Decomposição de Cholesky

Larissa Trindade

1. Definição

Definição

Decomposição de Cholesky ou Fatoração de Cholesky é um método de álgebra linear para *resoluções de sistemas lineares.*

Para utilizar este método e necessário que a matriz do sistema linear seja **quadrada** (n x n), **simétrica** e **definida positiva.**

Para utilizar a decomposição de Cholesky é utilizado a equação (1) onde **A** é a *matriz inicial* e **L** é uma *matriz triangular inferior* com elementos da diagonal principal estritamente positivos.

$$A = LL^T$$

Definição

Matiz Definida Positiva

- Todos os elementos da diagonal principal sejam positivos
- Determinante das subMatrizes> 0

$$\begin{pmatrix} 4 & 12 & -16 \\ 12 & 37 & -43 \\ -16 & -43 & 98 \end{pmatrix}$$

Definição

Matiz Definida Positiva

$$\begin{pmatrix} 4 & 12 & -16 \\ 12 & 37 & -43 \\ -16 & -43 & 98 \end{pmatrix}$$

$$det(4) = 4$$

$$det(A_{2x2}) = 4$$

$$det(A) = 36$$

Definição

$$A = LL^T$$

$$\begin{pmatrix} 4 & 12 & -16 \\ 12 & 37 & -43 \\ -16 & -43 & 98 \end{pmatrix} =$$

$$= \begin{pmatrix} 2 & 0 & 0 \\ 6 & 1 & 0 \\ -8 & 5 & 3 \end{pmatrix} \begin{pmatrix} 2 & 6 & -8 \\ 0 & 1 & 5 \\ 0 & 0 & 3 \end{pmatrix}$$

2. Aplicações

Aplicações

Muito usada principalmente para solução numérica de equações lineares

- Calculo de minimos quadrados
- Otimização não linear
- Inversão de Matrizes
- Simulação de Monte Carlo
- Filtro de Kalman
- Processamento de imagens *(tese da uff)*

Aplicações

$$\begin{cases} 1x_1 + 1x_2 & = 1 \\ 1x_1 + 2x_2 - 1x_3 & = 1 \\ \quad - 1x_2 + 3x_3 & = 2 \end{cases}$$

1. Forma matricial

$$\underbrace{\begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 3 \end{pmatrix}}_{} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

Aplicações

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 3 \end{pmatrix}$$

2. Aplicando Cholesky

$$A = LL^T = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & \sqrt{2} \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & \sqrt{2} \end{pmatrix}$$

Aplicações

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 3 \end{pmatrix}$$

2. Aplicando Cholesky

$$A = LL^T = \begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{pmatrix}$$

Aplicações

$$\begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{pmatrix}$$

$$= \begin{pmatrix} l_{11}^2 & l_{11}l_{21} & l_{11}l_{31} \\ l_{11}l_{21} & l_{21}^2 + l_{22}^2 & l_{21}l_{31} + l_{22}l_{32} \\ l_{11}l_{31} & l_{21}l_{31} + l_{22}l_{32} & l_{31}^2 + l_{32}^2 + l_{33}^2 \end{pmatrix}$$

Aplicações

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 3 \end{pmatrix}$$

$$l_{11}^2 = 1 \rightarrow l_{11} = \sqrt{1} = 1$$

$$l_{11} l_{21} = 1 \rightarrow l_{21} = \frac{1}{l_{11}} = 1$$

$$l_{11} l_{31} = 0 \rightarrow l_{31} = \frac{0}{l_{11}} = 0$$

$$l_{21}^2 + l_{22}^2 = 2 \rightarrow l_{22} = \sqrt{2 - l_{21}^2} = \sqrt{1} = 1$$

$$l_{21} l_{31} + l_{22} l_{32} = -1 \rightarrow l_{32} = \frac{-1 - l_{31} l_{21}}{l_{22}} = \frac{-1 - (0 \cdot 1)}{1} = 1$$

$$l_{31}^2 + l_{32}^2 + l_{33}^2 = 3 \rightarrow l_{33} = \sqrt{3 - l_{31}^2 - l_{32}^2} = \sqrt{3 - 0 - 1} = \sqrt{2}$$

Aplicações

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 3 \end{pmatrix}$$

$$A = LL^T = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & \sqrt{2} \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & \sqrt{2} \end{pmatrix}$$

Aplicações

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & \sqrt{2} \end{pmatrix} \qquad \boldsymbol{Y} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

