CSC336: Assignment 3

Due on Friday, Nov 23, 2018

Zhongtian Ouyang
1002341012

Problem 1

Pivoting

a)

```
b = [1; 2];
x = [1; 1];

fprintf('%12s   %12s   %12s\n', 'gamma', 'error(1)', 'error(2)');
for k = 1:10
    gamma = 10^(-2 * k);
    L = [1 0;1/gamma 1];
    U = [gamma 1-gamma; 0 2-(1/gamma)];
    y = L\b;
    xhat = U\y;
    error = xhat - x;
    fprintf('%12e   %12e   %12e\n', gamma, error(1,1), error(2,1));
end
```

```
>> A3Q1a
       gamma        error(1)        error(2)
1.000000e-02   8.881784e-16   0.000000e+00
1.000000e-04   -1.101341e-13   0.000000e+00
1.000000e-06   2.875566e-11   0.000000e+00
1.000000e-08   5.024759e-09   0.000000e+00
1.000000e-10   8.274037e-08   0.000000e+00
1.000000e-12   -2.212172e-05   0.000000e+00
1.000000e-14   -7.992778e-04   0.000000e+00
1.000000e-16   1.102230e-01   0.000000e+00
1.000000e-18   -1.000000e+00   0.000000e+00
1.000000e-20   -1.000000e+00   0.000000e+00
```

From the above table of output, we can easily obeserve that as the value of gamma decrease, the magnitude of error(1) increase, and eventually reaches -1, which means $\hat{x}[1,1]$ evaluates to 0. In this process error(2) is always very close to 0.

b)

```
b = [1; 2];
x = [1; 1];
P2 = [0 1; 1 0];

fprintf('%12s   %12s   %12s\n', 'gamma', 'error(1)', 'error(2)');
for k = 1:10
    gamma = 10^(-2 * k);
    L2 = [1 0;gamma 1];
    U2 = [1 1; 0 1-2*gamma];
    bhat = P2 * b;
    y = L2\bhat;
    xhat = U2\y;
```

```
    error = xhat - x;
    fprintf('%12e   %12e   %12e\n', gamma, error(1,1), error(2,1));
end
```

```
       gamma         error(1)         error(2)
1.000000e-02    0.000000e+00    0.000000e+00
1.000000e-04    0.000000e+00    0.000000e+00
1.000000e-06    0.000000e+00    0.000000e+00
1.000000e-08    0.000000e+00    0.000000e+00
1.000000e-10    0.000000e+00    0.000000e+00
1.000000e-12    0.000000e+00    0.000000e+00
1.000000e-14    0.000000e+00    0.000000e+00
1.000000e-16    0.000000e+00    0.000000e+00
1.000000e-18    0.000000e+00    0.000000e+00
1.000000e-20    0.000000e+00    0.000000e+00
```

From this new output table, we can see that as gamma decrease, both error(1) and error(2) remain very close to 0. A comparision between result in part (a) and result in part (a) would clearly show that with pivoting, we can get better results when solving linear systems, especially when matrix A have a entry on diagonal with a small absolute value or even 0

c)

```
b = [1; 2];
x = [1; 1];

fprintf('%12s   %12s   %12s\n', 'gamma', 'error(1)', 'error(2)');
for k = 1:10
    gamma = 10^(-2 * k);
    L = [1 0;1/gamma 1];
    U = [gamma 1-gamma; 0 2-(1/gamma)];
    y = L\b;
    xhat = U\y;

    A = [gamma 1-gamma; 1 1];
    r = b - A * xhat;
    z = L\r;
    e = U\z;
    x2 = xhat + e;

    error = x2 - x;
    fprintf('%12e   %12e   %12e\n', gamma, error(1,1), error(2,1));
end
```

```
       gamma         error(1)         error(2)
1.000000e-02    0.000000e+00    0.000000e+00
1.000000e-04    0.000000e+00    0.000000e+00
1.000000e-06    0.000000e+00    0.000000e+00
1.000000e-08    0.000000e+00    0.000000e+00
```

```
1.000000e-10    0.000000e+00    0.000000e+00
1.000000e-12    0.000000e+00    0.000000e+00
1.000000e-14    0.000000e+00    0.000000e+00
1.000000e-16    0.000000e+00    0.000000e+00
1.000000e-18    0.000000e+00    0.000000e+00
1.000000e-20    0.000000e+00    0.000000e+00
```

As gamma decrease, the accuracy of $\tilde{x}$ keep the same, at least the error is 0 for the first seven digits. The effectiveness of iterative refinement is suprsingly good in our case. With only one iteration, it reduce error to an negligible size. The reason should be that our error was so large as mentioned in textbook.

