

UNIVERSIDADE FEDERAL DO RIO GRANDE - FURG
PROGRAMA DE PÓS-GRADUAÇÃO EM MODELAGEM COMPUTACIONAL
CURSO DE DOUTORADO EM MODELAGEM COMPUTACIONAL

Tese de Doutorado

O Método de Monte Carlo aplicado ao Cálculo Fracionário

Luverci do Nascimento Ferreira

Orientador: Prof. Dr. Matheus Jatkoske Lazo
Co-orientador: Prof^a. Dr Antonio Gledson O. Goulart

Rio Grande, 2024

Luverci do Nascimento Ferreira

“O Método de Monte Carlo aplicado ao Cálculo Fracionário”

Tese apresentada ao Programa de Pós-Graduação em Modelagem Computacional da Universidade Federal do Rio Grande - FURG, como requisito parcial para obtenção do Grau de Doutor. Área de concentração: Modelagem Computacional.

Aprovado em: 19/12/2024

BANCA EXAMINADORA

Documento assinado digitalmente
 **MATHEUS JATKOSKE LAZO**
Data: 19/12/2024 17:04:20-0300
Verifique em <https://validar.iti.gov.br>

Prof. Dr. Matheus Jatkoske Lazo (Orientador - FURG)

Documento assinado digitalmente
 **ANTONIO GLEDSON DE OLIVEIRA GOULART**
Data: 20/12/2024 09:50:52-0300
Verifique em <https://validar.iti.gov.br>

Prof. Dr. Antonio Gledson de Oliveira Goulart (Co-orientador - FURG)

Documento assinado digitalmente
 **RUBENS DE FIGUEIREDO CAMARGO**
Data: 19/12/2024 17:12:25-0300
Verifique em <https://validar.iti.gov.br>

Prof. Dr. Rubens de Figueiredo Camargo (UNESP)



Prof. Dr. Gastão Silves Ferreira Frederico (UFC)



Prof. Dr. Julian Moises Seije Suarez (FURG)

Documento assinado digitalmente
 **BARBARA DENICOL DO AMARAL RODRIGUEZ**
Data: 19/12/2024 18:35:17-0300
Verifique em <https://validar.iti.gov.br>

Profa. Dra. Bárbara Denicol do Amaral Rodriguez (FURG)

Rio Grande - RS
2024

Ficha Catalográfica

F383m Ferreira, Luverci do Nascimento.
 O método de Monte Carlo aplicado ao cálculo fracionário / Luverci
 do Nascimento Ferreira. – 2024.
 156 f.

 Tese (doutorado) – Universidade Federal do Rio Grande –
 FURG, Programa de Pós-Graduação em Modelagem Computacional,
 Rio Grande/RS, 2024.

 Orientador: Dr. Matheus Jatkoske Lazo.

 Coorientador: Dr. Antonio Gledson de Oliveira Goulart.

 1. Cálculo fracionário 2. Monte Carlo 3. Monte Carlo variacional
 4. Cálculo variacional 5. Equações diferenciais fracionárias I. Lazo,
 Matheus Jatkoske II. Goulart, Antonio Gledson de Oliveira III. Título.

CDU 004:517.9

Catálogo na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344

Aos meus filhos Antonio e Francisco.

AGRADECIMENTOS

Em primeiro lugar, agradeço aos meus predecessores, minha querida mãe Maria Antonieta, por ter me dado o presente da vida. Assim como ela, meu pai Luverci Ferreira (in memoriam);

Meus avós Severino Luiz do Nascimento, por sua garra e sabedoria, e minha avó Hortência Correia do Nascimento (in memoriam) por sua doçura e firmeza, por terem me dado os ensinamentos iniciais da vida, sem os quais minha caminhada nunca chegaria onde chegou;

Meu tio Antônio Luiz do Nascimento, por ter sido fonte de inspiração e me descortinar e me fazer interessar pelo mundo acadêmico. À toda minha grande família, por sua diversidade e alegria;

Ao meu orientador Prof. Dr. Matheus Lazo, por toda sua paciência, grande amizade e olhar cuidadoso na construção desse trabalho e de outras parcerias. À Aline de Moura pela insistência, colaboração, acolhimento e dotes culinários;

Ao meu co-orientador Prof. Dr. Antonio Gledson de Oliveira Goulart, pelo incentivo irrestrito na escrita do trabalho;

Aos membros da banca examinadora Prof. Dr. Rubens de Figueiredo Camargo, Prof. Dr. Gastão Silves Ferreira Frederico, Prof. Dr. Julian Moisés Sejje Suarez; Profa. Dra. Bárbara Denicol do Amaral Rodriguez, por terem aceitado o convite para avaliação deste

trabalho e contribuírem para o avanço da pesquisa em matemática;

Aos professores e professoras da modelagem computacional, pelas aprendizagens ao longo dos anos de curso, em especial ao Prof. Dr. Sebastião Cícero Pinheiro Gomes e Profa. Dra. Cátia Maria dos Santos Machado;

Aos meus grandes amigos Adriano De Cezaro, Fabiana Travessini e Edite Taufer, que, diante das dificuldades enfrentadas, sempre se propuseram a me ajudar;

Aos meus amigos, Denise Martinez, Adriana Pereira, André Meneghetti, João Prolo, Ricardo Leite, que me apoiaram e incentivaram muito na construção dessa tese;

À minha família gaúcha Luiza Schneider, Olmar Schneider (in memoriam), Fernanda Schneider, José Schaedler e Márcia Schaedler, que, nos momentos em que estive ausente, acolheram e deram todo amor e carinho aos meus filhos;

Ao meu amor, minha companheira e parceira de vida, Carolina Chagas Schneider, por sua cumplicidade, carinho, compreensão, incentivo e por acreditar no meu potencial sempre.

Aos amores da minha vida, Antonio Schneider do Nascimento e Francisco Schneider do Nascimento, meus filhos, por quem toda estrada trilhada vale a pena e para quem desejo caminhos mais suaves, mas igualmente engrandecedores.

*“A matemática é a mais
alta das ciências,
o dom mais alto que os
deuses deram aos homens.”*

Arquimedes de Siracusa

RESUMO

Na presente tese, analisamos e desenvolvemos métodos numéricos para resolver equações diferenciais fracionárias, e também para obter os extremantes de funcionais, utilizando o Método de Monte Carlo. Propomos algoritmos para calcular derivadas e integrais fracionárias, para resolver problemas de valor inicial, sistemas dinâmicos, problemas de fronteira, e para obter o extremante de funcionais. As soluções obtidas foram comparadas com soluções exatas e com soluções obtidas por outros métodos numéricos, apresentando uma boa convergência dos algoritmos. Mostramos com esses resultados que os algoritmos propostos nesta tese permitem resolver todos esses tipos de problemas, fornecendo um método numérico de fácil implementação para obter as soluções de problemas não triviais envolvendo derivadas e integrais fracionárias.

Palavras-chave: Cálculo Fracionário, Monte Carlo, Monte Carlo Variacional, Cálculo Variacional, Equações Diferenciais Fracionárias.

ABSTRACT

This thesis presents new numerical methods based on the Monte Carlo Method for solving fractional differential equations and variational problems. We develop algorithms to compute fractional derivatives and integrals and to solve initial value problems, dynamic systems, boundary value problems, and optimization. The proposed algorithms show good convergence compared to exact and other numerical solutions. Our results demonstrate the versatility and efficiency of these methods for solving a wide range of non-trivial problems with fractional derivatives and integrals.

Keywords: Fractional Calculus, Monte Carlo, Variational Monte Carlo, Variational Calculus, Fractional Differential Equation.

LISTA DE FIGURAS

1	Cálculo das derivadas usando a definição de derivada em (2.2).	27
2	Modelagem Simulação de Buffon.	46
3	Modelagem Simulação de Buffon.	47
4	Cálculo do π com 10 e 50 agulhas.	48
5	Cálculo do π com 100 e 1000 agulhas.	48
6	Cálculo do π com 10 e 50 pontos.	49
7	Cálculo do π com 100 e 1000 pontos.	49
8	contents	59
9	Convergência do MMC para a derivada à esquerda de Riemann-Liouville de ordem $\alpha = 0.3$ para a função $f(x) = x$. No gráfico (a) temos $i = 10000$, e no gráfico (b) temos $i = 100000$. Nas inserções dos gráficos (a) e (b) apresentamos a convergência para a derivada à di- reita de Riemann-Liouville da função $f(x) = 10 - x$	61
10	contents	64

11	Convergência do MMC para a derivada de Caputo ordem $\alpha = 0.4$ para a função $f(x) = x$. Em (a), (b) e (c) temos $i = 1000, 10000$ e 100000 , respectivamente. Nos insertos dos gráficos (a), (b) e (c) apresentamos a convergência para derivadas de Caputo à direita da função $f(x) = 10 - x$	66
12	Convergência do MMC para derivadas de Caputo à esquerda com ordem $\alpha = 0.4$ para a função $f(x) = \sin(x)$. No inserto do gráfico, apresentamos a convergência para a derivada de Caputo à direita da função $f(x) = \sin(10 - x)$	67
13	Comparação da solução obtida pelo MMC com a solução exata para $\alpha = 1, 0.75, 0.5$ e $\beta = 0$	73
14	Convergência da solução obtida pelo MMC para $\alpha = 0.75$ e $\alpha = 0.5$ e $\beta = 0$. Na Figura (a) temos $i = 100$ e em (b) temos $i = 1000$	74
15	Comparação da solução obtida pelo MMC com a solução exata para $\alpha = 0.75$ e $\beta = -0.25, 0.5, 2$. Em (a) temos $i = 10$ e em (b) temos $i = 100$	74
16	A convergência da solução do MMC para o PVI (5.13), e com derivadas de ordens $\alpha = 1, 0.75$ e 0.5 , respectivamente. Em (a), (c), e (e), temos $i = 10$. Em (b), (d), e (f), temos $i = 100$	76
17	A convergência da solução do MMC para o PVI (5.13) com derivadas de ordens $\alpha = 0.75$, para $L = 50, 100, 250$ e com $i = 1000, 500, 200$, respectivamente.	77

18	Convergência da solução do MMC para o PVI (5.15) com derivadas de ordens $\alpha = 2, 1.75, 1.5$ e 1.25 , respectivamente. Em (a), (b), (c), e (d), temos $L = 500$ e $i = 100$	78
19	Convergência da solução do MMC para o PVI (5.15) com derivada de ordem $\alpha = 1.75$ e $L = 500$. Em (a) temos $i = 1000$, e em (b) temos $i = 10000$	78
20	Comparação da solução exata com a obtida pelo MMC para $\alpha = 21$ da equação 5.17.	80
21	Comparação da solução exata com a obtida pelo MMC para $\alpha = 20.75$ da equação 5.17.	80
22	Comparação da solução exata com a obtida pelo MMC para $\alpha = 20.5$ da equação 5.17.	81
23	Convergência da solução do MMC para o PVI (5.19) com derivada de ordem $\alpha = 0.75$. Em (a) temos $i = 10$, e em (b) temos $i = 100$. . .	81
24	Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 5$	86
25	Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 10$	86
26	Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 100$	87
27	Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 1000$	87