3. Utilizando a triangular inferior junto de Y (resposta do sistema)

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & \sqrt{2} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}$$

$$(z_1, z_2, z_3) = (1{,}0, \sqrt{2})$$

$$L^T = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & \sqrt{2} \end{pmatrix} \qquad Z = \begin{pmatrix} 1 \\ 1 \\ \sqrt{2} \end{pmatrix}$$

4. Utilizando a triangular superior junto de Z (resposta do sistema de L)

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & \sqrt{2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \sqrt{2} \end{pmatrix}$$

$$(x_1, x_2, x_3) = (0,1,1)$$

3. Algoritmo

Foi pego um algoritmo da *13th Marathon of Parallel Programming – WSCAD 2018* que possui o problema da decomposição de cholesky

13th Marathon of Parallel Programming – WSCAD 2018 7

Problem D

Cholesky Decomposition[1]

Every symmetric, positive matrix A can be decomposed into a product of an unique lower triangular matrix L and its transpose:

$$A = LL^T \tag{3}$$

L is called the *Cholesky factor* of A, and can be interpreted as a generalized square root of A.

Your task is to improve performance of the source-code using parallel strategies. We are not interested in finding out which decomposition is better, therefore is not allowed to change the Cholesky decomposition algorithm.

Input

The input file contains only one test case. The first line contains the size of a square matrix ($0 < N \leq 10^4$). Next, N lines are the rows of the matrix, N real numbers per row.

The input must be read from a file named cholesky.in.

Output

The output must have the lower Cholesky factor L from the symmetric matrix A.

The output must be written to a file named cholesky.out.

Algoritmo

Divisão:

1. Reservar espaço de memória
2. Pegar a matriz no arquivo cholesky.in e alocar no espaço reservado
3. Fazer a decomposição de Cholesky
4. Inserir a resposta em um arquivo
5. Desalocar espaço de memória

```
void cholesky(double** A, int n)
void show_matrix(double** A, int n)
void mat_zero(double** Ax, int n)
double** mat_new(int n)
void mat_gen(FILE* file,double** s, int n)
void mat_del(double** x)
int main()
```

Algoritmo

main

```
int n;
FILE* file = fopen("cholesky.in", "r");
fscanf(file, "%d", &n);

double** A = mat_new(n);
mat_gen(file, A,  n);
fclose(file);

cholesky(A, n);
show_matrix(A, n);

mat_del(A);
```

cholesky.in

```
3
4 12 -16
12 37 -43
-16 -43 98
```

Algoritmo

mat_new

```
double** mat_new(int n) {
    int i;
    double** x = malloc(sizeof(double*) * n);
    assert(x != NULL);

    for(i = 0; i < n; i++){
            x[i]  = malloc(sizeof(double) * n);
            assert(x != NULL);
    }
    mat_zero(x,n);
    return x;
}
```

mat_zero

```
void mat_zero(double** x, int n) {
    int i, j;
    for(i = 0; i < n; i++){
        for(j = 0; j < n; j++){
            x[i][j]  = 0.0;
        }
    }
}
```

x

0	0	0
0	0	0
0	0	0

Algoritmo

mat_gen

```
void mat_gen(FILE* file, double** s, int n) {
    int i, j;
    for(i = 0; i < n; i++) {
        for(j = 0; j<n; j++) {
            fscanf(file, "%lf", &s[i][j]);
        }
    }
}
```

file

4	12	-16
12	37	-43
-16	-43	98

s

0	0	0
0	0	0
0	0	0

s

4	12	-16
12	37	-43
-16	-43	98

Algoritmo

cholesky

```
void cholesky(double** A, int n) {
  for (int i = 0; i < n; i++)
    for (int j = 0; j < (i + 1); j++) {
      double s = 0;
      for (int k = 0; k < j; k++)  s += A[i][k] * A[j][k];
      A[i][j] = (i == j) ? sqrt(A[i][i] - s) : (1.0 / A[j][j] * (A[i][j] - s));
    }
}
```

A

4 12 -16

12 37 -43

-16 -43 98

Algoritmo

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} l_{11}^2 & l_{11}l_{21} & l_{11}l_{31} \\ l_{11}l_{21} & l_{21}^2 + l_{22}^2 & l_{21}l_{31} + l_{22}l_{32} \\ l_{11}l_{31} & l_{21}l_{31} + l_{22}l_{32} & l_{31}^2 + l_{32}^2 + l_{33}^2 \end{pmatrix}$$

