

# Trabalho Computacional 1 - ALN

João Lucas Duim

08 de Abril de 2021

## 1 Questão 1

No arquivo “Metodo-Jacobi.sci” encontra-se o código da função utilizada para resolver um sistema linear  $Ax = b$  usando o algoritmo iterativo de Jacobi.

## 2 Questão 2

No arquivo “Metodo-Gauss-Seidel-1.sci” encontra-se o código da função utilizada para resolver um sistema linear  $Ax = b$  usando o algoritmo iterativo de Gauss-Seidel e a função “inv” do Scilab.

No arquivo “Metodo-Gauss-Seidel-2.sci” encontra-se o código da função utilizada para resolver um sistema linear  $Ax = b$  usando o algoritmo iterativo de Gauss-Seidel e a função “Resolve-Sistema-Lx” implementada no mesmo arquivo para resolver sistemas em que a matriz dos coeficientes é triangular inferior.

## 3 Questão 3

Veja, na figura 1, a função Metodo-Jacobi aplicada ao sistema dado no enunciado usando o vetor nulo como aproximação inicial.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
[Icons]
Scilab 6.1.0 Console
A =
1. -4. 2.
0. 2. 4.
6. -1. -2.
--> b = [2; 1; 1]
b =
2.
1.
1.
--> x0 = [0;0;0];
--> [xk1,e,k,rk]=Metodo_Jacobi(A,b,x0,10^(-7),1000,%inf)
xk1 =
Nan
Nan
Nan
e =
Nan
k =
609.
rk =
Nan
-->
```

Figure 1: Função Metodo-Jacobi aplicada ao sistema dado

Veja, na figura 2, a função Metodo-Gauss-Seidel-1 aplicada ao sistema dado no enunciado usando o vetor nulo como aproximação inicial.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
[Icons]
Scilab 6.1.0 Console
A =
1. -4. 2.
0. 2. 4.
6. -1. -2.
--> b = [2; 1; 1]
b =
2.
1.
1.
--> x0 = [0;0;0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf)
xk1 =
Inf
Inf
Nan
e =
Nan
k =
448.
rk =
Nan
-->
```

Figure 2: Função Metodo-Gauss-Seidel-1 aplicada ao sistema dado

Veja, na figura 3, a função Metodo-Gauss-Seidel-2 aplicada ao sistema dado no enunciado usando o vetor nulo como aproximação inicial.

```
Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?
Scilab 6.1.0 Console
A =
1. -4. 2.
0. 2. 4.
6. -1. -2.
--> b = [2; 1; 1]
b =
2.
1.
1.
--> x0 = [0;0;0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf)
xk1 =
Inf
Nan
Nan
e =
Nan
k =
448.
rk =
Nan
-->
```

Figure 3: Função Metodo-Gauss-Seidel-2 aplicada ao sistema dado

Reordenando as equações do sistema dado a fim de tornar a matriz dos coeficientes de diagonal estritamente dominante, temos:

$$\begin{cases} 6x - y - 2z = 1 \\ x - 4y + 2z = 2 \\ 0x + 2y + 4z = 1 \end{cases}$$

Veja, na figura 4, a função Metodo-Jacobi aplicada ao sistema com equações reordenadas usando o vetor nulo como aproximação inicial.

```

Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?
Scilab 6.1.0 Console
A =
6. -1. -2.
1. -4. 2.
0. 2. 4.
--> b = [1; 2; 1]
b =
1.
2.
1.
--> x0 = [0;0;0];
--> [xk1,e,k,rk]=Metodo_Jacobi(A,b,x0,10^(-7),100,%inf)
xk1 =
0.25
-0.25
0.375
e =
8.298D-08
k =
23.
rk =
0.0000002
-->

```

Figure 4: Função Metodo-Jacobi aplicada ao sistema com equações reordenadas

Veja, na figura 5, a função Metodo-Gauss-Seidel-1 aplicada ao sistema com equações reordenadas usando o vetor nulo como aproximação inicial.

```

Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?
Scilab 6.1.0 Console
A =
6. -1. -2.
1. -4. 2.
0. 2. 4.
--> b = [1; 2; 1]
b =
1.
2.
1.
--> x0 = [0;0;0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),100,%inf)
xk1 =
0.25
-0.25
0.375
e =
6.209D-08
k =
13.
rk =
6.209D-08
-->

```

Figure 5: Função Metodo-Gauss-Seidel-1 aplicada ao sistema com equações reordenadas

