Atividade 2 - Pedro José Garcia 11846943

1)

1) Temos o sistema tridiagonal $Ax=b$, na forma de matriz:

$$\begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \cdots & 0 \\ 0 & a_{32} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{mm} \end{pmatrix} \cdot \begin{pmatrix} x_{11} \\ x_{21} \\ x_{31} \\ \vdots \\ x_{m1} \end{pmatrix} = \begin{pmatrix} b_{11} \\ b_{21} \\ b_{31} \\ \vdots \\ b_{m1} \end{pmatrix}$$

, aplicando o algoritmo de Thomas, temos que dividir a linha 1 das matrizes A e b pelo pivô, que nesse caso é o elemento a_{11}, então $a'_{11}=\frac{a_{11}}{a_{11}}=1$, $a'_{12}=\frac{a_{12}}{a_{11}}$ e $b'_{11}=\frac{b_{11}}{a_{11}}$, logo as matrizes ficam assim:

$$\begin{pmatrix} 1 & \frac{a_{12}}{a_{11}} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & \cdots & 0 \\ 0 & a_{32} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{mm} \end{pmatrix} \cdot \begin{pmatrix} x_{11} \\ x_{21} \\ x_{31} \\ \vdots \\ x_{m1} \end{pmatrix} = \begin{pmatrix} \frac{b_{11}}{a_{11}} \\ b_{21} \\ b_{31} \\ \vdots \\ b_{m1} \end{pmatrix}$$

, agora vamos eliminar a_{21}, multiplicando a primeira linha das matrizes A e b por a_{21} e subtraída pela segunda linha, então $a'_{21}=a_{21}-1\cdot a_{21}=0$, $a'_{22}=a_{22}-\frac{a_{12}\cdot a_{21}}{a_{11}}$, $a'_{23}=a_{23}-0\cdot a_{21}=+a_{23}$ e $b'_{21}=b_{21}-\frac{b_{11}\cdot a_{21}}{a_{11}}$, logo as matrizes ficam assim:

$$\begin{pmatrix} 1 & \frac{a_{12}}{a_{11}} & 0 & \cdots & 0 \\ 0 & a_{22}-\frac{a_{12}\cdot a_{21}}{a_{11}} & a_{23} & \cdots & 0 \\ 0 & a_{32} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{mm} \end{pmatrix} \cdot \begin{pmatrix} x_{11} \\ x_{21} \\ x_{31} \\ \vdots \\ x_{m1} \end{pmatrix} = \begin{pmatrix} \frac{b_{11}}{a_{11}} \\ b_{21}-\frac{b_{11}\cdot a_{21}}{a_{11}} \\ b_{31} \\ \vdots \\ b_{m1} \end{pmatrix}$$

, logo vamos normalizar a linha 2 dividindo pelo pivô $a_{22}-\frac{a_{12}\cdot a_{21}}{a_{11}}$:

$$\begin{pmatrix} 1 & \frac{a_{12}}{a_{11}} & 0 & \cdots & 0 \\ 0 & 1 & \frac{a_{23}\cdot a_{11}}{a_{22}\cdot a_{11}-a_{12}\cdot a_{21}} & \cdots & 0 \\ 0 & a_{32} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{mm} \end{pmatrix} \cdot \begin{pmatrix} x_{11} \\ x_{21} \\ x_{31} \\ \vdots \\ x_{m1} \end{pmatrix} = \begin{pmatrix} \frac{b_{11}}{a_{11}} \\ \frac{b_{21}\cdot a_{11}-b_{11}\cdot a_{21}}{a_{22}\cdot a_{11}-a_{12}\cdot a_{21}} \\ b_{31} \\ \vdots \\ b_{m1} \end{pmatrix}$$

, repetindo esses passos, conseguimos encontrar um conjunto de equações para o algoritmo de Thomas, da seguinte forma:

$$\begin{cases} a'_{ij}=\frac{a_{ij}}{a_{ii}} & \text{se } i=1 \text{ e } j=2 \text{ até } m \\ a'_{ij}=\frac{a_{ij}}{a_{ii}-a_{i(i-1)}\cdot a'_{i(j-1)}} & \text{se } i=2 \text{ até } m \text{ e } j=2 \text{ até } m \end{cases}$$

$$\begin{cases} b'_{ii}=\frac{b_{ii}}{a_{ii}} & \text{se } i=1 \\ b'_{ij}=\frac{b_{ij}-b'_{(i-1)j}\cdot a_{ij}}{a_{ii}-a'_{ji}\cdot a_{ij}} & \text{se } i=2 \text{ até } m \text{ e } j=1 \text{ até } m \end{cases}$$

após a matriz tridiagonal se tornar uma matriz triangular superior, os valores do vetor x podem ser calculados usando a substituição regressiva, então $x_{m1}=b_{m1}$ e $x_{i1}=b_{i1}-a_{ij}\cdot x_{(i+1)1}$ para $i=m-1$ até 1 e $j=2$ até m, portanto, a parte teórica do algoritmo de Thomas foi demonstrada.

O algoritmo de Thomas é mais eficiente que a eliminação de Gauss clássica, pelo fato de ele usar uma matriz tridiagonal, ou seja, o processo é realizado na diagonal principal e nas diagonais adjacentes a ela somente, enquanto que na eliminação de Gauss clássica o processo é realizado na matriz inteira, portanto, o algoritmo de Thomas consegue ser mais eficiente do que a eliminação de Gauss clássica.

