Mikhail Gaerlan
21 February 2018
MAT 226B Freund

Problem 1

(a)

Let $x_j = hj$, $y_k = hk$, and $h = 1/(m+1)$ for $j,k = 1,\ldots,m$, and let $b_{0j} = b_0(x_j)$, $b_{1j} = b_1(x_j)$, $c_{0k} = c_0(y_k)$, and $c_{1k} = c_1(y_j)$. To include the boundary conditions, we include new matrices $B \in \mathbb{R}^{m\times m}$ and $C \in \mathbb{R}^{m\times m}$ such that

$$B = \begin{bmatrix} b_{01} & b_{02} & \cdots & b_{0m} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \\ b_{11} & b_{12} & \cdots & b_{1m} \end{bmatrix} \qquad C = \begin{bmatrix} c_{01} & 0 & \cdots & 0 & c_{11} \\ c_{02} & 0 & \cdots & 0 & c_{12} \\ \vdots & \vdots & & \vdots & \vdots \\ c_{0m} & 0 & \cdots & 0 & c_{1m} \end{bmatrix}.$$

Then, $T_m V + V T_m - B - C = h^2 F$, so

$$\begin{aligned} F' &= Z^T \left(h^2 F + B + C\right) Z \\ v'_{jk} &= \frac{f'_{jk}}{\lambda_j + \lambda_k}, \quad \text{for all } j,k = 1,2,\ldots,m \\ V &= Z V' Z^T \end{aligned}$$

(b)

```
function V = solvePoisson(m,F,b0,b1,c0,c1)
%Solve two-dimensional Poisson's equation
%   input  - m             matrix size
%            F             right-hand side
%            b0,b1,c0,c1   boundary conditions
%   output - V             solution matrix
h = 1/(m+1);G = h^2*F;
G(1,:) = G(1,:)+b0; G(end,:) = G(end,:)+b1;
G(:,1) = G(:,1)+c0; G(:,end) = G(:,end)+c1;
G = multZ(h,multZ(h,G.').');
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
```

(c)

$$
\begin{aligned}
v(x,0) &= b_0(x) = 0 \\
v(x,1) &= b_1(x) = \sin(\beta\pi x)\cos(\gamma\pi) \\
v(0,y) &= c_0(y) = 0 \\
v(1,y) &= c_1(y) = y^\alpha \sin(\beta\pi)\cos(\gamma\pi y) \\
\frac{\partial v(x,y)}{\partial x} &= \beta\pi y^\alpha \sin(\beta\pi x)\sin(\gamma\pi y) \\
\frac{\partial^2 v(x,y)}{\partial x^2} &= -\beta^2\pi^2 y^\alpha \sin(\beta\pi x)\cos(\gamma\pi y) \\
\frac{\partial v(x,y)}{\partial y} &= \alpha y^{\alpha-1}\sin(\beta\pi x)\cos(\gamma\pi y) - \gamma\pi y^\alpha \sin(\beta\pi x)\sin(\gamma\pi y) \\
\frac{\partial^2 v(x,y)}{\partial y^2} &= \alpha(\alpha-1)y^{\alpha-2}\sin(\beta\pi x)\cos(\gamma\pi y) - \alpha\gamma\pi y^{\alpha-1}\sin(\beta\pi x)\sin(\gamma\pi y) \\
&\quad -\gamma\pi\alpha y^{\alpha-1}\sin(\beta\pi x)\sin(\gamma\pi y) - \gamma^2\pi^2 y^\alpha \sin(\beta\pi x)\cos(\gamma\pi y) \\
f(x,y) &= \beta^2\pi^2 y^\alpha \sin(\beta\pi x)\cos(\gamma\pi y) - \alpha(\alpha-1)y^{\alpha-2}\sin(\beta\pi x)\cos(\gamma\pi y) \\
&\quad +2\alpha\gamma\pi y^{\alpha-1}\sin(\beta\pi x)\sin(\gamma\pi y) + \gamma^2\pi^2 y^\alpha \sin(\beta\pi x)\cos(\gamma\pi y)
\end{aligned}
$$

(d)

