

# Comparação entre Julia e outras linguagens de programação na eficiência de execução do método de Newton-Raphson para solução de sistema de equações não-lineares

André Rodrigues Bezerra Madruga  
Bruno Matias de Sousa  
José Ricardo Bezerra de Araújo  
Levy Gabriel da Silva Galvão

15 de novembro de 2018

# Sumário

- 1 Introdução
- 2 Objetivos
- 3 Implementação
- 4 Resultados
- 5 Conclusões

# Introdução

- Vários problemas sem solução analítica;

# Introdução

- Vários problemas sem solução analítica;
- Necessidade de métodos iterativos;

# Introdução

- Vários problemas sem solução analítica;
- Necessidade de métodos iterativos;
- Computação numérica;

# Introdução

- Vários problemas sem solução analítica;
- Necessidade de métodos iterativos;
- Computação numérica;
- Várias linguagens de programação;

# Introdução

- Vários problemas sem solução analítica;
- Necessidade de métodos iterativos;
- Computação numérica;
- Várias linguagens de programação;
- Julia: recente e eficiente.

# Objetivos

- Solução de sistemas de equações não-lineares pelo método de Newton-Raphson;



# Objetivos

- Solução de sistemas de equações não-lineares pelo método de Newton-Raphson;
- Utilizar *Fortran 95*, *Julia* e *Python*;

# Objetivos

- Solução de sistemas de equações não-lineares pelo método de Newton-Raphson;
- Utilizar *Fortran 95*, *Julia* e *Python*;
- Comparar a eficiência de execução do algoritmo por cada linguagem;

# Sistemas de equações não-lineares

## Sistema 1:

$$\textcircled{1} \quad x_2 + x_3 - e^{-x_1} = 0$$

$$\textcircled{2} \quad x_1 + x_3 - e^{-x_3} = 0$$

$$\textcircled{3} \quad x_1 + x_2 - e^{-x_3} = 0$$

## Sistema 2:

$$\textcircled{1} \quad \frac{1}{2} \text{sen}(x_1 x_2) - \frac{x_2}{4\pi} - \frac{x_1}{2} = 0$$

$$\textcircled{2} \quad \left(1 - \frac{1}{4\pi}\right)(e^{2x_1} - e) - \frac{ex_2}{\pi} - 2ex_1 = 0$$

## Configurações do *hardware* e *software*

- Sistema Operacional Microsoft Windows 10 Home Single Language
- Processador Intel(R) Core(TM) i5-8250U Quadri Core 2.5 GHz com Turbo Max até 3.4 GHz
- Placa de vídeo dedicada Geforce MX150 2 GBIntel® HD Graphics 620
- Memória RAM 8GB DDR4 2133 MHz
- Disco rígido (HD) 1 TB 5400 RPM

## Fluxograma

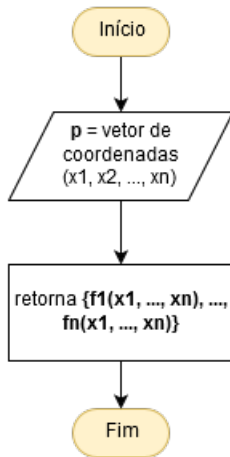


Figura: Fluxograma da função  $f()$ .

## Fluxograma

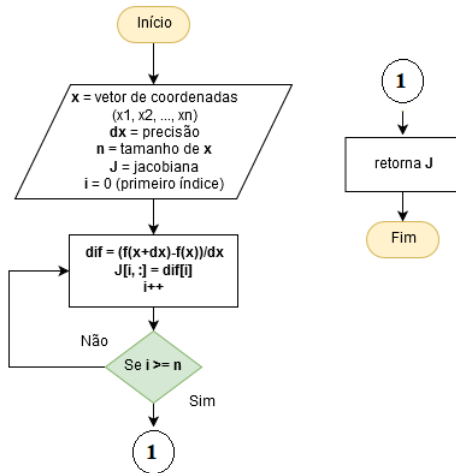


Figura: Fluxograma da função jac().

## Fluxograma

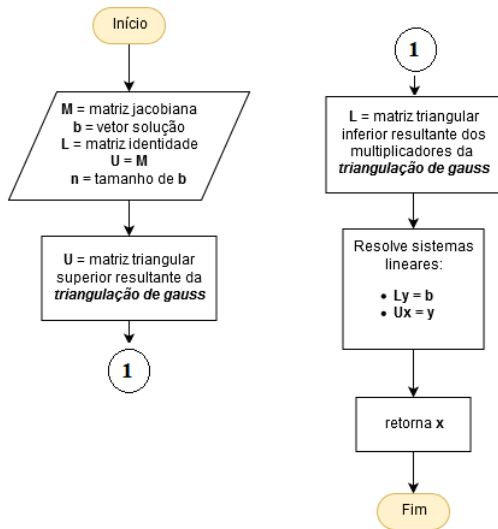


Figura: Fluxograma da função LU().

