UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70210 ALDO HENRIQUE XAVIER 19HEL110 - 1 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 6 & 1 & 5 & 3 & 4 & 69 \\ 2 & 3 & 7 & 4 & 5 & 63 \\ 7 & 7 & 4 & 6 & 2 & 110 \\ 4 & 3 & 6 & 4 & 6 & 74 \\ 7 & 1 & 4 & 4 & 1 & 75 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70227 ALVARO JOSE BARANOSKI 19HEL110 - 2 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 2 & 7 & 7 & 3 & 94 \\ 4 & 2 & 2 & 5 & 2 & 61 \\ 7 & 5 & 7 & 7 & 2 & 127 \\ 4 & 4 & 6 & 1 & 1 & 89 \\ 7 & 6 & 6 & 1 & 6 & 149 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70234 ANAJULIA DE COUTO FLORIANO 19HEL110 - 3 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 1 & 7 & 3 & 4 & 118 \\ 7 & 1 & 6 & 5 & 5 & 134 \\ 2 & 1 & 2 & 2 & 1 & 44 \\ 1 & 3 & 2 & 4 & 4 & 77 \\ 2 & 6 & 7 & 5 & 7 & 160 \end{array} \right)$$

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

x	y	z	w	t

110-70234 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70827 ANDRE LUIS O. ANDRADE 19HEL110 - 4 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 5 & 4 & 2 & 1 & 34 \\ 7 & 1 & 7 & 3 & 3 & 43 \\ 7 & 6 & 7 & 5 & 3 & 51 \\ 3 & 1 & 1 & 4 & 3 & 14 \\ 6 & 2 & 4 & 7 & 7 & 40 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70834 ANDRE LUIZ
DOMINGOS 19HEL110 - 5 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 3 & 7 & 1 & 5 & 28 \\ 3 & 5 & 7 & 1 & 1 & 34 \\ 6 & 7 & 3 & 7 & 5 & 63 \\ 3 & 1 & 2 & 1 & 7 & 21 \\ 2 & 6 & 1 & 4 & 6 & 53 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70241 BIANCA LIMA DE OLIVEIRA 19HEL110 - 6 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 3 & 2 & 4 & 1 & 85 \\ 1 & 2 & 3 & 1 & 3 & 40 \\ 2 & 3 & 7 & 1 & 2 & 75 \\ 6 & 4 & 3 & 7 & 6 & 93 \\ 3 & 7 & 5 & 7 & 4 & 86 \end{array}\right)$$

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

x	y	z	w	t

110-70241 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70258 BRUNO
MACHADO CARNEIRO 19HEL110 - 7 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 2 & 7 & 5 & 4 & 6 & 108 \\ 1 & 3 & 7 & 7 & 2 & 86 \\ 1 & 3 & 6 & 4 & 5 & 74 \\ 5 & 6 & 4 & 5 & 1 & 88 \\ 5 & 6 & 7 & 1 & 7 & 89 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70265 CAIO HEITOR BARZ MOLINARI 19HEL110 - 8 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 3 & 5 & 1 & 7 & 2 & 67 \\ 4 & 6 & 6 & 5 & 7 & 97 \\ 3 & 1 & 1 & 5 & 3 & 55 \\ 6 & 3 & 7 & 1 & 4 & 60 \\ 7 & 5 & 3 & 4 & 4 & 78 \end{array} \right)$$

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

x	y	z	w	t

110-70265 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71156 CAIO
MAGANHINI LOPES 19HEL110 - 9 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 3 & 3 & 5 & 7 & 4 & 116 \\ 2 & 5 & 7 & 6 & 7 & 142 \\ 2 & 7 & 7 & 2 & 4 & 93 \\ 1 & 7 & 2 & 3 & 6 & 92 \\ 5 & 4 & 6 & 7 & 6 & 149 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70272 DANIEL SANTOS DA SILVA 19HEL110 - 10 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 3 & 1 & 6 & 4 & 82 \\ 7 & 5 & 6 & 6 & 2 & 129 \\ 4 & 4 & 3 & 4 & 6 & 107 \\ 3 & 5 & 6 & 6 & 7 & 151 \\ 2 & 6 & 4 & 5 & 2 & 102 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70289 ELEN DE JESUS SANTANA 19HEL110 - 11 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 1 & 5 & 6 & 1 & 3 & 38 \\ 5 & 3 & 5 & 3 & 6 & 52 \\ 7 & 6 & 1 & 2 & 7 & 64 \\ 2 & 7 & 6 & 6 & 1 & 84 \\ 1 & 1 & 4 & 5 & 7 & 36 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70841 FELIPE NAVES MENDONCA 19HEL110 - 12 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 3 & 1 & 4 & 5 & 3 & 48 \\ 6 & 6 & 2 & 4 & 7 & 87 \\ 5 & 1 & 4 & 7 & 1 & 52 \\ 1 & 7 & 5 & 3 & 4 & 62 \\ 6 & 5 & 3 & 2 & 7 & 70 \end{array} \right)$$

