Mikhail Gaerlan
16 March 2018
MAT 226B Freund

Problem 1

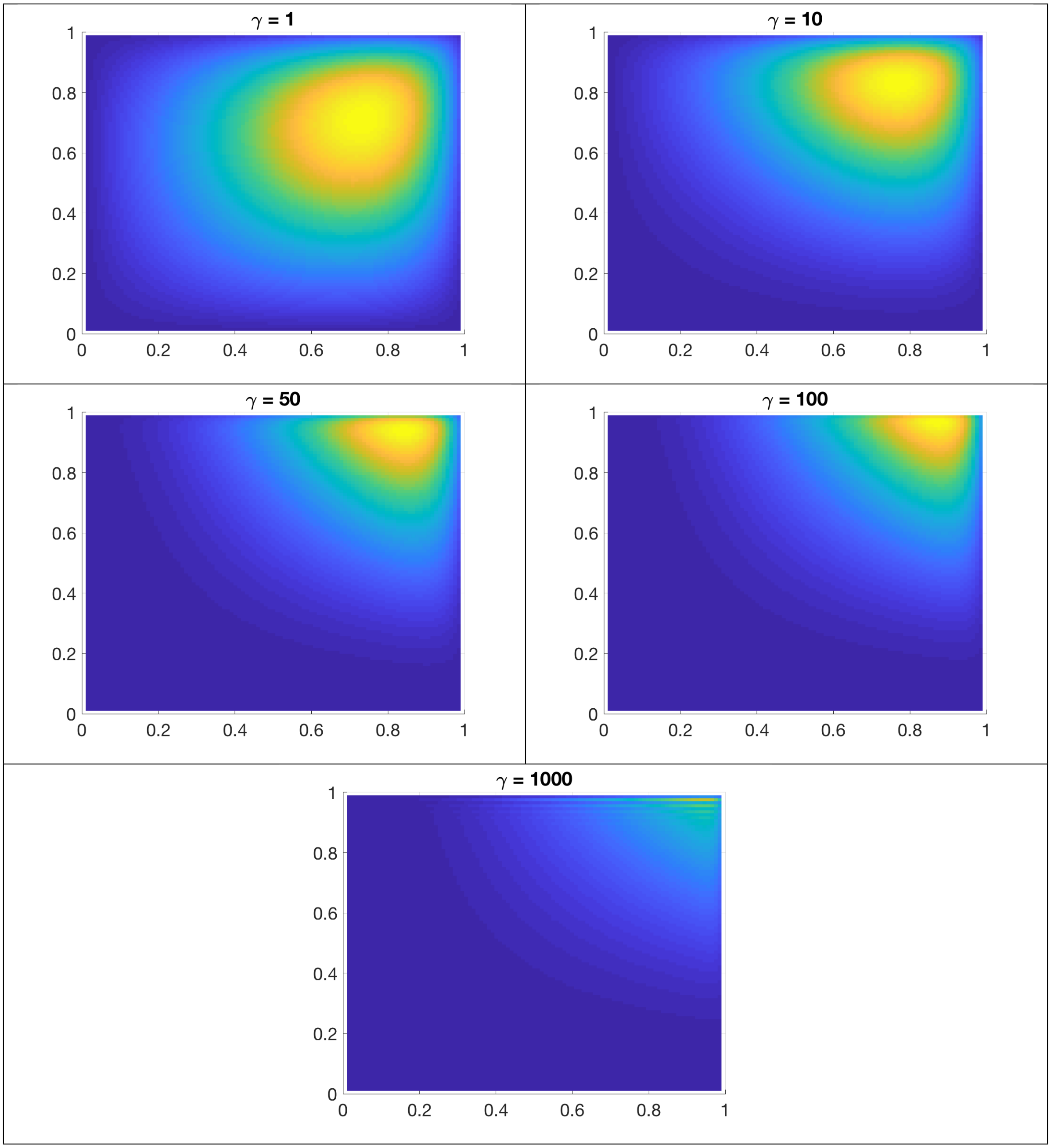
```
function V = solvePoisson(m,F)
%Solve two-dimensional Poisson's equation T_mV + VT_m = F
%   input  - m             matrix size
%            F             right-hand side
%   output - V             solution matrix
h = 1/(m+1); G = multZ(h,multZ(h,F.').'); V = zeros(m,m);
for k = 1:m; for j = 1:m; V(j,k) = G(j,k)/(l(h,j)+l(h,k)); end; end
V = multZ(h,multZ(h,V.').');
end
function lam = l(h,j); lam = 2*(1-cos(pi*h*j)); end
function W = multZ(h,V)
%Multiply ZV where Z is the eigenvector matrix of T_m
%   input  - V is an n x n matrix
%   output - W = ZV
m = size(V,1);
Vt = vertcat(zeros(1,m),V,zeros(m+1,m));
Wt = fft(Vt);
W = -sqrt(2*h)*imag(Wt(2:m+1,:));
end
```

```
function v = multAp(x,gamma,m)
%Multiply a vector v = (A_0 + gamma A_1) x
%   input  - x       input vector
%          - gamma   constant
%          - m       size of vector x
%   output - v       output vector
h = 1/(m+1); v = x + gamma*solveM1(multA1(x));
    function v = solveM1(x)
        H = reshape(x,[m,m]);
        V = solvePoisson(m,H);
        v = reshape(V,[m*m,1]);
    end
    function v = multA1(b)
        v = vertcat(b(m+1:end),zeros(m,1));
        v(m+1:end) = v(m+1:end)-b(1:end-m);
        v = v*h/2;
    end
end
```

```
function [v,iter,relres] = poissonSolve(m,b,gamma,tol)
%Solve two-dimensional Poisson's equation with extra x-derivative
%   input  - m            matrix size