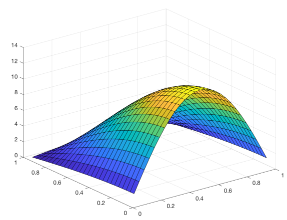
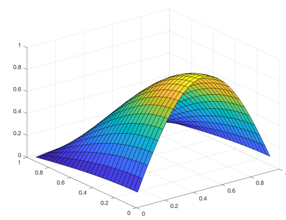
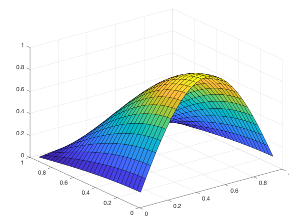
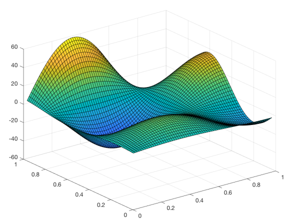
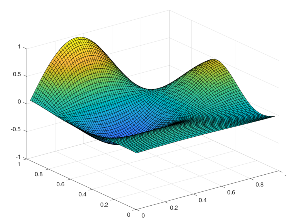
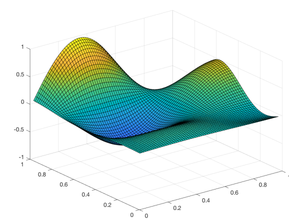
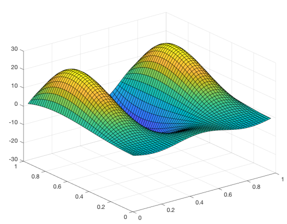
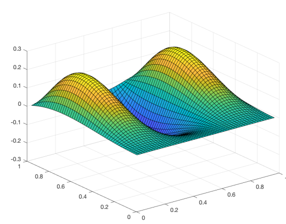
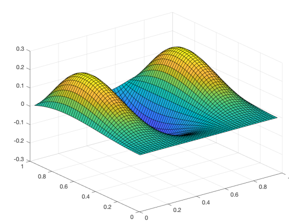
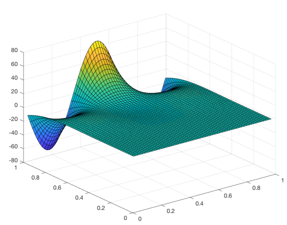
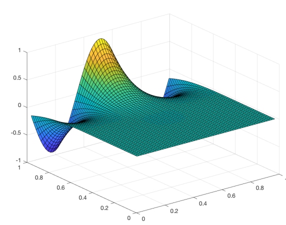
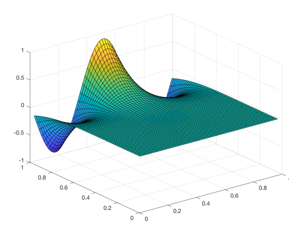
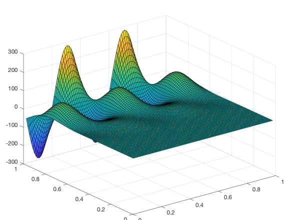
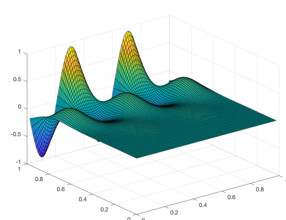
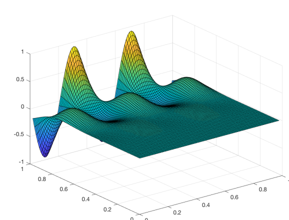
```
params = ...
    [0 1   0.5  ;...
    1 1.5 2  ;...
    2 3   0.5;...
    5 3   1  ;...
    5 5   3  ];
n = size(params,1); errors = zeros(n,1); ms = zeros(n,1);
for i = 1:n
    for m = 10:1000
        h = 1/(m+1);
        a=params(i,1);b=params(i,2);g=params(i,3);
        [X,Y]=meshgrid(h:h:1-h);
        x = X(1,:);y = Y(:,1);
        F = f(X,Y,a,b,g);
```

```
        b0 = v(x,0,a,b,g);b1 = v(x,1,a,b,g);
        c0 = v(0,y,a,b,g);c1 = v(1,y,a,b,g);
        Vapprox = solvePoisson(m,F,b0,b1,c0,c1);
        Vactual = v(X,Y,a,b,g);
        error = max(abs(Vapprox(:)-Vactual(:)));
        if error < 5.0e-4
            errors(i)=error; ms(i)=m;
            surf(X,Y,F);saveas(gcf,strcat("../Figures/poisson_rhs_",int2str(i),".png"));
            surf(X,Y,Vapprox);saveas(gcf,strcat("../Figures/poisson_approx_",int2str(i),".png"));
            surf(X,Y,Vactual);saveas(gcf,strcat("../Figures/poisson_actual_",int2str(i),".png"));
            break;
        end
    end
end
matrix2latex(errors,'../Tables/poissonerrors.tex','alignment','r','format','%-.15e');
matrix2latex(ms,'../Tables/poissonms.tex','alignment','r','format','%-4d');

function true = v(x,y,a,b,g)
true = y.^a.*sin(b.*pi.*x).*cos(g.*pi.*y);
end

function fun = f(x,y,a,b,g)
fun = b.^2.*pi.^2.*y.^a.*sin(b.*pi.*x).*cos(g.*pi.*y)...
    -a.*(a-1).*y.^(a-2).*sin(b.*pi.*x).*cos(g.*pi.*y)...
    +2.*a.*g.*pi.*y.^(a-1).*sin(b.*pi.*x).*sin(g.*pi.*y)...
    +g.^2.*pi.^2.*y.^a.*sin(b.*pi.*x).*cos(g.*pi.*y);
end
```

