Math/CS 714: Assignment 2

1. **Iterative methods (2 points).**

   (a) Consider the 2×2 matrix

   $$A = \begin{pmatrix} 1 & \rho \\ -\rho & 1 \end{pmatrix}. \tag{1}$$

   Under what conditions will the Jacobi and Gauss–Seidel methods converge?

   (b) Consider the $n \times n$ matrix

   $$C = \begin{pmatrix} 3 & -1 & & & & & \\ -1 & 3 & -1 & & & & \\ & -1 & 3 & -1 & & & \\ & & & \ddots & & & \\ & & & -1 & 3 & -1 \\ & & & & -1 & 3 \end{pmatrix}. \tag{2}$$

   Starting from $u_0 \in \mathbb{R}^n$, for which values of $\omega \in \mathbb{R}$ does the iteration

   $$u_{k+1} = u_k + \omega(b - Cu_k) \tag{3}$$

   for $k = 0, 1, 2, \ldots$ converge to a solution of $Cu = b$? What iterative method from the lectures does this most closely resemble? How is it different?

2. **Triangular domain revisited with conjugate gradient (8 points).** Question 4 of the first homework assignment looked at solving the Poisson equation

   $$\nabla^2 u = f \tag{4}$$

   on the triangular domain T with vertices at $(0, 0)$, $(1, 0)$, and $(s, \frac{1}{2})$ where $s = \frac{\sqrt{3}}{2}$. Dirichlet boundary conditions $u = 0$ are applied on the boundary ∂T. For $n \in \mathbb{N}$ and $h = 1/n$, the domain is discretized with points

   $$\mathbf{x}_{i,j} = (h(i + \tfrac{1}{2}j), hsj)) \tag{5}$$

   for $0 \le i \le n$, $0 \le j \le n - i$. The points where $i = 0$, $j = 0$, or $i + j = n$ are boundary points, and all others are interior. Let $u_{i,j}$ be the numerical approximation for $u(\mathbf{x}_{i,j})$. The Laplacian is discretized using

   $$\nabla_6^2 u_{i,j} = \frac{2(-6u_{i,j} + u_{i+1,j} + u_{i,j-1} + u_{i-1,j+1} + u_{i-1,j} + u_{i,j+1} + u_{i+1,j-1})}{3h^2}. \tag{6}$$

   As in Homework 1, consider solving Eq. (4) using the f that creates the exact solution

   $$u^{\text{ex}}(x, y) = \left((2y - \sqrt{3})^2 - 3(2x - 1)^2\right) \sin y. \tag{7}$$

The problem can be expressed as a linear system $Au = f$ where $u \in \mathbb{R}^N$ is the solution vector at the $N = (n-1)(n-2)/2$ interior grid points. Define an appropriate 2-norm as

$$\|r\|_2 = \sqrt{\frac{s}{2n^2} \sum_{j=1}^{n-1} \sum_{i=1}^{n-j-1} r_{i,j}^2} \tag{8}$$

for a vector r describing a field on the grid.

(a) For a range of grid sizes from $n = 10$ up to at least $n = 160$ measure the wall-clock time T_n to solve the system using your code.[1] By making a log–log plot of T_n versus n, fit the timing data to

$$T_n = an^b, \tag{9}$$

and comment on the exponent b.

(b) Write a code to implement the conjugate gradient (CG) algorithm to solve $Au = f$. Your code should not explicitly build A as a dense matrix. It should either represent A as a sparse matrix, or contain a function that can directly compute Aq for a given vector q. The CG algorithm should terminate when $\|r\|_2 < 10^{-10}$, where r is the residual vector. For the case when $n = 40$ test that your program gives the same results as the original version.

(c) For $n = 10, 20, 40, 80, 160$, calculate the number of iterations k required in order to reach the termination criterion.

(d) Measure the wall-clock time of the CG algorithm to solve the linear system for a range of grid sizes from $n = 10$ up to at least $n = 160$. Fit the data to Eq. (9) and comment on how the exponent compares to your result from part (a).

(e) Consider the block Jacobi preconditioner M with block sizes of $w = \lfloor \sqrt{N} \rfloor$. In general, w will not evenly divide N, and therefore let the final block of M be smaller. Implement the preconditioned CG algorithm, and repeat part (c) to measure the number of iterations required to reach the termination criterion.