Problem 2

Partial Pivoting vs Complete Pivoting

a)
From the question, we know P = I. With that fact, we know $M_k = I - m_k e_k^T$ and $U = M_4 M_3 M_2 M_1 A$

$$m_1 = \begin{bmatrix} 0 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}, M_1 A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & -1 & 1 & 0 & 2 \\ 0 & -1 & -1 & 1 & 2 \\ 0 & -1 & -1 & -1 & 2 \end{bmatrix}$$

$$m_2 = \begin{bmatrix} 0 \\ 0 \\ -1 \\ -1 \\ -1 \end{bmatrix}, M_2(M_1 A) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & -1 & 1 & 0 & 2 \\ 0 & -1 & -1 & 1 & 2 \\ 0 & -1 & -1 & -1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & -1 & 1 & 4 \\ 0 & 0 & -1 & -1 & 4 \end{bmatrix}$$

$$m_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -1 \\ -1 \end{bmatrix}, M_3(M_2 M_1 A) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & -1 & 1 & 4 \\ 0 & 0 & -1 & -1 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & -1 & 8 \end{bmatrix}$$

$$m_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}, M_4(M_3 M_2 M_1 A) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & -1 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 0 & 16 \end{bmatrix} = U$$

For L, we can use the formula we derived in lecture:

$$L = I + m_1 e_1^T + m_2 e_2^T + m_3 e_3^T + m_4 e_4^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix}$$

b)

Still P = I, so $AQ = LU, M_4M_3M_2M_1AQ_1Q_2Q_3Q_4 = U, L = M_1^{-1}M_2^{-1}M_3^{-1}M_4^{-1}$

$$Q_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}, AQ_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & -1 \\ 1 & -1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 1 & -1 \\ 1 & -1 & -1 & -1 & -1 \end{bmatrix}$$

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{bmatrix}, M_1AQ_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & -2 \\ 0 & -1 & 1 & 0 & -2 \\ 0 & -1 & -1 & 1 & -2 \\ 0 & -1 & -1 & -1 & -2 \end{bmatrix}$$

$$Q_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}, M_1AQ_1Q_2 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 1 \\ 0 & -2 & 1 & 0 & -1 \\ 0 & -2 & -1 & 1 & -1 \\ 0 & -2 & -1 & -1 & -1 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 & 1 \end{bmatrix}, M_2M_1AQ_1Q_2 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & -2 \\ 0 & 0 & -1 & 1 & -2 \\ 0 & 0 & -1 & -1 & -2 \end{bmatrix}$$

$$Q_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, M_2M_1AQ_1Q_2Q_3 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 0 & 1 \\ 0 & 0 & -2 & 1 & -1 \\ 0 & 0 & -2 & -1 & -1 \end{bmatrix}$$

$$M_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 1 \end{bmatrix}, M_3M_2M_1AQ_1Q_2Q_3 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 0 & 1 \\ 0 & 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & -1 & -2 \end{bmatrix}$$

$$Q_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, M_3M_2M_1AQ_1Q_2Q_3Q_4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & -2 & -1 \end{bmatrix}$$

$$M_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}, M_4M_3M_2M_1AQ_1Q_2Q_3Q_4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -2 & 1 & 0 & 0 \\ 0 & 0 & -2 & 1 & 0 \\ 0 & 0 & 0 & -2 & 1 \\ 0 & 0 & 0 & 0 & -2 \end{bmatrix} = U$$

For L, again we can use the formula we derived in lecture:

$$L = M_1^{-1}M_2^{-1}M_3^{-1}M_4^{-1} = I + m_1e_1^T + m_2e_2^T + m_3e_3^T + m_4e_4^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$Q = Q_1Q_2Q_3Q_4 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

c)

```
n = 60;
A = ones(n,n);
A = A - triu(A);
A = eye(n) - A;
A = A + [ones(n-1,1); 0] * [zeros(1,n-1),1];
Q = diag(ones(n-1,1),1);
Q(n,1) = 1;
[L1, U1, P1] = lu(A);
fprintf("2^(59): %d, U1(n,n):%f\n",2^59 , U1(n,n));
[L2, U2] = lu(A*Q);
fprintf("max(U2):%f\n", max(max(abs(U2))));
x = ones(n,1);
b = A * x;
y = L1 \ b;
x1 = U1 \ y;
fprintf("infinity norm of error matrix using patial pivoting:%f\n", norm(x - x1, inf));
y = L2 \ b;
z = U2 \ y;
x2 = Q * z;
fprintf("infinity norm of error matrix using complete pivoting:%f\n", norm(x - x2, inf))
    ;
```

```
>> A3Q2
2^(59): 576460752303423488, U1(n,n):576460752303423488.000000
max(U2):2.000000
infinity norm of error using patial pivoting:1.000000
infinity norm of error using complete pivoting:0.000000
```

