Math 342: Homework 6

Connor Emmons

Documentation: I used ChatGPT to look up Latex commands and to double check some of the algebra performed in this homework. The main Homework 6 MatLab script and all required dependencies are located in the Homework 6 folder found here: https://github.com/Connor-Lemons/Emmons-Math-342. No other resources used.

Problem 1: Use Gaussian elimination with backwards substitution and two-digit rounding arithmetic to solve the following system:

$$\begin{gathered} -x_1 + 4x_2 + x_3 = 8 \\ \tfrac{5}{3}x_1 + \tfrac{2}{3}x_2 + \tfrac{2}{3}x_3 = 1 \\ 2x_1 + x_2 + 4x_3 = 11 \end{gathered}$$

First, implement the system using two-digit rounding (note that inverse operations do not contribute to the operation count):

$$\begin{gathered} -x_1 + 4x_2 + x_3 = 8 \\ 1.7x_1 + 0.67x_2 + 0.67x_3 = 1 \\ 2x_1 + x_2 + 4x_3 = 11 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: } 0 \\ \text{Multiplication/Division: } 0 \end{gathered}$$

Perform the elimination:

$$\begin{gathered} -x_1 + 4x_2 + x_3 = 8 \\ 0x_1 + 7.5x_2 + 2.4x_3 = 15 \\ 0x_1 + 9x_2 + 6x_3 = 27 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: } 6 \\ \text{Multiplication/Division: } 8 \end{gathered}$$

$$\begin{gathered} -x_1 + 4x_2 + x_3 = 8 \\ 0x_1 + 7.5x_2 + 2.4x_3 = 15 \\ 0x_1 + 0x_2 + 3.1x_3 = 9 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: } 8 \\ \text{Multiplication/Division: } 11 \end{gathered}$$

Perform the back substitution:

$$\begin{gathered} -x_1 + 4x_2 + x_3 = 8 \\ 0x_1 + 7.5x_2 + 2.4x_3 = 15 \\ 0x_1 + 0x_2 + x_3 = 2.9 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: } 8 \\ \text{Multiplication/Division: } 12 \end{gathered}$$

$$\begin{gathered} -x_1 + 4x_2 + x_3 = 8 \\ 0x_1 + x_2 + 0x_3 = 1.1 \\ 0x_1 + 0x_2 + x_3 = 2.9 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: } 9 \\ \text{Multiplication/Division: } 14 \end{gathered}$$

$$\begin{gathered} x_1 + 0x_2 + 0x_3 = -0.7 \\ 0x_1 + x_2 + 0x_3 = 1.1 \\ 0x_1 + 0x_2 + x_3 = 2.9 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: } 11 \\ \text{Multiplication/Division: } 17 \end{gathered}$$

Problem 2: Solve the four linear systems:

$$\begin{gathered} 2x_1 - 3x_2 + x_3 = 2 \\ x_1 + x_2 - x_3 = -1 \\ -x_1 + x_2 - 3x_3 = 0 \end{gathered} ; \quad \begin{gathered} 2x_1 - 3x_2 + x_3 = 6 \\ x_1 + x_2 - x_3 = 4 \\ -x_1 + x_2 - 3x_3 = 5 \end{gathered} ; \quad \begin{gathered} 2x_1 - 3x_2 + x_3 = 0 \\ x_1 + x_2 - x_3 = 1 \\ -x_1 + x_2 - 3x_3 = -3 \end{gathered} ; \quad \begin{gathered} 2x_1 - 3x_2 + x_3 = -1 \\ x_1 + x_2 - x_3 = 0 \\ -x_1 + x_2 - 3x_3 = 0 \end{gathered}$$

First use Gaussian elimination on the augmented matrix:

$$\begin{bmatrix} 2 & -3 & 1 & 2 & 6 & 0 & -1 \\ 1 & 1 & -1 & -1 & 4 & 1 & 0 \\ -1 & 1 & -3 & 0 & 5 & -3 & 0 \end{bmatrix} \qquad \begin{gathered} \text{Addition/Subtraction: } 0 \\ \text{Multiplication/Division: } 0 \end{gathered}$$

