UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70865 ALEXANDRE S A DELDUQUE MACEDO 19HEA110 - 1 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 1 & 3 & 6 & 7 & 7 & 115 \\ 2 & 3 & 4 & 7 & 2 & 94 \\ 7 & 1 & 2 & 3 & 4 & 76 \\ 1 & 6 & 6 & 3 & 3 & 64 \\ 6 & 4 & 3 & 4 & 1 & 74 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70872 ARIANE APARECIDA CRUZ SILVA 19HEA110 - 2 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 7 & 1 & 5 & 1 & 1 & 70 \\ 6 & 7 & 3 & 2 & 6 & 64 \\ 3 & 3 & 7 & 1 & 5 & 66 \\ 3 & 2 & 3 & 7 & 6 & 56 \\ 7 & 4 & 3 & 5 & 3 & 67 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71170 ERICK BUENO C DOS SANTOS 19HEA110 - 3 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 2 & 2 & 1 & 2 & 1 & 19 \\ 1 & 3 & 2 & 5 & 2 & 30 \\ 7 & 7 & 7 & 3 & 3 & 101 \\ 5 & 7 & 1 & 3 & 7 & 53 \\ 4 & 7 & 6 & 1 & 2 & 89 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70889 FABIANA ANDREA SILVA MORALES 19HEA110 - 4 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 5 & 6 & 3 & 2 & 90 \\ 6 & 5 & 3 & 7 & 5 & 115 \\ 4 & 3 & 2 & 1 & 3 & 68 \\ 6 & 5 & 2 & 4 & 7 & 125 \\ 7 & 6 & 1 & 3 & 7 & 128 \end{array}\right)$$

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

110-70889 - 25/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70896 FELIPE BORTOLLETTO CIVITATE 19HEA110 - 5 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 7 & 1 & 2 & 7 & 30 \\ 1 & 6 & 2 & 5 & 2 & 40 \\ 1 & 2 & 1 & 4 & 5 & 26 \\ 6 & 6 & 5 & 3 & 2 & 47 \\ 4 & 7 & 2 & 6 & 3 & 43 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71187 GABRIEL CHIMALESKI COSTA 19HEA110 - 6 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 6 & 4 & 4 & 6 & 5 & 107 \\ 7 & 2 & 4 & 2 & 7 & 112 \\ 7 & 3 & 1 & 2 & 6 & 107 \\ 6 & 3 & 2 & 1 & 5 & 92 \\ 6 & 2 & 1 & 7 & 3 & 90 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70908 GABRIELA MONTEIRO XAVIER 19HEA110 - 7 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 1 & 1 & 5 & 6 & 7 & 40 \\ 1 & 1 & 6 & 6 & 2 & 49 \\ 2 & 1 & 5 & 7 & 6 & 52 \\ 7 & 5 & 3 & 3 & 2 & 109 \\ 5 & 5 & 1 & 1 & 7 & 74 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70915 GRENDA IZABELI MENEZES SILVA 19HEA110 - 8 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 4 & 3 & 6 & 6 & 5 & 106 \\ 7 & 6 & 3 & 6 & 3 & 91 \\ 5 & 6 & 5 & 7 & 6 & 120 \\ 5 & 4 & 5 & 2 & 6 & 92 \\ 5 & 7 & 7 & 2 & 1 & 48 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70922 HALLYPHER GUERREIRO ZOLLNER 19HEA110 - 9 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 5 & 5 & 1 & 1 & 5 & 35 \\ 2 & 7 & 1 & 6 & 3 & 33 \\ 4 & 1 & 7 & 2 & 3 & 41 \\ 4 & 3 & 3 & 3 & 1 & 22 \\ 1 & 7 & 4 & 3 & 4 & 35 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70939 HIAN DA SILVA PINTO 19HEA110 - 10 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 3 & 5 & 1 & 7 & 43 \\ 6 & 1 & 6 & 7 & 2 & 87 \\ 6 & 2 & 3 & 6 & 7 & 69 \\ 3 & 3 & 3 & 4 & 2 & 58 \\ 7 & 6 & 4 & 7 & 5 & 104 \end{array}\right)$$

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

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70946 IGOR HERNAN ANDRADE SANTOS 19HEA110 - 11 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 4 & 4 & 5 & 2 & 7 & 107 \\ 7 & 4 & 5 & 4 & 3 & 88 \\ 5 & 5 & 3 & 7 & 3 & 101 \\ 2 & 5 & 4 & 2 & 3 & 89 \\ 7 & 6 & 6 & 2 & 5 & 115 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70953 ISADORA FONSECA CARRARI 19HEA110 - 12 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 7 & 2 & 1 & 6 & 4 & 98 \\ 7 & 1 & 6 & 7 & 5 & 138 \\ 2 & 3 & 4 & 7 & 3 & 104 \\ 4 & 4 & 2 & 3 & 6 & 71 \\ 1 & 2 & 5 & 1 & 4 & 53 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70960 JESSICA FURTUOSO HARMATIUK 19HEA110 - 13 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 5 & 7 & 6 & 5 & 7 & 121 \\ 4 & 2 & 1 & 2 & 5 & 43 \\ 3 & 1 & 1 & 3 & 5 & 41 \\ 2 & 1 & 3 & 3 & 6 & 49 \\ 6 & 6 & 1 & 6 & 1 & 89 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70977 JOAO GUILHERME LIMA MARIANO 19HEA110 - 14 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 2 & 7 & 3 & 1 & 66 \\ 6 & 7 & 6 & 3 & 7 & 96 \\ 3 & 3 & 5 & 1 & 4 & 45 \\ 6 & 6 & 5 & 3 & 2 & 95 \\ 4 & 3 & 7 & 3 & 7 & 70 \end{array} \right)$$

