Project One Template

MAT350: Applied Linear Algebra

Justin Guida

September 18th 2025

Problem 1

Develop a system of linear equations for the network by writing an equation for each router (A, B, C, D, and E). Make sure to write your final answer as A**x**=**b** where A is the 5x5 coefficient matrix, **x** is the 5x1 vector of unknowns, and **b** is a 5x1 vector of constants.

Solution:

%

% 1.) Logic:

% Total inflow at each router = Total outflow at that router

%

% 2.) Apply Flow-in = Flow-out logic for every Router

%

% Router A:

% **inflow:** 100 Mbps from Sender to Router A

% **outflow:** Router B(x1), Router E(x2), Router C(x2)

% **balance**: $x_1 + 2x_2 = 100$

%

% Router B:

% **inflow:** Router A(x1), Router D(x2)

% **outflow:** Receiver(x3), Router E(x5)

% **balance:** $x_1 + x_2 = x_3 + x_5$

%

% Router C:

% **inflow:** Sender(50 Mbps), Router A(x2)

% **outflow:** Router E(x3), Router D(x5)

% **balance:** $50 + x_2 = x_3 + x_5$

%

% Router D:

% **inflow:** Router E(x4), from Router C(x5)

% **outflow:** Receiver(120 Mbps), Router B(x2)

% **balance:** $x_4 + x_5 = 120 + x_2$

%

% Router E:

% **inflow:** Router A(x2), Router C(x3), Router B(x5)

% **outflow:** Router D(x4)

% **balance:** $x_2 + x_3 + x_5 = x_4$

%

% 3.) Rearranged into Standard Form:

%

% Router A:

% **Standard Form:** $x_1 + 2x_2 + 0x_3 + 0x_4 + 0x_5 = 100$

%

% Router B:

% **Standard Form**: $x_1 + x_2 - x_3 + 0x_4 - x_5 = 0$

%

% Router C:

% **Standard Form:** $0x_1 + x_2 - x_3 + 0x_4 - x_5 = -50$

%

% Router D:

% **Standard Form:** $0x_1 - x_2 + 0x_3 + x_4 + x_5 = 120$

%

% Router E

% **Standard Form:** $0x_1 + x_2 + x_3 - x_4 + x_5 = 0$

% What I did:

I wrote flow-in = flow-out for each router and rearranged to standard form to build A and b.

Problem 2

Use MATLAB to construct the augmented matrix [A **b**] and then perform row reduction using the rref() function. Write out your **reduced matrix and identify the free and basic variables of the system**.