$$\begin{bmatrix} 2 & -3 & 1 & 2 & 6 & 0 & -1 \\ 0 & \frac{5}{2} & -\frac{3}{2} & -2 & 1 & 1 & \frac{1}{2} \\ 0 & -\frac{1}{2} & -\frac{5}{2} & 1 & 8 & -3 & -\frac{1}{2} \end{bmatrix} \qquad \begin{gathered} \text{Addition/Subtraction: } 12 \\ \text{Multiplication/Division: } 14 \end{gathered}$$

$$\begin{bmatrix} 2 & -3 & 1 & 2 & 6 & 0 & -1 \\ 0 & \frac{5}{2} & -\frac{3}{2} & -2 & 1 & 1 & \frac{1}{2} \\ 0 & 0 & -\frac{28}{10} & \frac{3}{5} & \frac{41}{5} & -\frac{14}{5} & -\frac{4}{10} \end{bmatrix} \quad \begin{array}{c}\text{Addition/Subtraction: 17}\\ \text{Multiplication/Division: 20}\end{array}$$

Perform the back substitution:

$$\begin{bmatrix} 2 & -3 & 1 & 2 & 6 & 0 & -1 \\ 0 & \frac{5}{2} & -\frac{3}{2} & -2 & 1 & 1 & \frac{1}{2} \\ 0 & 0 & 1 & -\frac{3}{14} & -\frac{41}{14} & 1 & \frac{1}{7} \end{bmatrix} \quad \begin{array}{c}\text{Addition/Subtraction: 17}\\ \text{Multiplication/Division: 24}\end{array}$$

$$\begin{bmatrix} 2 & -3 & 1 & 2 & 6 & 0 & -1 \\ 0 & 1 & 0 & -\frac{13}{14} & -\frac{19}{14} & 1 & \frac{2}{7} \\ 0 & 0 & 1 & -\frac{3}{14} & -\frac{41}{14} & 1 & \frac{1}{7} \end{bmatrix} \quad \begin{array}{c}\text{Addition/Subtraction: 21}\\ \text{Multiplication/Division: 33}\end{array}$$

$$\begin{bmatrix} 1 & 0 & 0 & -\frac{2}{7} & \frac{17}{7} & 1 & -\frac{1}{7} \\ 0 & 1 & 0 & -\frac{13}{14} & -\frac{19}{14} & 1 & \frac{2}{7} \\ 0 & 0 & 1 & -\frac{3}{14} & -\frac{41}{14} & 1 & \frac{1}{7} \end{bmatrix} \quad \begin{array}{c}\text{Addition/Subtraction: 29}\\ \text{Multiplication/Division: 45}\end{array}$$

The solutions to each x-value for each system are contained in the last 4 columns of the matrix. Each column represents a system, and each element within those columns represents the values of the x-values.

Now find the solve the systems by solving first for the inverse:

$$\begin{bmatrix} 2 & -3 & 1 \\ 1 & 1 & -1 \\ -1 & 1 & -3 \end{bmatrix} \quad \begin{array}{c}\text{Addition/Subtraction: 0}\\ \text{Multiplication/Division: 0}\end{array}$$

First find the determinant:

$$\begin{vmatrix} 2 & -3 & 1 \\ 1 & 1 & -1 \\ -1 & 1 & -3 \end{vmatrix} = -14 \quad \begin{array}{c}\text{Addition/Subtraction: 5}\\ \text{Multiplication/Division: 3}\end{array}$$

Next find the cofactor matrix:

$$\begin{bmatrix} \begin{vmatrix} 1 & -1 \\ 1 & -3 \end{vmatrix} & -\begin{vmatrix} 1 & -1 \\ -1 & -3 \end{vmatrix} & \begin{vmatrix} 1 & 1 \\ -1 & 1 \end{vmatrix} \\ -\begin{vmatrix} -3 & 1 \\ 1 & -3 \end{vmatrix} & \begin{vmatrix} 2 & 1 \\ -1 & -3 \end{vmatrix} & -\begin{vmatrix} 2 & -3 \\ -1 & 1 \end{vmatrix} \\ \begin{vmatrix} -3 & 1 \\ 1 & -1 \end{vmatrix} & -\begin{vmatrix} 2 & 1 \\ 1 & -1 \end{vmatrix} & \begin{vmatrix} 2 & -3 \\ 1 & 1 \end{vmatrix} \end{bmatrix} = \begin{bmatrix} -2 & 4 & 2 \\ -8 & -5 & 1 \\ 2 & 3 & 5 \end{bmatrix}$$

$$\begin{array}{c}\text{Addition/Subtraction: 14}\\ \text{Multiplication/Division: 25}\end{array}$$

