INSTITUTO FEDERAL DE MINAS GERAIS

Bacharelado em Ciência da Computação

Disciplina: Matemática Computacional

Trabalho Prático

Professor: Diego Mello da Silva

Formiga-MG
8 de maio de 2018

Sumário

1 Introdução

Este documento apresenta a especificação do trabalho prático da disciplina de Matemática Computacional, no valor de 30 pontos.

O trabalho consiste na implementação de um software capaz de calcular a integral numérica de uma função tabelada. O software utilizará um método de interpolação polinomial cujo polinomio interpolador é determinado com uso de sistemas lineares. De posse do polinômio o software implementará uma variante do método de integral numérica baseado em aproximação por trapézios. O software será implementado utilizando a linguagem C. O trabalho deverá ser testado em ambiente Linux Ubuntu sendo compilado nos compiladores GNU C Compiler (`gcc`).

O trabalho poderá ser feito em grupo de **até 2 alunos**. O prazo para entrega do trabalho é de 4 semanas contadas a partir da data em que for anunciado. O trabalho deverá ser postado no sistema acadêmico, na atividade correspondente da disciplina de Matemática Computacional. Deve-se informar os membros do grupo e matrícula.

A seção 6 detalhará os requisitos da aplicação que deverá ser entregue neste trabalho prático.

2 Regra do Trapézio

O cálculo de integral numérica pela regra do trapézio consiste em, dados uma função $f(x)$ e um intervalo $\big[a, b\big]$, estimar a área sobre a curva $f(x)$ aproximando-a por meio da área de um ou mais trapézios. Quanto mais trapézios forem adotados, melhor será a aproximação da curva.

Para exemplificar, seja a integral definida

$$\int_{0.4}^{2.0} e^x \sin(10x) + 8\, dx = 12.48287$$

As figuras Fig 1 a Fig 6 apresentam o gráfico da função no intervalo $\big[0.4, 2.0\big]$, a área do(s) trapézio(s) usado(s) na integração numérica e o valor da aproximação de $f(x)$ para 1, 2, 5, 10, 25 e 100 trapézios. Os resultados numéricos estão disponíveis na tabela a seguir. Observe que quanto mais trapézios são empregados na integração, mais o resultado calculado se aproxima do resultado analítico 12.48287, e menor é o erro absoluto entre o valor analítico e o valor estimado.

No. Trapézios	Int. Numérica	Erro Absoluto
1	17.29342	4.81055
2	13.62152	1.13865
5	13.01824	0.53537
10	12.59091	0.10804
25	12.49932	0.01645
100	12.48389	0.00102

