Vetorização:
Decomposição LU

Flávio Henrique Lopes Barbosa
Jaysa Keylla Siqueira Barbosa
José Augusto Agripino de Oliveira

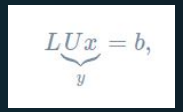
Decomposição LU
Lower Upper

Um dos motivos para introduzir a decomposição LU é que ela fornece uma maneira eficiente de calcular a matriz inversa, a qual tem muitas aplicações na engenharia; ela também fornece um meio de avaliar o condicionamento do sistema.

A decomposição pode ser dividida em dois passos:

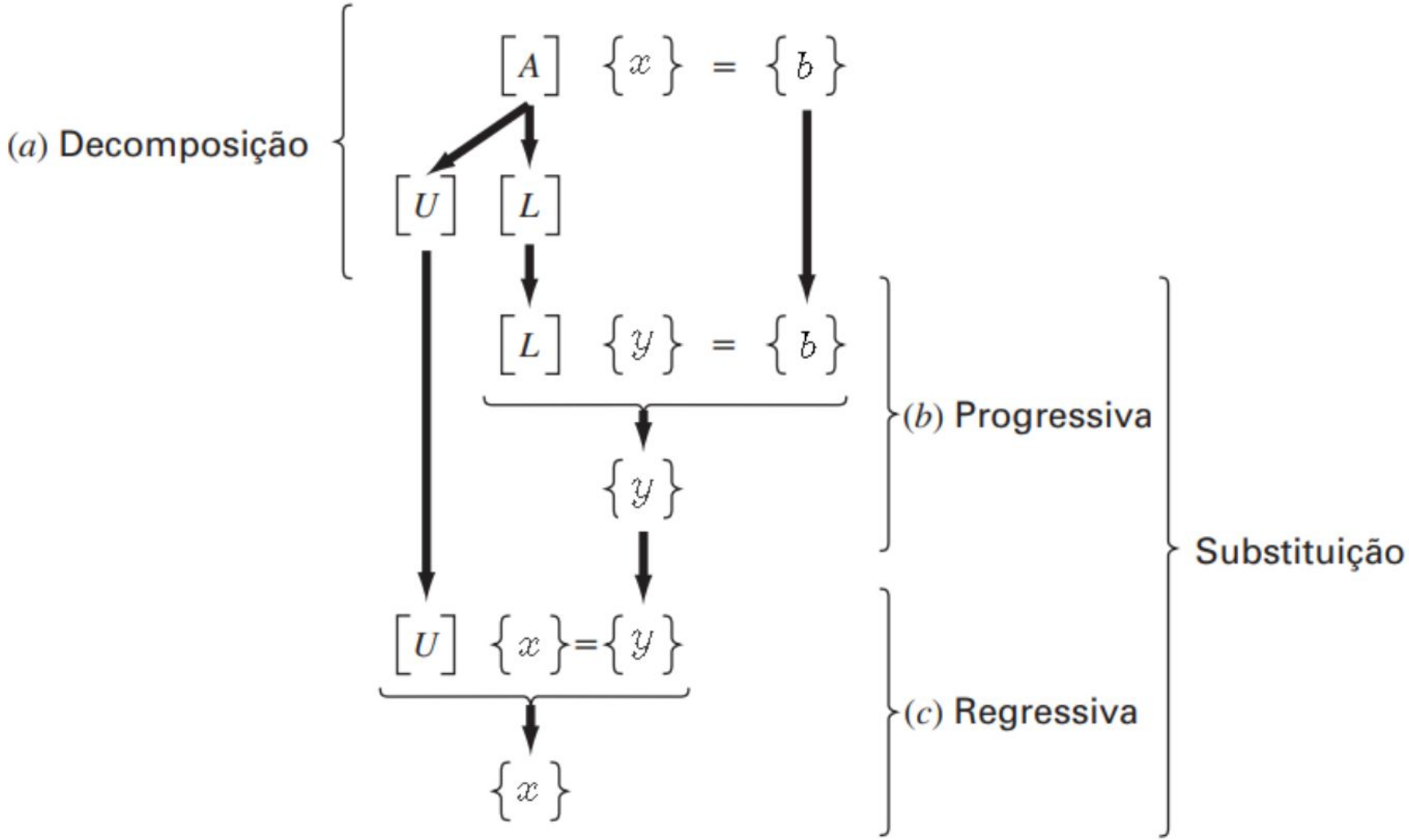
1 – Passo de decomposição: a matriz **A** é fatorada em duas matrizes triangulares, uma inferior L com elementos da diagonal principal iguais a 1, e uma superior **U** , onde, realizando a multiplicação **L×U**, obtemos a matriz **A**.

