

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 4 & 7 & 5 & 7 & 4 & | & 136 \\ 1 & 5 & 1 & 3 & 6 & | & 58 \\ 4 & 4 & 2 & 2 & 4 & | & 74 \\ 1 & 4 & 5 & 4 & 5 & | & 92 \\ 6 & 1 & 4 & 4 & 1 & | & 94 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t
---	---	---	---	---



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 3 & 1 & 7 & 6 & 5 \\ 1 & 5 & 3 & 3 & 7 \\ 7 & 1 & 1 & 3 & 5 \\ 2 & 3 & 6 & 7 & 1 \\ 1 & 4 & 1 & 5 & 6 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 62 \\ 79 \\ 55 \\ 54 \\ 72 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -\mathbf{9} & -\mathbf{11} \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 4 & 3 & 2 & 7 & 43 \\ 7 & 2 & 4 & 1 & 67 \\ 2 & 7 & 3 & 5 & 54 \\ 5 & 2 & 4 & 3 & 61 \\ 6 & 6 & 3 & 6 & 65 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t
---	---	---	---	---



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -\mathbf{9} & -\mathbf{11} \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
    import numpy as np
    a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
    b=np.array([23,27,25],float)
    lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 7 & 7 & 5 & 2 & 5 \\ 7 & 7 & 4 & 6 & 1 \\ 6 & 2 & 4 & 4 & 5 \\ 5 & 1 & 6 & 6 & 7 \\ 4 & 5 & 5 & 5 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 141 \\ 128 \\ 119 \\ 146 \\ 114 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A .

A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
    import numpy as np
    a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
    b=np.array([23,27,25],float)
    lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 3 & 6 & 2 & 2 & 5 \\ 7 & 5 & 7 & 7 & 7 \\ 4 & 1 & 3 & 7 & 7 \\ 5 & 5 & 5 & 1 & 1 \\ 7 & 6 & 4 & 4 & 7 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 69 \\ 127 \\ 92 \\ 55 \\ 103 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
    import numpy as np
    a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
    b=np.array([23,27,25],float)
    lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 4 & 2 & 5 & 7 & 7 \\ 3 & 1 & 5 & 4 & 7 \\ 2 & 4 & 6 & 6 & 1 \\ 2 & 6 & 4 & 1 & 7 \\ 3 & 2 & 6 & 6 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 116 \\ 93 \\ 56 \\ 63 \\ 91 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A .

A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
    import numpy as np
    a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
    b=np.array([23,27,25],float)
    lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 5 & 2 & 2 & 7 & 5 \\ 2 & 6 & 1 & 2 & 2 \\ 4 & 5 & 6 & 1 & 1 \\ 3 & 7 & 4 & 7 & 6 \\ 5 & 3 & 7 & 6 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 94 \\ 51 \\ 37 \\ 108 \\ 84 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 5 & 7 & 4 & 2 & 5 \\ 5 & 4 & 6 & 3 & 3 \\ 6 & 5 & 6 & 5 & 3 \\ 7 & 3 & 7 & 5 & 7 \\ 5 & 2 & 3 & 6 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 108 \\ 111 \\ 135 \\ 151 \\ 104 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 3 & 6 & 5 & 6 & 1 & 61 \\ 6 & 1 & 4 & 5 & 3 & 73 \\ 7 & 4 & 4 & 3 & 6 & 99 \\ 3 & 1 & 4 & 2 & 5 & 46 \\ 1 & 3 & 3 & 4 & 6 & 44 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t
---	---	---	---	---



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 3 & 7 & 7 & 1 & 2 \\ 5 & 7 & 5 & 6 & 1 \\ 5 & 1 & 2 & 6 & 3 \\ 7 & 7 & 5 & 1 & 4 \\ 1 & 6 & 4 & 4 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 57 \\ 109 \\ 83 \\ 71 \\ 66 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A .