Solution:

```
% Purpose: build A and b from the router equations above
%
% A is the 5x5 coefficient matrix
A = [1, 2, 0, 0, 0; % Router A
1, 1, -1, 0, -1; % Router B
0, 1, -1, 0, -1; % Router C
0, -1, 0, 1, 1;  % Router D
0, 1, 1, -1, 1;] % Router E
```

```
A = 5×5
     1     2     0     0     0
     1     1    -1     0    -1
     0     1    -1     0    -1
     0    -1     0     1     1
     0     1     1    -1     1
```

```
% b is a 5x1 vector of constants.
% **Remember**: the constant is from the right-hand side of a router's
equation.
b = [100; 0; -50; 120; 0]
```

```
b = 5×1
   100
     0
   -50
   120
     0
```

```
% Purpose: Make the augmented matrix and row-reduce to find the pivots.
%
% Construct the augmented matrix [A, b]
% You do this by using concatenation for A and b
%  5×5 A and 5×1 b --> Ab will be 5×6.
Ab = [A b]
```

```
Ab = 5×6
     1     2     0     0     0   100
     1     1    -1     0    -1     0
     0     1    -1     0    -1   -50
     0    -1     0     1     1   120
     0     1     1    -1     1     0
```

```
% Gauss-Jordan elimination to find the Reduced Row Echelon Form
% R holds the result for the solution to the system!
R = rref(Ab)
```

```
R = 5×6
     1     0     0     0     0    50
     0     1     0     0     0    25
     0     0     1     0     0    30
     0     0     0     1     0   100
     0     0     0     0     1    45
```

```
% Now I need to print the reduced matrix
% As the teacher said: don't end the line with ';' if you want MATLAB to
display the result.

% Display Title
disp('Reduced Row Echelon Form of the augmented matrix:');
```

```
Reduced Row Echelon Form of the augmented matrix:
```

```
% Print Augmented Matrix
disp(R);
```

```
     1     0     0     0     0    50
     0     1     0     0     0    25
     0     0     1     0     0    30
     0     0     0     1     0   100
     0     0     0     0     1    45
```

```
% There are no free variables! Corresponding columns contain a leading 1
% x₁, x₂, x₃, x₄, x₅ are all basic variables

% You can also code this to print the free variables!


% Count the columns in A to find the variables
num_vars = size(A,2);



% Loop through the variables 1 --> num_vars
for j= 1:num_vars

    % Now you check for the current columns index J in the list of pivot
% columns using a simple if statement
    if any(R(:, j) == 1) % Check if the current column is a pivot column

        % print basic variables
        fprintf('Variable x%d is a basic variable.\n', j);
    else

        % print free variables
        fprintf('Variable x%d is a free variable.\n', j);

    % end if
    end
% end for loop
end
```

```
Variable x1 is a basic variable.
Variable x2 is a basic variable.
Variable x3 is a basic variable.
Variable x4 is a basic variable.
Variable x5 is a basic variable.
```

```
% What I did:
%
% I formed [A b], ran rref, and found pivots in all 5 columns had no free
variables.
```

Problem 3

Use MATLAB to **compute the LU decomposition of A**, i.e., find A = LU. For this decomposition, find the transformed set of equations L**y** = **b**, where **y** = U**x**. Solve the system of equations L**y** = **b** for the unknown vector **y**.

Solution:

```
% Purpose: LU so we can solve in two simple steps.
%
% Compute the LU decomposition of A
[L, U] = lu(A)
```

```
L = 5×5
    1.0000         0         0         0         0
    1.0000    1.0000         0         0         0
         0   -1.0000    1.0000         0         0
         0    1.0000   -0.5000    1.0000         0
         0   -1.0000         0   -1.0000    1.0000
U = 5×5
     1     2     0     0     0
     0    -1    -1     0    -1
     0     0    -2     0    -2
     0     0     0     1     1
     0     0     0     0     1
```

```
% Purpose: Forward solve for y.
%
% Solve the system of equations Ly = b for the unknown vector y.
y = L\b % Forward substitution to solve for y
```

```
y = 5×1
   100
  -100
  -150
   145
    45
```

```
% Display output to screen using fprintf
% Display as one output box not 2 using \n
fprintf('Solving the system Ly = b for the unknown vector y.\nThe solution
vector y is:\n');
```

```
Solving the system Ly = b for the unknown vector y.
The solution vector y is:
```

```
disp(y);
```

```
   100
  -100
  -150
   145
    45
```

```
% What I did:
% I computed LU and solved Ly=b to get y.
```

Problem 4

Use MATLAB to **compute the inverse** of U using the inv() function.

Solution:

```
% Purpose: To get U⁻¹ and x = U⁻¹y.
%
% Inverse of U using inv() function
Uinv = inv(U);

% Display print message
fprintf('Computing the inverse of U gives the result:\n');
```

```
Computing the inverse of U gives the result:
```

```
disp(Uinv);
```

```
    1.0000    2.0000   -1.0000         0         0
         0   -1.0000    0.5000         0         0
         0         0   -0.5000         0   -1.0000
         0         0         0    1.0000   -1.0000
         0         0         0         0    1.0000
```

```
% What I did:
%
% I computed U^{-1}.
```

Problem 5

Compute the solution to the original system of equations by transforming **y** into **x**, i.e., compute **x** = inv(U)**y**.

