1. [12 points] Compute (by hand) the matrix-vector product $Ax$ for

$$A = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -3 & 6 \\ 2 & 0 & -4 \end{bmatrix}. $$

and the column vectors

$$x_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad x_2 = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}, \quad x_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad x_4 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}. $$

2. [16 points] Consider the matrices

$$A = \begin{bmatrix} 1 & 0 \\ 2 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 3 & 2 \\ 0 & -1 \end{bmatrix}. $$

(a) Compute (by hand) $2A - 3B$.

(b) Compute (by hand) $AB$, $BA$, $(AB)^T$, $A^TB^T$, and $B^TA^T$.

(c) From the calculations in part (b): does $AB = BA$?

(d) From the calculations in part (b): does $(AB)^T = A^TB^T$ or does $(AB)^T = B^TA^T$?

3. [36 points: 9 points per part] This problem is designed to warm up your matrix creation and multiplication skills in MATLAB, and to introduce you to a fun application in network theory. The technique involves a slick way to count paths in graphs. (With this method you can analyze, for example, genetic networks or social networks.)

A graph is a collection of nodes connected by edges. The graph below has 7 nodes (gray circles) connected by 9 edges (straight lines).
How “close” are two nodes in this network? One way to assess closeness is to count the number of paths that go from one node to another. For example, there are 2 paths of length 2 from node 1 to node 5 (1–3–5 and 1–4–5), but no paths of length 2 from node 1 to node 7.

As the paths get longer, they get much harder to count. For example, there are already 18 paths of length 4 between nodes 1 and 5 (allowing multiple visits to a node): it is hard to correctly count these by hand. (Try it, if you like.) Matrices come to the rescue!

With each graph, we associate an adjacency matrix that encodes the edges. A graph with \( n \) nodes gives an \( n \times n \) adjacency matrix, \( A \). The entries in \( A \) are all zero, except that the \((j,k)\) of \( A \) is 1 if there is an edge from node \( j \) to node \( k \):

\[
a_{j,k} = \begin{cases} 
1, & \text{there is an edge connecting node } j \text{ to node } k; \\
0, & \text{otherwise.}
\end{cases}
\]

For the graph above, the first row of the adjacency matrix is

\[
[0, 1, 1, 1, 0, 0, 0]
\]

since node 1 is connected to nodes 2, 3, and 4, but not nodes 1, 5, 6, and 7. (By convention, nodes are not connected to themselves, so the diagonal of \( A \) is all zero.)

Neat fact (proved using the formula for matrix-matrix multiplication at the end of the assignment): The number of paths of length \( p \) between node \( j \) and node \( k \) is given by the \((j,k)\) entry of the \( p \)th matrix power, \( A^p \):

\[
A^p = A \cdot A \cdot \ldots \cdot A
\]

Use MATLAB to answer the following questions.

(a) Write down the \( 7 \times 7 \) adjacency matrix \( A \) for the graph shown above and enter it in MATLAB.

(b) Compute \( A^2 \) (thus counting the number of paths of length 2 between every pair of nodes).

(c) How many paths of length 10 are there between nodes 1 and 5?

(d) You can evaluate the importance of node \( j \) by looking at the \((j,j)\) entry of \( A^p \), which reveals the number of paths of length \( p \) from node \( j \) to itself. For example, the \((1,1)\) entry of \( A^2 \) is 3, since there are 3 paths of length 2 that go from node 1 to node 1. (These are 1–2–1, 1–3–1, 1–4–1.)

Look at the diagonal entries of \( A^p \) for various values of \( p \). Based on this data, which node do you think is most important? Explain. (Show some data, but you should not report all the \( A^p \) matrices that you compute. Be judicious.)
4. [36 points: 20 points for (a); 8 points each for (b) and (c)]

Consider the following circuit with six resistors.

(a) Work through the first three steps of our circuit modeling procedure:

(1) compute potential drops, \( \mathbf{e} = \mathbf{v} - \mathbf{A}\mathbf{x} \);
(2) compute the current in each resistor, \( \mathbf{y} = \mathbf{K}\mathbf{e} \);
(3) apply Kirchhoff’s current law, \( \mathbf{0} = \mathbf{A}^T\mathbf{y} \).

At each step, write out the individual scalar equations (e.g., \( e_1 = v_0 - x_1 \)), and then the matrix–vector form that collects those scalar equations together. In particular, specify the entries of \( \mathbf{x} \), \( \mathbf{y} \), \( \mathbf{v} \), \( \mathbf{e} \), and \( \mathbf{A} \).

(b) Now assume \( R_1 = R_2 = R_3 = R_4 = R_5 = R_6 = 1 \). Work out the entries of the matrix \( \mathbf{S} = \mathbf{A}^T\mathbf{K}\mathbf{A} \).