%            b            right-hand side vector
%            gamma        parameter
%            tol          error tolerance
%            v0           initial vector
%   output - v            solution vector
%          - iter         number of iterations
%          - relres       relative residual norm of final GMRES iterate
[v,~,relres,iters,~] = gmres(@(x)multAp(x,gamma,m),solveM1(b),[],tol,m^2);
iter = iters(2);
    function v = solveM1(x)
        H = reshape(x,[m,m]);
        V = solvePoisson(m,H);
        v = reshape(V,[m*m,1]);
    end
end
```

```
clearvars; m = 100; h = 1/(m+1); [Y,X]=meshgrid(h:h:1-h);
tol = 1e-10; ms = []; relress = [];
x = X(1,:);y = Y(:,1); F = f(X,Y);
b0 = ones(1,m); b1 = ones(1,m); c0 = ones(m,1); c1 = ones(m,1);
for gamma = [1,10,50,100,1000]
    G = h^2*F; G(1,:) = G(1,:)+b0; G(end,:) = G(end,:)+b1;
    G(:,1) = G(:,1)+c0+gamma*c0*h/2; G(:,end) = G(:,end)+c1-gamma*c1*h/2;
    [v,iter,relres] = poissonSolve(m,reshape(G,[m*m,1]),gamma,tol);
    ms = [ms,iter]; relress = [relress,relres]; clf; colorbar;
    surf(X,Y,reshape(v,[m,m]),'LineStyle','none'); hold on; view(2);
    title(strcat("\gamma = ",int2str(gamma))); set(gca,'FontSize',20);
    saveas(gcf,strcat("../Figures/homework5_1_",int2str(gamma),".png"));
end
matrix2latex(ms','../Tables/homework5_1_iters.tex','alignment','r')
matrix2latex(relress','../Tables/homework5_1_relres.tex','alignment','r','format','%-.15e')
function fun = f(x,y); fun = x.^3.*y.^2.*exp(2-x-y); end
```

