

# Universidade de São Paulo

## Instituto de Física de São Carlos

### **Lista 5**

Pedro Calligaris Delbem 5255417

Professor: Attilio Cucchieri

Junho de 2025

# Sumário

1	Exercício 1	2
2	Exercício 2	3
3	Exercício 3	4
4	Exercício 4	9

# 1 Exercício 1

Tarefa: Na lista 5, foi considerado o poço de potencial infinito no intervalo  $[0, L]$ , para uma partícula de massa  $m$ , ou seja, a equação

$$-\frac{\hbar^2}{2m}\nabla^2\psi_j(x) = E_j\psi_j(x). \quad (1)$$

Para encontrar as autofunções

$$\psi_j(x) \propto \sin\left(\frac{j\pi x}{L}\right)$$

foi considerada a matriz

$$\begin{pmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \cdots & 0 \\ 0 & 1 & -2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & -2 \end{pmatrix}$$

Agora, considere a matriz

$$\begin{pmatrix} -1 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \cdots & 0 \\ 0 & 1 & -2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & -2 \end{pmatrix} \quad (2)$$

Que tipo de soluções para o poço de potencial infinito você espera encontrar nesse caso? Motive sua resposta.

## Resposta:

Para um ponto no extremo inicial da grade, a discretização de diferenças finitas para a primeira derivada é:

$$\left.\frac{d\psi}{dx}\right|_{x=0} \approx \frac{\psi_1 - \psi_0}{h} \quad (3)$$

A Condição de Contorno de Neumann é:

$$\left.\frac{d\psi}{dx}\right|_{x=0} = 0 \quad (4)$$

O que nos leva a

$$\psi_0 = \psi_1 \quad (5)$$

Substituindo isto na discretização de diferenças finitas para a segunda derivada - no ponto  $x_i$ , temos:

$$\left.\frac{d^2\psi}{dx^2}\right|_{x=x_1} \approx \frac{\psi_2 - 2\psi_1 + (\psi_1)}{h^2} = \frac{\psi_2 - \psi_1}{h^2} \quad (6)$$

Que resulta em dois termos, justamente com os coeficientes -1 e 1, que temos na matriz dada. Logo, espera-se que a solução seja tal que obedeça a condição de contorno de Neumann e portanto o resultado será:

$$\psi_j(x) \propto \cos\left(\frac{j\pi x}{2L}\right)$$

## 2 Exercício 2

Tarefa: Usando o *power method* e a matriz (2), calcule a energia do estado fundamental  $E_0$  com precisão de  $10^{-4}$ . Compare o resultado com o valor exato. Faça um gráfico da autofunção normalizada e compare com a solução exata.

O código foi compilado com o comando:

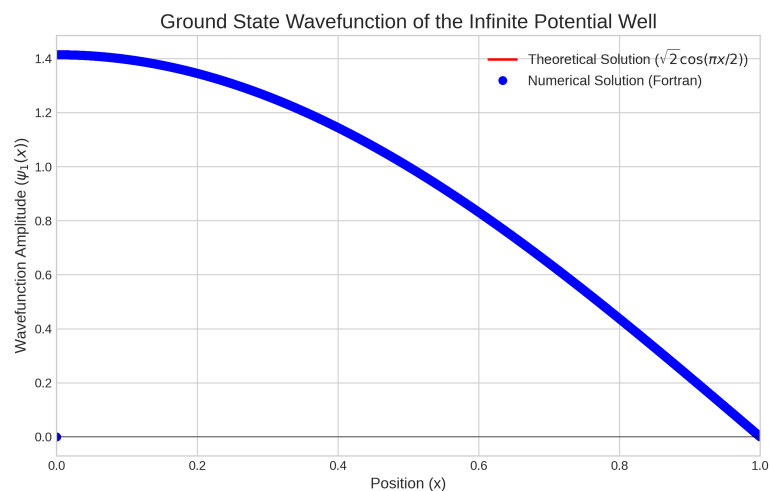
```
gfortran -ffree-form -ffree-line-length-none P2-5255417-ex-2.f90 -Wall -Wextra -pedantic -o P2-5255417-ex-2.exe
```

Resultados:

Utilizou-se  $N=100000$  como se fosse  $N=\infty$ , obtendo a energia:

```
pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/projeto2
> $ ./P2-5255417-ex-2.exe
Insert matrix dimension:
100000
Insert tolerance
1.0e-4
computed E0:    2.4674258 real E0:    2.4674011
```

E obteve-se a seguinte autofunção:



