MAT 226B Large Scale Matrix Computation
Homework 2

Ahmed Mahmoud

February, 13th 2020

Problem 1:

(a) For matrix `HW2_P1_a.mat`, the values of specified entries in I and J vectors are:

(a) $J(100000) = 23414$
(b) $J(150000) = 16652$
(c) $J(200000) = 30830$
(d) $J(250000) = 37823$
(e) $J(300000) = 17067$
(f) $I(10000) = 67441$
(g) $I(20000) = 134934$
(h) $I(30000) = 202446$
(i) $I(40000) = 269986$
(j) $I(50000) = 337481$

(b) For matrix `HW2_P1_b.mat`, The vector J and I are

$$J^T = \begin{bmatrix} 1 & 3 & 4 & 5 & 8 & 12 & 13 & 15 & 16 & 18 & 21 & 24 & 26 & 27 \end{bmatrix}$$

$$I^T = \begin{bmatrix} 8 & 13 & 15 & 12 & 6 & 10 & 11 & 3 & 6 & 12 & 15 & 8 & 2 & 5 & 14 & \ldots\ldots \end{bmatrix}$$
$$\begin{bmatrix} \ldots\ldots 8 & 9 & 2 & 6 & 11 & 4 & 10 & 14 & 8 & 11 & 3 & 2 & 4 & 7 \end{bmatrix}$$

Problem 2:

(a) For the given matrix $U^{(k)}$, there will be **10** fill-in elements as shown in the following matrix if the k-th step of sparse LU factorization is performed without pivoting where the fill-in elements are shown with **+**

$$U^{(k)} = \begin{bmatrix} * & 0 & 0 & 0 & 0 & * & 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & * & * \\ * & 0 & * & * & 0 & * & 0 & 0 & + & 0 & * \\ * & 0 & 0 & 0 & 0 & * & 0 & 0 & + & * & 0 \\ 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & 0 & * \\ * & 0 & 0 & 0 & 0 & + & * & * & + & 0 & 0 \\ 0 & 0 & 0 & * & * & 0 & 0 & 0 & * & 0 & * \\ * & 0 & 0 & 0 & 0 & + & 0 & 0 & + & * & 0 \\ * & 0 & 0 & * & 0 & + & 0 & * & + & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & * & * & 0 & 0 & * \\ * & * & * & 0 & * & + & 0 & 0 & + & 0 & 0 \end{bmatrix}$$

The location of the fill-in elements is: $(6,6)$, $(8,6)$, $(9,6)$, $(11,6)$, $(3,9)$, $(4,9)$, $(6,9)$, $(8,9)$, $(9,9)$, and $(11,9)$.

(b) In order to determine the number of fill-in elements with pivot element via Markowitz criterion, we first need to specify the candidate pivot elements. Since we have parameter $\alpha = 0.1$ and

$$\min_{i,l:u_{i,l}\neq 0} |u_{il}| \geq \frac{1}{7} \max_{i,l} |u_{il}|$$

Then any nonzero entry in $U^{(k)}$ is a candidate pivot. Now, we need to apply Markowitz criterion i.e., swap the rows and columns with least number of nonzero entries. Let c_l and $r_i \in \mathbb{R}^{11}$ be the two vectors that define the number of nonzero entry per each column and row respectively.

$$c_l = \begin{bmatrix} 7 & 2 & 2 & 4 & 2 & 4 & 2 & 3 & 2 & 3 & 5 \end{bmatrix}$$

$$r_i = \begin{bmatrix} 3 & 3 & 5 & 3 & 3 & 3 & 4 & 2 & 3 & 3 & 4 \end{bmatrix}$$

Row 8 has the least number of nonzero entry among all rows. Columns 2, 3, 5, 7, and 9 has the lowest number of nonzero entry among all columns. However row 8 coincides with column 2, 3, 5, 7, and 9 in zero entry so non of them can be choose as a pivot element. Thus, we need to choose a row with the second lowest number of nonzero i.e., 3 nonzero entry per row. Rows 1, 2, 4, 5, 6, 9, and 10 satisfy this condition. Following Markowitz criterion, we chose the row with the lowest index i.e., row 1. It is possible now to choose one of the above

columns that has 2 nonzero entry and meets row 1 in a nonzero entry i.e, column 9. Thus, **the pivot element is (1, 9)** which minimizes $(r_i - 1)(c_l - 1)$ to be 2. Thus, the worst case number of fill-in elements is 2.