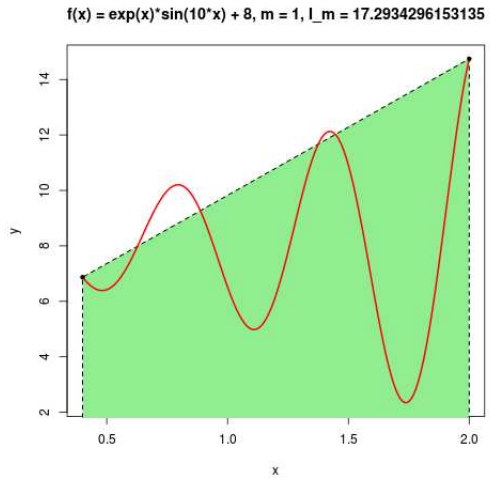

Figura 1: $m = 1, \int f(x) \approx 17.29342$

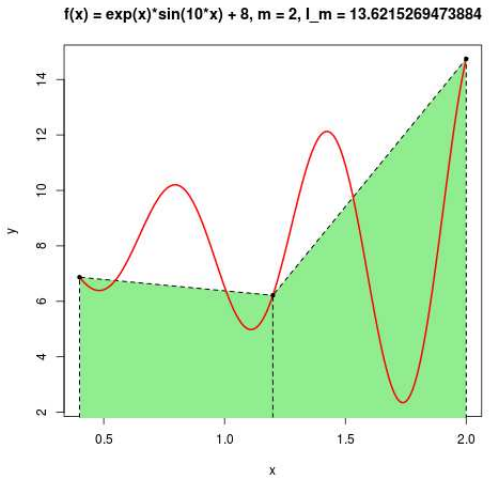

Figura 2: $m = 2, \int f(x) \approx 13.62152$

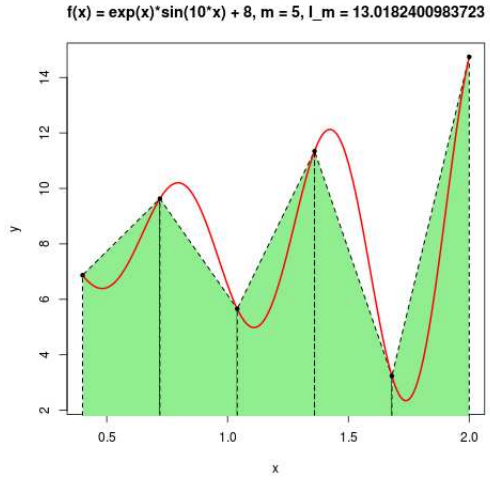

Figura 3: $m = 5, \int f(x) \approx 13.01824$

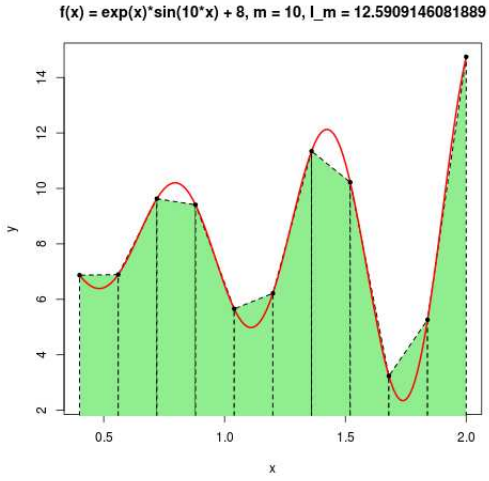

Figura 4: $m = 10, \int f(x) \approx 12.59091$

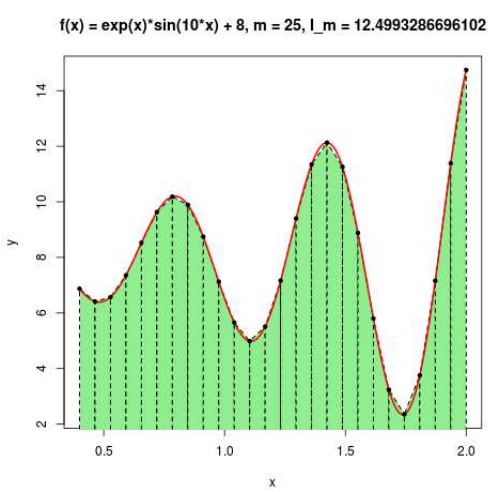

Figura 5: $m = 25, \int f(x) \approx 12.49932$

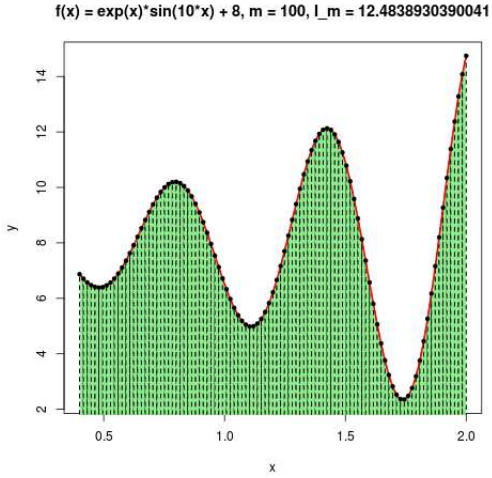

Figura 6: $m = 100, \int f(x) \approx 12.48389$

3 Interpolação Polinomial

Interpolação é uma ferramenta útil quando se dispõe de uma tabela de valores (x_i, y_i), $i = 1, 2, \ldots, n$ observados e de onde existe a necessidade de se obter um valor intermediário que não consta nesta tabela. Interpolar um valor x_i significa calcular o valor de $f(x_i)$, inexistente na tabela de valores conhecidos. Exemplo: qual é a estimativa de $f(5)$, se o ponto $\big(5, f(5)\big)$ não consta na tabela a seguir?