Veja, na figura 6, a função Metodo-Gauss-Seidel-2 aplicada ao sistema com equações reordenadas usando o vetor nulo como aproximação inicial.

```

Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?
Scilab 6.1.0 Console
A =
6. -1. -2.
1. 4. 2.
0. 2. 4.
--> b = [1; 2; 1]
b =
1.
2.
1.
--> x0 = [0;0;0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),100,%inf)
xk1 =
0.25
-0.25
0.375
e =
6.209D-08
k =
13.
rk =
6.209D-08
-->

```

Figure 6: Função Metodo-Gauss-Seidel-2 aplicada ao sistema com equações reordenadas

Comentários: Nas figuras de 1 a 3 nota-se que os métodos das 3 funções divergiram, não obtendo uma solução para o problema inicialmente apresentado. No entanto, nas figuras de 4 a 6, foram feitas trocas de linhas na matriz dos coeficientes do sistema dado (e as mesmas trocas no vetor  $b$ , ou seja, foi feita simplesmente uma reordenação das equações do sistema) com o intuito de transformá-la em uma matriz com diagonal estritamente dominante, para a qual pode-se garantir que os métodos implementados nas 3 funções convergirão para uma solução, exatamente como apresentado nas referidas figuras.

## 4 Questão 4

O referido sistema é:

$$\begin{cases} 2x - y + z = -1 \\ 2x + 2y + 2z = 4 \\ -x - y + 2z = -5 \end{cases}$$

### 4.1 (a)

Veja, na figura 7, a função Metodo-Jacobi aplicada ao sistema dado usando o vetor nulo como aproximação inicial e um limite de 25 iterações.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
[Icons]
Scilab 6.1.0 Console
--> A = [2 -1 1; 2 2 2; -1 -1 2]
A =

  2. -1.  1.
  2.  2.  2.
 -1. -1.  2.

--> b = [-1; 4; -5]
b =

 -1.
  4.
 -5.

--> x0 = [0; 0; 0]
x0 =

  0.
  0.
  0.

--> [xk1,e,k,rk]=Metodo_Jacobi(A,b,x0,10^(-7),25,%inf)
"Número máximo de iterações atingido."
-->
-->
-->
-->
```

Figure 7: Função Metodo-Jacobi aplicada ao sistema dado

## 4.2 (b)

Veja, na figura 8, a função Metodo-Gauss-Seidel-1 aplicada ao sistema dado usando o vetor nulo como aproximação inicial e precisão de  $10^{-5}$  na norma-infinito.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
[Icons]
Scilab 6.1.0 Console
A =

  2. -1.  1.
  2.  2.  2.
 -1. -1.  2.

--> b = [-1; 4; -5]
b =

 -1.
  4.
 -5.

--> x0 = [0; 0; 0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_1(A,b,x0,10^(-5),100,%inf)
xk1 =

 1.0000023
 1.9999975
-1.0000001
e =

 0.0000073
k =

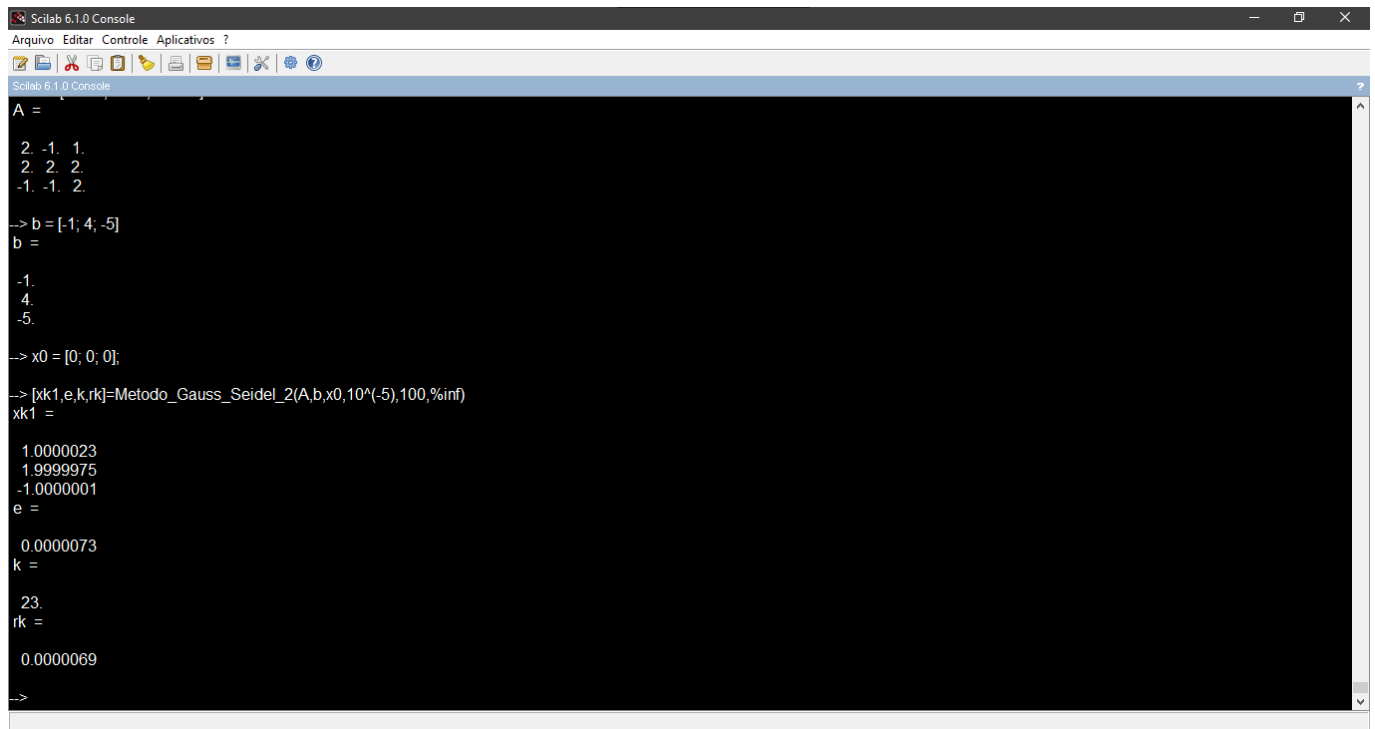
 23.
rk =

 0.0000069
-->
```

