Solução dos Exercícios de Álgebra Linear Computacional

Juan Lopes

Julho 2014

Conteúdo

Introdução

Este é o documento explicativo da entrega dos exercícios para a disciplina *Álgebra Linear: Aspectos Teóricos e Computacionais*, ministrada pelo professor Ricardo Carvalho de Barros.

O arquivo acompanha os códigos-fonte das soluções dos exercícios. Todos os programas foram implementados em C++, utilizando a entrada e saída padrões

para testar seu funcionamento, todos também acompanham um exemplo de arquivo de entrada e sua respectiva saída esperada, para facilitar testes automáticos.

As soluções foram escritas e testadas em Linux, utilizando o compilador `g++ 4.7.2`. Este pacote também possui um script para facilitar o teste das implementações.

Esta entrega cobre completamente todos os exercícios que requeriam implementação de programas. Alguns exercícios foram omitidos, como o `3.1`, cuja solução pode ser encontrada trivialmente utilizando o programa escrito para o exercício `3.3`.

Os arquivos com os códigos-fonte encontram-se na raiz do pacote, os respectivos exemplos de entrada e saída encontram-se em arquivos `.in` e `.out` dentro do diretório `data`.

É possível utilizar a linha de comando para compilar, executar e verificar o resultado. Por exemplo, para o exercício `1.1`, basta executar o comando:

```
g++ exercicio_1.1.cpp &&
    ./a.out < data/exercicio_1.1.in |
    diff - data/exercicio_1.1.out
```

ou utilizar o script

```
./test.sh 1.1
```

O script `test.sh` também permite testar todos os programas em sequência automaticamente. Para tanto, basta executá-lo sem argumentos.

```
./test.sh
```

1 Capítulo 1 - Matrizes

1.1 Igualdade de matrizes

Enunciado Faça um programa que leia duas matrizes e verifique se elas são iguais.

Solução O código-fonte do programa encontra-se em `exercicio_1.1.cpp`.

Entrada Cada entrada consiste em duas matrizes, como no exemplo:

```
2 3
1 2 3
4 5 6
2 3
1 2 3
4 5 6
```

Saída A saída é a string `T` caso as matrizes sejam iguais ou `F` caso sejam diferentes:

```
T
```

1.2 Soma, subtração e produto por escalar

Enunciado Faça um programa que faça a soma e subtração de matrizes e o produto de uma matriz por um escalar.

Solução O código-fonte do programa encontra-se em `exercicio_1.2.cpp`.

Entrada Cada entrada começa com uma string informando a operação (`add`, `sub` ou `mul`). Para `add` e `sub` são lidas duas matrizes; para `mul` é lida uma matriz e um número.

```
add
2 3
1 2 3
4 5 6
2 3
1 2 3
4 5 6

sub
2 3
1 2 3
4 5 6
2 3
2 3 4.5
5 6 7

mul
2 3
1 2 3
4 5 6
5
```

Saída A saída é a matriz resultante da operação escolhida, seguida por `---`.

```
2 4 6
8 10 12
---
```

```
-1 -1 -1.5
-1 -1 -1
---
5 10 15
20 25 30
---
```

1.3 Transposição

Enunciado Faça um programa que leia uma matriz e calcule sua transposta.

Solução O código-fonte do programa encontra-se em `exercicio_1.3.cpp`.

Entrada Cada entrada consiste na matriz a ser transposta.

```
2 3
1 2 3
4 5 6
```

Saída A saída é a matriz resultante da operação, seguida por `---`.

```
1 4
2 5
3 6
---
```

1.4 Produto de matrizes

Enunciado Faça um programa que calcule o produto de matrizes.

Solução O código-fonte do programa encontra-se em `exercicio_1.4.cpp`.

Entrada Cada entrada consiste em duas matrizes.

```
3 2
2 1
4 2
5 3

2 2
1 -1
0 4
```

Saída A saída é a matriz resultante da operação, seguida por `---`.

```
2 2
4 4
5 7
---
```

3 Capítulo 3 - Solução de Sistemas Lineares

3.3 Método de Gauss

Enunciado Faça um programa que calcule a solução de sistemas lineares utilizando o método de Gauss.

Solução O código-fonte do programa encontra-se em `exercicio_3.3.cpp`.

