MA580: Project #4, due Nov. 25

Problem 1 Solve Problems 5.5 and 5.13 from Demmel's book.

5.5)

$\lambda_k = min_{V\ such\ that\ dim V = m-k+1} max_{u \in V-0} \frac{u^T(A+H)u}{u^T u}$

$= min_{m-k+1} max \frac{u^T A u}{u^T u} + \frac{u^T H u}{u^T u}$

$= min_{m-k+1} max_{u \in V-0}\ \rho_A +\ min_{m-k+1} max_{u \in V-0}\ \rho_H$

(and $max \rho_H = max(\theta_i) = \theta_i$)

$\leq min_{m-k+1} max_{u \in V-0}\ \rho_A + \theta_1$

$\leq \alpha_k + \theta_1$

Similary we get,

$\lambda_k = max_{S v such\ that\ dim\ S=k}\ min_{u \in S-0} \frac{u^T(A+H)u}{u^T u}$

$max_{S v such\ that\ dim\ S=k}\ min_{u \in S-0} \frac{u^T A u}{u^T u} + \frac{u^T H u}{u^T u}$

(and $min(\rho_H = min(\theta_i) = \theta_n$

$\lambda_k \geq \alpha_k + \theta_k$

Therefore, $\alpha_k + \theta_n \leq \lambda_k \geq \alpha_k + \theta_1$

The eigen values of a symmetric positive definite matrix are real and positive which means that it can't be zero and $\lambda_k \geq \alpha_k$

5.13)

$x_0 = [0, \ldots, 1]^T$ size= $m \times 1$

QR iteration with shift which is a_{nn},

$a^k_{nn} = e^T_m A^k e_m$ From the class, $A^k = (Q^k)^T A^{k-1} Q^k$

By induction,$A^k = (\underset{\sim}{Q}^k)^T A \underset{\sim}{Q}^k$ where, $\underset{\sim}{Q}^k = \prod_{j=1}^k Q^j$

Thus we get, $a^k_{nn} = e^T_m (Q^k)^T A^{k-1} Q^k e_m$ and, $a^k_{nn} = e^T_m \underset{\sim}{Q}^k)^T A \underset{\sim}{Q}^k e_m$

and it changes to, $a^k_{nn} = q^{kT}_m A q^k_m$

where, $\mu^k = q^{kT}_m A q^k_m$

and μ^k and q^k_m are identical to those that are computed by the Rayleigh quotient iteration starting with e_m.

Thus, the sequences of Rayleigh quotients is identical to the sequence of shifts for shifted QR iterations.

Problem 2 If a two-dimensional membrane (drum) Ω with a fixed edge $\partial\Omega$ is struck, it will oscillate. It can be shown that the displacement $u(x, y, t)$ at some point $(x, y) \in \Omega$ and at time t is described by combinations of terms of the form $u_i(x, y)\sin\sqrt{\lambda_i}t$ where $(u_i, \lambda_i$ is an eigenfunction/eigenvalue pair of $-\Delta = -\frac{\partial^2}{\partial x^2} - \frac{\partial^2}{\partial y^2}$, i.e.

$$\begin{aligned} -\Delta u_i(x, y) &= \lambda_i\, u_i(x, y) \quad (x, y) \in \Omega, \\ u_i(x, y) &= 0 \qquad\qquad\;\; (x, y) \in \partial\Omega, \end{aligned}$$

1. If $\Omega = (0, 1)^2$, check that for $p, q \in \mathbb{N}$,

$$\begin{aligned} u_{pq}(x, y) &= \sin(p\pi x)\, \sin(q\pi y), \\ \lambda_{pq} &= (p^2 + q^2)\, \pi^2, \end{aligned}$$

are eigenfunctions and eigenvalues. Plot the first three $(p = 1, q = 1)$, $(p = 2, q = 1)$, $(p = 2, q = 2)$.