Figure 8: Função Metodo-Gauss-Seidel-1 aplicada ao sistema dado

Veja, na figura 9, a função Metodo-Gauss-Seidel-2 aplicada ao sistema dado usando o vetor nulo

como aproximação inicial e precisão de  $10^{-5}$  na norma-infinito.



```
Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?

A =
2 -1. 1.
2. 2. 2.
-1. -1. 2.

--> b = [-1; 4; -5]
b =
-1.
4.
-5.

--> x0 = [0; 0; 0];

--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_2(A,b,x0,10^(-5),100,%inf)
xk1 =
1.0000023
1.9999975
-1.0000001
e =
0.0000073
k =
23.
rk =
0.0000069
-->
```

Figure 9: Função Metodo-Gauss-Seidel-2 aplicada ao sistema dado

Comentários: Para o item (a), basta analisar a figura 7 que demonstra que 25 iterações não foram suficientes para se obter uma solução. Para o item (b), vemos que 23 iterações foram suficientes para que o método convergisse para uma solução. Vale ressaltar que os 2 itens dessa questão apresentam a maior eficiência obtida pelo método Gauss-Seidel quando comparado ao método de Jacobi.

## 5 Questão 5

O referido sistema é:

$$\begin{cases} 1x_1 + 0x_2 - x_3 = 0.2 \\ -0.5x_1 + 1x_2 - 0.25x_3 = -1.425 \\ x_1 - 0.5x_2 + x_3 = 2 \end{cases}$$

### 5.1 (a)

Veja, na figura 10, a função Metodo-Gauss-Seidel-1 aplicada ao sistema dado com tolerância de  $10^{-2}$  e um limite de 300 iterações.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
Scilab 6.1.0 Console
A =
1.  0. -1.
-0.5  1. -0.25
1. -0.5  1.
--> b = [0.2; -1.425; 2]
b =
0.2
-1.425
2.
--> x0 = [0; 0; 0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_1(A,b,x0,10^(-2),300,%inf)
xk1 =
0.8975131
-0.8018652
0.7015543
e =
0.0064659
k =
13.
rk =
0.0040412
-->
```

Figure 10: Função Metodo-Gauss-Seidel-1 aplicada ao sistema dado

Veja, na figura 11, a função Metodo-Gauss-Seidel-2 aplicada ao sistema dado com tolerância de  $10^{-2}$  e um limite de 300 iterações.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
Scilab 6.1.0 Console
A =
1.  0. -1.
-0.5  1. -0.25
1. -0.5  1.
--> b = [0.2; -1.425; 2]
b =
0.2
-1.425
2.
--> x0 = [0; 0; 0];
--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_2(A,b,x0,10^(-2),300,%inf)
xk1 =
0.8975131
-0.8018652
0.7015543
e =
0.0064659
k =
13.
rk =
0.0040412
-->
```