3)

Tempos para cada método em segundos

Algoritmo 1

Dimensão	1	2	3	4	5	Média
64	0.000633	0.000605	0.000314	0.001785	0.000143	0.000696
128	0.000824	0.000850	0.000842	0.000868	0.001702	0.001017
256	0.004139	0.000781	0.001659	0.000815	0.002639	0.002006
512	0.006707	0.006264	0.007213	0.007077	0.003293	0.006110
1024	0.012926	0.025982	0.011569	0.019740	0.025155	0.019074
2048	0.160742	0.080839	0.073761	0.170700	0.076173	0.112443

Algoritmo 2

Dimensão	1	2	3	4	5	Média
64	0.007266	0.001966	0.001247	0.000417	0.001407	0.002460
128	0.012246	0.005697	0.002800	0.001837	0.003588	0.005233
256	0.011654	0.008101	0.002231	0.003542	0.006639	0.006433
512	0.014337	0.007731	0.008274	0.006508	0.017256	0.010821
1024	0.059469	0.026942	0.029442	0.032358	0.027630	0.035168
2048	0.226598	0.295540	0.247610	0.241016	0.241207	0.162441

Algoritmo 3

Dimensão	1	2	3	4	5	Média
64	0.001620	0.001437	0.000253	0.000251	0.001028	0.000917
128	0.002566	0.001351	0.000563	0.000961	0.000720	0.001232
256	0.002469	0.001870	0.001053	0.001599	0.001484	0.001695
512	0.004587	0.003376	0.003299	0.004352	0.002985	0.003719
1024	0.008863	0.009376	0.007620	0.008509	0.008660	0.008605
2048	0.052450	0.051610	0.051604	0.053658	0.055831	0.053030