Diagonal Principal

$$l_{11}^2 = a_{11} \rightarrow l_{11} = \sqrt{a_{11}}$$

$$l_{21}^2 + l_{22}^2 = a_{22} \rightarrow l_{22} = \sqrt{a_{22} - l_{21}^2}$$

$$l_{31}^2 + l_{32}^2 + l_{33}^2 = a_{33} \rightarrow l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2}$$

$$\boxed{l_{kk} = \sqrt{a_{kk} - \sum_{j=1}^{k-1} l_{kj}^2}}$$

Abaixo da Diagonal

i > k

$$l_{11}l_{21} = a_{21} \rightarrow l_{21} = \frac{1}{l_{11}} a_{21}$$

$$l_{11}l_{31} = a_{31} \rightarrow l_{31} = \frac{1}{l_{11}} a_{31}$$

$$l_{21}l_{31} + l_{22}l_{32} = a_{32} \rightarrow l_{32} = \frac{1}{l_{22}} (a_{32} - l_{31}l_{21})$$

$$\boxed{l_{ik} = \frac{1}{l_{kk}} \left(a_{ik} - \sum_{j=1}^{k-1} l_{ij} l_{kj} \right)}$$

Algoritmo

cholesky

```
void cholesky(double** A, int n) {
  for (int i = 0; i < n; i++)
    for (int j = 0; j < (i + 1); j++) {
      double s = 0;
      for (int k = 0; k < j; k++)  s += A[i][k] * A[j][k];
      A[i][j] = (i == j) ? sqrt(A[i][i] - s) : (1.0 / A[j][j] * (A[i][j] - s));
    }
}
```

A

4	**12**	**-16**
12	**37**	**-43**
-16	**-43**	**98**

$$l_{kk} = \sqrt{a_{kk} - \sum_{j=1}^{k-1} l_{kj}^2}$$

$$l_{ik} = \frac{1}{l_{kk}}\left(a_{ik} - \sum_{j=1}^{k-1} l_{ij} l_{kj}\right)$$

4. Execução

Foi montado um arquivo em python para gerar 5 entradas:

- *cholesky_1000.in*
- *cholesky_2000.in*
- *cholesky_5000.in*
- *cholesky_7000.in*
- *cholesky_10000.in*

Arquivo foi criado e testado mais de 10 vezes com entradas entre 10 a 20

Verificação: Próprio programa da Decomposição de Cholesky (caso algo esteja errado e a matriz não esteja correta, o arquivo .out possui algumas linhas com **#INDO**)

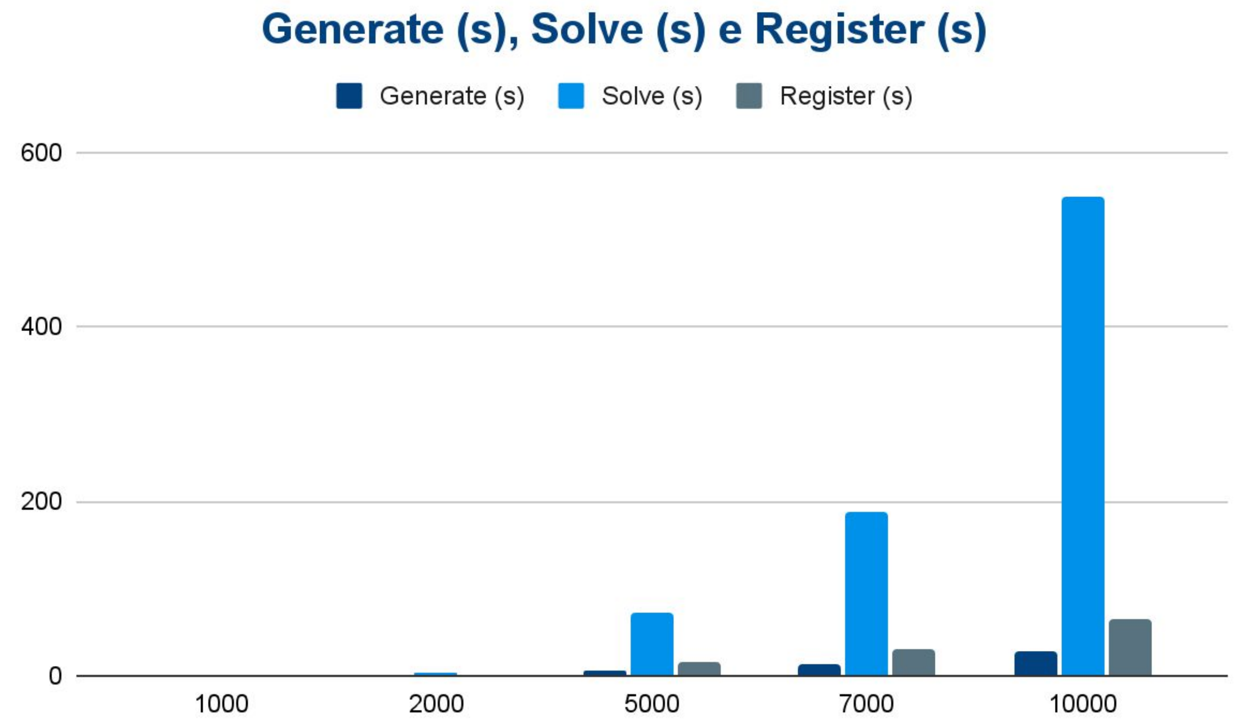
```python
import sys
import os
import random
import numpy as np

n = 10000
m = 0
mat = np.zeros((n,n))

base_path = sys.path[0]

for i in range (0,n):
  if (i > 0): m=1
  for j in range(0,n):
    if(i==j):
      mat[i][j] = mat[i-m][i-m] + random.randint(5.0, 25.0) * 15
    else:
      mat[i][j] = random.randint(5.0, 25.0)
      mat[j][i] = random.randint(5.0, 25.0)

file = open(os.path.join(base_path, 'output_10000.txt'), 'w')
file.write(f'{n}\n')

for i in range(0, n):
    for j in range(0, n):
        file.write(f'{mat[i][j]} ')
    file.write(f'\n')

file.close()
```