x	$f(x)$
1	30
4	37
7	50
9	22
10	17

Muitas vezes a tabela é obtida de dados experimentais ou de campo, tabelas estatísticas e funções complexas onde não se conhece sua forma analítica. Para estes casos, não se pode calcular $f(x_i)$ formalmente, pois $f : \mathbb{R} \to \mathbb{R}$ é desconhecida. A solução consiste em determinar, por meio de interpolação, um polinômio $p_n(x)$ de grau menor ou igual a n **que passa por todos os pontos da tabela** e de onde pode-se calcular o valor de $p_n(x)$. Chamamos este polinômio de **polinômio interpolador**.

Uma das aplicações de sistemas lineares consiste em, dado um conjunto de $(n + 1)$ valores de x e $y = f(x)$, determinar um polinômio de grau n que passa por todos os pontos $\big(x, y\big)$.

Para ilustrar a técnica, observe o exemplo a seguir. Sejam três pontos base (x_0, y_0), (x_1, y_1) e (x_2, y_2), com x_i distintos de uma função $f(x)$. O polinômio interpolador $p_n(x)$ é tal que passa pelos três pontos apresentados.

Logo, é fato que

$$y_0 = a_0 + a_1(x_0) + a_2(x_0)^2$$

$$y_1 = a_0 + a_1(x_1) + a_2(x_1)^2$$

$$y_2 = a_0 + a_1(x_2) + a_2(x_2)^2$$

pois se o polinômio interpolador $p_n(x)$ passa pelos três pontos, então os parâmetros a_0, a_1 e a_2 são comuns às três igualdades acima. Neste exemplo, o polinômio $p_n(x)$ é uma função quadrática. Os valores de x_i e y_i são constantes e dados pela tabela, mas os valores de a_0, a_1 e a_2 são desconhecidos, ou seja, são as incógnitas do problema.

Observando as igualdades acima concluímos que todas as equações são lineares (pois as incógnitas a_0, a_1 e a_2 são todas de grau 1), de forma que estas incógnitas podem ser encontradas mediante a resolução do seguinte sistema de equações lineares:

$$\begin{cases} a_0 + a_1(x_0) + a_2(x_0)^2 = y_0 \\ a_0 + a_1(x_1) + a_2(x_1)^2 = y_1 \\ a_0 + a_1(x_2) + a_2(x_2)^2 = y_2 \end{cases}$$

Na forma matricial, temos:

$$\begin{bmatrix} 1 & x_0 & (x_0)^2 \\ 1 & x_1 & (x_1)^2 \\ 1 & x_2 & (x_2)^2 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix}$$

Observe que na matriz de coeficientes acima, os termos de cada linha estão em progressão geométrica. Tal matriz é conhecida na álgebra linear como **matriz de Vandermonde**. Pelos três pontos passa um único polinômio de grau 2; esta afirmação pode ser generalizada, dizendo que por n pontos passa um único polinômio de grau $(n-1)$.

De maneira mais geral, os coeficientes a_0, a_1, ..., a_n de um polinômio interpolador de grau $(n-1)$ podem ser obtidos mediante a resolução do seguinte sistema linear:

$$\begin{bmatrix} 1 & x_0 & (x_0)^2 & (x_0)^3 & \dots & (x_0)^{(n-1)} \\ 1 & x_1 & (x_1)^2 & (x_1)^3 & \dots & (x_1)^{(n-1)} \\ 1 & x_2 & (x_2)^2 & (x_2)^3 & \dots & (x_2)^{(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & (x_n)^2 & (x_n)^3 & \dots & (x_n)^{(n-1)} \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{(n-1)} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{(n-1)} \end{bmatrix}$$

Resolvendo o sistema, encontra-se $\boxed{p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_{(n-1)}x^{(n-1)}}$. A partir de $p(x)$, pode-se fazer estimativas de qualquer valor no intervalo (x_0, x_n).

