Numerical methods

Home Work 3

Problems 1, 2, 4, 5

Jacob Needham

A01874339

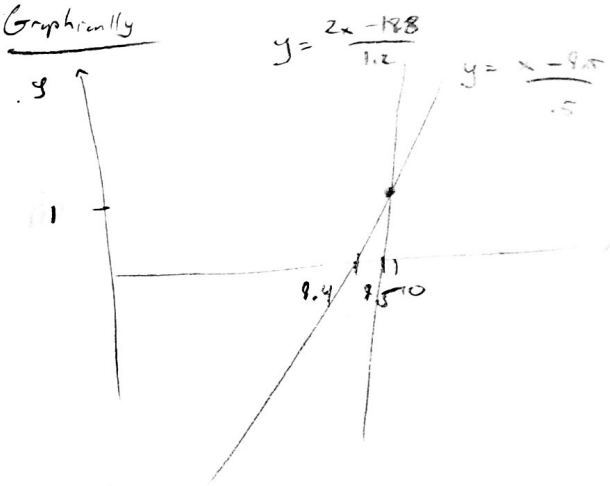
Friday Jan 25th & Mon Jn 28th

1) Problem 9.7 in book.

Given Equations $0.5x_1 - x_2 = -9.5$

$1.02x_1 - 2x_2 = -18.8$

Graphically



2) Det of $\begin{vmatrix} .5 & -1 \\ 1.02 & -2 \end{vmatrix}$

$D = (.5)(-2) - (-1)(1.02)$

$= .02$

3) Via graphing, I know there is a solution to the unknows. The determinate suggests that the solution may be numerically unstable.

4) $x_2 = .5x_1 + 9.5$

$1.02x_1 - 2(.5x_1 + 9.5) = -18.8$

$1.02x_1 - x_1 - 19 = -18.8$

$.02x_1 = .2$

$x_1 = 10$

$x_2 = 14.5$

2) Problem 9.8

Naive Gauss Elimination

$$\begin{bmatrix} 10 & 2 & -1 & 27 \\ -3 & -6 & 2 & -61.5 \\ 1 & 1 & 5 & -21.5 \end{bmatrix}$$

$R_1 \cdot \frac{3}{10} + R_2 = R_2$

$R_1 \cdot \frac{1}{10} - R_3 = R_3$

$$\Downarrow$$

$$\begin{bmatrix} 10 & 2 & -1 & 27 \\ 0 & \frac{-27}{5} & \frac{17}{10} & -53.4 \\ 0 & -\frac{4}{5} & -\frac{51}{10} & 24.2 \end{bmatrix}$$

$R_2 \left(\frac{5}{27}\right)\left(\frac{4}{5}\right) - R_3 = R_3$

$$\Downarrow$$

$$\begin{bmatrix} 10 & 2 & -1 & 27 \\ 0 & \frac{-27}{5} & \frac{17}{10} & -53.4 \\ 0 & 0 & \frac{289}{54} & -\frac{289}{9} \end{bmatrix}$$

$$x_3 = \frac{-289}{9}\left(\frac{54}{289}\right)$$

$$= -6$$

$$x_2 = \left(-53.4 - \frac{17}{10}x_3\right)\left(\frac{5}{27}\right)$$

$$= 8$$

$$x_1 = \left(27 + x_3 - 2x_2\right)\frac{1}{10}$$

$$= \frac{1}{2}$$

```
'''
AUTHOR @Jacob Neeham
Textbook problem 9.18
This progam takes an input matrix defined by the book and outputs
the solution vector. The method for solving the output vector is
though forward elimination suplimented by partial pivoting
'''

#setting up matrix from user
matrix = [[1,2,-1],[5,2,2],[-3,5,-1]]
b = [2,9,1]
n=len(matrix)
o=None
#solution space
Xn = [None]*n
#lists to hold the maxes and ratios of each row
theMax = [None]*n
ratio = [None]*n
k = i = j = 0

def partialPivoting(o,matrix,i):
    for row in matrix:
        theMax[o] = max(row, key=abs)
        o += 1
    for m in range(len(matrix)):
        ratio[m] = abs(matrix[m][m]/theMax[m])
    if i < n:
        if ratio[i-1] < max(ratio[i:], key=abs):
            swapIndex = max(ratio[i:])
            matrix[i-1] = matrix[swapIndex]

def forwardElim(k,i,j,matrix):
    for k in range(1,n):
        for i in range(k+1,n+1):

            #partial pivoting
            o=0
            partialPivoting(o,matrix,i)

            #forward elimination
            factor = matrix[i-1][k-1]/matrix[k-1][k-1]
            for j in range(k+1,n+1):
                matrix[i-1][j-1] = matrix[i-1][j-1] - factor * matrix[k-1][j-1]
            b[i-1] = b[i-1] - factor * b[k-1]

def backwardSub(Xn,b,matrix):
    Xn[n-1] = b[n-1]/matrix[n-1][n-1]
    for i in range(n-1,0,-1):
        Sum = b[i-1]
        for j in range(i+1,n+1):
            Sum = Sum - matrix[i-1][j-1] * Xn[j-1]
        Xn[i-1] = Sum/matrix[i-1][i-1]

def main(o,i,j,k,matrix,b,Xn):
    #calling each function
    forwardElim(k,i,j,matrix)
    backwardSub(Xn,b,matrix)
```

```
    #printing each answer
    w=1
    for element in Xn:
        print("x",w," = ",element,sep="")
        w += 1
main(o,i,j,k,matrix,b,Xn)
```

