

Universidade de São Paulo

Instituto de Física de São Carlos

Lista 4

Pedro Calligaris Delbem 5255417

Professor: Attilio Cucchieri

Maio de 2025

Sumário

1	Matrix Operations	2
1.1	Exercício 1	2
1.2	Exercício 2	4

1 Matrix Operations

1.1 Exercício 1

Tarefa: Calcule todos os autovalores da matriz simétrica e tridiagonal A , definida por:

$$A_{mm} = -2, \quad A_{m,m-1} = A_{m-1,m} = 1$$

a qual está relacionada à discretização da derivada segunda em uma dimensão:

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}.$$

Use as seguintes relações de recorrência para o polinômio característico $P_n(\lambda) = \det(A - \lambda I)$:

$$\begin{aligned} P_1(\lambda) &= A_{11} - \lambda, \\ P_2(\lambda) &= (A_{22} - \lambda)P_1(\lambda) - A_{21}A_{12}, \\ P_3(\lambda) &= (A_{33} - \lambda)P_2(\lambda) - A_{32}A_{23}P_1(\lambda), \\ &\vdots \\ P_m(\lambda) &= (A_{mm} - \lambda)P_{m-1}(\lambda) - A_{m,m-1}A_{m-1,m}P_{m-2}(\lambda), \\ &\vdots \\ P_n(\lambda) &= (A_{nn} - \lambda)P_{n-1}(\lambda) - A_{n,n-1}A_{n-1,n}P_{n-2}(\lambda). \end{aligned}$$

Use o método de Newton-Raphson para calcular numericamente as raízes de $P_n(\lambda)$, ou seja, os autovalores da matriz A .

Sugestão: escreva uma relação de recorrência também para as derivadas primeira e segunda, em relação à variável λ , das funções $P_m(\lambda)$. Considere o caso $n \times n$, onde n é uma entrada (input) do código.

Compare os resultados numéricos com o resultado exato:

$$\lambda_m = -4 \sin^2 \left(\frac{m\pi}{2(n+1)} \right), \quad m = 1, 2, \dots, n.$$

Sugestão: use as desigualdades

$$\max_{j=1,\dots,n} \left(A_{jj} + \sum_{k \neq j} |A_{jk}| \right) \geq \lambda_m \geq \min_{j=1,\dots,n} \left(A_{jj} - \sum_{k \neq j} |A_{jk}| \right)$$

para organizar a busca inicial das raízes de $P_n(\lambda)$.

Derivando a relação de recorrência com relação a lambda, obtemos:

$$P'_m(\lambda) = (A_{mm} - \lambda)P'_{m-1}(\lambda) - A_{m,m-1}A_{m-1,m}P'_{m-2}(\lambda) - P_{m-1}(\lambda)$$

Para determinar o chute inicial do Newton-Raphson, temos:

$$\begin{aligned} A_{jj} &= -2, \\ \sum_{k \neq j} |A_{jk}| &= |A_{j,j-1}| + |A_{j,j+1}| = 1 + 1 = 2, \quad \text{para } j = 2, \dots, n-1, \\ \sum_{k \neq j} |A_{jk}| &= 1, \quad \text{para } j = 1 \text{ ou } j = n, \end{aligned}$$

E, utilizando a desigualdade dada, temos:

$$-4 \geq \lambda_{\text{máx}} \geq 0$$

Logo, é razoável começar com o chute inicial entre 0 e -4.

O código foi compilado com o comando:

gfortran L4-5255417-ex-1.f90 -Wall -Wextra -pedantic -ffree-form -o L4-5255417-ex-1.exe

Resultados:

Testou-se os seguintes casos:

```
pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticomp/lista4
> $ ./L4-5255417-ex-1.exe
Insert matrix dimension:
2
Insert tolerance
1.0e-8
Mean Error: 0.000000000000 Max Error: 0.000000000000 +/- 1.00E-08

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticomp/lista4
> $ ./L4-5255417-ex-1.exe
Insert matrix dimension:
10
Insert tolerance
1.0e-8
Mean Error: 0.000000000227 Max Error: 0.00000001058 +/- 1.00E-08

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticomp/lista4
> $ ./L4-5255417-ex-1.exe
Insert matrix dimension:
100
Insert tolerance
1.0e-8
Mean Error: 0.000000000010 Max Error: 0.00000000112 +/- 1.00E-08

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticomp/lista4
> $ ./L4-5255417-ex-1.exe
Insert matrix dimension:
1000
Insert tolerance
1.0e-8
Mean Error: 0.000000000001 Max Error: 0.000000000006 +/- 1.00E-08
```

Figura 1: Dimensão da Matriz = 1000

Percebe-se que a diferença - tanto média quanto para o maior caso - se manteve dentro da tolerância em todos os casos mostrando a eficiência do método empregado.

1.2 Exercício 2

Tarefa: Considere a mesma matriz A do exercício anterior e o vetor \vec{d} com elementos $d_i = \delta_{i,j}$, onde j é escolhido aleatoriamente entre os valores $1, 2, \dots, n$. Deseja-se resolver numericamente o sistema de equações lineares $A\vec{x} = \vec{d}$, utilizando o **algoritmo de Thomas**, específico para matrizes tridiagonais.

