Foundations of Computational Math – Final Exam Part 1

Sam Reeves

```r
library(dplyr)
library(ggplot2)
library(igraph)
library(stats)
```

Part 1. Playing with Pagerank

1. The A and B Matrices (5)

Form the A matrix. Then, introduce decay and form the B matrix as we did in the course notes.

$$B = 0.85 \times A + \frac{0.15}{n}$$

```r
n <- 6
A <- matrix(c(0, 1/2, 1/2, 0, 0, 0,
              0, 0, 0, 0, 0, 0,
              1/3, 1/3, 0, 0, 1/3, 0,
              0, 0, 0, 0, 1/2, 1/2,
              0, 0, 0, 1/2, 0, 1/2,
              0, 0, 0, 1, 0, 0),
            ncol = n, byrow = TRUE,
            dimnames = list(c(1:6), c(1:6)))

B <- 0.85 * A + 0.15 / n
```

2. The Rank Vector (5)

Start with a uniform rank vector r and perform power iterations on B till convergence. That is, compute the solution $r = B^n \times r$. Attempt this for a sufficiently large n so that r actually converges.

```r
r_i <- rep(1/6, 6)

power_iteration <- function(old_matrix) {
  # Performs one power iteration
  # Returns a new matrix of the same dimensions

  new_matrix <- old_matrix
    for (j in seq(1:nrow(old_matrix))) {
      for (k in seq(1:ncol(old_matrix))) {
      new_matrix[j, k] <- old_matrix[j,] %*% old_matrix[,k]
      }
    }
    return(new_matrix)
}

converge <- function(old_matrix, n = 1) {
  # Performs power iterations until values converge
  # Returns the number of iterations and the matrix

  new_matrix <- power_iteration(old_matrix)
  n <- n + 1

  if (identical(new_matrix, old_matrix)) {
    return(list(n, new_matrix))
  } else {
    converge(new_matrix, n)
  }
}

converge(B)
```

```
## [[1]]
## [1] 16
##
## [[2]]
##   1 2 3 4 5 6
## 1 0 0 0 0 0 0
## 2 0 0 0 0 0 0
## 3 0 0 0 0 0 0
## 4 0 0 0 0 0 0
## 5 0 0 0 0 0 0
## 6 0 0 0 0 0 0
```

3. Eigen-Decomposition (10)

Compute the eigen-decomposition of B and verify that you indeed get an eigenvalue of 1 as the largest eigenvalue and that its corresponding eigenvector is the same vector that you obtained in the previous power iteration method. Further, this eigenvector has all positive entries and it sums to 1.

```
eigen(B)
```

```
## eigen() decomposition
## $values
## [1]  0.95165271+0i -0.42500000+0i -0.42500000-0i  0.41436582+0i -0.34733611+0i
## [6] -0.01868242+0i
##
## $vectors
##               [,1]                        [,2]                        [,3]
## [1,] -0.2159123+0i -5.345225e-01-0.000000e+00i -5.345225e-01+0.000000e+00i
## [2,] -0.0572329+0i  0.000000e+00+1.832268e-10i  0.000000e+00-1.832268e-10i
## [3,] -0.2980792+0i  5.345225e-01+0.000000e+00i  5.345225e-01-0.000000e+00i
## [4,] -0.5358032+0i -2.672612e-01+0.000000e+00i -2.672612e-01-0.000000e+00i
## [5,] -0.5358032+0i -2.672612e-01-0.000000e+00i -2.672612e-01+0.000000e+00i
## [6,] -0.5358032+0i  5.345225e-01+0.000000e+00i  5.345225e-01+0.000000e+00i
##                [,4]            [,5]           [,6]
## [1,] -0.77869913+0i -0.775079691+0i  0.55060201+0i
## [2,] -0.07537631+0i  0.009090375+0i -0.61511270+0i
## [3,] -0.61034825+0i  0.631781588+0i  0.56386946+0i
## [4,]  0.07169632+0i  0.002637034+0i -0.01322899+0i
## [5,]  0.07169632+0i  0.002637034+0i -0.01322899+0i
## [6,]  0.07169632+0i  0.002637034+0i -0.01322899+0i
```

4. Graph Algorithm (10)

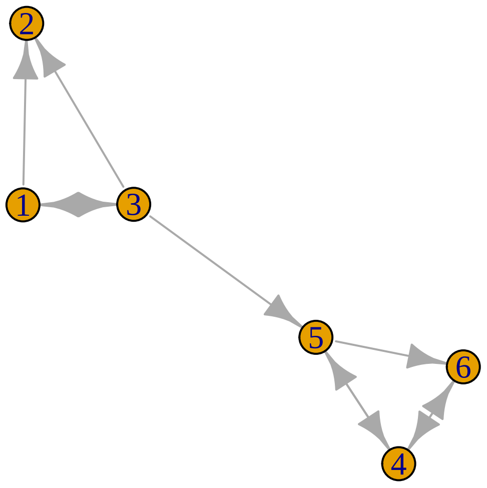
Use the *graph* package in R and its *page.rank* method to compute the PageRank of the graph as given in A. Note that you don't need to apply decay. The package starts with a connected graph and applies decay internally. Verify that you do get the same PageRank vector as the two approaches above.

```
edges <- matrix(c(1, 2, 1/2,
                  1, 3, 1/2,
                  3, 1, 1/3,
                  3, 2, 1/3,
                  3, 5, 1/3,
                  4, 5, 1/2,
                  4, 6, 1/2,
                  5, 4, 1/2,
                  5, 6, 1/2,
                  6, 4, 1),
                ncol = 3, byrow = TRUE)

edges_1 <- c(1,2,1,3,3,1,3,2,3,5,4,5,4,6,5,4,5,6,6,4)
g <- graph(edges_1)
```

```
g <- set_edge_attr(g, "w", value = edges[,3])
```

```
plot(g)
```



```
page.rank(g)
```

```
## $vector
## [1] 0.05170475 0.07367926 0.05741241 0.34870369 0.19990381 0.26859608
##
## $value
## [1] 1
##
## $options
## NULL
```