Math 170A: Homework 1

Merrick Qiu

Question 1

1. $$\begin{aligned} X &= L_{ji}^a A \\ &= (I_n + B_{ji}^a)A \\ &= A + B_{ji}^a A \end{aligned}$$

   For all rows $k \neq j$, the kth row of $B_{ji}^a A$ is all zeros since the kth row of B_{ji}^a is all zeros. The jth row of $B_{ji}^a A$ is aR_i since the jith entry of B_{ji}^a is a. Adding A to $B_{ji}^a A$ yields in all the rows being equal to the row in A except for the jth row, which is equal to $R_j + aR_i$. Thus, $X = L_{ji}^a A = A + B_{ji}^a A$ is the result of the row operation $R_j + aR_i \to R_j$.

2. A matrix that is $(L_{ji}^a)^{-1}$ will perform the row operation $R_j - aR_i$. From part (a), we know that this matrix is

   $$\begin{aligned} (L_{ji}^a)^{-1} &= L_{ji}^{-a} \\ &= I_n + B_{ji}^{-a} \\ &= I_n - B_{ji}^a \end{aligned}$$

Question 2

Since $1 \leq i < j < n$ and $1 \leq i < l < k \leq n$, $B^a_{ji} B^b_{kl} = 0$.

$$
\begin{aligned}
L^a_{ji} L^b_{kl} &= (I + B^a_{ji})(I + B^b_{kl}) \\
&= I + B^a_{ji} + B^b_{kl} + B^a_{ji} B^b_{kl} \\
&= I + B^a_{ji} + B^b_{kl} \\
&= I + D
\end{aligned}
$$

Question 3

1. For the base case row n, $a_{nn}x_n = 0$ so $x_n = 0$. For row j where $i < j < n$, assume that $x_k = 0$ for all $k > j$. Then we have that $a_{jj}x_j + \sum_{k=j+1}^{n} a_{jk}x_k = 0$. Substituting in $x_k = 0$ for all $k > j$, implies that $x_j = 0$. By induction we have that $x_j = 0$ for all $i < j \leq n$, so x has the same pattern of zeros as b.

2. We have that $AA^{-1} = I$, which implies that for jth column of A^{-1}

$$Ax_j = e_j \quad j = 1, \cdots, n$$

where n is the size of the matrix and e_j is the j-th unit vector. Since e_j has all zeros after j, invoking part (a) implies that x_j also has all zeros after j. Applying this to every column in A^{-1} shows that A^{-1} is also upper triangular.

Question 4

```
1  function result = multiplyA_BX(A, B, x)
2      [n, n1] = size(A);
3      [n2, n3] = size(B);
4      n4 = size(x);
5      if n ~= n1 | n ~= n2 | n ~= n3 | n ~= n4
6          error("Wrong dimensions");
7      end
8  
9      Bx = zeros(n, 1);
10     for i = 1:n
11         for j = 1:n
12             Bx(i) = Bx(i) + B(i,j) * x(j);
13         end
14     end
15 
16     result = zeros(n, 1);
17     for i = 1:n
18         for j = 1:n
19             result(i) = result(i) + A(i,j) * Bx(j);
20         end
21     end
22 end
```

Line 12 has 2 flops and it is run n^2 times. This is also true of line 19 so there are a total of $4n^2$ flops to calculate $A(Bx)$.

```
1  function result = multiplyA_BX(A, B, x)
2      [n, n1] = size(A);
3      [n2, n3] = size(B);
4      n4 = size(x);
5      if n ~= n1 | n ~= n2 | n ~= n3 | n ~= n4
6          error("Wrong dimensions");
7      end
8
9      AB = zeros(n);
10     for i = 1:n
11         for j = 1:n
12             for k = 1:n
13                 AB(i,j) = AB(i,j) + A(i,k)*B(k,j);
14             end
15         end
16     end
17
18     result = zeros(n, 1);
19     for i = 1:n
20         for j = 1:n
21             result(i) = result(i) + AB(i,j) * x(j);
22         end
23     end
24 end
```

Line 13 has 2 flops and it is run for n^3 times. Line 21 has 2 flops and it is run for n^2 times. There are a total of $2n^3 + 2n^2$ flops to calculate $(AB)x$.

Comparing the two algorithms, we see that $4n^2 < 2n^3 + 2n^2$ so computing $A(Bx)$ is more efficient than $(AB)x$.