A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
    import numpy as np
    a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
    b=np.array([23,27,25],float)
    lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 4 & 1 & 7 & 3 & 7 & 83 \\ 5 & 3 & 3 & 4 & 5 & 67 \\ 1 & 1 & 5 & 2 & 2 & 46 \\ 5 & 7 & 3 & 4 & 5 & 71 \\ 1 & 1 & 4 & 7 & 2 & 71 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 4 & 3 & 2 & 5 & 4 & 35 \\ 5 & 2 & 5 & 5 & 5 & 33 \\ 1 & 1 & 7 & 2 & 6 & 14 \\ 5 & 6 & 1 & 3 & 7 & 55 \\ 6 & 6 & 7 & 3 & 4 & 46 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A – o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o “serviço” diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 1 & 4 & 7 & 2 & 1 & | & 60 \\ 3 & 5 & 6 & 7 & 4 & | & 91 \\ 4 & 2 & 6 & 1 & 1 & | & 54 \\ 5 & 3 & 5 & 3 & 6 & | & 83 \\ 5 & 7 & 4 & 4 & 3 & | & 109 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 3 & 1 & 4 & 3 & 2 & 27 \\ 1 & 6 & 2 & 1 & 4 & 23 \\ 4 & 5 & 3 & 5 & 7 & 53 \\ 4 & 6 & 3 & 3 & 6 & 48 \\ 7 & 4 & 2 & 7 & 7 & 76 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A .

A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 7 & 6 & 3 & 2 & 6 & 88 \\ 1 & 7 & 2 & 4 & 2 & 62 \\ 1 & 1 & 3 & 5 & 6 & 85 \\ 4 & 3 & 7 & 3 & 4 & 60 \\ 5 & 1 & 3 & 5 & 4 & 71 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -\mathbf{9} & -\mathbf{11} \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 2 & 4 & 6 & 3 & 4 & 83 \\ 6 & 1 & 7 & 2 & 3 & 56 \\ 1 & 1 & 7 & 4 & 6 & 82 \\ 3 & 6 & 4 & 3 & 3 & 83 \\ 3 & 2 & 3 & 7 & 6 & 93 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -\mathbf{9} & -\mathbf{11} \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 5 & 2 & 6 & 4 & 6 \\ 3 & 4 & 3 & 6 & 4 \\ 4 & 2 & 2 & 2 & 6 \\ 4 & 4 & 3 & 2 & 1 \\ 2 & 6 & 3 & 1 & 4 \end{array} \right) \begin{array}{c} 57 \\ 69 \\ 40 \\ 39 \\ 57 \end{array}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t
---	---	---	---	---



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 5 & 5 & 2 & 4 & 4 \\ 6 & 4 & 5 & 1 & 5 \\ 1 & 2 & 3 & 2 & 1 \\ 5 & 7 & 6 & 2 & 3 \\ 4 & 1 & 1 & 4 & 2 \end{pmatrix} \begin{pmatrix} 61 \\ 66 \\ 36 \\ 78 \\ 42 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 5 & 5 & 7 & 3 & 3 & 63 \\ 6 & 4 & 4 & 4 & 1 & 51 \\ 3 & 7 & 1 & 6 & 1 & 66 \\ 6 & 4 & 4 & 4 & 2 & 56 \\ 7 & 4 & 3 & 4 & 1 & 47 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -\mathbf{9} & -\mathbf{11} \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 7 & 6 & 3 & 6 & 2 & 85 \\ 2 & 2 & 7 & 4 & 2 & 37 \\ 7 & 3 & 7 & 5 & 4 & 72 \\ 2 & 7 & 1 & 7 & 6 & 56 \\ 6 & 2 & 2 & 5 & 2 & 59 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 7 & 6 & 3 & 2 & 5 & | & 125 \\ 2 & 6 & 7 & 3 & 5 & | & 134 \\ 2 & 3 & 3 & 7 & 2 & | & 85 \\ 3 & 5 & 2 & 2 & 4 & | & 93 \\ 1 & 5 & 2 & 4 & 3 & | & 86 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembrando que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 5 & 7 & 3 & 4 & 4 & | & 88 \\ 3 & 4 & 4 & 5 & 5 & | & 96 \\ 4 & 5 & 3 & 2 & 4 & | & 71 \\ 1 & 1 & 3 & 3 & 6 & | & 82 \\ 4 & 7 & 6 & 5 & 5 & | & 106 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A .