$\frac{\partial u}{\partial x} = p\pi\, cos(p\pi x) sin(q\pi y)$

$\frac{\partial^2 u}{\partial x^2} = -p^2\pi^2\, sin(p\pi x) sin(q\pi y)$

$\frac{\partial u}{\partial y} = q\pi\, cos(q\pi y) sin(p\pi x)$

$\frac{\partial^2 u}{\partial y^2} = -q^2\pi^2\, sin(q\pi y) sin(p\pi x)$

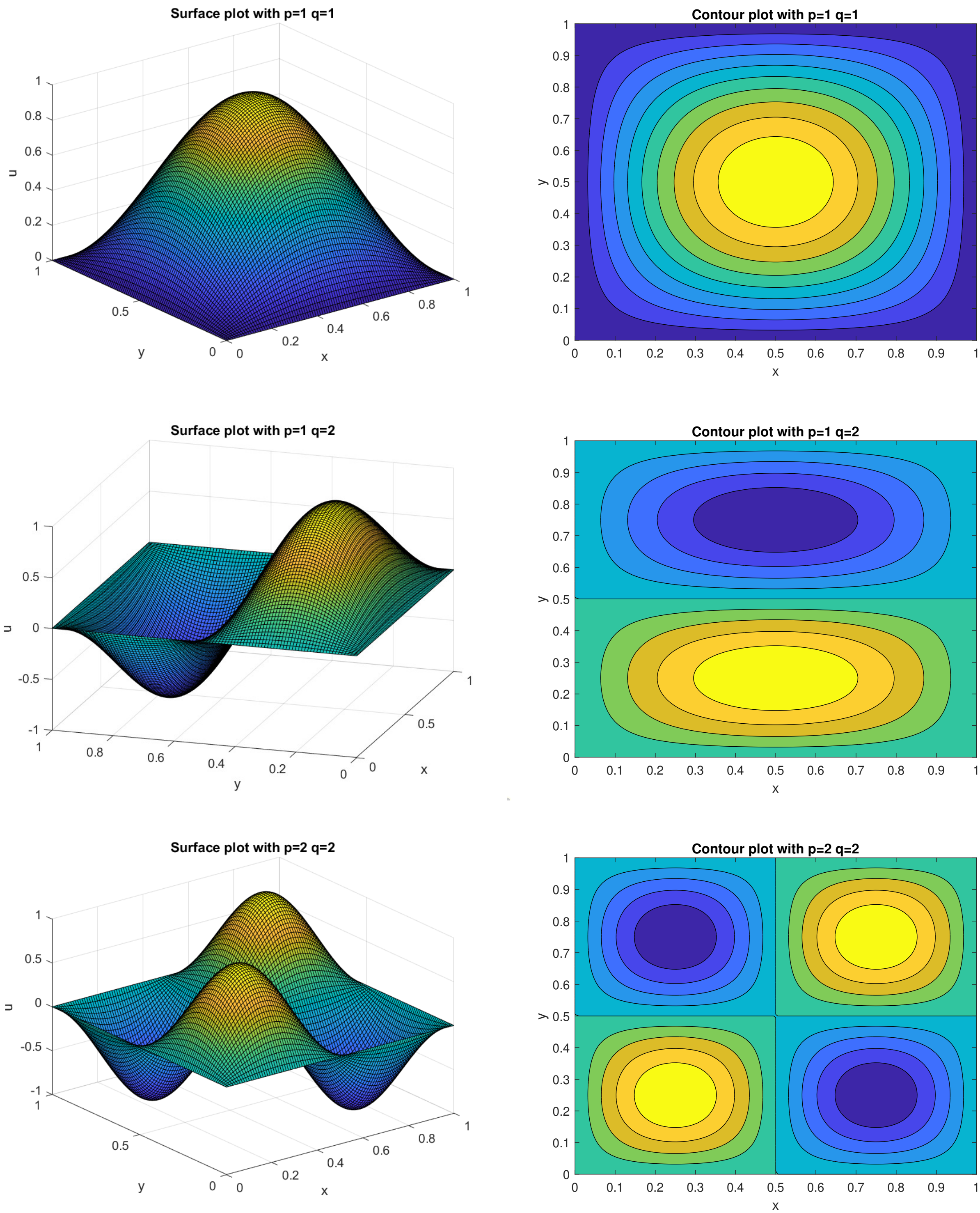
$-\Delta u = -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2}$
$= -(-p^2\pi^2\, sin(p\pi x) sin(q\pi y)) - (-q^2\pi^2\, sin(q\pi y) sin(p\pi x))$

