Lab 4

Hong Xu

September 21, 2015

Lab 4

Hong Xu

09/21/2015

Team member: Feilun Wu

```
load("pageRank.RData")
## 1. create a link count
# transpose first, make sure myData is in the (j -> i) format
myData <- t(myData)
n_links <- colSums(myData)
M <- diag(n_links)

## 2. inverse the link matrix
MInv <- solve(M)

## 3. compute the BrokenRank
BR <- myData %*% MInv
BR_eig <- eigen(BR)
# find the eigenvectors with eigenvalue 1
BR_vector <- BR_eig$vectors[, which(signif(BR_eig$values, 6) == 1)]
# normalize
(BR_vector <- scale(BR_vector, center = FALSE,
                    scale = as.numeric(apply(as.matrix(BR_vector), 2, sum))))
```

```
##        [,1]        [,2]
## [1,] 0.0+0i 0.3333333+0i
## [2,] 0.0+0i 0.3333333+0i
## [3,] 0.0+0i 0.3333333+0i
## [4,] 0.5+0i 0.0000000+0i
## [5,] 0.5+0i 0.0000000+0i
## attr(,"scaled:scale")
## [1]  1.414214 -1.732051
```

```
## Problems: the resulting vectors are not UNIQUE,
## which means the BrokenRank vector is ambiguously defined.

## 4. compute PageRank
n <- nrow(BR)
E <- matrix(rep(1, n*n), n, n)
d <- 0.85
# make a strong connected Markov Chain
PR <- ((1 - d) / n) * E + d * myData %*% MInv
```

```
# the rest steps are the same as BrokenRank
PR_eig <- eigen(PR)
PR_vector <- PR_eig$vectors[, which(signif(PR_eig$values, 6) == 1)]
(PR_vector <- scale(PR_vector, center = FALSE, scale = as.numeric(sum(PR_vector))))
```

```
##        [,1]
## [1,] 0.2+0i
## [2,] 0.2+0i
## [3,] 0.2+0i
## [4,] 0.2+0i
## [5,] 0.2+0i
## attr(,"scaled:scale")
## [1] -2.236068
```

```
## Now we have a unique PageRank vector.
```

```
myNewData <- read.csv("http://www.ats.ucla.edu/stat/data/mat25.txt",
                      header = FALSE, sep = " ")
myPageRank <- function(link_matrix, d=0.85) {
  ##
  ## The function that calculate the PageRank vector
  ## for a given link matrix.
  ##
  ## Input:
  ## link_matrix, a matrix with links go i -> j
  ## d, numerical, the dampening parameter
  ##
  ## Output:
  ## p, the PageRank vector
  ##

  # transpose the link matrix so that we have j -> i
  link_matrix <- t(link_matrix)
  # compose the matrix to find the PageRank vector
  M <- diag(colSums(link_matrix))
  MInv <- solve(M)
  A <- link_matrix %*% MInv
  # build a strong conneted Markov Chain
  n <- nrow(A)
  E <- matrix(rep(1, n*n), n, n)
  A <- ((1 - d) / n) * E + d * A
  # find the eigenvector
  A_eig <- eigen(A)
  PR_vector <- A_eig$vectors[, which(signif(A_eig$values, 6) == 1)]
  # normalize
  PR_vector <- scale(PR_vector, center = FALSE,
                     scale = as.numeric(apply(as.matrix(PR_vector), 2, sum)))

  return(as.numeric(PR_vector))
}

myPageRank(as.matrix(myNewData), 0.85)
```

```
##  [1] 0.05096317 0.03545182 0.04341318 0.02855733 0.06138548 0.04926776
##  [7] 0.02557665 0.06735780 0.03449904 0.04142908 0.02761109 0.07174358
## [13] 0.03078005 0.04893792 0.02291655 0.02499144 0.00600000 0.02527793
## [19] 0.04136165 0.02794755 0.05808002 0.05580470 0.02626592 0.05023857
## [25] 0.04414171
```