Solution:

```
% Compute the solution to the original system of equations by transforming
y into x
% Already did y = L \ b;
% compute x = inv(U)y.
x = Uinv * y; % Compute the solution vector x
fprintf('The final solution for vector x is:\n');
```

```
The final solution for vector x is:
```

```
disp(x);
```

```
    50
    25
    30
   100
    45
```

```
% Expected flows are
% x = [50; 25; 30; 100; 45] Mbps

% Final flows (Mbps): x = [50; 25; 30; 100; 45]
```

Problem 6

Check your answer for x_1 using Cramer's Rule. Use MATLAB to compute the required determinants using the det() function.

Solution:

```
% Purpose: Cramer's Rule check
% Check the answer for x using Cramer''s Rule (using det())
detA = det(A);

% Display the determinant of A formatted to 2 decimal places
fprintf('The determinant of matrix A is: %.2f\n', detA);
```

```
The determinant of matrix A is: 2.00
```

```
num_vars = size(A, 2);  % Need to define
nums = zeros(num_vars,1); % Store the det A_j

% Simple loop to create each A_j and compute the determinant
for j = 1:num_vars
    Aj = A; % Creates the copy of A
    Aj(:, j) = b;  % Replace the j-th column with b
    detAj = det(Aj); % Compute the determinant of A_j
    nums(j) = detAj; % Need to Store the determinant of A_j
    fprintf('The determinant of A_%d is: %.2f\n', j, detAj);
%end for loop
end
```

```
The determinant of A_1 is: 100.00
The determinant of A_2 is: 50.00
The determinant of A_3 is: 60.00
The determinant of A_4 is: 200.00
The determinant of A_5 is: 90.00
```

```
% I can use Cramer''s solution vector:
x_cramer = nums / detA;
disp('The x from Cramer''s Rule =');
```

```
The x from Cramer's Rule =
```

```
disp(x_cramer);
```

```
   50.0000
   25.0000
   30.0000
  100.0000
   45.0000
```

```
% Print as integers no decimals, horizontal
fprintf('Cramer''s: [%.0f %.0f %.0f %.0f %.0f]\n', x_cramer);
```

```
Cramer's: [50 25 30 100 45]
```