Figure 11: Função Metodo-Gauss-Seidel-2 aplicada ao sistema dado

Alterando as equações do sistema conforme pedido, temos:



$$\begin{cases} 1x_1 + 0x_2 - 2x_3 = 0.2 \\ -0.5x_1 + 1x_2 - 0.25x_3 = -1.425 \\ x_1 - 0.5x_2 + x_3 = 2 \end{cases}$$

Veja, na figura 12, a função Metodo-Gauss-Seidel-1 aplicada ao novo sistema dado com tolerância de  $10^{-2}$  e um limite de 300 iterações.

```

Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?
Scilab 6.1.0 Console
--> A = [1 0 -2; -0.5 1 -0.25; 1 -0.5 1]
A =

    1.    0.   -2.
   -0.5    1.  -0.25
    1.  -0.5    1.

--> b = [0.2; -1.425; 2]
b =

    0.2
   -1.425
    2.

--> x0 = [0; 0; 0]
x0 =

    0.
    0.
    0.

--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_1(A,b,x0,10^(-2),300,%inf)

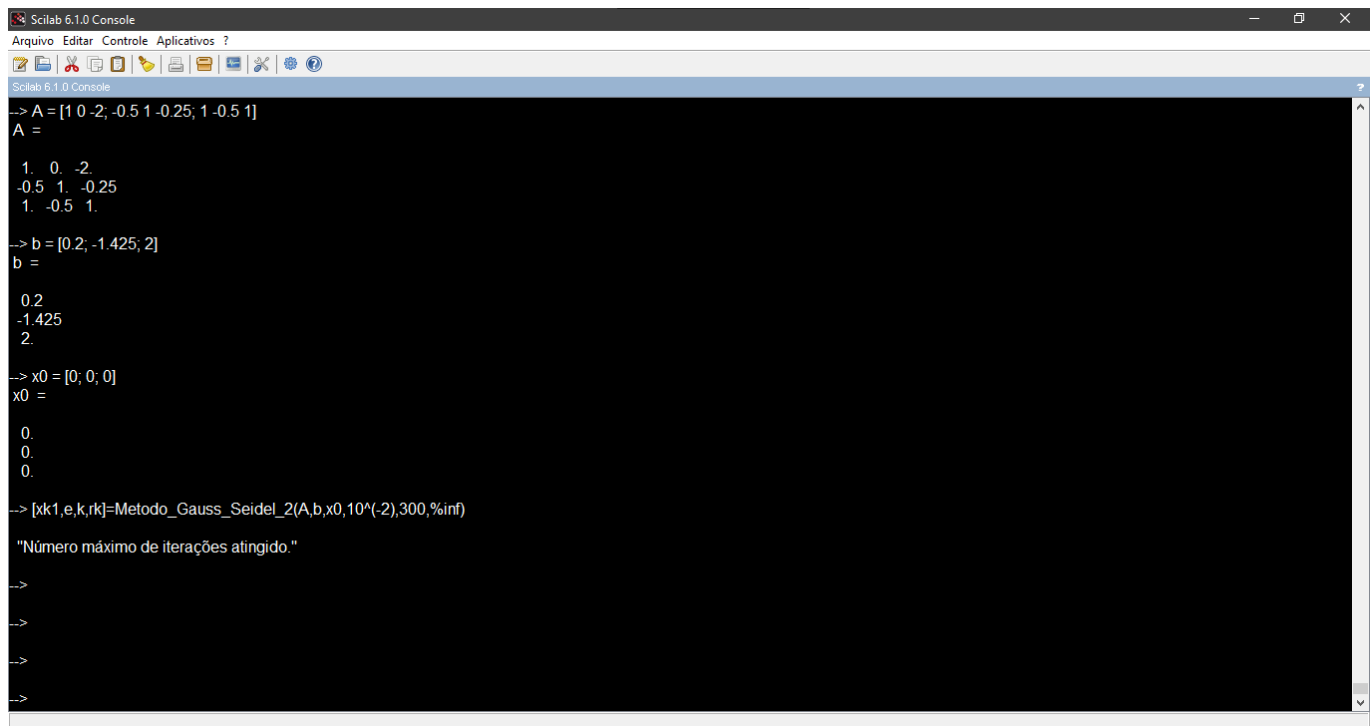
"Número máximo de iterações atingido."

-->
-->
-->
-->