case	m	error
i)	27	4.728617202247598e-04
ii)	68	4.988921326820606e-04
iii)	53	4.837994973282411e-04
iv)	58	4.914479216866496e-04
v)	86	4.983532030559124e-04

case	right-hand side	actual	approximation
i)			
ii)			
iii)			
iv)			
v)			

Problem 2

(a)

The centered-difference approximation for the partial difference equations has the form:

$$\begin{aligned}
\frac{2v_{j,k}-v_{j-1,k}-v_{j+1,k}}{h_x^2}+\frac{2v_{j,k}-v_{j,k-1}-v_{j,k+1}}{h_y^2}+\sigma v_{j,k} &= f\left(x_j,y_k\right)\\
\frac{1}{h_x^2}T_{m_x}V+\frac{1}{h_y^2}VT_{m_y}-\frac{1}{h_x^2}B-\frac{1}{h_y^2}C+\sigma V &= F\\
T_{m_x}V+\frac{h_x^2}{h_y^2}VT_{m_y}+h_x^2\sigma V &= h_x^2\left(F+\frac{1}{h_x^2}B+\frac{1}{h_y^2}C\right)\\
T_{m_x}V+\alpha VT_{m_y}+\beta V &= \tilde{F}
\end{aligned}$$

where

$$\alpha=\left(\frac{h_x}{h_y}\right)^2 \qquad \beta=h_x^2\sigma$$

$$\tilde{F}=h_x^2\left(F+\frac{1}{h_x^2}B+\frac{1}{h_y^2}C\right)$$

$$B=\begin{bmatrix} b_{01} & b_{02} & \cdots & b_{0m}\\ 0 & 0 & \cdots & 0\\ \vdots & \vdots & & \vdots\\ 0 & 0 & \cdots & 0\\ b_{11} & b_{12} & \cdots & b_{1m}\end{bmatrix} \qquad C=\begin{bmatrix} c_{01} & 0 & \cdots & 0 & c_{11}\\ c_{02} & 0 & \cdots & 0 & c_{12}\\ \vdots & \vdots & & \vdots & \vdots\\ c_{0m} & 0 & \cdots & 0 & c_{1m}\end{bmatrix}$$

$$b_{0j}=g\left(x_j,0\right) \qquad c_{0k}=g\left(0,y_k\right)$$

$$b_{1j}=g\left(x_j,b\right) \qquad c_{1k}=g\left(a,y_k\right)$$

(b)

$$\begin{aligned}
T_{m_x}V+\alpha VT_{m_y}+\beta V &= \tilde{F}\\
Z_{m_x}^TT_{m_x}Z_{m_x}Z_{m_x}^TVZ_{m_y}+Z_{m_x}^TVZ_{m_y}Z_{m_y}^TVZ_{m_y}+\beta Z_{m_x}^TVZ_{m_y} &= Z_{m_x}^T\tilde{F}Z_{m_y}^T\\
\Lambda_{m_x}V'+\alpha V'\Lambda_{m_y}+\beta V' &= \tilde{F}'\\
\lambda_jv'_{jk}+\alpha v'_{jk}\lambda_k+\beta v'_{jk} &= \tilde{f}'_{jk}\\
v'_{jk} &= \frac{\tilde{f_{jk}}'}{\lambda_j+\alpha\lambda_k+\beta}\\
V &= Z_{m_x}^TV'Z_{m_y}
\end{aligned}$$

The number of flops would be of order $\mathcal{O}\left(m_xm_y\log\left(m_x\right)+m_ym_x\log\left(m_x\right)\right)=\mathcal{O}\left(m_xm_y\log\left(m_xm_y\right)\right)$.