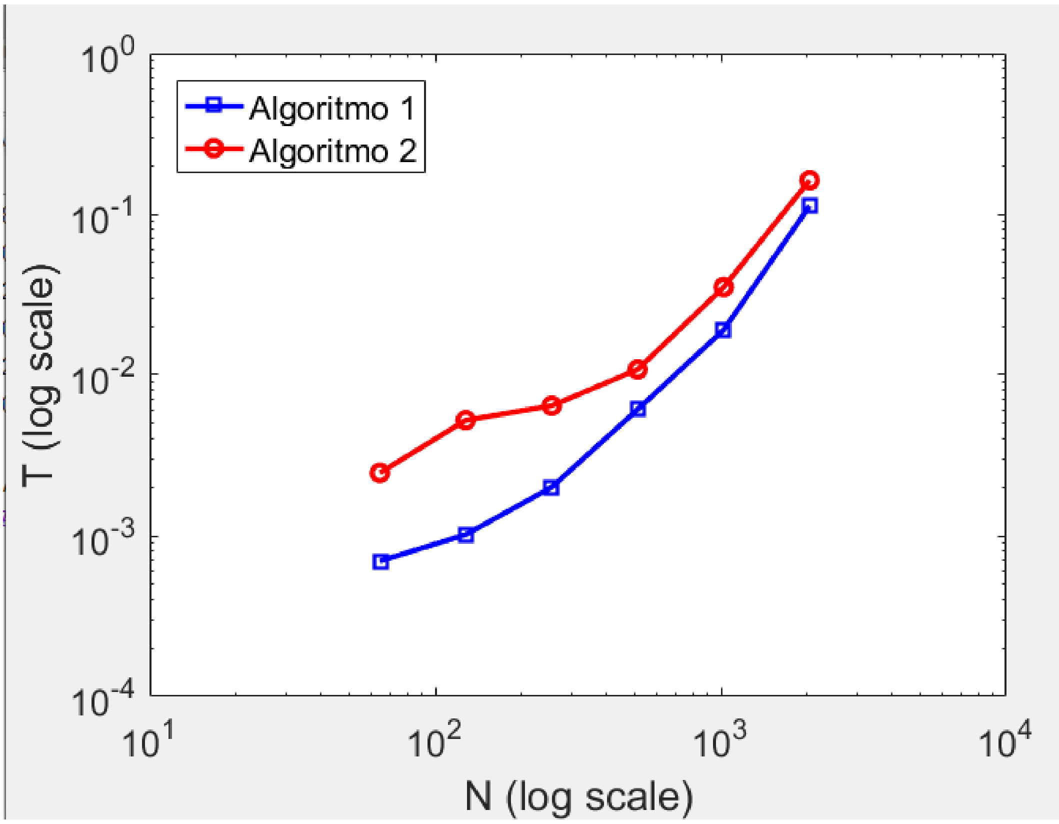
Algoritmo 4

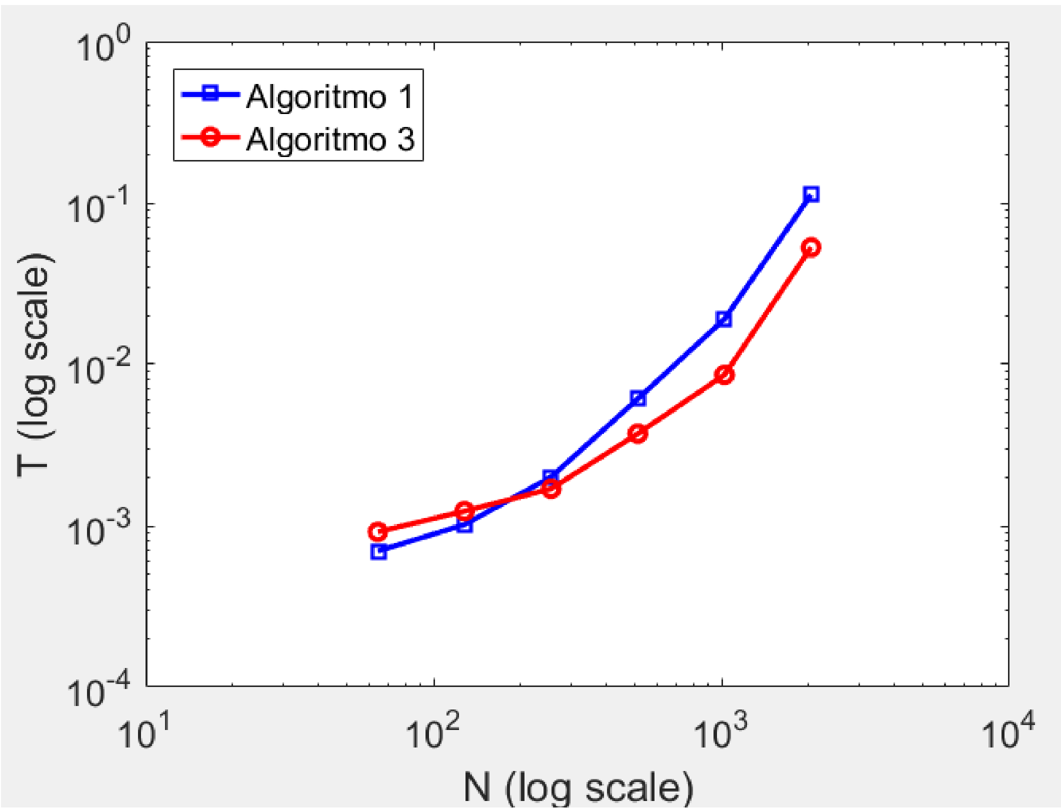
Dimensão	1	2	3	4	5	Média
64	0.008365	0.001823	0.001154	0.001218	0.001194	0.002750
128	0.010274	0.005426	0.004949	0.004066	0.004905	0.005924
256	0.031110	0.027306	0.025999	0.027089	0.027580	0.027816
512	0.353171	0.354622	0.357822	0.391379	0.363820	0.364162
1024	3.776463	3.736921	3.811951	3.818809	3.777163	3.784261
2048	31.775395	31.875705	31.972255	31.747490	32.021431	31.878455

