PURDUE UNIVERSITY · CS 31400
NUMERICAL METHODS

HOMEWORK
David F. Gleich
October 12, 2016

Please answer the following questions in complete sentences in submit the solution on Blackboard October 14th, 2016 by midnight. Note: You can have an optional extension to October 16th, 2016 by 5pm, but then we cannot guarantee that you will get your graded homework back before the midterm.

Updates 2016-10-12 * Changed Matlab to Julia in a few places. * Removed one problem that we won't cover the material (Problem 5.2)

Homework 4

Problem 1 (10 points)

Consider a certain corporation that has three fields of operation; it mines coal, produces gasoline, and generates electricity. Each of these activities makes use of varying amounts of three products. Suppose that in order to produce one unit of coal, the corporation consumes

$$\begin{bmatrix} \text{0 units of coal} \\ \text{1 unit of gasoline} \\ \text{1 unit of electricity} \end{bmatrix}$$

to produce one unit of gasoline, the corporation consumes

$$\begin{bmatrix} \text{0 units of coal} \\ \text{1/5 unit of gasoline} \\ \text{2/5 unit of electricity} \end{bmatrix}$$

and to produce one unit of electricity,

$$\begin{bmatrix} \text{1/5 units of coal} \\ \text{2/5 unit of gasoline} \\ \text{1/5 unit of electricity} \end{bmatrix}$$

Let

$$\mathbf{x} = \begin{bmatrix} x_1 = \text{ number of units of coal produced} \\ x_2 = \text{ number of units of gasoline produced} \\ x_3 = \text{ unumber of units of electricity produced} \end{bmatrix}.$$

This is called a production vector. Suppose the corporation needs to produce 100 units of each product to sell above and beyond its internal requirements.

Derive and solve a linear system for the amount of each material the corporation has to produce to meet this demand. (Hint: don't forget to take the corporations own use into account!)

Problem 2 (10 points)

We'll derive a matrix equation for the PageRank linear system. For this problem, intuitive but *logical* arguments suffice and you need not formally prove these statements. Although, proofs are the most *logical* arguments possible and are accepted and encouraged!

Consider the adjacency matrix of a graph $\boldsymbol{A}$. Let $\boldsymbol{D}$ be a diagonal matrix with the number of out-links from each node on the diagonal. (Assume that each node has at least one out-link.)

The matrix $\boldsymbol{D}^{-1}$ is also a diagonal matrix where we replace each diagonal entry with it's reciprocal. (Note, if this isn't a familiar fact, take a moment to derive it. The property that $\boldsymbol{D}\boldsymbol{D}^{-1} = \boldsymbol{I}$ is handy to use.)

1. (5 points) Argue or prove that if $\mathbf{x}$ is a vector giving the probability that the surfer is on each page, then $\mathbf{y} = \boldsymbol{A}^T\boldsymbol{D}^{-1}\mathbf{x}$ is a vector giving the probability that the surfer is on each page after following one link from the graph.
2. (5 points) Now let $\mathbf{x}$ be the probability that the surfer is on each page after browsing for a really long time. Recall the argument from class, and argue or prove. why:
$$\mathbf{x} = \alpha\boldsymbol{A}^T\boldsymbol{D}^{-1}\mathbf{x} + (1-\alpha)\mathbf{e}/n$$
where $\mathbf{e}$ is the vector of all ones and n is the number of nodes in the graph.
3. (0 points) Rewrite this into the linear system:
$$(\boldsymbol{I} - \alpha\boldsymbol{A}\boldsymbol{D}^{-1})\mathbf{x} = (1-\alpha)\mathbf{e}/n.$$
This give us an easy way to compute PageRank scores on a computer without simulating the Markov chain!

Problem 3 (5 points)

Consider the LU factorization from the quiz.

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}}_{\boldsymbol{P}} \underbrace{\begin{bmatrix} 1.9410 & -0.3510 & 0.7352 & 0.4583 \\ -0.1639 & -1.7494 & 0.0623 & -0.6425 \\ 0.1269 & 1.8231 & -0.6234 & -2.1405 \\ -1.1361 & 0.2914 & -1.4423 & 0.2333 \end{bmatrix}}_{\boldsymbol{A}} =$$

$$\underbrace{\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.0654 & 1.0000 & 0.0000 & 0.0000 \\ 0.5853 & 0.0465 & 1.0000 & 0.0000 \\ -0.0844 & -1.9637 & 0.5331 & 1.0000 \end{bmatrix}}_{\boldsymbol{L}} \underbrace{\begin{bmatrix} 1.9410 & -0.3510 & 0.7352 & 0.4583 \\ 0.0000 & 1.8460 & -0.6715 & -2.1705 \\ 0.0000 & 0.0000 & -0.9807 & 0.6026 \\ 0.0000 & 0.0000 & 0.0000 & -3.0168 \end{bmatrix}}_{\boldsymbol{U}}$$

Report the errors in $\boldsymbol{L}$ and $\boldsymbol{U}$.

