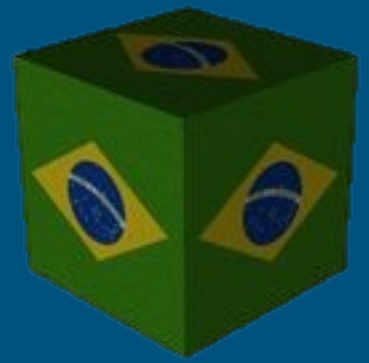
Trabalho de Métodos Numéricos I

Sistema de Equações

Equipe: Hexaedro Convexo

Integrantes:

- Claudemir Woche Vasconcelos (389575)
- Felipe Albuquerque Brito da Silva (386587)
- Francisco Yuri Martins (391379)
- Luiz Felipe Sousa Maciel (386597)
- Pedro Henrique Sousa da Silva (389577)
- Rodrigo Fabrício Meneses (376176)



Tema 4:

Uma determinada reação química produz uma quantidade *c* de *CO2* medida em ppm (parte por milhão) que pode variar dependendo das condições ambientais

Nesse caso, podem-se ter quantidades c_1, c_2, ..., c_n diferentes.

Essas quantidades podem ser calculadas a partir da solução de um sistema de equações lineares

$$Ac = d$$

Métodos:

- Fatoração *LU*
- Redução de *Doolittle*

Procedimento Fatoração LU

1. Considere um sistema linear $Ax = b$
2. Fatoramos a matriz A como produto de uma matriz L triangular inferior e uma matriz U triangular superior, ou seja, $A = LU$
3. Sendo assim, o sistema pode ser reescrito da seguinte forma:

$$Ax = b \rightarrow (LU)x = b \rightarrow L(Ux) = b \rightarrow Ly = b \quad e \quad Ux = y$$

4. Isto significa que, ao invés de resolvermos o sistema original, podemos resolver o sistema triangular inferior $Ly = b$ e, então, o sistema triangular superior $Ux = y$

Algoritmo Fatoração Lu

```
Matrix Solver::successiveSubstitution(const Matrix& lower, const Matrix& b) {

    Matrix y = Matrix(lower.getRow(), 1);

    y(0, 0, b(0,0)/lower(0,0));

    for(int i=1; i < b.getRow(); i++) {
        for(int j=0; j < i; j++)
            y(i, 0, y(i, 0) + lower(i, j)*y(j, 0));

        y(i, 0, (b(i,0) - y(i,0))/lower(i,i));
    }

    return y;

}

Matrix Solver::retroSubstitution(const Matrix& upper, const Matrix& y) {

    int n = upper.getRow();
    Matrix x = Matrix(n, 1);

    x(n-1, 0, y(n-1, 0)/upper(n-1, n-1));

    for(int i=n-2; i >= 0; i--) {
        for(int j=n-1; j > i; j--)
            x(i, 0, x(i, 0) + upper(i, j)*x(j, 0));

        x(i, 0, (y(i, 0) - x(i, 0))/upper(i, i));
    }

    return x;

}
```

Procedimento Redução de Doolittle

1. Primeiro multiplica-se a primeira linha de L pela j-ésima coluna de U e iguala-se a a_{1j}, obtendo-se os elementos u_{1j}.

2. Depois multiplica-se a i-ésima linha de L pela primeira coluna de U, igualando-se a a_{i1} de onde se obtém os elementos l_{i1}.

3. Repete-se o processo para as demais linhas e colunas até se obter os demais elementos de L e U.

Algoritmo DooLittle

```
std::vector<Matrix> Solver::luDoLittle(const Matrix& matriz) {
    int n = matriz.getRow();
    int m = matriz.getCol();
    std::vector<Matrix> lu(2);
    if(n == m) {
        Matrix lower(n, n);
        Matrix upper(n, n);
        double sum;
        for(int i=0; i < n; i++)
            for(int j =0; j < n; j++)
                if(i == j) lower(i, j, 1);
        for(int i=0; i < n; i++) {
            for(int k=i; k < n; k++) {
                sum = 0;
                for(int r=0; r < i; r++)
                    sum += lower(i, r)*upper(r, k);
                upper(i, k, matriz(i, k) - sum);
            }
            for(int k=i; k < n; k++) {
                sum = 0;
                for(int r=0; r < i; r++)
                    sum += lower(k, r)*upper(r, i);
                lower(k, i, (matriz(k, i) - sum) / upper(i, i));
            }
        }
        std::cout << "L = \n" << lower << "\n";
        std::cout << "U = \n" << upper << "\n";
        std::cout << "LU = \n" << lower*upper << "\n";
        lu[0] = lower;
        lu[1] = upper;

        return lu;
    }
    else {
        std::cout << "número de colunas diferente do numero de linhas\n";
        std::cout << "col = " << m << "\n";
        std::cout << "row = " << n << "\n";
    }
```

Pivoteamento parcial para fatoração LU

Na estratégia de pivoteamento parcial, antes de iniciar o j-ésimo estágio, permutam-se linhas da matriz A_{j-1} de modo a obter

$$\left|{a_{jj}}^{j-1}\right| \geq \left|{a_{ij}}^{j-1}\right| para\ todo\ i = j, ..., n$$