To determine the fill-in elements location, we need first to bring the pivot element to the top-left corner by swapping column 1 and 9 and then perform the k-th step of sparse LU factorization. The location of the two fill-in is shown in the following matrix.

$$U^{(k)} = \begin{bmatrix} * & 0 & 0 & 0 & 0 & * & 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & * & * \\ 0 & 0 & * & * & 0 & * & 0 & 0 & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & * & 0 & 0 & * & * & 0 \\ 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & 0 & 0 \\ * & 0 & 0 & * & * & \mathbf{+} & 0 & 0 & \mathbf{+} & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & * & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & * & * & 0 & 0 & * \\ 0 & * & * & 0 & * & 0 & 0 & 0 & * & 0 & 0 \end{bmatrix}$$

Problem 3:

(a) In order to generalize the fast elliptical solver, we need account for the nonzero boundary conditions and add it to the F matrix. Let $V_{jk} \approx V(x_i, y_k) \in \mathbb{R}^{m\times m}$ and $f_{j,k} \approx f(x_j, y_k) \in \mathbb{R}^{m\times m}$ where m is the number of grid point in each dimension and $i, k = 1, \ldots, m$. The centered finite-difference formula is

$$4v_{jk} - v_{j-1,k} - v_{j+1,k} - v_{j,k-1} - v_{j,k+1} = h^2 f_{j,k}, \quad j, k = 1, \ldots, m$$

The compact formulation of the 2D centered finite-difference from which we derived the FFT-based fast elliptical solver is

$$T_m V + V T_m = h^2 F$$

where $h = \frac{1}{m+1}$ and $T_m \in \mathbb{R}^{m\times m}$ is the centered finite-difference in 1D i.e.,

$$T_m = \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix}$$

The nonzero boundary conditions only changes F when $j = 1$ or m and $k = 1$ or m as follows. With zero boundary condition at $j = 1$, $v_{j-1,k} = v_{0,k}$ $(k = 1, \ldots m)$ used to not affect F but now since $v_{0,k} = c0(y)$, we need to add this to the RHS F. We do the same thing for the remaining three boundaries. This will adjust F such that

$$F(1,:) = F(1,:) + \frac{b_0(x)}{h^2}$$

$$F(m,:) = F(m,:) + \frac{b_1(x)}{h^2}$$

$$F(:,1) = F(:,1) + \frac{c_0(y)}{h^2}$$

$$F(:,m) = F(:,m) + \frac{c_1(y)}{h^2}$$

Note that we divide by h^2 since F will be multiplied by h^2 and so it cancels out. We substitute the discrete values x and y at grid point j in $b_0(x), b_1(x), c_0(y),$ and $c_1(x)$ as $x = \frac{j}{m+1}$ and $y = \frac{j}{m+1}$. After adjusting F, the same derivation will follow as given in the class.

(b) Function `FFTSolver()` in `problem_3.m` implements the FFT-based solver.

(c) To determine function b_0, b_1, c_0, and c_1 we simply plug in their definition in $v(x, y)$ definition

$$
\begin{aligned}
b_0 &= v(x, 0) = 0 \\
b_1 &= v(x, 1) = x^{\alpha} cos(\beta\pi x) sin(\gamma\pi) \\
c_0 &= v(0, y) = 0^{\alpha} sin(\gamma\pi y) \\
c_1 &= v(1, y) = cos(\beta\pi) sin(\gamma\pi y)
\end{aligned}
$$

f can be determined from its definition i.e., $f(x, y) = -\frac{\partial^2 v(x,y)}{\partial x^2} - \frac{\partial^2 v(x,y)}{\partial y^2}$ where

$$
\begin{aligned}
\frac{\partial^2 v(x, y)}{\partial x^2} &= sin(\gamma\pi y) \left[\left(\alpha(\alpha - 1) x^{\alpha - 2} + \beta^2 \pi^2 x^{\alpha} \right) cos(\beta\pi x) - 2\alpha\beta\pi x^{\alpha - 1} sin(\beta\pi x) \right] \\
\frac{\partial^2 v(x, y)}{\partial y^2} &= -\alpha^2 \pi^2 x^{\alpha} cos(\beta\pi x) sin(\gamma\pi y)
\end{aligned}
$$

(d) Table 1 shows the chosen m for different test cases and the corresponding absolute error. We choose the grid size such that error is always less than 0.0001. Figures 2 to 6 show the right-hand side function f, the exact solution v, and the computed solution for all the test cases.