Problem 3

Let $c = [c_0, c_1, c_2, \ldots, c_{n-1}]^T$. The Frobenius norm can be written as

$$\begin{aligned}
\|C - T\|_F &= \operatorname{trace}\left((C-T)^T (C-T)\right) \\
&= \operatorname{trace}\left(\left(C^T - T^T\right)(C-T)\right) \\
&= \operatorname{trace}\left(C^T C - C^T T - T^T C - T^T T\right)
\end{aligned}$$

Thus,

$$\begin{aligned}
P = [p_{ij}]_{i,j=0,\ldots,n-1} &= C^T C \\
&= \begin{bmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ c_2 & c_1 & c_0 & \cdots & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & c_{n-3} & \cdots & c_0 \end{bmatrix} \begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & \cdots & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & \cdots & c_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_1 & c_2 & c_3 & \cdots & c_0 \end{bmatrix} \\
p_{ii} &= c_i^2 + c_{i-1}^2 + \cdots + c_1^2 + c_0^2 + c_{n-1}^2 + c_{n-2}^2 + \cdots + c_{i+1}^2 \\
&= c^T c \\
\operatorname{trace}(P) &= \sum_{i=0}^{n-1} p_{ii} \\
&= n c^T c
\end{aligned}$$

$$\begin{aligned}
Q = [q_{ij}]_{i,j=0,\ldots,n-1} &= C^T T \\
&= \begin{bmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ c_2 & c_1 & c_0 & \cdots & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & c_{n-3} & \cdots & c_0 \end{bmatrix} \begin{bmatrix} t_0 & t_1 & t_2 & \cdots & t_{n-1} \\ t_1 & t_0 & t_1 & \cdots & t_{n-2} \\ t_2 & t_1 & t_0 & \cdots & t_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & t_{n-2} & t_{n-3} & \cdots & t_0 \end{bmatrix} \\
q_{ii} &= c_i t_i + c_{i-1} t_{i-1} + \cdots + c_1 t_1 + c_0 t_0 + c_{n-1} t_1 + c_{n-2} t_2 + \cdots + c_{i+1} t_{n-i-1} \\
&= \sum_{k=0}^{i} c_{i-k} t_{i-k} + \sum_{k=1}^{n-i-1} c_{n-k} t_k \\
\operatorname{trace}(Q) &= \sum_{i=0}^{n-1} q_{ii} \\
&= \sum_{i=0}^{n-1} \left(\sum_{k=0}^{i} c_{i-k} t_{i-k} + \sum_{k=1}^{n-i-1} c_{n-k} t_k \right)
\end{aligned}$$

Similarly,

$$
\begin{aligned}
R = [r_{ij}]_{i,j=0,\ldots,n-1} &= T^T C \\
&= \begin{bmatrix} t_0 & t_1 & t_2 & \cdots & t_{n-1} \\ t_1 & t_0 & t_1 & \cdots & t_{n-2} \\ t_2 & t_1 & t_0 & \cdots & t_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & t_{n-2} & t_{n-3} & \cdots & t_0 \end{bmatrix} \begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & \cdots & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & \cdots & c_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_1 & c_2 & c_3 & \cdots & c_0 \end{bmatrix} \\
r_{ii} &= t_i c_i + t_{i-1} c_{i-1} + \cdots + t_1 c_1 + t_0 c_0 + t_1 c_1 + t_2 c_2 + \cdots + t_{n-i-1} c_{n-i-1} \\
&= \sum_{k=0}^{i} t_{i-k} c_{i-k} + \sum_{k=1}^{n-i-1} t_k c_k \\
\operatorname{trace}(R) &= \sum_{i=0}^{n-1} r_{ii} \\
&= \sum_{i=0}^{n-1} \left(\sum_{k=0}^{i} t_{i-k} c_{i-k} + \sum_{k=1}^{n-i-1} t_k c_k \right)
\end{aligned}
$$

Thus, $\operatorname{trace}(Q) = \operatorname{trace}(R)$. Let $\nabla = \left[\frac{\partial}{\partial c_1}, \ldots, \frac{\partial}{\partial c_{n-1}}\right]$. To minimize $\|C - T\|_F$, set

$$
\begin{aligned}
(\nabla \|C - T\|_F)_j &= \frac{\partial}{\partial c_j} \operatorname{trace}\left(C^T C - C^T T - T^T C - T^T T\right) \\
&= \frac{\partial}{\partial c_j} \left[\sum_{k=0}^{n-1} c_k^2 - 2 \sum_{i=0}^{n-1} \left(\sum_{k=0}^{i} c_{i-k} t_{i-k} + \sum_{k=1}^{n-i-1} c_{n-k} t_k \right) \right] \\
&= 2n c_j - 2\left[(n-j)\, t_j + j t_{n-j}\right] \\
&= 0 \\
c_j &= \frac{1}{n}\left[(n-j)\, t_j + j t_{n-j}\right]
\end{aligned}
$$

Since there is only one solution to $\nabla \|C - T\|_F = 0$, then C_T is unique. Also, since

$$
\begin{aligned}
c_{n-j} &= \frac{1}{n}\left[(n - (n-j))\, t_{n-j} + (n-j)\, t_{n-(n-j)}\right] \\
&= \frac{1}{n}\left[j t_{n-j} + (n-j)\, t_j\right] \\
&= c_j,
\end{aligned}
$$

then c_j is symmetric.

