ME426: Applied Computational Methods in Mechanical Sciences

# Submitted By: Himanshu Kumar (16ME234) August 14, 2019

# Assignment 2

PROBLEM STATEMENT:

Solving the linear system AX=B by cholesky, Doolittle and croute’s method.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | 1 | 1 | 3 | 2 |
| 1 | 2 | 1 | 9 | 5 |
| 1 | 2 | 9 | 1 | 5 |
| 3 | 1 | 1 | 7 | 1 |
| 2 | 1 | 5 | 1 | 8 |
|  |  |  |  |  |

|  |
| --- |
| -2 |
| 4 |
| 3 |
| -5 |
| 1 |

A = B =

Python Code:

import math

import time

a = [[2,1,1,3,2],

[1,2,2,1,1],

[1,2,9,1,5],

[3,1,1,7,1],

[2,1,5,1,8]]

b = [-2,4,3,-5,1]

def disp\_mat(z,n):

for row in range(n):

print(z[row])

def doolittle(x,b):

for i in range(0,len(x)):

if (len(x[i])==len(x)):

pass

else:

print("\n Non-Square matrix, returning None")

return(None)

#for square matrix

n = len(x)

u= [[0 for i in range(n)] for j in range(n)]

l= [[0 for i in range(n)] for j in range(n)]

for i in range(n):

l[i][i]=1

for j in range(i,n):

s= sum(u[k][j] \* l[i][k] for k in range(i-1))

u[i][j] = x[i][j] - s

for j in range(i+1,n):

s= sum(l[j][k]\*u[k][i] for k in range(i-1))

l[j][i] = (x[j][i] - s) / u[i][i]

print("\n doolittle U")

disp\_mat(u,n)

print("\n doolittle L")

disp\_mat(l,n)

# two steps : 1) LZ=B 2)UX=Z

z=[0 for i in range(n)]

sol=[0 for i in range(n)]

for i in range(n):

s= sum(l[i][j]\*z[j] for j in range(i-1))

z[i] = b[i]- s

for c in range(n):

i=(n-1)-c

s=sum(u[i][j]\*sol[j] for j in range(i+1,n))

sol[i]= (z[i]-s)/u[i][i]

print(sol)

return(sol)

def croute(x,b):

for i in range(0,len(x)):

if (len(x[i])==len(x)):

pass

else:

print("\n Non-Square matrix, returning None")

return(None)

#for square matrix

n = len(x)

u= [[0 for i in range(n)] for j in range(n)]

l= [[0 for i in range(n)] for j in range(n)]

for i in range(n):

u[i][i]=1

for j in range(n):

s= sum(l[j][k]\*u[k][i] for k in range(i-1))

l[j][i]= (a[j][i]-s)

for j in range(i+1,n):

s= sum(l[i][k]\*u[k][j] for k in range(i-1))

u[i][j] = (a[i][j] - s)/l[i][i]

print("\n croute U")

disp\_mat(u,n)

print("\n croute L")

disp\_mat(l,n)

z=[0 for i in range(n)]

sol=[0 for i in range(n)]

for i in range(n):

s= sum(l[i][j]\*z[j] for j in range(i-1))

z[i] = (b[i]- s)/l[i][i]

for c in range(n):

i=(n-1)-c

s=sum(u[i][j]\*sol[j] for j in range(i+1,n))

sol[i]= (z[i]-s)

print(sol)

return(sol)

def cholesky(x,b):

n = len(x)

for i in range(0,len(x)):

if (len(x[i])==len(x)):

pass

else:

print("\n Non-Square matrix, returning None")

return(None)

for i in range(n):

for j in range(i):

if(x[i][j] != x[j][i]):

print("\n Non-Symmetric matrix, returning None")

return(None)

u= [[0 for i in range(n)] for j in range(n)]

z=[0 for i in range(n)]

sol=[0 for i in range(n)]

for i in range(n):

s = sum(u[i][k]\*u[i][k] for k in range(i-1))

u[i][i] = math.sqrt(x[i][i]-s)

for j in range(i+1,n):

s = sum(u[k][i]\*u[k][j] for k in range(i-1))

u[i][j] = (x[i][j]-s)/u[i][i]

print("\n cholesky U")

disp\_mat(u,n)

for i in range(n):

s = sum(u[i][j]\*z[j] for j in range(i-1))

z[i]=(b[i]-s)/u[i][i]

for c in range(n):

i=(n-1)-c

s= sum(u[i][j]\*sol[j] for j in range(i+1,n))

sol[i]= (z[i]-s)/u[i][i]

print(sol)

return(sol)

# ans= cholesky(a,b)

# ans = doolittle(a,b)

# ans = croute(a,b)

try:

ans= cholesky(a,b)

if(ans!= None):

pass

except:

try:

ans = doolittle(a,b)

except:

ans = croute(a,b)

print ("\n CPU time: ", time.process\_time(),'s')

RESULT:

[-6.4183673469387745, 4.836734693877551, -1.0816326530612244, 1.2653061224489794, 1.6428571428571428]

CPU TIMING:

1.Cholesky = CPU time: 0.125 s

2.Doolittle = CPU time: 0.171875 s

3.Croute = CPU time: 0.140625 s

PROBLEM STATEMENT:

Solve the Tri-diagonal matrix system by Thomas algorithm.

|  |  |  |  |
| --- | --- | --- | --- |
| 2.08 | -1 | 0 | 0 |
| -1 | 2.08 | -1 | 0 |
| 0 | -1 | 2.08 | -1 |
| 0 | 0 | -1 | 2.08 |

|  |
| --- |
| 41.6 |
| 1.6 |
| 1.6 |
| 201.6 |

A = B =

Python Code:

import time

a= [[2.08,-1,0,0],

[-1,2.08,-1,0],

[0,-1,2.08,-1],

[0,0,-1,2.08]]

b= [46,1.6,1.6,201.6]

def thomas(a,b):

n= len(b)

for i in range(1,n):

a[i][i-1]=a[i][i-1]/a[i-1][i-1]

a[i][i] = a[i][i]-a[i][i-1]\*a[i-1][i];

b[i]= b[i]-a[i][i-1]\*b[i-1]

a[i][i-1]=0

#backward substitution

x=[0 for i in range(n)]

x[n-1]= b[n-1]/a[n-1][n-1]

for k in range(n-1):

i=n-2-k

x[i] = ( b[i] + x[i+1] )/ a[i][i]

print(x)

thomas(a,b)

print ("\n CPU time: ", time.process\_time(),'s')

RESULT:

[61.07393661902537, 85.43378816757277, 115.028342769526, 152.22516479304133]

CPU time: 0.125 s