Sistemas Lineares

J. A. Salvador

Sumário

Introdução

Problema: Achar o ponto de equilíbrio entre a curva de oferta e de demanda

```
> restart; with(plots):
Warning, the name changecoords has been redefined
```

```
> o := y = 10 - 2 * x ;  d := y = 3/2*x + 1;
```

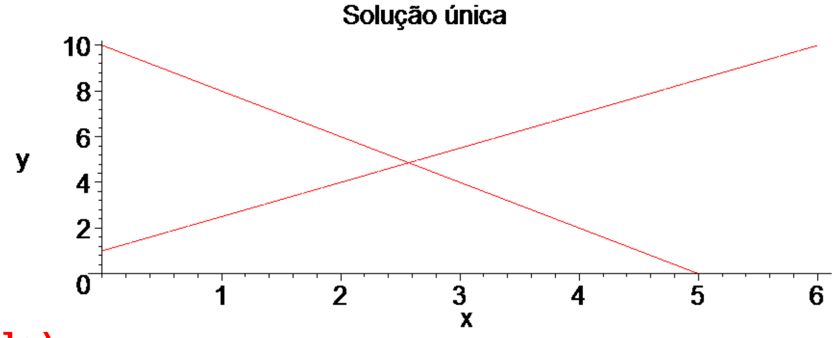
$$o := y = 10 - 2\,x$$

$$d := y = \frac{3\,x}{2} + 1$$

```
> solve( { o, d}, {x,y});
```

$$\{x = \frac{18}{7}, y = \frac{34}{7}\}$$

```
> implicitplot( {o, d}, x=0..10, y=0..10, title = `Solução
  única`);
```




```
> with(linalg):
> A1 := matrix(2,2, [2,1,-3/2,1]);
```

$$A1 := \begin{bmatrix} 2 & 1 \\ \frac{-3}{2} & 1 \end{bmatrix}$$

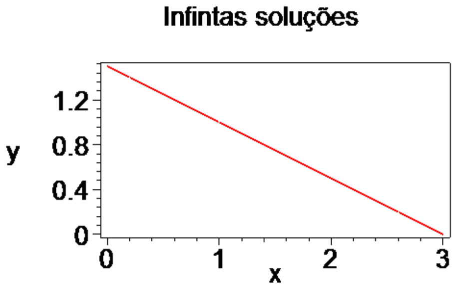
```
> 'det(A1)' = det(A1);
```

$$\det(A1) = \frac{7}{2}$$

```
> restart; with(plots):
```
Warning, the name changecoords has been redefined

```
> r := x + 2*y = 3;  s := 2*x + 4*y = 6;
```
$$r := x + 2\,y = 3$$
$$s := 2\,x + 4\,y = 6$$

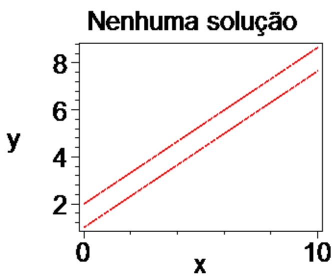
```
> solve( { r, s}, {x,y});
```
$$\{x = -2\,y + 3, y = y\}$$

```
> plots[implicitplot]( {r, s}, x=0..10, y=0..10, title=`Infintas
  soluções`);
```



```
> A2 := matrix(2,2, [1, 2, 2, 4]);
```
$$A2 := \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

```
> 'det(A2)' = det(A2);
```
$$\det(A2) = 0$$