Problem 4

(a)

```
function x = toeplitzpcg(t,b)
%Solve a system Tx=b where T is symmetric positive-definite Toeplitz.
%   input  - t      row vector for symetric Toeplitz matrix, T
%          - b      column vector
%          - tol    tolerance
%          - maxit  maximum iterations
%   output - x      vector
    x = pcg(@toeplitzm,b,1e-9,10000);

    function y = toeplitzm(x)
        y = toeplitz([t(end:-1:2),t],x);
    end
end
```

(b)

```
function x = toeplitzpcgcirc(t,b)
%Solve a system Tx=b where T is symmetric positive-definite Toeplitz.
%Uses a right preconditioner C^-1.
%   input  - t      row vector for symetric Toeplitz matrix, T
%          - b      column vector
%          - tol    tolerance
%   output - x      solution
n = length(b); c = ((n:-1:1).*t+(0:(n-1)).*t([1,end:-1:2]))/n;
x = pcg(@toeplitzm,b,1e-9,10000,@circulantinv);

    function y = circulantinv(xx)
        y = conj(fft(conj(((conj(fft(c'))).^(-1)).*fft(xx))))/n;
    end

    function y = toeplitzm(xx)
        y = toeplitz([t(end:-1:2),t],xx);
    end
end
```

(c)

```
n=10;b=ones(n,1);for p=[1,0.1,0.01]
    t=(1+sqrt(0:(n-1))).^(-p);
    matrix2latex(toeplitzpcg(t,b),strcat("../Tables/pcg_",sprintf('%03d',100*p),".tex"),'alignment','r','format','%-.15e')
    matrix2latex(toeplitzpcgcirc(t,b),strcat("../Tables/pcgcirc_",sprintf('%03d',100*p),".tex"),'alignment','r','format','%-.15e')
end

n=1e6;b=ones(n,1);for p=[1,0.1]
    t=(1+sqrt(0:(n-1))).^(-p);
    x = toeplitzpcg(t,b);
    matrix2latex(real(x([1,100000,500000,700000,1000000])),strcat("../Tables/pcglarge_",sprintf('%03d',100*p),".tex"),'alignment','r','format','%-.15e')
    x = toeplitzpcgcirc(t,b);
    matrix2latex(real(x([1,100000,500000,700000,1000000])),strcat("../Tables/pcgcirclarge_",sprintf('%03d',100*p),".tex"),'alignment','r','format','%-.15e')
end
```

For $n = 10$, the output is,

$p = 1$	$p = 0.1$	$p = 0.01$
without preconditioning		
3.450926863794943e-01	1.975370703537823e-01	1.857935191671665e-01
2.428161529760083e-01	1.138792774427789e-01	1.045415317727272e-01
2.061924099579956e-01	8.775407132745998e-02	7.961205681990180e-02
1.896360060540769e-01	7.691072994649119e-02	6.937790019383741e-02
1.828445601954354e-01	7.265683421537614e-02	6.538557146064886e-02
1.828445601954350e-01	7.265683421537471e-02	6.538557146065589e-02
1.896360060540770e-01	7.691072994648715e-02	6.937790019375377e-02
2.061924099579958e-01	8.775407132745379e-02	7.961205681977551e-02
2.428161529760081e-01	1.138792774427705e-01	1.045415317726247e-01
3.450926863794944e-01	1.975370703537805e-01	1.857935191670622e-01
with preconditioning		
3.450926863795118e-01	1.975370703537795e-01	1.857935201528620e-01
2.428161529760274e-01	1.138792774427788e-01	1.045415204154867e-01
2.061924099580156e-01	8.775407132745931e-02	7.961210248459726e-02
1.896360060540973e-01	7.691072994648539e-02	6.937782132363908e-02
1.828445601954564e-01	7.265683421537593e-02	6.538561503746061e-02
1.828445601954559e-01	7.265683421537601e-02	6.538561503743219e-02
1.896360060540976e-01	7.691072994648594e-02	6.937782132361736e-02
2.061924099580162e-01	8.775407132745613e-02	7.961210248460469e-02
2.428161529760276e-01	1.138792774427781e-01	1.045415204154808e-01
3.450926863795120e-01	1.975370703537801e-01	1.857935201529732e-01