Entrada Cada entrada consiste em uma matriz estendida do sistema do sistema a ser resolvido.

```
4 5
-1 3 5 2 10
1 9 8 4 15
0 1 0 1 2
2 1 1 -1 -3
```

Saída A saída é a matriz resolvida do sistema, seguida por `---`.

```
1.000 0.000 0.000 0.000 -1.000
0.000 1.000 0.000 0.000 -0.000
0.000 0.000 1.000 0.000 1.000
0.000 0.000 0.000 1.000 2.000
---
```

3.4 Método de Gauss – Pivoteamento Parcial

Enunciado Faça um programa que calcule a solução de sistemas lineares utilizando o método de Gauss com a estratégia de pivoteamento parcial.

Solução O código-fonte do programa encontra-se em `exercicio_3.4.cpp`.

Entrada Cada entrada consiste em uma matriz estendida do sistema do sistema a ser resolvido.

```
4 5
-1 3 5 2 10
1 9 8 4 15
0 1 0 1 2
2 1 1 -1 -3
```

Saída A saída é a matriz resolvida do sistema, seguida por `---`.

```
1.000 0.000 0.000 0.000 -1.000
0.000 1.000 0.000 0.000 0.000
0.000 0.000 1.000 0.000 1.000
0.000 0.000 0.000 1.000 2.000
---
```

3.6 Decomposição LU

Enunciado Faça um programa que decomponha uma matriz A dada no exemplos 3.6 e 3.7 e no exercício 3.5 em matrizes L e U.

Solução O código-fonte do programa encontra-se em `exercicio_3.6.cpp`.

Entrada Cada entrada consiste em uma matriz a ser decomposta.

```
4 4
2.1756 4.0231 -2.1732 5.1967
-4.0231 6.0000 0 1.1973
-1.0000 -5.2107 1.1111 0
6.0235 7.0000 0 -4.1561
```

Saída A saída são duas matrizes L e U, seguidas por `---`.

```
L:
1.0000 0.0000 0.0000 0.0000
-1.8492 1.0000 0.0000 0.0000
-0.4596 -0.2501 1.0000 0.0000
2.7687 -0.3079 -5.3523 1.0000
U:
2.1756 4.0231 -2.1732 5.1967
0.0000 13.4395 -4.0187 10.8070
0.0000 0.0000 -0.8930 5.0917
0.0000 0.0000 0.0000 12.0361
---
```

3.7 Decomposição LU - Solução de Sistemas

Enunciado Faça um programa FORTRAN que resolva sistemas lineares utilizando a decomposição L e U .

Solução O código-fonte do programa encontra-se em `exercicio_3.7.cpp`.

Entrada Cada entrada consiste em uma matriz estendida do sistema a ser resolvido.

```
3 4
4 0 -3 -2
3 -4 1 9
1 2 2 3
```

Saída A saída são as diversas matrizes criadas nos passos da solução, seguidas por `---`. A última matriz contém a solução do sistema.

```
LY = B:
1.0000 0.0000 0.0000 -2.0000
0.7500 1.0000 0.0000 9.0000
0.2500 -0.5000 1.0000 3.0000
solved Y:
1.0000 0.0000 0.0000 -2.0000
0.0000 1.0000 0.0000 10.5000
0.0000 0.0000 1.0000 8.7500
UX = Y:
4.0000 0.0000 -3.0000 -2.0000
0.0000 -4.0000 3.2500 10.5000
0.0000 0.0000 4.3750 8.7500
solved X:
1.0000 0.0000 0.0000 1.0000
0.0000 1.0000 0.0000 -1.0000
0.0000 0.0000 1.0000 2.0000
---
```

3.8 Decomposição LU - Pivoteamento Parcial

Enunciado Refaça o programa do exercício anterior, utilizando também a estratégia de pivoteamento parcial.

Solução O código-fonte do programa encontra-se em `exercicio_3.8.cpp`.