For the purposes of operation counting, taking the transpose of a matrix does not contribute to either operation count. Find the inverse from the cofactor matrix and the determinant (do not count taking the reciprocal of the determinant as an operation):

$$\frac{1}{-14}\begin{bmatrix}-2 & 4 & 2\\-8 & -5 & 1\\2 & 3 & 5\end{bmatrix}^T = \begin{bmatrix}\frac{2}{7} & \frac{4}{7} & -\frac{1}{7}\\-\frac{2}{7} & \frac{5}{14} & -\frac{3}{14}\\-\frac{1}{7} & -\frac{1}{14} & -\frac{5}{14}\end{bmatrix} \begin{array}{c}\text{Addition/Subtraction: 14}\\\text{Multiplication/Division: 34}\end{array}$$

Left multiplying the matrix containing the constraints on the columns by the inverse gives the solutions to the systems:

$$\begin{bmatrix}\frac{2}{7} & \frac{4}{7} & -\frac{1}{7}\\-\frac{2}{7} & \frac{5}{14} & -\frac{3}{14}\\-\frac{1}{7} & -\frac{1}{14} & -\frac{5}{14}\end{bmatrix}\begin{bmatrix}2 & 6 & 0 & -1\\-1 & 4 & 1 & 0\\0 & 5 & -3 & 0\end{bmatrix} = \begin{bmatrix}-\frac{2}{7} & \frac{17}{7} & 1 & -\frac{1}{7}\\-\frac{13}{14} & -\frac{19}{14} & 1 & \frac{2}{7}\\-\frac{3}{14} & -\frac{41}{14} & 1 & \frac{1}{7}\end{bmatrix}$$
$$\text{Addition/Subtraction: 38}$$
$$\text{Multiplication/Division: 70}$$

The solution is read similarly from this matrix as from the last 4 columns of the matrix as a result of Gaussian elimination.

Gaussian elimination is more operation efficient. If Gaussian elimination was used to find the inverse of the matrix instead of the determinant and adjoint method, then the number of operations would be the same.

Problem 3:

Exercise 2d: Find the row interchanges required to solve the linear system via standard Gaussian elimination:

$$x_1 - x_2 + x_3 = 5$$
$$7x_1 + 5x_2 - x_3 = 8$$
$$2x_1 + x_2 + x_3 = 7$$

First Pass:

$$(E_2 - 7E_1) \rightarrow E_2; (E_3 - 2E_1) \rightarrow E_3$$

Second Pass:

$$\left(E_3 - \frac{1}{4}E_2\right) \rightarrow E_3$$

Resulting Matrix:

$$\begin{bmatrix}1 & -1 & 1 & 5\\0 & 12 & -8 & -27\\0 & 0 & 1 & \frac{15}{4}\end{bmatrix}$$

Exercise 4d: Find the row interchanges required to solve the linear system via Gaussian elimination with partial pivoting:

$$x_1 - x_2 + x_3 = 5$$
$$7x_1 + 5x_2 - x_3 = 8$$
$$2x_1 + x_2 + x_3 = 7$$

First Pass:

$$E_1 \leftrightarrow E_2$$

Second Pass:

$$\left(E_2 - \tfrac{1}{7}E_1\right) \to E_2; \left(E_3 - \tfrac{2}{7}E_1\right) \to E_3$$

Third Pass:

$$E_2 \leftrightarrow E_3$$

Fourth Pass:

$$\left(E_3 + \tfrac{4}{3}E_2\right) \to E_3$$

Resulting Matrix:

$$\begin{bmatrix} 7 & 5 & -1 & 8 \\ 0 & \frac{9}{7} & \frac{9}{7} & \frac{38}{7} \\ 0 & 0 & \frac{20}{9} & \frac{80}{9} \end{bmatrix}$$

Exercise 6d: Find the row interchanges required to solve the linear system via Gaussian elimination with scaled partial pivoting.

$$x_1 - x_2 + x_3 = 5$$
$$7x_1 + 5x_2 - x_3 = 8$$
$$2x_1 + x_2 + x_3 = 7$$

Note that all the pivots are the maximum elements in their respective rows (excluding the entries from the $\mathbf{b}$ vector, which scaled partial pivoting ignores), and thus the scaling factors are all 1. Because the scaling factors are only calculated at the beginning of the process and are maintained, this will proceed using the same row interchanges as standard Gaussian elimination as shown in the solution to Exercise 2d.

