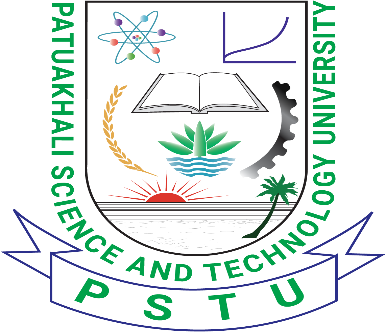
**Lab Problem: 04.**  


**Course code: CCE-312.**

**Course Title: Numerical Methods sessional.**

**Remarks & Signature:**

**Name of the Lab Report:** Solve Real world problem and Simul equation using Gauss-Jordan method.

**Submitted To**

**Professor Dr. Md. Samsuzzaman.**

**Chairman,**

**Department of Computer and Communication Engineering.**

**Faculty of Computer Science & Engineering.**

**Submitted By**

**HASAN AHAMMAD**

**ID No: 1902073**

**Reg No: 08779**

**Level- 3, Semester- 1**

**Session: 2019-2020**

**Faculty of Computer Science & Engineering.**

**Patuakhali Science & Technology University.**

**Dumki, Patuakhali-8602.**

1. **An investor has $10,000 to invest in two types of financial instruments: stocks and bonds. The expected return from stocks is 8%, and from bonds is 5%. The investor wants the total return to be $700. Find the amount invested in each type Using *Gauss-Jordan* method after that implement it using *Python.***

* Solve using Gauss-Jordan method.

from the Given problem we find two equations:-

0.08x+0.05y=700-----------------------------------------------------------(i)

x+y=10000-------------------------------------------------------------------(ii)

here, x=stocks and y=bonds.

Converting given equations into matrix form

R1<-R1/0.08

R2<-R2-R1

R2<-R2/.375

R1<-R1-.625xR2

So, x=6666.667 and y=3333.3333.

Hence amount of stocks invested 6666 and amount of bonds invested 3333.

* **Implement using Python:**

import numpy as np  
  
# Coefficients matrix  
coefficients = np.array([[0.08, 0.05], [1, 1]])  
  
# Constants vector  
constants = np.array([700, 10000])  
  
# Augmented matrix  
augmented\_matrix = np.column\_stack((coefficients, constants))  
  
# Applying Gauss-Jordan elimination  
rows, cols = augmented\_matrix.shape  
  
for i in range(rows):  
 # Normalize the pivot row  
 augmented\_matrix[i] = augmented\_matrix[i] / augmented\_matrix[i, i]  
  
 # Eliminate other rows  
 for j in range(rows):  
 if i != j:  
 augmented\_matrix[j] = augmented\_matrix[j] - augmented\_matrix[j, i] \* augmented\_matrix[i]  
  
# Extract the solution  
solution = augmented\_matrix[:, -1]  
  
# Print the solution  
print("Amount invested in stocks:", solution[0])  
print("Amount invested in bonds:", solution[1])

1. **Solve the following system by the Gauss-Jordan method and Implement it using Python.**

**3x1-0.1x2-0.2x3=7.85**

**0.1x1+7x2-0.3x3=-19.3**

**0.3x1-0.2x2+10x3=71.4**

* **Step-01:** First we have to express the coefficients and the right-hand side as an augmented matrix:
* **Step-02:** Divide Row 1 by 3 to make the leading coefficient 1 in the first row:

r1’=r1/3

* **Step-03:** Subtract 0.1 times Row 1 from Row 2 and 0.3 times Row 1 from Row 3 to make the entries below the leading 1 in Row 1 equal to 0:

r2’=r2-r1x0.1 and r3’=r3-r1x0.3

* **Step-04:** Divide Row 2 by 7.0033 to make the leading coefficient 1 in the second row:

r2’=r2/7.00333

* **Step-05:** Reduction of x2 terms from first and third equation we can use,

r1’=r1+r2x0.0333 and r3’=r3+r2x0.1900

* **Step-06:** Divide Row 3 by 10.01200 to make the leading coefficient 1 in the third row:

r3’=r3/10.01200

* **Step-07:** Finally, reducing x3 terms from equation 1 and 2 we need,

r1’=r1+r3x0.0680629 and r2’=r2+r3x0.0418848

Now, we find the value of x1, x2 and x3,

x1=3.0

x2=-2.50 and

x3=7.0

* **Implement using python:**
* import numpy as np  
    
  # Coefficients matrix  
  coefficients = np.array([[3, -0.1, -0.2], [0.1, 7, -0.3], [0.3, -0.2, 10]])  
    
  # Constants vector  
  constants = np.array([7.85, -19.3, 71.4])  
    
  # Augmented matrix  
  augmented\_matrix = np.column\_stack((coefficients, constants))  
    
  # Applying Gauss-Jordan elimination  
  rows, cols = augmented\_matrix.shape  
    
  for i in range(rows):  
   # Normalize the pivot row  
   augmented\_matrix[i] = augmented\_matrix[i] / augmented\_matrix[i, i]  
    
   # Eliminate other rows  
   for j in range(rows):  
   if i != j:  
   augmented\_matrix[j] = augmented\_matrix[j] - augmented\_matrix[j, i] \* augmented\_matrix[i]  
    