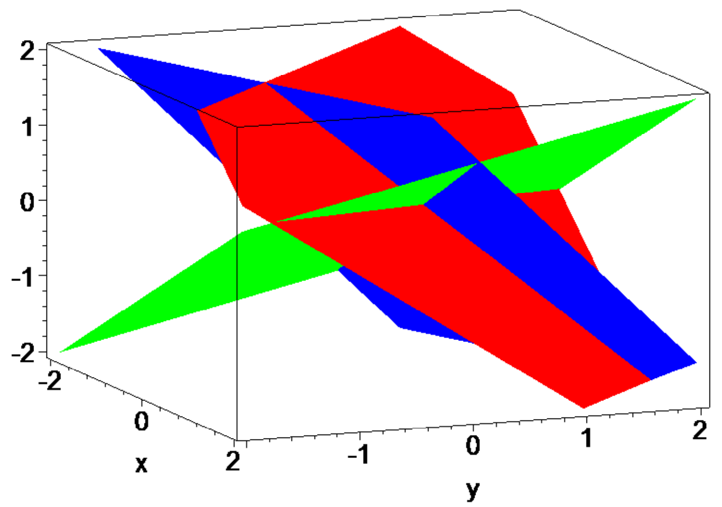
```
> restart; with(plots):
> e1 := 2*x - 3*y = -3;  e2 := 4*x - 6*y = -12;
```
$$e1 := 2\,x - 3\,y = -3$$
$$e2 := 4\,x - 6\,y = -12$$

```
> solve( { e1, e2}, {x,y});
> implicitplot( {e1, e2}, x=0..10, y=0..10, title= `Nenhuma
  solução`);
```



```
> A3 := matrix(2,2, [2, -3, 4, -6]);
> 'det(A3)' = det(A3);
```
$$A3 := \begin{bmatrix} 2 & -3 \\ 4 & -6 \end{bmatrix}$$
$$\det(A3) = 0$$

```
> with(plots):
> implicitplot3d(  x + y + z = 1,  x=-2..2, y=-2..2, z=-2..2,
  color = red): g1 := %:
> implicitplot3d( x- 3*y - 2*z = -1,  x=-2..2, y=-2..2, z=-2..2,
```

```
  color = blue): g2 := %:
> implicitplot3d( 2*x + y -3*z = 0,  x=-2..2, y=-2..2, z=-2..2,
  color = green): g3 := %:
> display(g1,g2,g3);
```




```
> A4 := matrix(3,3, [1, 1, 1, 1, -3, -2, 2, 1, -3]);
> 'det(A4)' = det(A4);
```

$$A4 := \begin{bmatrix} 1 & 1 & 1 \\ 1 & -3 & -2 \\ 2 & 1 & -3 \end{bmatrix}$$

$$\det(A4) = 17$$

\>

Em geral, um sistema linear de n equações a n incógnitas x_j é escrito na forma:

$$a_{11}\, x_1 + a_{12}\, x_2 \;+\; \dots \;+ a_{1\,n}\, x_n = b_1$$
$$a_{21}\, x_1 + a_{22}\, x_2 \;+\; \dots \;+ a_{2\,n}\, x_n = b_2$$
$$\dots$$
$$a_{m1}\, x_1 + a_{m2}\, x_2 \;+\; \dots \;+ a_{mn}\, x_n = b_m\,,$$

em que a_{ij} são os coeficientes, $1 \le i$ e $i \le m$ e $1 \le j$ e $j \le n$, x_j são as variáveis , $j = 1 .. n$ e b_i são constantes, $i = 1 .. m$.

Sistemas Triangulares

Um sistema linear de n equações a n incógnitas x_j é chamado *triangular superior* se os coeficientes $a_{ij} = 0$ sempre que
$\boldsymbol{i < j}$. Um exemplo típico é:

$$a_{11}\, x_1 + a_{12}\, x_2 \;+\; \dots \;+ a_{1\,n}\, x_n = b_1$$
$$a_{22}\, x_2 \;+\; \dots \;+ a_{2\,n}\, x_n = b_2$$
$$.......$$
$$a_{nn}\, x_n = b_n\,.$$

O algoritmo clássico para se resolver este tipo de sistema é o da substituição inversa. Determina-se x_n a partir da última equação. Uma vez calculado x_n, substitui-se esse valor na penúltima equação para obter x_{n-1}. Continuando com essas substituições obteremos todos os x_j. Observe que as divisões por a_{jj} são permitidas porque se o sistema for do tipo *possível e determinado* o determinante da matriz dos coeficientes será diferente de zero. No nosso caso o determinante é precisamente o produto dos elementos da diagonal da matriz do sisitema;

$$\det(A) = a_{11}\, a_{22} \ldots a_{nn} ,$$

e portanto todos os coeficientes a_{jj} serão diferentes de zero.