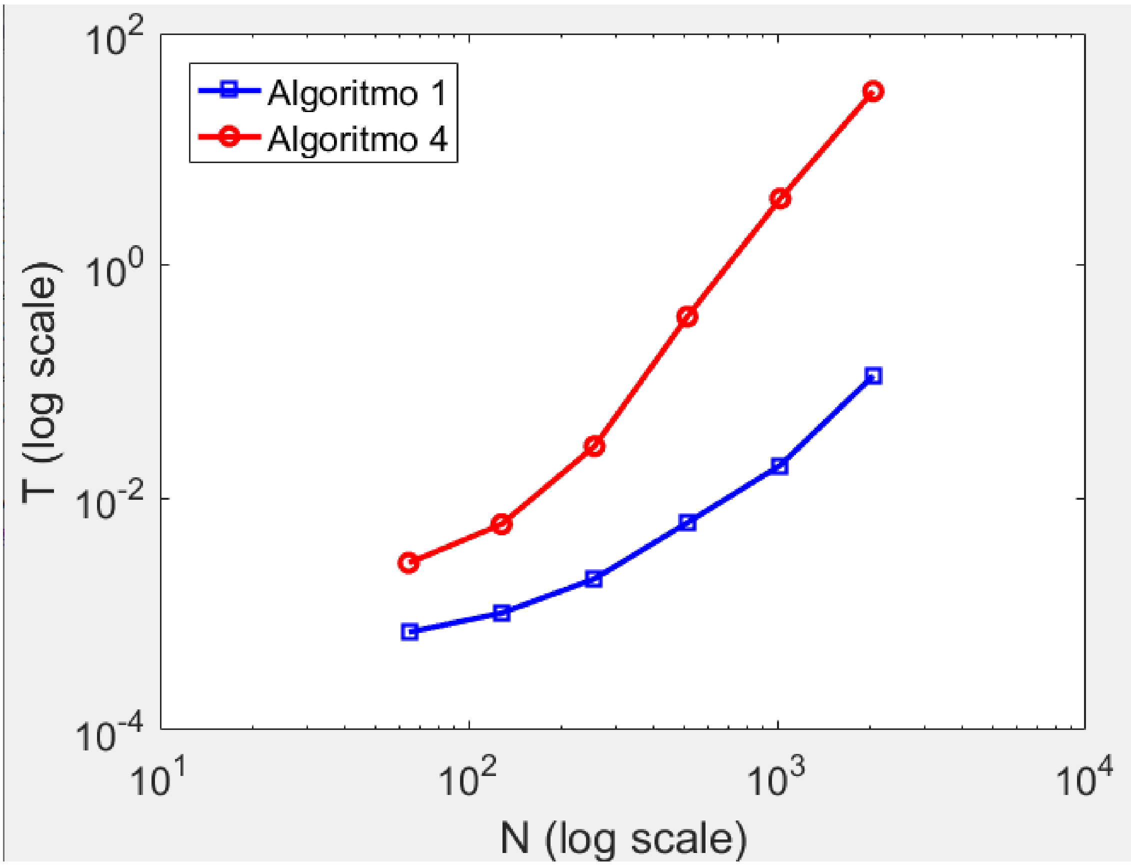
Algoritmo 5

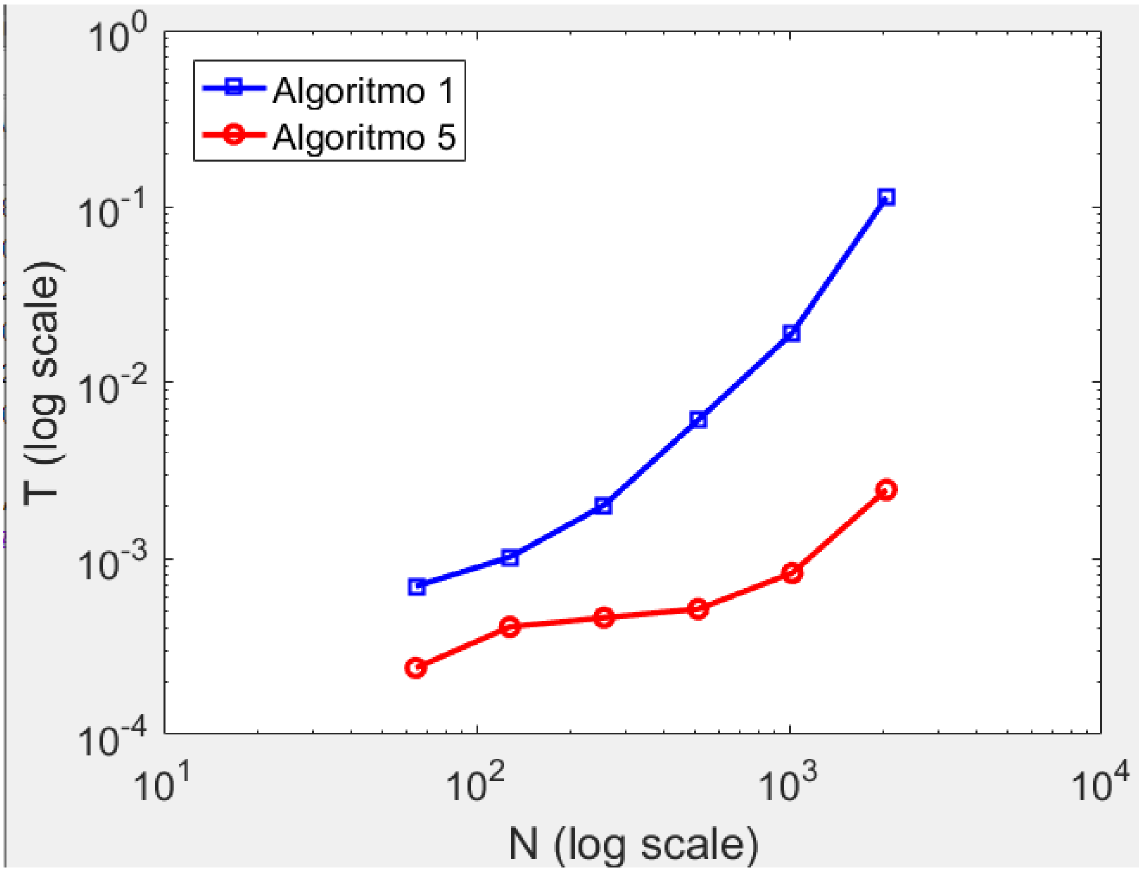
Dimensão	1	2	3	4	5	Média
64	0.000308	0.000244	0.000397	0.000175	0.000071	0.000239
128	0.000441	0.000351	0.000464	0.000319	0.000472	0.000409
256	0.000543	0.000408	0.000399	0.000322	0.000635	0.000461
512	0.000527	0.000643	0.000501	0.000309	0.000604	0.000516
1024	0.000720	0.001242	0.000718	0.000641	0.000821	0.000828
2048	0.007571	0.002471	0.000477	0.001183	0.000599	0.002460