A Figura 7 ilustra o plot de um polinômio interpolador $p_4(x)$ que passa por 5 pontos tabelados no intervalo $\left[0.0, 1.0\right]$.

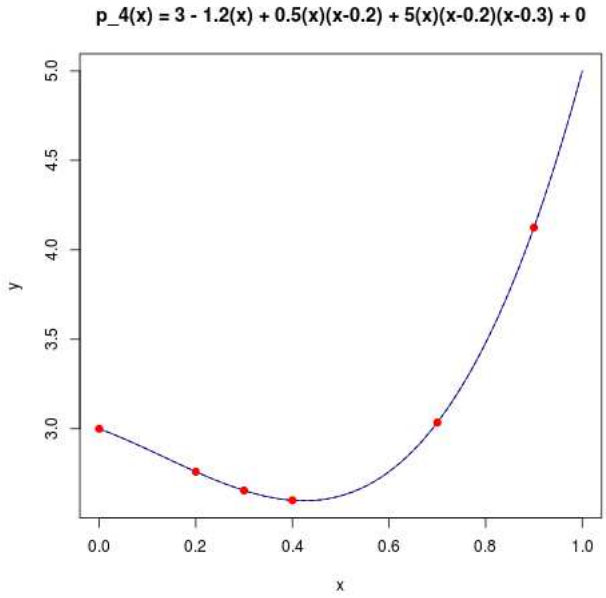

Figura 7: Exemplo de Polinômio Interpolador

4 Calculando a Integral de Funções Aproximadas

Nas Seções 2 e 3 foram dados os fundamentos que nos permitirão construir um software capaz de integrar funções matematicamente complexas aproximando-as por funções mais simples.

Seja $f(x)$ uma função difícil de integrar no intervalo $\left[a, b\right]$, com $a < b$. Tomando-se $(n + 1)$ pontos equidistantes entre sí no intervalo $\left[a, b\right]$ podemos determinar um polinômio interpolador $p_n(x)$ que aproxima $f(x)$ por uma função polinomial de grau n, independente se a função original contêm elementos não lineares, tais como componentes exponenciais, senoidais e logarítmicas.

De posse do polinômio interpolador $p_n(x)$ podemos calcular facilmente uma aproximação para qualquer valor de $x \in \left[a, b\right]$ simplesmente avaliando-se x por meio de $p_n(x)$. Computacionalmente, esta tarefa pode ser feita de maneira eficiente por meio do algoritmo de Horner.

Com uma quantidade suficientemente razoável de trapézios, pode-se calcular uma aproximação de integral numérica de qualquer função, polinomial ou não. Juntando-se ambos os conceitos, este projeto propõe a construção de um um software capaz de (i) aproximar qualquer função cujos pares ordenados (x, y), com $x \in \left[a, b\right]$ sejam informado por um polinômio interpolador e (ii) calcular a integral deste polinômio no intervalo $\left[a, b\right]$.

5 Recomendações Gerais

Toda a implementacao do aplicativo deverá ser feita em um único arquivo de código fonte, denominado `trapezium.c`. Para organizar o código, deve-se fazer uso de procedimentos e funções capazes de implementar os requisitos do *software*, especificados na seção 6.

6 Especificação

Esta seção apresenta as recomendações gerais e requisitos esperados para a implementação do aplicativo `trapezium` descrito neste trabalho prático de implementação. Detalhes sobre cada requisito, assim como o barema de correção serão apresentados ao longo da seção.