(c) Under the conditions of part (b), use Gaussian elimination (by hand, not MATLAB) to solve the system \( (\mathbf{A}^T\mathbf{K}\mathbf{A})\mathbf{x} = \mathbf{A}^T\mathbf{K}\mathbf{v} \) for the unknown potential vector

\[
\mathbf{x} = \begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4
\end{bmatrix}
\]

(Since the vector \( \mathbf{v} \) contains the voltage variable \( v_0 \), your answer \( \mathbf{x} \) should also contain the variable \( v_0 \), as in equation (2.12) in the course notes. Here is a quick check on your answer: notice that the structure of the circuit and the fact that all resistances are equal ensures that \( x_2 = x_3 \). Does your answer agree?)

5. [optional challenge problem: 6 bonus points: 2 points for (a); 4 points for (b)]

(a) Perform the modeling steps for the branched circuit with 16 resistors in Figure 2.4 of the class notes, a primitive model of a branched neuron. Implement the model in MATLAB (with \( R_j = 1 \Omega \) for all \( j \) and \( v_0 = 1 \) V). Show the values you obtain for \( x_1, \ldots, x_8 \).

(b) Neurons have a more sophisticated branching structure than the model in part (a).

(See [https://en.wikipedia.org/wiki/Neuron](https://en.wikipedia.org/wiki/Neuron) for illustrations.)

Develop a MATLAB code that will implement a model like the example in Figure 2.4, but now with these more general features.
• Let the left “trunk” of the circuit (before the branch) contain \( N \) “compartments” (assemblies of one horizontal and one vertical resistor). (Figure 2.4 has 2 compartments in this trunk; Figure 2.1 has three compartments.) This unit will define nodes \( x_1, \ldots, x_N \), and models the neuron’s long axon.

• At \( x_N \), fork into \( b > 1 \) branches, modeling the neuron’s synapses. (Figure 2.4 has \( b = 2 \) branches.)

• Let each branch have \( M \) compartments. (Figure 2.4 has \( M = 2 \) compartments per branch.) Notice the additional vertical resistors at the start of each branch (resistors \( R_6 \) and \( R_{12} \) in Figure 2.4.) These branches will define nodes:

  \[
  \begin{align*}
  \text{branch 1:} & \quad x_{N+1}, \ldots, x_{N+M+1} \\
  \vdots & \quad \vdots \\
  \text{branch } b & \quad x_{N+b(M+1)-M}, \ldots, x_{N+b(M+1)}.
  \end{align*}
  \]

Test your model with \( N = 32 \), \( b = 4 \), \( M = 16 \) (with \( v_0 = 1 \) V and \( R_j = 1 \) Ω).

• Display the nonzero pattern of \( S = A^T K A \) by producing a “spy plot” using the \texttt{spy(S)} command in MATLAB.

• Produce plots showing \( x_1, \ldots x_N \) (for the axon) and \( x_{N+1}, \ldots x_{N+M+1} \) (for the first branch).

**Appendix.** Proof that \((A^p)_{j,k}\) counts the paths of length \( p \) between nodes \( j \) and \( k \) in a graph. (See Problem 3).

Let \( A \) denote the adjacency matrix for a graph with \( n \) nodes, so that

\[
a_{j,k} = \begin{cases} 
1, & \text{there is an edge connecting node } j \text{ to node } k; \\
0, & \text{otherwise.}
\end{cases}
\]

The proof will use induction. Notice that \( a_{j,k} \) counts the number of paths of length 1 between nodes \( j \) and \( k \). Thus the base case \((p = 1)\) for the induction holds.

Now make the inductive assumption that \((A^p)_{j,k}\) counts the paths of length \( p \) between nodes \( j \) and \( k \). We will prove that \((A^{p+1})\) counts the paths of length \( p + 1 \) between nodes \( j \) and \( k \).

Any path of length \( p + 1 \) from node \( j \) to node \( k \) can be broken into two subpaths:

path of length \( p + 1 \) from \( j \) to \( k \) = (any path of length 1 from \( j \) to \( \ell \)) \( \cup \) (any path of length \( p \) from \( \ell \) to \( k \))

for some node \( \ell \in \{1, \ldots, n\} \). To count all paths of length \( p + 1 \), sum up over all the possible intermediate nodes \( \ell \), and using the facts that:

• \((A)_{j,\ell}\) counts the paths of length 1 from \( j \) to \( \ell \) (base case);

• \((A^p)_{\ell,k}\) counts the paths of length \( p \) from \( \ell \) to \( k \) (inductive assumption).

Thus, we can calculate:

\[
\text{# of paths of length } p + 1 \text{ from } j \text{ to } k = \sum_{\ell=1}^{n} (\text{# of paths of length 1 from } j \text{ to } \ell) \times (\text{# of paths of length } p \text{ from } \ell \text{ to } k) \\
= \sum_{\ell=1}^{n} (A)_{j,\ell}(A^p)_{\ell,k},
\]

but this last formula is simply the matrix-matrix multiplication formula for the \((j,k)\) entry of \( AA^p = A^{p+1} \): the number of paths of length \( p + 1 \) from node \( j \) to node \( k \) is \((A^{p+1})_{j,k}\).

Thus, by induction, we have proved that the number of paths of length \( p \) between nodes \( j \) and \( k \) is \((A^p)_{j,k}\).