# Fluxograma

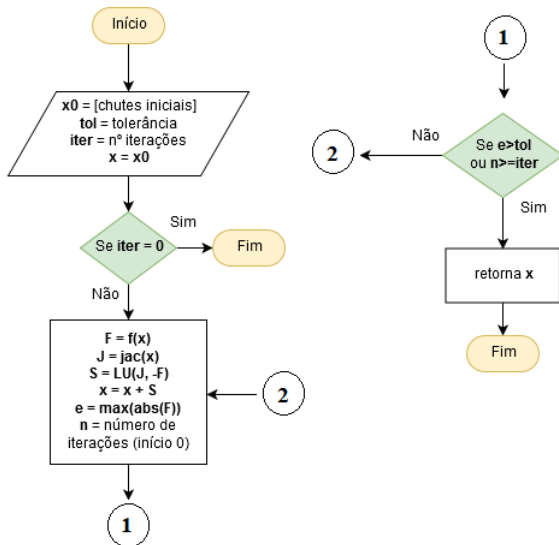


Figura: Fluxograma da função new\_rap().



## Fluxograma

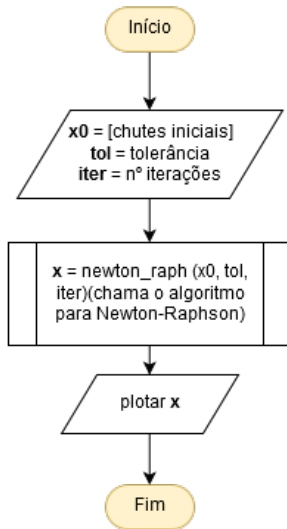


Figura: Fluxograma da main.



Figura: Marca abnTeX2. Fonte: <<http://www.abntex.net.br/>>

## Tempo de execução

	Solução
sis_1_sol1	[0.35173371, 0.35173371, 0.35173371]
sis_1_sol2	[-0.83202504, 1.14898375, 1.14898375]
sis_2_sol1	[0.11116545 -2.26144905]
sis_2_sol2	[-0.52596071, 0.78465031]
sis_2_soln	[-4.28925635, 24.05879992]

Figura: Tabela de soluções para cada sistema.

## Tempo de execução

	Fortran <i>gcc 5.1.0</i>	Julia <i>1.0.1</i>	Python <i>3.7</i>	Chute inicial
sis_1_sol1	1.032	1	1.258	[0.5, 0.5, 0.5]
sis_1_sol2	1.002	1	1.444	[0, 1, 2]
sis_2_sol1	1.167	1	1.060	[0.5, -2]
sis_2_sol2	0.948	1	1.156	[0, 0]
sis_2_soln	1.169	1	1.175	[0.25, 0.25]

Figura: Tabela de comparação de tempo de execução relativo, tendo o tempo de Julia como 1.

## Tempo de execução

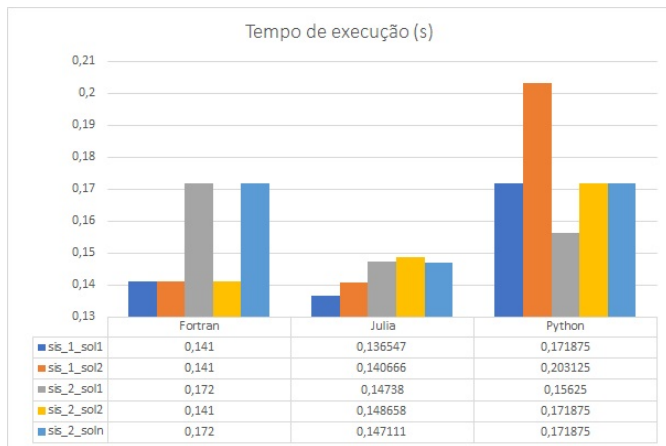


Figura: Gráfico e tabela de comparação do tempo de execução absoluto.

## Tempo de execução

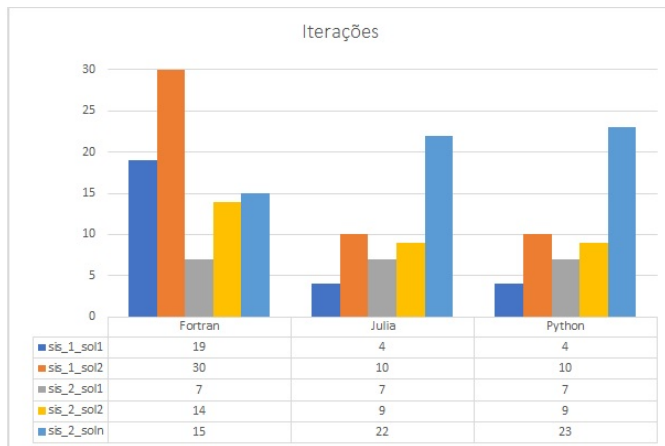


Figura: Tabela de comparação de iterações.

## Conclusões

- Vários problemas sem solução analítica;
- Necessidade de métodos iterativos;
- Computação numérica;
- Várias linguagens de programação;
- Julia: recente e eficiente.