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

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70984 KAREN LETICIA SARTORI 19HEA110 - 15 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 2 & 6 & 7 & 4 & 4 & 89 \\ 5 & 7 & 6 & 5 & 1 & 88 \\ 3 & 6 & 5 & 2 & 4 & 70 \\ 3 & 4 & 7 & 7 & 4 & 88 \\ 5 & 6 & 5 & 1 & 1 & 61 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 70991 KARLA KOSOSKI BARTH MACEDO 19HEA110 - 16 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 2 & 2 & 2 & 2 & 4 & 58 \\ 1 & 2 & 2 & 3 & 1 & 48 \\ 7 & 6 & 3 & 5 & 7 & 127 \\ 3 & 1 & 3 & 1 & 3 & 43 \\ 1 & 2 & 3 & 3 & 5 & 75 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71006 LETICIA
ERTHAL CORDEIRO 19HEA110 - 17 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 7 & 1 & 6 & 2 & 3 & 75 \\ 6 & 6 & 7 & 7 & 1 & 96 \\ 7 & 2 & 5 & 4 & 7 & 117 \\ 5 & 5 & 6 & 4 & 1 & 75 \\ 5 & 6 & 5 & 5 & 5 & 113 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71013 LIAH
CHIURATTO BUBA 19HEA110 - 18 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 4 & 1 & 5 & 7 & 6 & 62 \\ 6 & 5 & 2 & 1 & 1 & 57 \\ 7 & 6 & 3 & 2 & 1 & 72 \\ 3 & 2 & 3 & 5 & 6 & 60 \\ 4 & 7 & 5 & 4 & 3 & 92 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71020 LISA ISABELLA
FISCHER 19HEA110 - 19 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 6 & 7 & 7 & 7 & 2 & 89 \\ 7 & 5 & 4 & 6 & 6 & 123 \\ 2 & 2 & 5 & 3 & 5 & 77 \\ 4 & 1 & 1 & 1 & 5 & 74 \\ 3 & 2 & 4 & 7 & 4 & 88 \end{array}\right)$$

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

110-71020 - 25/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71037 LUIS GABRIEL COUTINHO ROCHA F 19HEA110 - 20 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 7 & 7 & 1 & 2 & 4 & 61 \\ 6 & 2 & 7 & 4 & 6 & 47 \\ 5 & 5 & 3 & 4 & 5 & 50 \\ 3 & 7 & 6 & 5 & 6 & 67 \\ 7 & 6 & 7 & 5 & 3 & 69 \end{array}\right)$$

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

110-71037 - 25/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71044 MARIA GABRIELA FERNANDES LOPES 19HEA110 - 21 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 1 & 7 & 7 & 1 & 37 \\ 1 & 7 & 1 & 4 & 6 & 87 \\ 5 & 4 & 3 & 1 & 7 & 79 \\ 6 & 4 & 7 & 2 & 3 & 62 \\ 4 & 4 & 6 & 5 & 3 & 64 \end{array}\right)$$

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

110-71044 - 25/09

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71051 MARIA LUIZA MOCELIM DE MATTOS 19HEA110 - 22 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 5 & 3 & 7 & 6 & 55 \\ 4 & 3 & 7 & 5 & 4 & 73 \\ 2 & 2 & 1 & 6 & 6 & 37 \\ 6 & 2 & 7 & 6 & 7 & 95 \\ 3 & 3 & 7 & 1 & 6 & 79 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71068 MATEUS DOS SANTOS NEVES 19HEA110 - 23 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 1 & 6 & 7 & 3 & 2 & 88 \\ 2 & 3 & 5 & 5 & 3 & 65 \\ 2 & 5 & 6 & 4 & 2 & 83 \\ 2 & 2 & 3 & 7 & 7 & 58 \\ 1 & 4 & 4 & 3 & 5 & 64 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71075 PAOLA
GEBAUER MATEOS 19HEA110 - 24 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 3 & 3 & 5 & 7 & 5 & 63 \\ 6 & 1 & 6 & 4 & 6 & 77 \\ 7 & 4 & 1 & 7 & 1 & 30 \\ 5 & 6 & 3 & 7 & 3 & 54 \\ 7 & 7 & 2 & 2 & 2 & 55 \end{array}\right)$$

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

UFPR - UP - UTFPR - PUCPr – 31/08/2019 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71101 RAUL BECKERT VESSARO 19HEA110 - 25 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 5 & 1 & 2 & 4 & 6 & 68 \\ 3 & 6 & 1 & 5 & 7 & 75 \\ 6 & 5 & 6 & 3 & 3 & 64 \\ 2 & 4 & 5 & 3 & 4 & 42 \\ 1 & 4 & 6 & 2 & 7 & 39 \end{array}\right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71118 VICTORIA FERRO MAESTRELLI 19HEA110 - 26 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 2 & 6 & 5 & 3 & 6 & 72 \\ 4 & 3 & 3 & 3 & 2 & 34 \\ 1 & 1 & 2 & 1 & 2 & 18 \\ 1 & 2 & 4 & 4 & 6 & 51 \\ 6 & 6 & 2 & 2 & 3 & 50 \end{array} \right)$$

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 - 12:12:02.2
Prof Dr P Kantek (pkantek@gmail.com)
Sistemas Lineares: Método LU VIVO735y, V: 1.05 71125 VITOR KURTEN
FEITOSA 19HEA110 - 27 apos 25/09, 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

$$\left(\begin{array}{ccccc|c} 7 & 6 & 2 & 1 & 3 & 68 \\ 6 & 3 & 7 & 5 & 2 & 60 \\ 1 & 5 & 3 & 7 & 3 & 53 \\ 2 & 1 & 2 & 4 & 4 & 35 \\ 3 & 4 & 5 & 5 & 7 & 72 \end{array}\right)$$

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