2 – Resolução do sistema: **L** e **U** são usadas para determinar a solução do sistema, **x**, através do processo: **Ax = b**
Se **A = LU**, então **LUx = y**. Defina um vetor de incógnitas auxiliar, **y**:



Ou seja, **Ux = y.** Logo, **Ly = b .**

Observe que no sistema acima, **L** é uma matriz triangular inferior, **b** é a matriz de termos independentes do sistema original e **y** é o vetor de incógnitas auxiliar. Resolvendo **Ly=b** usando substituição progressiva, podemos usar a relação **Ux=y** para encontrar xx por substituição regressiva, já que **U** é triangular superior.


(a) Decomposição
[A] {x} = {b}
[U] [L]
[L] {y} = {b}
{y}
(b) Progressiva
[U] {x}={y}
{x}
(c) Regressiva
Substituição

$$LU = \begin{pmatrix} 1 & & & & \\ \ell_{21} & 1 & & \bigcirc & \\ \ell_{31} & \ell_{32} & 1 & & \\ \dots & \dots & \dots & \ddots & \\ \ell_{n1} & \ell_{n2} & \ell_{n3} & \dots & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ & u_{22} & u_{23} & \dots & u_{2n} \\ & & u_{33} & \dots & u_{3n} \\ & \bigcirc & & \ddots & \vdots \\ & & & & u_{nn} \end{pmatrix}$$

GCC

O GCC é um compilador com recursos avançados e com as flags de otimização, o programador pode indicar ao compilador para que procure por laços e otimize-os .Tais recursos foram aplicados ao código de decomposição LU para matrizes.

Resultados

-Matriz de ordem 4000

-Matriz de ordem 5657

-Matriz de ordem 8000

-Matriz de ordem 11314

-Matriz de ordem 16000

Comando utilizados

1. $ gcc -pg -O3 -ftree-vectorize -fopt-info-vec-missed -fopt-info-vec-optimized decomposicaoLU.c -o decomposicaoLU

2. $ ./decomposicaoLU

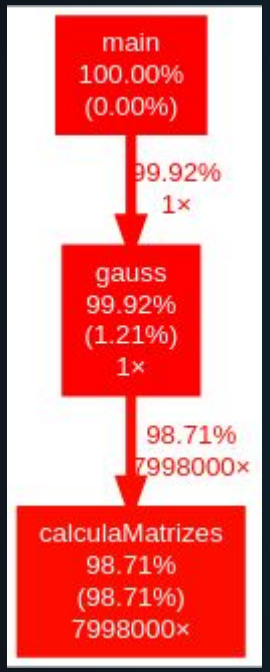
3. $ gprof decomposicaoLU gmon.out > orden_n.txt

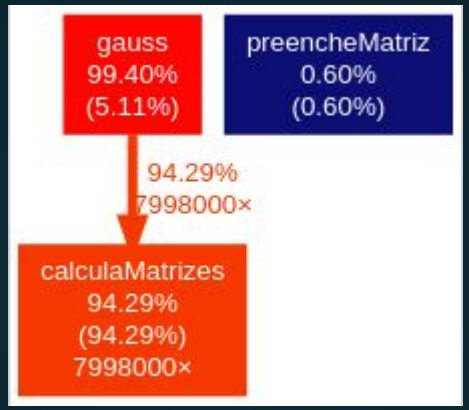
4. $ gprof decomposicaoLU | gprof2dot > out_n.dot