Test Case	m	Error
(I) $\alpha = 0, \beta = 1, \gamma = 1$	60	0.000075
(II) $\alpha = 1, \beta = 2, \gamma = 1$	200	0.000069
(III) $\alpha = 2, \beta = 2, \gamma = 3$	150	0.000078
(IV) $\alpha = 5, \beta = 2, \gamma = 3$	80	0.000048
(V) $\alpha = 5, \beta = 3, \gamma = 5$	200	0.000094

Figure 1: Chosen m and absolute error for different test cases

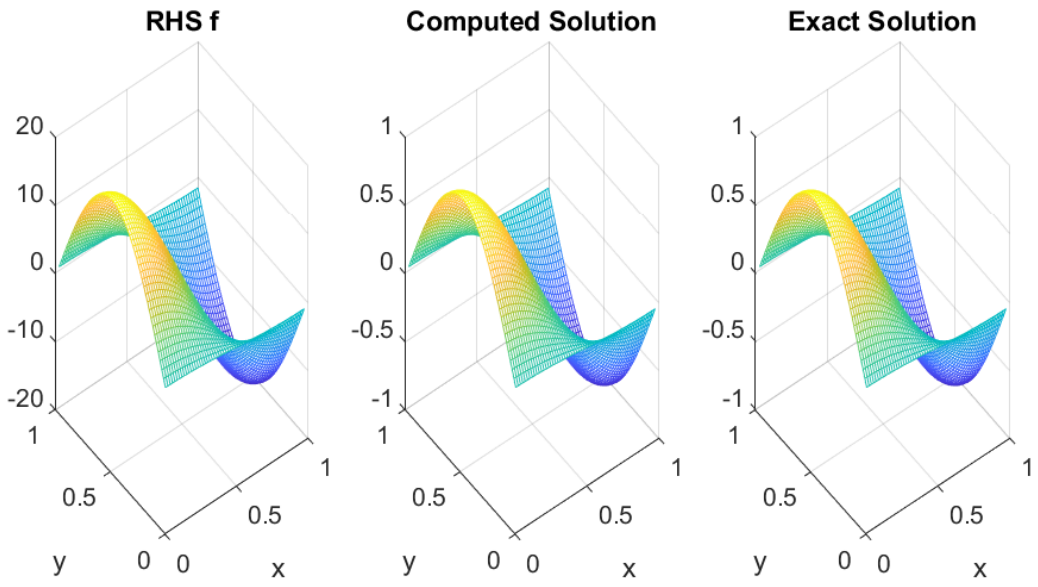

Figure 2: Test Case I

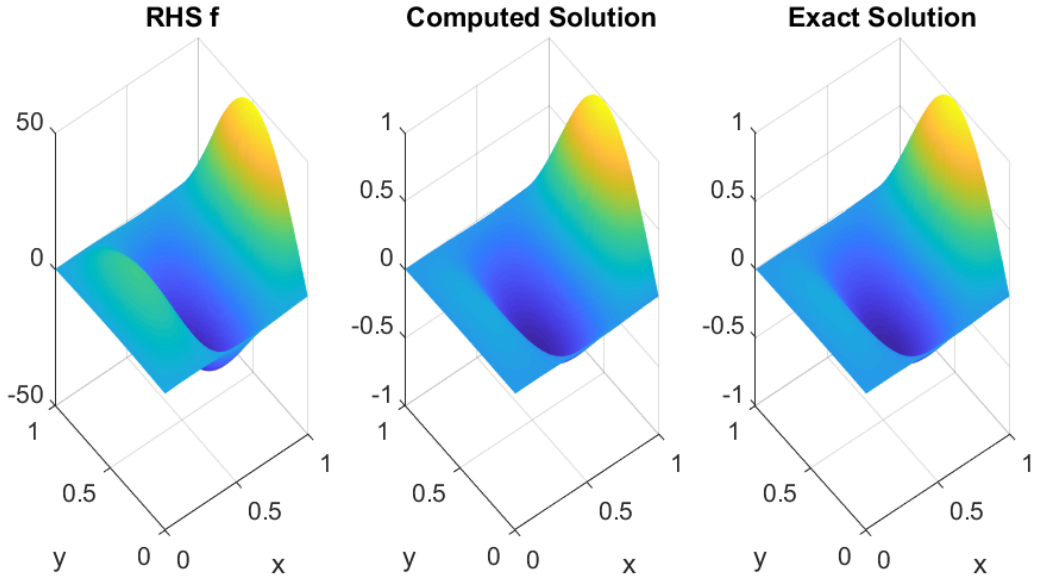

Figure 3: Test Case II

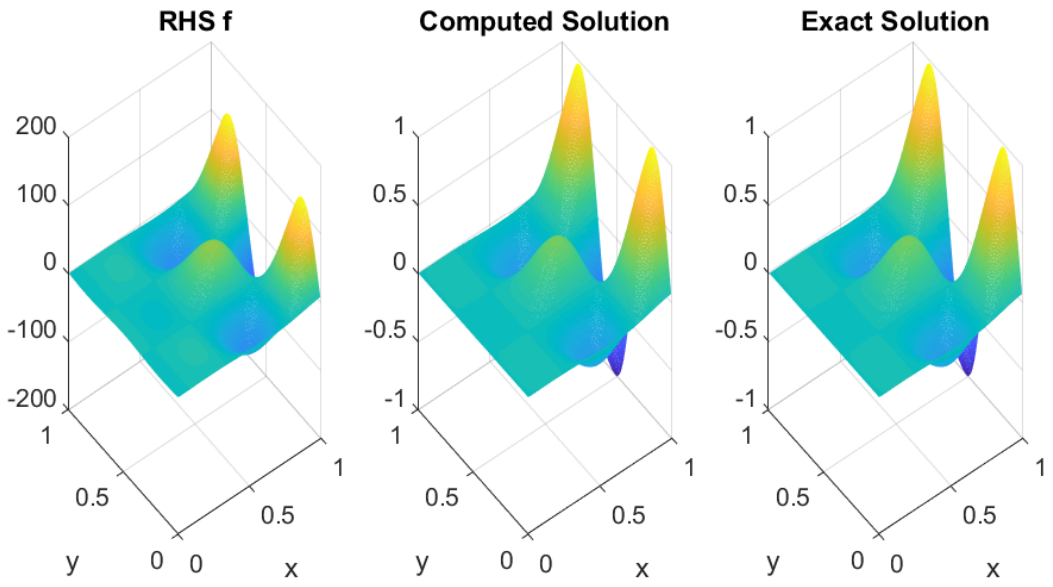

Figure 4: Test Case III

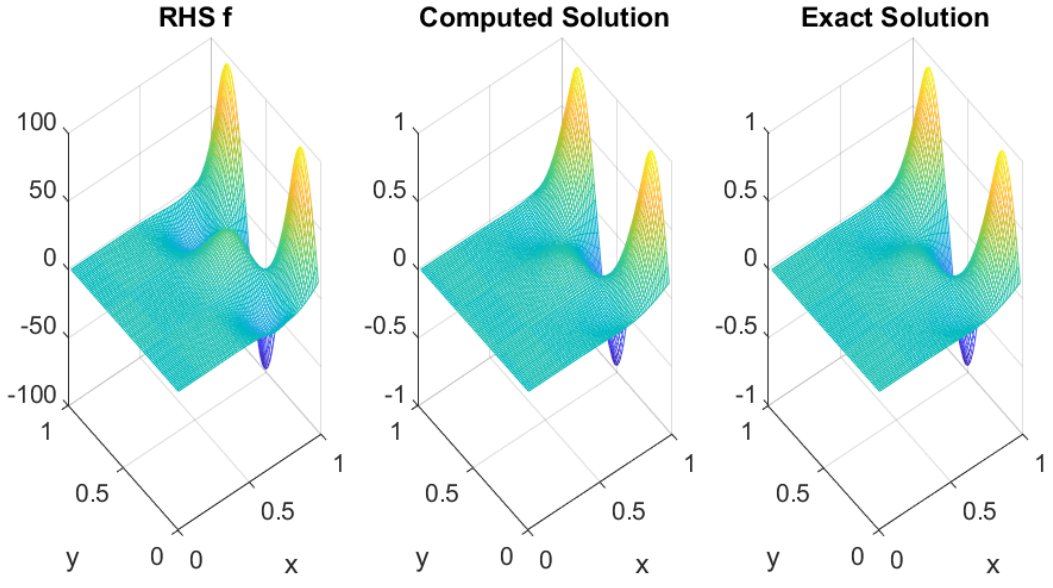

Figure 5: Test Case IV

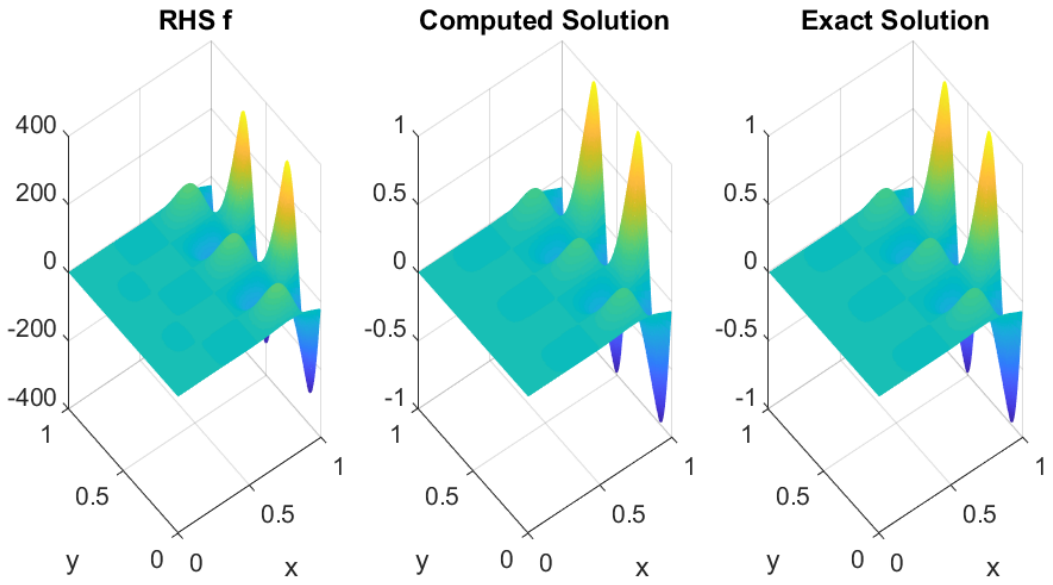

Figure 6: Test Case V

Problem 4:

(a) Let $M \in \mathbb{R}^{n\times n} \succ 0$, $v_1, v_2, \ldots, v_k\mathbb{R}^n$ be k linearly independent vectors, and let $x^*, x_0 \in \mathbb{R}^n$. To minimize the following

$$\| x^* - x_k \|_M = min_{x\in x_0+S} \| x^* - x||_M$$

where $S = \text{span}\{v_1, v_2, \ldots, v_k\} \subset \mathbb{R}^n$, we can write x_k in the form