Problem 4: Use LU factorization to solve the following linear system:

$$2x_1 - x_2 + x_3 = -1$$
$$3x_1 + 3x_2 + 9x_3 = 0$$
$$3x_1 + 3x_2 + 5x_3 = 4$$

LU factorization begins with Gaussian elimination (excluding the $\mathbf{b}$ vector), keeping careful track of the row interchanges performed:

$$A = \begin{bmatrix} 2 & -1 & 1 \\ 3 & 3 & 9 \\ 3 & 3 & 5 \end{bmatrix}$$

Addition/Subtraction: 0
Multiplication/Division: 0

$E_1 \to E_1; \left(E_2 - \frac{3}{2}E_1\right) \to E_2; \left(E_3 - \frac{3}{2}E_1\right) \to E_3$:

$$\begin{bmatrix} 2 & -1 & 1 \\ 0 & \frac{9}{2} & \frac{15}{2} \\ 0 & \frac{9}{2} & \frac{7}{2} \end{bmatrix}$$

Addition/Subtraction: 4
Multiplication/Division: 6

$E_2 \to E_2; (E_3 - E_2) \to E_3$ (note that this step still counts multiplying by 1 as an operation, only inverse operations will be excluded):

$$U = \begin{bmatrix} 2 & -1 & 1 \\ 0 & \frac{9}{2} & \frac{15}{2} \\ 0 & 0 & -4 \end{bmatrix}$$

Addition/Subtraction: 5
Multiplication/Division: 8

There is an implicit third step $E_3 \to E_3$.

The lower triangular matrix is formed from the coefficients used to describe the row swaps, with each column corresponding to each step:

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{3}{2} & 1 & 0 \\ \frac{3}{2} & 1 & 1 \end{bmatrix}$$

Addition/Subtraction: 5
Multiplication/Division: 8

Note that this gives $A = LU$. The vector $\mathbf{y}$ is found by solving:

$$y_1 = \frac{b_1}{L_{1,1}}; y_i = \frac{1}{L_{1,1}}\left[b_i - \sum_{j=1}^{i-1} L_{\mathrm{i,j}} y_j\right], i = 2,3,\ldots,n$$

Unlike during the Gaussian elimination, because this method of generating L always produces ones on the diagonal, any multiplication/division involving any element of the diagonal of L does not count toward the total operation count (because during the elimination, this system just happened to have a coefficient of 1 for the row swap).

$$y_1 = -1$$

Addition/Subtraction: 5
Multiplication/Division: 8

$$y_2 = \frac{3}{2}$$

Addition/Subtraction: 6
Multiplication/Division: 9

$$y_3 = 4$$

Addition/Subtraction: 8
Multiplication/Division: 11

$$\mathbf{y} = \begin{bmatrix} -1 \\ \frac{3}{2} \\ 4 \end{bmatrix}$$

Addition/Subtraction: 8
Multiplication/Division: 11

Solve using the $\mathbf{y}$ vector, the U matrix, and backwards substitution. Note that performing this process is exactly the same as if the original A matrix were augmented with the $\mathbf{b}$ vector and standard Gaussian elimination was used. This is confirmed by the operation counts. The coefficients of the following systems of equations are the coefficients described by the U matrix.

$$\begin{gathered} 2x_1 - x_2 + x_3 = -1 \\ 0x_1 + \frac{9}{2}x_2 + \frac{15}{2}x_3 = \frac{3}{2} \\ 0x_1 + 0x_2 + x_3 = -1 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: 8} \\ \text{Multiplication/Division: 12} \end{gathered}$$

$$\begin{gathered} 2x_1 - x_2 + x_3 = -1 \\ 0x_1 + x_2 + 0x_3 = 2 \\ 0x_1 + 0x_2 + x_3 = -1 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: 9} \\ \text{Multiplication/Division: 14} \end{gathered}$$

$$\begin{gathered} x_1 + 0x_2 + 0x_3 = 1 \\ 0x_1 + x_2 + 0x_3 = 2 \\ 0x_1 + 0x_2 + x_3 = -1 \end{gathered} \qquad \begin{gathered} \text{Addition/Subtraction: 11} \\ \text{Multiplication/Division: 17} \end{gathered}$$

