Lab 4, April 6, 2016
Numerical Linear Algebra, Spring 2016

1. Apply the (unpreconditioned) CG algorithm to the linear system $A\mathbf{x} = \mathbf{b}$ with symmetric and positive definite (SPD) matrix A using the following code and function cgsolve.m. Modify the code to plot the residual norms versus iteration number (you may use semilogy or plot).

```
n = 1000;
A = randn(n);
A = A'*A; % making A symmetric and positive definite
b = randn(n,1);
maxIter = 10000;
x0 = zeros(n,1); % initial guess
tic;
[x, numIter] = cgsolve(A, b, x0, maxIter)
toc;
fprintf('Residual is %s\n', norm(b-A*x));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x, numIter] = cgsolve(A, b, x0, maxIter);
% cgsolve : Solve Ax=b by conjugate gradients
% [x, numIter] = cgsolve(A, b, maxIter);
% Given symmetric positive definite matrix A and vector b,
% this runs conjugate gradient to solve for x in Ax=b.
% It iterates until the residual norm is reduced by 10^-6,
% or for at most maxIter number of iterations.
%
n = length(b);
r = b - A*x0;
rTr = r'*r;
p = r;
x = x0;
numIter = 0;

while norm(r) > 10^-6 & numIter < maxIter
    numIter = numIter + 1;
    Ap = A*p;
    a = rTr/(p'*Ap);
    x = x + a*p;
```

```
    r = r - a*Ap;
    rTr_old = rTr;
    rTr = r'*r;
    beta = rTr/rTr_old;
    p = r + beta*p;
end
```

2. Apply the CG algorithm to the $n \times n$ matrix

$$A = \begin{pmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \dots & 0 \\ & & \dots & & \\ 0 & 0 & \dots & & -1 \\ 0 & 0 & \dots & -1 & 2 \end{pmatrix}$$

(you have to generate it first, use size $n = 10, 20, 30, ...$) and $\mathbf{b} = (b_1, b_2, \dots, b_n)^T$ with $b_i = (i/(n+1))^2$. Take the initial guess to be the zero vector. (In Chapter 13, we will see that this problem corresponds to a finite difference method for the two-point boundary value problem.)