From the above output of the program, we can verify the statements in the question. Firstly, the max of U1 matrix is exactly the same as $2^{n-1} = 2^{59} = 576460752303423488$, while the max of U2 matrix is still 2, just as it was for the smaller version. The infinity norm for x - x1 is 1. Since infinity norm is max row sum, and x1 is a 60 x 1 matrix, it means that some value of x1 have an error of 1 to the exact solution which should be 1, making the relative error 100%. We got a poor estimation of x. While at the same time, the infinity norm for x - x2 if 0.000000, showing that every term in x2 is at least very close to x's, which means x2 is a good estimation of x.

Problem 3

Solving Matrix

a)

$$P_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, P_1 A = \begin{bmatrix} 2 & -4 & -2 \\ -1 & 3 & 2 \\ 1 & 1 & -1 \end{bmatrix}, M_1 = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -1/2 & 0 & 1 \end{bmatrix}, M_1 P_1 A = \begin{bmatrix} 2 & -4 & -2 \\ 0 & 1 & 1 \\ 0 & 3 & 0 \end{bmatrix}$$

$$P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, P_2 M_2 P_1 A = \begin{bmatrix} 2 & -4 & -2 \\ 0 & 3 & 0 \\ 0 & 1 & 1 \end{bmatrix}, M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1/3 & 1 \end{bmatrix}, M_2 P_2 M_1 P_1 A = \begin{bmatrix} 2 & -4 & -2 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix} = U$$

$$P = P_2 P_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Find L following the method in lecture note

$$\hat{M}_1 = P_2 M_1 P_2^T = \begin{bmatrix} 1 & 0 & 0 \\ -1/2 & 1 & 0 \\ 1/2 & 0 & 1 \end{bmatrix}, \hat{m}_1 = \begin{bmatrix} 0 \\ 1/2 \\ -1/2 \end{bmatrix}$$

$$L = \hat{M}_1^{-1} M_2^{-2} = I + \hat{m}_1 e_1^T + m_2 e_2^T = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -1/2 & 1/3 & 1 \end{bmatrix}$$

b)

$Ax = b \Rightarrow PAx = Pb \Rightarrow LUx = Pb$ and then we can just solve using the usual $Ly = PbandUx = y$

$$Pb = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$Ly = Pb \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -1/2 & 1/3 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 - 1/2 * y1 = 0 \\ 1 + 1/2 * y1 - 1/3 * y2 = 1 \end{bmatrix}$$

$$Ux = y \Rightarrow \begin{bmatrix} 2 & -4 & -2 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} (0 + 4x_2 + 2x_3)/2 = 1 \\ 0/3 = 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$Ax = \begin{bmatrix} -1 & 3 & 2 \\ 2 & -4 & -2 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = b$$

c)

$$u = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, v = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}, uv^T = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

d)

Following the algorithm from text book, first we solve Az = u using LU factorization from (a):

$$Pb = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$Ly = Pu \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -1/2 & 1/3 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 - 1/2 * y1 = 1 \\ 0 + 1/2 * y1 - 1/3 * y2 = -1/3 \end{bmatrix}$$

$$Uz = y \Rightarrow \begin{bmatrix} 2 & -4 & -2 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ -1/3 \end{bmatrix} \Rightarrow \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} (0 + 4x_2 + 2x_3)/2 = 1/3 \\ 1/3 \\ -1/3 \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ -1/3 \end{bmatrix}$$

Then using the same method, we solve for Ay = b. In fact, y is just x we computed in b:

$$y = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

Now we can compute x for $\hat{A}x = b$

$$v^T y = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = -1, v^T z = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ -1/3 \end{bmatrix} = 1/3$$

$$x = y + \frac{v^T y}{1 - v^T z} z = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} + \frac{-1}{1 - 1/3} \begin{bmatrix} 1/3 \\ 1/3 \\ -1/3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} - \frac{3}{2} \begin{bmatrix} 1/3 \\ 1/3 \\ -1/3 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -1/2 \\ 3/2 \end{bmatrix}$$

This is the result we want, we can verify that $\hat{A}x = b$.