Problem 4 (The least squares cluster! 25 points)

1. Chapter 7, problem 14a; but you may not use Julia's "backslash" operation for this part as mentioned in the book. Instead, you must write your own least squares solver using the Julia function `qr`. (See page 170 in the text for the algorithm.)
2. Chapter 7, problem 14b.
3. Chapter 7, problem 14, comment on the differences you see.
4. Chapter 7, problem 15b.
5. Chapter 7, problem 16; this is the teams problem from class. Solve this least-squares problem where Purdue (T1) gets score $r_1 = 100$. And you must derive a least-squares problem where r_1 is not a variable. (Hint: it should be a 5-by-3 matrix for the least-squares problem.)

Problem 5 (Accuracy and conditioning, 10 points)

1. Chapter 6, problem 2
2. **Removed, you do not have to complete this problem.** Show than an orthogonal matrix has condition number 1. (Hint: the submultipliative property is very helpful here.)

Problem 6 (Iterative methods, 15 points)

This problem asks you to report on and explain the results of the following Julia script! Your explanations should be simple. Don't think too much about this and write the first thing that comes to mind.

1. (1 points) Start by downloading two images.

```
download("http://academics.davidson.edu/math/chartier/Numerical/code/dyoung.jpg","dyoung.jpg");
download("http://academics.davidson.edu/math/chartier/Numerical/code/jacobi.jpg","jacobi.jpg");
```

2. (1 points) Show and explain the following output.

```
using Images
im1 = load("jacobi.jpg")
im2 = load("dyoung.jpg")

N = max(size(im1)...)

# You don't have to explain what this function is doing
# You just have to describe the result of using it
# From http://math.mit.edu/~stevenj/18.303/fall13/lecture-10.html
diff1(M) = [ [1.0 zeros(1,M-1)]; diagm(ones(M-1),1) - eye(M) ]
sdiff1(M) = sparse(diff1(M))
# make the discrete -Laplacian in 2d, with Dirichlet boundaries
function Laplacian(Nx, Ny)
   Dx = sdiff1(Nx)
   Dy = sdiff1(Ny)
   Ax = Dx' * Dx
   Ay = Dy' * Dy
   return kron(speye(Ny), Ax) + kron(Ay, speye(Nx))
end

# Normalize A to have 1 on the diagonal
# (This simplifies many thing that would
# otherwise cause us to divide everything
# by 4.)
A = Laplacian(size(im1,1),size(im2,2))
A = A/4

## Show a piece of the matrix A
% A is a sparse matrix, so what does this show?
A[1:10,1:10]

## Try this one too
full(A[1:10,1:10])
```

3. (1 points) Show and explain the following output.

```
#
using Plots
```

```
function myspy(A::SparseMatrixCSC;ms=5.0)
    xn,yn = size(A)
    ai,aj,av = findnz(A)
    scatter(ai,aj,zcolor=av,ms=ms,legend=false,
        yflip=true, aspect_ratio=:equal,
        m=(:auto,1.0,stroke(0)),
        c=:viridis,
        grid=false,
        xlim=(0.5, xn+0.5), ylim=(0.5, yn+0.5),)
end

# You only have to explain this one!
myspy(A)

# And this one
myspy(A;ms=1)
plot!(xlim=(1,512),ylim=(1,512))
```

4. (1 points) Show and explain the following output.

```
# What does multiplication by A do?
X = convert(Array{Float64,2},data(im1))'
x0 = reshape(X,N*N)
x = A*x0;
grayim(reshape(4*x,N,N)') # scaling by 4 makes the picture nicer
```

5. (1 points) Show and explain the following output.

```
## Create b such that image2 is answer to Ax = b
b = A*reshape(convert(Array{Float64,2},data(im2))', N*N)
x0 = reshape(convert(Array{Float64,2},data(im1))', N*N)

## Show the initial vector
grayim(reshape(x0,N,N)')

## Show the right-hand-side vector
grayim(reshape(4*b,N,N)')
```

6. (5 points) Explain what the linear system $\boldsymbol{A}\mathbf{x} = \mathbf{b}$ for this problem solves. (Hint: you may want to look at the next two problems first, which actually solve it!)

7. (1 poins) Run the Richardson method on this problem.

```
# Run the Richardson method
# This is incredibly slow in Julia
omega = 1;
x = x0;
niter = 20;

@gif for i=1:niter
    r = b-A*x;
    x = x + omega*r;
    @show i

    heatmap(-reshape(x,N,N);c=:grays,yflip=true,colorbar=false)
end
```

8. (4 points) Modify the above code to answer the following questions. (These are simple modifications) What does the solution look like after 20 iterations?

How many iterations does it take before you think the solution "looks right"? Hint, it may be useful to use the following code.

```
function richardson(A,b,x0,omega,niter)
    x = x0
    for i=1:niter
        r = b-A*x;
        x = x + omega*r;
    end
    return x
end
omega = 1.0
# This almost works!
x = richardson(A,b,x0,omega,niter)
grayim(reshape(x,N,N)')

# And play it to get the information you need.
```

Problem $\star$ (Not graded, but good practice for the midterm

1. Chapter 7, problem 6
2. Let $\boldsymbol{A}$ be non-singular and let $\lambda, \mathbf{v}$ be an eigenvalue, eigenvector pair of A. Show that $1/\lambda, \mathbf{v}$ are an eigenvalue, eigenvector pair of $\boldsymbol{A}^{-1}$.
3. (Show than an orthogonal matrix has condition number 1. (Hint: the submultipliative property is very helpful here.)