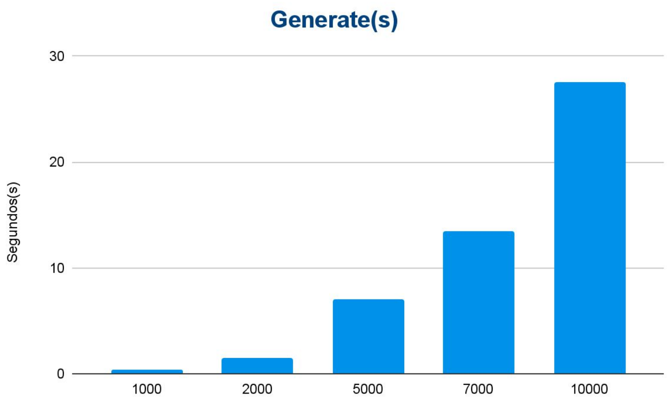
Execução

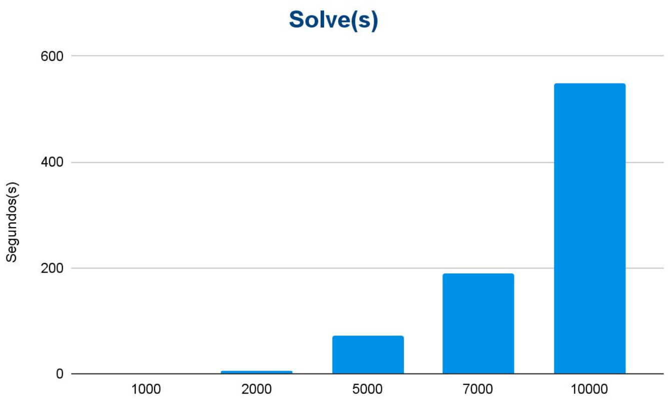
Execução - Média

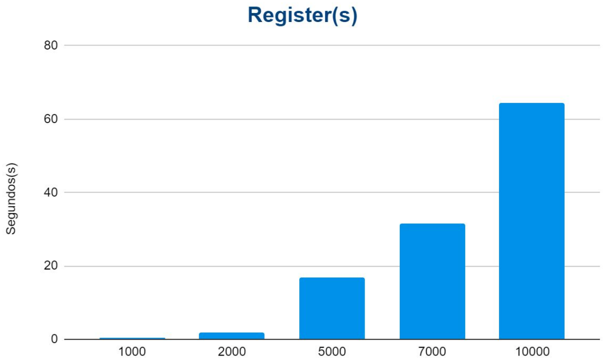
Tamanho	Generate (s)	Solve (s)	Register (s)
1000	0,387052375	0,60347875	0,493512875
2000	1,473929875	4,7312955	1,84189425
5000	7,046217333	71,45393867	16,928008
7000	13,50594214	188,9381343	31,48132257
10000	27,5403333	548,1843978	64,5537512

Média realizada com a remoção dos outliers


Generate (s), Solve (s) e Register (s)
Generate (s)
Solve (s)
Register (s)
600
400
200
0
1000
2000
5000
7000
10000