$= (p^2 + q^2)\pi^2\, sin(p\pi x) sin(q\pi y)$

$= \lambda_i\, u_i(x, y)$

Hence, we have proven that $u_{pq}(x, y)$ is eigenfunction and λ_{pq} gives eigen value of above domain.

```
u=zeros([100 100]);
x1=linspace(0,1,100);
y1=x1;
global x;
global y;
[x,y]=meshgrid(x1,y1);
p=1;
q=2;
    for a=1:100
        for b=1:100
            temp1=sin(p*pi*x(a,b));
            temp2=sin(q*pi*y(a,b));
            u(a,b)=temp1*temp2;
        end
    end
%contourf(x,y,u);
surf(x,y,u);
xlabel("x");
ylabel("y");
zlabel("u");
title("Surface plot with p=1 q=2");
```



2. For the same case $\Omega = (0,1)^2$, consider the method from Problem 2, Project 2. Verify that the exact eigenvalues of the approximate problem, i.e., the eigenvalues of $\mathbb{A}/\Delta x^2$ are given by

$$\tilde{\lambda}_{pq} = \frac{4}{\Delta x^2}\left(\sin^2(\frac{p\pi\Delta x}{2}) + \sin^2(\frac{q\pi\Delta x}{2})\right).$$

$-\Delta u_i(x,y) = \lambda_i\, u_i(x,y)$

$-\frac{u_{i+1,j}(x,y)-2u_{i,j}(x,y)+u_{i-1,j}(x,y)}{\Delta x^2} - \frac{u_{i,j+1}(x,y)-2u_{i,j}(x,y)+u_{i,j-1}(x,y)}{\Delta x^2} = \lambda_i\, u_i(x,y)$

$(\frac{4}{\Delta x^2} - \lambda)u_{i,j}(x,y) - \frac{u_{i+1,j}(x,y)-u_{i-1,j}(x,y)}{\Delta x^2} - \frac{u_{i,j+1}(x,y)-u_{i,j-1}(x,y)}{\Delta x^2}$

$(\frac{4}{\Delta x^2} - \lambda)sin(p\pi x_{ij})sin(q\pi y_{ij}) - \frac{sin(p\pi(x_{ij}+\Delta x))sin(q\pi y_{ij})}{\Delta x^2} - \frac{sin(p\pi(x_{ij}-\Delta x))sin(q\pi y_{ij})}{\Delta x^2} - \frac{sin(p\pi(x_{ij}))sin(q\pi y_{ij}+\Delta x)}{\Delta x^2} - \frac{sin(p\pi(x_{ij}))sin(q\pi y_{ij}-\Delta x)}{\Delta x^2} = 0$

using the trignometric identity$sin(A+B) + sin(A-B) = 2sin(A)cos(B)$
Therefore, we use this to simplify the above terms and after cancelling out common terms we get,

$\frac{4}{\Delta x^2} - \lambda \;=\; \frac{2}{\Delta x^2}(cos(p\pi\Delta x) + cos(q\pi\Delta x))$

$\lambda = \frac{4}{\Delta x^2} - \frac{2}{\Delta x^2}(cos(p\pi\Delta x) + cos(q\pi\Delta x))$