γ	Iterations	Relative Residual Norm
1	7	3.877478556473064e-11
10	18	3.548776304593922e-11
50	50	8.226758443269179e-11
100	83	8.780792025563864e-11
1000	391	9.733282686797891e-11


γ = 1
γ = 10
γ = 50
γ = 100
γ = 1000
0
0.2
0.4
0.6
0.8
1

Problem 2

$$x_1^{\mathrm{CG}} = x_0 + \frac{r_0^T r_0}{r_0^T A r_0} r_0$$

Problem 3

With an initial vector r, we can use the Hermitian Lanczos process to find an H_k and V_k such that $A \approx V_k H_k V_k^H$. Then

$$\begin{aligned}
(A\lambda)^2 &\approx V_k H_k V_k^H \lambda V_k H_k V_k^H \lambda \\
&\approx V_k H_k V_k^H V_k H_k V_k^H \lambda^2 \\
&\approx V_k H_k H_k V_k^H \lambda^2 \\
&\approx V_k \left(H_k \lambda\right)^2 V_k^H \\
A^j &\approx V_k \left(H_k \lambda\right)^j V_k^H \\
e^{A\lambda} b &\approx \left(\sum_{j=0}^{\infty} \frac{1}{j!} V_k \left(H_k \lambda\right)^j V_k^H \right) b \\
&\approx V_k \left(\sum_{j=0}^{\infty} \frac{1}{j!} \left(H_k \lambda\right)^j \right) V_k^H b \\
&\approx V_k e^{H_k \lambda} V_k^H b
\end{aligned}$$

which only requires $k \times k$, $k \times n$, and $n \times k$ matrix multiplications where $k \ll n$.

Problem 4

```
function [Hkt,Vkk] = arnoldi(A,r,kmax,tol)
%Use the Arnoldi process on a matrix A
%   input  - A     matrix
%            r     starting vector
%            kmax  max iterations
%            tol   norm(q) tolerance
%   output - Hkt   upper-Hessenberg matrix
%            Vkk   matrix of Arnoldi vectors
beta = norm(r); Vkk = r/beta;
for k = 1:kmax
    q = A(Vkk(:,k));
    for j = 1:k; Hkt(j,k) = Vkk(:,j)'*q; q = q-Hkt(j,k)*Vkk(:,j); end
    Hkt(k+1,k) = norm(q);
    if norm(q) <= tol; break; end
    Vkk(:,k+1) = q/Hkt(k+1,k);
end
end
```

```
clearvars; m = 100; h = 1/(m+1); kmax = 300;
load("HW5_P4.mat"); mins=[]; maxes=[]; minlambdas=[]; maxlambdas=[];
for gamma = [1,10,50,100,1000]
    [Hkt,Vkk] = arnoldi(@(z)multAp(z,gamma,m),r,kmax,0);
    Hk = Hkt(1:end-1,:); [Z,Lt] = eig(Hk); lambdas = diag(Lt);
    rhos = Hkt(end,end)*abs(Z(end,:));
    clf; plot(lambdas,'.');
    title(strcat("\gamma = ",int2str(gamma))); set(gca,'FontSize',20);
    saveas(gcf,strcat("../Figures/homework5_4_",int2str(gamma),".png"));
    rhomin = min(rhos); rhomax = max(rhos);
    mins = [mins;rhomin]; maxes = [maxes;rhomax];
    i = find(rhos==rhomin); j = find(rhos==rhomax);
    matrix2latex(lambdas(i),strcat("../Tables/homework5_4_mins_",num2str(gamma),".tex"),'alignment','r','format','%-.15e')
    matrix2latex(lambdas(j),strcat("../Tables/homework5_4_maxes_",num2str(gamma),".tex"),'alignment','r','format','%-.15e')
end
matrix2latex(mins,"../Tables/homework5_4_mins.tex",'alignment','r','format','%-.15e')
matrix2latex(maxes,"../Tables/homework5_4_maxes.tex",'alignment','r','format','%-.15e')
```