(f) Repeat part (d), fitting the timing data to Eq. (9)

(g) **Optional.** Would this question be possible if ∇_3^2 was used instead of ∇_6^2?

(h) **Optional.** With the block Jacobi preconditioner, the ordering of the points within the linear system will affect the performance. In particular, having a large number of matrix entries within the Jacobi blocks will likely make the preconditioner more effective. Consider approaches to reorder the points to achieve better performance.

[1] If you had difficulty with completing this question, you can alternatively use the code `poisson_tri.py` from the homework 1 solutions.

3. **ODE integration methods (7 points).**

   (a) Consider solving the differential equation $y' = f(t, y)$ at timepoints t_k with corresponding numerical solutions y_k. The multi-step Nyström numerical method is based upon the integral relation

   $$y(t_{k+1}) = y(t_{k-1}) + \int_{t_{k-1}}^{t_{k+1}} f(t, y)dt. \tag{10}$$

   Derive an implicit multi-step numerical method by approximating the integrand $f(t, y)$ with the polynomial interpolant using the function values at t_{k-2}, t_{k-1}, t_k, and t_{k+1}. Your method should have the form

   $$y_{k+1} = y_{k-1} + h(\alpha f_{k-2} + \beta f_{k-1} + \gamma f_k + \eta f_{k+1}) \tag{11}$$

   where α, β, γ, and η are constants to be determined, h is the timestep interval size, and $f_l = f(t_l, y_l)$ for an arbitrary l.

   (b) Find the exact solution to the second-order differential equation

   $$y''(t) + 2y'(t) + 26y(t) = 0. \tag{12}$$

   subject to the initial conditions $y(0) = 1$, $y'(0) = 0$.

   (c) Write Eq. (12) as a coupled system of two first-order differential equations for $\mathbf{y} = (y, v) = (y, y')$. Solve the system over the interval $0 \leq t \leq 3$ with a timestep of $h = 0.02$ using your multi-step method from part (a).[2]

   Before Eq. (11) can be applied, $\mathbf{y}_1$ and $\mathbf{y}_2$ must be calculated accurately. Use one of the following two approaches:

   i. set them based on the exact solution from part (b),
   ii. calculate them using the classical fourth-order Runge–Kutta method.

   Plot the exact and numerical solutions over the range $0 \leq t \leq 3$.

   Make a log–log plot of the absolute error between the numerical and exact values of y at $t = 3$ as a function of h, over the range from $h = 10^{-3}$ to $h = 10^{-1}$. Show that your method is fourth-order accurate.

   (d) Suppose that instead of setting $\mathbf{y}_1$ and $\mathbf{y}_2$ accurately, you instead make use of forward Euler steps. Create a log–log plot of the absolute error of y at $t = 3$ as a function of h. Determine the order of accuracy, and discuss why this is the case.

4. **Linear difference equation (3 points).**

   (a) Find the general solution of the linear difference equation

   $$U_{n+3} + 2U_{n+2} - 4U_{n+1} - 8U_n = 0. \tag{13}$$

[2] Even though Eq. 11 is an implicit method and would be hard to solve in general, the linearity of the system in Eq. 12 means that for this case, the update equation for $\mathbf{y}_{k+1}$ can be derived analytically.

(b) Determine the particular solution with initial data $U_0 = 4$, $U_1 = -2$, and $U_2 = 8$.

(c) Consider the iteration

$$\begin{pmatrix} U_{n+1} \\ U_{n+2} \\ U_{n+3} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 8 & 4 & -2 \end{pmatrix} \begin{pmatrix} U_n \\ U_{n+1} \\ U_{n+2} \end{pmatrix}, \tag{14}$$

which we can define as $\mathbf{U}_{n+1} = A\mathbf{U}_n$. The matrix A is called the companion matrix for the difference equation in Eq. (13). A general solution of the difference equation is given by $\mathbf{U}_n = A^n\mathbf{U}_0$. If $A = RJR^{-1}$ is the Jordan canonical form for A, then $A^n = RJ^nR^{-1}$. Determine the eigenvalues and Jordan canonical form for this matrix and show how this is related to the general solution found in part (a).