Matriz de ordem 4000

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls   s/call   s/call  name
 98.71     72.73    72.73  7998000     0.00     0.00  calculaMatrizes
  1.21     73.62     0.89        1     0.89    73.62  gauss
  0.08     73.68     0.06        1     0.06     0.06  preencheMatriz
```

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  us/call  us/call  name
 94.97     9.41     9.41  7998000     1.18     1.18  calculaMatrizes
  5.19     9.93     0.51                             gauss
  0.61     9.99     0.06                             preencheMatriz
```



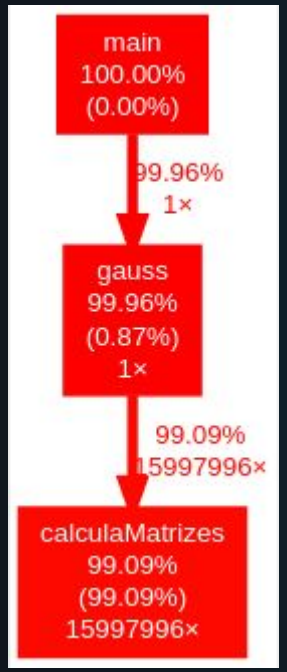


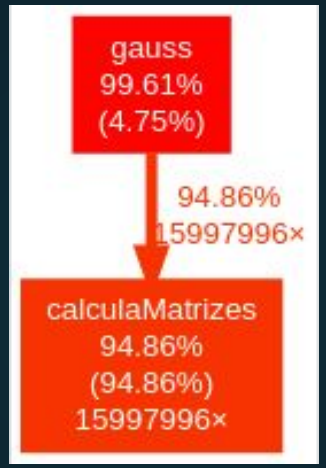
Matriz de ordem 4000

Matriz de ordem 5657

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls   s/call   s/call  name
 99.09    213.03   213.03 15997996     0.00     0.00  calculaMatrizes
  0.87    214.90     1.87        1     1.87   214.90  gauss
  0.03    214.97     0.07        1     0.07     0.07  preencheMatriz
  0.00    214.98     0.01                             _init
```

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  us/call  us/call  name
 95.57     21.96    21.96 15997996     1.37     1.37  calculaMatrizes
  4.78     23.06     1.10                             gauss
  0.39     23.15     0.09                             preencheMatriz
```



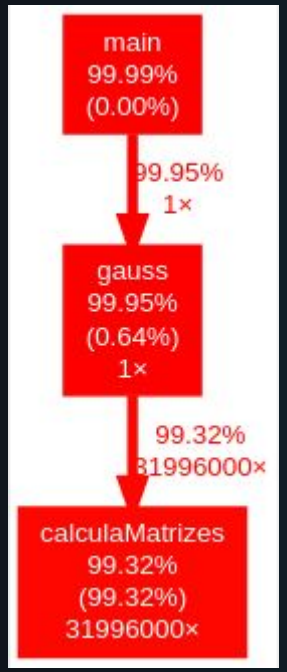


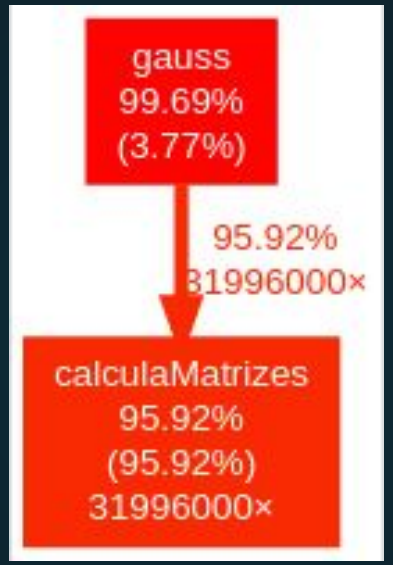
Matriz de ordem 5657

Matriz de ordem 8000

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls   s/call   s/call  name
 99.32    593.13   593.13 31996000     0.00     0.00  calculaMatrizes
  0.64    596.93     3.80        1     3.80   596.93  gauss
  0.04    597.17     0.24        1     0.24     0.24  preencheMatriz
  0.01    597.20     0.03                             _init
```

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  us/call  us/call  name
 96.65     67.89    67.89 31996000     2.12     2.12  calculaMatrizes
  3.80     70.56     2.67                             gauss
  0.32     70.79     0.22                             preencheMatriz
```