28	Solução do sistema (6.6) com os parâmetros, $a, b, c, d = 1, \alpha, \beta = 0.8$ e as condições iniciais dadas por $x_0 = 1$ e $y_0 = 0.75$. Em (a), (b), (c) e (d), temos $i = 10, i = 100, i = 1000$ e $i = 10000$, respectivamente.	89
29	Convergência do MMCV para o primeiro algoritmo com $\alpha = 1.5$ e 200000, 1000000 e 5000000 passos.	95
30	Convergência do MMCV para o primeiro algoritmo. Em (a) temos $\alpha = 1.25$ e 5000000 passos, e em (b) temos $\alpha = 1.75$ e 5000000 passos.	95
31	Convergência do MMCV para o segundo algoritmo. Em (a) temos $\alpha = 1.75$ e $n = 200000$ passos, e em (b) temos $\alpha = 1.95$ e $n =$ 200000 passos.	96
32	Convergência do MMCV para o primeiro algoritmo com $\alpha = 1.5$ e $n = 25000000$ passos.	97
33	Convergência do MMCV para o segundo algoritmo com $\alpha = 1.5$ e $n = 200000$ passos.	98
34	Convergência do MMCV para o primeiro algoritmo com $\alpha = 1$ e $n = 10000000$ iterações.	103
35	Função desvio, erro absoluto e erro relativo.	104
36	Convergência do MMCV para o primeiro algoritmo com $\alpha = 0.75$ e $n = 10000000$ passos.	105
37	Função desvio.	105
38	Convergência do MMCV para o primeiro algoritmo com $\alpha = 1$ e $n = 5000000$ passos.	106
39	Convergência do MMCV para o primeiro algoritmo com $\alpha = 0.75$ e $n = 5000000$ passos.	107

40	Partícula P deslizando numa curva $y(x)$ sem atrito do ponto A até o ponto B	108
41	Convergência do MMCV para a braquistócrona com $\alpha = 1$ e $n = 10000000$ passos.	109
42	Convergência do MMCV para a braquistócrona com $\alpha = 0.5$ e $n = 10000000$ passos.	110
43	Função desvio para a braquistócrona com $\alpha = 0.5$ e $n = 10000000$ passos.	111
44	Monte Carlo Variacional para o PVF 7.36 com 10000 passos com $\alpha = 1.5$	114
45	Monte Carlo Variacional para o PVF 7.3.2 com 10000 passos considerando $\alpha = 1$	116
46	Convergência do MMCV para o PVF 7.3.2 com 10000 passos considerando $\alpha = 0.3$, $\alpha = 0.50$ e $\alpha = 1$	117
47	Convergência do MMCV para a braquistócrona com 10000 passos e considerando $\alpha = 1$	118
48	Monte Carlo Variacional para a braquistócrona com 100000 passos e considerando $\alpha = 0.3$, $\alpha = 0.5$ e $\alpha = 0.8$	120

LISTA DE TABELAS

1	Limites de confiança para vários valores de acordo com [24]	52
---	---	----

LISTA DE ABREVIATURAS E SIGLAS

CF Cálculo Fracionário

MMC Método de Monte Carlo

MMCV Método de Monte Carlo Variacional

MDF Método das Diferenças Finitas

PVI Problema de Valor Inicial

PVF Problema de Valor de Fronteira

SUMÁRIO

1	Introdução	21
1.1	Estrutura da Tese	23
2	O CF de Riemann-Liouville e Caputo	25
2.1	As integrais de Riemann-Liouville	28
2.2	As derivadas de Riemann-Liouville	31
2.2.1	Algumas propriedades	33
2.2.2	Relações entre integrais e derivadas de Riemann-Liouville	36
2.3	As derivadas de Caputo	39
2.3.1	Relações entre integrais de Riemann-Liouville e derivadas de Caputo	40
2.4	Função de Mittag-Leffler	42
3	O Método de Monte Carlo	45
3.1	Cálculo do valor de π	45
3.1.1	Agulhas de Buffon	45
3.1.2	Método do círculo	48
3.2	Alguns Teoremas Clássicos da Probabilidade	49
3.3	O desenvolvimento do Método de Monte Carlo	52
3.3.1	Cálculo de integrais definidas	53

3.3.2	Algoritmo de Metropolis	55
3.3.3	Monte Carlo Variacional	56
4	O Método de Monte Carlo para integrais e derivadas fracionárias	58
4.1	O MMC para integrais de Riemann-Liouville	58
4.2	O MMC para derivadas de Riemann-Liouville	60
4.3	O MMC para derivadas de Caputo	62
5	O Método de Monte Carlo para Equações Diferenciais Ordinárias Fracionárias	68
5.1	Problema de valor inicial com Caputo	68
5.2	Problema de valor inicial com Riemann-Liouville	70
5.3	Resultados numéricos	71
5.3.1	Exemplo 1: PVI de Caputo com $f(x, y) = x^\beta$	71
5.3.2	Exemplo 2: PVI de Caputo com $f(x, y) = y$	75
5.3.3	Exemplo 3: PVI de Caputo com $f(x, y) = -y$	77
5.3.4	Exemplo 4: PVI de Caputo com ordem grande e $f(x, y) = y$	79
5.3.5	Exemplo 5: PVI de Riemann-Liouville com $f(x, y) = y$	79
6	O Método de Monte Carlo para sistemas	
	dinâmicos fracionários	82
6.1	Modelo SIR fracionário	82
6.1.1	O MMC para o modelo SIR fracionário	83
6.1.2	Resultados numéricos	85
6.2	Sistema Lotka-Volterra	88
7	Método de Monte Carlo Variacional: Problemas de fronteira para equações	
	diferenciais fracionárias e extremantes do Cálculo Variacional Fracionário	90

7.1	Problemas de fronteira para equações diferenciais fracionárias	90
7.1.1	Primeiro algoritmo	91
7.1.2	Segundo algoritmo	92
7.1.3	Exemplo 1	93
7.1.4	Exemplo 2	96
7.2	O Cálculo variacional fracionário	97
7.2.1	Um algoritmo para o extremante de funcionais	101
7.2.2	Exemplo 1	102
7.2.3	Exemplo 2	105
7.2.4	Exemplo 3	107
7.3	Um algoritmo de MMCV usando aproximação por funções potência	110
7.3.1	Exemplo 1	112
7.3.2	Exemplo 2	114
7.3.3	Exemplo 3	117
8	Considerações finais	121
	Referências	123
A	Funções Especiais	129
A.1	Função Gamma	129
A.2	Função Beta	129
A.3	Função de Gel'fand-Shilov	130
B	O Teorema de Cauchy para integrais repetidas	131

C	Algoritmos	134
C.1	Integral Fracionária	134
C.2	Derivada de Riemann-Liouville	137
C.3	Derivada de Caputo	139
C.4	PVI de Caputo com $f(x, y) = x^\beta$	141
C.5	PVI de RL com $f(x, y) = y$	143
C.6	Sistemas Dinâmicos Fracionários	144
C.7	PVF com o MMCV ponto a ponto	147
C.8	PVF com MMCV usando a função potência	149

1 INTRODUÇÃO

O Cálculo Fracionário (CF), como são chamadas as generalizações do cálculo para derivadas e integrais de ordem não inteira, teve seu início em 1695 quando Leibniz escreveu uma carta à l'Hôpital propondo uma derivada de ordem $\frac{1}{2}$ [60, 62]. Outros grandes matemáticos da história também contribuíram para o desenvolvimento do CF, entre eles podemos citar nomes como Lacroix, Liouville, Grünwald, Letnikov, Abel, Riemann e muitos outros. Apesar de ser quase tão antigo quanto o Cálculo usual (com derivadas e integrais de ordem inteira), só nas últimas cinco décadas o CF despertou maior atenção devido a suas aplicações em vários campos da ciência e da engenharia [35, 51, 67]. Esta demora no surgimento de aplicações do CF se explica devido às dificuldades de se interpretar física e geometricamente as derivadas e integrais não inteiras. Até hoje, mais de trezentos anos após sua criação, ainda não temos uma interpretação física e uma interpretação geométrica unanimemente aceita para essas derivadas e integrais. Recentemente foi proposta uma interpretação geométrica como “sombras de uma área” e uma interpretação física como “sombras do passado” [62]. Embora essas interpretações ainda pareçam não totalmente satisfatórias, ficou evidente nestas últimas cinco décadas que o CF é uma ferramenta matemática extremamente importante para aplicações que envolvam o estudo de sistemas complexos [35, 51, 67].

As derivadas fracionárias são, em geral, operadores não-locais. Consequentemente, os sistemas complexos em que utiliza-se o CF são sistemas com comportamento não-local ou com dinâmica dependente de memória. Entre as áreas de aplicações temos vários campos da ciência e da engenharia, incluindo escoamento de fluidos, meios viscosos, reologia, transporte difusivo, redes elétricas, teoria eletromagnética, teoria de campo, pro-

babilidade, etc. [35, 51, 67]. Neste sentido, a construção de métodos eficientes para a solução de equações diferenciais fracionárias, principalmente os casos não-lineares, é um grande desafio para a modelagem matemática de sistemas complexos. Além disso, ainda existem poucas teorias sobre as soluções numéricas e analíticas de equações diferenciais fracionárias, em comparação com o que temos para equações com derivadas de ordem inteira. Do ponto de vista prático, o caráter não-local desses operadores impõe dificuldades técnicas para a obtenção de resultados numéricos de modelos descrevendo sistemas realistas. Por exemplo, para obter soluções numéricas de equações diferenciais parciais, limita-se frequentemente o modelo matemático impondo-se condições de fronteira simples (por exemplo, retangular) que não descrevem fielmente o sistema real [7, 25, 76]. Outro exemplo de dificuldade técnica surge do fato de que existem diversas definições para derivadas fracionárias, e para cada definição, é necessário aplicar um método numérico adequado [23, 80].

Visando contornar essas dificuldades, e obter um método numérico que possa ser adaptado e empregado facilmente na resolução de equações diferenciais fracionárias envolvendo diversas formulações de derivadas, tanto para problemas de valor inicial como para problemas de fronteira, na presente tese propomos a construção de algoritmos baseados no Método de Monte Carlo (MMC) para a resolução de equações diferenciais ordinárias fracionárias. Além disso, mostramos que os algoritmos que propomos para a solução de problemas de fronteira podem ser utilizados diretamente para obter-se o extremo de funcionais.

O MMC é muito utilizado como uma poderosa ferramenta para a solução de problemas aplicados, em especial, na Física, Biologia, Engenharia e Finanças [71]. A origem histórica do MMC remonta ao trabalho pioneiro de Conde de Buffon que, em 1777, propôs o uso de uma amostra de números aleatórios para encontrar uma aproximação para o número π [36, 39]. No problema da agulha de Conde de Buffon, uma agulha de comprimento L é lançada de forma aleatória em um plano cortado por linhas retas paralelas separadas por uma distância $d > L$. Jogando a agulha várias vezes, Conde de Buffon mostrou que a probabilidade P de a agulha cruzar uma dessas linhas é dada por $P = \frac{2L}{\pi d}$,

em concordância com a análise matemática do problema. Hoje em dia, o MMC representa uma ampla classe de métodos/algoritmos que usam amostragem aleatória para obter resultados numéricos ou fornecer distribuições de probabilidade. No entanto, o uso do MMC na resolução de equações diferenciais ordinárias usuais (com derivadas de ordem inteira) é um tema pouco estudado na literatura [3] e ainda não explorado no contexto do CF. Os algoritmos propostos neste trabalho, para equações diferenciais fracionárias envolvendo derivadas de Riemann-Liouville e de Caputo, podem ser facilmente adaptados e aplicados em problemas envolvendo outras formulações de derivadas fracionárias, e para sistemas de equações diferenciais.