Problem 10.2

A) $\begin{bmatrix} 10 & 2 & -1 \\ -3 & -6 & 2 \\ 1 & 1 & 5 \end{bmatrix}$ $R_2 = R_1 \frac{3}{10} + R_2$ $R_3 = -R_1 \frac{1}{10} + R_3$

$\begin{bmatrix} 10 & 2 & 1 \\ 0 & -\frac{27}{5} & \frac{17}{10} \\ 0 & -4/5 & -9/10 \end{bmatrix}$ $R_3 = -R_2\left(\frac{5}{27}\right)\left(\frac{4}{5}\right) + R_3$

$\begin{bmatrix} 10 & 2 & 1 \\ 0 & -27/5 & 17/10 \\ 0 & 0 & 289/54 \end{bmatrix}$ = Upper Matrix

The Lower matrix is the collection of all factors used to create the upper

Lower Matrix $= \begin{bmatrix} 1 & 0 & 0 \\ -3/10 & 1 & 0 \\ 1/10 & -4/27 & 1 \end{bmatrix}$

$L \quad U = A$

$\begin{bmatrix} 1 & 0 & 0 \\ -3/10 & 1 & 0 \\ 1/10 & -4/27 & 0 \end{bmatrix} \begin{bmatrix} 10 & 2 & 1 \\ 0 & -27/5 & 17/10 \\ 0 & 0 & 289/54 \end{bmatrix} = \begin{bmatrix} 10 & 2 & -1 \\ -3 & -6 & 2 \\ 1 & 1 & 5 \end{bmatrix}$

B) $[L]\bar{D} = \bar{B}$

$\begin{bmatrix} 1 & 0 & 0 \\ -3/10 & 1 & 0 \\ 1/10 & 4/27 & 1 \end{bmatrix} \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} 27 \\ -16.5 \\ -21.5 \end{bmatrix}$

$D_1 = 27$

$D_2 = -16.5 + D_1\left(\frac{3}{10}\right)$

$= -53.4$

$D_3 = -21.5 - \frac{4}{27} D_2 - \frac{1}{10} D_1$

$= -42.86$

$\bar{U}\bar{x} = \bar{D}$

$\begin{bmatrix} 10 & 2 & 1 \\ 0 & -27/5 & 17/10 \\ 0 & 0 & 289/54 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 27 \\ -53.4 \\ -42.86 \end{bmatrix}$

$x_3 = -42.86\left(\frac{54}{289}\right)$

$= -8.00844$

$x_2 = -\frac{5}{27}\left(-53.4 - \frac{17}{10}x\right)$

$= 7.3677$

$x_3 = \frac{1}{10}(27 - 2x_2 - x$

$= 2.027$

C) $[L]\bar{D} = \bar{B}$

$\begin{bmatrix} 1 & 0 & 0 \\ -3/10 & 1 & 0 \\ 1/10 & 4/27 & 1 \end{bmatrix} \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} 12 \\ 18 \\ -6 \end{bmatrix}$

$D_1 = 12$

$D_2 = 18 + \frac{3}{10} D_1$

$= \frac{108}{5}$

$D_3 = -6 - \frac{4}{27} D_2 - \frac{1}{10} D_1$

$= -10.4$

$[U]\bar{D} = \bar{B}$

$\begin{bmatrix} 10 & 2 & 1 \\ 0 & -27/5 & 17/10 \\ 0 & 0 & 289/54 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 12 \\ 108/5 \\ -10.4 \end{bmatrix}$

$x_3 = -1.94325$

$x_2 = \frac{5}{27}\left(\frac{108}{5} + \frac{17}{10} x_3\right)$

$= 3.38824$

$x_1 = \frac{1}{10}(12 - 2x_2 - x_1)$

$= .71667$

Problem 10.10

A)
$$\begin{bmatrix} 8 & 2 & 1 \\ 3 & 7 & 2 \\ 2 & 3 & 9 \end{bmatrix} \quad R_2 = -\frac{3}{8}R_1 + R_2 \qquad R_3 = -\frac{2}{8}R_1 + R_2$$

$$\begin{bmatrix} 8 & 2 & 1 \\ 0 & \frac{25}{4} & 13/8 \\ 0 & 5/2 & 35/4 \end{bmatrix} \quad R_3 = -\left(\frac{4}{25}\right)\frac{2}{5}R_2 + R_3$$

$$\begin{bmatrix} 8 & 2 & 1 \\ 0 & 25/4 & 13/8 \\ 0 & 0 & \frac{81}{10} \end{bmatrix} = \text{Upper Matrix}$$