$\lambda \;=\; \frac{2}{\Delta x^2}(1 - cos(p\pi\Delta x) + 1 - cos(q\pi\Delta x))$

$\lambda \;=\; \frac{2}{\Delta x^2}(1 - cos(\frac{2p\pi\Delta x}{2}) + 1 - cos(\frac{2q\pi\Delta x}{2}))$

using the identity, $1 - cos(2A) = 2sin^2(A)$ we get,

$\lambda \;=\; \frac{2}{\Delta x^2}(2sin^2(\frac{p\pi\Delta x}{2}) + 2sin^2(\frac{q\pi\Delta x}{2}))$

$\tilde{\lambda} \;=\; \frac{4}{\Delta x^2}(sin^2(\frac{p\pi\Delta x}{2}) + sin^2(\frac{q\pi\Delta x}{2}))$

Hence proven.

3. Write your own inverse power method iteration code (with shift) and compute the first three eigenvalues and eigenvectors in the case $n = 100$. Discuss convergence.

```
n=100;
A=gallery("poisson",n);
tol=1e-5;
% ews=eig(A);
delx=1/(n);
p=1;
q=2;
s1=4*(sin(p*pi*delx/2)*sin(p*pi*delx/2)+sin(q*pi*delx/2)*sin(q*pi*delx
    /2));
p=2;
q=2;
s2=4*(sin(p*pi*delx/2)*sin(p*pi*delx/2)+sin(q*pi*delx/2)*sin(q*pi*delx
    /2));

mu = 0;
%B= A-mu*eye(size(A));
%Binv= inv(B);

%disp(ews);
lamda=zeros([3 1]);
%resvec=zeros(1000);
v=zeros([n*n 1]);
v(n*n,1)=1;
told=1;
```

