Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Computação Científica
Análise Numérica - Profa.: Sandra Augusta Santos
Alunos:
Giovanna Queiroz Gorri, RA: 198048
Beatriz Gomes da Silva, RA: 194631
Julia Machado Moretto, RA: 176953
Pedro Eduardo Faria Pietrafeza, RA: 185616

Projeto 1

Campinas
2020

Sumário

1 Introdução **III**

2 Desenvolvimento **III**
2.1 Item 1 III
2.2 Item 2 IV
2.3 Item 3 V
2.4 Item 4 VII
2.5 Item 5 VIII
2.6 Item 6 VIII

3 Apêndice **X**
3.1 Item 5 X

4 Conclusão **XII**

5 Referências **XII**

1 Introdução

Nesse projeto utilizamos os conceitos de matriz envelope, aplicando a estrutura de dados a algumas aplicações de matrizes esparsas.

2 Desenvolvimento

2.1 Item 1

Escrevemos o algoritmo no GNU Octave para podermos testá-lo computacionalmente.

```
X = zeros(n,1);

for i = 1:n
  X(i) = B(i);
 endfor

 X(n) = X(n)/DIAG(n);
 #percorre as colunas da ultima para primeira
 for i = n-1:-1:1
    #se os elementos da coluna nao forem todos nulos
   if ENV_col(i+2) != ENV_col(i+1)
     j = ENV_lin(ENV_col(i+1)) ;
     indice = ENV_col(i+1);
     #Para cada elemento da coluna atualizamos o vetor X
     while j <= i
       X(j) = X(j) -  (ENV(indice) * X(i+1)); #
       indice = indice +1;
       j= j+1;
     endwhile
  endif

   X(i) = X(i)/DIAG(i);
  endfor
```

Para testá-lo utilizamos a matriz abaixo:

$$\mathbf{A} = \begin{pmatrix} 12 & 15 & 0 & 20 & 0 & 0 \\ 0 & 56 & 12 & 0 & 0 & 0 \\ 0 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 0 & 5 \\ 0 & 0 & 0 & 0 & 23 & 0 \\ 0 & 0 & 0 & 0 & 0 & 16 \end{pmatrix}$$

Escrita na forma:

```
DIAG = [12, 56, 7, 7, 23, 16];
```

```
ENV = [15, 12, 20, 0, 0, 5,0,0];
ENV_col = [1, 1, 2, 3, 6, 6, 8];
ENV_lin = [1, 2, 1, 2, 3, 4, 5];
B = [1;5;2;3;8;7];
n=6;
```

Obtivemos

```
X =

  -0.145196
   0.028061
   0.285714
   0.116071
   0.347826
   0.437500
```

Que foi o mesmo valor obtido após rodarmos

```
A = [12, 15, 0, 20, 0, 0;
0, 56, 12, 0, 0, 0;
0, 0, 7, 0, 0, 0;
0, 0, 0, 7, 0, 5;
0, 0, 0, 0, 23, 0;
0, 0, 0, 0, 0, 16];

B = [1;5;2;3;8;7];
X = zeros(1,6);
X=A\B;
```

2.2 Item 2

Queríamos representar a matriz A como uma matriz envelope:

$$\mathbf{A} = \begin{pmatrix} a_{11} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 & a_{25} & 0 & 0 \\ 0 & 0 & a_{33} & 0 & 0 & 0 & 0 \\ 0 & a_{42} & 0 & a_{44} & 0 & a_{46} & 0 \\ 0 & 0 & 0 & 0 & a_{55} & 0 & 0 \\ 0 & a_{62} & 0 & 0 & 0 & a_{66} & a_{67} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{77} \end{pmatrix}$$