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

x	y	z	w	t

110-70841 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71251 GIOVANA ASSIS
JIENTARA 19HEL110 - 13 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 2 & 3 & 6 & 6 & 6 & 94 \\ 7 & 3 & 6 & 5 & 4 & 112 \\ 4 & 1 & 6 & 1 & 1 & 52 \\ 5 & 4 & 4 & 4 & 3 & 94 \\ 1 & 1 & 2 & 5 & 3 & 59 \end{array}\right)$$

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

x	y	z	w	t

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 6 & 3 & 2 & 6 & 2 & 43 \\ 3 & 5 & 2 & 2 & 7 & 73 \\ 4 & 5 & 3 & 7 & 2 & 38 \\ 1 & 6 & 2 & 1 & 5 & 47 \\ 6 & 1 & 1 & 3 & 2 & 40 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70308 GUILHERME FONTES DUARTE 19HEL110 - 15 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 2 & 1 & 7 & 7 & 128 \\ 3 & 2 & 7 & 4 & 6 & 115 \\ 6 & 3 & 4 & 6 & 3 & 105 \\ 1 & 2 & 7 & 6 & 2 & 83 \\ 1 & 4 & 5 & 6 & 7 & 126 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70315 HENRIQUE DOMICIANO OSINSKI 19HEL110 - 16 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 4 & 3 & 2 & 6 & 74 \\ 7 & 6 & 4 & 2 & 6 & 83 \\ 6 & 7 & 2 & 7 & 7 & 133 \\ 1 & 7 & 3 & 6 & 7 & 132 \\ 3 & 3 & 3 & 1 & 2 & 39 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70322 HENRIQUE MARTINS DOMANSKI 19HEL110 - 17 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 1 & 2 & 7 & 2 & 1 & 18 \\ 3 & 7 & 5 & 1 & 4 & 62 \\ 3 & 1 & 6 & 4 & 3 & 46 \\ 4 & 3 & 4 & 3 & 4 & 61 \\ 7 & 1 & 1 & 2 & 6 & 78 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70339 HERICK DENKE
19HEL110 - 18 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 5 & 1 & 7 & 2 & 48 \\ 2 & 4 & 1 & 2 & 2 & 35 \\ 7 & 3 & 3 & 7 & 1 & 31 \\ 4 & 4 & 6 & 5 & 4 & 56 \\ 4 & 4 & 7 & 5 & 2 & 45 \end{array} \right)$$

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

x	y	z	w	t

110-70339 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70346 IURI PAVANI DE SOUZA 19HEL110 - 19 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 6 & 3 & 7 & 4 & 100 \\ 3 & 3 & 4 & 1 & 6 & 39 \\ 6 & 1 & 6 & 4 & 2 & 70 \\ 3 & 1 & 3 & 5 & 7 & 37 \\ 7 & 6 & 6 & 6 & 1 & 104 \end{array}\right)$$

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

x	y	z	w	t

110-70346 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70858 JHONATAN K. RIBEIRO 19HEL110 - 20 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 5 & 2 & 5 & 1 & 88 \\ 5 & 5 & 1 & 2 & 2 & 66 \\ 1 & 1 & 6 & 7 & 7 & 118 \\ 1 & 3 & 4 & 7 & 3 & 102 \\ 2 & 7 & 2 & 6 & 5 & 125 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70353 JODERSON GOMES DA SILVA 19HEL110 - 21 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 4 & 7 & 3 & 7 & 83 \\ 6 & 7 & 5 & 2 & 2 & 105 \\ 4 & 5 & 5 & 3 & 5 & 86 \\ 7 & 6 & 1 & 3 & 7 & 83 \\ 7 & 6 & 2 & 5 & 3 & 103 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70360 LEONARDO CORNELIO KUIPERS 19HEL110 - 22 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 4 & 1 & 4 & 6 & 76 \\ 6 & 1 & 5 & 4 & 3 & 60 \\ 4 & 4 & 6 & 6 & 3 & 93 \\ 4 & 5 & 3 & 2 & 2 & 75 \\ 1 & 3 & 5 & 1 & 1 & 46 \end{array}\right)$$