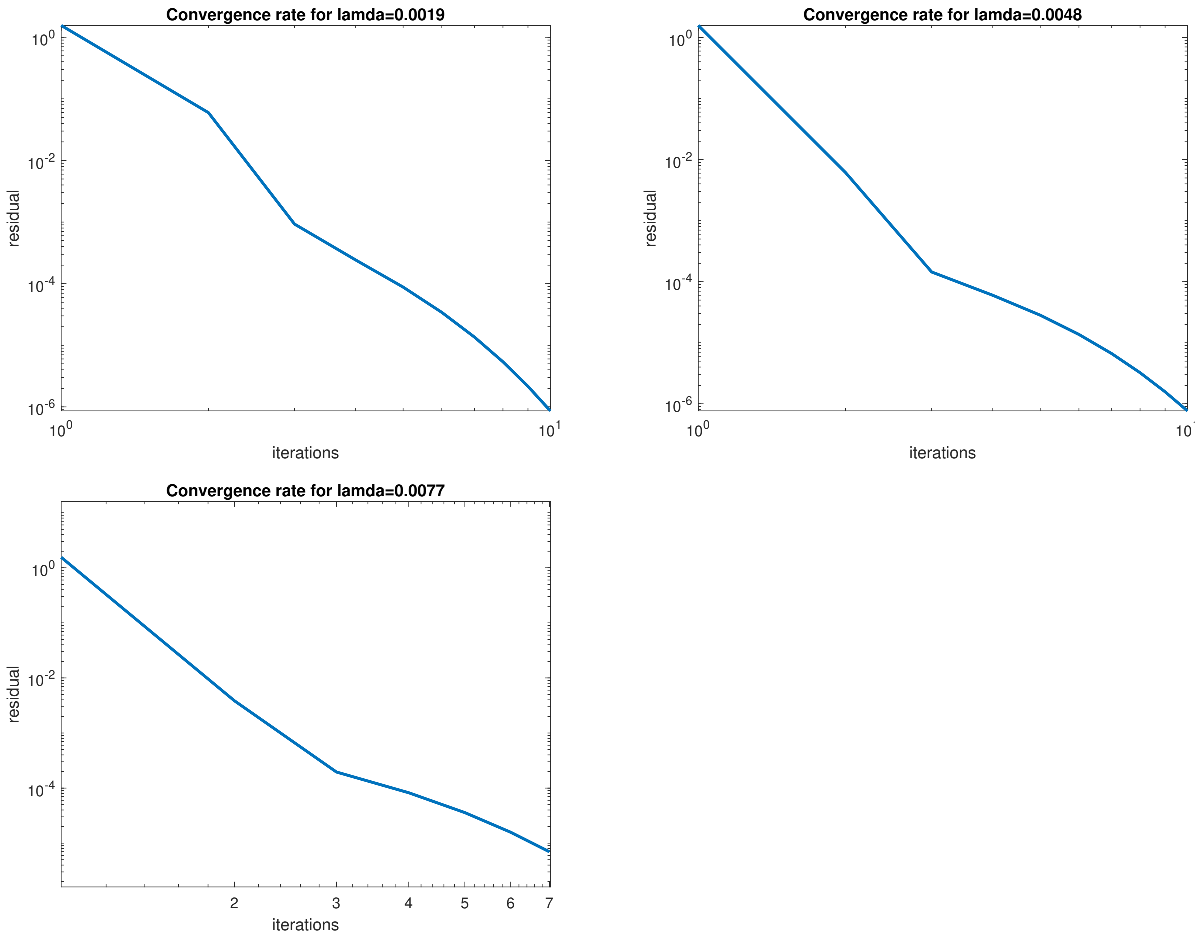
```
 ev=zeros(3,n*n);
 iter=1;
 count=zeros([3  1]);
for  i=1:3

     if  i==1
         mu=0;
         mu
     elseif  i==2
         mu=abs(s1-lamda(1))+0.3*(s2-s1);
         mu
     elseif  i==3
         mu=abs(s2-lamda(1))+0.3*(s2-s1);
         mu
     end
     B= A-mu.*eye(size(A));
     Binv= inv(B);

     iter=1
     for  k = 1:100
     %while  resvec(i,iter) <= 1e-2

         y=Binv*v;
         size(y);
         v=y/norm(y);
         size(v);
         t1=A*v;
         t2=v'*t1;

         if  abs(t2 - told)/told > tol
              lamda(i)=t2;
              ev(i,:)=v;
         end
         told=lamda(i);

         resvec(i,k)=norm(A*v-lamda(i).*v)/norm(v);
         % iter=iter+1;
     end

      v=zeros([n*n  1]);
      v(n*n,1)=1;
end

lamda

%loglog(1:20,resvec(2,100:),'LineWidth',2);
```