Problem 5: Implement the Jacobi and Gauss-Siedel iterative methods and used them to solve the following system of equations:

$$\begin{gathered} 3x_1 - x_2 + x_3 = 1 \\ 3x_1 + 6x_2 + 2x_3 = 0 \\ 3x_1 + 3x_2 + 7x_3 = 4 \end{gathered}$$

Jacobi Iteration:

k	0	1	2	3	4	5	6	7	8	9
$x_1^{(k)}$	0	0.3333	0.1429	0.0714	0.0408	0.0368	0.0349	0.0352	0.0350	0.0351
$x_2^{(k)}$	0	0	−0.3571	−0.2143	−0.2568	−0.2313	−0.2398	−0.2357	−0.2373	−0.2366
$x_3^{(k)}$	0	0.5714	0.4286	0.6633	0.6327	0.6640	0.6548	0.6592	0.6574	0.6581

Code can be found at the end of the document or in the GitHub page.

Gauss-Siedel:

k	0	1	2	3	4	5	6
$x_1^{(k)}$	0	0.3333	0.1111	0.0529	0.0396	0.0361	0.0354
$x_2^{(k)}$	0	−0.1667	−0.2222	−0.2328	−0.2360	−0.2366	−0.2368
$x_3^{(k)}$	0	0.5000	0.6190	0.6485	0.6556	0.6573	0.6578

Code can be found at the end of the document or in the GitHub page.

Problem 6: Solve the following system using both standard Gaussian elimination and the conjugate gradient method:

$$0.1x_1 + 0.2x_2 = 0.3$$
$$0.2x_1 + 113x_2 = 113.2$$

Perform Gaussian elimination, first rounding the entries to account for two digit rounding arithmetic:

$$0.1x_1 + 0.2x_2 = 0.3$$
$$0.2x_1 + 110x_2 = 110$$

$$0.1x_1 + 0.2x_2 = 0.3$$
$$0x_1 + 110x_2 = 110$$

Perform backwards substitution:

$$0.1x_1 + 0.2x_2 = 0.3$$
$$0x_1 + x_2 = 1$$

$$x_1 + 0x_2 = 1$$
$$0x_1 + x_2 = 1$$

Perform conjugate gradient method, first rounding the entries to account for two digit rounding arithmetic:

$$0.1x_1 + 0.2x_2 = 0.3$$
$$0.2x_1 + 110x_2 = 110$$

Because the preconditioning matrix is the identity matrix, the $\tilde{A}$ matrix is simply the coefficients of the system.

$$\tilde{A} = C^{-1}AC^{-T} = A = \begin{bmatrix} 0.1 & 0.2 \\ 0.2 & 110 \end{bmatrix}$$

Note that for this example, the initial guess will be the zero vector. Define:

$$\mathbf{v}^{(\mathbf{k+1})} = C^{-T}\mathbf{w}^{(\mathbf{k})} + \tilde{s}_k\mathbf{v}^{(\mathbf{k})} = \mathbf{w}^{(\mathbf{k})} + \tilde{s}_k\mathbf{v}^{(\mathbf{k})}; \mathbf{w}^{(\mathbf{k})} = \tilde{\mathbf{b}} - \tilde{A}\tilde{\mathbf{x}}^{(k)}; \tilde{s}_k = \frac{\langle \mathbf{w}^{(\mathbf{k})}, \mathbf{w}^{(\mathbf{k})} \rangle}{\langle \mathbf{w}^{(\mathbf{k-1})}, \mathbf{w}^{(\mathbf{k-1})} \rangle}$$

Start:

$$\mathbf{w} = \mathbf{v}^{(\mathbf{1})} = \begin{bmatrix} 0.3 \\ 110 \end{bmatrix}$$

$$\alpha = \langle \mathbf{w}, \mathbf{w} \rangle = 12000$$

Iterate:

$$\mathbf{w} = \begin{bmatrix} 0.3 \\ 110 \end{bmatrix} - \frac{12000}{\left\langle \begin{bmatrix} 0.3 \\ 110 \end{bmatrix}, \begin{bmatrix} 0.1 & 0.2 \\ 0.2 & 110 \end{bmatrix}\begin{bmatrix} 0.3 \\ 110 \end{bmatrix} \right\rangle} \begin{bmatrix} 0.3 \\ 110 \end{bmatrix} = \begin{bmatrix} 0.3 \\ 110 \end{bmatrix}$$

