MA/CS 615 Spring 2019 Homework #5

Due before class starts on April 11th. Late homework will not be given any credit. Collaboration is OK but not encouraged. Indicate on your report whether you have collaborated with others and whom you have collaborated with.

1. (60 points) *General iterative methods and conjugate gradient method.* In MATLAB, we can use the functions *numgrid* and *delsq* to generate the classical second order discrete Laplacian with homogeneous Dirichlet b.c. on a few more general domains:

```
Length = 10;      % size of geometry

% 'numgrid' generates uniform grid points for the interior
%  of the specified shape.
% other possible shapes include L,S,N,C,D,A,H,B
Domain_grid = numgrid('L',Length); % L shape domain

% 'delsq' generate the second order discrete Laplacian
%  defined on the grid points
A = delsq(Domain_grid);
n = size(A,1);
b = rand(n,1);

figure;
% visualize the domain and grid points
subplot(1,2,1), spy(Domain_grid), title('domain shape')
% visualize the sparsity pattern of the matrix A
subplot(1,2,2), spy(A), title('system matrix')

%% Generate a linear system Ax=b
n = size(A,1);
b = rand(n,1);
x_true = A\b;
```

For example, if we specify the domain as the square, then the obtained matrix above is exactly $K \otimes Id + Id \otimes K$. Use the script above to generate a linear system $Ax = b$ where A is $n \times n$ and b is generated randomly. You can use whatever shape you like and use a proper length such that n is at least around one thousand. You can verify in MATLAB that the matrix A above is indeed symmetric and positive definite:

```
norm(A-A','fro') % If A is real symmetric, then this should be zero
min(eig(A))      % The smallest eigenvalue should be positive
```

The following can be used to compute the decomposition $A = D - L - U$:

```
D=diag(A);
L=-tril(A)+diag(D);
U=D-L-A;
```

Implement the following methods:

- Jacobi iteration: $Dx^{k+1} = (D - A)x^k + b$.
- Weighted Jacobi iteration: $D/wx^{k+1} = (D/w - A)x^k + b$, with $w = \frac{2}{3}$.
- Gauss-Seidel iteration: $(D - L)x^{k+1} = Ux^k + b$.
- Successive overrelaxation (SOR):

$$\frac{1}{w}(D - wL)x^{k+1} = \frac{1}{w}[(1-w)D + wU]x^k + b,$$

  where w can be chose as a number close to 2. You can try different w, e.g., use $w = 1.9$.
- Steepest Descent for minimizing the cost function $f(x) = \frac{1}{2}x^T Ax - x^T b$.
- Conjuage Gradient (CG) method for solving $Ax = b$.
- Preconditioned Conjuage Gradient (PCG): use CG to solve $PAP^T y = Pb$. Use the incomplete Cholesky factorization as the preconditioner, i.e., if $A \approx \tilde{L}\tilde{L}^T$ then set $P = \tilde{L}^{-1}$. Use the function *ichol* for the incomplete Cholesky factorization. You can check how good the approximation $\tilde{L}\tilde{L}^T$ is and the sparsity pattern of $\tilde{L}$:

```
Lt=ichol(A);
spy(Lt);
norm(A-Lt*Lt','fro')/norm(A)
```

Store the following quantities for each iteration:

1. Relative error $\|x^k - x\|/\|x\|$.
2. Relative residue $\|b - Ax^k\|/\|b\|$.
3. For CG and Steepest Descent only: cost function $f(x_k)$ where $f(x) = \frac{1}{2}x^T Ax - x^T b$.

Use zero initial guesss thus the initial relative error and the initial relative residue are both one for each method. For CG and PCG, stop the iteration if the relative residue is smaller than some parameter, e.g., 10^{-14} or 10^{-15}. For all other methods, set the iteration number as $n/2$ where n is the size of the vector b.

Explain what parameter you have used for SOR and the size of your matrix A. Show the following four figures:

1. One figure of the domain grid points.
2. One figure of semilogy plot of relative residue for seven methods. Add legend in the figure by the following

```
legend('Steepest Descent','Jacobi','Weighted ...
   Jacobi','Gauss-Seidel','SOR','CG','PCG');
```

3. One figure of semilogy plot of relative error for seven methods.

4. One figure of the loglog plot of the cost function for CG and SD only. Notice that in CG and SD, the cost function is supposed to decrease as iteration number increases, while the error and residue are not necessarily monotonically descreasing.