1.1 Estrutura da Tese

O trabalho está organizado em capítulos que fornecem uma análise didática e organizada para a análise e implementação do método.

No Capítulo 2 é feito um estudo e análise do Cálculo Fracionário, nas suas formulações mais utilizadas, que é a formulação de Caputo e a de Riemann-Liouville, onde os principais resultados são analisados para dar suporte teórico ao uso de tais formulações no método que será desenvolvido nos capítulos 4, 5 e 6.

No Capítulo 3, há uma explicação para o entendimento do Método de Monte Carlo (MMC) do ponto de vista histórico, bem como uma justificativa teórica para justificar a sua implementação como ferramenta matemática.

No Capítulo 4, desenvolvemos o MMC primeiramente para cálculo das integrais de Riemann-Liouville e, por consequência, para as derivadas de Riemann-Liouville e de Caputo. Simulações numéricas são feitas para exemplificar o funcionamento do MMC.

No Capítulo 5, formulamos um MMC para resolver o Problema de Valor Inicial (PVI) tanto nas formulações de Caputo quanto de Riemann-Liouville (e também de ordem inteira), utilizando a teoria do Cálculo Fracionário. Soluções numéricas são obtidas pelo MMC proposto para resolver PVIs de Caputo e também PVIs de ordem inteira. Para comparar a aplicação do MMC e analisar graficamente sua convergência, os casos estudados são com solução analítica conhecida.

No Capítulo 6, exemplificamos o processo para o uso em sistemas dinâmicos fracionários descrevendo a solução de um sistema SIR fracionário e em um sistema Lotka-Volterra fracionário.

No Capítulo 7, fornecemos uma aplicação do MMC, utilizando o algoritmo de Metropolis e o Método de Monte Carlo Variacional (MMCV) para analisar a solução de equações fracionárias com problemas de valor de fronteira (PVF) e minimizantes de funcionais dentro do escopo do Cálculo Variacional Fracionário.

Finalmente, no Capítulo 8, apresentamos as considerações finais e sugestões de continuação do trabalho.

2 O CF DE RIEMANN-LIOUVILLE E CAPUTO

No final do século XVII, a comunidade científica ainda vivia a efervescência do nascimento do Cálculo Infinitesimal [10, 15], quando o matemático francês Guillaume François Antonie, conhecido como Marquês de L'Hôpital, através de uma carta [47] fez uma pergunta aparentemente ingênua e intrigante ao matemático co-fundador do Cálculo Infinitesimal, Leibniz, perguntando qual seria o significado do operador diferencial $\frac{d^n y}{dx^n}$ quando $n = 1/2$. A pergunta de L'Hôpital, assim como a resposta de Leibniz, foi bem específica, onde n seria um número racional fixo. Generalizando a pergunta, podemos questionar: “e se n for um número real qualquer?”, porém L'Hôpital foi bem específico e a resposta de Leibniz foi:

“Você pode ver por isso, senhor, que uma quantidade como $d^{\frac{1}{2}}\overline{xy}$ ou $d^{1:2}\overline{xy}$. Embora a série infinita e a geometria sejam relações distantes, a série infinita só admite o uso de expoentes que sejam inteiros positivos e negativos, e ainda não conhece o uso de expoentes fracionários.”

E no final da carta, Leibniz finaliza a explicação

“Assim, segue-se que $d^{1:2}x$ será igual $x\sqrt{dx} : x$. Parece que um dia se tirarão consequências muito úteis desses paradoxos, pois dificilmente há paradoxos sem utilidade.”

A partir dessa carta, tem-se o nascimento do CF, como são chamadas as generalizações do cálculo para derivadas e integrais de ordem não inteira [60, 62]. E desde então grandes matemáticos da história de forma direta ou indireta contribuíram para o desenvolvimento

do CF, entre eles podemos destacar Lacroix, Liouville, Grünwald, Letnikov, Abel, Riemann e muitos outros. Apesar de ser contemporâneo ao Cálculo usual de derivadas e integrais de ordem inteira, a pesquisa envolvendo o CF ganhou destaque maior nas últimas décadas [49] especialmente desde 1974, quando ocorreu a primeira conferência internacional sobre o tema. Bertram Ross organizou a conferência que ocorreu na Universidade de New Haven, Connecticut, em 1974 [66]. Talvez, um dos fatores que contribuíram para atrasar o surgimento de aplicações do CF, foi a dificuldade de interpretar física e geometricamente as derivadas e integrais não inteiras.

Historicamente, o nome “derivada fracionária”, para designar derivadas de ordem não inteira, foi utilizado pela primeira vez pelo matemático francês Sylvestre François Lacroix (1765-1843) que, em apenas duas páginas no seu trabalho [13] envolvendo o cálculo diferencial e integral, mesmo que de forma indireta, contribuiu para o desenvolvimento do CF [21]. Inspirado em resultados de Euler, Lacroix generalizou a fórmula para a n -ésima derivada de monômios ($y = x^m$) dada por

$$\frac{d^n x^m}{dx^n} = D^n x^m = \frac{m!}{(m-n)!} x^{m-n}, \quad (2.1)$$

para m e n inteiros com $m > n$, substituindo os m e n por números reais β e α . Temos então

$$\frac{d^\alpha x^\beta}{dx^\alpha} = D^\alpha x^\beta = \frac{\Gamma(\beta+1)}{\Gamma(\beta-\alpha+1)} x^{\beta-\alpha}, \quad (2.2)$$

onde Γ é a função Gamma (veja anexo A.1). Apresentamos abaixo um exemplo e alguns gráficos de derivadas fracionárias pela definição Lacroix.

Exemplo 2.0.1. *Como exemplo, se $\alpha = 1/2$ e $\beta = 1$ na equação (2.2) temos*

$$\left(\frac{d}{dx}\right)^{1/2} x = D^{1/2} x = \frac{\Gamma(2)}{\Gamma(3/2)} x^{1/2} = \frac{2\sqrt{x}}{\sqrt{\pi}}. \quad (2.3)$$

Na Figura 1 calculamos as derivadas com $\beta = 0$ e $\beta = 1$ para diferentes valores de α , observamos que a derivada de uma função constante, quando $\alpha \neq 1$ é diferente de zero, contrariando a noção da definição usual de derivada do Cálculo Diferencial e Integral.

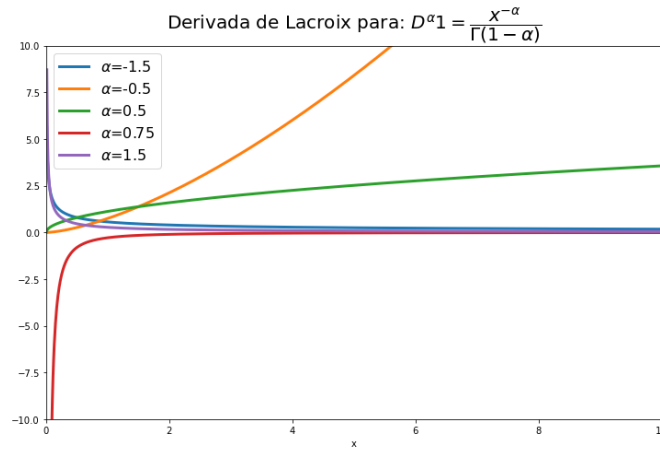
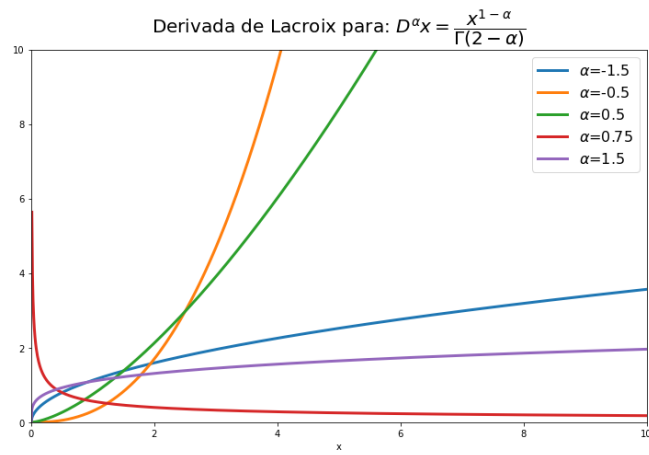
(a) com $\beta = 0$.(b) com $\beta = 1$.

Figura 1: Cálculo das derivadas usando a definição de derivada em (2.2).

Embora seja a definição mais simples para derivada fracionária, a definição no sentido Lacroix é muito restritiva, pois só se aplica a funções analíticas, além de não ser uma definição formalmente rigorosa.

A próxima definição da derivada fracionária que apresentamos nesta breve revisão histórica é a de Grunwald-Letnikov [32]. Por envolver uma série infinita com coeficientes binomiais, permite uma fácil discretização e criação de algoritmos para sua implementação numérica, além de ser baseada na noção usual de derivada como um limite da variação da função. A derivada de Grunwald-Letnikov surge quando utilizamos a definição de derivada usual com ordens inteiras e generalizamos para uma ordem arbitrária [58]. Assim, considerando α um número real qualquer, a derivada fracionária de ordem α , segundo

Grunwald-Letnikov, é definida pelo limite:

$$\frac{d^\alpha f(x)}{dx^\alpha} = \lim_{h \rightarrow 0} \frac{1}{h^\alpha} \sum_{m=0}^{\infty} (-1)^m \binom{\alpha}{m} f(x + (\alpha - m)h), \quad (2.4)$$

que é obtido através da continuação analítica da definição usual de derivada de ordem inteira n . Apesar de ser mais geral que a definição de Lacroix, a formulação de Grunwald-Letnikov não é muito utilizada (fora da utilização em métodos numéricos) por ser difícil utilizar algebricamente a equação (2.4) mesmo para funções simples.

Atualmente existem diversas formulações de CF. Na tese [74] é analisado e apresentado dois critérios para determinar quando uma definição de derivada fracionária deva ser validada, usa-se um critério proposto por Bertran Ross em 1975 [66] e outro critério por Ortigueira e Tenreiro Machado no artigo “*Whats is a fractional derivative*” [61]. Em [75] é proposta uma classificação dos operadores de ordem não inteira divididos em quatro classes distintas, denominadas: derivadas fracionárias clássicas; derivadas modificadas; operadores “fracionários” locais e operadores “fracionários” com núcleo não singular.

No presente trabalho, iremos considerar duas das formulações clássicas mais populares, as derivadas fracionárias de Riemann-Liouville e de Caputo [22, 41, 58, 60, 69]. Apesar de considerar apenas essas duas formulações do CF, os algoritmos que propomos nesta tese podem ser facilmente adaptados para outras formulações. Nas próximas seções deste capítulo, apresentaremos algumas das definições e propriedades básicas dos CFs de Riemann-Liouville e Caputo.

2.1 As integrais de Riemann-Liouville

Apesar de existirem diversas propostas para derivadas fracionárias, um grande número dessas formulações está de alguma forma conectado com a continuação analítica da fórmula de Cauchy para integrais repetidas.

