CPSC 302, Term 1, Winter 2017–2018 Numerical Computation for Algebraic Problems

Total	/48

Assignment 5 – Iterative Methods for Linear Systems
Due Nov 29, 2017, beginning of class

Reminder:

1. When submitting your hardcopy assignment, you must also include all your code (m-files) and plots. **In addition** to the hardcopy, all code and plots should be submitted electronically using the `handin` command. **Codes/plots/outputs that have not been submitted in hard copy will NOT be graded.**

2. For both your hardcopy and electronic submission, make sure to **include your name and UBC student number**.

3. If you cannot hand in your hardcopy assignment in class, you may hand it in to the Assignment Box marked "CPSC 302" in ICCS X235 **before** lecture on the due date.

4. **Create legible plots/outputs, satisfy coding standards, and document your code!**

---

1. **Stationary Method.** Let $\alpha \in \mathbb{R}$ be a scalar, and consider the iterative scheme

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha(\mathbf{b} - A\mathbf{x}_k)\,.$$

This is the gradient descent method with a fixed step size α.

(a) (2 points) If $A = M - N$ is the splitting associated with this method, state what M and the iteration matrix T are.

(b) Suppose A is symmetric positive definite and its eigenvalues are $\lambda_1 > \lambda_2 > \ldots > \lambda_n > 0$.

i. (4 points) Derive a condition on α that guarantees convergence of the scheme to the solution $\mathbf{x}$ for any initial guess.

ii. (5 points) Show that the best value for the step size in terms of maximizing the speed of convergence is

$$\alpha = \frac{2}{\lambda_1 + \lambda_n}\,.$$

Find the spectral radius of the iteration matrix in this case, and express it in terms of the condition number of A.

(c) (2 points) Determine whether the following statement is true or false. Justify your answer.

"If A is strictly diagonally dominant and $\alpha = 1$, then the iterative scheme converges to the solution for any initial guess $\mathbf{x}_0$."

2. Consider the two-dimensional partial differential equation

$$-\Delta u + \alpha^2 u = f\,,$$

where $\Delta u = \left(\frac{\delta^2 u}{\delta x^2} + \frac{\delta^2 u}{\delta y^2}\right)$ and α is a given real scalar, on the unit square $\Omega = (0,1) \times (0,1)$, subject to homogeneous Dirichlet boundary conditions: $u = 0$ on $\delta\Omega$. The matrix that corresponds to a straightforward extension of (7.1) in the textbook for this differential problem is

$$B = A + (\alpha h)^2 I\,,$$

where A is the block tridiagonal matrix from Example 7.1 in the textbook, I is the identity matrix of size n, and h is the grid width, given by $h = \frac{1}{N+1}$ where $N = \sqrt{n}$. Recall that the eigenvalues of A are

$$\lambda_{l,m} = 4 - 2\left(\cos(l\pi h) + \cos(m\pi h)\right), \quad 1 \le l, m \le N\,.$$

(a) (5 points) Consider $\alpha = \sqrt{2}\pi$. Derive a formula for computing the condition number $\kappa_2(B)$, without actually forming the matrix B. How does $\kappa_2(B)$ depend on n?

(b) (16 points) Write a program that solves the problem for different values of α (specified below), with a right-hand side vector defined so that the solution is a vector of all 1's. In MATLAB the instruction

```
B = delsq(numgrid('S',N+2)) + c*speye(n);
b = B*ones(N^2,1);
```

where $c = (\alpha h)^2$, generates a *sparse* matrix representation of our B and the right-hand side. Your program will find the numerical solution using the Jacobi, Gauss–Seidel, SOR, and CG methods. For CG use the MATLAB command `pcg` and run it once without preconditioning and once preconditioned with incomplete Cholesky IC(0). For SOR, use the formula from Example 7.8 in the textbook for finding the optimal ω_{opt} and apply the scheme only for this value. As a stopping criterion use $\|\mathbf{r}_k\|_2/\|\mathbf{r}_0\|_2 < 10^{-6}$. Also, impose an upper bound of 2000 iterations. That is, if a scheme fails to satisfy the accuracy requirement on the relative norm of the residual after 2000 iterations, it should be stopped. For each of the methods, start with a zero initial guess. Your program should print out the following:

- The sizes N and n, $\kappa_2(B)$, and iteration counts for the five cases (Jacobi, Gauss–Seidel, SOR, CG, PCG).
- Plots of the relative residual norms $\frac{\|\mathbf{r}_k\|_2}{\|\mathbf{b}\|_2}$ vs. iterations. Use the MATLAB command `semilogy` for plotting.