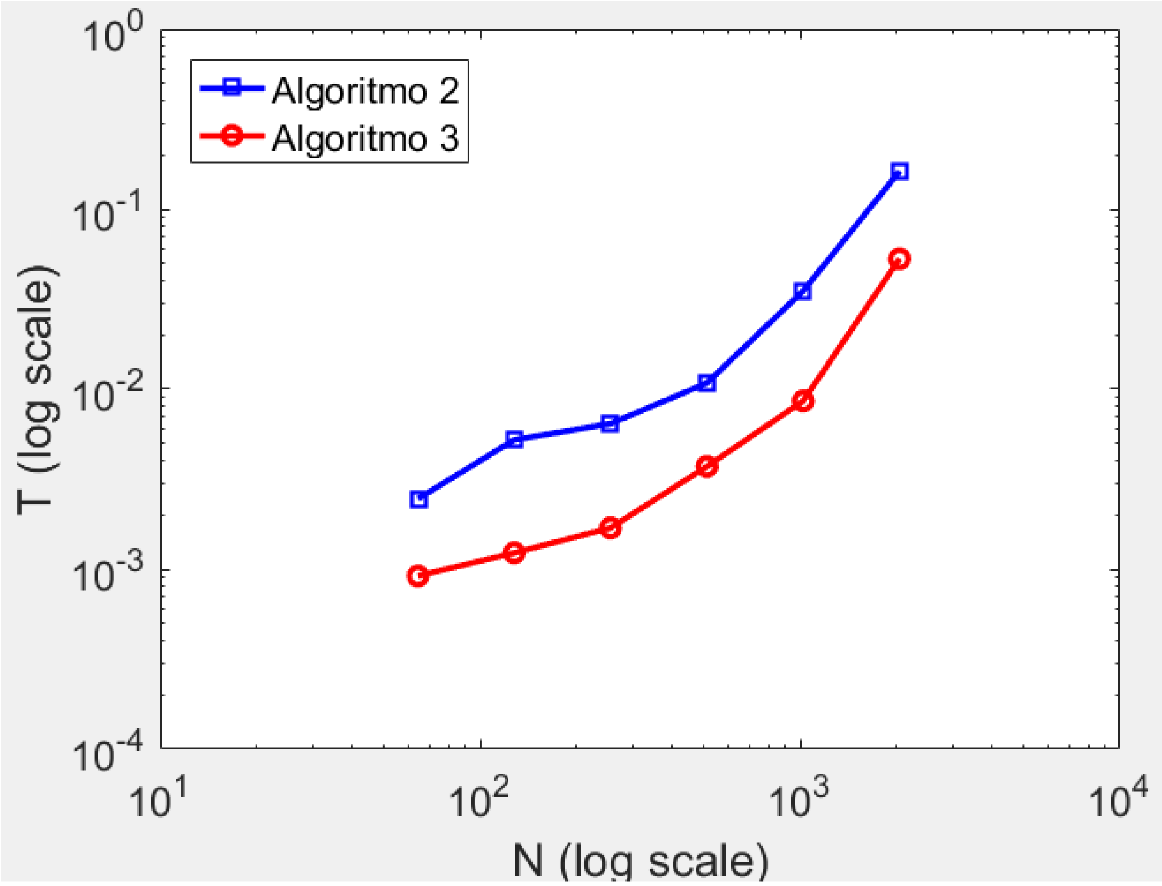
Gráficos dimensão n x tempo



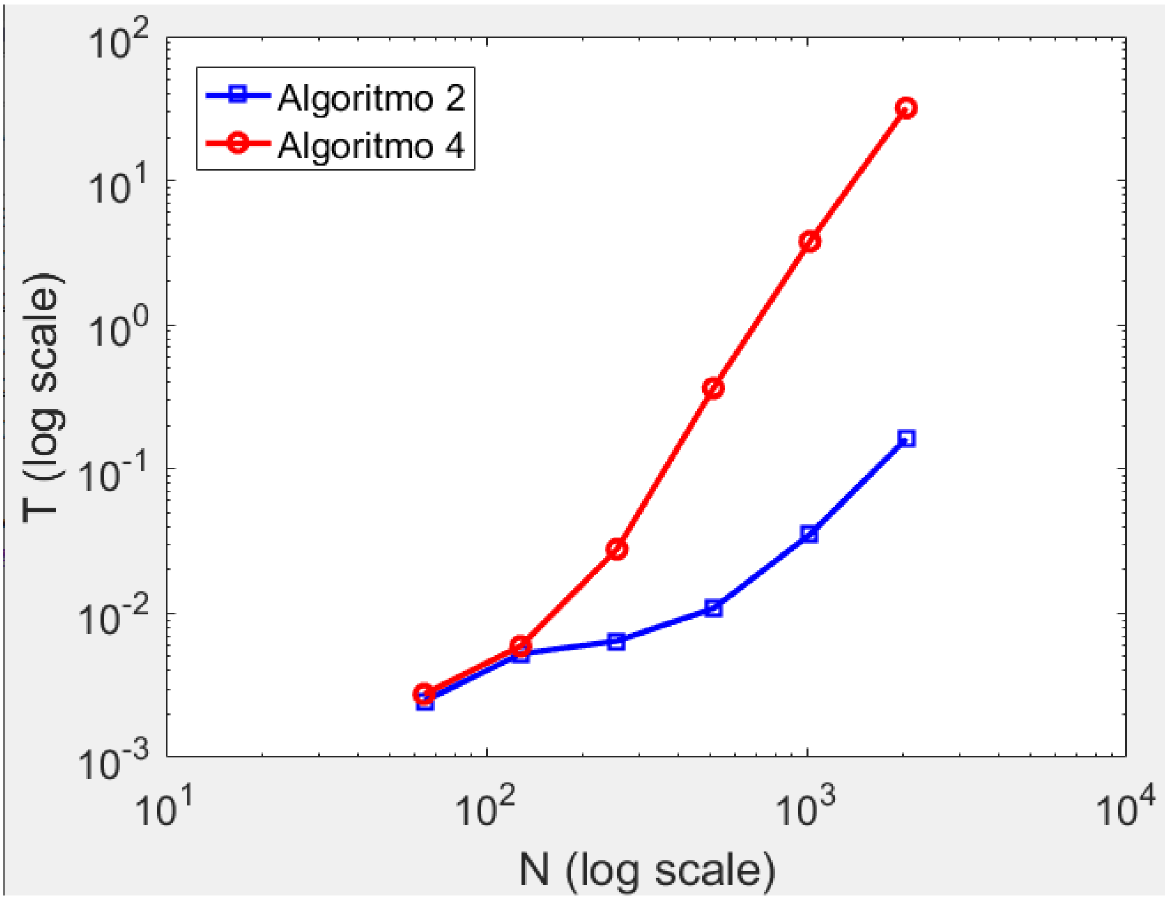



Algoritmo 1
Algoritmo 4
T (log scale)
N (log scale)
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{1}
10^{2}
10^{3}
10^{4}