For $n = 100$, the output is,

	$p = 1$	$p = 0.1$
without preconditioning		
$x(1)$	1.676284540637890e-02	9.532317527874454e-04
$x(100000)$	4.147019469754698e-04	1.985347566261856e-06
$x(500000)$	3.202634777824108e-04	1.221933191103328e-06
$x(700000)$	3.346753478040451e-04	1.327454413917721e-06
$x(1000000)$	1.676284540576385e-02	9.532317519653615e-04
with preconditioning		
$x(1)$	1.676284622516175e-02	9.532780564171566e-04
$x(100000)$	4.147019516876527e-04	1.985338116172135e-06
$x(500000)$	3.202634824103388e-04	1.221936166062850e-06
$x(700000)$	3.346753506434133e-04	1.327449171076912e-06
$x(1000000)$	1.676284622546426e-02	9.532780564631677e-04

Problem 5

(a)

```
function [J,I] = get_lower(A)
%Find the sparse column format of a the lower-triangular part of a matrix
%   Input  - A  sparse matrix
%   Output - J  row indices
%            I  column pointers
%            V  nonzero entries
[J,K] = find(tril(A)); I = find(J-K==0);I = [I;nnz(tril(A))+1];
end
```

(b)

```
function V = iCholesky(A,J,I)
%Find the elements of an incomplete Cholesky matrix
%   Input  - A  sparse matrix
%            J  row indices
%            I  column pointers
%   Output - V  entries of incomplete Cholesky factorization
n = size(A,1); [~,~,V] = find(tril(A));
for k = 1:n
    if V(I(k)) <= 0; break; end
    V(I(k))=sqrt(V(I(k)));
    indJ = (I(k)+1):(I(k+1)-1);
    V(indJ) = V(indJ)/V(I(k));
    for j = indJ
        indI = I(J(j)):(I(J(j)+1)-1);
        [~,rowsJ,rowsI] = intersect(J(indJ),J(indI));
        rowsJ = rowsJ + I(k); rowsI = rowsI + I(J(j)) - 1;
        vj = V(j); V(rowsI) = V(rowsI) - vj*V(rowsJ);
    end
end
end
```

(c)

```
function x = solve_lower(c,J,I,V)
%Solve Lx = c where L is a lower-triangular matrix
%   Input  - J  row indices
%            I  pointers
%            V  nonzero entries
%            c  right-hand side
%   Output - x  solution
x = c;
for k = 1:length(x)
    x(k) = x(k)/V(I(k));
    indices = (I(k)+1):(I(k+1)-1); rows = J(indices);
    x(rows) = x(rows) - x(k)*V(indices);
end
end
```

(d)

```
function c = solve_lowert(b,J,I,V)
%Solve L^Tc=b where L is lower-triangular
%   Input  - J  row indices
%            I  pointers
%            V  nonzero entries
%            b  right-hand side
%   Output - c  solution
c = b;c(end) = c(end)/V(end);
for k = (length(c)-1):-1:1
    columns = (I(k)+1):(I(k+1)-1); rows = J(columns);
    c(k) = (c(k) - sum(V(columns).*c(rows)))/V(I(k));
end
end
```