Teorema 2.1.1 (Fórmula de Cauchy). *Seja $f : [a, b] \rightarrow \mathbb{R}$ uma função integrável segundo*

Riemann no intervalo $[a, b]$. Então, temos para a integral de ordem n ($n \in \mathbb{N}$):

$$\begin{aligned} \int_a^x f(\tilde{x})(d\tilde{x})^n &= \int_a^x \int_a^{x_n} \int_a^{x_{n-1}} \cdots \int_a^{x_3} \int_a^{x_2} f(x_1) dx_1 dx_2 \cdots dx_{n-1} dx_n \\ &= \frac{1}{\Gamma(n)} \int_a^x \frac{f(u)}{(x-u)^{1-n}} du \quad (n \in \mathbb{N}), \end{aligned} \quad (2.5)$$

onde Γ é a função gamma de Euler.

A prova da fórmula de Cauchy é apresentada no Anexo B e pode ser encontrada em diversos livros textos (por exemplo, em [60]). A continuação analítica da (2.5) nos dá uma definição para a integral de ordem não inteira (historicamente chamada de ordem fracionária). Essa integral de ordem fracionária é a pedra fundamental do CF de Riemann-Liouville, assim como de diversas outras formulações [22, 41, 58, 60, 69]. As integrais fracionárias obtidas a partir da (2.5) são chamadas de integrais à direita e à esquerda de Riemann-Liouville:

Definição 2.1.1. *Seja $\alpha \in \mathbb{R}_+$. Os operadores ${}_a J_x^\alpha$ e ${}_x J_b^\alpha$ definidos em $L_1[a, b]$ por*

$${}_a J_x^\alpha f(x) = \frac{1}{\Gamma(\alpha)} \int_a^x \frac{f(u)}{(x-u)^{1-\alpha}} du \quad (\alpha \in \mathbb{R}_+) \quad (2.6)$$

e

$${}_x J_b^\alpha f(x) = \frac{1}{\Gamma(\alpha)} \int_x^b \frac{f(u)}{(u-x)^{1-\alpha}} du \quad (\alpha \in \mathbb{R}_+), \quad (2.7)$$

com $a < b$ e $a, b \in \mathbb{R}$, são chamados de integrais fracionárias à esquerda e à direita de Riemann-Liouville de ordem $\alpha \in \mathbb{R}_+$, respectivamente.

Observação 2.1.1. *Quando o expoente da integral de Riemann-Liouville é zero, ou seja, $\alpha = 0$, o operador fracionário se comporta como um operador identidade. Assim, o operador de ordem zero é um caso limite que se reduz à função original.*

Para α inteiro, as integrais de Riemann-Liouville (2.6) e (2.7) coincidem com integrais repetidas usuais (2.5). Além disso, a partir das equações (2.6) e (2.7) é fácil ver que as integrais fracionárias de Riemann-Liouville convergem para quaisquer funções integráveis f se $\alpha > 1$. Mais ainda, é possível provar a convergência de (2.6) e (2.7) para funções $f \in L_1[a, b]$ mesmo quando $0 < \alpha < 1$ [22].

As integrais fracionárias de Riemann-Liouville (2.6) e (2.7) preservam uma importante propriedade algébrica das integrais usuais:

Teorema 2.1.2. *Seja $\alpha, \beta \geq 0$ e $f \in L_1[a, b]$. Então*

$${}_a J_x^\alpha {}_a J_x^\beta f(x) = {}_a J_x^{\alpha+\beta} f(x) \quad e \quad {}_x J_b^\alpha {}_x J_b^\beta f(x) = {}_x J_b^{\alpha+\beta} f(x). \quad (2.8)$$

Prova (veja [22]): Vamos apresentar a prova para as derivadas à direita (para as derivadas à esquerda segue uma prova similar). Da definição 2.1.1 temos

$${}_a J_x^\alpha {}_a J_x^\beta f(x) = \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^x (x-t)^{\alpha-1} \int_a^t (t-u)^{\beta-1} f(u) du dt. \quad (2.9)$$

Como as integrais existem para $f \in L_1[a, b]$, usando o Teorema de Fubini podemos trocar a ordem de integração

$$\begin{aligned} {}_a J_x^\alpha {}_a J_x^\beta f(x) &= \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^x \int_u^x (x-t)^{\alpha-1} (t-u)^{\beta-1} f(u) dt du \\ &= \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^x f(u) \int_u^x (x-t)^{\alpha-1} (t-u)^{\beta-1} dt du. \end{aligned} \quad (2.10)$$

Fazendo a mudança de variável $t = u + s(x-u)$ obtemos

$${}_a J_x^\alpha {}_a J_x^\beta f(x) = \frac{1}{\Gamma(\alpha)\Gamma(\beta)} \int_a^x f(u) (x-u)^{\alpha+\beta-1} \int_0^1 (1-s)^{\alpha-1} s^{\beta-1} ds du. \quad (2.11)$$

Identificando $\int_0^1 (1-s)^{\alpha-1} s^{\beta-1} ds = B(\alpha, \beta) = \Gamma(\alpha)\Gamma(\beta)/\Gamma(\alpha+\beta)$ com a função Beta de Euler $B(\alpha, \beta)$ resulta, finalmente,

$${}_a J_x^\alpha {}_a J_x^\beta f(x) = \frac{1}{\Gamma(\alpha+\beta)} \int_a^x f(u) (x-u)^{\alpha+\beta-1} du = {}_a J_x^{\alpha+\beta} f(x). \quad (2.12)$$

□

Uma importante consequência do Teorema (2.1.2) é a comutatividade dos operadores integrais:

Corolário 2.1.1. *Sobre as considerações do Teorema (2.1.2) temos*

$${}_a J_x^\alpha {}_a J_x^\beta f(x) = {}_a J_x^\beta {}_a J_x^\alpha f(x) \quad e \quad {}_x J_b^\alpha {}_x J_b^\beta f(x) = {}_x J_b^\beta {}_x J_b^\alpha f(x). \quad (2.13)$$

Em outras palavras, os conjuntos dos operadores $\{{}_a J_x^\alpha : L_1[a, b] \rightarrow L_1[a, b]; \alpha \geq 0\}$ e $\{{}_x J_b^\alpha : L_1[a, b] \rightarrow L_1[a, b]; \alpha \geq 0\}$ formam semigrupos com respeito à concatenação, onde o elemento neutro é o operador identidade, ${}_a J_x^0$ e ${}_x J_b^0$, respectivamente. Estas propriedades (2.13) são uma generalização direta do caso usual de integrais de ordem inteira, onde as integrais também comutam formando um semigrupo.

Observação 2.1.2. *Com a função de Gel'fand-Shilov (veja a equação (A.3) no Apêndice A) e a convolução de Laplace tem-se uma outra forma simples e elegante de definir a integral fracionária dada por:*

$${}_0 J_x^\alpha = \Phi_{-\alpha}(t) * f(t) = \frac{1}{\Gamma(-\alpha)} \int_0^t \frac{f(\tau)}{(t-\tau)^{1+\alpha}} d\tau \quad (\alpha \in \mathbb{R}_+) \quad (2.14)$$

Usando a propriedade da função Gel'fand-Shilov (veja a (A.5) no Apêndice A) obtêm-se a prova do teorema 2.1.2.

2.2 As derivadas de Riemann-Liouville

Os operadores de integração ${}_a J_x^\alpha$ e ${}_x J_b^\alpha$, definidos em (2.6) e (2.7), desempenham papel central no CF de Riemann-Liouville. Antes de definir as derivadas fracionárias de Riemann-Liouville, relembremos que para inteiros positivos $n > m$ vale a identidade $D_x^m f(x) = D_x^n {}_a J_x^{n-m} f(x)$, onde D_x^m é uma derivada usual de ordem inteira m .

Definição 2.2.1 (Derivadas de Riemann-Liouville). *As derivadas fracionárias de ordem $\alpha > 0$ ($\alpha \in \mathbb{R}$) de Riemann-Liouville à esquerda e à direita são definidas, respectivamente, por ${}_a D_x^\alpha f(x) = D_x^n {}_a J_x^{n-\alpha} f(x)$ e ${}_x D_b^\alpha f(x) = (-1)^n D_x^n {}_x J_b^{n-\alpha} f(x)$ com $n = [\alpha] + 1$, ou seja,*

$${}_a D_x^\alpha f(x) = \frac{1}{\Gamma(n-\alpha)} \frac{d^n}{dx^n} \int_a^x \frac{f(u)}{(x-u)^{1+\alpha-n}} du \quad (n = [\alpha] + 1, \alpha \in \mathbb{R}_+, a \in \mathbb{R}) \quad (2.15)$$

e

$${}_x D_b^\alpha f(x) = \frac{(-1)^n}{\Gamma(n - \alpha)} \frac{d^n}{dx^n} \int_x^b \frac{f(u)}{(u - x)^{1+\alpha-n}} du \quad (n = [\alpha] + 1, \alpha \in \mathbb{R}_+^*, b \in \mathbb{R}), \quad (2.16)$$

onde $D_x^n = \frac{d^n}{dx^n}$ é uma derivada usual de ordem inteira n .

Uma consequência importante das equações (2.15) e (2.16) é que as derivadas de Riemann-Liouville são operadores não locais. O operador à esquerda (à direita) definido pela expressão integro-diferencial (2.15) (e (2.16)) depende de valores da função à esquerda (à direita) de x , isto é, $a \leq u \leq x$ ($x \leq u \leq b$).

Observação 2.2.1. Quando α é um inteiro, as derivadas fracionárias de Riemann-Liouville (2.15) e (2.16) reduzem às derivadas usuais de ordem inteira α .

Observação 2.2.2. Para $0 < \alpha < 1$ as derivadas fracionárias de Riemann-Liouville (2.15) e (2.16) podem ser aplicadas mesmo em funções não diferenciáveis. Esta propriedade é uma consequência direta do fato das derivadas de Riemann-Liouville serem dadas a partir da derivada de primeira ordem da integral fracionária da função. Podemos tomar a derivada de Riemann-Liouville mesmo de funções não diferenciáveis em todo o domínio, como a função de Weierstrass [43].

Para ilustrar a aplicação das derivadas de Riemann-Liouville em funções simples, vamos calcular a derivada à esquerda ${}_a D_x^\alpha f(x)$ (2.15) da função potência $f(x) = (x - a)^\beta$.