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

x	y	z	w	t

110-70360 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70377 LUCAS ANTONIO THEODORO SANTOS 19HEL110 - 23 apos 26/09, 50%
/ _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 2 & 4 & 2 & 5 & 4 & 67 \\ 7 & 5 & 5 & 6 & 3 & 102 \\ 6 & 7 & 1 & 1 & 7 & 50 \\ 5 & 7 & 1 & 5 & 5 & 85 \\ 4 & 5 & 2 & 4 & 1 & 71 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70384 LUCAS BOHN
DUTRA 19HEL110 - 24 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 2 & 2 & 7 & 3 & 55 \\ 3 & 4 & 3 & 6 & 3 & 44 \\ 3 & 5 & 3 & 3 & 2 & 27 \\ 7 & 3 & 6 & 1 & 1 & 36 \\ 2 & 7 & 3 & 7 & 2 & 42 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70391 LUCAS CORDEIRO NEVES 19HEL110 - 25 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 2 & 7 & 2 & 6 & 127 \\ 5 & 1 & 6 & 5 & 7 & 125 \\ 1 & 6 & 7 & 5 & 5 & 106 \\ 5 & 4 & 3 & 1 & 3 & 78 \\ 3 & 5 & 6 & 4 & 2 & 80 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70403 LUCAS
FAGANELLO SILVA 19HEL110 - 26 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 6 & 6 & 7 & 7 & 7 & 109 \\ 4 & 7 & 4 & 6 & 4 & 99 \\ 1 & 6 & 6 & 7 & 5 & 93 \\ 7 & 5 & 5 & 1 & 3 & 67 \\ 7 & 1 & 3 & 2 & 5 & 47 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70410 LUIZ AUGUSTO SHOITI TSURUKAWA 19HEL110 - 27 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 2 & 3 & 5 & 2 & 6 & 70 \\ 5 & 3 & 1 & 6 & 5 & 73 \\ 1 & 5 & 3 & 2 & 3 & 55 \\ 1 & 6 & 5 & 4 & 6 & 83 \\ 7 & 2 & 7 & 3 & 7 & 119 \end{array} \right)$$

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

x	y	z	w	t

110-70410 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70427 MARCOS A. DA COSTA 19HEL110 - 28 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 3 & 6 & 2 & 6 & 6 & 75 \\ 4 & 2 & 6 & 3 & 6 & 80 \\ 6 & 3 & 7 & 5 & 7 & 108 \\ 6 & 7 & 1 & 2 & 2 & 67 \\ 1 & 5 & 6 & 4 & 6 & 78 \end{array} \right)$$

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

x	y	z	w	t

110-70427 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70434 MARIANA YAEDA ARTUSO 19HEL110 - 29 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 1 & 6 & 2 & 6 & 85 \\ 1 & 6 & 2 & 5 & 2 & 72 \\ 6 & 6 & 6 & 4 & 5 & 127 \\ 1 & 7 & 3 & 5 & 2 & 87 \\ 4 & 7 & 1 & 5 & 3 & 83 \end{array}\right)$$

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

x	y	z	w	t

110-70434 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71220 MATHEUS CARDOSO CASTILHO DIAS 19HEL110 - 30 apos 26/09, 50% _____ / / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 3 & 1 & 4 & 3 & 6 & 57 \\ 6 & 7 & 4 & 7 & 1 & 123 \\ 5 & 5 & 6 & 1 & 3 & 98 \\ 5 & 5 & 1 & 2 & 7 & 89 \\ 3 & 4 & 3 & 7 & 3 & 77 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70441 MATHEUS DA SILVA QUADROS 19HEL110 - 31 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 2 & 1 & 4 & 6 & 5 & 42 \\ 6 & 4 & 2 & 4 & 2 & 74 \\ 6 & 6 & 4 & 3 & 1 & 104 \\ 5 & 7 & 4 & 6 & 3 & 110 \\ 5 & 7 & 7 & 2 & 3 & 124 \end{array} \right)$$

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

x	y	z	w	t

110-70441 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70458 MICCAELA
BOGASZ SANTIAGO 19HEL110 - 32 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 7 & 6 & 4 & 1 & 43 \\ 5 & 1 & 5 & 2 & 1 & 20 \\ 4 & 6 & 1 & 4 & 5 & 41 \\ 6 & 6 & 5 & 1 & 7 & 60 \\ 3 & 2 & 5 & 3 & 2 & 24 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71194 NATALIA GONCALVES P CHOMA 19HEL110 - 33 apos 26/09, 50% _____ / / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 1 & 2 & 4 & 3 & 1 & 22 \\ 1 & 1 & 3 & 6 & 6 & 31 \\ 6 & 6 & 1 & 1 & 4 & 33 \\ 7 & 2 & 2 & 7 & 1 & 54 \\ 5 & 1 & 3 & 3 & 3 & 42 \end{array} \right)$$