$$\beta = \left\langle \begin{bmatrix} 0.3 \\ 110 \end{bmatrix}, \begin{bmatrix} 0.3 \\ 110 \end{bmatrix} \right\rangle = 12000$$

$$s_1 = \frac{\beta}{\alpha} = 1$$

$$\mathbf{v}^{(\mathbf{2})} = \begin{bmatrix} 0.3 \\ 110 \end{bmatrix} + \begin{bmatrix} 0.3 \\ 110 \end{bmatrix} = \begin{bmatrix} 0.6 \\ 220 \end{bmatrix}$$

Gaussian elimination is better.

Problem 7: Apply fixed-point iteration and the Gauss-Siedel method to the following system with the following fixed-point problem representation:

$$x_1^2 - 10x_1 + x_2^2 + 8 = 0$$
$$x_1x_2^2 + x_1 - 10x_2 + 8 = 0$$

$$x_1 = g_1(x_1, x_2) = \frac{x_1^2 + x_2^2 + 8}{10}$$
$$x_2 = g_2(x_1, x_2) = \frac{x_1x_2^2 + x_1 + 8}{10}$$

First, take the partials of g_1, g_2 and find their maximum values on $[0,1.5]$.

$$\begin{gathered} 0.2000\,x_1 \\ 0.2000\,x_2 \\ 0.1000\,{x_2}^2 + 0.1000 \\ 0.2000\,x_1\,x_2 \end{gathered}$$

The first two entries are the partial derivative of g_1 with respect to x_1 then x_2, and the last two entries are the partial derivatives of g_2 following the same pattern. Note that all these are maximized when $x_1 = x_2 = 1.5$. This gives:

$$\begin{gathered} 0.3000 \\ 0.3000 \\ 0.3250 \\ 0.4500 \end{gathered}$$

Note that for $K = 0.9$ (and noting that $n = 2$):

$$\begin{matrix} 0.3000 \\ 0.3000 \\ 0.3250 \\ 0.4500 \end{matrix} \leq \frac{K}{n} = 0.45$$

Therefore, by Theorem 10.6, $\mathbf{G} = (g_1, g_2)^T$ has a unique fixed point in $D = \{(x_1, x_2)^T | 0 \leq x_1, x_2 \leq 1.5\}$.

Fixed Point Iteration:

0	0.8000	0.9312	0.9739	0.9898	0.9959	0.9984	0.9994	0.9997	0.9999	1.000	1.000	1.000
0	0.8000	0.9312	0.9739	0.9898	0.9959	0.9984	0.9994	0.9997	0.9999	1.000	1.000	1.000

Each column in this table represents an iteration, with the top row giving successive values of x_1 and the bottom row x_2. Fixed point iteration took 12 iterations to converge.

Gauss Acceleration:

0	0.8000	0.9414	0.9821	0.9945	0.9983	0.9995	0.9998	0.9999	1.000	1.000	1.000
0	0.8800	0.9670	0.9901	0.9969	0.9990	0.9997	0.9999	1.000	1.000	1.000	1.000

Note that this converged 1 iteration sooner than standard fixed point. Code can be found attached or in the GitHub page.

Problem 8: Implement both Newton's method and Broyden's method to solve the following system:

$$3x_1^2 - x_2^2 = 0$$
$$3x_1x_2^2 - x_1^3 - 1 = 0$$

Newton's Method:

k	0	1	2	3	4	5
$x_1^{(k)}$	0	0.6111	0.5037	0.5000	0.5000	0.5000
$x_2^{(k)}$	0	0.8333	0.8525	0.8660	0.8660	0.8660

Code can be found at the end of the document or in the GitHub page.

Broyden's Method:

k	0	1	2	3	4	5	6	7	8
$x_1^{(k)}$	1	0.6111	0.5229	0.4946	0.4957	0.4996	0.5000	0.5000	0.5000
$x_2^{(k)}$	1	0.8333	0.8243	0.8407	0.8547	0.8652	0.8660	0.8660	0.8660

Code can be found at the end of the document or in the GitHub page.

Both methods converge to the solution, though Newton's method accomplishes this in just 5 iterations compared to Broyden's 8. This is expected as this is about the difference between Newton's for a single function and the secant method, which Broyden's is analogous to.

