Homework 5 Solutions :: MATH 504

Your homework submission must be a single pdf called "LASTNAME-hw5.pdf" with your solutions to all theory problem to receive full credit. All answers must be typed in Latex.

1. Create an 5×5 matrix A using the command hilb(5) in Matlab, or scipy.linalg.hilbert(5) in Python. Generate a random vector x, and compute $b = Ax$. Add a tiny amount of noise to b, call it $\hat{b}$. Then recover $\hat{x}$ from $A\hat{x} = \hat{b}$.

   How accurate is the recovered solution? Why did this happen? You don't need to provide any code or console output, just describe what you did and what you got in a few sentence.

```
library(Matrix)
set.seed(504)
A<-Hilbert(5)
x<-matrix(1:5,ncol=1)
(b<-A%*%x)
```

```
## 5 x 1 Matrix of class "dgeMatrix"
##          [,1]
## [1,] 5.000000
## [2,] 3.550000
## [3,] 2.814286
## [4,] 2.346429
## [5,] 2.017460
```

```
set.seed(504)
epsilon<-matrix(rnorm(5, mean=0, sd=0.25), ncol = 1)
(b_hat<-b+epsilon)
```

```
## 5 x 1 Matrix of class "dgeMatrix"
##          [,1]
## [1,] 4.899501
## [2,] 4.349702
## [3,] 3.084476
## [4,] 2.385127
## [5,] 1.936173
```

```
(x_hat<-solve(A)%*%b_hat)
```

```
## 5 x 1 Matrix of class "dgeMatrix"
##             [,1]
## [1,]   -63.11198
## [2,]   828.55528
## [3,] -2929.10718
## [4,]  3970.60272
## [5,] -1812.73893
```

```
values<-eigen(A)$values
values<-sort(values, decreasing = TRUE)
paste("The condition number of the matrix A is", round(values[1]/values[5]),".", sep=" ")
```

```
## [1] "The condition number of the matrix A is 476607 ."
```

One immediately sees how a small change to b caused $\hat{x}$ to vary greatly from x. The reason of this phenomenon is that Hilbert matricies are ill-conditioned (as observed by the high condition number of 476607 above) and result in unstable systems. The realtive change in our example is (using l_2 norms) is calculated below:

```
norm2<-function(a){
  norm<-0
  for (i in 1:dim(a)[1]) {
    norm<-norm+(a[i,])^2
  }
  norm<-sqrt(norm)
  return(norm)
}
RelativeChange<-function(b, b_hat){
  dif<-b-b_hat
  change<-norm2(dif)/norm2(b)
  return(change)
}
deltaB<-RelativeChange(b,b_hat)
deltaX<-RelativeChange(x,x_hat)
paste("The relative change in b is", round(deltaB,4), ";meanwhile the relative change in x, due
 to change in b, is", round(deltaX,4), ".", sep=" ")
```

```
## [1] "The relative change in b is 0.1152 ;meanwhile the relative change in x, due to change in
b, is 717.6055 ."
```

2. (Coding) Construct any 3×3 invertible symmetric matrix with no entry equal to 0.

   a) Using the function **eig** in Matlab or equivalent in other programming languages to find the dominant eigenvalue $\lambda^*_{\max}$ and its corresponding eigenvector v^*.

   b) Use the Power Method to find the (approximate) dominant eigenvector $v^{(k)}$ and eigenvalue μ_k of this matrix for different stopping criteria

$$\frac{\|v^{(k)} - v^*\|_2}{\|v^*\|_2} \leq \epsilon$$

   Record these data in the following table for given different ϵ values.