Vê-se que a função analítica e a numérica se sobrepõe totalmente, de modo que o método numérico corresponde perfeitamente à função analítica conhecida. (O autovetor resultante foi salvo no arquivo "P2-5255417-ex-2-results.txt")

### 3 Exercício 3

Tarefa: Considere o método de Householder, estudado na lista 6. Naquele caso, para os elementos fora da diagonal  $k_i$ , foi usado o sinal oposto de  $a_{i-1,i}$ . O que acontece quando o sinal de  $k_i$  é o mesmo de  $a_{i-1,i}$ ? Estude o problema para a mesma matriz considerada na lista 6, ou seja

$$A = \begin{pmatrix} -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} & 0 & 0 & 0 \\ \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} & 0 & 0 \\ -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} & 0 \\ 0 & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} & -\frac{1}{12} \\ 0 & 0 & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} & \frac{4}{3} \\ 0 & 0 & 0 & -\frac{1}{12} & \frac{4}{3} & -\frac{5}{2} \end{pmatrix}$$

Qual é a relação entre a matriz tridiagonal obtida neste caso e a matriz tridiagonal obtida na lista 6?

Resultados:

Obteve-se os seguintes resultados:

```

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/projeto2
> $ ./P2-5255417-ex-3a.exe
Insert matrix dimension:
10

-----> N = 10 <-----
Original Matrix A (first 5x5):
-2.500000    1.333333   -0.083333    0.000000    0.000000
 1.333333   -2.500000    1.333333   -0.083333    0.000000
-0.083333    1.333333   -2.500000    1.333333   -0.083333
 0.000000   -0.083333    1.333333   -2.500000    1.333333
 0.000000    0.000000   -0.083333    1.333333   -2.500000
Tridiagonal Matrix T (first 5x5):
-2.500000   -1.335935    0.000000    0.000000    0.000000
-1.335935   -2.666018    1.333384    0.000000    0.000000
 0.000000    1.333384   -2.666649    1.333335   -0.000000
 0.000000    0.000000    1.333335   -2.666666    1.333333
 0.000000    0.000000   -0.000000    1.333333   -2.666667

--- Transformation Verification  $O^T A O = A_t$  ---
Difference norm  $||A_t - O^T A O||$ : 0.38752E-14

Smallest eigenvalue (lambda_min): -0.08384774
Largest eigenvalue (lambda_max): -5.20135675

--- Verification for Eigenvalue smallest ---
Eigenvalue: -0.08384774
Norm of residual  $||Ay - \lambda y||$ : 0.21547E-07

--- Verification for Eigenvalue biggest ---
Eigenvalue: -5.20135675
Norm of residual  $||Ay - \lambda y||$ : 0.15036E-05

```

```

> $ ./P2-5255417-ex-3b.exe
Insert matrix dimension:
10

-----> N = 10 <-----
Original Matrix A (first 5x5):
-2.500000    1.333333   -0.083333    0.000000    0.000000
 1.333333   -2.500000    1.333333   -0.083333    0.000000
-0.083333    1.333333   -2.500000    1.333333   -0.083333
 0.000000   -0.083333    1.333333   -2.500000    1.333333
 0.000000    0.000000   -0.083333    1.333333   -2.500000
Tridiagonal Matrix T (first 5x5):
-2.500000    1.335935   -0.000000   -0.000000   -0.000000
 1.335935   -2.666018   -1.333384   -0.000000   -0.000000
-0.000000   -1.333384   -2.666649    1.333335   -0.000000
-0.000000   -0.000000    1.333335   -2.666666    1.333333
-0.000000   -0.000000   -0.000000    1.333333   -2.666667

--- Transformation Verification  $O^T A O = A_t$  ---
Difference norm  $||A_t - O^T A O||$ : 0.67461E-14

Smallest eigenvalue (lambda_min): -0.08384774
Largest eigenvalue (lambda_max): -5.20135675

--- Verification for Eigenvalue smallest ---
Eigenvalue: -0.08384774
Norm of residual  $||Ay - \lambda y||$ : 0.17713E-07

--- Verification for Eigenvalue biggest ---
Eigenvalue: -5.20135675
Norm of residual  $||Ay - \lambda y||$ : 0.15179E-05