Entrada Cada entrada consiste em uma matriz estendida do sistema a ser resolvido.

```
3 4
3 -4 1 9
1 2 2 3
4 0 -3 -2
```

Saída A saída são as diversas matrizes criadas nos passos da solução (inclusive P), seguidas por `---`. A última matriz contém a solução do sistema.

```
P:
0.0000 0.0000 1.0000
1.0000 0.0000 0.0000
0.0000 1.0000 0.0000
LY = PB:
1.0000 0.0000 0.0000 -2.0000
0.7500 1.0000 0.0000 9.0000
0.2500 -0.5000 1.0000 3.0000
solved Y:
1.0000 0.0000 0.0000 -2.0000
0.0000 1.0000 0.0000 10.5000
0.0000 0.0000 1.0000 8.7500
UX = Y:
4.0000 0.0000 -3.0000 -2.0000
0.0000 -4.0000 3.2500 10.5000
0.0000 0.0000 4.3750 8.7500
solved X:
1.0000 0.0000 0.0000 1.0000
0.0000 1.0000 0.0000 -1.0000
0.0000 0.0000 1.0000 2.0000
---
```

3.9 Fatoração de Cholesky - LDL^t

Enunciado Fatore a matriz dada em um produto LDL^t, onde L é triangular inferior, com todos os elementos diagonais iguais a 1, e D é uma matriz a diagonal.

Solução O código-fonte do programa encontra-se em `exercicio_3.9.cpp`.

Entrada Cada entrada consiste em uma matriz a ser fatorada.

```
4 4
16 -4 12 -4
-4 2 -1 1
```

```
12 -1 14 -2
-4 1 -2 83
```

Saída A saída são matrizes L, D e L^t, seguidas por `---`.

```
L:
1.0000 0.0000 0.0000 0.0000
-0.2500 1.0000 0.0000 0.0000
0.7500 2.0000 1.0000 0.0000
-0.2500 0.0000 1.0000 1.0000
D:
16.0000 0.0000 0.0000 0.0000
0.0000 1.0000 0.0000 0.0000
0.0000 0.0000 1.0000 0.0000
0.0000 0.0000 0.0000 81.0000
Lt:
1.0000 -0.2500 0.7500 -0.2500
0.0000 1.0000 2.0000 0.0000
0.0000 0.0000 1.0000 1.0000
0.0000 0.0000 0.0000 1.0000
---
```

3.10 Fatoração de Cholesky - GG^t

Enunciado Use o algoritmo de Cholesky para encontrar uma fatoração da forma $A = GG^t$ para as matrizes que seguem.

Solução O código-fonte do programa encontra-se em `exercicio_3.10.cpp`.

Entrada Cada entrada consiste em uma matriz a ser fatorada.

```
4 4
4 1 1 1
1 3 -1 1
1 -1 2 0
1 1 0 2
```

Saída A saída é a matriz G, seguida por `---`.

```
2.0000 0.0000 0.0000 0.0000
0.5000 1.6583 0.0000 0.0000
0.5000 -0.7538 1.0871 0.0000
0.5000 0.4523 0.0836 1.2403
---
```

3.12 Fatoração de Cholesky - Sistemas

Enunciado Faça um programa que resolva sistemas lineares utilizando a decomposição de Cholesky.

Solução O código-fonte do programa encontra-se em `exercicio_3.12.cpp`.

Entrada Cada entrada consiste em uma matriz estendida a ser resolvida.

```
4 5
4 1 1 1 0.65
1 3 -1 1 0.05
1 -1 2 0 0
1 1 0 2 0.5
```

Saída A saída são as diversas matrizes criadas nos passos da solução, seguidas por `---`. A última matriz contém a solução do sistema.

```
GY = B:
2.0000 0.0000 0.0000 0.0000 0.6500
0.5000 1.6583 0.0000 0.0000 0.0500
0.5000 -0.7538 1.0871 0.0000 0.0000
0.5000 0.4523 0.0836 1.2403 0.5000
solved Y:
1.0000 0.0000 0.0000 0.0000 0.3250
0.0000 1.0000 0.0000 0.0000 -0.0678
0.0000 0.0000 1.0000 0.0000 -0.1965
0.0000 0.0000 0.0000 1.0000 0.3101
GtX = Y:
2.0000 0.5000 0.5000 0.5000 0.3250
0.0000 1.6583 -0.7538 0.4523 -0.0678
0.0000 0.0000 1.0871 0.0836 -0.1965
0.0000 0.0000 0.0000 1.2403 0.3101
solved X:
1.0000 0.0000 0.0000 0.0000 0.2000
0.0000 1.0000 0.0000 0.0000 -0.2000
0.0000 0.0000 1.0000 0.0000 -0.2000
0.0000 0.0000 0.0000 1.0000 0.2500
---
```

3.13 Método de Jacobi

Enunciado Faça um programa para resolver os sistemas lineares do exercício 1, com $\epsilon = 10^{-3}$.