Lower Matrix

$$L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{3}{8} & 1 & 0 \\ \frac{2}{8} & \frac{2}{5} & 1 \end{bmatrix}$$

$$L \qquad U \qquad = \qquad A$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 3/8 & 1 & 0 \\ 2/8 & 2/5 & 1 \end{bmatrix} \begin{bmatrix} 8 & 2 & 1 \\ 0 & 25/4 & 13/8 \\ 0 & 0 & 81/10 \end{bmatrix} = \begin{bmatrix} 8 & 2 & 1 \\ 3 & 7 & 2 \\ 2 & 3 & 9 \end{bmatrix}$$

B) $\text{Det } A = \text{Det } L \cdot \text{Det } U$

$$\text{Det } L = \begin{vmatrix} 1 & 0 & 0 \\ 3/8 & 1 & 0 \\ 2/8 & 2/5 & 1 \end{vmatrix}$$

$$= 1$$

$$\text{Det } U = \begin{vmatrix} 8 & 2 & 1 \\ 0 & 25/4 & 13/8 \\ 0 & 0 & 81/10 \end{vmatrix}$$

$$= 8 \cdot 25/4 \cdot 81/10$$

$$= 405$$

$$\therefore \text{Det}(A) = 1 \cdot 432.3$$

$$= 405$$

```
'''
AUTHOR @Jacob Neeham
Textbook problem 9.18
This progam takes an input matrix defined by the user and outputs an inverted
matrix and a solution vector based using LU decompostion. The method for
solving the output vector is naive forward elimination.
'''
#creating base matrix
import np
matrix = np.array([])

#asking user for size of matrix, values to populate matrix, and solution matrix
n = int(input("How many rows and columns would you like in your square matrix? "))
matrix = []*n
for row in range(n):
    matrix.append([input("Values for row {} ".format(row+1)).split(",")])

b = np.array([input("Input the solution vector seperated by commas").split(",")])
b = b.transpose()

#unit vectors to find inverse
b0 = [1,0,0]
b1 = [0,1,0]
b2 = [0,0,1]

#blank vectors for inverse solution
d  = [None]*n
d0 = [None]*n
d1 = [None]*n
d2 = [None]*n

X0 = [None]*n
X1 = [None]*n
X2 = [None]*n

#blank Upper matrix
U = []
for row in range(n):
    U.append([0,0,0])

#blank Lower matrix
L = []
for row in range(n):
    L.append([0,0,0])
for cell in range(n):     #this
    L[cell][cell] = 1

#creating decompose matrix
def decompose(matrix, n):
    for k in range(1,n):
        for i in range(k+1,n+1):
            factor = matrix[k-1][k-1]/matrix[i-1][k-1]
            print('factor', factor)
            matrix[i-1][k-1] = factor
            for j in range(k+1,n+1):
                matrix[i-1][j-1] = matrix[i-1][j-1] - factor*matrix[k-1][j-1]
```

```
#making U matrix
def upper(matrix, n, U):
    for row in range(n):
        for counter in range(row,n):
            U[row][counter] = matrix[row][counter]

#making L matrix
def lower(matrix, n, L):
    for row in range(n):
        for counter in range(row):
            L[row][counter] = matrix[row][counter]

#solving the [L]*d = b
def forwardSub(b,d,L):
    d[0] = b[0]
    Sum = d[0]
    for row in range(1,n):
        d[row] = b[row]-Sum
        Sum = d[row]

    Sum = b[0]
    for row in range(0,n):
        d[row] =  b[row]/(L[row][row-1]+ 1 + Sum)
        Sum = d[row]
#solving [U]*b = x
def backwardSub(Xn,b,matrix):
    Xn[n-1] = b[n-1]/matrix[n-1][n-1]
    for i in range(n-1,0,-1):
        Sum = b[i-1]
        for j in range(i+1,n+1):
            Sum = Sum - matrix[i-1][j-1] * Xn[j-1]
        Xn[i-1] = Sum/matrix[i-1][i-1]

#calling all functions necessary to invert matrix
def inverse(matrix,n,U,L,b0,b1,b2,X0,X1,X2):
    decompose(matrix, n)
    upper(matrix, n, U)
    lower(matrix, n, L)
    forwardSub(b0,d0,L)
    forwardSub(b1,d1,L)
    forwardSub(b2,d2,L)
    backwardSub(X0,d0,U)
    backwardSub(X1,d1,U)
    backwardSub(X2,d2,U)

    #transposing Xn solution vectors in matrix
    Transpose = np.array([X0,X1,X2])
    Transpose.transpose()

    #printing inverse matrix
    print('Input Matrix')
    for row in matrix:
        print(row)
    print("Inverse Matrix")
    for row in Transpose:
        print(row)
```

```
def printLUDecomposition(U,L):
    print('U:')
    for row in U:
        print(row)
    print('L:')
    for row in L:
        print(row)

def solution(matrix,n,U,L,b0,b1,b2,X0,X1,X2):
    decompose(matrix, n)
    upper(matrix, n, U)
    lower(matrix, n, L)
    forwardSub(b,d,L)


def main(matrix,n,U,L,b0,b1,b2,X0,X1,X2):
    inverse(matrix,n,U,L,b0,b1,b2,X0,X1,X2)
    printLUDecomposition(U,L)
```