Relatório EP3

Eduardo Brancher Urenha - 8587409[1]

[1]eduardo.urenha@usp.br

1. Introducao

O programa foi compilado com `gcc -o ep3 ep3.c -lm` e executado com `./ep3`. O programa executara os experimentos e informara o resultado no terminal.

Obs: Peco desculpas pela falta de acentuacao, estou usando um teclado nao ABNT.

2. Parte 1

O metodo escolhido para a interpolacao foi o metodo de Newton. Para isso, foram criadas duas funcoes:

`newton_interpolant(double* xarr, double* fxarr, int length, double div_differences[length][length], double coefs[length])`

que recebe os valores de x em `xarr` e os valores da funcao em `fxarr`, e uma matriz `div_differences` para construir a matriz de diferencas divididas. Os coeficientes na diagonal dessa matriz sao colocdos em `coefs`. Esses dois argumentos devem ser declarados no escopo global para manter a persistencia apos a execucao da funcao. Optou-se por esse tipo de retorno e passagem de argumentos para evitar erros com alocacao de memoria.

Uma vez que o vetor de coeficientes foi determinado, usa-se a funcao

`evaluate_newton_polynomial(double x, double* xarr, int length, double* coefs)`

para avaliar o polinomio em x, essa funcao retorna um `double`. A funcao evaluate foi chamada nos pontos de interpolacao para avaliar a corretude do polinomio interpolador. Podemos ver na figura abaixo que os valores sao identicos aos valores fornecidos no enunciado. Como a avaliacao usa a forma de Newton, em que os valores originais de $x_0, x_1 ...$ sao parte do polinomio, torna-se necessario passar os valores originais como argumento.

x (em metros)	$F(x)$ (em N)	$\theta(x)$ (em radianos)	$F(x)\cos(\theta(x))$
0	0.0	0.50	0.0000
5	9.0	1.40	1.5297
10	13.0	0.75	9.5120
15	14.0	0.90	8.7025
20	10.5	1.30	2.8087
25	12.0	1.48	1.0881
30	5.0	1.50	0.3537

```
(base) edu@Maxwell:~/LabNum/ep3$ gcc -o ep3 ep3.c -lm
(base) edu@Maxwell:~/LabNum/ep3$ ./ep3
O valor do polinomio para F avaliado em 0.0000 foi 0.0000.
O valor do polinomio para F avaliado em 5.0000 foi 1.5297.
O valor do polinomio para F avaliado em 10.0000 foi 9.5120.
O valor do polinomio para F avaliado em 15.0000 foi 8.7025.
O valor do polinomio para F avaliado em 20.0000 foi 2.8087.
O valor do polinomio para F avaliado em 25.0000 foi 1.0881.
O valor do polinomio para F avaliado em 30.0000 foi 0.3537.
```

Figura 1: Teste da corretude do polinomio

Em seguida, para a integracao, usa-se as funcoes
`integrate_simpson(double start, double end, double h, int length, double coefs[length], double xarr[length])`
`integrate_trapezoidal(double start, double end, double h, int length, double coefs[length], double xarr[length])`
que fazem essencialmente a mesma coisa, definem um intervalo de tamanho `h` para a integracao por partes e depois chamam as funcoes
`simpson_area(double start, double end, int length, double coefs[length], double xarr[length])`
`trapezoidal_area(double start, double end, int length, double coefs[length], double xarr[length])`
as quais respectivamente calculam o polinomio nas pontas do intervalo e no ponto medio, e nas pontas do intervalo apenas, e depois usam as formulas de simpson e do trapezio para calcular a area naquele intervalo. Para os experimentos, optou-se por `h` inicial igual a 5, e amostrou-se os pontos do polinomio em 0, 5, 10, 15, 20, 25, 30. Em seguida as areas em cada intervalo foram calculadas e somadas. Depois, o valor de h foi diminuido pela metade em cada iteracao, dobrando o numero de intervalos, mas garantindo que os pontos originais de interpolacao sejam incluidos como pontas de alguns intervalos. A area foi calculada assim sucessivas vezes ate que convergisse. A convergencia foi determinada se a diferenca entre a area anteriormente calculada e a atual, em modulo, fosse menor do que 10^{-6}. Nota-se que o mesmo processo foi aplicado para a integracao por Simpson, com a excecao de que o `h` inicial era 10. Os resultados estao apresentados na figura abaixo. Nota-se que Simpson converge com um intervalo muito maior, o que era esperado pois o metodo do trapezio tem erro de ordem $O(h^2)$ e o metodo de Simpson tem erro $O(h^4)$.

```
Integrando Simpson com 192 intervalos.
A area obtida foi 117.1316.
------------------------------
Integrando trapezio com 49152 intervalos.
A area obtida foi 117.1316.
------------------------------
```

Figura 2: Resultados da integracao

Nota-se que os numeros correspondem, em ordem de magnitude, ao esperado, pois 192 tem ordem de magnitude 10^2 e 49152 tem ordem de magnitude 10^4. De modo que o tamanho do intervalo necessario para a convergencia em Simpson foi 100 vezes maior do que no trapezio.

