CPSC 517, Fall 2013
Assignment 1, due Thursday October 10th

1. For this question, please read Saad's iterative methods book (link provided on course webpage), Sections 3.4 & 3.5, pages 92 through 96. You may also find it helpful to read Chapter 2.
   Consider the convection-diffusion equation
   $$-\Delta u + (\sigma, \tau)\nabla u = f,$$
   discretized on a uniform mesh on the unit square (in two dimensions), $(0,1) \times (0,1)$, where the number of gridpoints is n, i.e., the mesh-size is $h = 1/(n+1)$. Notice that $\sigma = \tau = 0$ gives us the Poisson equation with a Laplace operator.
   (a) Write a function that gets as parameters two *mesh Reynolds numbers*
   $$\beta = \frac{\sigma h}{2}, \quad \gamma = \frac{\tau h}{2},$$
   and generates the discrete convection-diffusion operator with constant coefficients in *compressed sparse row* (CSR) form. Assume Dirichlet boundary conditions.
   Next, write a matrix-vector product function that gets a square matrix A in CSR form and a vector x as input and generates a vector $y = Ax$ as output. If you use MATLAB, you may debug and test the correctness of your code by comparing your results to the Laplacian/Convection-Diffusion m-file provided on the course webpage, named `lcd.m`. Write a script that generates the output of matrix-vector products using a few random input vectors, a few choices of mesh Reynolds numbers, and a few choices of mesh sizes. For each case, confirm that the output of your code is identical to that of the m-file provided by printing out the norm of the difference between the two output vectors. If you do not use MATLAB, make sure to provide enough evidence that verifies that your code is correct.
   (b) Use your code to apply the simple Richardson extrapolation scheme,
   $$x_{k+1} = x_k + \alpha r_k,$$
   for the Poisson equation ($\beta = \gamma = 0$) with
   $$\alpha = \frac{2}{\lambda_{\min}(A) + \lambda_{\max}(A)}.$$

(See question 2 for the eigenvalues of A in this case.)
Test your code on a few problems of a moderate size. Assume homogeneous Dirichlet boundary conditions, and just generate random right-hand side vectors. Comment on the convergence of your scheme. Specifically, determine the asymptotic rate of convergence of this scheme by determining what the spectral radius of the iteration matrix is, and comment on whether the iteration count matches your expectations. As a convergence criterion, go until

$$\frac{\|r_k\|}{\|b\|} \leq 10^{-6},$$

where $r_k = b - Ax_k$. Take $x_0 = 0$.

2. The eigenvalues of the discrete Laplacian are given by

$$\lambda_{i,j} = 4 - 2\left(\cos\frac{i\pi}{n+1} + \cos\frac{j\pi}{n+1}\right), \quad 1 \leq i, j \leq n\,.$$

Note that the matrix is n^2-by-n^2. Suppose you are given an arbitrary right-hand side. For the following schemes, find *exactly* the spectral radius of the iteration matrix, and determine in big O notation how many iterations it will take to reduce the initial error by a factor 10^m.

(a) Jacobi
(b) (hard) block Jacobi, where the matrix M associated with the splitting $A = M - N$ is the tridiagonal part of A.
(c) SOR with the optimal relaxation parameter, given by

$$\omega_{\text{opt}} = \frac{2}{1 + \sqrt{1 - \rho_J^2}},$$

where ρ_J is the spectral radius of the Jacobi iteration matrix. The spectral radius of the SOR iteration matrix is given by $\rho_{\omega_{\text{opt}}} = \omega_{\text{opt}} - 1$; there is no need to prove this.

3. Consider again the convection-diffusion equation with constant coefficients, this time with different boundary conditions on the same domain (unit square), as follows: $u = 0$ on $x = 0$ and $x = 1$, and $\frac{\partial u}{\partial y} = 0$ on $y = 0$ and $y = 1$.

(a) Determine the linear system (matrix as well as right-hand side) in this case.

(b) What would be the right-hand side vector if the solution of the equation happens to be

$$u(x, y) = \sin(\pi x)\cos(\pi y),$$

with the above stated boundary conditions?

(c) Modify the script `lcd.m` or write your own script to generate a linear system with the boundary conditions given in (a) and the specific right-hand side given in (b).

Run your code for $n = 2^k$ with a few choices of k, increasing k as much as you can, at least up to $k = 10$. (But do try to go higher if you can, for full satisfaction!)

Start by using backslash and just making sure that your code is correct: Since the exact solution in this case is known, compute the norm of the difference between the exact solution and the computed solution. The infinity norm may be particularly handy here: `norm(e,Inf)`.

Next, try a few iterative methods. For each run, start with a zero initial guess and plot the convergence history for the relative residual $\frac{\|r_k\|}{\|b\|}$. Use the following schemes: (i) Jacobi, (ii) Block Jacobi as in question 2, (iii) Gauss-Seidel, (iv) SOR with nine values, $\omega = 1.1, 1.2, 1.3, \ldots, 1.9$, (v) GMRES, using the MATLAB command `gmres`, and (vi) preconditioned GMRES, with ILU(0), using the MATLAB command `ilu`. Plot the relative residuals for all schemes (for SOR just pick the fastest one), for a particular mesh size on the same graph, using a logarithmic scale; the `hold` and `semilogy` commands may come handy here. It is OK to hand in the results only for one value of n (the maximal that you have gone for).

Use a stopping criterion based on maximum iteration count (say, 2000) and the magnitude of the relative residual (say, 10^{-6}).

4. Consider the saddle-point linear system

$$\underbrace{\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}}_{\mathcal{K}} \underbrace{\begin{pmatrix} u \\ p \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} f \\ g \end{pmatrix}}_{b},$$

where A is n-by-n, symmetric positive definite, and B is m-by-n with $m < n$.

(a) Show that if B has full row rank, the matrix $\mathcal{K}$ is nonsingular.
(b) Show that $\mathcal{K}$ is symmetric indefinite in two different ways: by showing that $x^T\mathcal{K}x$ does not have the same sign for any nonzero x (we have basically shown that in class), and by showing that the matrix has both positive and negative eigenvalues.
(c) A possible algorithm for solving the linear system $\mathcal{K}x = b$ is based on constructing a sequence of approximations for u and p as follows:

For $k = 0, 1, \ldots$
Solve $Au_{k+1} = f - B^T p_k$
Compute $p_{k+1} = p_k + \alpha(Bx_{k+1} - g)$

i. Show that when the scheme converge, it will indeed converge to the solution of the original linear system, $\mathcal{K}x = b$.
ii. Write down the corresponding iteration matrix and find the value of α that gives optimal convergence. (Provide the $(n + m)$-by-$(n + m)$ iteration matrix, not an iteration matrix of a reduced size.)
iii. Estimate the asymptotic rate of convergence of this scheme.
iv. State two or more numerical difficulties that using the optimal α entails.

5. Consider the iteration

$$x_{k+1} = x_k + \alpha_k d_k,$$

where $d_k = A^T f_k \neq 0$ is the search direction and α_k is a scalar. Denote the error vectors by $e_k = x - x_k$, where x is the exact solution.
(a) Determine α_k so that $\|e_{k+1}\|_2$ is minimized. NOTE: Since e_k or x are not known, α_k should not contain those quantities.
(b) Show that for this choice of α_k the error vector e_{k+1} is orthogonal to d_k.
(c) Show that $\|e_{k+1}\|_2 \leq \|e_k\|_2$.
(d) Suppose $f_k = r_k$ for all k. Determine whether the algorithm converges for an arbitrary initial guess.