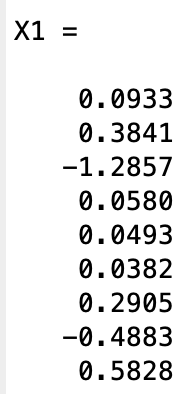
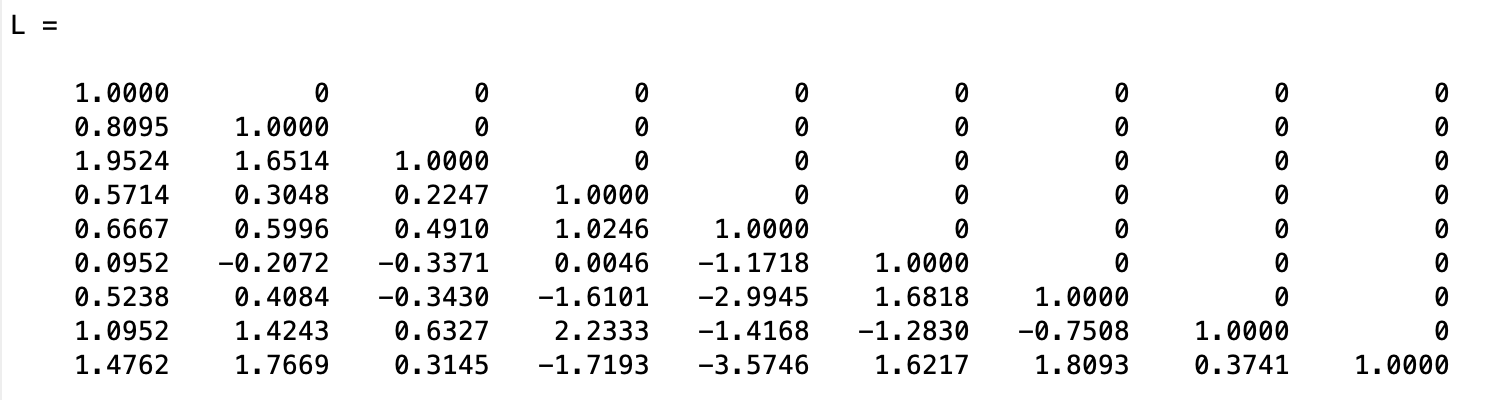
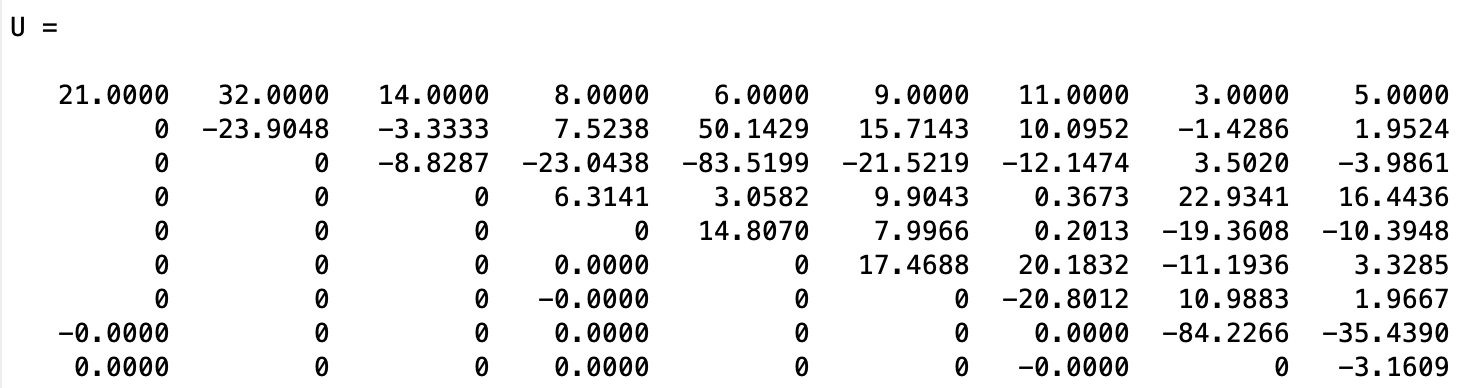
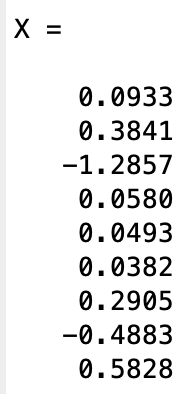
**CS 323 Homework 2**

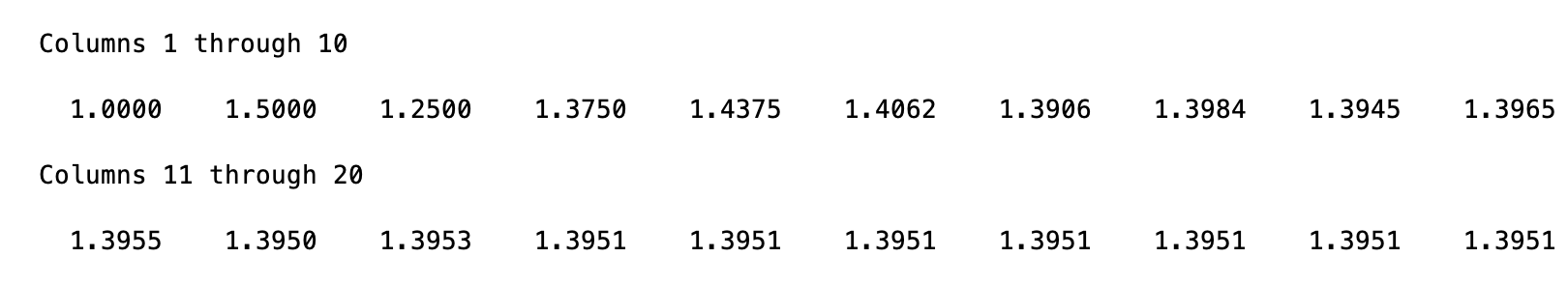
Problem 1

1. Q1: X (Points: 5)
   1. 
2. Q2: Show the specific L matrix, U matrix, and the resulting X (Points: 30)
   1. 
   2. 
   3. 
3. Q3: show the relative error (Points: 5)
   1. The relative error = 6.7411e-16