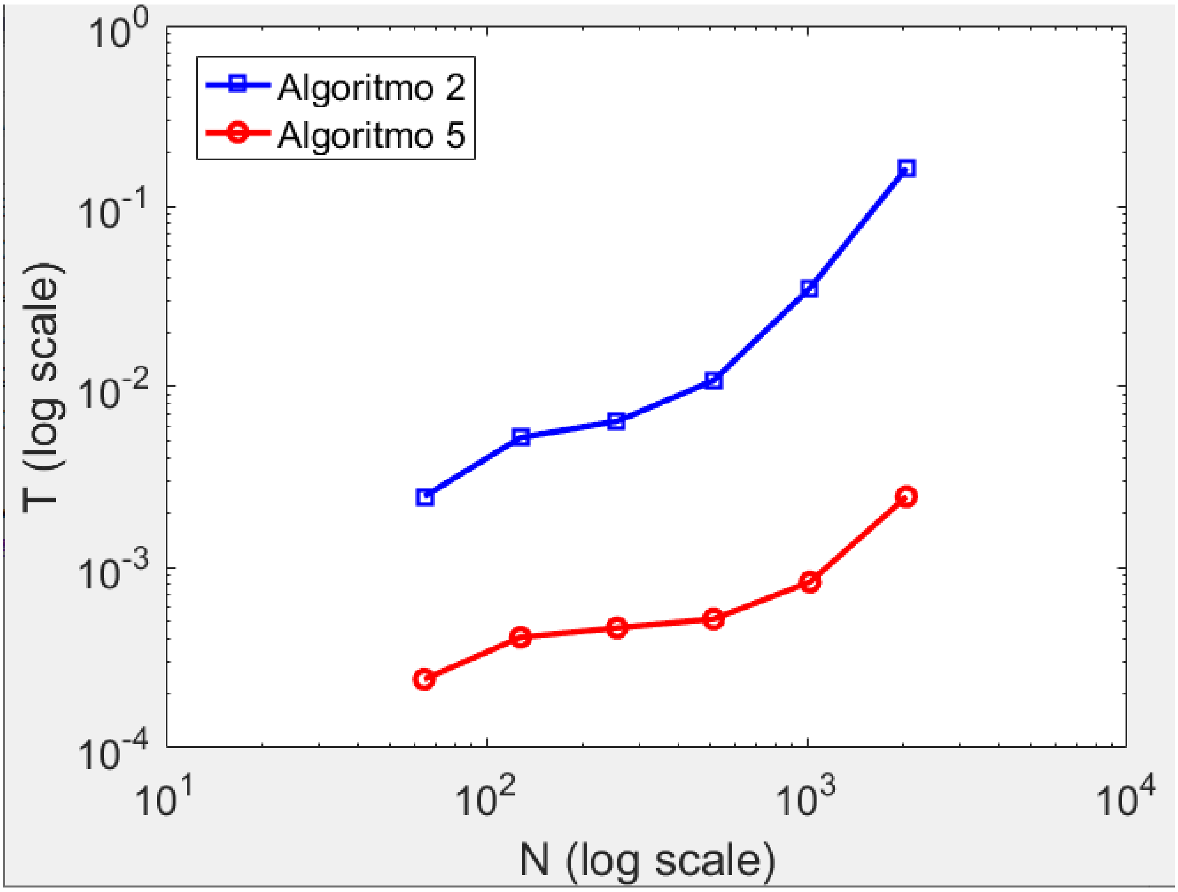

Algoritmo 1
Algoritmo 5
T (log scale)
N (log scale)
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{1}
10^{2}
10^{3}
10^{4}