O programa abaixo resolve um sistema triangular superior. A sintaxe é:

`TS(A)`

onde A é a matriz aumentada (também chamada completa) n por $n+1$ associada ao sistema. A coluna $n+1$ é justamente a coluna dos termos independentes b_j. Usamos aqui o "pacote" `linalg` do Maple para trabalhar com matrizes. Note que o programa sabe testar se o sistema é triangular ou não.

```
> restart: with(linalg):
Warning, new definition for norm
Warning, new definition for trace
> TS := proc(a)
> local n, x, i, j, t, soma:
> n := rowdim(a): # dimensão do espaço linhas.
> # Teste para sistema triangular.
>   for i from 2 to n do
>    for j from 1 to i-1 do
>     if a[i,j] <> 0 then ERROR(`Este sistema não é triangular
  superior`) fi
>    od:
>   od:
> # Substituição inversa.
>   x[n] := a[n,n+1]/a[n,n]:
>    for j from 1 to n-1 do
>      soma := 0:
>       for t from n-j+1 to n do
>        soma := soma + a[n-j,t]*x[t]:
>       od:
>      x[n-j] := (a[n-j,n+1]-soma)/a[n-j,n-j]:
>    od:
> # Escrevendo vetor solução.
> vector( [seq(x[s], s=1..n)] ):
> end:
> # Exemplos
> A1 := matrix([ [2,1,3,2], [0,3,-5,8] , [0,0,2,-2] ]);
```

$$A1 := \begin{bmatrix} 2 & 1 & 3 & 2 \\ 0 & 3 & -5 & 8 \\ 0 & 0 & 2 & -2 \end{bmatrix}$$

```
> TS(A1);
```

$$[2, 1, -1]$$

Vejamos um sistema não triangular

```
> A2 := matrix([ [2,4,3,2], [7,3,5,3] , [3,0,2,-1] ]);
```

$$A2 := \begin{bmatrix} 2 & 4 & 3 & 2 \\ 7 & 3 & 5 & 3 \\ 3 & 0 & 2 & -1 \end{bmatrix}$$

```
> TS(A2);
Error, (in TS) Este sistema não é triangular superior

>
```

Método de Eliminação de Gauss

Seja $S\,x = b$ um sistema linear. Aplicando sobre esse sistema uma sequência de operações elementares escolhidas entre:

- E1) Trocas 2 equações;
- E2) Multiplicar uma equação por uma constante não nula;
- E3) Adicionar um múltiplo de uma equação a uma outra equação,

obtemos um novo sistema $M\,x = B$ equivalente ao sistema $S\,x = b$.

Supomos que o determinante da matriz de sistema linear $S\,x = b$ seja não nulo, isto é, $\det(S) \neq 0$.

Veremos agora o algoritmo de eliminação de Gauss, também conhecido por método de *escalonamento*, que usa os resultados acima para triangularizar a matriz do sistema linear. O método de Eliminação de Gauss, consiste em transformar um sistema linear (não triangular superior) num sistema triangular superior equivalente, por meio de operações elementares evitando o cálculo da matriz inversa do sistema. O processo é feito através da matriz aumentada e utiliza a noção de pivô e dos chamados multiplicadores m_{ij} , $i = 1, 2, 3, \dots, n$ e $j = 1, 2, 3, \dots, n$.

A eliminação é efetuada por colunas e chamaremos de etapa k do processo a fase que eleimnia a variável x_k das equações $k+1$, $k = 2, \dots, n$.

Denotaremos por $a_{ij}^{\{k\}}$ o coeficiente da linha i e coluna j no final da k-ésima etapa, bem como $b_i^{\{k\}}$ o i-ésimo elemento do vetor constante b no final da etapa k.

Considerando que o determinante da matriz do sistema $S\,x = b$ é não nulo, $\det(S) \neq 0$, é sempre possível reescrever o sistema de modo que $a_{11} \neq 0$, usando apenas a operação elementar E1.