  # Extract the solution  
  solution = augmented\_matrix[:, -1]  
    
  # Print the solution  
  print("Solution:")  
  for i, value in enumerate(solution):  
   print(f"x{i+1} = {value}")

1. **Using Gauss-Jordan method solve the following.**

**.3x1+.52x2+x3=0.01**

**.5x1+.3x2+.5x3=.67**

**.1x1+.3x2+.5x3=-.44**

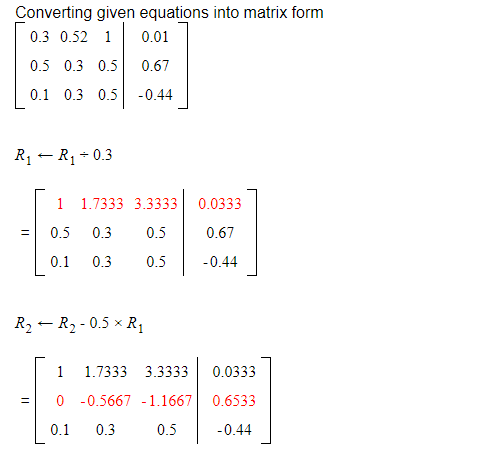
**After that implement it using python.**

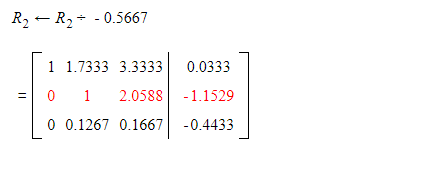
Given equations:

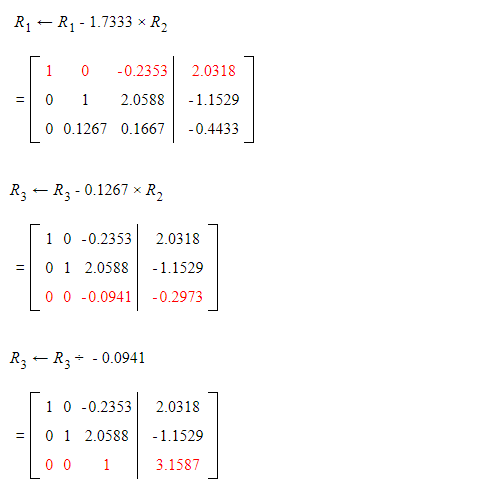
.3x1+.52x2+x3=0.01---------------------------------------------------------(i)

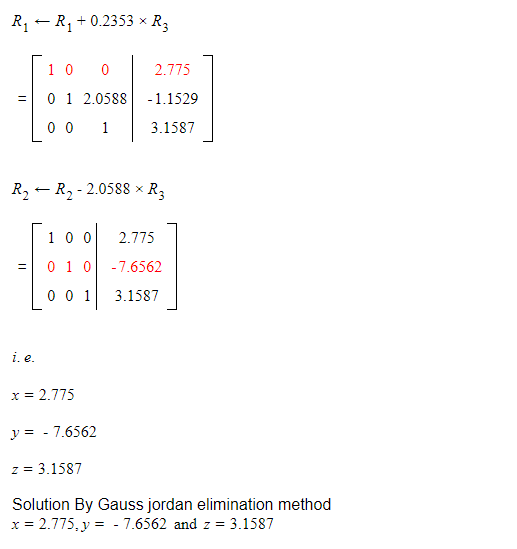
.5x1+.3x2+.5x3=.67---------------------------------------------------------(ii)

.1x1+.3x2+.5x3=-.44-------------------------------------------------------(iii)

****

****

****

****

* **Implement using Python:**
* import numpy as np  
    
  # Coefficients matrix  
  coefficients = np.array([[0.3, 0.52, 1], [0.5, 0.3, 0.5], [0.1, 0.3, 0.5]])  
    
  # Constants vector  
  constants = np.array([0.01, 0.67, -0.44])  
    
  # Augmented matrix  
  augmented\_matrix = np.column\_stack((coefficients, constants))  
    
  # Applying Gauss-Jordan elimination  
  rows, cols = augmented\_matrix.shape  
    
  for i in range(rows):  
   # Find the pivot row  
   pivot\_row = i  
   for j in range(i + 1, rows):  
   if abs(augmented\_matrix[j, i]) > abs(augmented\_matrix[pivot\_row, i]):  
   pivot\_row = j  
    
   # Swap the current row with the pivot row  
   augmented\_matrix[[i, pivot\_row]] = augmented\_matrix[[pivot\_row, i]]  
    
   # Normalize the pivot row  
   augmented\_matrix[i] = augmented\_matrix[i] / augmented\_matrix[i, i]  
    
   # Eliminate other rows  
   for j in range(rows):  
   if i != j:  
   augmented\_matrix[j] = augmented\_matrix[j] - augmented\_matrix[j, i] \* augmented\_matrix[i]  
    
  # Extract the solution  
  solution = augmented\_matrix[:, -1]  
    
  # Print the solution  
  print("Solution:")  
  for i, value in enumerate(solution):  
   print(f"x{i+1} = {value}")