6.1 Requisito 01 - Entrada de Dados

A entrada e saída de dados será feita por meio de arquivos, informados como argumentos de linha de comando (ver no *slide* de introdução à disciplina como pode-se implementar em C a leitura de argumentos por linha de comando). A sintaxe esperada para o `trapezium` é:

```
c:\> trapezium.exe <arquivo entrada> <arquivo saída>
```

ou

```
user@machine:~$ trapezium.bin <arquivo entrada> <arquivo saída>
```

dependendo se a aplicação foi compilada em ambiente Windows ou Linux. Independente do caso, são requeridos dois parâmetros, que correspondem ao caminho do arquivo de entrada (ver formato no Requisito 01, descrito na seção 6.1) e ao caminho do arquivo de saída (ver formato no Requisito 03, na seção 6.3).

A entrada de dados do software `trapezium` será dada por arquivos de entrada, escritos no seguinte formato com cada linha do arquivo de entrada iniciando por um caracter de instrução que indica o conteúdo da linha correspondente. O formato dos arquivos de entrada é dado pelo modelo a seguir:

```
# <String. Linha de comentário, ignorada até o '\n'>
n <Inteiro. Número de pontos tabelados>
x <Real. Lista com (n) valores de 'x' da tabela>
y <Real. Lista com (n) valores de 'x' da tabela>
a <Real. Limite inf. do intervalo para calculo da integral definida>
```

```
b <Real. Limite sup. do intervalo para calculo da integral definida>
i <Inteiro. Número de pontos em [min(x), max(x)] a interpolar>
p <Real. Lista com (i) pontos a interpolar>
t <Inteiro. Numero de trapézios a implementar>
```

O arquivo descreve basicamente duas informações, a saber. Primeiro, ela lista todos os pontos tabelados de uma função mais complexa cujo interesse é saber a intergral definida de qualquer intervalo $\big[a, b\big]$ contido no intervalo $\big[\min(x), \max(x)\big]$. A segunda lista os pontos dentro deste intervalo que não estão tabelados e cujos valores interpolados são de interesse do usuário. Segue um exemplo de arquivo de entrada.

```
# Lista de pontos tabelados entre [1.6, 6.7]
x 1.6 2.5 3.0 4.2 5.4 6.7
y -1.5 -2.1 0.3 1.4 2.1 3.9
# Intervalo da integral definida em [2,5]
a 2
b 5
# Lista de pontos a interpolar
i 3
p 2.0 3.5 4.0
# Computar integral com 20 trapezios
t 20
```

O software deverá tratar todas as entradas possíveis, avisando o usuário caso o arquivo de entrada especificado esteja fora do formato esperado. Por exemplo, não é possível nem interpolar um ponto fora do intervalo $\big[\min(x), \max(x)\big]$, nem mesmo especificar um intervalo para cálculo da integral definida fora de $\big[\min(x), \max(x)\big]$. Se menos argumentos do que o esperado foram informados pelo usuário, então o software deverá informar tal situação no console.

IMPORTANTE: Junto com o arquivo de código fonte pede-se que o grupo envie 5 (cinco) arquivos de entrada diferentes, cujos valores tabelados foram obtidos de funções diversas. Usar os comentários para indicar a função original de onde a tabela foi obtida.

6.2 Requisito 02 - Saída de Dados (Console)

Neste requisito espera-se que o grupo implemente saída no console conforme o exemplo abaixo, em que os pontos 2.0, 3.5 e 4.0 são interpolados a partir do exemplo de entrada:

```
Trapezium: Interpolador/Integrador Numerico

Polinomio Interpolador:
```

```
P(x) = 136.7750819 + -213.5468033*x + 121.1971443*x^2 + -31.9880293*x^3 +
          3.9898198*x^4 + -0.1897431*x^5

Valores Interpolados:

P(2.0) = -3.668846
P(3.5) = 1.605210
P(4.0) = 1.605194

Integral em [2.0, 5.0] = 0.5987762
```

Em resumo, a saida no console deve apresentar o polinomio interpolador, os valores informados para interpolar, e o valor da integral definida de acordo com os parâmetros de entrada.