3. Parte 2

3.1. Variaveis Aleatorias

Para a geracao das variaveis aleatorias, usou-se a funcao `rand()` para gerar numeros entre 0 e 1 ao dividir o resultado de `rand()` por `RAND_MAX`. Para as integracoes unidimensionais, usou-se a mesma variavel aleatoria (ou seja, a semente foi definida apenas uma vez) pois as chamadas de rand() sao independentes entre si e isso torna o tempo de execucao do programa muito menor.

Para o metodo multidimensional, contudo, isso seria insuficiente pois precisamos de duas variaveis aleatorias independentes entre si. Desse modo, a semente foi alterada e n valores aleatorios gerados e guardados num vetor. Depois a semente foi novamente alterada e n valores aleatorios gerados e guardados em outro vetor. Dessa forma, garantimos n valores independentes de duas variaveis aleatorias independentes.

3.2. Mudancas de Variaveis

Para as integrais 2 e 3 do enunciado, foi necessaria uma mudanca de variaveis para que o intervalo de integracao fosse $[0, 1]$. Dessa forma, foram feitas as seguintes modificacoes.

3.2.1. Integral de x^3

Escrevendo:

$$x = au + b$$

e usando que

$$x = 3 \implies u = 0$$

temos

$$x = au + 3$$

Usando que

$$x = 7 \implies u = 1$$

temos

$$7 = a + 3$$

de onde

$$a = 4$$

Temos portanto

$$x = 4u + 3$$

e

$$dx = 4du$$

Logo

$$\int_3^7 x^3 dx = 4\int_0^1 (4u+3)^3 du$$

Verifica-se que

$$\int_3^7 x^3 dx = \frac{7^4}{4} - \frac{3^4}{4} = 600.25 - 20.25 = 580$$

e

$$4\int_0^1 (4u+3)^3 du = \int_0^1 g^3 dg = \frac{(4\cdot 1+3)^3}{4} - \frac{(4\cdot 0+3)^3}{4} = \frac{7^4}{4} - \frac{3^4}{4} = 580$$

com $g = (4u+3)$ e $dg = 4du$, validando a troca de variaveis.

3.2.2. Integral de e^{-x}

Separamos

$$\int_0^\infty e^{-x} dx = 1$$

em

$$\int_0^1 e^{-x} dx + \int_1^\infty e^{-x} dx = 1$$

Precisamos agora tratar da mudanca de variaveis em

$$\int_1^\infty e^{-x} dx$$

Tomando

$$u = 1 - e^{1-x}$$

Temos $u = 0$ se $x = 1$ e $u \to 1$ se $x \to \infty$. Isolando x em $u = 1 - e^{1-x}$ temos

$$x = 1 - ln(1-u)$$

e

$$dx = \frac{1}{1-u} du$$

Logo fazendo integracao por partes temos

$$\int_1^\infty e^{-x}dx = \int_0^1 e^{\ln(1-u)-1} \cdot \frac{1}{1-u}du = 0.36788$$

