MODUL II SISTEM PERSAMAAN LINEAR

Tujuan :

1. Dapat menentukan penyelesaian sistem persamaan linear dengan metode Gauss, metode Decomposisi Choleski
2. Mencari besarnya kesalahan dari suatu perhitungan solusi sistem persamaan linear dengan metode Gauss, dan metode Decomposisi Choleski

Petunjuk Praktikum :

1. Lengkapi penggal program di bawah ini serta cetak keluarannya.
2. Buatlah laporan praktikum. Adapun isi laporan meliputi :
   a. Dasar teori untuk menentukan solusi system persamaan linear metode tersebut di atas

      Misalkan matrik $A = \begin{bmatrix} 4 & 1 & 2 \\ 1 & 3 & 1 \\ 1 & 2 & 5 \end{bmatrix}$ $B = \begin{bmatrix} 16 \\ 10 \\ 12 \end{bmatrix}$

   b. Pembahasan Program
   c. Pembahasan hasil/keluaran

A. Penggal Program SPL_Metode Gauss:

```
{*************************************************************}
{       Program untuk Menyelesaian Sistem Persamaan Linear      }
{                 Ax = B   dengan Metode  Gauss                 }
{             Dibuat oleh :                                     }
{                   Nama        :                               }
{                   NIM         :                               }
{                   Prog.Studi  :                               }
{*************************************************************}
```

```
## module gaussElimin
''' x = gaussElimin(a,b).
   penyelesaian [a]{b} = {x} dengan metode Eliminasi Gauss.
'''

import numpy as np

def gaussElimin(a,b):
   n = len(b)
   # Elimination Phase
   for k in range(0,n-1):
      for i in range(k+1,n):
         if a[i,k] != 0.0:
            lam = a [i,k]/a[k,k]
            a[i,k+1:n] = a[i,k+1:n] - lam*a[k,k+1:n]
            b[i] = b[i] - lam*b[k]
   # Back substitution
   for k in range(n-1,-1,-1):
      b[k] = (b[k] - np.dot(a[k,k+1:n],b[k+1:n]))/a[k,k]
   return b
```

contoh code penggunaan metode eliminasi Gauss

```
#!/usr/bin/python
## Contoh Metode eliminasi Gauss
import numpy as np
from gaussElimin import *
def vandermode(v):
   n = len(v)
   a = np.zeros((n,n))
   for j in range(n):
      a[:,j] = v**(n-j-1)
   return a

v = np.array([1.0, 1.2, 1.4, 1.6, 1.8, 2.0])
b = np.array([0.0, 1.0, 0.0, 1.0, 0.0, 1.0])
a = vandermode(v)
aOrig = a.copy() # Save original matrix
bOrig = b.copy() # and the constant vector
x = gaussElimin(a,b)
det = np.prod(np.diagonal(a))
print('a = ',a)
print('b = ',b)
print('x =\n',x)
print('\ndet =',det)
print('\nCheck result: [a]{x} - b =\n',np.dot(aOrig,x) - bOrig)
input('\nPress return to exit')
```

B. Penggal Program SPL_Metode Decomposisi Choleski

```
{*************************************************************}
{          Program untuk Menyelesaian Sistem Persamaan Linear                }
{                      Ax = B   dengan Metode  Dekomposisi Choleski                          }
{                    Dibuat oleh :                                                    }
{                              Nama              :                                    }
{                              NIM               :                                    }
{                              Prog.Studi        :                                    }
{*************************************************************}
## module choleski
''' L = choleski(a)
   Dekomposisi Choleski : [L][L]transpose = [a]
   x = choleskiSol(L,b)
'''
import numpy as np
import math
import error
def choleski(a):
   n = len(a)
   for k in range(n):
      try:
        a[k,k] = math.sqrt(a[k,k] \
             - np.dot(a[k,0:k],a[k,0:k]))
      except ValueError:
         error.err('Matrix is not positive definite')
      for i in range(k+1,n):
         a[i,k] = (a[i,k] - np.dot(a[i,0:k],a[k,0:k]))/a[k,k]
   for k in range(1,n): a[0:k,k] = 0.0
   return a


## module choleskiSol
''' L = choleski(a)
   Dekomposisi Choleski : [L][L]transpose = [a]
   x = choleskiSol(L,b)
'''
import numpy as np
import math
import error

def choleskiSol(L,b):
   n = len(b)
   # Solution of [L]{y} = {b}
   for k in range(n):
      b[k] = (b[k] - np.dot(L[k,0:k],b[0:k]))/L[k,k]
   # Solution of [L_transpose]{x} = {y}
   for k in range(n-1,-1,-1):
      b[k] = (b[k] - np.dot(L[k+1:n,k],b[k+1:n]))/L[k,k]
   return b
```

```
#!/usr/bin/python
## contoh dekomposisi choleski

import numpy as np
from choleski import *
from choleskiSol import *
a = np.array([[ 1.44, -0.36, 5.52, 0.0], \
        [-0.36, 10.33, -7.78, 0.0], \
        [ 5.52, -7.78, 28.40, 9.0], \
        [ 0.0, 0.0, 9.0, 61.0]])
b = np.array([0.04, -2.15, 0.0, 0.88])
aOrig = a.copy()
L = choleski(a)
x = choleskiSol(L,b)
print("x =",x)
print('\nCheck: A*x =\n',np.dot(aOrig,x))
input("\nPress return to exit")
```