Então teremos:

$$\mathbf{DIAG} = \begin{pmatrix} a_{11} & a_{22} & a_{33} & a_{44} & a_{55} & a_{66} & a_{77} \end{pmatrix}$$

$$\mathbf{ENV_{sup}} = \begin{pmatrix} a_{25} & 0 & 0 & a_{46} & 0 & a_{67} \end{pmatrix}$$

$$\mathbf{ENV_{COLsup}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 4 & 6 & 7 \end{pmatrix}$$

$$\mathbf{ENV_{LINsup}} = \begin{pmatrix} 2 & 3 & 4 & 4 & 5 & 6 \end{pmatrix}$$

$$\mathbf{ENV_{inf}} = \begin{pmatrix} a_{42} & 0 & a_{62} & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{ENV_{LINinf}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 3 & 3 & 7 & 7 \end{pmatrix}$$

$$\mathbf{ENV_{COLinf}} = \begin{pmatrix} 2 & 3 & 2 & 3 & 4 & 5 \end{pmatrix}$$

2.3 Item 3

Podemos escrever a fatoração LU de uma matriz pelo algoritmo abaixo.

Algoritmo 1: Fatoração LU, Dolittle

Dados: A = n x n, L = identidade de ordem n, U = matriz de 0's n x n
Resultado: A = LU

1 **para** *i = 1 a n* **faça**
2 **para** *j = 1 a n* **faça**
3 $$U_{ij} = \frac{A_{ij} - \sum_{k=1}^{i-1} L_{ik}U_{kj}}{L_{ii}}$$
4 OBS:
$$L_{ii} = 1$$
5 **fim**
6 **para** *j = 1 a n* **faça**
7 $$L_{ji} = \frac{A_{ji} - \sum_{k=1}^{i-1} L_{jk}U_{ki}}{U_{jj}}$$
8 **fim**
9 **fim**

Para mostrar que os envelopes se conservam na fatoração, façamos por indução.

Usaremos o algoritmo acima para calculas os termos de L e U da demonstração abaixo.

Inicialmente com o envelope da porção superior de A e a matriz U. Chamaremos de Z o conjnto de posições dos elementos que não fazem parte do envelope na porção superior de A.

Se i = 1

$$U_{1j} = \frac{A_{1j} - 0}{L_{ii}} = \frac{A_{1j} - 0}{1} = A_{1j}$$

Portanto, se

$$A_{1j} = 0,$$

ou seja, A_{1j} está em Z,

$$U_{1j} = 0$$

e também está em Z.

Hipótese de indução
Se todos $A_{lj} = 0$ Para $l < i$, todos

$$U_{ij} = 0,$$

ou seja se A_{lj} está em Z, todos U_{ij} também estão em Z.

Faremos agora para l+1

$$U_{l+1j} = \frac{A_{l+1j} - \sum_{k=1}^{l} L_{ik}U_{kj}}{1}$$

Se A_{l+1j} está em Z, sabemos que A_{lj} também está em Z.
Da hipótese de indução temos que se A_{lj} está em Z, todos U_{kj} do somatório acima são 0.
Portanto

$$\sum_{k=1}^{l} L_{ik}U_{kj} = 0$$

Logo,

$$U_{l+1j} = \frac{A_{l+1j}}{1}$$

$$U_{l+1j} = A_{l+1j}$$

Portanto se

$$A_{l+1j} = 0$$

e A_{l+1j} está em Z, temos que

$$U_{l+1j} = 0$$

e também está em Z.
Agora faremos com o envelope da porção inferior de A e a matriz L. Chamaremos de W o conjnto de posições dos elementos que não fazem parte do envelope na porção inferior de A.

Seguindo a mesma lógica da porção superior
Se i = 1

$$L_{ji} = \frac{A_{j1} - 0}{U_{jj}} = \frac{A_{1j} - 0}{U_{jj}} = \frac{A_{1j}}{U_{jj}}$$

Portanto, se

$$A_{j1} = 0,$$

ou seja, A_{j1} está em W,

$$L_{j1} = 0$$

e também está em W.

Hipótese de indução
Se todos $A_{jl} = 0$ Para $l < i$, todos

$$L_{ji} = 0,$$

ou seja se A_{lj} está em W, todos L_{ji} também estão em W.