```

Figure 12: Função Metodo-Gauss-Seidel-1 aplicada ao novo sistema

Veja, na figura 13, a função Metodo-Gauss-Seidel-2 aplicada ao novo sistema dado com tolerância de  $10^{-2}$  e um limite de 300 iterações.



```
Scilab 6.1.0 Console
Arquivo Editar Controle Aplicativos ?
[Icons]
Scilab 6.1.0 Console
--> A=[1 0 -2; -0.5 1 -0.25; 1 -0.5 1]
A =

  1.  0. -2.
-0.5  1. -0.25
  1. -0.5  1.

--> b=[0.2; -1.425; 2]
b =

  0.2
-1.425
  2.

--> x0=[0; 0; 0]
x0 =

  0.
  0.
  0.

--> [xk1,e,k,rk]=Metodo_Gauss_Seidel_2(A,b,x0,10^(-2),300,%inf)
"Número máximo de iterações atingido."
-->
-->
-->
-->
```

Figure 13: Função Metodo-Gauss-Seidel-2 aplicada ao novo sistema

Comentários: Para o item (a), observe nas figuras 10 e 11 as soluções aproximadas para o sistema dado obtidas pelas funções Gauss-Seidel-1 e Gauss-Seidel-2 em apenas 13 iterações. Para o item (b), observe nas figuras 12 e 13 que ambas as funções do método Gauss-Seidel não foram capazes de encontrar uma solução imposto o limite de 300 iterações, apesar de ter sido feita apenas uma pequena mudança no sistema original.

## 6 Questão 6

No arquivo “Gerador-Sistema.sci” encontra-se o código da função utilizada para gerar matrizes  $A_{n \times n}$  com diagonal estritamente dominante e vetores  $b$  de dimensão  $n$ .

Veja, na figura 14, os tempos, em segundos, envolvidos na aplicação de cada uma das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 ao sistema  $Ax = b$  gerado para  $n = 10$ , bem como os valores de  $e$ ,  $k$  e  $r_k$ , mostrando que uma solução foi de fato encontrada.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
Scilab 6.1.0 Console
--> x0 = zeros(10,1);
--> [A,b] = Gerador_Sistema(10);
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    0.1096676
--> disp(e,k,rk)
    3.616D-08
    10.
    4.363D-08
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    0.0029163
--> disp(e,k,rk)
    3.616D-08
    10.
    4.363D-08
-->
```

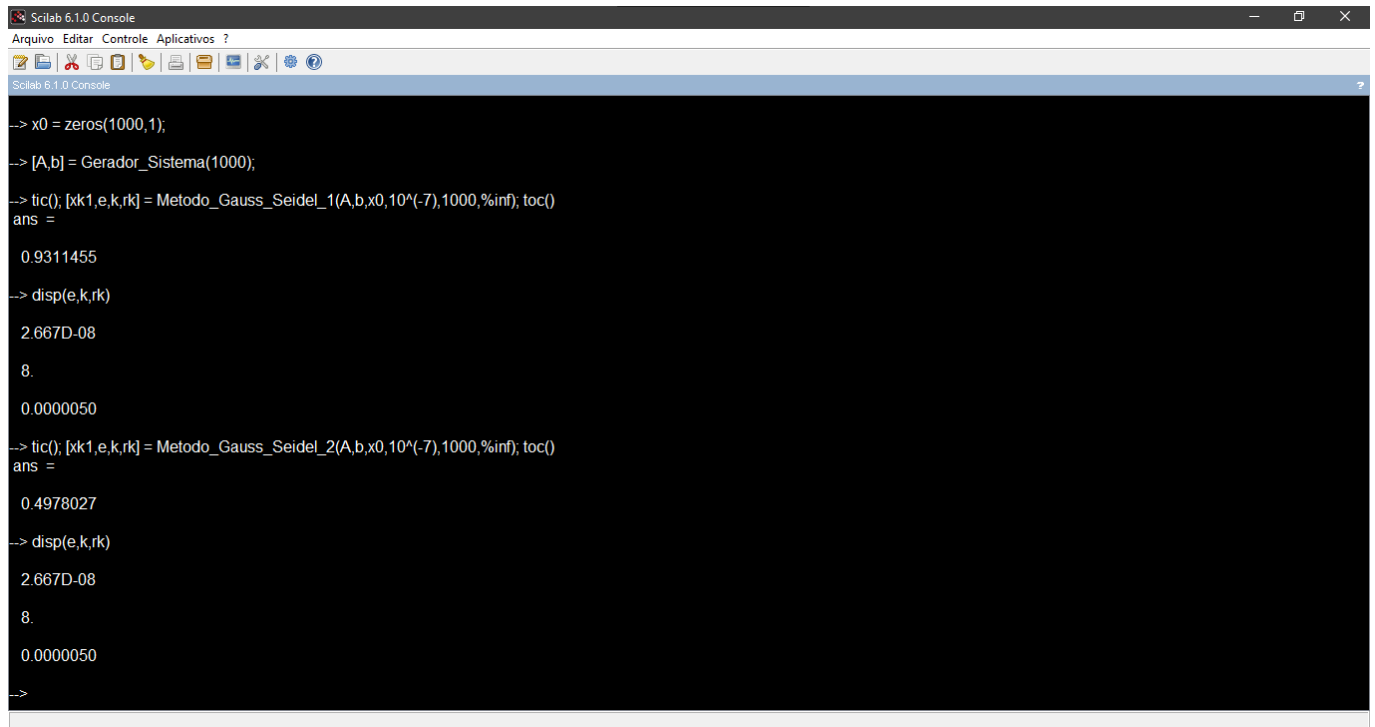
Figure 14: Tempo de execução das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 aplicadas ao sistema gerado para  $n = 10$

Veja, na figura 15, os tempos, em segundos, envolvidos na aplicação de cada uma das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 ao sistema  $Ax = b$  gerado para  $n = 100$ , bem como os valores de  $e$ ,  $k$  e  $r_k$ , mostrando que uma solução foi de fato encontrada.