Seja $A^{\{0\}}$ a matriz aumentada do sistema $S\,x = b$, em que $a_{ij}^{\{0\}} = a_{ij}$, $b_i^{\{0\}} = b_i$ e $a_{11}^{\{0\}} \neq 0$.

Primeira etapa

Mantemos a primeira equação com $a_{11} \neq 0$.

A eliminação da variável x_1 das demais equações $i = 2, 3, \dots$n é feita do seguinte modo: da i-ésima equação subtraímos a primeira equação multiplicada por $m_{i1} = \dfrac{a_{i1}^{\{0\}}}{a_{11}^{\{0\}}}$.

O elemento $a_{11}^{\{0\}}$ é o pivô da primeira etapa, e m_{i1} é o multiplicador da primeira etapa.
Assim, $a_{1j}^{\{1\}} = a_{1j}^{\{0\}}$, $j = 1, 2, ..., n$ e $b_i^{\{1\}} = b_1^{\{0\}}$ e $a_{ij}^{\{1\}} = a_{ij}^{\{0\}} - m_{i1}\, a_{1j}^{\{0\}}$, $i = 2, ..., n$ e $j = 1$, ..., n.

Segunda etapa

Mantemos a segunda equação se $a_{22} \neq 0$, caso contrário, existe pelo menos um elemento $a_{i2} \neq 0$, caso contrário o determinante de matriz do novo sistema equivalente ao primeiro seria nulo.Assim, sempre é possível reescrever a matriz $A^{\{1\}}$ sem alterar a posição da primeira linha de modo que o pivô seja não nulo.

Os multiplicadores desta etapa são os elementos $m_{i2} = \dfrac{a_{i2}^{\{1\}}}{a_{22}^{\{1\}}}$, para $i = 3, ..., n$.

A variável x_2 é eliminada da terceira equação em diante da seguinte forma: da i-ésima equação subraimos a segunda equação multiplicada por m_{i2}.

>

Assim, $a_{ij}^{\{2\}} = a_{ij}^{\{1\}}$, $i = 1, 2$ e $j = i, i+1, ..., n$ e $b_i^{\{2\}} = b_1^{\{1\}}$ para $i = 1, 2$ e
$a_{ij}^{\{2\}} = a_{ij}^{\{1\}} - m_{i2}\, a_{2j}^{\{1\}}$, $i = 3, ..., n$ e $j = 2, ..., n$ e $\{\, b_i^{2} \,\} = b_1^{\{1\}} - m_{i2}\, b_2^{\{1\}}$ para $i = 3, ..., n$.

Segue-se esse raciocínio até a etapa $n-1$ e a matrix final $A^{\{n-1\}}$ será uma matriz triangular superior.

O sistema ? é um sistema triangular superior, obtido através de operações elementares com suas linhas e a solução pode ser obtida por substituição: Na última equação obtemos o valor de x_n. Substituindo o valor de x_n na n-1-ésima equação, obtemos o valor de x_{n-1}, e assim sucessivamente podemos obter a solução do sistema.

```
> restart;
> A0 := matrix([ [3,2,4,1], [1,1,2,2] , [4,3,-2,3] ]);
```

$$A0 := \begin{bmatrix} 3 & 2 & 4 & 1 \\ 1 & 1 & 2 & 2 \\ 4 & 3 & -2 & 3 \end{bmatrix}$$

```
> 'A0[1,1]' = A0[1,1]; # pivô
```

$$A0_{1,1} = 3$$

```
> m[21] := A0[2,1]/A0[1,1];  m[31] := A0[3,1]/A0[1,1];
```

$$m_{21} := \frac{1}{3}$$

$$m_{31} := \frac{4}{3}$$

```
> L1 := [3, 2, 4, 1]; L2 := [1, 1, 2, 2]; L3 := [4, 3, -2, 3];
```

