

Zadanie rozwiązane za pomocą wzoru Shermana-Morrisona oraz funkcji solve z bilbioteki numpy.linalg która rozwiązuje układ rówań metoda LU.

**import** numpy **as** np  
**from** numpy.linalg **import** norm  
**from** numpy.linalg **import** solve  
A=np.matrix([  
[-116.66654,583.33346, -333.33308, 100.00012, 100.00012],  
[ 583.33346, -116.66654, -333.33308, 100.00012, 100.00012],  
[-333.33308, -333.33308, 133.33383, 200.00025, 200.00025],  
[ 100.00012, 100.00012, 200.00025, 50.000125, -649.99988],  
[ 100.00012, 100.00012, 200.00025,-649.99988,50.000125  
]])  
b1=np.array([  
 [-0.33388066],  
 [ 1.08033290],  
 [-0.98559856],  
 [ 1.31947922],  
 [-0.09473435]  
])  
b2=np.array([  
 [-0.33388066],  
 [ 1.08033290],  
 [-0.98559855],  
 [ 1.32655028],  
 [-0.10180541],  
])  
b3=np.array([  
 [ 0.72677951],  
 [ 0.72677951],  
 [-0.27849178],  
 [ 0.96592583],  
 [ 0.96592583],  
])  
b4=np.array([  
 [ 0.73031505],  
 [ 0.73031505],  
 [-0.27142071],  
 [ 0.96946136],  
 [ 0.96946136],  
])  
morison\_mat=np.matrix([[1, 0, 0, 0, 1],  
 [0, 0, 0, 0, 0],  
 [0, 0, 0, 0, 0],  
 [0, 0, 0, 0, 0],  
 [1, 0, 0, 0, 1]])  
**def Morison**(A,b1):  
 Abis=A-morison\_mat  
 u=np.array([1,0,0,0,1])  
 z=solve(Abis,b1)  
 q=np.array([solve(Abis,u)])  
 v=u.transpose()  
 w=z-((v.dot(z))/(1+v.dot(q.transpose())))\*q.transpose()  
 **return** (w)  
  
b=[b1,b2,b3,b4]  
z=[]  
**for** i **in** range(4):  
 z.append(Morison(A,b[i]))  
n=[]  
n.append(norm(b1-b2))  
n.append(norm(b3-b4))  
n.append(norm((z[0]-z[1])/norm(b1-b2)))  
n.append(norm((z[2]-z[3])/norm(b3-b4)))  
print (n)

**Rozwiązania:**

**||b1-b2||=0.0099999889523588337**

**||b3-b4||=0.010000003094494577**

**||z1-z2||/||b1-b2||=0.001595176674760602**

**||z3-z4||/||b3-b4||=1003.7641153590034**