γ	Iterations	Relative Residual Norm
1	0.000000000000000e+00	1.226016708116490e-01
10	0.000000000000000e+00	8.580633957899474e-03
50	0.000000000000000e+00	4.159214697792878e-02
100	0.000000000000000e+00	7.755234807196754e-02
1000	0.000000000000000e+00	8.129363507071738e-01

$\gamma = 1$

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmin}}\,(\rho_i)$
5.925824662088014e-16
-7.169849399663779e-17
1.000000000000002e+00+1.124896368980689e-01i
1.000000000000002e+00-1.124896368980689e-01i

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmax}}\,(\rho_i)$
8.613669983806195e-01

$\gamma = 10$

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmin}}\,(\rho_i)$
1.000000000000003e+00+1.124896368980685e+00i
1.000000000000003e+00-1.124896368980685e+00i

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmax}}\,(\rho_i)$
9.999586452890994e-01+6.220077627914600e-02i
9.999586452890994e-01-6.220077627914600e-02i

$\gamma = 50$

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmin}}\,(\rho_i)$
1.000000000000001e+00+5.624481844903410e+00i
1.000000000000001e+00-5.624481844903410e+00i
9.999999999999996e-01+3.556289316821839e+00i
9.999999999999996e-01-3.556289316821839e+00i
1.000000000000000e+00+3.553704482968765e+00i
1.000000000000000e+00-3.553704482968765e+00i

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmax}}\,(\rho_i)$
1.000307573385462e+00+3.092463649917729e-01i
1.000307573385462e+00-3.092463649917729e-01i

$\gamma = 100$

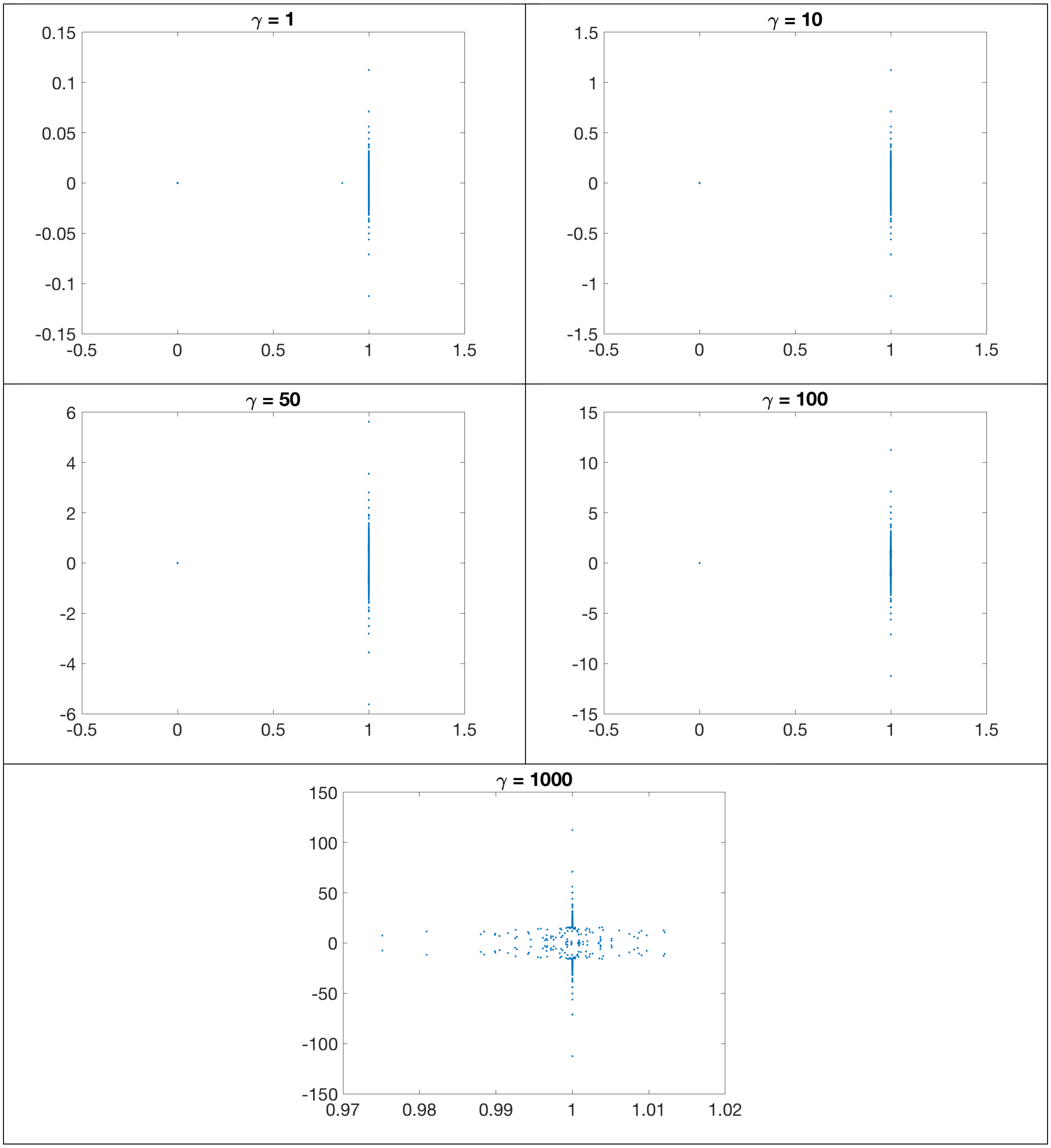
λ_j such that $j = \underset{1\leq i\leq k}{\text{argmin}}\,(\rho_i)$
1.000000000000001e+00+1.124896368980683e+01i
1.000000000000001e+00-1.124896368980683e+01i
9.999999999999971e-01+7.112578633643686e+00i
9.999999999999971e-01-7.112578633643686e+00i
1.000000000000004e+00+7.107408965937533e+00i
1.000000000000004e+00-7.107408965937533e+00i