Problem 4

Voice delay

a)

```
function y = perm_a(p,x)
    y = x;
    for i = 1:length(p)
        y([i p(i)]) = y([p(i) i]);
    end
end
```

b)
A little comment on my method: The way q represent the permutation matrix P is where the 1 is for each row in P, which is the same as the order of rows for some vector x after doing a permutation Px. So I just pass in a vector v = [1,2,...,n], do a permutation Pv, the result would be the order of the rows, which is the same as q.

```
function q = perm_b(p)
    q = perm_a(p, (1:length(p)+1).').';
end
```

c)

```
function y = perm_c(q,x)
    y = x;
    for i = 1:length(q)
        y(i) = x(q(i));
    end
end
```

Testing:
The test is run with the following code.

```
p = [5, 4, 9, 10, 6, 8, 10, 9, 10];
x = [1 : 10]';
y1 = perm_a(p,x)
q = perm_b(p)
y2 = perm_c(q,x)
```

The output of the test

```
y1 =

     5
     4
     9
    10
     6
     8
     2
     3
     7
     1


q =

     5     4     9    10     6     8     2     3     7     1


y2 =

     5
     4
     9
    10
     6
     8
     2
     3
     7
     1
```

Problem 5

Gradient descent with momentum

a)

```
function p = perm_d(q)
    p = zeros(1,length(q) - 1);
    cur_x = 1:length(q);
    position = 1:length(q); %stores the position of a value in cur_x
    for i = 1:length(p)
        j = position(q(i));
        p(i) = j;
        position(cur_x(j)) = i;
        position(cur_x(i)) = j;
        cur_x([i j]) = cur_x([j i]);
    end
end
```

The output for testing

```
>> p = [5, 4, 9, 10, 6, 8, 10, 9, 10]

p =

     5     4     9    10     6     8    10     9    10

>> q = perm_b(p)

q =

     5     4     9    10     6     8     2     3     7     1

>> p1 = perm_d(q)

p1 =

     5     4     9    10     6     8    10     9    10
```

My algorithm is in time proportional to n because there is only one for loop which will run n times and the operations in the for loop are O(1)

b)
$Q = P_{n-1}.....P_2P_1 \Rightarrow det(Q) = det(P_{n-1}) \times det(P_2) \times ...... \times det(P_1)$. For p(i) in vector p, if p(i) == i, it means ith row is changed with ith row, so P_i is I, det(P_i) = 1. Otherwise, det(P_i) = -1.

```
function determinant = detQ(p)
    determinant = 1;
    for i = 1:length(p)
        if p(i) == i
```

```
            determinant = determinant * 1;
        else
            determinant = determinant * -1;
        end
    end
end
```

Prove the hint:
if P_k = I, we have proved in class by induction that diagonal matrix's determinant is the product of diagonal. So det(I) = 1*1*1..... = 1. For any elementary permutation matrix P_k that is not I, we can obtain P_k by interchange two rows of I. Then, we can prove $det(P_k) = -1$ by following induction:
Induct on the matrix size n. I_n denotes an identity matrix with size n, P denotes to a row swapping elementary permutation matrix that is not I.
Base Case: n = 2, $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $P_k = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $det(I) = 1, det(P_k) = -1 = -det(I)$
Inductive step:
$n > 2$. Assume for all P with size (n-1) x (n-1), $det(P_k) = -det(I_{n-1}) = -1$
Suppose P^n is an n x n row swapping elementary permuataion matrix, row k and row l are swapped to produce P^n from I_n.
For a positive integer j such that $j \neq k, j \neq l$, we expand on row j to calculate det(P^n). The 1 in j row must also be in column j because j is the two rows we swapped. Let $\hat{P}$ be the (n-1) x (n-1) matrix formed by crossing out row j, column j from P^n. $\hat{P}$ is also a row swapping elementary permutation matrix because the two swapped row are not crossed out. From assumption $det(\hat{P}) = -det(I_{n-1})$. Since $p_{jj} = 1$, every element except p_{ji} is 0:
$det(P^n) = (-1)^{j+j} p_{jj} det(\hat{P}) = (-1)^{j+j} p_{jj} (-1) det(I_{n-1}) = -((-1)^{j+j} p_{jj} det(I_{n-1}))$
If we expand on row j to calculate $det(I_n)$, $det(I_n) = (-1)^{j+j} p_{jj} det(I_{n-1})$
$det(P^n) = -det(I_n) = -1$

We can conclude that any row swapping elementary permutation matrix P that is not I, det(P) = -1