Homework 6

```
clear; clc;
```

Problem 5 (5a, 7a)

```
A = [3 -1 1;
    3 6 2;
    3 3 7];
B = [1; 0; 4];
x_0 = zeros(3, 1);
TOL = 1e-3;

x_Jacobi = JacobiIter(A, B, x_0, TOL);
x_Gauss = GaussSiedel(A, B, x_0, TOL);

sym(x_Jacobi)
```

```
ans =
```

$$\begin{pmatrix} 0 & 0.3333 & 0.1429 & 0.0714 & 0.0408 & 0.0368 & 0.0349 & 0.0352 & 0.0350 & 0.0351 \\ 0 & 0 & -0.3571 & -0.2143 & -0.2568 & -0.2313 & -0.2398 & -0.2357 & -0.2373 & -0.2366 \\ 0 & 0.5714 & 0.4286 & 0.6633 & 0.6327 & 0.6640 & 0.6548 & 0.6592 & 0.6574 & 0.6581 \end{pmatrix}$$

```
sym(x_Gauss)
```

```
ans =
```

$$\begin{pmatrix} 0 & 0.3333 & 0.1111 & 0.0529 & 0.0396 & 0.0361 & 0.0354 \\ 0 & -0.1667 & -0.2222 & -0.2328 & -0.2360 & -0.2366 & -0.2368 \\ 0 & 0.5000 & 0.6190 & 0.6485 & 0.6556 & 0.6573 & 0.6578 \end{pmatrix}$$

Problem 7

```
syms x1 x2

g1(x1, x2) = (x1^2 + x2^2 + 8)/10;
g2(x1, x2) = (x1*x2^2 + x1 + 8)/10;

a(x1, x2) = [diff(g1, x1);
            diff(g1, x2);
            diff(g2, x1);
            diff(g2, x2);]
```

```
a(x1, x2) =
```

$$\begin{pmatrix} 0.2000\,x_1 \\ 0.2000\,x_2 \\ 0.1000\,x_2^2 + 0.1000 \\ 0.2000\,x_1\,x_2 \end{pmatrix}$$

```
a(1.5, 1.5)
```

```
ans =
```

$$\begin{pmatrix} 0.3000 \\ 0.3000 \\ 0.3250 \\ 0.4500 \end{pmatrix}$$

```
x_fixed = zeros(2,1);
XO = zeros(2,1);
TOL = 1e-5;
flag = 0;

for k = 1:1000
    for i = 1:2

        x_fixed(i, k) = g1(XO(1), XO(2));
        x_fixed(i, k) = g2(XO(1), XO(2));

        if (max(abs(x_fixed(:, k) - XO)) < TOL)
            max(abs(x_fixed(:, k) - XO));
            x_fixed = sym([zeros(2,1) x_fixed])
            flag = 1;
            break
        end

    end

    if flag == 1
        break
    end

    XO = x_fixed(:, k);
end
```

```
x_fixed =
```

$$\begin{pmatrix} 0 & 0.8000 & 0.9312 & 0.9739 & 0.9898 & 0.9959 & 0.9984 & 0.9994 & 0.9997 & 0.9999 & 1.000 & 1.000 & 1.000 \\ 0 & 0.8000 & 0.9312 & 0.9739 & 0.9898 & 0.9959 & 0.9984 & 0.9994 & 0.9997 & 0.9999 & 1.000 & 1.000 & 1.000 \end{pmatrix}$$

```
x_gauss_fixed = zeros(2,1);
```

```
x_gauss_fixed = 2×1
     0
     0
```

```
XO = zeros(2,1);
TOL = 1e-5;
flag = 0;
```

```
for k = 1:1000
    for i = 1:2

        if i == 1
            x_gauss_fixed(i, k) = g1(XO(1), XO(2));
        else
            x_gauss_fixed(i, k) = g2(x_gauss_fixed(1, end), XO(2));
        end

        if (max(abs(x_gauss_fixed(:, k) - XO)) < TOL)
            max(abs(x_gauss_fixed(:, k) - XO));
            x_gauss_fixed = sym([zeros(2,1) x_gauss_fixed])
            flag = 1;
            break
        end

    end

    if flag == 1
        break
    end
    XO = x_gauss_fixed(:, k);
end
```

```
x_gauss_fixed =
```

$$\begin{pmatrix} 0 & 0.8000 & 0.9414 & 0.9821 & 0.9945 & 0.9983 & 0.9995 & 0.9998 & 0.9999 & 1.000 & 1.000 & 1.000 \\ 0 & 0.8800 & 0.9670 & 0.9901 & 0.9969 & 0.9990 & 0.9997 & 0.9999 & 1.000 & 1.000 & 1.000 & 1.000 \end{pmatrix}$$