```

```

> $ diff A eigenvector-a.txt A eigenvector-b.txt
2,10d1
< -0.1249
< 0.2328
< -0.3226
< 0.3868
< -0.4202
< 0.4202
< -0.3868
< 0.3226
< -0.2328
11a3,11
> -0.2328
> 0.3226
> -0.3868
> 0.4202
> -0.4202
> 0.3868
> -0.3226
> 0.2328
> -0.1249
13,22c13,22
< 0.1142
< 0.2271
< 0.3216
< 0.3894
< 0.4247
< 0.4247
< 0.3894
< 0.3216
< 0.2271
< 0.1142
---
> -0.1142
> -0.2271
> -0.3216
> -0.3894
> -0.4247
> -0.4247
> -0.3894
> -0.3216
> -0.2271
> -0.1142

```



```

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/projeto2
> $ diff At_eingenvector-a.txt At_eingenvector-b.txt
2c2
< -0.1249
---
> 0.1249
4,10c4,10
< 0.3550
< -0.4223
< 0.4478
< -0.4291
< 0.3678
< -0.2702
< 0.1478
---
> -0.3550
> 0.4223
> -0.4478
> 0.4291
> -0.3678
> 0.2702
> -0.1478
13c13
< 0.1142
---
> -0.1142
15,21c15,21
< -0.2856
< -0.3466
< -0.3859
< -0.4009
< -0.3906
< -0.3559
< -0.3010
---
> 0.2856
> 0.3466
> 0.3859
> 0.4009
> 0.3906
> 0.3559
> 0.3010

```

Nota-se que todas as verificações - em ambos os códigos - resultaram em valores pequenos, mostrando que o método funcionou bem para ambos os códigos.

Entretanto, ao comparar os arquivos que contêm os autovetores resultantes ("At\_eingenvector-a.txt" e "A\_eingenvector-a.txt" para o sinal original e "At\_eingenvector-b.txt" e "A\_eingenvector-b.txt" para o sinal modificado) - além de comparar as matrizes impressas como resultado dos códigos - percebe-se que há uma diferença de sinal em alguns elementos da matriz (além de uma pequena diferença no valor absoluto de alguns elementos), enquanto os autovetores apresentam - além da diferença de sinal - uma inversão da "ordem" dos elementos (apesar de serem os mesmos).

Deste modo, percebe-se que o método funcionou bem para ambos os sinais - encontrando, para cada sinal, uma das duas soluções possíveis. O erro esperado - devido ao erro de subtração numérica - não foi um problema devido a um "bom comportamento" da matriz.

## 4 Exercício 4

Tarefa: Considere novamente o problema da lista 6. Tente resolvê-lo usando a menor quantidade possível de memória. Indique, em função do tamanho  $n$  da matriz, qual é a quantidade de espaço de memória utilizado pelas variáveis (do tipo real) no programa 3

O código foi compilado com o comando:

```
gfortran -ffree-form -ffree-line-length-none P2-5255417-ex-4.f90 -Wall -Wextra  
-pedantic -o P2-5255417-ex-4.exe
```

### Modificações Feitas no Algoritmo

As seguintes alterações foram implementadas para passar de uma abordagem com alto consumo de memória para uma otimizada:

1. **Eliminação da Matriz de Transformação (0):** A mudança mais significativa foi deixar de construir e armazenar a matriz de transformação ortogonal explícita  $O$  (de tamanho  $n \times n$ ).
2. **Eliminação da Cópia da Matriz Original (A):** A cópia da matriz original, que era mantida para verificações, foi removida. O algoritmo agora opera “in-place” em uma única matriz ( $A_t$ ).
3. **Armazenamento Implícito dos Vetores de Householder:** Em vez de descartar os vetores de Householder  $u_k$  a cada passo, suas informações essenciais são agora armazenadas:
  - Uma parte do vetor  $u_k$  (os elementos de  $u_{k+2}$  em diante) é guardada na parte triangular inferior da matriz  $A_t$ , aproveitando o espaço que é zerado pelo próprio algoritmo.
  - O primeiro elemento de cada vetor  $u_k$  (`u1_storage`) e o escalar de transformação  $\beta_k$  (`betas`) são guardados em dois novos vetores auxiliares de dimensão  $n$ .
4. **Criação da Sub-rotina `backward_transformation`:** Uma nova sub-rotina foi criada com a finalidade de aplicar as transformações de Householder em ordem inversa ( $P_{n-2}, \dots, P_1$ ) diretamente no autovetor `yt`. Ela reconstrói cada transformação a partir das informações salvas e substitui a operação `matmul(0, yt)`.

# Comparação do Uso de Memória

## Versão Antiga (Não Otimizada)

Nesta versão, o programa mantém múltiplas cópias da matriz e constrói explicitamente a matriz de transformação  $O$ .

