**SME0602 - Cálculo Numérico – 1º semestre 2020**

**Prof. Elias Salomão Helou Neto**

**Projeto Prático 1**

**Zeros de Funções de Uma Variável**

**Alunos:**

**Paulo Katsuyuki Muraishi Kamimura 10277040**

**Guilherme Eiji Ichibara 10310700**

**Data: 30/05/2020**

[1. Introdução 3](#_Toc41750654)

[2. Questões 3](#_Toc41750655)

[2.1. Método de Newton 3](#_Toc41750656)

[2.2. Método de Halley 4](#_Toc41750657)

[2.3. Implementações 5](#_Toc41750658)

# 1. Introdução

# 2. Questões

# 2.1. Método de Newton

O d-ésimo método de Householder é dado por:

(1)

Substituindo em (1) temos:

(2)

Usando a regra do quociente em :

(3)

Substituindo (3) em (2):

Cortando o elemento em comum no numerador e denumerador obtemos:

# 2.2. Método de Halley

Substituindo em (1) temos:

(4)

Sabemos dado pela equação (3), para calcular derivamos novamente (3) também utilizando regra do quociente:

(5)

Obs: No passo acima foi necessário realizar a regra da cadeia na hora de realizar , resultando em .

Substituindo (3) e (5) em (4):

Cortando os termos e 2 do numerador e enumerador:

Distribuindo os termos para deixar com uma aparência da equação do enunciado do trabalho:

# 2.3. Implementações

A implementação se baseia nos arquivos *main.c, equations.c e methods.c.*

* **main.c** – função principal que invoca a execução de todos os métodos e também responsável por salvar os resultados num .csv para facilitação da transpor os dados para uma planilha excel. Vale ressaltar que tal algoritmo não é necessário para a realização dos cálculos das raízes, só foi feito para facilitar a leitura dos dados.
* **equations.c** – contém a definição das equações 1, 2 e 3 apresentado no enunciado. Cada função contém dois parâmetros, a variável x e a ordem de sua derivada, variando assim qual equação ele retornará.

|  |
| --- |
| double f(double x, int i){  switch (i){  case 0:  return (x - cos(x));  case 1:  return (1 + sin(x));  case 2:  return (cos(x));  default:  return -1;  }  } |

|  |
| --- |
| double g(double x, int i){  switch (i){  case 0:  return (x\*x\*x - 9\*x\*x + 27\*x -27);  case 1:  return (3\*x\*x-18\*x+27);  case 2:  return (6\*x-18);  default:  return -1;  }  } |

|  |
| --- |
| double h(double x, int i){  switch (i){  case 0:  return (exp(x) - cos(x));  case 1:  return (exp(x) + sin(x));  case 2:  return (exp(x) + cos(x));  default:  return -1;    } |

* **methods.c** – contém a implementação dos métodos para aproximação de raízes: bissecção, secante, Newton e Halley. Cada método recebe como parâmetro os valores iniciais e dependendo de qual método pode receber dois valores iniciais -bissecção e secante.

Também recebe como um parâmetro um vetor para armazenar todos os resultados a fim de possibilitar a utilização desses dados posteriormente. Por fim, a função de cada implementação dos métodos retorna à quantidade de iterações que foram necessárias para cumprir os critérios de tolerância absoluta determinado no enunciado da questão.

A fim de cumprir tal critério foi determinado uma condição de parada implementada na condição de uma rotina do...while. Temos a seguir o um exemplo da implementação do método de newton aplicado à equação 1.

# 2.3.1 Execução do programa

Para compilar o programabasta abrir o terminal na pasta do projeto e executar o seguinte comando para compilar os arquivos.c.

|  |
| --- |
| gcc main.c equations.c methods.c -o main |

Para executar o programaexecute o seguinte comando no terminal aberto na pasta do projeto.

|  |
| --- |
| ./main |

Após a execução do programa será criado três arquivos .csv na pasta do projeto, sendo eles resultados1.csv, resultados2.csv e resultados3.csv. Cada um contendo os resultados dos quatro métodos. Resultado1 são os dados da primeira equação, resultado2 são os dados da segunda equação assim por diante.

Vale ressaltar alguns pontos. A função responsável por converter os resultados já calculados e armazenados em um vetor para um arquivo .csv é opcional e foi somente uma forma de facilitar a visualização dos dados e não faz parte da implementação dos cálculos e dos métodos em si.

Foi discutido se isso influenciaria no desempenho do método numérico e chegou-se à conclusão que o registro dos dados não significa que para o método realizou cálculos desnecessários para a obtenção das raízes.

# 2.4. Uso das implementações