6.3 Requisito 03 - Saída de Dados (Script)

Além de gerar saída no console o aplicativo `trapezium` deverá gerar um arquivo de saída contendo um *script* na linguagem R para mostrar graficamente o resultado das integrações e interpolações. O modelo de construção do *script* é dado a seguir.

```
#########################################################################
# Script automatico gerado por 'trapezium', software de interpolação #
# e integracao numerica                                                 #
#########################################################################

#
# Parametros. Devem ser preenchidos pelo software
#

# Nome da figura
nome <- 'saida.png'

# Dados tabelados
x.tab <- c(1.6, 2.5, 3.0, 4.2, 5.4, 6.7);
y.tab <- c(-1.5, -2.1, 0.3, 1.4, 2.1, 3.9);

# Pontos interpolados, calculados pelo 'trapezium'
x.int <- c(2.0, 3.5, 4.0);
y.int <- c(-3.668846, 1.60521, 1.605194);

# Coeficientes do polinomio interpolador
coef <- c(136.7750819, -213.5468033, 121.1971443,
       -31.9880293, 3.9898198, -0.1897431);

# Numero de pontos da tabela
n.tab <- 6;

# Numero de pontos a interpolar
n.int <- 3;
```

```
# Numero de trapezios
n.tpz <- 20;

# Titulo
titulo <- 'P(x) = 136.7750819 + -213.5468033*x + 121.1971443*x^2 +
           -31.9880293*x^3 + 3.9898198*x^4 + -0.1897431*x^5'

#
# Esta parte do script deve funcionar desde que os parametros
# acima estejam devidamente preenchidos. E' a parte estatica
# do script. Copiar exatamente desta forma no arquivo de saida.
#

# Calcula o valor interpolado para o pto x
polinomio <- function(x, coef, n)
{
    resultado <- 0;
    for(i in 1:n)
    {
        resultado <- resultado + coef[i]*(x^(i-1));
    }
    return(resultado);
}

#
# Aqui comecam os comandos para plotar os resultados
#

# Cria o arquivo .png
png(nome);

# Gerando figura com 100 pontos
gap <- (max(x.tab) - min(x.tab)) / 100;
x <- seq(min(x.tab), max(x.tab), gap);
y <- polinomio(x, coef, n.tab);
plot(x,y,type='l', main=titulo);

# Plota os trapezios
h  <- (max(x.tab) - min(x.tab)) / n.tpz;
xp <- seq(min(x.tab), max(x.tab), h);
yp <- polinomio(xp, coef, n.tab);
for(i in 1:(n.tpz))
{
    polygon(c(xp[i], xp[i], xp[i+1], xp[i+1], xp[i]),
            c(0, yp[i], yp[i+1], 0, 0),
            col='yellow', border='black', lty=2, lwd=1.3);
}

# Pontos da tabela
for(i in 1:n.tab)
{
    points(x.tab[i], y.tab[i], col='red', pch=19);
}

# Pontos interpolados
```

```
for(i in 1:n.tab)
{
    points(x.int[i], y.int[i], col='blue', pch=19);
}

# Encerra a criacao do arquivo .png
dev.off();
```

O *script* acima permite gerar um arquivo no formato .png quando executado dentro do ambiente computacional R Studio. Observe que existe uma seção onde todos os parâmetros devem ser informados em arquivo, parâmetros estes que advém dos resultados calculados no `trapezium`. Preencha-os corretamente que o restante do *script* será capaz de realizar o plot. O arquivo acima representa uma possível saída para o arquivo de entrada ilustrado no Requisito 01.

A execução do *script* acima no ambiente R Studio irá gerar a Figura 8. Em amarelo estão os trapézios que formam a aproximação da integral numérica. Em vermelho, os pontos tabelados. Em azul, os pontos interpolados. A figura irá validar cada entrada de dados informada.