Problem 2

1. Q1: write down the specific derivations to predict the minimal iterative number of a converged result, and the result. (Points: 15)
   * 1. Initialization: Find a, b such that f(a) \* f(b) < 0 This means there is a root € (a, b) such that f(r) = 0
     2. Set c = (a+b)/2 (Middle point)
     3. Do loop if f(c) == 0, done else: if f(c) \* f(a) < 0, pick the interval [a, c] if f(c) \* f(b) < 0, pick the interval [c, b] Iterate the procedure until stop criteria satisfied (c is the converged result) end loop

* + 1. Therefore,

1. Q2: show the initial guess, results xk at each iterative step, where k represents iterative step, and the converged result. (Points: 30)
   * 1. 
     2. The converged result = 1.3951
2. Q3: plot X axis: k (iterative step number), Y axis: |f(xk)| (the absolute value of residual). (Points: 15)
   * 1. 

function Y=backward\_subsititution (L,B,dim1, dim2)

Y = [];

tmp\_L = L;

tmp\_B = B;

for j = 1:dim1

m = tmp\_L(j, j); % the element in the diagonal

y\_j = tmp\_B(j, 1) / m;

Y = [Y, y\_j];

for i = (j+1):dim1

n = tmp\_L(i, j); % the elements under the diagonal

a = - (n / m);

M = eye(dim1);

M(i, j) = a; %j-th row \* a + i-th row

tmp\_L = M \* tmp\_L;

tmp\_B = M \* tmp\_B;

end

end

Y = Y';

%y = L \ B % test using matlab function

function X=forward\_subsitution(U,Y,dim1, dim2)

X = [];

tmp\_U = U;

tmp\_Y = Y;

for j = dim1:-1:1

m = tmp\_U(j, j); % the element in the diagonal

x\_j = tmp\_Y(j, 1) / m;

X = [x\_j, X];

for i = (j-1):-1:1

n = tmp\_U(i, j); % the elements under the diagonal

a = - (n / m);

M = eye(dim1);

M(i, j) = a; %j-th row \* a + i-th row

tmp\_U = M \* tmp\_U;

tmp\_Y = M \* tmp\_Y;

end

end

X = X';

End

function [L, U] = LUcalculator(A, dim1, dim2)

M\_inv = eye(dim1);

U = A;

for j = 1:dim1

m = U(j, j); % the element in the diagonal

for i = (j+1):dim1

n = U(i, j); % the elements under the diagonal

a = - (n / m);

tmp\_inv = eye(dim1);

tmp\_inv(i, j) = -a;

M\_inv = M\_inv \* tmp\_inv;

U(i, :) = U(i, :) + U(j, :) \* a; %j-th row \* a + i-th row

end

end

L = M\_inv;

End

A = [21 32 14 8 6 9 11 3 5; 17 2 8 14 55 23 19 1 6; 41 23 13 5 11 22 26 7 9;

12 11 5 8 3 15 7 25 19; 14 7 3 5 11 23 8 7 9; 2 8 5 7 1 13 23 11 17;

11 7 9 5 3 8 26 13 17; 23 1 5 19 11 7 9 4 16; 31 5 12 7 13 17 24 3 11];

B = [2; 5; 7; 1; 6; 9; 4; 8; 3];

C = inv(A);

X1 = C \* B;

[dim1, dim2] = size(A);

[L, U] = LUcalculator(A, dim1, dim2);

Y = backward\_subsititution (L,B,dim1, dim2);

X = forward\_subsitution(U,Y,dim1, dim2);

error = norm(X1-X)/norm(X)

function [initial\_guess, count, result]=bise(a\_0, b\_0, tol, F)

result=[];

if F(a\_0)\*F(b\_0)>0

error('cannot use Bisection')

end

count=0; max\_count=100;

c=(a\_0+b\_0)/2;

initial\_guess=c;

result=[result c];

a\_n=a\_0; b\_n=b\_0;

%========Bisection loop====

while (abs(F(c))>tol && b\_n-a\_n>tol && count<=max\_count)

if F(a\_n)\*F(c)<0

a\_n=a\_n; b\_n=c;

elseif F(a\_n)\*F(c)>0

a\_n=c; b\_n=b\_n;

end

c=(a\_n+b\_n)/2;

count=count+1;

if count>=max\_count %%===max iterative steps

break;

end

result=[result c];

end

result

end

%% Bisection method for Nonlinear equations F(x)==0

F =@(x) exp(cos(x)+cos(x^2))+cos(x)-1; % f(x)=x^2-1 ==0

a\_0=0; b\_0=2; % check if f(a)\*f(b)<0

tol=1e-6;

%% ========Bisectio function===

[initial\_guess, count, result]=Bisection(a\_0, b\_0, tol, F);