```
Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
Scilab 6.1.0 Console
--> x0 = zeros(100,1);
--> [A,b] = Gerador_Sistema(100);
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    0.0040785
--> disp(e,k,rk)
    9.556D-08
    9.
    0.0000007
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    0.0222805
--> disp(e,k,rk)
    9.556D-08
    9.
    0.0000007
-->
```

Figure 15: Tempo de execução das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 aplicadas ao sistema gerado para  $n = 100$

Veja, na figura 16, os tempos, em segundos, envolvidos na aplicação de cada uma das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 ao sistema  $Ax = b$  gerado para  $n = 1000$ , bem como os valores de  $e$ ,  $k$  e  $r_k$ , mostrando que uma solução foi de fato encontrada.



```

--> x0 = zeros(1000,1);
--> [A,b] = Gerador_Sistema(1000);
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf), toc()
ans =

0.9311455
--> disp(e,k,rk)

2.667D-08
8.
0.0000050
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf), toc()
ans =

0.4978027
--> disp(e,k,rk)

2.667D-08
8.
0.0000050
-->

```

Figure 16: Tempo de execução das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 aplicadas ao sistema gerado para  $n = 1000$

Veja, na figura 17, os tempos, em segundos, envolvidos na aplicação de cada uma das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 ao sistema  $Ax = b$  gerado para  $n = 2000$ , bem como os valores de  $e$ ,  $k$  e  $r_k$ , mostrando que uma solução foi de fato encontrada.