Temos

$${}_a D_x^\alpha (x - a)^\beta = \frac{1}{\Gamma(n - \alpha)} \frac{d^n}{dx^n} \int_a^x \frac{(u - a)^\beta}{(x - u)^{1+\alpha-n}} du, \quad (2.17)$$

onde $n = [\alpha] + 1$, e devemos ter $\beta > -1$ em ordem de garantir a convergência da integral em (2.17). Fazendo a mudança de variável $s = (u - a)/(x - a)$ obtemos

$$\begin{aligned} {}_a D_x^\alpha (x - a)^\beta &= \frac{1}{\Gamma(n - \alpha)} \frac{d^n}{dx^n} (x - a)^{\beta - \alpha + n} \int_0^1 s^\beta (1 - s)^{n - \alpha - 1} ds \\ &= \frac{\Gamma(\beta + 1)}{\Gamma(n - \alpha + \beta + 1)} \frac{d^n}{dx^n} (x - a)^{\beta - \alpha + n}, \end{aligned} \quad (2.18)$$

onde identificamos $\int_0^1 s^\beta (1 - s)^{n - \alpha - 1} ds = B(n - \alpha, \beta + 1)$ com a função Beta de Euler (veja anexo A.2). Temos duas possibilidades para a derivada (2.18). A primeira, quando

$\alpha - \beta \in \mathbb{N}$, o lado direito da (2.18) se reduz a um polinômio de grau inteiro $n - (\alpha - \beta) \in \{0, 1, 2, \dots, n - 1\}$ e, portanto, a derivada é nula. Na segunda, quando $\alpha - \beta \notin \mathbb{N}$ a derivada é não nula:

$${}_a D_x^\alpha (x - a)^\beta = \begin{cases} 0 & \text{se } \alpha - \beta \in \mathbb{N} \\ \frac{\Gamma(\beta + 1)}{\Gamma(\beta - \alpha + 1)} (x - a)^{\beta - \alpha} & \text{se } \alpha - \beta \notin \mathbb{N} \end{cases} \quad (\alpha \in \mathbb{R}_+, \beta > -1), \quad (2.19)$$

onde usamos a propriedade $\Gamma(z + 1) = z\Gamma(z)$ da função Gamma de Euler. De forma equivalente, temos para a derivada à direita (2.16)

$${}_x D_b^\alpha (b - x)^\beta = \begin{cases} 0 & \text{se } \alpha - \beta \in \mathbb{N} \\ \frac{\Gamma(\beta + 1)}{\Gamma(\beta - \alpha + 1)} (b - x)^{\beta - \alpha} & \text{se } \alpha - \beta \notin \mathbb{N} \end{cases} \quad (\alpha \in \mathbb{R}_+, \beta > -1). \quad (2.20)$$

Observação 2.2.3. É importante notar que a derivada de Riemann-Liouville (2.15) e (2.16) de uma constante não é zero para $\alpha \notin \mathbb{N}$.

$${}_a D_x^\alpha 1 = \frac{(x - a)^{-\alpha}}{\Gamma(1 - \alpha)}, \quad {}_x D_b^\alpha 1 = \frac{(b - x)^{-\alpha}}{\Gamma(1 - \alpha)}. \quad (2.21)$$

Para $\alpha \in \mathbb{N}$ recuperamos as derivadas usuais de ordem inteira, e o lado direito da (2.21) se torna igual a zero devido aos polos da função Gamma de Euler. Devido ao fato de a derivada de uma constante ser diferente de zero, as equações diferenciais envolvendo derivadas de Riemann-Liouville requerem condições iniciais e de contorno não usuais [22, 41, 58, 60, 69].

2.2.1 Algumas propriedades

Além da derivada da constante ser não nula, outra propriedade importante das derivadas inteiras que não é satisfeita pelas definições de Riemann-Liouville é a comutatividade. No Cálculo usual, a comutatividade dos operadores é uma consequência direta da bem co-

hecida Lei dos Expoentes para as derivadas e integrais de ordem inteira $m, n \in \mathbb{N}_+$,

$$\begin{aligned} {}_a J_x^m {}_a J_x^n f(x) &= {}_a J_x^n {}_a J_x^m f(x) = {}_a J_x^{m+n} f(x), \\ D_x^m D_x^n f(x) &= D_x^n D_x^m f(x) = D_x^{m+n} f(x), \end{aligned} \quad (2.22)$$

sendo D_x^n e ${}_a J_x^m$ as derivadas e integrais usuais de ordem inteira. Enquanto a primeira relação em (2.22) é estendida para as integrais fracionárias de Riemann-Liouville (2.6) e (2.7) (veja a (2.13)), para as derivadas de Riemann-Liouville (2.15) e (2.16) podemos ter situações como:

$$\begin{aligned} {}_a D_x^\alpha {}_a D_x^\beta f(x) &= {}_a D_x^\beta {}_a D_x^\alpha f(x) \neq {}_a D_x^{\alpha+\beta} f(x), \\ {}_a D_x^\alpha {}_a D_x^\beta f(x) &\neq {}_a D_x^\beta {}_a D_x^\alpha f(x) = {}_a D_x^{\alpha+\beta} f(x). \end{aligned} \quad (2.23)$$

Em outras palavras, a Lei dos Expoentes não é satisfeita pelas derivadas fracionárias de Riemann-Liouville (2.15) e (2.16), que conseqüentemente não formam um semigrupo. Vamos ilustrar as situações expostas em (2.23) com dois exemplos.

Exemplo 2.2.1. No caso da primeira relação em (2.23), vamos tomar $f(x) = (x - a)^{-1/2}$ com $\alpha = \beta = 1/2$. Então, usando (2.19) obtemos ${}_a D_x^{1/2} f(x) = 0$ e, portanto, ${}_a D_x^{1/2} {}_a D_x^{1/2} f(x) = 0$, mas ${}_a D_x^{1/2+1/2} f(x) = D_x^1 f(x) = -(x - a)^{-3/2}/2$.

Exemplo 2.2.2. No caso da segunda relação em (2.23), vamos tomar $f(x) = (x - a)^{1/2}$ com $\alpha = 1/2$ e $\beta = 3/2$. Então, usando (2.19) obtemos ${}_a D_x^{1/2} f(x) = \sqrt{\pi}/2$ e ${}_a D_x^{3/2} f(x) = 0$. Portanto, temos ${}_a D_x^{1/2} {}_a D_x^{3/2} f(x) = 0$, mas ${}_a D_x^{3/2} {}_a D_x^{1/2} f(x) = {}_a D_x^{1/2+3/2} f(x) = D_x^2 f(x) = -(x - a)^{-3/2}/4$.

Em (2.23) consideramos apenas derivadas à esquerda, mas os mesmos resultados podem ser automaticamente estendidos para derivadas à direita. No que segue o restante do capítulo, utilizaremos apenas derivadas e integrais à esquerda, com exceção da fórmula de integração por partes no final do capítulo.

Vamos investigar agora a generalização de algumas regras clássicas do cálculo usual envolvendo mais de uma função na mesma derivada. Essas generalizações são importantes para simplificar o cálculo de derivadas fracionárias de funções mais complexas.

Teorema 2.2.1. *Sejam f e g duas funções definidas em $[a, b]$ tal que ${}_aD_x^\alpha f(x)$ e ${}_aD_x^\alpha g(x)$ existam. Sejam ainda $c_1, c_2 \in \mathbb{R}$ duas constantes quaisquer. Então, ${}_aD_x^\alpha (c_1 f(x) + c_2 g(x))$ existe, e*

$${}_aD_x^\alpha (c_1 f(x) + c_2 g(x)) = c_1 {}_aD_x^\alpha f(x) + c_2 {}_aD_x^\alpha g(x). \quad (2.24)$$

Prova: A prova segue diretamente da definição (2.15) pois derivadas e integrais usuais são operadores lineares $\square$.

Outra regra importante do Cálculo é a chamada Regra do Produto, ou Regra de Leibniz. A regra para derivada primeira do produto de duas funções $f(x)$ e $g(x)$, $D_x^1 (f(x)g(x)) = (D_x^1 f(x)) g(x) + f(x) (D_x^1 g(x))$, pode ser facilmente estendida para derivada de ordem inteira $m \in \mathbb{N}_+$ qualquer:

Teorema 2.2.2 (Regra de Leibniz). *Seja $m \in \mathbb{N}_+$, e seja $f, g \in C^m[a, b]$ duas funções. Então*

$$D_x^m (f(x)g(x)) = \sum_{k=0}^m \binom{m}{k} (D_x^k f(x)) (D_x^{m-k} g(x)). \quad (2.25)$$

A Regra de Leibniz é aplicável a quaisquer funções de classe $C^m[a, b]$, além de ser simétrica nas funções $f(x)$ e $g(x)$, ou seja, podemos trocar a ordem das funções em (2.25) que não altera o resultado. Essas propriedades da Regra de Leibniz não são preservadas quando utilizamos derivadas de Riemann-Liouville.

Teorema 2.2.3 (Regra de Leibniz para derivadas de Riemann-Liouville). *Sejam f e g funções analíticas em $[a, b]$. E seja $n = [\alpha] + 1$ com $\alpha \in \mathbb{R}_+$. Então*

$${}_aD_x^\alpha (f(x)g(x)) = \sum_{k=0}^{n-1} \binom{\alpha}{k} (D_x^k f(x)) ({}_aD_x^{\alpha-k} g(x)) + \sum_{k=n}^{\infty} \binom{\alpha}{k} (D_x^k f(x)) ({}_aJ_x^{k-\alpha} g(x)). \quad (2.26)$$

A prova do Teorema 2.2.3 é mais elaborada e pode ser encontrada, por exemplo, em [22, 58, 60]. É importante notar que, diferente da Regra de Leibniz para derivadas inteiras (2.25), para derivadas fracionárias (2.26) a derivada do produto só é aplicável às funções analíticas e não é mais simétrica nas funções $f(x)$ e $g(x)$.

Finalmente, terminamos esta subseção com a Regra da Cadeia. Para derivadas de Riemann-Liouville essa regra é importante para ilustrar o quão complicado pode ser calcular derivadas fracionárias de funções simples. A Regra da Cadeia para derivadas inteiras de ordem m já é de pouca utilidade quando m é grande. Essa regra é dada pela fórmula de Faà di Bruno.

Teorema 2.2.4 (Fórmula de Faà di Bruno). *Sejam $f, g \in C^m[a, b]$ e $m \in \mathbb{N}$. Então*

$$D_x^m g(f(x)) = m! \sum_{k=1}^m D_u^k g(u)|_{u=f(x)} \sum \prod_{l=1}^m \frac{1}{P_l!} \left(\frac{D_x^l f(x)}{l!} \right)^{P_l}, \quad (2.27)$$

onde a segunda soma $\sum$ estende-se sobre todas as combinações de inteiros positivos $P_1, P_2, \dots, P_m$ tais que

$$\sum_{l=1}^m l P_l = m \text{ e } \sum_{l=1}^m P_l = k. \quad (2.28)$$

Uma demonstração para a Fórmula de Faà di Bruno pode ser encontrada em [38].

Para derivadas de Riemann-Liouville temos [60, 58, 22]:

Teorema 2.2.5 (Fórmula de Faà di Bruno para derivadas de Riemann-Liouville). *Sejam $f(x)$ e $g(x)$ funções analíticas em $[a, b]$, e seja $\alpha \in \mathbb{R}_+$. Então*

$$\begin{aligned} {}_a D_x^\alpha g(f(x)) &= \sum_{j=1}^{\infty} \binom{\alpha}{j} \frac{j! (x-a)^{j-\alpha}}{\Gamma(j-\alpha+1)} \sum_{k=1}^j D_u^k g(u)|_{u=f(x)} \sum \prod_{l=1}^j \frac{1}{P_l!} \left(\frac{D_x^l f(x)}{l!} \right)^{P_l} \\ &\quad + \frac{(x-a)^{-\alpha}}{\Gamma(1-\alpha)} g(f(x)), \end{aligned} \quad (2.29)$$

onde a terceira soma $\sum$ estende-se sobre todas as combinações de inteiros positivos $P_1, P_2, \dots, P_j$ tais que

$$\sum_{l=1}^j l P_l = m \text{ e } \sum_{l=1}^j P_l = k. \quad (2.30)$$

2.2.2 Relações entre integrais e derivadas de Riemann-Liouville

Nesta seção vamos estudar as relações mais importantes entre as derivadas e integrais de Riemann-Liouville. Para um estudo das propriedades algébricas dos operadores derivada e integral de Riemann-Liouville recomendamos ao leitor o excelente livro de Miller

e Ross [58].

