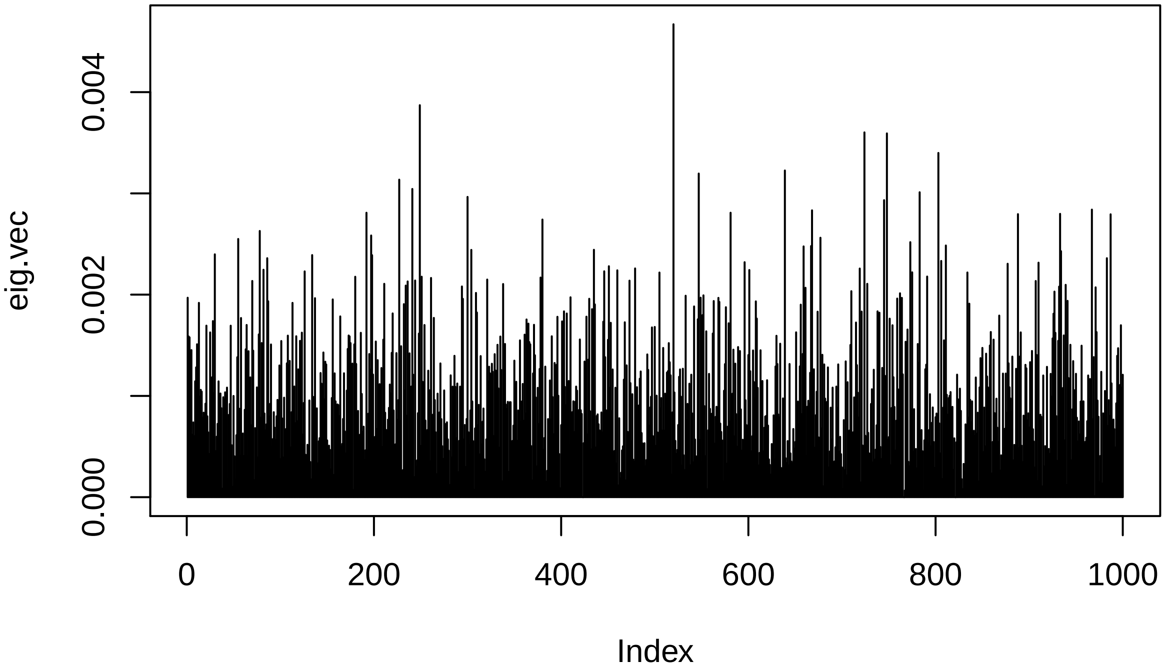
```
pr.2 <- pr2(M)
```



```
ind <- 1:1000
M <- matrix(rep(0,1000^2), nrow = 1000, ncol = 1000)
for (i in 1:1000) {
  a <- sample(x = 1:10, size = 1)
  b <- sample(x = 1:1000, size = a)
  M[b, i] = 1/a
}

cost1 <- system.time(
  pr1(M)
)
```



```r
cost2 <- system.time(
  pr2(M)
)
```



```r
cost1
```

```
##    user  system elapsed
##   0.039   0.001   0.041
```

```r
cost2
```

```
##    user  system elapsed
##   7.216   0.083   7.596
```

```r
effic <- cost1[2] < cost2[2]

cat("The power iteration method is more efficient: ", effic)
```

```
## The power iteration method is more efficient:  TRUE
```

```r
# print("I believe this is incorrect as any matrix larger than the one given halts the while")
# print("loop, but eigenvalue method is more efficient when I \'force it'\ in a 100x100")
# print("without publishing")

# ind <- 1:1000
# M <- matrix(rep(0,1000^2), nrow = 1000, ncol = 1000)
# for (i in 1:1000) {
#   a <- sample(x = 1:10, size = 1)
#   b <- sample(x = 1:1000, size = a)
#   M[b, i] = 1/a
# }
#
# cost1 <- system.time(
#   pr1(M)
# )
```

```r
#
# cost2 <- system.time(
#   pr2(M)
# )

# M <- matrix(rnorm(1000), 100, 100)
# pr.1 <- pr1(M)
# pr.2 <- pr2(M)
#
# cost1 <- system.time(
#   pr1(M)
# )
#
# cost2 <- system.time(
#   pr2(M)
# )
#
# effic <- cost1[2] < cost2[2]
# cat("The power iteration method is more efficient: ", effic)
```