```

Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
[Icons]
Scilab 6.1.0 Console
--> x0 = zeros(2000,1);
--> [A,b] = Gerador_Sistema(2000);
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    9.0315923
--> disp(e,k,rk)
    1.331D-08
    8.
    0.0000050
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    1.5215198
--> disp(e,k,rk)
    1.331D-08
    8.
    0.0000050
-->

```

Figure 17: Tempo de execução das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 aplicadas ao sistema gerado para  $n = 2000$

Veja, na figura 18, os tempos, em segundos, envolvidos na aplicação de cada uma das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 ao sistema  $Ax = b$  gerado para  $n = 5000$ , bem como os valores de  $e$ ,  $k$  e  $r_k$ , mostrando que uma solução foi de fato encontrada.

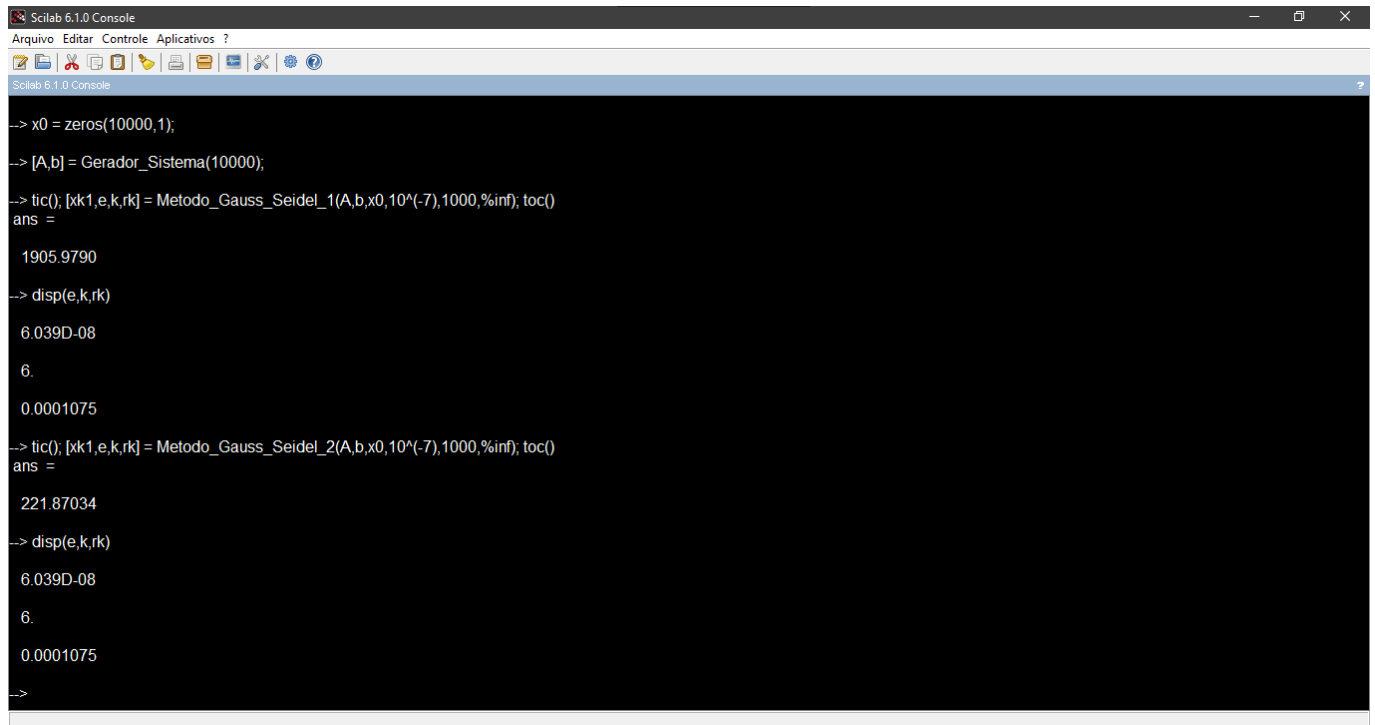
```

Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?
[Icons]
Scilab 6.1.0 Console
--> x0 = zeros(5000,1);
--> [A,b] = Gerador_Sistema(5000);
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf); toc()
ans =
   125.76871
--> disp(e,k,rk)
   4.257D-08
    7.
   0.0000154
--> tic(); [xk1,e,k,rk] = Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf); toc()
ans =
    8.0988666
--> disp(e,k,rk)
   4.257D-08
    7.
   0.0000154
-->

```

Figure 18: Tempo de execução das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 aplicadas ao sistema gerado para  $n = 5000$

Veja, na figura 19, os tempos, em segundos, envolvidos na aplicação de cada uma das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 ao sistema  $Ax = b$  gerado para  $n = 10000$ , bem como os valores de  $e$ ,  $k$  e  $r_k$ , mostrando que uma solução foi de fato encontrada.



```

Scilab 6.1.0 Console
Arquivo  Editar  Controle  Aplicativos ?

--> x0 = zeros(10000,1);
--> [A,b] = Gerador_Sistema(10000);
--> tic(); [pk1,e,k,rk] = Metodo_Gauss_Seidel_1(A,b,x0,10^(-7),1000,%inf), toc()
ans =

1905.9790
--> disp(e,k,rk)

6.039D-08
6.
0.0001075
--> tic(); [pk1,e,k,rk] = Metodo_Gauss_Seidel_2(A,b,x0,10^(-7),1000,%inf), toc()
ans =

221.87034
--> disp(e,k,rk)