ϵ	iteration	$\lvert\mu_k - \lambda^*_{\max}\rvert$	$\frac{\|v^{(k)}-v^*\|_2}{\|v^*\|_2}$	$\frac{\|v^{(k)}-v^{(k-1)}\|_2}{\|v^{(k-1)}\|_2}$
10^{-3}				
10^{-6}				
10^{-9}				

```r
#Define a symmetric matrix
A<-matrix(c(1:4, 5, 4:1), nrow=3, byrow = T)
A
```

```
##      [,1] [,2] [,3]
## [1,]    1    2    3
## [2,]    4    5    4
## [3,]    3    2    1
```

```r
#Check if the matrix is invertible
A%*%solve(A)
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

```r
library(Matrix)
ev<-eigen(A)
(values<-ev$values)
```

```
## [1]  8.5311289 -2.0000000  0.4688711
```

```r
(vectors<-ev$vectors)
```

```
##            [,1]          [,2]       [,3]
## [1,] -0.3744299 -7.071068e-01  0.4420619
## [2,] -0.8482951 -2.254475e-16 -0.7804887
## [3,] -0.3744299  7.071068e-01  0.4420619
```

```r
Vec<-matrix(nrow = 1, ncol=3)
domV<-matrix(ncol=1, nrow=3)
for(i in 1:3){
  Vec[1,i]<-as.character(as.vector(t(vectors[, which(values==max(abs(values)))]))[i])
}
for(i in 1:3){
  domV[i,1]<-t(vectors[, which(values==max(abs(values)))])[i]
}
domEig<-max(abs(values))
paste("The dominant eigenvalue is", max(abs(values)),".",sep=" ")
```

```
## [1] "The dominant eigenvalue is 8.53112887414927 ."
```

```r
paste("The corresponding eigenvector is ( ",
      round(as.numeric(Vec[1,1]),5),",", round(as.numeric(Vec[1,2]),5),","
      , round(as.numeric(Vec[1,3]),5) ,")'.", sep=" " )
```

```
## [1] "The corresponding eigenvector is (  -0.37443 , -0.8483 , -0.37443 )'."
```

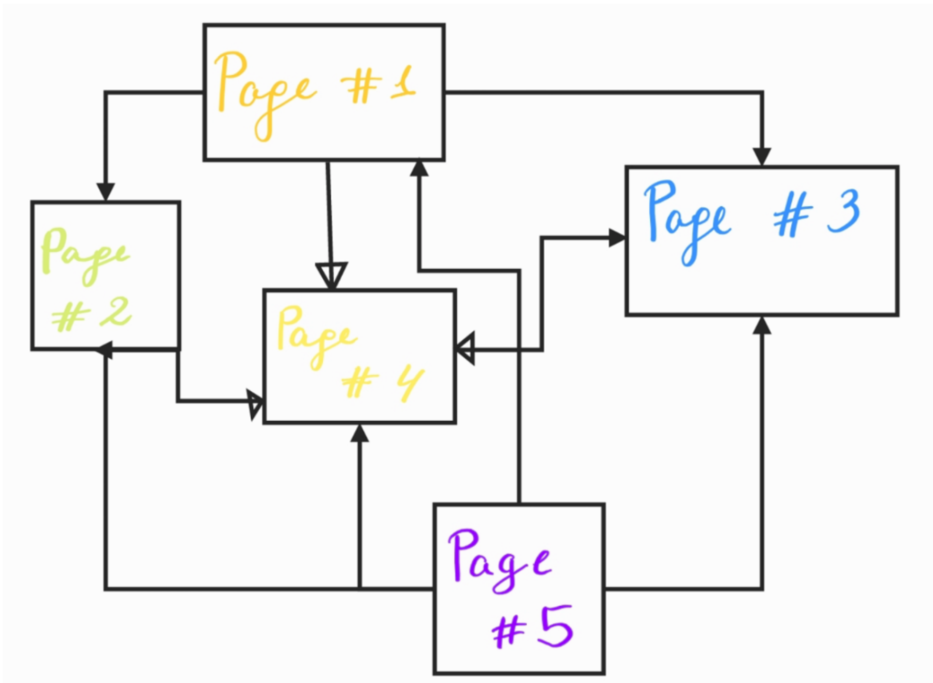
```r
#Define l2 norm function
l2<-function(x){
  v<-as.vector(x)
  norm=0
  for (i in 1:length(v)) {
    norm<-norm+(v[i])^2
  }
  norm<-sqrt(norm)
  return(norm)
}