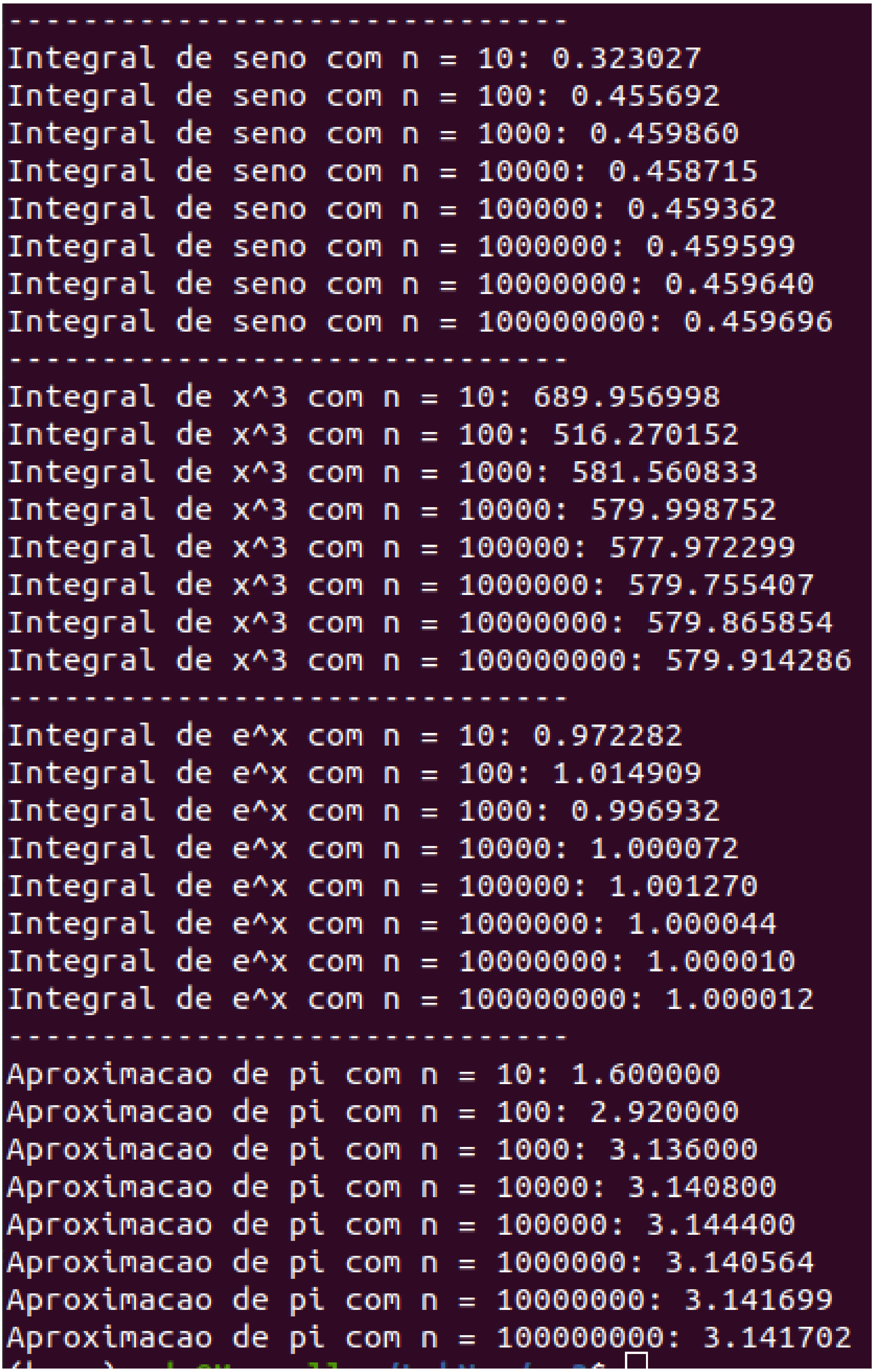
Logo

$$\int_0^1 e^{-x}dx + \int_0^1 e^{\ln(1-u)-1} \cdot \frac{1}{1-u}du = 0.63212 + 0.36788 = 1$$

Validando a mudanca de variaveis.

3.3. Resultados

O criterio de convergencia foi uma diferenca relativa entre os valores absolutos da aproximacao e da aproximacao anterior menor que 10^{-6} ou 8 iteracoes. Os valores relativamente baixos foram escolhidos por limitacoes de alocacao de memoria dos vetores de numeros aleatorios e tempo de execucao. A cada iteracao o valor n foi aumentado em 10 vezes, comecando de n = 10. As funcoes para calculo das areas foram implementadas simplesmente aplicando o resultado no enunciado para a somatoria. Os resultados encontram-se elencados na figura abaixo.



```
-----------------------------
Integral de seno com n = 10: 0.323027
Integral de seno com n = 100: 0.455692
Integral de seno com n = 1000: 0.459860
Integral de seno com n = 10000: 0.458715
Integral de seno com n = 100000: 0.459362
Integral de seno com n = 1000000: 0.459599
Integral de seno com n = 10000000: 0.459640
Integral de seno com n = 100000000: 0.459696
-----------------------------
Integral de x^3 com n = 10: 689.956998
Integral de x^3 com n = 100: 516.270152
Integral de x^3 com n = 1000: 581.560833
Integral de x^3 com n = 10000: 579.998752
Integral de x^3 com n = 100000: 577.972299
Integral de x^3 com n = 1000000: 579.755407
Integral de x^3 com n = 10000000: 579.865854
Integral de x^3 com n = 100000000: 579.914286
-----------------------------
Integral de e^x com n = 10: 0.972282
Integral de e^x com n = 100: 1.014909
Integral de e^x com n = 1000: 0.996932
Integral de e^x com n = 10000: 1.000072
Integral de e^x com n = 100000: 1.001270
Integral de e^x com n = 1000000: 1.000044
Integral de e^x com n = 10000000: 1.000010
Integral de e^x com n = 100000000: 1.000012
-----------------------------
Aproximacao de pi com n = 10: 1.600000
Aproximacao de pi com n = 100: 2.920000
Aproximacao de pi com n = 1000: 3.136000
Aproximacao de pi com n = 10000: 3.140800
Aproximacao de pi com n = 100000: 3.144400
Aproximacao de pi com n = 1000000: 3.140564
Aproximacao de pi com n = 10000000: 3.141699
Aproximacao de pi com n = 100000000: 3.141702
```

Figura 3: Resultados da integracao por Monte Carlo

Podemos notar que de modo geral, para a integral unidimensional o aumento do numero de pontos aumenta a precisao. Contudo o mesmo nao se verifica necessariamente para a integral multidimensional, sendo que dos dois ultimos resultados, o com mais pontos se mostra pior. Isso talvez se de por algum vies no gerador de numeros aleatorios ao usar duas variaveis com sementes diferentes.