Em palavras, o pivô é escolhido como sendo um dos elementos de maior valor absoluto dentre

$${a_{jj}}^{j-1}, {a_{j+1j}}^{j-1}, \cdots, {a_{nj}}^{j-1}$$

Pivoteamento

```
MATRIZ K = 0
   7.00000   30.00000    8.00000
   9.00000    8.00000   30.00000
  20.00000    7.00000    9.00000

troca linha 1 por linha 3

MATRIZ K = 1
  20.00000    7.00000    9.00000
   9.00000    8.00000   30.00000
   7.00000   30.00000    8.00000

troca linha 2 por linha 3

MATRIZ K = 2
  20.00000    7.00000    9.00000
   7.00000   30.00000    8.00000
   9.00000    8.00000   30.00000

matriz pivoteada =
  20.00000    7.00000    9.00000
   7.00000   30.00000    8.00000
   9.00000    8.00000   30.00000
```

Análise dos procedimentos

Fatoração LU para o Exemplo

- Tema 4 :

$$[A] = \begin{bmatrix} 20 & 7 & 9 \\ 7 & 30 & 8 \\ 9 & 8 & 30 \end{bmatrix} \qquad \{d\} = \begin{bmatrix} 16 \\ 38 \\ 38 \end{bmatrix} \qquad \{c\} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

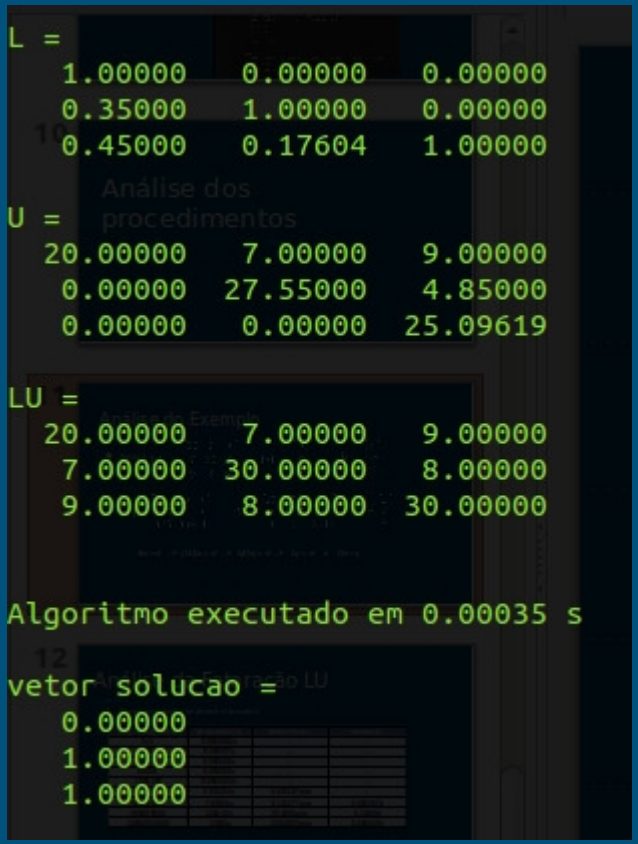
$$[L] = \begin{bmatrix} 1 & 0 & 0 \\ 0.35 & 1 & 0 \\ 0.45 & 0.18 & 1 \end{bmatrix} \qquad [U] = \begin{bmatrix} 20 & 7 & 9 \\ 0 & 27.55 & 4.85 \\ 0 & 0 & 25.10 \end{bmatrix} \qquad [y] = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Ac = d → (LU)c = d → L(Uc) = d → Ly = d e Uc = y

Solução do Exemplo

- Quantidades de CO2 em ppm :

$$\begin{Bmatrix} c_1 \\ c_2 \\ c_3 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 1 \\ 1 \end{Bmatrix}$$



Análise da Fatoração LU com pivoteamento

- Tempo em relação ao tamanho da matriz

TAM. DA MATRIZ	SEGUNDO(S)	MINUTO(S)	HORA(S)
3x3	0.0000067s	-	-
5X5	0.000070s	-	-
10X10	0.000150s	-	-
50X50	0.005217s	-	-
100X100	0.019179s	-	-
500X500	1.82641s	0,030440s	-
1000X1000	14.6837s	0,24472min	0,004078h
5000X5000	1824.17s	30,4028min	0,506713h
10000X10000	14586.9s	243.115min	4,051916h

Análise da Redução de Dolittle

- Tempo em relação ao tamanho da matriz

TAM. DA MATRIZ	SEGUNDO(S)	MINUTO(S)	HORA(S)
3x3	0.0000082s	-	-
5X5	0.000097s	-	-
10X10	0.000208s	-	-
50X50	0.001905s	-	-
100X100	0.0083454s	-	-
500X500	0.911254s	0.015187min	-
1000X1000	7.87655s	0.131277min	0.002187h
5000X5000	1202.43s	20.0405min	0.33400h
10000X10000	13205s	220.0833min	3.668055h

Conclusão

1. Para o exemplo não foi necessário o pivoteamento parcial.
2. Redução de DooLittle é mais rápido que a Fatoração LU.
3. Caso necessário mudar o vetor {d}, não é necessário aplicar todo o método novamente.
4. Ajuda a simplificar os cálculos, dependendo dos valores de [A] e {d}.
5. Como temos que calcular 2 substituições, para matrizes grandes o método é demorado.
6. Número de passos é sempre fixo.

Metodologia

- Programação C++
- IDE : Clion, Visual Studio Code
- Sistema Operacional : Linux – Ubuntu
- Computador : Core i3(5º geração), 4gb de memória RAM.

Fontes

- Ruggiero, Marcia A. Gomes, Cálculo Numérico.
- Site: www.math.tecnico.ulisboa.pt/~calves/courses/sis-lin/capii12.html
- Site: https://www.ufrgs.br/numerico/livro/sdsl-fatoracao_lu.html
- Site: http://www.ime.unicamp.br/~valle/Teaching/2015/MS211/Aula7.pdf