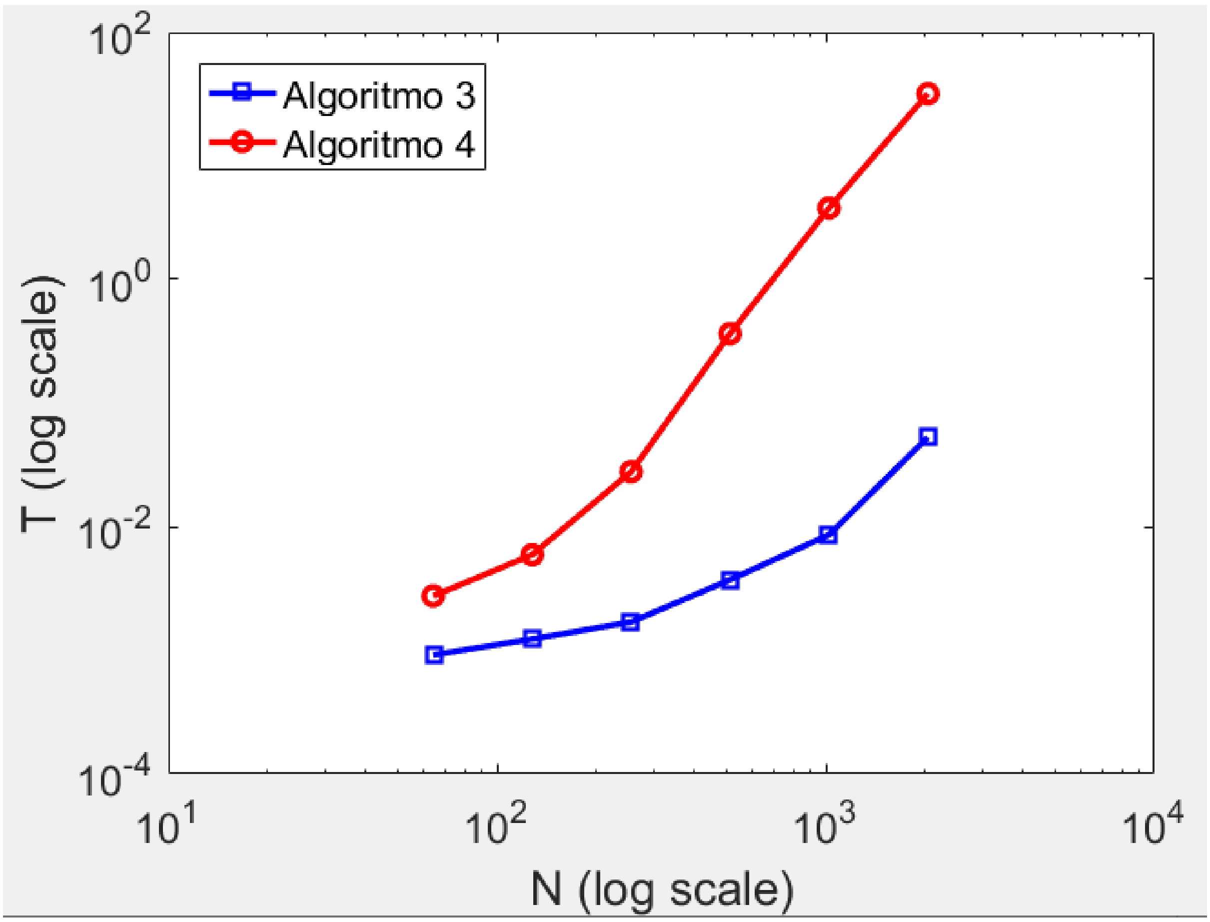


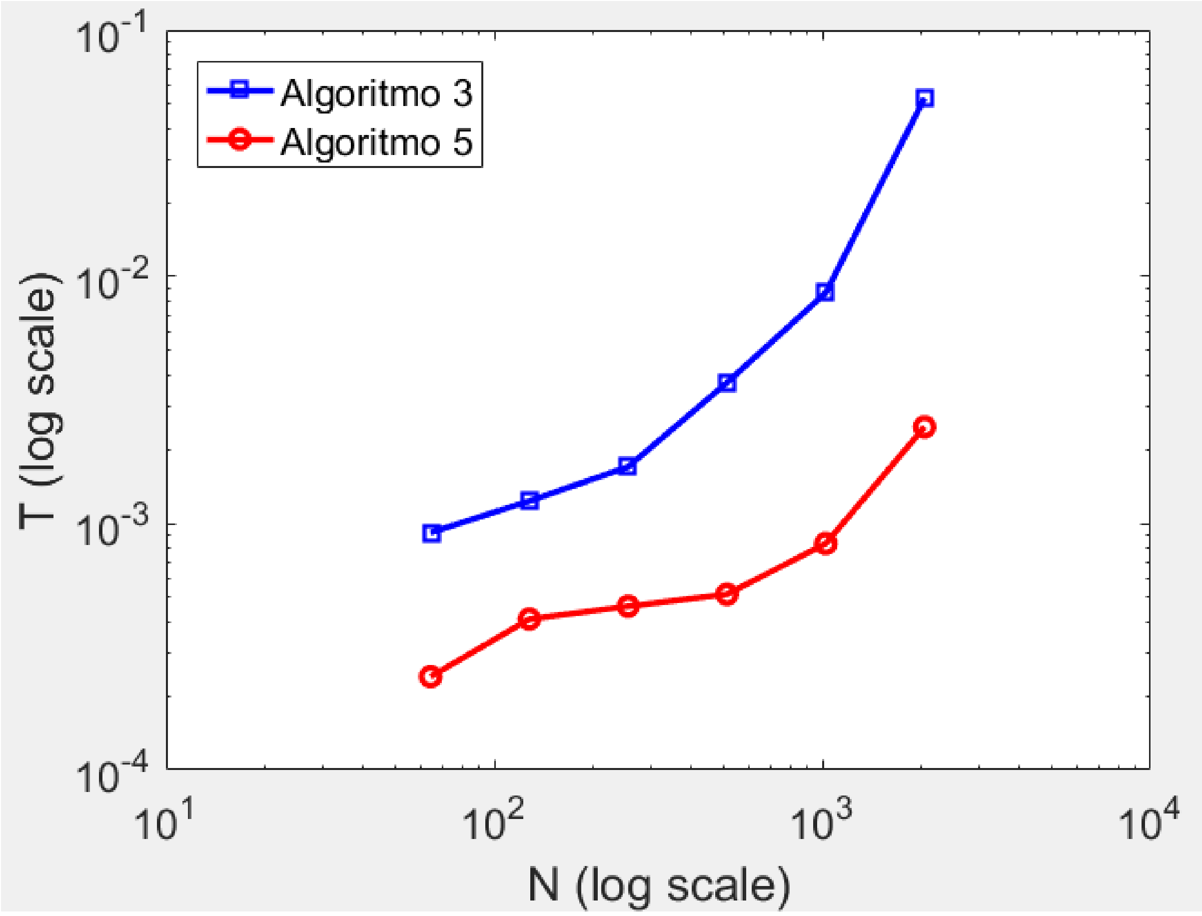
Algoritmo 2
Algoritmo 3
T (log scale)
N (log scale)
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^1
10^2
10^3
10^4