#Define the three error rates
error1<-10^(-3)
error2<-10^(-6)
error3<-10^(-9)

#Code the power method
pwr<-function(mat, x0, error){
  itr<-0
  v<-x0/l2(x0)
  vi<-v
  while (l2(v-domV)/l2(domV)>error) {
    v<-x0/l2(x0)
    x0<-mat%*%v
    mu<-t(v)%*%x0
    vi<-cbind(vi, v)
    itr<-itr+1
  }
  second<-abs(mu-domEig)
  third<-l2(vi[,ncol(vi)]-domV)/l2(domV)
  fourth<-l2(vi[,ncol(vi)]-vi[,ncol(vi)-1])/l2(vi[,ncol(vi)-1])
  value<-c(itr, second,third,fourth)
  return(value)
}

#Set initial guess
x<-matrix(c(0,-0.2,0), ncol=1)
G<-pwr(A,x,error1)
H<-pwr(A,x,error2)
I<-pwr(A,x,error3)
errors<-c(error1, error2, error3)
dat<-rbind(G,H,I)
dat<-cbind(errors, dat)
row.names(dat)<-1:3
dat
```

```
##   errors
## 1  1e-03 4 3.754159e-04 1.326794e-04 2.279492e-03
## 2  1e-06 6 1.133608e-06 4.007907e-07 6.891594e-06
## 3  1e-09 9 1.881890e-10 6.653604e-11 1.144094e-09
```

ϵ	iteration	$\lvert\mu_k - \lambda^*_{\max}\rvert$	$\frac{\lVert v^{(k)}-v^*\rVert_2}{\lVert v^*\rVert_2}$	$\frac{\lVert v^{(k)}-v^{(k-1)}\rVert_2}{\lVert v^{(k-1)}\rVert_2}$
10^{-3}	4	0.0003754158994038193	0.00013267938697090925	0.002279491553692625
10^{-6}	6	0.0000011336084497771	0.00000040079065390989	0.000006891593685110
10^{-9}	9	0.0000000001881890199	0.00000000006653604084	0.000000001144094242

Note that in practice, we don't know the exact eigenvalues and eigenvectors. So the stopping criteria needs to be replaced by $\frac{\|v^{(k)}-v^{(k-1)}\|_2}{\|v^{(k-1)}\|_2} < \epsilon$.

3. (Coding) Build a connected network graph of 5 nodes, that is, a network with 5 pages. Determine the highest rated web page using the page rank approach discussed in the lecture.




```r
#Construct the adjecency matrix
adjMat<-matrix(c(
  c(0,1,1,1,0),
  c(0,0,0,1,0),
  c(0,0,0,1,0),
  c(0,1,1,0,0),
  c(1,1,1,1,0)
), nrow=5, byrow = T)

#Normalize by row
for (i in 1:5){
  adjMat[i,]<-adjMat[i,]/sum(adjMat[i,])
}

info<-eigen(t(adjMat))
(vals<-info$values)
```

```
## [1]  1 -1  0  0  0
```

```r
(vecs<-info$vectors)
```

```
##           [,1]       [,2]       [,3]       [,4]          [,5]
## [1,] 0.0000000  0.0000000  0.0000000  0.8164966  -8.164966e-01
## [2,] 0.4082483 -0.4082483 -0.7071068 -0.1360828   1.360828e-01
## [3,] 0.4082483 -0.4082483  0.7071068 -0.1360828   1.360828e-01
## [4,] 0.8164966  0.8164966  0.0000000 -0.5443311   5.443311e-01
## [5,] 0.0000000  0.0000000  0.0000000  0.0000000  1.636397e-291
```

```r
Vec1<-vecs[,which(vals==1)]
probVec<-Vec1/sum(Vec1)
paste("Page #", which(probVec==max(probVec)), "is the most popular.", sep=" ")
```

```
## [1] "Page # 4 is the most popular."
```