(e)

```matlab
clear variables; load('pcg_small.mat'); global J I V; n = size(A,1);
[x,~,~,iter] = pcg(A,b,1e-9,1000,[],[],ones(n,1));
matrix2latex(x,"../Tables/snocond.tex",'alignment','r','format','%-.15e');
fileID = fopen('../Tables/snoconditer.tex','w'); fprintf(fileID,'%d',iter);fclose(fileID);
fileID = fopen('../Tables/snocondnorm.tex','w'); fprintf(fileID,'%.15e',norm(A*x-b)/norm(b));fclose(fileID);

[J,I]=get_lower(A); V = iCholesky(A,J,I); col = 3; m = length(J);
rows = ceil(m/col); gap = rows*col-m;
[x,~,~,iter] = pcg(A,b,1e-9,10000,@solve_lowerr,@solve_lowertt,ones(n,1));
matrix2latex(x,"../Tables/scond.tex",'alignment','r','format','%-.15e');
fileID = fopen('../Tables/sconditer.tex','w'); fprintf(fileID,'%d',iter);fclose(fileID);
fileID = fopen('../Tables/scondnorm.tex','w'); fprintf(fileID,'%.15e',norm(A*x-b)/norm(b));fclose(fileID);
for j = (m+1):(m+gap); J(j) = NaN; V(j) = NaN; end
matrix2latex(reshape(J,[rows,col]),"../Tables/scondJ.tex",'alignment','r','format','%-d');
matrix2latex(reshape(I,[13,2]),"../Tables/scondI.tex",'alignment','r','format','%-d');
matrix2latex(reshape(V,[rows,col]),"../Tables/scondV.tex",'alignment','r','format','%-.15e');

load('pcg_large.mat'); n = size(A,1);
[x,~,~,iter] = pcg(A,b,1e-9,10000,[],[],ones(n,1));
matrix2latex(x([1,10000,100000,20000,262144]),"../Tables/lnocond.tex",'alignment','r','format','%-.15e');
fileID = fopen('../Tables/lnoconditer.tex','w'); fprintf(fileID,'%d',iter);fclose(fileID);

[J,I]=get_lower(A); V = iCholesky(A,J,I);
[x,~,~,iter] = pcg(A,b,1e-9,10000,@solve_lowerr,@solve_lowertt,ones(n,1));
matrix2latex(x([1,10000,100000,20000,262144]),"../Tables/lcond.tex",'alignment','r','format','%-.15e');
fileID = fopen('../Tables/lconditer.tex','w'); fprintf(fileID,'%d',iter);fclose(fileID);
matrix2latex(J([1,10,100,1000,end]),"../Tables/lcondJ.tex",'alignment','r','format','%-d');
matrix2latex(I([1,10,100,1000,end]),"../Tables/lcondI.tex",'alignment','r','format','%-d');
matrix2latex(V([1,10,100,1000,end]),"../Tables/lcondVL.tex",'alignment','r','format','%-.15e');

function y=solve_lowerr(xx); global J I V; y=solve_lower(xx,J,I,V); end
function y=solve_lowertt(xx); global J I V; y=solve_lowert(xx,J,I,V); end
```

The small matrix had the following output.

	without preconditioning	with preconditioning
iterations	13	11
$\frac{\|Ax-b\|_2}{\|b_2\|}$	4.668821516582010e-16	3.024858452994199e-10
x	$\begin{bmatrix} \texttt{-4.615232329669224e-02} \\ \texttt{5.560346406107559e-02} \\ \texttt{4.674441847347385e-01} \\ \texttt{1.436167453017343e-01} \\ \texttt{-2.747396794324581e+00} \\ \texttt{-9.595158505895890e-01} \\ \texttt{-5.588795197458957e-01} \\ \texttt{-5.625708181717426e-01} \\ \texttt{4.963846660536811e-01} \\ \texttt{4.553705821620747e-01} \\ \texttt{-8.540096449002311e-01} \\ \texttt{1.546983563153979e-01} \\ \texttt{7.076179967126332e-02} \\ \texttt{-7.644048349576092e-02} \\ \texttt{-2.310798723702473e+00} \\ \texttt{-1.815291036261631e+00} \\ \texttt{-1.576605857315947e+00} \\ \texttt{-2.519078345042125e+00} \\ \texttt{-9.076946955455132e-01} \\ \texttt{-1.120184575920670e+00} \\ \texttt{-1.946389613822066e-01} \\ \texttt{1.198432367141279e-01} \\ \texttt{-1.119327367250933e+00} \\ \texttt{-3.088950654587257e+00} \\ \texttt{-7.670423173687755e-01} \end{bmatrix}$	$\begin{bmatrix} \texttt{-4.615232331993605e-02} \\ \texttt{5.560346408639215e-02} \\ \texttt{4.674441847709141e-01} \\ \texttt{1.436167453602924e-01} \\ \texttt{-2.747396794334826e+00} \\ \texttt{-9.595158508044294e-01} \\ \texttt{-5.588795196477444e-01} \\ \texttt{-5.625708181081200e-01} \\ \texttt{4.963846659080517e-01} \\ \texttt{4.553705824433909e-01} \\ \texttt{-8.540096447547476e-01} \\ \texttt{1.546983564603329e-01} \\ \texttt{7.076179962288556e-02} \\ \texttt{-7.644048328883853e-02} \\ \texttt{-2.310798723518883e+00} \\ \texttt{-1.815291036181375e+00} \\ \texttt{-1.576605856924276e+00} \\ \texttt{-2.519078344987517e+00} \\ \texttt{-9.076946956506157e-01} \\ \texttt{-1.120184576068447e+00} \\ \texttt{-1.946389613226679e-01} \\ \texttt{1.198432364684508e-01} \\ \texttt{-1.119327367504965e+00} \\ \texttt{-3.088950654440442e+00} \\ \texttt{-7.670423169847281e-01} \end{bmatrix}$

The vectors J, I, and V_L correspond to the CSC format of the incomplete Cholesky factorization of A and were reshaped to fit onto the page.