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmax}}\,(\rho_i)$
1.000575887620807e+00+2.346007827840137e-02i
1.000575887620807e+00-2.346007827840137e-02i

$\gamma = 1000$

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmin}}\,(\rho_i)$
1.000000000000009e+00+1.124896368980682e+02i
1.000000000000009e+00-1.124896368980682e+02i
9.999999999999911e-01+7.112578633643690e+01i
9.999999999999911e-01-7.112578633643690e+01i
1.000000000000001e+00+7.107408965937532e+01i
1.000000000000001e+00-7.107408965937532e+01i

λ_j such that $j = \underset{1\leq i\leq k}{\text{argmax}}\,(\rho_i)$
9.751427114428090e-01+7.465537420035416e+00i
9.751427114428090e-01-7.465537420035416e+00i


γ = 1
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.5
0
0.5
1
1.5
γ = 10
1.5
1
0.5
0
-0.5
-1
-1.5
-0.5
0
0.5
1
1.5
γ = 50
6
4
2
0
-2
-4
-6
-0.5
0
0.5
1
1.5
γ = 100
15
10
5
0
-5
-10
-15
-0.5
0
0.5
1
1.5
γ = 1000
150
100
50
0
-50
-100
-150
0.97
0.98
0.99
1
1.01
1.02

Problem 5

```
function Tk = hermlanc(A,r,kmax,tol)
%Apply the Hermitian Lanczos process for a Hermitian matrix
%   input  - A     Hermitian matrix
%          - r     starting vector
%          - kmax  max iterations
%          - tol   norm(q) tolerance
%   output - Tk    tridiagonal matrix
beta = norm(r); vk = r/beta;
for k = 1:kmax
    q = A*vk; if k > 1; q = q - beta*vk1; end
    Tk(k,k) = vk'*q; q = q - Tk(k,k)*vk; beta = norm(q);
    if beta <= tol; break; end
    if k ~= kmax
        Tk(k+1,k) = beta; Tk(k,k+1) = beta;
        vk1 = vk; vk = q/beta;
    end
end
end
```