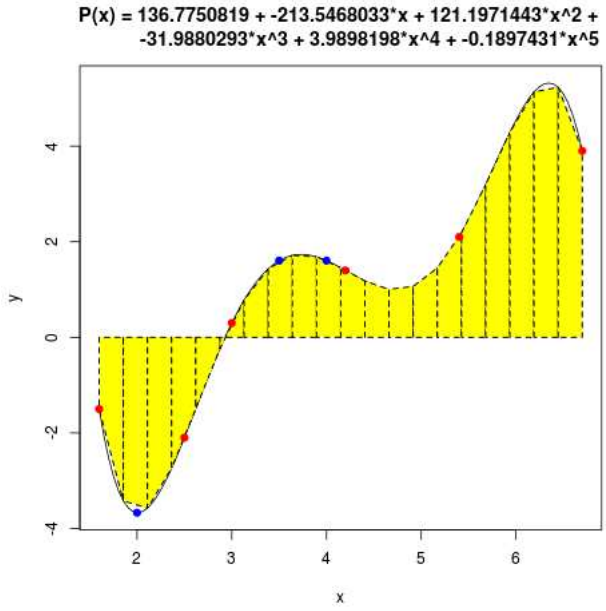

Figura 8: Exemplo de saída do *script* gerado por `trapezium`, com 20 trapézios

6.4 Requisito 04 - Determinação do Polinômio Interpolador

Esta seção descreve o requisito onde, dados os valores de x e $f(x)$ tabelados em arquivo de entrada deve-se determinar os coeficientes a_0, a_1, ..., $a_{(n-1)}$ do polinômio interpolador $\boxed{p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_{(n-1)}x^{(n-1)}}$.

Tais coeficientes serão determinados por meio da resolução do sistema linear descrito na Seção 3. Deve-se construir uma matriz A de Vandermonde com os

valores de x tabelados, um vetor b de termos independentes, e calcular o vetor x solução do sistema $Ax = b$ que guardará os valores de cada coeficiente de $p_n(x)$.

O método exigido para resolver o sistema linear $Ax = b$ será o método da **Eliminação de Gauss Pivotal**, que transforma o sistema $Ax = b$ em $Ux = d$.

6.5 Requisito 05 - Resolver $Ux = d$ por Gauss Pivotal

No Requisito 04 determina-se que o sistema $Ax = b$ que calcula os coeficientes do polinômio interpolador $p_n(x)$ seja resolvido pelo método da Eliminação de Gauss Pivotal. A Eliminação de Gauss Pivotal converte um sistema $Ax = b$ em um sistema triangular superior $Ux = d$, cujas incógnitas podem ser determinadas por meio de substituição retroativa. Desta forma, este requisito exige que o método de substituição retroativa seja implementado para obter os parâmetros de $p_n(x)$.

6.6 Requisito 06 - Avaliação de $p_n(x)$ usando Horner

O valor do polinômio interpolador que aproxima a função $f(x)$ deverá ser calculado para determinar os valores da integral definida $\int_a^b p_n(x)dx$ no intervalo $\left[a, b\right]$, assim como o valor de $p_n(x)$ para os pontos de interesse informados no arquivo de entrada.

Neste requisito pede-se para que tal avaliação seja realizada pelo algoritmo de Horner, que deverá receber a lista dos coeficientes de $p_n(x)$, o grau deste polinômio e o valor de x cujo $p_n(x)$ é requisitado via arquivo de entrada.

6.7 Requisito 07 - Cálculo da Área do Trapézio

Neste requisito o grupo deverá implementar uma função que calcula a área de um **único** trapézio. A área do trapézio depende das seguintes informações: comprimento da base maior (B), comprimento da base menor (b) e altura (h), relacionados na seguinte fórmula:

$$A = \frac{(B + b) \times h}{2}$$

A função deverá receber, portanto, os parâmetros acima modelados com os tipos de dados mais adequados, e retornar o valor da área calculada segundo a fórmula dada. Observe que os valores de B e b são obtidos da avaliação de $p_n(x)$, enquanto que o valor de h pode ser determinado pela quantidade de sub-divisões do intervalo $\left[a, b\right]$ informado.