$$x_k = x_0 + \alpha_1 v_1 + \alpha_2 v_2 + \cdots + \alpha_k v_k, \quad \alpha_1, \alpha_2, \ldots, \alpha_k \in \mathbb{R}$$

Problem 5:

(a) A single iteration of the CG algorithm requires:

- Three SAXPY each takes $2n^2$ flops
- Two inner products each takes $2n^2 - 1$
- Single matrix-vector multiplication which takes for a general (dense) matrix $2n^2 - n$

Thus, the total flops requires for a single iteration is

$$3 * 2n^2 + 2 * (2n^2 - 1) + 2n^2 - n = 12n^2 - n - 2 \text{ flops}$$

(b) The $k-$iterate of the CG algorithm has the following property

$$\frac{\| x^* - x_k \|_A}{\| x^* - x_0 \|_A} \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k$$

where x^* is the solution i.e., $x^* = A^{-1}b$ and κ is the condition number of A. Given that $\| x^* - x_0 \|_A = \frac{1}{2}$ and $\kappa = 9$, then to find the minimum number k of iterations that generates $\| x^* - x_k \|_A \leq 10^{-12}$ we need to solve for k in the following equation

$$\frac{10^{-12}}{0.5} \leq 2 \left(\frac{\sqrt{9} - 1}{\sqrt{9} + 1} \right)^k$$

which gives $k <= 39.8631$. Thus, **the minimum number of iteration is** 40.