O algoritmo consiste em duas fases: uma substituição direta modificada para calcular coeficientes intermediários d'_m e c'_m , e uma substituição retroativa para obter a solução final \vec{x} . Abaixo estão as fórmulas utilizadas:

$$\begin{aligned}d'_1 &= \frac{d_1}{b_1}, & c'_1 &= \frac{c_1}{b_1}, \\d'_m &= \frac{d_m - a_m d'_{m-1}}{b_m - a_m c'_{m-1}}, & c'_m &= \frac{c_m}{b_m - a_m c'_{m-1}}, \quad \text{para } m = 2, 3, \dots, n-1, \\d'_n &= \frac{d_n - a_n d'_{n-1}}{b_n - a_n c'_{n-1}}.\end{aligned}$$

A solução \vec{x} é então obtida pela substituição retroativa:

$$\begin{aligned}x_n &= d'_n, \\x_m &= d'_m - c'_m x_{m+1}, \quad \text{para } m = n-1, n-2, \dots, 1.\end{aligned}$$

Considere o caso de uma matriz tridiagonal de dimensão $n \times n$, com n sendo um input do código. Após resolver o sistema, verifique a solução calculando o vetor resíduo:

$$\vec{r} = A\vec{x} - \vec{d}.$$

Esse vetor deve ser próximo de zero (em norma) se a solução numérica estiver correta.

O código foi compilado com o comando:

```
gfortran L4-5255417-ex-2.f90 -Wall -Wextra -pedantic -ffree-form -o L4-
```

Resultados:

Testou-se os seguintes casos:

```

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/lista4
> $ ./L4-5255417-ex-2.exe
Insert matrix dimension:
8
The solution x of Ax = d (with j = 1) and the residue vector are:
1      -0.888889      0.000000
2      -0.777778      0.000000
3      -0.666667      0.000000
4      -0.555556      0.000000
5      -0.444444      0.000000
6      -0.333333      0.000000
7      -0.222222      0.000000
8      -0.111111      0.000000

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/lista4
> $ ./L4-5255417-ex-2.exe
Insert matrix dimension:
8
The solution x of Ax = d (with j = 5) and the residue vector are:
1      -0.444444      0.000000
2      -0.888889      0.000000
3      -1.333333      0.000000
4      -1.777778      0.000000
5      -2.222222      0.000000
6      -1.666667      0.000000
7      -1.111111      0.000000
8      -0.555556      0.000000

```

Figura 2: Testes para matriz de dimensão 8

```

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/lista4
> $ ./L4-5255417-ex-2.exe
Insert matrix dimension:
10
The solution x of  $Ax = d$  (with  $j = 8$ ) and the residue vector are:
 1   -0.272727   0.000000
 2   -0.545455   0.000000
 3   -0.818182   0.000000
 4   -1.090909   0.000000
 5   -1.363636   0.000000
 6   -1.636364   0.000000
 7   -1.909091   0.000000
 8   -2.181818   0.000000
 9   -1.454545   0.000000
10   -0.727273   0.000000

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/lista4
> $ ./L4-5255417-ex-2.exe
Insert matrix dimension:
10
The solution x of  $Ax = d$  (with  $j = 2$ ) and the residue vector are:
 1   -0.818182   0.000000
 2   -1.636364   0.000000
 3   -1.454545   0.000000
 4   -1.272727   0.000000
 5   -1.090909   0.000000
 6   -0.909091   0.000000
 7   -0.727273   0.000000
 8   -0.545455   0.000000
 9   -0.363636   0.000000
10   -0.181818   0.000000

```

Figura 3: Testes para matriz de dimensão 10

```

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/lista4
> $ ./L4-5255417-ex-2.exe
Insert matrix dimension:
20
The solution x of  $Ax = d$  (with  $j = 3$ ) and the residue vector are:
 1  -0.857143  0.000000
 2  -1.714286  0.000000
 3  -2.571429  0.000000
 4  -2.428571  0.000000
 5  -2.285714  0.000000
 6  -2.142857  0.000000
 7  -2.000000  0.000000
 8  -1.857143  0.000000
 9  -1.714286  0.000000
10  -1.571429  0.000000
11  -1.428571  0.000000
12  -1.285714  0.000000
13  -1.142857  0.000000
14  -1.000000  0.000000
15  -0.857143  0.000000
16  -0.714286  0.000000
17  -0.571429  0.000000
18  -0.428571  0.000000
19  -0.285714  0.000000
20  -0.142857  0.000000

```

(a)

```

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/lista4
> $ ./L4-5255417-ex-2.exe
Insert matrix dimension:
20
The solution x of  $Ax = d$  (with  $j = 15$ ) and the residue vector are:
 1  -0.285714  0.000000
 2  -0.571429  0.000000
 3  -0.857143  0.000000
 4  -1.142857  0.000000
 5  -1.428571  0.000000
 6  -1.714286  0.000000
 7  -2.000000  0.000000
 8  -2.285714  0.000000
 9  -2.571429  0.000000
10  -2.857143  0.000000
11  -3.142857  0.000000
12  -3.428571  0.000000
13  -3.714286  0.000000
14  -4.000000  0.000000
15  -4.285714  0.000000
16  -3.571429  0.000000
17  -2.857143  0.000000
18  -2.142857  0.000000
19  -1.428571  0.000000
20  -0.714286  0.000000

```

(b)

Figura 4: Teste para matriz de dimensão 20

Como todos os vetores resíduo obtidos foram nulos, sabemos que o código funciona de acordo com o esperado - obtendo o autovetor x corretamente.