$$J = \begin{bmatrix} 1 & 7 & 18 \\ 2 & 21 & 23 \\ 9 & 24 & 24 \\ 21 & 8 & 16 \\ 2 & 11 & 17 \\ 6 & 23 & 20 \\ 7 & 9 & 17 \\ 3 & 13 & 19 \\ 10 & 10 & 21 \\ 20 & 16 & 24 \\ 4 & 22 & 18 \\ 6 & 11 & 25 \\ 7 & 12 & 19 \\ 8 & 14 & 20 \\ 15 & 25 & 20 \\ 5 & 13 & 21 \\ 16 & 19 & 22 \\ 18 & 21 & 25 \\ 22 & 14 & 23 \\ 24 & 18 & 24 \\ 6 & 23 & 25 \\ 11 & 15 & \mathtt{NaN} \end{bmatrix} \qquad I = \begin{bmatrix} 1 & 41 \\ 5 & 44 \\ 8 & 48 \\ 11 & 51 \\ 16 & 55 \\ 21 & 57 \\ 23 & 59 \\ 26 & 60 \\ 29 & 61 \\ 31 & 63 \\ 34 & 64 \\ 35 & 65 \\ 38 & 66 \end{bmatrix}$$

$$V_L = \begin{bmatrix} 2.000000000000000e{+}00 & 1.866369023889256e{+}00 & -5.163977794943222e{-}01 \\ -5.000000000000000e{-}01 & -5.357997197768198e{-}01 & -5.163977794943222e{-}01 \\ -5.000000000000000e{-}01 & -5.357997197768198e{-}01 & -5.163977794943222e{-}01 \\ -5.000000000000000e{-}01 & 1.936491673103709e{+}00 & 1.866369023889256e{+}00 \\ 1.936491673103709e{+}00 & -5.163977794943222e{-}01 & -5.357997197768198e{-}01 \\ -5.163977794943222e{-}01 & -5.163977794943222e{-}01 & -5.357997197768198e{-}01 \\ -5.163977794943222e{-}01 & 1.936491673103709e{+}00 & 1.926893525934186e{+}00 \\ 2.000000000000000e{+}00 & -5.163977794943222e{-}01 & -5.189700346910373e{-}01 \\ -5.000000000000000e{-}01 & 1.936491673103709e{+}00 & -5.189700346910373e{-}01 \\ -5.000000000000000e{-}01 & -5.163977794943222e{-}01 & -5.189700346910373e{-}01 \\ 2.000000000000000e{+}00 & -5.163977794943222e{-}01 & 1.793506806975281e{+}00 \\ -5.000000000000000e{-}01 & 1.856408358530099e{+}00 & -5.575668829974966e{-}01 \\ -5.000000000000000e{-}01 & 2.000000000000000e{+}00 & 1.860863498549972e{+}00 \\ -5.000000000000000e{-}01 & -5.000000000000000e{-}01 & -5.373849295121450e{-}01 \\ -5.000000000000000e{-}01 & -5.000000000000000e{-}01 & 1.781610534830862e{+}00 \\ 2.000000000000000e{+}00 & 1.932183566158592e{+}00 & 1.710477015490919e{+}00 \\ -5.000000000000000e{-}01 & -5.175491695067657e{-}01 & 1.866369023889256e{+}00 \\ -5.000000000000000e{-}01 & -5.175491695067657e{-}01 & -5.357997197768198e{-}01 \\ -5.000000000000000e{-}01 & 1.936491673103709e{+}00 & 1.788854381999832e{+}00 \\ -5.000000000000000e{-}01 & -5.163977794943222e{-}01 & 1.710824975476217e{+}00 \\ 1.866369023889256e{+}00 & -5.163977794943222e{-}01 & 1.775397936033366e{+}00 \\ -5.357997197768198e{-}01 & 1.936491673103709e{+}00 & \mathtt{NaN} \end{bmatrix}$$

The large matrix had the following output where J, I, and V_L correspond to CSC format of the the incoomplete Cholesky factorization.

	without preconditioning	with preconditioning
iterations	249	111
$x\,(1)$	4.286321404907984e-01	4.286321980474398e-01
$x\,(10000)$	7.121924304536499e-01	7.121923784405517e-01
$x\,(100000)$	-2.453852875592730e+00	-2.453852852351424e+00
$x\,(200000)$	-2.497480687000048e+00	-2.497480719783645e+00
$x\,(262144)$	1.938920155215448e+00	1.938920151693724e+00

index	J	I	V_L
1	1	1	2.449489742783178e+00
10	45882	63	-4.082482904638631e-01
100	132301	685	-4.082482904638631e-01
1000	102399	6887	-4.082482904638631e-01
end	262144	1036289	2.220117073041080e+00