(a)

```
clearvars; load("HW5_P5a.mat"); m = 2;
A = make_3d_laplacian(m); kmax = 2000; Tk = hermlanc(A,r,kmax,1e-10);
matrix2latex(eig(Tk),"../Tables/homework5_5_a_approx.tex",'alignment','r',
    'format','%-.15e')
for i=1:m;for j=i:m; for l=j:m;lambdas(i,j,l)=lambda(i,j,l,m);end;end;end
matrix2latex(lambdas(lambdas~=0),"../Tables/homework5_5_a_exact.tex",'
    alignment','r','format','%-.15e')
function lam = lambda(i,j,l,m)
lam = 2.*(3-cos((i.*pi)./(m+1))-cos((j.*pi)./(m+1))-cos((l.*pi)./(m+1)));
end
```

$\tilde{\lambda}_k$	λ_k
3.000000000000000e+00	3.000000000000000e+00
5.000000000000003e+00	5.000000000000000e+00
6.999999999999999e+00	7.000000000000000e+00
9.000000000000002e+00	9.000000000000000e+00

(b)

```
clearvars; load("HW5_P5b.mat"); m = 69;
A = make_3d_laplacian(m);
for kmax = [250,500,1000,2000]
    clearvars -except kmax A m r;
    Tk = hermlanc(A,r,kmax,1e-14); eigs = eig(Tk);
    for i=1:m;for j=i:m; for l=j:m;ls(i,j,l)=le(i,j,l,m);end;end;end
    ls = ls(ls~=0);
    matrix2latex(mink(eigs,10),strcat("../Tables/homework5_5_b_min_approx_",num2str(kmax),".tex"),'alignment','r','format','%-.15e')
    matrix2latex(mink(ls,10),strcat("../Tables/homework5_5_b_min_exact_",num2str(kmax),".tex"),'alignment','r','format','%-.15e')
    matrix2latex(maxk(eigs,10),strcat("../Tables/homework5_5_b_max_approx_",num2str(kmax),".tex"),'alignment','r','format','%-.15e')
    matrix2latex(maxk(ls,10),strcat("../Tables/homework5_5_b_max_exact_",num2str(kmax),".tex"),'alignment','r','format','%-.15e')
end
function lam = le(i,j,l,m)
lam = 2.*(3-cos((i.*pi)./(m+1))-cos((j.*pi)./(m+1))-cos((l.*pi)./(m+1)));
end
```

$k = 250$ smallest		largest	
$\tilde{\lambda}_k$	λ_k	$\tilde{\lambda}_k$	λ_k
6.041600777348929e-03	6.041600752111798e-03	1.199395839789970e+01	1.199395839924789e+01
1.207914596602798e-02	1.207914584426306e-02	1.198792085346275e+01	1.198792085415574e+01
1.811677334763728e-02	1.811669093641410e-02	1.198186057903147e+01	1.198188330906359e+01
2.213178125207190e-02	2.212821029887158e-02	1.197777726554531e+01	1.197787178970113e+01
2.817776082327112e-02	2.415423602856515e-02	1.197301741988887e+01	1.197584576397144e+01
3.430388477604603e-02	2.816575539102262e-02	1.197046186175222e+01	1.197183424460898e+01
4.073916196053342e-02	3.420330048317366e-02	1.196317078231754e+01	1.196579669951683e+01
4.844284495010276e-02	3.616855663748186e-02	1.195560060935483e+01	1.196383144336252e+01
5.491496666673977e-02	3.821481984563135e-02	1.194813249018734e+01	1.196178518015437e+01
6.427178030291032e-02	4.220610172963291e-02	1.193967841648353e+01	1.195779389827037e+01