Problem 8

```
syms x1 x2
F(x1, x2) = [3*x1^2 - x2^2; 3*x1*x2^2 - x1^3 - 1];
x_0 = [1; 1];
TOL = 1e-6;

x_sol_NewtonSystem = NewtonSystem(F, x_0, TOL);
x_sol_Broyden = Broyden(F, x_0, TOL);

sym(x_sol_NewtonSystem)
```

```
ans =
```

$$\begin{pmatrix} 0 & 0.6111 & 0.5037 & 0.5000 & 0.5000 & 0.5000 \\ 0 & 0.8333 & 0.8525 & 0.8660 & 0.8660 & 0.8660 \end{pmatrix}$$

```
sym(x_sol_Broyden)
```

```
ans =
```

$$\begin{pmatrix} 1 & 0.6111 & 0.5229 & 0.4946 & 0.4957 & 0.4996 & 0.5000 & 0.5000 & 0.5000 \\ 1 & 0.8333 & 0.8243 & 0.8407 & 0.8547 & 0.8652 & 0.8660 & 0.8660 & 0.8660 \end{pmatrix}$$

```
function [x] = JacobiIter(A, B, x_0, TOL)

    x = x_0;
    XO = x_0;

    for k = 1:1000

        for i = 1:length(x_0)

            sum = 0;

            for j = 1:length(x_0)

                if (j == i)
                    continue
                end

                sum = sum + A(i, j)*XO(j);

            end

            x(i, k) = (1/A(i, i))*(-sum + B(i));

        end

        if (max(abs(x(:, k) - XO)) < TOL)
            x = [x_0 x];
            return
        end

        XO = x(:, k);

    end

end
```

```
function [x] = GaussSiedel(A, B, x_0, TOL)

    x = x_0;
    XO = x_0;

    for k = 1:1000

        for i = 1:length(x_0)

            sum_1 = 0;

            for j = i+1:length(x_0)

                if (j == i)
                    continue
                end

                sum_1 = sum_1 + A(i, j)*XO(j);

            end

            sum_2 = 0;

            for j = 1:i-1

                sum_2 = sum_2 + A(i, j)*x(j, k);

            end

            x(i, k) = (1/A(i, i))*(-sum_1 - sum_2 + B(i));

        end

        if (max(abs(x(:, k) - XO)) < TOL)
            x = [x_0 x];
            return
        end

        XO = x(:, k);

    end

end
```

```
function x_sol = NewtonSystem(F, x_0, TOL)

    syms x1 x2

    x_NewtonSystem = x_0;

    for k = 1:1000
        J(x1, x2) = jacobian(F);
        y = mldivide(double(J(x_NewtonSystem(1), x_NewtonSystem(2))), double(-F↙
(x_NewtonSystem(1), x_NewtonSystem(2))));

        x_NewtonSystem = x_NewtonSystem + y;
        x_sol(:, k) = x_NewtonSystem;

        if max(abs(y)) < TOL
            break
        end
    end

    x_sol = [zeros(2,1) x_sol];

end
```

```
function x_sol = Broyden(F, x_0, TOL)

    syms x1 x2

    x_Broyden = x_0;

    J(x1,x2) = jacobian(F);

    A = inv(J(x_0(1), x_0(2)));
    v = F(x_0(1), x_0(2));

    s = -A*v;
    x_Broyden = x_Broyden + s;
    x_sol(:, 1) = x_Broyden;

    for k = 2:1000

        w = v;
        v = F(x_Broyden(1), x_Broyden(2));
        y = v - w;
        z = -A*y;
        p = -s'*z;
        u_trans = s'*A;
        A = A + (1/p)*(s + z)*u_trans;
        s = -A*v;
        x_Broyden = x_Broyden + s;
        x_sol(:, k) = x_Broyden;
        if (max(abs(s)) < TOL)
            break
        end

    end

    x_sol = [x_0 x_sol];

end
```