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

x	y	z	w	t

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 3 & 1 & 6 & 3 & 60 \\ 1 & 7 & 6 & 4 & 2 & 33 \\ 3 & 6 & 1 & 1 & 4 & 23 \\ 5 & 5 & 2 & 5 & 7 & 65 \\ 7 & 5 & 2 & 7 & 5 & 73 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71163 PEDRO HENRIQUE L ARAUJO 19HEL110 - 35 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 1 & 4 & 5 & 2 & 37 \\ 5 & 5 & 4 & 4 & 3 & 73 \\ 2 & 2 & 7 & 4 & 1 & 30 \\ 6 & 5 & 5 & 5 & 3 & 76 \\ 6 & 3 & 1 & 6 & 3 & 65 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70472 RODOLPHO DE SOUZA DAVID 19HEL110 - 36 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 4 & 1 & 1 & 6 & 1 & 85 \\ 3 & 4 & 4 & 6 & 1 & 104 \\ 4 & 4 & 3 & 5 & 1 & 94 \\ 5 & 7 & 2 & 7 & 7 & 135 \\ 5 & 7 & 6 & 4 & 1 & 114 \end{array}\right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70489 TAYANE
CORDEIRO FARIA 19HEL110 - 37 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 5 & 4 & 4 & 6 & 129 \\ 5 & 2 & 3 & 1 & 2 & 71 \\ 7 & 4 & 6 & 1 & 7 & 150 \\ 3 & 7 & 6 & 4 & 1 & 90 \\ 4 & 3 & 5 & 3 & 2 & 80 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70496 THIAGO ENRIQUE B. FIORESE 19HEL110 - 38 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 3 & 2 & 5 & 2 & 57 \\ 4 & 3 & 2 & 4 & 2 & 54 \\ 5 & 5 & 7 & 6 & 6 & 135 \\ 1 & 5 & 7 & 1 & 5 & 116 \\ 7 & 1 & 2 & 7 & 2 & 55 \end{array} \right)$$

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

x	y	z	w	t

110-70496 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70515 VALTER
WERGENSKI 19HEL110 - 39 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 7 & 2 & 7 & 5 & 108 \\ 7 & 1 & 5 & 2 & 2 & 65 \\ 1 & 4 & 2 & 5 & 5 & 56 \\ 1 & 3 & 7 & 1 & 7 & 45 \\ 2 & 2 & 2 & 1 & 7 & 44 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70522 VICTOR EMANUEL VENTURI 19HEL110 - 40 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 7 & 6 & 1 & 6 & 1 & 69 \\ 4 & 4 & 5 & 6 & 3 & 94 \\ 5 & 1 & 3 & 5 & 5 & 60 \\ 1 & 4 & 1 & 1 & 4 & 24 \\ 7 & 7 & 7 & 3 & 2 & 107 \end{array} \right)$$

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

x	y	z	w	t

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70539 VINICIUS
GUILHERME F. SILVA 19HEL110 - 41 apos 26/09, 50% _____ / _____ /

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 3 & 4 & 6 & 3 & 3 & 90 \\ 7 & 1 & 5 & 6 & 7 & 78 \\ 6 & 6 & 5 & 4 & 1 & 104 \\ 3 & 3 & 3 & 2 & 6 & 66 \\ 3 & 4 & 4 & 4 & 7 & 89 \end{array} \right)$$

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

x	y	z	w	t

110-70539 - 26/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:20.0
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71244 VINICIUS LUIS ALFIERI 19HEL110 - 42 apos 26/09, 50% _____ / _____ / _____

Método LU

No método LU, deve-se transformar a matriz A (dos coeficientes) em um produto matricial $L.U$ 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 $[A].\{x\} = \{b\}$. A primeira coisa a fazer é calcular L e U de modo que $[L].[U] = [A]$. Agora, fica $[L].[U].\{x\} = \{b\}$. Separando esta expressão, se faz $[U].\{x\} = \{y\}$ e finalmente $[L].\{y\} = \{b\}$. Ou seja, primeiro se calcula y em $[L].\{y\} = \{b\}$ e depois se calcula x em $[U].\{x\} = \{y\}$.

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:

$$[L].[U] = [A]$$

ou

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

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 $(i = 2, 3...)$
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:

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

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

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

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

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

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

```
# 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

$$\left(\begin{array}{ccccc|c} 5 & 1 & 4 & 7 & 6 & 75 \\ 5 & 1 & 2 & 7 & 2 & 53 \\ 1 & 4 & 1 & 4 & 2 & 59 \\ 2 & 6 & 3 & 2 & 1 & 69 \\ 7 & 6 & 1 & 4 & 3 & 107 \end{array} \right)$$

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

x	y	z	w	t