Use two grids with $N = 31$ and $N = 63$, and repeat your experiments for three values of α: $\alpha = 0$, $\alpha^2 = 10$, and $\alpha^2 = 1000$. Explain the observed differences in speed of convergence.

3. Suppose we wish to solve $A\mathbf{x} = \mathbf{b}$. Writing A in the form:

$$A = D - L - U\,,$$

where D is diagonal, L is strictly lower triangular, and U is strictly upper triangular, the Jacobi method can be written as:

$$\mathbf{x}_{k+1} = D^{-1}\left[(L+U)\mathbf{x}_k + \mathbf{b}\right]\,.$$

The file `gsMorph.m` solves a linear system with this formulation of the Jacobi method. The matrix A is constructed as the 5-point finite difference Laplacian on a square using the `delsq` command in MATLAB. In this problem, the vector $\mathbf{b}$ is the product $A\mathbf{x}$ where $\mathbf{x}$ consists of the grayscale values of an image of David M. Young. As an initial guess, we let $\mathbf{x}_0$ be the vector of grayscale values of an image of Carl Gustav Jacobi. We then visualize $\mathbf{x}_k$ as a vector of grayscale values after each iteration of the Jacobi method to create a morph from the image of Jacobi to the image of Young.

(a) (2 points) In the code, you will find the value of the variable `diagonalIncrement` to be 0. Run the code with `diagonalIncrement = 0.5`. This increments every diagonal entry of A by 0.5. Then run the code again with this variable set to 1 and finally to 2. Describe the effect on the morph from these different values. How does incrementing this value affect the rate of convergence of the iterative method? What effect must incrementing the diagonal entries of A have on the spectral radius of the resulting iteration matrix?

(b) (3 points) Set `diagonalIncrement` to be 0.5. Modify the code to implement the Gauss–Seidel method instead of the Jacobi method. Also, change the morph to begin with an image of Gauss rather than Jacobi.

4. The *smoothing factor* μ^* for a discrete operator is defined as the worst (i.e., smallest) factor by which high frequency components are reduced in a single relaxation step. For the two-dimensional Laplacian we have discussed throughout this chapter and a basic relaxation scheme, this can be stated as follows. Suppose $\mathbf{e}_0$ is the error before a relaxation step associated with a stationary iteration matrix T and $\mathbf{e}_1$ the error after that step, and write

$$\mathbf{e}_0 = \sum_{l,m=1}^{N} \alpha_{l,m}\mathbf{v}_{l,m} ,$$

where $\{\mathbf{v}_{l,m}\}_{l,m=1}^{N}$ are the eigenvectors of the iteration matrix. Then

$$\mathbf{e}_1 = \sum_{l,m=1}^{N} \alpha_{l,m}\mu_{l,m}\mathbf{v}_{l,m} ,$$

where $\{\mu_{l,m}\}_{l,m=1}^{N}$ are eigenvalues of the iteration matrix. The smoothing factor is thus given by

$$\mu^* = \max\left\{ |\mu_{l,m}| : \frac{N+1}{2} \le l \le N,\, 1 \le m \le N \right\} .$$

(a) (2 points) Denote the discrete Laplacian by A and the iteration matrix for damped Jacobi by T_ω. Confirm that the eigenvectors of A are the same as the eigenvectors of T_ω for this scheme.

(b) (5 points) Show that the optimal ω that gives the smallest smoothing factor over $0 \le \omega \le 1$ for the two-dimensional Laplacian is $\omega^* = 4/5$, and find the smoothing factor $\mu^* = \mu^*(\omega^*)$ in this case. Note: μ^* should not depend on the mesh size.

(c) (2 points) Show that Jacobi (i.e., the case $\omega = 1$) is not an effective smoother.