6.8 Requisito 08 - Cálculo de $\int_a^b p_n(x)dx$

Neste requisito, o grupo deverá implementar uma função que calcula a integral definida no intervalo $\left[a, b\right]$ usando a função escrita no Requisito 06 que calcula a área do trapézio. Para tal ela receberá como parâmetros o intervalo $\left[a, b\right]$ e o número m de trapézios a utilizar. A função deverá então segmentar o intervalo $\left[a, b\right]$ original na quantidade de sub-intervalos igual ao número de trapézios informado via arquivo de entrada. O valor da aproximação da integral calculada pelo método dos trapézios

será o somatório das áreas de cada um dos m trapézios, ou seja, a função deverá calcular a área de cada trapézio em separado e acumular os resultados para obter a aproximação da integral definida.

6.9 Requisito 09 - Corretude dos Resultados

Para que o trabalho prático tenha cumprido com sua proposta é preciso que a implementação do aplicativo `trapezium` gere resultados corretos. Neste requisito deve-se verificar o resultado dos métodos para diferentes entradas de dados, algumas fornecidas pelo grupo (5 arquivos de entrada) e algumas usadas pelo professor da disciplina. Somente pontuará neste requisito o grupo que completar os testes com 100% de corretude nos resultados. **Recorde que é preciso fornecer os 5 (cinco) arquivos de entrada junto com o código fonte da aplicação**.

6.10 Requisito 10 - Documentação do Código

Este requisito lida com a documentação da implementação em código por meio de comentários. Para atingir o objetivo deste requisito o grupo deverá comentar o cabeçalho do arquivo de entrada, informando o propósito da implementação, comentar cada função, procedimento ou afim, indicando o objetivo do procedimento assim como as entradas e saídas esperadas, comentar as principais estruturas de dados da aplicação e indicar em cada procedimento qual é o número do requisito que ele implementa.

7 Barema

A tabela a seguir descreve o barema usado pelo professor da disciplina como orientação para correção. Ele contêm o valor de cada requisito do trabalho que, se completamente cumprido, totalizará 30 pontos. Cabe ressaltar que trabalhos fora do especificado (codificação, linguagem, compilação e ambiente) não serão corrigidos e valerão zero. Também serão desconsiderados trabalhos plagiados ou com alto grau de similaridade.

Requisito	**Ptos**
Req 01 - Entrada de Dados	1.0
Req 02 - Saída de Dados (Console)	1.0
Req 03 - Saída de Dados (Script)	6.0
Req 04 - Determinação do Polinômio Interpolador	4.0
Req 05 - Resolver $Ux = d$ por Gauss Pivotal	2.0
Req 06 - Avaliar $p_n(x)$ usando Horner	1.0
Req 07 - Cálculo da Área do Trapézio	1.0
Req 08 - Cálculo de $\int_a^b p_n(x)dx$	5.0
Req 09 - Corretude dos Resultados	8.0
Req 10 - Documentação do Código	1.0
TOTAL	**30.0**

8 Bibliografia Recomendada

Referências

[Ruggiero] RUGGIERO, Márcia; LOPES, Vera Lúcia da Rocha. Cálculo Numérico - Aspectos Teóricos e Computacionais. 2^a edição. Editora Pearson Makron, 1996. ISBN: 978-85-346-0204-4.

[Sperandio] SPERANDIO, Décio; MENDES, João Teixeira. Cálculo Numérico: Características Matemáticas e Computacionais dos Métodos Numéricos. Editora Pearson Prentice Hall, 2003. ISBN: 85-87918-74-5.

[Barroso] BARROSO, Leonidas; CAMPOS FILHO, Frederico Ferreira. Cálculo Numérico (Com Aplicações). 2^a edição. Editora Harbra, 1987. ISBN: 85-29400-89-5

[Campos] CAMPOS FILHO, Frederico Ferreira. Algoritmos Numéricos, 2^a edição. Editora LTC (Grupo GEN), 2007. ISBN: 85-21615-37-8.