Faremos agora para l+1

$$L_{jl+1} = \frac{A_{jl+1} - \sum_{k=1}^{l} L_{jk}U_{ki}}{U_{jj}}$$

Se A_{jl+1} está em W, sabemos que A_{jl} também está em W.
Da hipótese de indução temos que se A_{jl} está em W, todos L_{jk} do somatório acima são 0.
Portanto

$$\sum_{k=1}^{l} L_{jk}U_{ki} = 0$$

Logo,

$$L_{jl+1} = \frac{A_{jl+1}}{U_{jj}}$$

Portanto se

$$A_{jl+1} = 0$$

e A_{jl+1} está em W temos que

$$L_{jl+1} = 0$$

e também está em W.

CQD

2.4 Item 4

Podemos escrever

$$B = PA$$

Se aplicarmos o algoritmo descrito no Item 3 a B, a demonstração de segue sem perda de

generalidade.
CQD

2.5 Item 5

Tentamos implementar a fatoração LU pelo algoritmo de Dolittle adaptado às estruturas de envelope. Já que pelo método de Dolittle a matriz L é percorrida por linhas e a U por colunas, assim como elas seriam armazenadas no final.
Infelizmente nossa habilidade de debugar algoritmos não foi muito boa, então nosso algoritmo não funcionou corretamente. A ideia dele foi a proposta no item 3. Segue no apêndice o algoritmo de fatoração LU que não funcionou.

Para testá-lo escrevemos matrizes na forma envelope e na forma padrão e aplicamos o segundo algoritmo do apêndice do item 5.
Para dar continuidade a resolução do sistema linear, utilizariamos as ideias de que

$$LY = B(1)$$

e em seguida:

$$UX = Y(2)$$

A parte (2) pode ser resolvida usando o algoritmo escrito no item 1 (que é funcional). Já a parte (1) pode ser resolvida com simples adaptações do algoritmo do item 1.

2.6 Item 6

Como o item 5 foi um infeliz caso de insucesso resolvemos o item 6 na forma padrão. Escrevemos o problema como:

$$AF = B$$

$$\mathbf{A} = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
\alpha & 0 & 0 & -1 & -\alpha & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
\alpha & 0 & 1 & 0 & \alpha & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
0 & 0 & 0 & 0 & \alpha & 1 & 0 & 0 & -\alpha & -1 \\
0 & 0 & 0 & & & & & & & \\
0 & 0 & 0 & 0 & \alpha & 0 & 1 & 0 & \alpha 0 & 0 \\
0 & 0 & 0 & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & -1 & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \alpha & 0 \\
0 & -\alpha & 0 & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha & 0 \\
1 & \alpha & 0 & & & & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -\alpha & 1 & & & & & & &
\end{pmatrix}$$

$$\mathbf{F} = \begin{pmatrix}
f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \\ f_{10} \\ f_{11} \\ f_{12} \\ f_{13}
\end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} 0 \\ 10 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 15 \\ 0 \\ 20 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Então aplicamos a função já prota do GNU Octave

```
1 F=A\B;
```

E obtivemos como resultado o vetor:

$$\mathbf{F} = \begin{pmatrix} -7.66032 \\ -15.41667 \\ 10.00000 \\ -0.83333 \\ -6.48181 \\ -15.41667 \\ 29.16667 \\ -0.83333 \\ -13.55288 \\ -10.41667 \\ 20.00000 \\ -14.73139 \\ -10.41667 \end{pmatrix}$$