Definimos em (2.15) e (2.16) as derivadas de Riemann-Liouville como sendo a derivada de ordem inteira $n = [\alpha] + 1$ de uma integral fracionária de ordem $n - \alpha$. No entanto, essa definição não é única, como pode ser vista pelo Lema abaixo:

Lema 2.2.1. *Seja $\alpha \in \mathbb{R}_+$, e $m \in \mathbb{N}_+$ tal que $m > \alpha$. Então*

$${}_a D_x^\alpha f(x) = D_x^m {}_a J_x^{m-\alpha} f(x). \quad (2.31)$$

Prova: A condição $m > \alpha$ implica que $m \geq n = [\alpha] + 1$. Da propriedade de semigrupo das derivadas inteiras (2.22), e das integrais fracionárias (2.8) e (2.13), obtemos

$$\begin{aligned} D_x^m {}_a J_x^{m-\alpha} f(x) &= D_x^n D_x^{m-n} {}_a J_x^{m-n} {}_a J_x^{n-\alpha} f(x) \\ &= D_x^n {}_a J_x^{n-\alpha} f(x) = {}_a D_x^\alpha f(x), \end{aligned} \quad (2.32)$$

onde $D_x^{m-n} {}_a J_x^{m-n} = 1$ já que D_x^{m-n} e ${}_a J_x^{m-n}$ são derivadas e integrais usuais de ordem inteira, e usamos a definição (2.15) $\square$.

Além de mostrar que as definições (2.15) e (2.16) não são únicas, o Lema acima permite provar que a derivada fracionária de Riemann-Liouville (2.15) é a operação inversa à integral de Riemann-Liouville (2.6):

Teorema 2.2.6. *Seja $\alpha > 0$. Então, para toda $f \in L_1[a, b]$ temos*

$${}_a D_x^\alpha {}_a J_x^\alpha f(x) = f(x). \quad (2.33)$$

Prova: O caso $\alpha = 0$ é trivial. Vamos então considerar $\alpha > 0$. Usando o Lema (2.2.1) e a propriedade (2.8) obtemos

$${}_a D_x^\alpha {}_a J_x^\alpha f(x) = D_x^m {}_a J_x^{m-\alpha} {}_a J_x^\alpha f(x) = D_x^m {}_a J_x^m f(x) = f(x) \quad \square. \quad (2.34)$$

Enquanto a derivada de Riemann-Liouville é a operação inversa da integral, podemos provar que a integral não é o inverso da derivada. No cálculo usual, temos o Teorema

Fundamental do Cálculo.

Teorema 2.2.7 (Teorema Fundamental do Cálculo). *Seja $f \in C^1[a, b]$, então,*

$${}_a J_x^1 D_x^1 f(x) = \int_a^x \frac{df(u)}{du} du = f(x) - f(a). \quad (2.35)$$

Podemos generalizar este resultado para o CF. O teorema fundamental análogo no CF de Riemann-Liouville, é enunciado por:

Teorema 2.2.8 (Teorema Fundamental do Cálculo de Riemann-Liouville). *Considere $0 < \alpha$, e seja ${}_a J_x^{n-\alpha} f(x) \in AC^n[a, b]$ e ${}_b J_x^{n-\alpha} f(x) \in AC^n[a, b]$ com $n = [\alpha] + 1$. Então, valem as duas igualdades a seguir:*

$${}_a J_x^\alpha D_x^\alpha f(x) = f(x) - \sum_{k=0}^{n-1} \frac{(x-a)^{\alpha-1-k}}{\Gamma(\alpha-k)} \lim_{u \rightarrow a^+} \frac{d^{n-k-1}}{du^{n-k-1}} {}_a J_u^{n-\alpha} f(u) \quad (2.36)$$

e

$${}_b J_x^\alpha D_b^\alpha f(x) = f(x) - \sum_{k=0}^{n-1} \frac{(-1)^{n-k-1} (b-x)^{\alpha-1-k}}{\Gamma(\alpha-k)} \lim_{u \rightarrow b^-} \frac{d^{n-k-1}}{du^{n-k-1}} {}_b J_u^{n-\alpha} f(u). \quad (2.37)$$

A prova do Teorema 2.2.8 pode ser encontrada em diversos livros textos, por exemplo, em [22, 41, 69].

Observação 2.2.4. *Em particular, se $0 < \alpha < 1$, (2.36), (2.37) se reduzem a:*

$${}_a J_x^\alpha D_x^\alpha f(x) = f(x) - \frac{(x-a)^{\alpha-1}}{\Gamma(\alpha)} \lim_{u \rightarrow a^+} {}_a J_u^{1-\alpha} f(u), \quad (2.38)$$

e

$${}_b J_x^\alpha D_b^\alpha f(x) = f(x) - \frac{(b-x)^{\alpha-1}}{\Gamma(\alpha)} \lim_{u \rightarrow b^-} {}_b J_u^{1-\alpha} f(u), \quad (2.39)$$

respectivamente.

Como veremos neste trabalho, o Teorema Fundamental do Cálculo de Riemann-Liouville 2.2.8 será de grande importância para a formulação de um MMC para EDOs fracionárias.

2.3 As derivadas de Caputo

As derivadas fracionárias de Caputo são definidas de forma semelhante às de Riemann-Liouville, mas trocando-se a ordem entre integração e derivação:

Definição 2.3.1 (Derivada fracionária de Caputo). *As derivadas de Caputo à esquerda e à direita de ordem $\alpha > 0$ ($\alpha \in \mathbb{R}$) são definidas, respectivamente, por ${}_a^C D_x^\alpha f(x) = {}_a J_x^{n-\alpha} D_x^n f(x)$ e ${}_x^C D_b^\alpha f(x) = (-1)^n {}_x J_b^{n-\alpha} D_x^n f(x)$ com $n = [\alpha] + 1$, ou seja:*

$${}_a^C D_x^\alpha f(x) = \frac{1}{\Gamma(n - \alpha)} \int_a^x \frac{f^{(n)}(u)}{(x - u)^{1+\alpha-n}} du \quad (n = [\alpha] + 1, \alpha \in \mathbb{R}_+^*, a \in \mathbb{R}) \quad (2.40)$$

e

$${}_x^C D_b^\alpha f(x) = \frac{(-1)^n}{\Gamma(n - \alpha)} \int_x^b \frac{f^{(n)}(u)}{(u - x)^{1+\alpha-n}} du \quad (n = [\alpha] + 1, \alpha \in \mathbb{R}_+^*, b \in \mathbb{R}), \quad (2.41)$$

onde $f^{(n)}(u) = \frac{d^n f(u)}{du^n}$ são derivadas de ordem inteira n , e $f^{(n)} \in L_1[a, b]$.

Observação 2.3.1. *Uma importante consequência das definições 2.2.1 e 2.3.1 é que derivadas fracionárias de Caputo (assim como as de Riemann-Liouville) são operadores não-locais. A derivada à esquerda (direita) (2.40) e (2.41) depende dos valores da função à esquerda (direita) de x , ou seja, $a \leq u \leq x$ ($x \leq u \leq b$). Por outro lado, é importante notar que as derivadas de Riemann-Liouville e Caputo estão conectadas entre si pelas seguintes relações:*

$${}_a^C D_x^\alpha f(x) = {}_a D_x^\alpha f(x) - \sum_{k=0}^{n-1} \frac{f^{(k)}(a)}{\Gamma(k - \alpha + 1)} (x - a)^{k-\alpha}, \quad (2.42)$$

e

$${}_x^C D_b^\alpha f(x) = {}_x D_b^\alpha f(x) - \sum_{k=0}^{n-1} \frac{f^{(k)}(b)}{\Gamma(k - \alpha + 1)} (b - x)^{k-\alpha}, \quad (2.43)$$

onde $n = [a] + 1$.

Observação 2.3.2. *Devido à derivada dentro da integral nas definições (2.40) e (2.41), é importante observar que, diferente das derivadas de Riemann-Liouville, as derivadas de*

Caputo (2.40) e (2.41) de uma constante são zero para todo $\alpha > 0$,

$${}_a^C D_x^\alpha 1 = 0, \quad {}_x^C D_b^\alpha 1 = 0. \quad (2.44)$$

Devido ao fato de a derivada de Caputo de uma constante ser zero, as equações diferenciais envolvendo derivadas de Caputo (diferentemente do caso de Riemann-Liouville) requerem condições iniciais e de contorno usuais [22, 60].

Observação 2.3.3. Da relação (2.42) e da derivada de Riemann-Liouville da função de Mittag-Leffler de um parâmetro (2.62), podemos observar que a função de Mittag-Leffler de um parâmetro é a autofunção do operador derivada de Caputo, ou seja,

$${}_a^C D_x^\alpha E_\alpha(\lambda x^\alpha) = \lambda E_\alpha(\lambda x^\alpha). \quad (2.45)$$

Observação 2.3.4. Usando a equação (2.14), temos a definição da derivada de Caputo sendo

$${}_0^C D_x^\alpha f(x) = {}_0 J_x^{n-\alpha} D_x^n f(x) = \Phi_{n-\alpha}(x) * D_x^n f(x). \quad (2.46)$$

2.3.1 Relações entre integrais de Riemann-Liouville e derivadas de Caputo

Finalizamos este capítulo com as relações entre as integrais de Riemann-Liouville e as derivadas de Caputo. Assim como nas derivadas de Riemann-Liouville, as derivadas de Caputo também são uma operação inversa às integrais de Riemann-Liouville [22, 60]:

Teorema 2.3.1. Seja $\alpha > 0$. Então para toda função $f \in L_1[a, b]$, temos:

$${}_a D_{x^+}^\alpha J_x^\alpha f(x) = f(x), \quad {}_x D_b^\alpha J_b^\alpha f(x) = f(x) \quad (2.47)$$

e

$${}_a^C D_{x^+}^\alpha J_x^\alpha f(x) = f(x), \quad {}_x^C D_b^\alpha J_b^\alpha f(x) = f(x) \quad (2.48)$$

quase em todo lugar.

Temos também a generalização do Teorema Fundamental do Cálculo para Caputo:

Teorema 2.3.2 (Teorema Fundamental do Cálculo de Caputo). *Considere $0 < \alpha$, e seja $f \in AC^n[a, b]$ ou $f \in C^n[a, b]$ com $n = [\alpha] + 1$. Então, valem as duas igualdades a seguir:*

$${}_a J_{xa}^{\alpha C} D_x^\alpha f(x) = f(x) - \sum_{k=0}^{n-1} \frac{f^{(k)}(a)}{k!} (x-a)^k, \quad (2.49)$$

onde $f^{(k)} = \frac{d^k}{dx^k} f$, e

$${}_b J_{xx}^{\alpha C} D_b^\alpha f(x) = f(x) - \sum_{k=0}^{n-1} \frac{(-1)^k f^{(k)}(b)}{k!} (b-x)^k. \quad (2.50)$$

A demonstração deste teorema pode ser encontrada, por exemplo, em [22, 41, 69].

Observação 2.3.5. *Em particular, se $0 < \alpha < 1$, as equações (2.49) e (2.50) se reduzem a:*

$${}_a J_{xa}^{\alpha C} D_x^\alpha f(x) = f(x) - f(a) \quad (2.51)$$

e

$${}_b J_{xx}^{\alpha C} D_b^\alpha f(x) = f(x) - f(b). \quad (2.52)$$

Assim como no CF de Riemann-Liouville, o Teorema Fundamental do Cálculo de Caputo 2.3.2 será de grande importância para a formulação de um MMC para EDOs fracionárias.