$$L1 := [3, 2, 4, 1]$$

$$L2 := [1, 1, 2, 2]$$

$$L3 := [4, 3, -2, 3]$$

```
> L1; L2 := L2 - m[21]*L1; L3 := L3 - m[31]*L1;
```

$$[3, 2, 4, 1]$$

$$L2 := \left[0, \frac{1}{3}, \frac{2}{3}, \frac{5}{3}\right]$$

$$L3 := \left[0, \frac{1}{3}, \frac{-22}{3}, \frac{5}{3}\right]$$

```
> A1 := matrix([ [3,2,4,1], [0,1/3,2/3,5/3] , [0,1/3,-22/3,5/3]
  ]);
```

$$A1 := \begin{bmatrix} 3 & 2 & 4 & 1 \\ 0 & \frac{1}{3} & \frac{2}{3} & \frac{5}{3} \\ 0 & \frac{1}{3} & \frac{-22}{3} & \frac{5}{3} \end{bmatrix}$$

```
> 'A1[2,2]' = A1[2,2]; # pivô
```

$$A1_{2,2} = \frac{1}{3}$$

```
> m[32] := A1[3,2]/A1[2,2];
```

$$m_{32} := 1$$

```
> 'L1'= L1; 'L2'= L2; 'L3'= L3;
```

$$L1 = [3, 2, 4, 1]$$

$$L2 = \left[0, \frac{1}{3}, \frac{2}{3}, \frac{5}{3}\right]$$

$$L3 = \left[0, \frac{1}{3}, \frac{-22}{3}, \frac{5}{3}\right]$$

```
> L1; L2; L3 := L3 - m[32]*L2;
```

$$[3, 2, 4, 1]$$

$$\left[0, \frac{1}{3}, \frac{2}{3}, \frac{5}{3}\right]$$

$$L3 := [0, 0, -8, 0]$$

```
> A2 := matrix([ L1, L2, L3]);
```

$$A2 := \begin{bmatrix} 3 & 2 & 4 & 1 \\ 0 & \frac{1}{3} & \frac{2}{3} & \frac{5}{3} \\ 0 & 0 & -8 & 0 \end{bmatrix}$$

```
>
```

De fato, o sistema

```
> sis0 := {3*x +2*y+4*z = 1, 1*x+y +2*z = 2, 4*x +3*y-2*z = 3};
```

$$sis0 := \{3\,x + 2\,y + 4\,z = 1, x + y + 2\,z = 2, 4\,x + 3\,y - 2\,z = 3\}$$

```
> solve( sis0, {x,y,z});
```

$$\{z = 0, y = 5, x = -3\}$$

possui a mesma solyução que o sistema triangular superior equivalente

```
> sisT0 := {3*x +2*y+4*z = 1, 1/3*y + 2/3*z = 5/3,  -8*z = 0};
> solve( sisT0, {x,y,z});
```

$$sisT0 := \{3\,x + 2\,y + 4\,z = 1, \frac{1}{3}\,y + \frac{2}{3}\,z = \frac{5}{3}, -8\,z = 0\}$$

$$\{z = 0, y = 5, x = -3\}$$

```
>
```

O programa dado a seguir triangulariza o sistema e opcionalmente fornece a solução. Define-se a Matriz aumentada do sistema A e usamos a sintaxe:

`eliminaG(A)`

em que A realmente é a matriz aumentada n por $n+1$ associada ao sistema linear.

Como de costume, a coluna $n+1$ é justamente a coluna dos termos independentes b_j do sistema.

```
> with(linalg):
> EliminaG := proc(a)
> local n, pivot, soma, antigo, i, j, k, p, s, x, m:
> n := rowdim(a);
> for k from 1 to n-1 do
>  # Escolhendo o pivot da etapa k
>    pivot := 0:
>    for i from k to n do
>      if abs(pivot) < abs(a[i,k]) then
>      pivot := a[i,k]: p:=i: fi
>    od:
>  # Trocando a linha k pela linha p
>      if k<>p then
>       for s from k to n+1 do
>        antigo := a[k,s]:
>        a[k,s] := a[p,s]:
>        a[p,s] := antigo:
>       od:
>      fi:
>  # Escalonado a coluna k
>    for i from k+1 to n do
>     m := a[i,k]/pivot:
>     a[i,k] := 0:
>       for j from k+1 to n+1 do
>         a[i,j] := a[i,j]-m*a[k,j]:
>       od:
>     od:
> od:
> # Escrevendo o sistema triangular
> op(a):
>     # Rotina para substituição inversa (opcional)
>     # x[n] := a[n,n+1]/a[n,n]:
```

```
>      # for j from 1 to n-1 do
>      #   soma:=0:
>      #     for s from n-j+1 to n do
>      #      soma := soma+a[n-j,s]*x[s]
>      #     od:
>      #   x[n-j] := (a[n-j,n+1]-soma)/a[n-j,n-j]
>      # od:
>      # Escrevendo a solução do sistema
>      # vector( [ seq(x[j], j=1..n) ] ):
> end:
>
> A3 := matrix([ [2,4,3,2,-4], [-2,7,3,5,3] , [3,0,1,2,-1], [3,3,3,3,3] ]);
```