(c) Solving $Ax = b$ via Cholesky factorization requires three steps; first the decomposition of $A = LL^T$ which requires (from Homework 1)

$$n + \frac{1}{2}n(n-1) + \frac{1}{3}n(n^2 - 1) \text{ flops}$$

Second, the forward substitution which requires

$$\sum_{k=0}^{n-1} (1 + 2k) = n^2 \text{ flops}$$

Third, the backward substitution which requires also n^2 flops. Thus, the total number of flops to solver $Ax = b$ is

$$n + \frac{1}{2}n(n-1) + \frac{1}{3}n(n^2 - 1) + 2n^2 = \frac{1}{3}n^3 + \frac{5}{2}n^2 + \frac{1}{6}n \text{ flops}$$

While solving the same system using CG up to $k = 40$ iterations requires (ignoring the initial operation to calculate r_0)

$$\underbrace{2n^2}_{r_0} + \underbrace{(480n^2 - 40n - 80)}_{k \text{ iterations}} \text{ flops}$$

(d) Using only the term of the highest power of n, the matrix size n for which k CG iteration (where $k = 40$) iterations requires only one tenth of the arithmetic operations that Cholesky approach requires can be obtain from

$$480n^2 = 0.1 * \frac{1}{3}n^3$$

which gives $n = 14400$