### 1. Variáveis Alocadas no Programa Principal (Globais)

Estas variáveis existem durante toda a execução do programa.

- Matriz Original **A**:  $n^2$  elementos
- Matriz de Trabalho **At**:  $n^2$  elementos
- Matriz de Transformação **O**:  $n^2$  elementos
- Vetores de Diagonais **d**, **e**:  $2n$  elementos
- Vetores de Autovetores **yt\_min/max**, **y\_min/max**:  $4n$  elementos

**Subtotal (Globais):**  $3n^2 + 6n$  elementos.

### 2. Pico de Memória Local (Dentro de Sub-rotinas)

A memória de pico ocorre quando o programa principal chama a sub-rotina que mais aloca memória localmente, neste caso, a `verify_transformation`.

- Matriz **C**:  $n^2$  elementos
- Matriz **TEMP**:  $n^2$  elementos

**Pico Local:**  $2n^2$  elementos.

### Total de Memória de Pico (Versão Antiga)

A memória total de pico é a soma das variáveis globais mais o pico de alocação local.

$$M_{\text{antiga}}(n) = \underbrace{(3n^2 + 6n)}_{\text{Globais}} + \underbrace{2n^2}_{\text{Pico Local}} = 5n^2 + 6n$$

Em bytes (assumindo 8 bytes por `real(dp)`):

$$\text{Memória}_{\text{antiga}} = (5n^2 + 6n) \times 8 \text{ bytes}$$

## Versão Nova (Otimizada)

Nesta versão, as matrizes **A** e **O** são eliminadas. A informação da transformação é guardada em vetores.

### 1. Variáveis Alocadas no Programa Principal (Globais)

- Matriz de Trabalho **At**:  $n^2$  elementos
- Vetores de Diagonais **d**, **e**:  $2n$  elementos
- Vetores de Autovetores **yt\_min/max**, **y\_min/max**:  $4n$  elementos
- Vetores Auxiliares **betas**, **u1\_storage**:  $2n$  elementos

**Subtotal (Globais):**  $n^2 + 8n$  elementos.

### 2. Pico de Memória Local (Dentro de Sub-rotinas)

A sub-rotina **verify\_transformation** foi eliminada. O pico de alocação local agora ocorre na **householder\_reduction**.

- Vetor **u**:  $n$  elementos
- Vetor **p**:  $n$  elementos
- Vetor **q**:  $n$  elementos

**Pico Local:**  $3n$  elementos.

### Total de Memória de Pico (Versão Otimizada)

A memória total de pico é a soma das novas variáveis globais mais o novo pico de alocação local.

$$M_{\text{otimizada}}(n) = \underbrace{(n^2 + 8n)}_{\text{Globais}} + \underbrace{3n}_{\text{Pico Local}} = n^2 + 11n$$

Em bytes:

$$\text{Memória}_{\text{otimizada}} = (n^2 + 11n) \times 8 \text{ bytes}$$

Versão	Fórmula de Memória (elementos)	Termo Dominante
Antiga	$5n^2 + 6n$	$5n^2$
Otimizada	$n^2 + 11n$	$n^2$

## Conclusão Comparativa

A otimização reduziu o uso de memória que escala com o quadrado da dimensão da matriz de  $5n^2$  para  $n^2$ . Isso representa uma **redução de 80%** no termo dominante, que é a principal fonte de consumo de memória para matrizes grandes.

Resultados:

```
> $ ./P2-5255417-ex-4.exe
Insert matrix dimension:
10

-----> N = 10 <-----

Smallest eigenvalue (lambda_min): -0.08384774
Largest eigenvalue (lambda_max): -5.20135675

--- Verification for Eigenvalue smallest ---
Eigenvalue: -0.08384774
Norm of residual ||Ay - lambday||: 0.40859E-07

--- Verification for Eigenvalue biggest ---
Eigenvalue: -5.20135675
Norm of residual ||Ay - lambday||: 0.15076E-05

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/projeto2
> $ diff At_eigenvector-a.txt At_eigenvector-4.txt

pedro@Pedro-Lenovo ~/Documentos/GitHub/quanticacomp/projeto2
> $ diff A_eigenvector-a.txt A_eigenvector-4.txt
```

Nota-se que todas as verificações resultaram em valores pequenos, comprovando que o código permanece funcional. Além disso, a pequena diferença em um elemento do vetor final pode ser atribuída a problemas de aproximação que ocorrem em ambos os códigos — não afetando a validade do resultado.