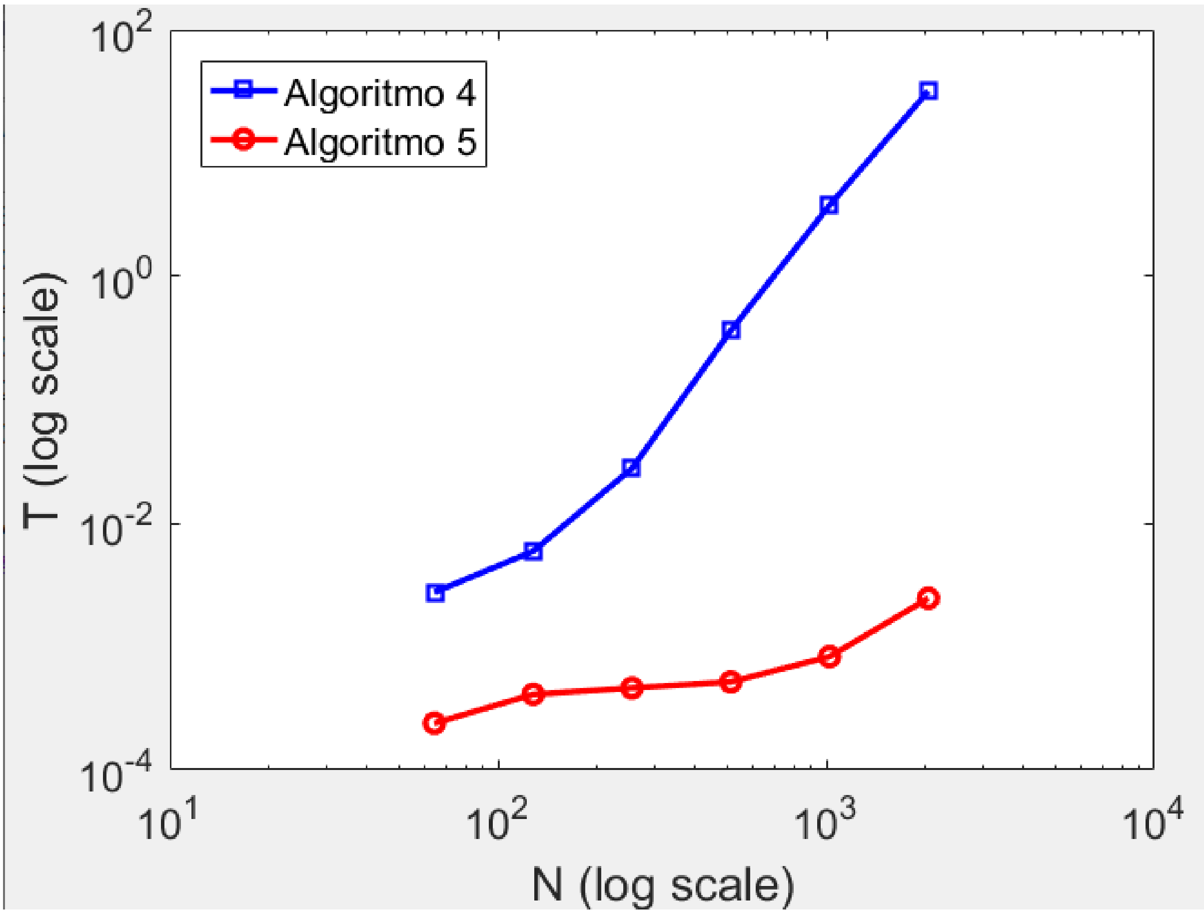


Algoritmo 2
Algoritmo 4
T (log scale)
N (log scale)
10^2
10^1
10^0
10^{-1}
10^{-2}
10^{-3}
10^1
10^2
10^3
10^4


Algoritmo 2
Algoritmo 5
T (log scale)
N (log scale)
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{1}
10^{2}
10^{3}
10^{4}


Algoritmo 3
Algoritmo 4
T (log scale)
N (log scale)
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{1}
10^{2}
10^{3}
10^{4}


Algoritmo 3
Algoritmo 5
T (log scale)
N (log scale)
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{1}
10^{2}
10^{3}
10^{4}


Algoritmo 4
Algoritmo 5
T (log scale)
N (log scale)
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{1}
10^{2}
10^{3}
10^{4}

Ordem de custo

Algoritmo 1: 1.6827 **Algoritmo 2:** 1.2363 **Algoritmo 3:** 1.3199
Algoritmo 4: 3.1875 **Algoritmo 5:** 0.6022

Observando os gráficos e a ordem de custo, percebe-se que o algoritmo 5 é o menos custoso e o mais rápido entre esses algoritmos, e que o algoritmo 4 é o mais lento e mais custoso. Além disso, apesar do algoritmo 1 ser mais rápido que o 2, ele é mais custoso, assim como o 3 é mais rápido que o 2 e mais custoso que ele também, outra relação que pode ser observada é que o algoritmo 3 é mais rápido que o algoritmo 1 nas dimensões 64 e 128, contudo nas dimensões maiores que a 128 o algoritmo 1 se torna mais rápido.

A ordem teórica é a logarítmica (log n), porque o algoritmo de Thomas não precisa percorrer a matriz inteira, ou seja, ele é inferior a ordem linear e não pode ser constante porque os tempos observados não são constantes, portanto, a ordem teórica é a logarítmica. Do mesmo modo, a ordem observada é a logarítmica também, haja vista que os tempos obtidos não são lineares e não são constantes.

Para calcular o tempo estimado, acredito que seja usar a fórmula tempo = dimensão elevada à ordem de custo (T = N^p)

O tempo estimado para cada método resolver um sistema com 1 milhão de incógnitas seria:
Algoritmo 1: 395.72 anos
Algoritmo 2: 0.829 anos
Algoritmo 3: 2.634 anos
Algoritmo 4: 4.23 x 10^11 anos
Algoritmo 5: 1.14 dias

O tempo estimado para cada método resolver um sistema com 1 bilhão de incógnitas seria:
Algoritmo 1: 4.42 x 10^8 anos
Algoritmo 2: 4245 anos
Algoritmo 3: 24004.5 anos
Algoritmo 4: 1.54 x 10^21 anos
Algoritmo 5: 3.04 dias