Terminamos esta seção enunciando outro teorema importante, a regra da Integração por partes, que será utilizada para o problema de extremantes de funcionais:

Teorema 2.3.3 (Integração por Partes). *Seja $0 < \alpha < 1$, e f uma função diferenciável em $[a, b]$. Então, para qualquer função $g \in L_1[a, b]$ temos:*

$$\int_a^b g(x) {}_a^C D_x^\alpha f(x) dx = \int_a^b f(x) {}_x D_b^\alpha g(x) dx + [{}_x D_b^{\alpha-1} g(x) f(x)]_a^b \quad (2.53)$$

e

$$\int_a^b g(x) {}_x^C D_b^\alpha f(x) dx = \int_a^b f(x) {}_a D_x^\alpha g(x) dx + [(-1)^n {}_a D_x^{\alpha-1} g(x) f(x)]_a^b. \quad (2.54)$$

A prova deste teorema pode ser encontrada em [69]. Observe que as fórmulas de integração por partes (2.53) e (2.54) relacionam as derivadas de Caputo à esquerda (à direita) com as derivadas de Riemann-Liouville à direita (à esquerda).

2.4 Função de Mittag-Leffler

Uma das funções especiais mais importantes do CF é a função de Mittag-Leffler [59]. Considerada a função rainha do CF [52] é uma generalização da função exponencial, surgindo frequentemente nas soluções de problemas físicos descritos por equações diferenciais e/ou integrais de ordem fracionária. Vamos analisar suas definições e algumas propriedades.

Vamos definir a função de Mittag-Leffler com um e dois parâmetros.

Definição 2.4.1 (Função de Mittag-Leffler de um parâmetro).

$$E_{\alpha}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + 1)} \quad \text{com } \Re(\alpha) > 0, z \in \mathbb{C}. \quad (2.55)$$

Se tomarmos $\alpha = 1$ em 2.4.1 temos

$$E_1(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(k + 1)} = \sum_{k=0}^{\infty} \frac{z^k}{k!} = \exp z. \quad (2.56)$$

A função 2.4.1 surgiu em 1903 no artigo de Mittag-Leffler [59]

Definição 2.4.2 (Função de Mittag-Leffler de dois parâmetros).

$$E_{\alpha,\beta}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + \beta)} \quad \text{com } \Re(\alpha) > 0, \Re(\beta) > 0, z \in \mathbb{C}. \quad (2.57)$$

Se $\beta = 1$, $E_{\alpha,1}(z) = E_{\alpha}(z)$. Temos algumas representações

As funções de Mittag-Leffler são extensões naturais da função exponencial, e as soluções de equações diferenciais de ordem fracionária são frequentemente expressas em termos de funções de Mittag-Leffler, de forma muito semelhante à maneira como as soluções de muitas equações diferenciais de ordem inteira podem ser expressas em

termos de funções exponenciais. A seguir seguem algumas propriedades das funções de Mittag-Leffler:

$$E_{\alpha,\beta}(z) = zE_{\alpha,\alpha+\beta}(z) + \frac{1}{\Gamma(\beta)}, \quad (2.58)$$

$$E_{\alpha,\beta}(z) = \beta E_{\alpha,\beta+1}(z) + \alpha z \frac{d}{dz} E_{\alpha,\beta+1}(z), \quad (2.59)$$

$$\left(\frac{d}{dz}\right)^m [z^{\beta-1} E_{\alpha,\beta}(z^\alpha)] = z^{\beta-m-1} E_{\alpha,\beta-m}(z^\alpha) \quad \Re(\beta - m) > 0, (m \in \mathbb{N}). \quad (2.60)$$

Para ilustrar a aplicação das derivadas de Riemann-Liouville em funções especiais, vamos calcular a derivada à esquerda da função de Mittag-Leffler de um parâmetro:

$$E_\alpha(\lambda x^\alpha) = \sum_{k=0}^{\infty} \frac{(\lambda x^\alpha)^k}{\Gamma(\alpha k + 1)} \quad \text{com } 0 < \alpha < 1. \quad (2.61)$$

No desenvolvimento do cálculo abaixo, utilizamos a equação 2.24 e a equação 2.19.

$$\begin{aligned}
{}_a D_x^\alpha (E_\alpha(\lambda x^\alpha)) &= {}_a D_x^\alpha \left(\sum_{k=0}^{\infty} \frac{(\lambda x^\alpha)^k}{\Gamma(\alpha k + 1)} \right) \\
&= \sum_{k=0}^{\infty} {}_a D_x^\alpha \left(\frac{(\lambda x^\alpha)^k}{\Gamma(\alpha k + 1)} \right) \\
&= \sum_{k=0}^{\infty} \frac{\lambda^k}{\Gamma(\alpha k + 1)} {}_a D_x^\alpha (x^{\alpha k}) \\
&= \sum_{k=0}^{\infty} \frac{\lambda^k}{\Gamma(\alpha k + 1)} \frac{\Gamma(\alpha k + 1)}{\Gamma(\alpha k - \alpha + 1)} x^{\alpha k - \alpha} \\
&= \sum_{k=0}^{\infty} \frac{\lambda^k x^{\alpha(k-1)}}{\Gamma(\alpha(k-1) + 1)} \tag{2.62} \\
&= \frac{x^{-\alpha}}{\Gamma(1 - \alpha)} + \sum_{k=1}^{\infty} \frac{\lambda^k x^{\alpha(k-1)}}{\Gamma(\alpha(k-1) + 1)} \\
&= \frac{x^{-\alpha}}{\Gamma(1 - \alpha)} + \sum_{j=0}^{\infty} \frac{\lambda^{(j+1)} x^{\alpha j}}{\Gamma(\alpha j + 1)} \\
&= \frac{x^{-\alpha}}{\Gamma(1 - \alpha)} + \lambda \sum_{j=0}^{\infty} \frac{(\lambda x^\alpha)^j}{\Gamma(\alpha j + 1)} \\
&= \frac{x^{-\alpha}}{\Gamma(1 - \alpha)} + \lambda E_\alpha(\lambda x^\alpha).
\end{aligned}$$

Como vimos, deste resultado poderemos verificar que a função de Mittag-Leffler de um parâmetro (2.61) é a autofunção do operador derivada fracionária de Caputo.

3 O MÉTODO DE MONTE CARLO

De forma bem genérica e geral, podemos dizer que o MMC se baseia na construção de uma sequência de números aleatórios que converge estatisticamente para o resultado exato de um determinado problema matemático. Neste capítulo, iremos apresentar uma breve revisão do MMC, um pouco da sua história e desenvolvimento, e como podemos utilizá-lo como um método numérico.

3.1 Cálculo do valor de π

Ao longo da história, muitos matemáticos desenvolveram métodos construtivos para determinar o valor do π , um dos mais famosos é devido ao matemático grego Arquimedes de Siracusa que, utilizando o chamado método da exaustão para o cálculo da área de uma circunferência de raio 1, encontrou aproximações do valor da constante π [16].

Nesta seção vamos analisar dois experimentos que ilustram a ideia do MMC. Esses experimentos utilizam a noção de aleatoriedade para construir sequências aleatórias de números para aproximar o valor do número π . O primeiro deles é chamado de “Agulhas de Buffon” e o segundo de “Método do Círculo”. É importante ressaltar que esses experimentos deram início ao desenvolvimento de métodos estatísticos para obter soluções aproximadas. Esses métodos/algoritmos estatísticos são hoje comumente chamados de MMC.

3.1.1 Agulhas de Buffon

Em 1733, o naturalista francês Georges-Louis Leclerc, conde de Buffon, propôs e resolveu o seguinte problema:

Problema 1. *Suponha que temos um piso feito de tiras paralelas de madeira, cada uma com a mesma largura, e colocamos uma agulha no piso. Qual é a probabilidade de a agulha passar por uma linha entre duas tiras?*

Esse problema ficou conhecido como problema da Agulha de Buffon, tendo a sua solução publicada somente no ano de 1777 em [17]. Neste trabalho, foi proposto um método estatístico para cálculo do valor do π .

Em 1812, o matemático francês Pierre-Simon Laplace, no seu artigo [46] sobre a Teoria da Probabilidade, mencionou o problema da agulha, não atribuindo sua origem a Buffon, porém mencionou a possibilidade de utilizar cálculos probabilísticos para determinar de forma experimental aproximações para o valor de π . Por este motivo, este problema também é tratado como "Problema das Agulhas de Buffon-Laplace".

Modelando o problema utilizando a geometria, encontraremos a probabilidade da agulha tocar em qualquer uma das linhas. Para isso, considere uma agulha de comprimento L lançada aleatoriamente entre duas linhas paralelas com uma distância entre elas de D . Seja y a distância entre o ponto médio da agulha até a linha mais próxima; θ é o ângulo que a agulha forma com a horizontal (veja a Figura 2). Considerando que $0 \leq \theta \leq \pi$

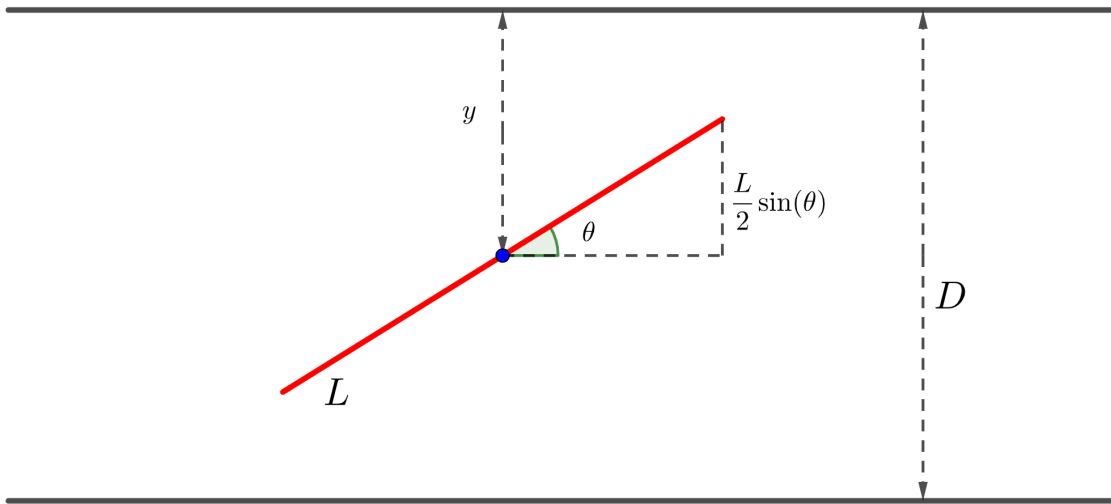


Figura 2: Modelagem Simulação de Buffon.

com $0 \leq y \leq \frac{D}{2}$, observamos que a agulha irá cruzar as linhas paralelas se $y \leq \frac{L}{2} \sin(\theta)$. Pela Figura 3 temos que a probabilidade geométrica [72] disso acontecer é calculada pela razão da área abaixo da curva com a área do retângulo, ou seja

$$P = \frac{\int_0^\pi \frac{L}{2} \sin(\theta) d\theta}{\frac{D}{2} \pi} = \frac{2L}{\pi D}. \quad (3.1)$$

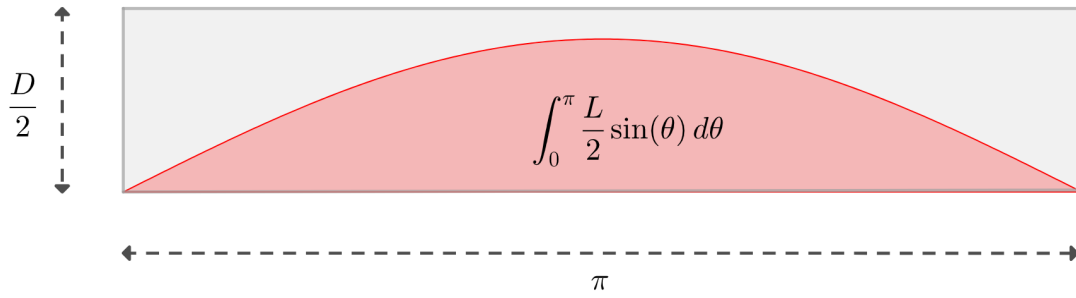
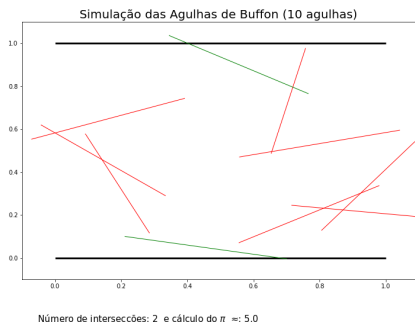


Figura 3: Modelagem Simulação de Buffon.