$$A3 := \begin{bmatrix} 2 & 4 & 3 & 2 & -4 \\ -2 & 7 & 3 & 5 & 3 \\ 3 & 0 & 1 & 2 & -1 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix}$$

```
> E3 := EliminaG(A3);
```

$$E3 := \begin{bmatrix} 3 & 0 & 1 & 2 & -1 \\ 0 & 7 & \frac{11}{3} & \frac{19}{3} & \frac{7}{3} \\ 0 & 0 & \frac{3}{7} & \frac{-12}{7} & 3 \\ 0 & 0 & 0 & -2 & \frac{-19}{3} \end{bmatrix}$$

Diretamente no Maple V teremos com gausselim do pacote linalg como calcular diretamente

```
> sis4 := {1*x +2*y+3*z = b1, 1*x+3*y +0*z = b2, 1*x +4*y+3*z = b3};
```

$$sis4 := \{x + 2\,y + 3\,z = b1, x + 3\,y = b2, x + 4\,y + 3\,z = b3\}$$

```
> A4 := array([[1,2,3],[1,3,0],[1,4,3]]);
```

$$A4 := \begin{bmatrix} 1 & 2 & 3 \\ 1 & 3 & 0 \\ 1 & 4 & 3 \end{bmatrix}$$

```
> gausselim(A4, 'r' , 'd');
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & -3 \\ 0 & 0 & 6 \end{bmatrix}$$

```
> 'posto(A4)' = r;
```

$$\text{posto}(A4) = 3$$

```
> 'det(A4)' = d;
```

$$\det(A4) = 6$$

E resolvendo o sisitema diretamente no Maple, temos

```
> solve( sis4, {x,y,z});
```

$$\{z = -\frac{1}{3} b2 + \frac{1}{6} b1 + \frac{1}{6} b3, y = -\frac{1}{2} b1 + \frac{1}{2} b3, x = \frac{3}{2} b1 - \frac{3}{2} b3 + b2\}$$

```
>
> C4 := linalg[concat](A4,array([[1,0,0],[0,1,0],[0,0,1]]));
```

$$C4 := \begin{bmatrix} 1 & 2 & 3 & 1 & 0 & 0 \\ 1 & 3 & 0 & 0 & 1 & 0 \\ 1 & 4 & 3 & 0 & 0 & 1 \end{bmatrix}$$

```
> Gaussjord(C4) mod 5;
```

$$\begin{bmatrix} 1 & 0 & 0 & 4 & 1 & 1 \\ 0 & 1 & 0 & 2 & 0 & 3 \\ 0 & 0 & 1 & 1 & 3 & 1 \end{bmatrix}$$

```
> Inverse(A4) mod 5;
```

$$\begin{bmatrix} 4 & 1 & 1 \\ 2 & 0 & 3 \\ 1 & 3 & 1 \end{bmatrix}$$

```
> alias(a=RootOf(x^4+x+1) mod 2): # GF(2^4)
  A5 := array([[1,a,a^2],[a,a^2,a^3],[a^2,a^3,1]]);
```

$$A5 := \begin{bmatrix} 1 & a & a^2 \\ a & a^2 & a^3 \\ a^2 & a^3 & 1 \end{bmatrix}$$