3 Apêndice

3.1 Item 5

Algoritmo falho de fatoração LU

```
1  #percorre todas linhas/colunas
2  for i=1:n
3    #checa se linha e colunas nao sao nulas
4    if(ENVcol_sup(i) != ENVcol_sup(i+1))
5      if(ENVlin_inf(i) != ENVlin_inf(i+1))
```

```
6       #atualiza os indices das colunas e linhas para estarem no ↩
           elemento certo para fazer os calculos do U como no ↩
           algoritmo do item 3.
7       soma = 0;
8       dif = ENVlin_sup(ENVcol_sup(i)) - ENVcol_inf(ENVlin_inf(i));
9       if(dif < 0)
10        indice_sup = ENVcol_sup(i)- dif;
11        indice_inf = ENVlin_inf(i);
12        inicio =  ENVcol_inf(ENVlin_inf(i));
13      else
14       indice_inf = ENVlin_inf(i)+dif;
15        indice_sup = ENVcol_sup(i);
16        inicio = ENVlin_sup(ENVcol_sup(i));
17      endif
18      #Calcula os U que pertencem ao envelope de A sobrescrevendo ↩
           na porcao superior de A e diagonal
19      while (inicio < i)
20        ENVsup(indice_sup) = (ENVsup(indice_sup)-soma*ENVlin_inf(↩
             indice_inf));
21        soma = soma + ENVsup(indice_sup);
22        inicio = inicio+1;
23        indice_sup = indice_sup + 1;
24        indice_inf = indice_inf + 1;
25      endwhile
26     DIAG(i) = (DIAG(i) - soma);
27
28      # Atualiza os indices para calculo dos elementos de L
29      soma = 0;
30      j = i+1;
31      dif = ENVcol_inf(ENVlin_inf(i)) - j;
32      if(dif > 0)
33        j = ENVcol_inf(ENVlin_inf(i));
34      endif
35
36      dif = ENVlin_sup(ENVcol_sup(i)) - ENVcol_inf(ENVlin_inf(i));
37      if(dif < 0)
38        indice_sup = ENVcol_sup(i)- dif;
39        indice_inf = ENVlin_inf(i);
40        inicio =  ENVcol_inf(ENVlin_inf(i));
41      else
42       indice_inf = ENVlin_inf(i)+dif;
43        indice_sup = ENVcol_sup(i);
44        inicio = ENVlin_sup(ENVcol_sup(i));
45      endif
46      if (j<inicio)
47        j=inicio;
48      else
49        dif = j - inicio;
```

```
50          indice_sup = indice_sup + dif;
51          indice_inf = indice_inf + dif;
52        endif
53        #Calcula os L que pertencem ao envelope de A sobrescrevendo ↩
             na porcao inferior de A e considerando que a diagonal de ↩
             L tem termos todos de valor 1
54        while(j<n)
55          ENVinf(indice_inf) = (ENVinf(indice_inf)-soma*ENVcol_sup(↩
               indice_sup))/DIAG(j);
56          soma = soma + ENVsup(indice_sup);
57          j = j+1;
58          indice_sup = indice_sup + 1;
59          indice_inf = indice_inf + 1;
60        endwhile
61
62      endif
63    endif
64  endfor
```

Algoritmo que usamos para validar o criado por nós

```
1   L=eye(n,n);
2   for j=1:n-1
3       for i=j+1:n
4           L(i,j) = A(i,j)/A(j,j);
5           A(i,j+1:n) = A(i,j+1:n) - L(i,j)*A(j,j+1:n);
6           A(i,j)=0;
7       endfor
8   endfor
```

4 Conclusão

Aprendemos a utilizar a estrutura de matriz envelope em matrizes esparsa. Vimos como ela é eficiente no armazenamento, mas também como seus índices são confusos no momento de implementação.
Durante o desenvolvimento do projeto aprendemos muitas coisas interessantes envolvendo esses tipos de matrizes, como o algoritmo de Dolittle, representação de matrizes esparsas por grafos e representação de algumas fatorações por árvores.

Foram muitas ideias novas que nos foram apresentadas e novas maneiras de se pensar quanto ao tratamento de matrizes esparsas.

5 Referências

- http://www.cas.mcmaster.ca/~cs777/presentations/LU-factorization.pdf

- https://pt.wikipedia.org/wiki/Elimina%C3%A7%C3%A3o_de_Gauss#Etapa_3
- https://www.cs.cornell.edu/~bindel/class/cs6210-f12/notes/lec14.pdf
- http://www.karlin.mff.cuni.cz/~mirektuma/ps/direct.pdf
- https://en.wikipedia.org/wiki/LU_decomposition#Doolittle_algorithm
- https://www.geeksforgeeks.org/doolittle-algorithm-lu-decomposition/
- https://www.ufrgs.br/reamat/CalculoNumerico/livro-oct/sdsl-fatoracao_lu.html
- https://polignu.org/latex/exemplo/matrizes-em-latex

Acessos em 04/2020.