UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA

RODRIGO DE CASTRO MICHELASSI
NUSP: 13672703

Exercício-Programa 3: Integração Numérica

São Paulo
2023

Sumário

1 Introdução

Esse EP tem como objetivo mostrar como funciona o processo de integração numérica estudado em aula, testando a integração analítica e comparando com os valores de suas aproximações por polinômios, de acordo com métodos específicos.

Na primeira parte do EP, foram implementadas as regras do Trapézio e Simpson, além do algoritmo de interpolação pelo método de LaGrange, para aproximarmos a integral de uma função desconhecida, apenas com dados descritos em uma tabela. O algoritmo implementado será discutido a frente.

Já na segunda parte, foi implementado a integração de 3 funções conhecidas pelo Método de Monte-Carlo, não visto em sala, por sua aplicação em integrais unidimensionais, além de aproximar o valor de π pela aplicação em integrais multidimensionais.

Os detalhes de implementação, assim como as aproximações e mudanças de variáveis implementadas para aproximar esses valores serão discutidos nos próximos tópicos.

2 Detalhes da Implementação

Antes de partirmos para falar sobre o programa em si, vamos falar um pouco sobre como ele pode ser testado.

O arquivo enviado conta com dois arquivos de extensão *.c*, são eles *compostas.c* e *montecarlo.c*, correspondendo às partes 1 e 2 do EP, respectivamente. Para testar os programas, é necessário possuir o compilador *gcc* em sua máquina. Feito isso, para compilar ambos os programas, utilizamos, no terminal:

$ *make compostas*

$ *make montecarlo*

Ao executar os programas, teremos uma pequena interface, onde poderemos realizar um teste por vez.

2.1 Compostas

Esse algoritmo consiste nos métodos de integrações compostas, dadas pelas regras do Trapézio e Simpson. Ao iniciar o programa, recebemos uma mensagem pedindo um valor para r, que aproxima a integral da função desconhecida da tabela, entre 0 e 30, a partir da interpolação por LaGrange e dos valores dados. Note que r representa o tamanho dos subintervalos os quais estaremos utilizando para aproximar nossa integral por partes, logo esse valor faz sentido quando r está em $(0, 30]$. Para valores acima desse intervalo, os resultados tendem a ser muito próximos aos obtidos com $r = 30$. Note também que, quanto menor o valor de r, maior será o valor do erro apresentado, e a diferença entre os resultados para ambos os métodos. Esses resultados serão apresentados em outra seção.

2.2 Monte Carlo

Esse algoritmo apresenta a aproximação para as integrais de Monte Carlo. Ao iniciar o programa, temos a apresentação de um pequeno menu, com 4 opções disponíveis, para aproximar as 3 integrais pedidas no enunciado ou aproximar o valor de PI. A depender da sua escolha, o programa será iniciado com os parâmetros corretos para aproximar cada valor, e será rodado uma vez.

Note que, como o algoritmo de Monte Carlo é probabilístico, obtemos a cada execução um valor diferente, porém sempre muito próximos. Os resultados serão comparados com as soluções analíticas das integrais a seguir.

3 Algoritmos Utilizados

3.1 Compostas

Para esse algoritmo, iniciamos recebendo o valor de r em cada uma das funções e aproximamos por ambos os métodos. Para isso, aplicamos os algoritmos com base nos livros, utilizando loops que rodam até r, para a regra do Trapézio, e até $\frac{r}{2}$ e $\frac{r}{2} - 1$, para a regra de Simpson. Dessa forma, para cada ponto ih aproximado, interpolamos esse ponto por LaGrange, para obtermos uma aproximação dessa função no eixo Y e obtermos a resposta esperada.

3.2 Monte Carlo

O algoritmo de Monte Carlo requer uma implementação mais elegante que o algoritmo anterior, por isso daremos mais enfoque para esse algoritmo. Para isso, temos uma resposta diferente gerada para cada função que queremos aproximar, e uma aplicação diferente para cada também.

i. $\int_0^1 sin(x)\, dx$.

Aqui, a ideia foi puramente aproximar o valor da integração pela regra unidimensional, somando a cada iteração o valor de sen($\frac{rand()}{RANDMAX}$), valores aleatórios, e retornando o resultado dessa soma por n = MAX.

ii. $\int_3^7 x^3\, dx$

Aqui, como queríamos aproximar os valores de uma integral de 3 a 7, o que não era possível pelo método de Monte Carlo tradicional. Todavia, podemos reescrever nosso método como:

$$\text{F} = \int_a^b f(x)\, dx$$
$$\iff \langle F^N \rangle = (b-a)\frac{1}{N-1}\sum_{i=0}^{N} F(X_i)$$
$$\iff X_i \in (a,b]\text{: } X_i = a + \xi_i(b-a)$$
$$\iff \langle F^N \rangle = (b-a)\frac{1}{N}\sum_{i=0}^{N} F(a + \xi_i(b-a))$$

com $\xi \in [a, b]$.