```
> Gausselim(A5,'r','d') mod 2;
```

$$\begin{bmatrix} 1 & a & a^2 \\ 0 & 0 & a \\ 0 & 0 & 0 \end{bmatrix}$$

```
> A6 := matrix([ [2,4,3,2,-4], [-2,3,4,5,3] , [2,4,3,2,-1],
  [3,3,3,3,3] ]);
```

$$A6 := \begin{bmatrix} 2 & 4 & 3 & 2 & -4 \\ -2 & 3 & 4 & 5 & 3 \\ 2 & 4 & 3 & 2 & -1 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix}$$

```
> B6 := EliminaG(A6);
```

$$B6 := \begin{bmatrix} 3 & 3 & 3 & 3 & 3 \\ 0 & 5 & 6 & 7 & 5 \\ 0 & 0 & \frac{-7}{5} & \frac{-14}{5} & -5 \\ 0 & 0 & 0 & 0 & -3 \end{bmatrix}$$

Ex. 31 i pag 189

```
> restart;
> A := matrix([ [1,-3,1,1], [6,-18,4,2] , [-1,3,-1,4] ]);
```

$$A := \begin{bmatrix} 1 & -3 & 1 & 1 \\ 6 & -18 & 4 & 2 \\ -1 & 3 & -1 & 4 \end{bmatrix}$$

```
> 'A[1,1]' = A[1,1]; # pivô
```

$$A_{1,1} = 1$$

```
> m[21] := A[2,1]/A[1,1];  m[31] := A[3,1]/A[1,1];
```

$$m_{21} := 6$$

$$m_{31} := -1$$

```
> L1 := [1, -3, 1, 1]; L2 :=  [6, -18, 4, 2]; L3 := [-1, 3, -1,
  4];
```

$$L1 := [1, -3, 1, 1]$$

$$L2 := [6, -18, 4, 2]$$

$$L3 := [-1, 3, -1, 4]$$

```
> L1; L2 := L2 - m[21]*L1; L3 := L3 - m[31]*L1;
```

$$[1, -3, 1, 1]$$

$$L2 := [0, 0, -2, -4]$$

$$L3 := [0, 0, 0, 5]$$

```
> A1 := matrix([ [1,-3,1,1], [0,0,-2,-4] , [0,0,0,5] ]);
```

$$A1 := \begin{bmatrix} 1 & -3 & 1 & 1 \\ 0 & 0 & -2 & -4 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

```
>
```

De fato, o sistema

```
> sis1 := {x -3*y+1*z = 1, 6*x-18*y +4*z = 2, -1*x +3*y-z = 4};
```

$$sis1 := \{x - 3y + z = 1, 6x - 18y + 4z = 2, -x + 3y - z = 4\}$$

```
> solve( sis1, {x,y,z});
```

Não possui solução, pois apenas numa etapa para obter o sistema triangular superior equivalente já encontramos $0\,z = 5$!

Sistemas Tridiagonais

No estudo de equações diferenciais de segunda ordem com condições de contorno, quando fazemos a discretização, somos conduzidos a resolução de sistemas lineres cuja matriz associada é tridiagonal.

Uma matriz é tridiagonal se $a_{ij} = 0$ sempre que $1 < |i - j|$. Um exemplo típico de matriz tridiagonal é:

$$\begin{bmatrix} 7 & 1 & 0 & 0 \\ 1 & 4 & 3 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 1 & 7 \end{bmatrix}$$

em que a diagonal principal e suas paralelas adjacentes são não nulas.

No que segue, faremos uma adaptação do método de eliminação de Gauss de modo a tirar vantagem da forma especial das matrizes tridiagonais. O procedimento Tridiag resolve um sistema tridiagonal através do método de eliminação de Gauss sem considerar a troca de linhas. O programa também testa se o sistema é tridiagonal. Note que após o escalonamento das linhas, teremos $a_{ij} = 0$ se $j = i + 2, \ldots, n$ onde $i = 1, \ldots, n - 2$. Isto simplificará sensivelmente a etapa da substituição inversa. A sintaxe do procedimento é

```
Tridiag(A)
```

onde A é a matriz aumentada n por $n+1$ do sistema.