we get,

$\lambda_{pq} = 0.0019 \qquad p = 1, q = 1$

$\lambda_{pq} = 0.0048 \qquad p = 1, q = 1$

$\lambda_{pq} = 0.0077 \qquad p = 1, q = 1$

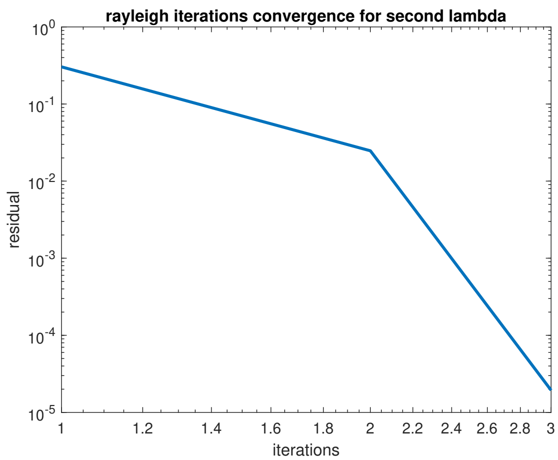


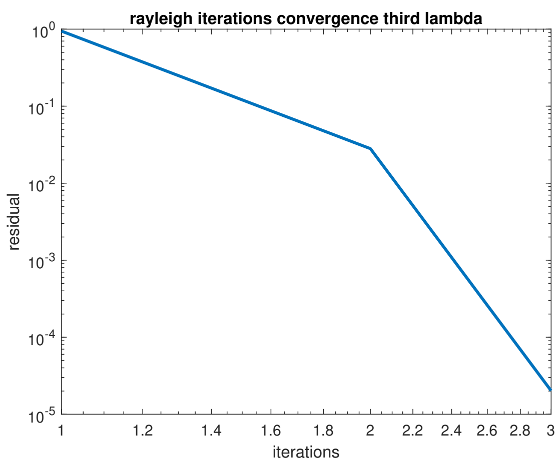
Convergence rate of inverse power method

Inverse Power method converges to a tolerance of 10^{-6} in 14 iterations.

4. Do Rayleigh iterations improve convergence? Discuss.

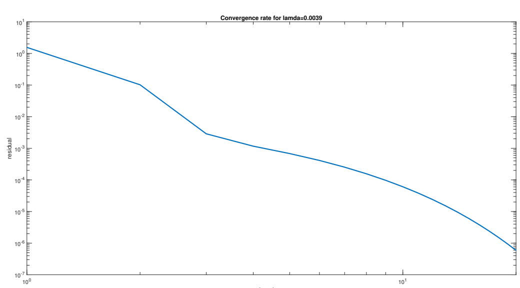
Yes they do improve convergence but they are more expensive.
Convergence rate for first λ is same as previous method as shift=0 for both of them.





5. Bonus: repeat the last two points for other domains Ω.

   Code and convergence for inverse power method with L shaped domain




```
n=100;

R = 'L' ; % Other possible shapes include S ,N,C,D,A,H,B

```

```
[A,N] = ellmat(n ,R) ;
n=7203
tol=1e-5;
 ews=eig(A);
delx=1/(n);
p=1;
q=2;
s1=4*(sin(p*pi*delx/2)*sin(p*pi*delx/2)+sin(q*pi*delx/2)*sin(q*pi*delx
    /2));
p=2;
q=2;
s2=4*(sin(p*pi*delx/2)*sin(p*pi*delx/2)+sin(q*pi*delx/2)*sin(q*pi*delx
    /2));

mu = 0;
%B= A-mu*eye(size(A));
%Binv= inv(B);

%disp(ews);
lamda=zeros([3 1]);
%resvec=zeros(1000);
v=zeros([7203 1]);
v(n,1)=1;
told=1;
 ev=zeros(3,n);
 iter=1;
 count=zeros([3 1]);
for i=1:3

    if i==1
        mu=0;
        mu
    elseif i==2
        mu=abs(s1-lamda(1))+0.3*(s2-s1);
        mu
    elseif i==3
        mu=abs(s2-lamda(1))+0.3*(s2-s1);
        mu
    end
    B= A-mu.*eye(size(A));
    Binv= inv(B);

    iter=1
    for k = 1:100
    %while resvec(i,iter) <= 1e-2

        y=Binv*v;
        size(y);
        v=y/norm(y);
        size(v);
        t1=A*v;
        t2=v'*t1;

        if abs(t2 - told)/told > tol
```

```
                lamda(i)=t2;
                ev(i,:)=v;
            end
            told=lamda(i);

            resvec(i,k)=norm(A*v-lamda(i).*v)/norm(v);
           % iter=iter+1;
        end

       % v=zeros([7203 1]);
        %v(7203,1)=1;
end

lamda

%loglog(1:20,resvec(2,100:),'LineWidth',2);

function [A,N] = ellmat(n ,R)
 % Generate and display the grid.
G = numgrid (R, n) ;
%spy (G)
%g = numgrid (R, n)

A = delsq(G) ; % discrete Laplacian
%spy(A)
N = sum(G(:)>0) ; % number of inner nodes
end
```