A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 1 & 6 & 4 & 6 & 1 & 34 \\ 2 & 5 & 2 & 4 & 3 & 38 \\ 1 & 7 & 3 & 2 & 4 & 30 \\ 5 & 4 & 3 & 1 & 2 & 61 \\ 7 & 7 & 1 & 2 & 4 & 76 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 1 & 1 & 5 & 4 & 2 & 44 \\ 5 & 7 & 1 & 3 & 2 & 32 \\ 3 & 7 & 4 & 4 & 4 & 56 \\ 3 & 2 & 1 & 2 & 1 & 18 \\ 3 & 5 & 2 & 1 & 6 & 42 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\begin{pmatrix} 7 & 1 & 6 & 2 & 3 \\ 5 & 4 & 5 & 3 & 5 \\ 1 & 1 & 3 & 2 & 6 \\ 6 & 1 & 1 & 1 & 3 \\ 6 & 2 & 7 & 6 & 5 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ w \\ t \end{pmatrix} = \begin{pmatrix} 79 \\ 91 \\ 37 \\ 41 \\ 120 \end{pmatrix}$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots, t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 6 & 4 & 3 & 1 & 4 & 24 \\ 7 & 3 & 5 & 7 & 5 & 82 \\ 2 & 1 & 3 & 4 & 1 & 46 \\ 7 & 5 & 6 & 1 & 7 & 37 \\ 2 & 5 & 2 & 2 & 6 & 38 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 6 & 5 & 2 & 7 & 1 & 60 \\ 4 & 7 & 3 & 4 & 1 & 51 \\ 5 & 3 & 1 & 1 & 2 & 48 \\ 6 & 5 & 2 & 1 & 5 & 84 \\ 2 & 3 & 2 & 6 & 3 & 54 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

calculo de L,U conforme [Fra07=Calculo Numerico,
 # Neide B Franco] pag. 126

```
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...], [...], ...], float)$ e $a2 = np.array([[...], [...], ...], float)$ e depois $print(np.dot(a1, a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{cccc|c} 4 & 2 & 7 & 4 & 2 & 47 \\ 7 & 6 & 6 & 3 & 2 & 41 \\ 3 & 7 & 2 & 3 & 7 & 41 \\ 3 & 4 & 2 & 5 & 6 & 39 \\ 2 & 3 & 4 & 1 & 1 & 26 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t



Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial LU onde a matriz U (de Upper) é a própria matriz A devidamente escalonada (como visto no método de Gauss) e a matriz L (de Lower) é uma matriz que contém unidades na diagonal principal, zeros acima destas posições e que contém os pivots do método de Gauss abaixo da diagonal principal.

Em termos esquemáticos, se começa com o sistema $[A] \cdot \{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L] \cdot [U] = [A]$. Agora, fica $[L] \cdot [U] \cdot \{x\} = \{b\}$. Separando esta expressão, se faz $[U] \cdot \{x\} = \{y\}$ e finalmente $[L] \cdot \{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L] \cdot \{y\} = \{b\}$ e depois se calcula x em $[U] \cdot \{x\} = \{y\}$.

A principal vantagem deste método é separar o trabalho em 2 etapas: primeiro o cálculo de L e U e depois y e x . Para sistemas onde há que se calcular muitos x' em função de muitos b' , mantendo-se constante A - o que é a imensa maioria dos casos, como se verá no exercício da tomografia computadorizada, este método adianta o "serviço" diminuindo o trabalho lá na frente.

Tudo começa olhando-se o seguinte esquema:

$$[L] \cdot [U] = [A]$$

ou

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$$

Daqui, e usando-se a álgebra matricial básica sai que $u_{1j} = a_{1j}$ ou seja, que a primeira linha de U é igual à primeira linha de A . A primeira coluna de L é $\ell_{i1} = \frac{a_{i1}}{u_{11}}$ para $(i = 2, 3, \dots)$

A segunda linha de U é $u_{2j} = a_{2j} - \ell_{21} \times u_{1j}$

A segunda linha de L é $\frac{a_{i2} - \ell_{i1} \times u_{12}}{u_{22}}$ e assim por diante.

Chega-se à seguinte formulação:

L e U podem ser calculado assim

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} \times u_{kj}, \quad i \leq j$$

$$\ell_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik} \times u_{kj} \right) / u_{jj}, \quad i > j$$

Seja por exemplo, resolver o sistema

$$\begin{cases} x + 4y + 4z = 23 \\ 3x + 3y + z = 27 \\ 2x + 5y + 2z = 25 \end{cases}$$

Aqui, as matrizes L e U são calculadas desta maneira:

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} ? & ? & ? \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ ? & 1 & 0 \\ ? & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ \mathbf{3} & 1 & 0 \\ \mathbf{2} & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & ? & ? \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & ? & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & \mathbf{0.33} & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & ? \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 2 & 0.33 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 4 & 4 \\ 0 & -9 & -11 \\ 0 & 0 & -\mathbf{2.33} \end{pmatrix} = \begin{pmatrix} 1 & 4 & 4 \\ 3 & 3 & 1 \\ 2 & 5 & 2 \end{pmatrix}$$