Generate(s)
Segundos(s)
30
20
10
0
1000
2000
5000
7000
10000


Solve(s)
Segundos(s)
600
400
200
0
1000
2000
5000
7000
10000


Register(s)
Segundos(s)
80
60
40
20
0
1000
2000
5000
7000
10000

Execução - 1000

Execução	Generate (s)	Solve (s)	Register (s)
1°	0,342469	0,581672	0,481214
2°	0,508772	0,574577	0,462293
3°	0,339272	0,548310	0,472918
4°	00,369826	0,620651	0,474532
5°	0,383386	0,570517	0,452311
6°	0,339147	0,571218	0,522628
7°	0,368630	0,753677	0,657165
8°	0,355526	0,596328	0,766208
9°	0,389681	0,663319	0,551114
10°	0,423866	0,697566	0,531093

Execução - 2000

Execução	Generate (s)	Solve (s)	Register (s)
1°	1,598549	4,989314	2,525010
2°	1,914635	5,398484	1,934002
3°	1,396757	5,068393	2,158218
4°	1,713304	6,092130	3,692245
5°	2,927275	6,588814	2,418348
6°	1,446488	4,421056	1,723762
7°	1,348847	4,429452	1,724675
8°	1,363118	4,498401	1,116115
9°	1,363182	4,508665	1,768656
10°	1,359863	4,536599	1,784716

Execução - 5000

Execução	Generate (s)	Solve (s)	Register (s)
1°	7.113030	69.001617	53.306959
2°	6.925493	69.296814	17.018382
3°	6.908914	69.438102	24.990658
4°	7.052154	69.300864	20.981646
5°	6.885594	69.041494	16.056242
6°	7.300490	69.289890	23.111511
7°	6.981481	71.209506	18.320577
8°	6.952455	77.275324	17.431558
9°	7.692526	72.690414	16.271677
10°	6.839755	69.210080	16.469612

Execução - 7000

Execução	Generate (s)	Solve (s)	Register (s)
1°	13.575829	188.677315	31.369206
2°	13.378792	189.239419	54.434722
3°	13.358338	188.475025	50.270260
4°	13.362614	189.491732	60.731177
5°	13.489216	188.825259	30.661505
6°	13.508018	189.053914	30.973957
7°	13.459068	188.403589	32.656515
8°	13.627997	191.024306	30.466398
9°	13.392843	188.578856	31.758228
10°	13.488624	188.003701	32.483449

Execução - 10000

Execução	Generate (s)	Solve (s)	Register (s)
1°	27.352833	545.852093	63.796613
2°	27.132130	548.536205	62.901952
3°	27.126506	549.374550	67.088829
4°	27.374843	547.498523	62.421641
5°	28.426419	545.971696	64.558884
6°	27.437113	546.402474	66.899263
7°	27.522488	546.901305	66.386058
8°	27.192757	554.708726	64.094123
9°	28.339267	549.157677	65.393501
10°	27.498977	547.440729	61.996648

Execução - 10000 **COM ERRO**

Execução	Generate (s)	Solve (s)	Register (s)
1°	33.114154	535.799407	43.946295
2°	33.202651	541.127521	47.962450
3°	33.062552	542.001453	48.675618
4°	33.110761	535.114704	44.623806
5°	33.146272	541.638319	44.412576
6°	34.466255	561.513665	46.756371
7°	33.346350	533.057264	41.692055
8°	33.355324	541.318380	47.200501
9°	33.700068	541.504962	64.692313
10°	33.432446	534.650485	43.165599

Obrigada!

lala.trindade.2008@gmail.com

https://github.com/LarissaTrin/CholeskyDecomposition

Referências

- http://lspd.mackenzie.br/marathon/18/problems.html
- http://lspd.mackenzie.br/marathon/18/problemset.pdf
- http://www.dma.uem.br/kit/topicos-especiais/cholesky.pdf
- http://wwwp.fc.unesp.br/~arbalbo/Iniciacao_Cientifica/sistemaslineares/teoria/3_Met_Cholesky.pdf
- https://www.ime.unicamp.br/~marcia/AlgebraLinear/fat_cholesky.html
- https://www.geeksforgeeks.org/cholesky-decomposition-matrix-decomposition/
- https://scistatcalc.blogspot.com/2020/05/matrix-utilities-cholesky-decomposition.html
- https://docente.ifrn.edu.br/julianaschivani/disciplinas/matematica-ii/matrizes/aplicacoes-praticas-de-sistemas-de-equacoes-lineares/view
- https://app.uff.br/riuff/bitstream/handle/1/22823/Joao%20Pedro.pdf?sequence=1&isAllowed=y