```
> restart; with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected

> Tridiag := proc(a)
> local x, i, j, k, l, m, n:
> n := rowdim(a):
> # Testando se é tridiagonal
>   for i from 1 to n do
>   for j from 1 to n do
>     if abs(i-j)>1 then
>       if a[i,j]<>0 then
>       ERROR(`O Sistema não é tridiagonal`)
>       fi:
>     fi:
>   od od:
> # Escalonando o sistema
>   for k from 1 to n-1 do
>     m := a[k+1,k]/a[k,k]:
>     a[k+1,k] := 0:
>     a[k+1,k+1] := a[k+1,k+1]-m*a[k,k+1]:
>     a[k+1,n+1] := a[k+1,n+1]-m*a[k,n+1]:
>   od:
> # Fazendo a substituição inversa.
>   x[n] := a[n,n+1]/a[n,n]:
>   for l from 1 to n-1 do
>     x[n-l] := (a[n-l,n+1]-a[n-l,n-l+1]*x[n-l+1])/a[n-l,n-l]
>   od:
> # Escrevendo o vetor solução.
>   vector( [seq(x[s], s=1..n)] ):
> end:
>
```

Exemplo de uma matriz que não é tridiagonal

```
> A7 := matrix([[7,1,5,1],[1,4,1,1],[2,1,4,1]]);
```

$$A7 := \begin{bmatrix} 7 & 1 & 5 & 1 \\ 1 & 4 & 1 & 1 \\ 2 & 1 & 4 & 1 \end{bmatrix}$$

```
> Tridiag(A7);
Error, (in Tridiag) O Sistema não é tridiagonal
```

Exemplo de uma matriz é tridiagonal

```
> A8 :=
  matrix([[7,1,0,0,9],[1,4,3,0,18],[0,1,4,1,18],[0,0,1,7,31]]);
```

$$A8 := \begin{bmatrix} 7 & 1 & 0 & 0 & 9 \\ 1 & 4 & 3 & 0 & 18 \\ 0 & 1 & 4 & 1 & 18 \\ 0 & 0 & 1 & 7 & 31 \end{bmatrix}$$

```
> sol8 := Tridiag(A8);
```

$$sol8 := [1, 2, 3, 4]$$

Exemplos

```
> restart; with(linalg):
Warning, the protected names norm and trace have been redefined and
unprotected
```

Ex.

```
> A4 := array([[3,2,1],[1,1,3],[-1,0,4]]);
```

$$A4 := \begin{bmatrix} 3 & 2 & 1 \\ 1 & 1 & 3 \\ -1 & 0 & 4 \end{bmatrix}$$

```
> b4 := transpose(array([[3,4,4]]));
```

$$b4 := \begin{bmatrix} 3 \\ 4 \\ 4 \end{bmatrix}$$

Resolvendo diretamente

```
> linsolve( A4, b4, 'r');
```

$$\begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

Ex.

```
> A5 := array([[0,8,1],[0,1,3],[2,0,4]]);
```

$$A5 := \begin{bmatrix} 0 & 8 & 1 \\ 0 & 1 & 3 \\ 2 & 0 & 4 \end{bmatrix}$$

```
> b5 := transpose(array([[-8,-1,4]]));
```

$$b5 := \begin{bmatrix} -8 \\ -1 \\ 4 \end{bmatrix}$$

```
> linsolve( A5, b5, 'r');
```

$$\begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix}$$

Ex.

```
> A6 := array([[2,1,1],[1,1,1],[0, 2,4]]);
```

$$A6 := \begin{bmatrix} 2 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 2 & 4 \end{bmatrix}$$

```
> gausselim(A6);
```

$$\begin{bmatrix} 2 & 1 & 1 \\ 0 & 2 & 4 \\ 0 & 0 & \frac{-1}{2} \end{bmatrix}$$

```
> LUdecomp(A6);
```

$$\begin{bmatrix} 2 & 1 & 1 \\ 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 2 \end{bmatrix}$$

```
>
```