Solução O código-fonte do programa encontra-se em `exercicio_3.13.cpp`.

Entrada Cada entrada consiste em uma matriz estendida a ser resolvida.

```
5 6
4 1 1 0 1 6
-1 -3 1 1 0 6
2 1 5 -1 -1 6
-1 -1 -1 4 0 6
0 2 -1 1 4 6
```

Saída A saída é o resultado de cada uma das iterações, seguidas por `---`.

```
Iteration #1. Epsilon=1.000000
-0.500000 -0.250000 0.000000 0.333333

Iteration #2. Epsilon=0.416000
-0.520833 -0.041667 -0.216667 0.416667

Iteration #3. Epsilon=0.229368
-0.647917 -0.069792 -0.195833 0.565278

Iteration #4. Epsilon=0.110179
-0.672830 0.004340 -0.256597 0.591319

Iteration #5. Epsilon=0.074705
-0.713064 0.001888 -0.251962 0.644589

Iteration #6. Epsilon=0.033859
-0.724610 0.026423 -0.271153 0.655638

Iteration #7. Epsilon=0.024954
-0.738303 0.027274 -0.270765 0.674062

Iteration #8. Epsilon=0.010937
-0.743025 0.035400 -0.277018 0.678781

Iteration #9. Epsilon=0.008515
-0.747800 0.036197 -0.277281 0.685148

Iteration #10. Epsilon=0.003628
-0.749656 0.038917 -0.279350 0.687093

Iteration #11. Epsilon=0.002948
-0.751340 0.039350 -0.279566 0.689308

Iteration #12. Epsilon=0.001224
```

```
-0.752056 0.040270 -0.280260 0.690085

Iteration #13. Epsilon=0.001032
-0.752654 0.040470 -0.280374 0.690862

Iteration #14. Epsilon=0.000418
-0.752927 0.040785 -0.280609 0.691166

---
```

3.14 Método de Gauss-Seidel

Enunciado Faça um programa para resolver sistemas lineares pelo método de Gauss-Seidel. Resolva os dois sistemas do exercício 1, considerando $\epsilon = 10^{-3}$.

Solução O código-fonte do programa encontra-se em `exercicio_3.14.cpp`.

Entrada Cada entrada consiste em uma matriz estendida a ser resolvida.

```
5 6
4 1 1 0 1 6
-1 -3 1 1 0 6
2 1 5 -1 -1 6
-1 -1 -1 4 0 6
0 2 -1 1 4 6
```

Saída A saída é o resultado de cada uma das iterações, seguidas por `---`.

```
Iteration #1. Epsilon=1.000000
1.500000 -2.500000 1.100000 1.525000 2.643750

Iteration #2. Epsilon=0.433865
1.189062 -1.521354 1.862396 1.882526 2.255645

Iteration #3. Epsilon=0.241892
0.850828 -1.035302 1.894363 1.927472 2.009374

Iteration #4. Epsilon=0.034274
0.782891 -0.987019 1.871616 1.916872 1.982195

Iteration #5. Epsilon=0.005662
0.783302 -0.998271 1.866147 1.912794 1.987474

Iteration #6. Epsilon=0.002079
```

```
0.786163 -1.002407 1.866070 1.912456 1.989607

Iteration #7. Epsilon=0.000261
0.786683 -1.002719 1.866283 1.912562 1.989790

---
```

4 Capítulo 4 – Autovalores e autovetores

4.3 Método das Potências

4.3.1 Solução Normal

Enunciado Faça um programa que leia uma matriz e determine um dos seus autovalores e o autovetor correspondente pelo método das potências.

Solução O código-fonte do programa encontra-se em `exercicio_4.3.1.cpp`.