$k = 500$ smallest		largest	
$\tilde{\lambda}_k$	λ_k	$\tilde{\lambda}_k$	λ_k
6.041600752111682e-03	6.041600752111798e-03	1.199395839924789e+01	1.199395839924789e+01
1.207914584426521e-02	1.207914584426306e-02	1.198792085415574e+01	1.198792085415574e+01
1.811669093641410e-02	1.811669093641410e-02	1.198188330906359e+01	1.198188330906359e+01
2.212821029887092e-02	2.212821029887158e-02	1.197787178970113e+01	1.197787178970113e+01
2.415423602856659e-02	2.415423602856515e-02	1.197584576397144e+01	1.197584576397144e+01
2.816575539102289e-02	2.816575539102262e-02	1.197183424460897e+01	1.197183424460898e+01
3.420330048307895e-02	3.420330048317366e-02	1.196579669952040e+01	1.196579669951683e+01
3.616855658492681e-02	3.616855663748186e-02	1.196383144337132e+01	1.196383144336252e+01
3.821481941474132e-02	3.821481984563135e-02	1.196178518246027e+01	1.196178518015437e+01
4.220609548937836e-02	4.220610172963291e-02	1.195779390071517e+01	1.195779389827037e+01

$k = 1000$ smallest		largest	
$\tilde{\lambda}_k$	λ_k	$\tilde{\lambda}_k$	λ_k
6.041600752106037e-03	6.041600752111798e-03	1.199395839924790e+01	1.199395839924789e+01
6.041600752113144e-03	1.207914584426306e-02	1.199395839924789e+01	1.198792085415574e+01
6.041600752114275e-03	1.811669093641410e-02	1.199395839924786e+01	1.198188330906359e+01
1.207914584426428e-02	2.212821029887158e-02	1.198792085415574e+01	1.197787178970113e+01
1.207914584426578e-02	2.415423602856515e-02	1.198792085415573e+01	1.197584576397144e+01
1.211821679530226e-02	2.816575539102262e-02	1.198791372575761e+01	1.197183424460898e+01
1.811669093641226e-02	3.420330048317366e-02	1.198188330906359e+01	1.196579669951683e+01
1.811669093641471e-02	3.616855663748186e-02	1.198188330906359e+01	1.196383144336252e+01
2.212821029887353e-02	3.821481984563135e-02	1.197787178970113e+01	1.196178518015437e+01
2.212821029887629e-02	4.220610172963291e-02	1.197787178970112e+01	1.195779389827037e+01

$k = 2000$ smallest		largest	
$\tilde{\lambda}_k$	λ_k	$\tilde{\lambda}_k$	λ_k
6.041600752108586e-03	6.041600752111798e-03	1.199395839924791e+01	1.199395839924789e+01
6.041600752108720e-03	1.207914584426306e-02	1.199395839924791e+01	1.198792085415574e+01
6.041600752109451e-03	1.811669093641410e-02	1.199395839924788e+01	1.198188330906359e+01
6.041600752112014e-03	2.212821029887158e-02	1.199395839924788e+01	1.197787178970113e+01
6.041600752115364e-03	2.415423602856515e-02	1.199395839924787e+01	1.197584576397144e+01
6.041602417087011e-03	2.816575539102262e-02	1.199395839911458e+01	1.197183424460898e+01
1.207914584424712e-02	3.420330048317366e-02	1.198792085415576e+01	1.196579669951683e+01
1.207914584426266e-02	3.616855663748186e-02	1.198792085415573e+01	1.196383144336252e+01
1.207914584426319e-02	3.821481984563135e-02	1.198792085415573e+01	1.196178518015437e+01
1.207914584426506e-02	4.220610172963291e-02	1.198792085415573e+01	1.195779389827037e+01

Depending on the value of k, the multiplicity of the eigenvalues will change. However, the accruacy of the smallest and largest values seems to not change much based on the value of k.