CFerrari_Assignment10

Charley Ferrari

Friday, April 03, 2015

Problem Set 1

You'll verify for yourself that PageRank works by performing calculations on a small universe of web pages.Let's use the 6 page universe that we had in the course notes. For this directed graph, perform the following calculations in R.

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

```
A <- matrix(c(0, 1/2, 1/2, 0, 0,  0,
              0, 0, 1, 0, 0, 0,
              1/4, 1/4, 0, 0, 1/4, 1/4,
              0, 0, 0, 0, 1/2, 1/2,
              0, 0, 0, 1/2, 0, 1/2,
              0, 0, 1/2, 1/2, 0, 0), nrow=6)

# Introduce decay d = 0.85

d <- 0.85

B <- 0.85*A + (0.15/6)
```

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

```
# Technically, this function will multiply A by B 'power' times
# But I'm going to use this to calculate powers.

matrixPower <- function(A, power, B){
  if(power == 1){
    return(A)
  }else{
    return(matrixPower(A %*% B, power - 1, B))
  }
}

r <- matrix(c(1/6,1/6,1/6,1/6,1/6,1/6),nrow=6)

k <- 500

rk <- matrixPower(B,k,B) %*% r

rk
```

```
##            [,1]
## [1,] 0.07735886
## [2,] 0.11023638
```

```
## [3,] 0.24639464
## [4,] 0.18635389
## [5,] 0.15655927
## [6,] 0.22309696
```

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)$values[1]
```

```
## [1] 1+0i
```

```
# Need to scale the Eigenvectors, I just divided the vectors to get my scale.

(rk / eigen(B)$vectors[,1]) * eigen(B)$vectors[,1]
```

```
##               [,1]
## [1,] 0.07735886+0i
## [2,] 0.11023638+0i
## [3,] 0.24639464+0i
## [4,] 0.18635389+0i
## [5,] 0.15655927+0i
## [6,] 0.22309696+0i
```

Use the graph package in R and its page.rank method to compute the Page Rank 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

```
library(igraph)


relations <- data.frame(from=c(1,1,2,3,3,3,3,4,4,5,5,6),
                        to = c(2,3,3,1,2,5,6,5,6,4,6,4),
                        weight = c(1,1,1,1,1,1,1,1,1,1,1,1))


g <- graph.data.frame(relations, directed=TRUE)

page.rank(g)
```

```
## $vector
##         1         2         3         4         5         6
## 0.04647041 0.06622033 0.10103720 0.33171134 0.18744772 0.26711301
##
## $value
## [1] 1
##
## $options
## NULL
```

```r
# Once again to scale:

(rk / page.rank(g)$vector) * page.rank(g)$vector
```

```
##           [,1]
## [1,] 0.07735886
## [2,] 0.11023638
## [3,] 0.24639464
## [4,] 0.18635389
## [5,] 0.15655927
## [6,] 0.22309696
```

```r
plot(g)
```