Entrada Cada entrada consiste em uma matriz a ser aplicada o método.

```
3 3
-4 14 0
-5 13 0
-1 0 2
```

Saída A saída é a descrição de cada uma das iterações do algoritmo.

```
Iteration #1. Epsilon=0.900000
Eigenvalue: 10.000000
Eigenvector:
1.000000 0.800000 0.100000

Iteration #2. Epsilon=0.211111
Eigenvalue: 7.200000
Eigenvector:
1.000000 0.750000 -0.111111

Iteration #3. Epsilon=0.076923
Eigenvalue: 6.500000
Eigenvector:
1.000000 0.730769 -0.188034

Iteration #4. Epsilon=0.032816
Eigenvalue: 6.230769
Eigenvector:
```

```
1.000000 0.722222 -0.220850

Iteration #5. Epsilon=0.015064
Eigenvalue: 6.111111
Eigenvector:
1.000000 0.718182 -0.235915

Iteration #6. Epsilon=0.007180
Eigenvalue: 6.054545
Eigenvector:
1.000000 0.716216 -0.243095

Iteration #7. Epsilon=0.003493
Eigenvalue: 6.027027
Eigenvector:
1.000000 0.715247 -0.246588

Iteration #8. Epsilon=0.001718
Eigenvalue: 6.013453
Eigenvector:
1.000000 0.714765 -0.248306

Iteration #9. Epsilon=0.000851
Eigenvalue: 6.006711
Eigenvector:
1.000000 0.714525 -0.249157

Iteration #10. Epsilon=0.000423
Eigenvalue: 6.003352
Eigenvector:
1.000000 0.714405 -0.249579

Iteration #11. Epsilon=0.000211
Eigenvalue: 6.001675
Eigenvector:
1.000000 0.714346 -0.249790

Iteration #12. Epsilon=0.000105
Eigenvalue: 6.000837
Eigenvector:
1.000000 0.714316 -0.249895

Iteration #13. Epsilon=0.000052
Eigenvalue: 6.000419
Eigenvector:
1.000000 0.714316 -0.249895
```

4.3.2 Método de Aitken

Enunciado Insira no programa do método das potências o método de Aitken.

Solução O código-fonte do programa encontra-se em `exercicio_4.3.2.cpp`.

Entrada Cada entrada consiste em uma matriz a ser aplicada o método.

```
3 3
-4 14 0
-5 13 0
-1 0 2
```

Saída A saída é a descrição de cada uma das iterações do algoritmo.

```
Iteration #1. Epsilon=0.900000
Eigenvalue: 0.000000
Eigenvector:
1.000000 0.800000 0.100000

Iteration #2. Epsilon=0.211111
Eigenvalue: 7.812500
Eigenvector:
1.000000 0.750000 -0.111111

Iteration #3. Epsilon=0.076923
Eigenvalue: 6.266667
Eigenvector:
1.000000 0.730769 -0.188034

Iteration #4. Epsilon=0.032816
Eigenvalue: 6.062500
Eigenvector:
1.000000 0.722222 -0.220850

Iteration #5. Epsilon=0.015064
Eigenvalue: 6.015385
Eigenvector:
1.000000 0.718182 -0.235915

Iteration #6. Epsilon=0.007180
Eigenvalue: 6.003831
```

```
Eigenvector:
1.000000 0.716216 -0.243095

Iteration #7. Epsilon=0.003493
Eigenvalue: 6.000957
Eigenvector:
1.000000 0.715247 -0.246588

Iteration #8. Epsilon=0.001718
Eigenvalue: 6.000239
Eigenvector:
1.000000 0.714765 -0.248306

Iteration #9. Epsilon=0.000851
Eigenvalue: 6.000060
Eigenvector:
1.000000 0.714525 -0.249157

Iteration #10. Epsilon=0.000423
Eigenvalue: 6.000015
Eigenvector:
1.000000 0.714405 -0.249579

Iteration #11. Epsilon=0.000211
Eigenvalue: 6.000004
Eigenvector:
1.000000 0.714346 -0.249790

Iteration #12. Epsilon=0.000105
Eigenvalue: 6.000001
Eigenvector:
1.000000 0.714316 -0.249895

Iteration #13. Epsilon=0.000052
Eigenvalue: 6.000000
Eigenvector:
1.000000 0.714316 -0.249895

---
```