2. (40 points) *Multigrid methods.* Consider using the second order finite difference solving the 2D Poisson equation $-u_{xx} - u_{yy} = f(x, y)$ with nonhomogeneous Dirichlet boundary conditions on $[0, 1] \times [0, 1]$. Let $f(x, y) = 2 \cos x \sin y$, and $u(x, y) = \cos x \sin y$. The Dirichlet boundary conditions of the values of $u(x, y) = \cos x \sin y$ along the boundary of the square. First move the boundary conditions to the right hand side so that you have a linear system $Ax = b$. Use eigenvector method to obtain the exact solution to this system. Verify you have implemented everything correctly by checking the second order accuracy. Then implement a V-cycle and a full multigird V-cycle (FMG) for this problem. Use $n = 2^k - 1$ grid points for each direction so x is a $n^2 \times 1$ vector and A is a $n^2 \times n^2$ matrix. For $n = 2^k - 1$, the interpolation matrix I and restriction matrix R can be constructed as:

```
I = spdiags(ones(n,1)*[1 2 1],-2:0,n,n);
I1 = I(:,1:2:end-1)/2;
I=kron(I1,I1);
R=I'/4;
```

Let A_h be the discrete Laplacian matrix on the fine mesh of size h, then you use either the discrete Laplacian matrix of smaller size or simply the matrix RA_hI as the operator A_{2h} on the coarse mesh of size $2h$. Show a table of relative errors each iteration of V-cycle and FMG for five iterations, and for four different problems $n = 2^k - 1$ with $k = 8, 9, 10, 11$. You can try smaller problems if $n = 2^{11} - 1$ is too large for your implementation or your computer. Also list the relative error of Conjugate Gradient after 100 iterations for the same problem as a comparison. A few remarks:

- Use 5 iterations of weighted Jacobi with $w = \frac{2}{3}$ as the iteration/relaxation in your multigrid method.

- Set the parameter of the coarsest level (on which you solve the system exactly) as $2^3 - 1$. For instance, if $n = 2^5 - 1$, then there are three levels.

3. (Bonus: 20 points). Consider solving a 1D variable coefficient problem:

$$-(a(x)u'(x))' = f(x), \quad x \in [0, 1],$$

with homogeneous Dirichlet boundary conditions. A conservative discretization must be used:

$$-\frac{1}{\Delta x^2}[-a_{j-\frac{1}{2}}u_{j-1} + (a_{j-\frac{1}{2}} + a_{j+\frac{1}{2}})u_j - a_{j+\frac{1}{2}}u_{j+1}] = f_j,$$

where $a_{j-\frac{1}{2}} = a(x_j - \frac{1}{2}\Delta x)$. The matrix vector form of this scheme is $Au = f$ where A is a real symmetric tridiagonal matrix:

$$A = -\frac{1}{\Delta x^2}\begin{pmatrix} a_{\frac{1}{2}} + a_{\frac{3}{2}} & -a_{\frac{3}{2}} & & \\ -a_{\frac{3}{2}} & a_{\frac{3}{2}} + a_{\frac{5}{2}} & -a_{\frac{5}{2}} & \\ & \ddots & \ddots & \ddots \end{pmatrix}.$$

An exact solution is $u(x) = \sin(\pi x)$ for which $a(x) = 1+\varepsilon\sin(\pi x)$ and $f(x) = \pi^2\sin(\pi x)a(x) - \varepsilon\pi^2\cos^2(\pi x)$. Here ε should be a number between 0 and 1. Set $\varepsilon = 0.6$. Implement this scheme first solve the linear system exactly in MATLAB to test the accuracy using this solution. Then implement the full multigrid V-cycle on this problem. Simply use RA_hI as the operator on the coarser grid. Show the relative error for each iteration of FMG up to 3 iterations for $n = 2^k - 1$ with $k = 10, 11, 12, 13$.