```
# calculo de L,U conforme [Fra07=Calculo Numerico,
#                               Neide B Franco] pag. 126
import numpy as np
def lu(a):
    t=len(a[0])
    u=np.zeros((t,t),float)
    l=np.zeros((t,t),float)
    for i in range(0,t):
        l[i,i]=1
    for i in range(0,t):
        for j in range(0,t):
            if i<=j:
                s=0
                for k in range(0,i):
                    s=s+l[i,k]*u[k,j]
                u[i,j]=a[i,j]-s
            if i>j:
                s=0
                for k in range(0,j):
                    s=s+l[i,k]*u[k,j]
                l[i,j]=(a[i,j]-s)/u[j,j]
    return(l,u)
```

```
l,u=lu(np.array([[1,4,4],[3,3,1],[2,5,2]]))
print(l)
print(u)
```

O cálculo de L e U podem ser feito usando-se os pivots e os multiplicadores do método de Gauss, como se pode ver em

```
import numpy as np
def lua1(a,b):
    print('-----a-----')
    print(a)
    n=len(a)
    x=[0]*n
    y=[0]*n
    L=np.zeros((n,n),float)
    for passo in range(0,n):
        for i in range(passo+1,n):
            pivot=a[i,passo]/a[passo,passo]
            for j in range(0,n):
                a[i,j]=a[i,j]-pivot*a[passo,j]
            L[i,passo]=pivot
    for i in range(0,n):
        L[i,i]=1
    y[0]=b[0]
    for i in range(1,n):
        y[i]=b[i]
        for j in range(0,i):
            y[i]=y[i]-L[i,j]*y[j]
    x[n-1]=y[n-1]/a[n-1,n-1]
    for i in range(n-1,-1,-1):
        x[i]=y[i]
        for j in range(i+1,n):
            x[i]=x[i]-a[i,j]*x[j]
        x[i]=x[i]/a[i,i]
    print('-----L-----')
    print(L)
    print('-----y-----')
    print(y)
    print('-----u-----')
    print(a)
    print('-----x-----')
    print(x)
import numpy as np
a=np.array([[1,4,4],[3,3,1],[2,5,2]],float)
b=np.array([23,27,25],float)
lua1(a,b)
```

Como é de se esperar, ambos os métodos devem dar o mesmo resultado. Lembre-se que no pacote NUMPY está lá a multiplicação matricial. Basta fazer $a1 = np.array([[...],[...],[...]],float)$ e $a2 = np.array([[...],[...],[...]],float)$ e depois $print(np.dot(a1,a2))$

🔧 Para você fazer

Resolva pelo método LU o sistema

$$\left(\begin{array}{ccccc|c} 2 & 7 & 3 & 5 & 3 & 82 \\ 3 & 2 & 5 & 1 & 6 & 96 \\ 5 & 2 & 3 & 7 & 1 & 71 \\ 3 & 7 & 1 & 1 & 6 & 86 \\ 4 & 4 & 3 & 6 & 4 & 95 \end{array} \right)$$

Indique em lugar próprio (no verso), as matrizes L e U correspondentes à solução. E responda aqui os valores INTEIROS de $x, y, \dots t$.

x	y	z	w	t
---	---	---	---	---