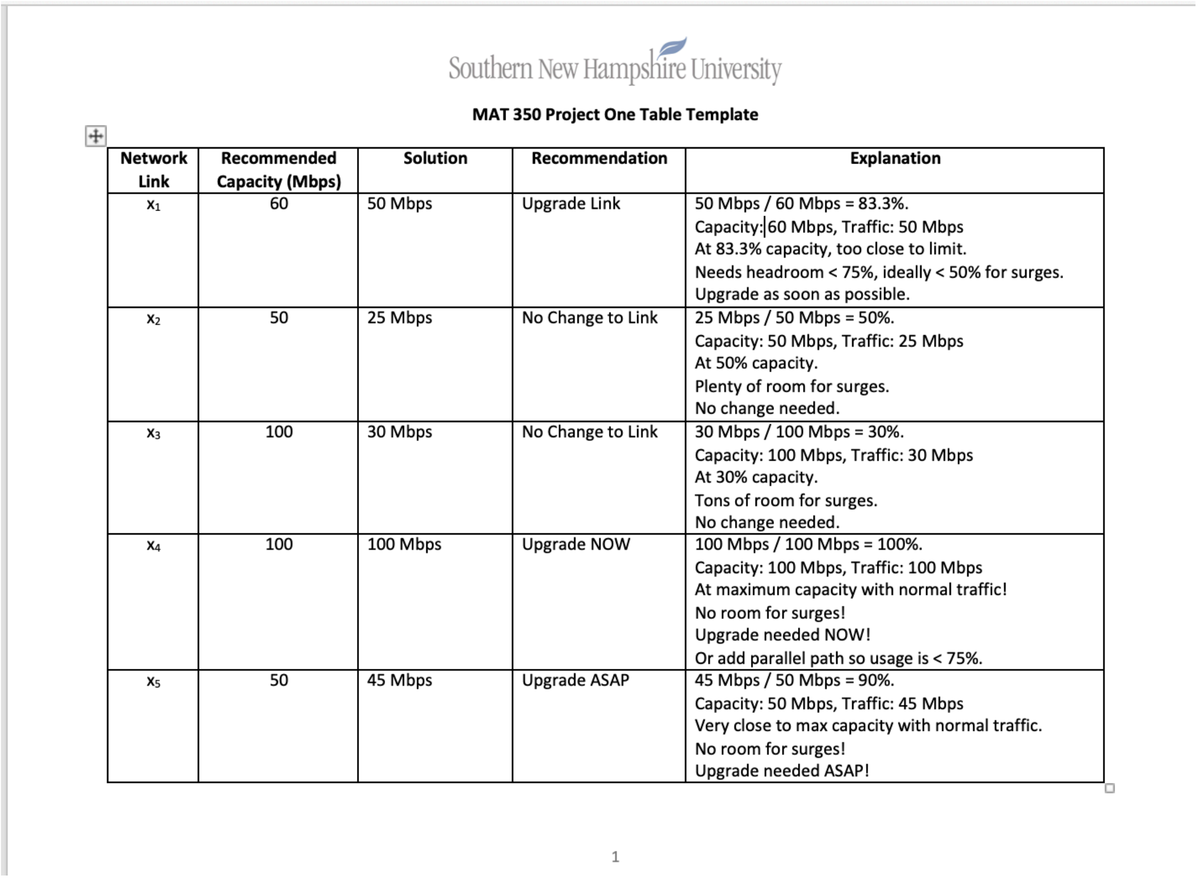
```
% What I did:

% I computed det(A) and each det(Aj), I then got x_cramer which matches x
```

Problem 7

The Project One Table Template, provided in the Project One Supporting Materials section in Brightspace, shows the recommended throughput capacity of each link in the network. Put your solution for the system of equations in the third column so it can be easily compared to the maximum capacity in the second column. In the fourth column of the table, provide recommendations for how the network should be modified based on your network throughput analysis findings. The modification options can be No Change, Remove Link, or Upgrade Link. In the final column, explain how you arrived at your recommendation.

Solution:

Fill out the table in the original project document and export your table as an image. Then, use the ***Insert*** *tab in the MATLAB editor to insert your table as an image.*



Southern New Hampshire University

MAT 350 Project One Table Template

Network Link	Recommended Capacity (Mbps)	Solution	Recommendation	Explanation
x_1	60	50 Mbps	Upgrade Link	50 Mbps / 60 Mbps = 83.3%. Capacity: 60 Mbps, Traffic: 50 Mbps At 83.3% capacity, too close to limit. Needs headroom < 75%, ideally < 50% for surges. Upgrade as soon as possible.
x_2	50	25 Mbps	No Change to Link	25 Mbps / 50 Mbps = 50%. Capacity: 50 Mbps, Traffic: 25 Mbps At 50% capacity. Plenty of room for surges. No change needed.
x_3	100	30 Mbps	No Change to Link	30 Mbps / 100 Mbps = 30%. Capacity: 100 Mbps, Traffic: 30 Mbps At 30% capacity. Tons of room for surges. No change needed.
x_4	100	100 Mbps	Upgrade NOW	100 Mbps / 100 Mbps = 100%. Capacity: 100 Mbps, Traffic: 100 Mbps At maximum capacity with normal traffic! No room for surges! Upgrade needed NOW! Or add parallel path so usage is < 75%.
x_5	50	45 Mbps	Upgrade ASAP	45 Mbps / 50 Mbps = 90%. Capacity: 50 Mbps, Traffic: 45 Mbps Very close to max capacity with normal traffic. No room for surges! Upgrade needed ASAP!

1

% What I did:

% I compared each flow (x1–x5) to its link capacity and only recommended upgrades when utilization was high, to leave headroom for peak usage.