6.039D-08
6.
0.0001075
-->

```

Figure 19: Tempo de execução das funções Metodo-Gauss-Seidel-1 e Metodo-Gauss-Seidel-2 aplicadas ao sistema gerado para  $n = 10000$

Comentários: Veja na figura 20 um gráfico contendo os tempos de execução das funções Gauss-Seidel-1 e Gauss-Seidel-2 obtidos nas figuras 14 a 19. Podemos concluir que apesar de em todas as aplicações feitas anteriormente das 2 funções para um mesmo sistema  $Ax = b$  os resultados obtidos serem praticamente idênticos, a diferença crucial entre as funções está no tempo de execução. A função Gauss-Seidel-1 utiliza em seu código o comando nativo do Scilab para calcular matrizes inversas, o que tem um custo computacional muito alto. Por outro lado, na Gauss-Seidel-2, contornamos esse impecilho utilizando a função Resolve-Sistema-Lx, que permitiu resolver os sistemas sem perda de corretude e de forma muito mais eficiente, principalmente para sistemas de dimensão ( $n$ ) muito grande.

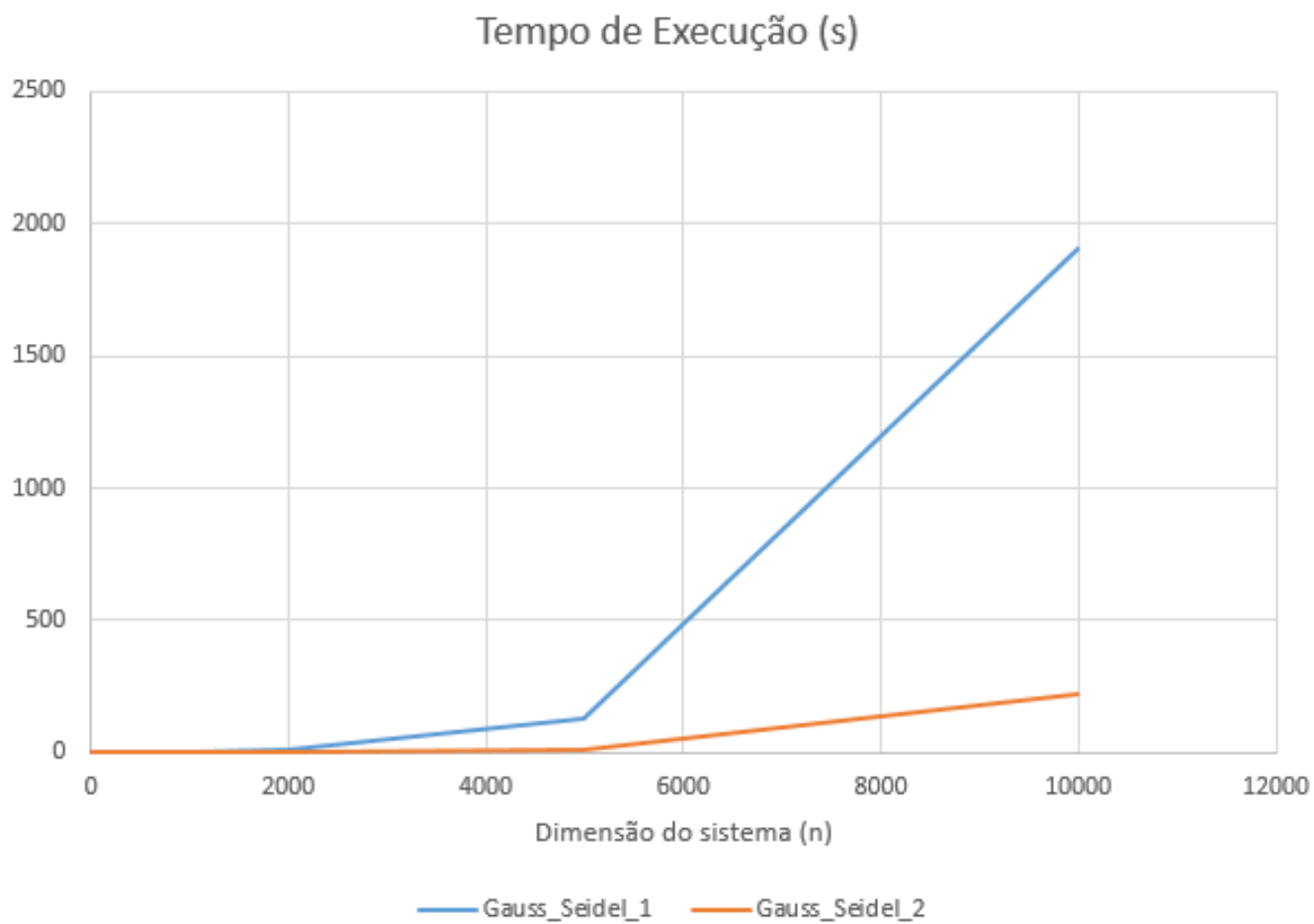


Figure 20: Gráfico contendo os tempos de execução obtidos anteriormente