Aplicando então esse algoritmo, encontramos por força bruta $\xi = 0.564$, que aproxima corretamente o valor da integral pedida.

iii. $\int_0^\infty e^{-x}\, dx$

Novamente, temos uma integral definida em um intervalo diferente de [0,1]. Portanto, para aproximarmos essa função pelo método de Monte Carlo, aproximamos as integrais:

- $\int_0^1 e^{-x}\, dx$
- $\int_0^1 \frac{1}{e}\, dx$

Isso pois, a segunda integral é equivalente a $\int_1^\infty e^{-x}\, dx$.

Dessa forma, foi possível aproximar corretamente o valor da função somando os resultados obtidos pelas duas integrais calculadas.

iv. π

Para esse cálculo, utilizamos o método multidimensional, pois ocorre uma necessidade de termos duas dimensões, x e y, representados no código, e aplicamos o próprio algoritmo dado no enunciado.

4 Testes e Resultados

4.1 Compostas

Como aqui desconhecemos a função aplicada, os resultados serão comparados entre si, e não com uma solução analítica da integral pedida. Pelo que foi mencionado na seção anterior, vimos que os valores devem variar para valores diferentes de $r \in (0, 30]$, e quanto mais próximo de 0, maior deve ser a diferença entre os valores nos dois métodos aplicados. Note então, alguns testes obtidos e uma pequena análise sobre os resultados:

Valor de r	Trapézio Composto	Simpson Composto
1	15.917	3.537
5	122.081	130.821
10	118.893	122.227
15	118.150	116.180
20	117.837	117.673
30	117.563	117.131

Dos dados apresentados na tabela, segue que para valores pequenos de r, próximos a 0, a integração possui um valor muito distante do que é proposto. Todavia, conforme vamos aumentando esse valor, começamos a obter dados mais interessantes. Note que, para r = 5, ainda há uma discrepância, principalmente para o método de Simpson, visto que esse depende de 3 pontos (e, pelos dados da tabela, dados no enunciado, para $r = 5$, temos apenas 2 pontos conhecidos), e conforme vamos aumentando os valores, observamos que o resultado converge, em ambos os métodos, para valores próximos de 117, o que nos leva a crer que ambas as aplicações estão corretas.

4.2 Monte Carlo

No método de Monte Carlo, ocorre uma diferença quanto à seção anterior, visto que agora queremos aproximar funções reais, então não é necessário encontrar valores diferentes para os quais elas convergem, mas sim aproximar o real valor da integral, e comparar com dados analíticos. Segue, a seguir, os valores reais das integrais, feitas pela resolucao analítica:

i. $\int_0^1 sin(x)\, dx =$ -cos(1) + 1 ≈ 0.46

ii. $\int_3^7 x^3 \, dx = 580$

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

iv. $\pi = \approx 3.141592...$

Portanto, temos que obter, pelo nosso algoritmo de Monte Carlo, valores aproximados desses obtidos analiticamente. Note que esse algoritmo é probabilístico e, durante a implementação do programa, utilizamos uma semente baseada no tempo para gerar os números aleatórios, então a cada execução podemos obter respostas diferentes, porém sempre próximas. Além disso, o algoritmo de Monte Carlo implica em uma resolução próxima de infinito para que a convergência dos valores das integrais e aproximações se aproximem mais da realidade. No nosso algoritmo, implementamos um valor MAX = 10^8 para aproximar o infinito, isso pois, utilizando um valor como INT_MAX tornaria o programa muito lentom, embora mais preciso, e o valor escolhido já torna os resultados satisfatórios. Seguem a seguir, os resultados obtidos:

$sen(x)$	x^3	e^{-x}	π
0.459663	580.799270	0.999991	3.141663

Como podemos observar, os valores obtidos se aproximam de forma satisfatória dos valores analíticos, com erros $e < 1$ em todos os casos, porém menores a depender da função que estamos integrando. Vale lembrar que, ao utilizarmos valores mais próximos de infinito, o erro tende a diminuir ainda mais.

5 Considerações Finais

Pelos resultados e análises feitas acima, podemos perceber que os algoritmos implementados possuem resultados satisfatórios e que se aproximam da realidade.

Mesmo sem saber, para as regras do Trapézio e Simpson, o valor real da integração, foi possível perceber empiricamente que a aproximação se aproxima da realidade pois ambos os métodos convergem para o mesmo valor.

Já no método de Monte Carlo, foi possível concluir, com base na comparação com as soluções analíticas, que o resultado se aproximava da realidade, além de realizar testes e operações com outros valores em código maiores para representar o "infinito", o que tornou possível analisar como o erro se comportava, aumentando conforme diminuíamos esse valor.

Dessa forma, concluímos que os algoritmos testados são eficientes e podem ser utilizados para aproximar os valores dessas integrais específicas testadas, além de ter a oportunidade de conhecer outras formas de aplicar o algoritmo de Monte Carlo para integrais que não estão compreendidas no intervalo [0,1], essencial para o funcionamento do algoritmo.