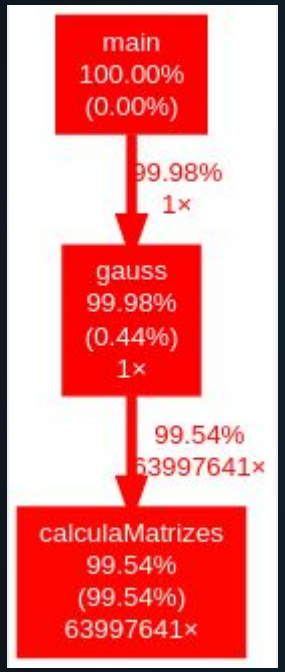


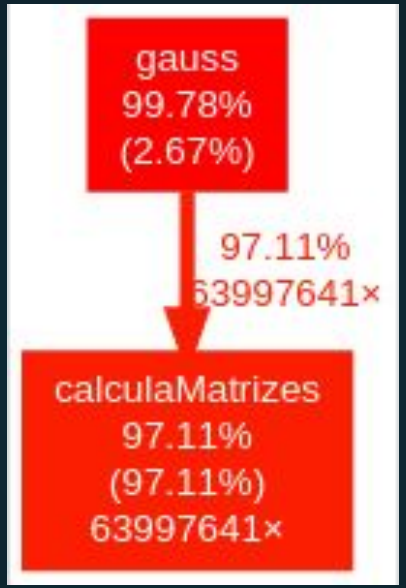
Matriz de ordem 8000

Matriz de ordem 11314

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  Ks/call  Ks/call  name
 99.54   1679.62  1679.62 63997641     0.00     0.00  calculaMatrizes
  0.44   1687.05     7.43        1     0.01     1.69  gauss
  0.02   1687.33     0.28        1     0.00     0.00  preencheMatriz
  0.00   1687.37     0.04                             _init
```

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  us/call  us/call  name
 97.83    192.50   192.50 63997641     3.01     3.01  calculaMatrizes
  2.69    197.79     5.29                             gauss
  0.22    198.21     0.42                             preencheMatriz
  0.01    198.22     0.01                             __libc_csu_init
```



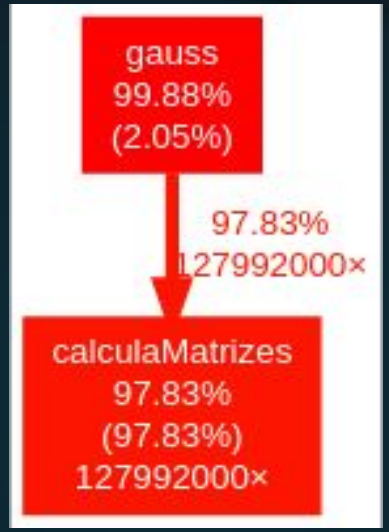


Matriz de ordem 11314

Matriz de ordem 16000

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  Ks/call  Ks/call  name
 99.37   2706.73  2706.73 127992000     0.00     0.00  calculaMatrizes
  0.61   2723.35    16.62        1     0.02     2.72  gauss
  0.02   2723.78     0.43        1     0.00     0.00  preencheMatriz
  0.00   2723.84     0.06                             _init
```

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  us/call  us/call  name
 98.58    560.87   560.87 127992000     4.38     4.38  calculaMatrizes
  2.06    572.60    11.73                             gauss
  0.12    573.31     0.71                             preencheMatriz
```



Matriz de ordem 16000

Conclusão

Os resultados obtidos com a auto-vetorização foram muito melhores que os resultados sem a auto-vetorização.

O gargalo, que é a função calculaMatrizes, teve um tempo muito menor de execução com a auto-vetorização.

Conforme a mensagem explicitada pelo GCC após as flags de auto-vetorização, faltaram laços de repetição para serem vetorizados.

- Laços das linhas 20 e 21;
- Laços das linhas 48 e 50.

Sendo assim, é necessário modificar o código para que seja possível a vetorização.