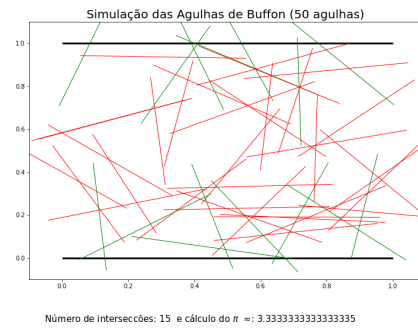
A probabilidade de interseção, P , é definida como o número de interseções pelo número total de agulhas jogadas, ou seja, $P = \frac{w}{n}$, com w sendo o número de agulhas que caíram sob o plano e em interseção com as retas. Já n é o número de agulhas lançadas. Sendo D a distância entre as retas paralelas e L o comprimento de cada agulha, com ($L \leq D$). Lançando aleatoriamente essas n agulhas no plano, da equação (3.1) obtemos o cálculo do valor de π .

$$\pi = \frac{2nL}{wD}. \quad (3.2)$$

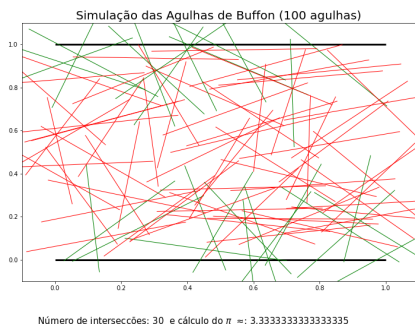
Nas figuras 4 e 5, o MMC para o problema das Agulhas de Buffon é implementado com a equação (3.2) para obter valores aproximados para a constante π . O algoritmo estatístico constituído, o MMC, para este problema consiste em sortear n pontos aleatórios em um plano e n ângulos aleatórios, ambos seguindo uma distribuição uniforme. Esses pontos e ângulos aleatórios vão representar a posição e o ângulo de n agulhas lançadas em um plano. Verifica-se então quantas dessas agulhas cruzam as linhas paralelas, e utiliza-se esse resultado para calcular a aproximação de π através de (3.2). Nas Figuras 4 e 5 apresentamos os resultados que obtivemos considerando os casos de 10, 50, 100 e 1.000 agulhas lançadas aleatoriamente. Os resultados apresentados nestas figuras, assim como todos os resultados apresentados nesta tese, foram implementados na linguagem Python.



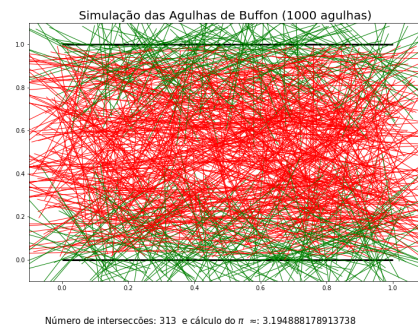
(a) 10 pontos.



(b) 50 pontos.

Figura 4: Cálculo do π com 10 e 50 agulhas.

(a) 100 pontos.



(b) 1000 pontos.

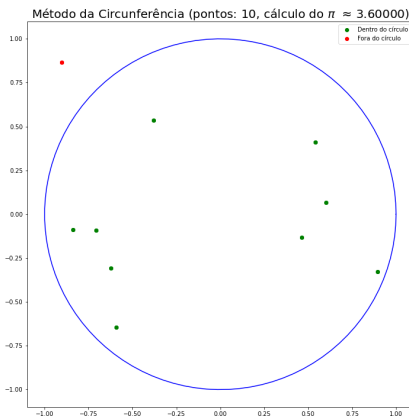
Figura 5: Cálculo do π com 100 e 1000 agulhas.

3.1.2 Método do círculo

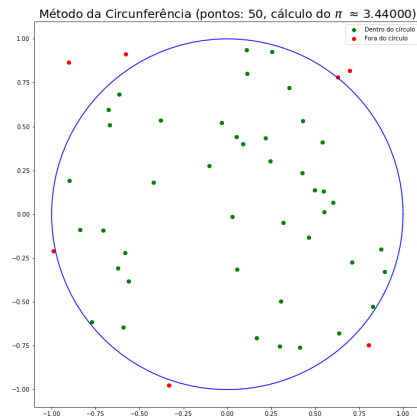
O segundo problema proposto para obter estatisticamente uma aproximação para o valor do número π faz uso de um experimento geométrico bem simples. O experimento consiste em desenhar um círculo de raio 1 inserido em um quadrado de lado 2 (veja a Figura 6). Ao sortear um ponto aleatório no interior do quadrado, uniformemente distribuído, a probabilidade P desse ponto estar no interior do círculo é $P = \frac{A_{cir}}{A_{quad}} = \frac{\pi}{4}$, onde $A_{cir} = \pi$ é a área do círculo e $A_{quad} = 4$ é a área do quadrado. Portanto, sorteados aleatoriamente N pontos, e sabendo que desses pontos $N_{acertos}$ estão no interior do círculo, podemos obter o valor aproximado de π como:

$$\pi \approx \frac{N_{acertos}}{N} A_{quad} = \frac{N_{acertos}}{N} 4. \quad (3.3)$$

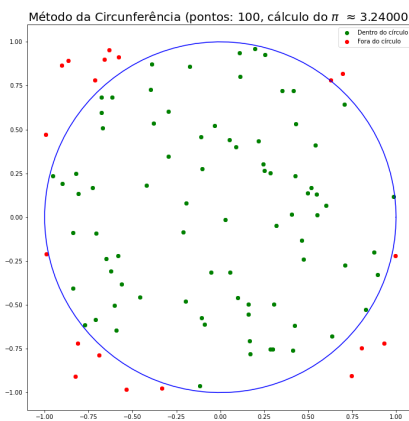
Nas Figuras 6 e 7, o MMC é implementado com a equação (3.3) para obter valores aproximados para a constante π . Neste caso, o MMC consiste em sortear n pontos no interior do quadrado, com distribuição uniforme, e verificar quantos destes pontos estão no interior do círculo. Apresentamos nessas figuras os resultados que obtivemos considerando os casos de 10, 50, 100 e 1000 pontos sorteados.



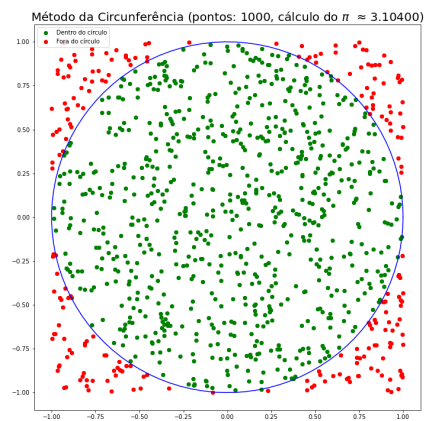
(a) 10 pontos.



(b) 50 pontos.

Figura 6: Cálculo do π com 10 e 50 pontos.

(a) 100 pontos.



(b) 1000 pontos.

Figura 7: Cálculo do π com 100 e 1000 pontos.

3.2 Alguns Teoremas Clássicos da Probabilidade

Nesta seção, iremos anunciar alguns Teoremas Fundamentais da Probabilidade que ajudarão a entender a convergência e o comportamento do MMC.

Para entender os Teoremas Fundamentais da Probabilidade, definem-se alguns conceitos que serão fundamentais para a modelagem probabilística de um experimento. Sendo assim, definem-se primeiramente alguns conceitos bem básicos.

Definição 3.2.1 (Espaço de probabilidade). *Um espaço de probabilidade é o trio $(\Omega, \mathcal{F}, \mathbb{P})$, onde Ω é um conjunto não vazio de resultados possíveis de um experimento, $\mathcal{F}$ é a σ -álgebra de subconjuntos de Ω e $\mathbb{P}$ é a função probabilidade definida em $\mathcal{F}$.*

Observação 3.2.1. Se $\Omega = \mathbb{R}$ podemos definir uma σ -álgebra nos conjuntos abertos em $\mathbb{R}$, chamada de σ -álgebra de Borel sobre $\mathbb{R}$, denotada como $\mathcal{B}(\mathbb{R})$.

Definição 3.2.2 (Definição Formal de Probabilidade). Considere um experimento aleatório $\mathbb{E}$ e um evento $\mathbb{A}$ do espaço amostral Ω . A probabilidade de $\mathbb{A}$, $\mathbb{P}(\mathbb{A})$ é uma função que associa um evento a um número real, satisfazendo os seguintes axiomas:

$$i) \mathbb{P}(\mathbb{A}) \geq 0 \quad \forall \mathbb{A} \subseteq \Omega;$$

$$ii) \mathbb{P}(\Omega) = 1;$$

$$iii) \text{ Sendo } \mathbb{A} \text{ e } \mathbb{B}, \text{ tais que } \mathbb{A} \cap \mathbb{B} = \emptyset, \text{ tem-se que } \mathbb{P}(\mathbb{A} \cup \mathbb{B}) = \mathbb{P}(\mathbb{A}) + \mathbb{P}(\mathbb{B}).$$

Com a definição 3.2.1 e a com a observação 3.2.1 define-se o conceito de variável aleatória.

Definição 3.2.3 (Variável aleatória). Seja $(\Omega, \mathcal{F}, \mathbb{P})$ um espaço de probabilidade. Uma variável aleatória real é uma função $X : \Omega \rightarrow \mathbb{R}$ de forma que, para qualquer conjunto Boreliano B , com $B \in \mathcal{B}(\mathbb{R})$, se satisfaz que o conjunto $X^{-1}(B)$, chamado de imagem inversa, é um elemento de $\mathcal{F}$, isto é,

$$X^{-1}(B) = \{\omega \in \Omega : f(\omega) \in B\} \in \mathcal{F},$$

para todo $B \in \mathcal{B}(\mathbb{R})$.

Observação 3.2.2. Uma variável aleatória X é discreta se sua imagem for um conjunto finito ou enumerável. Se a imagem for um conjunto não enumerável, dizemos que a variável aleatória é contínua.

Uma observação com relação à definição 3.2.3 é que operações básicas entre variáveis aleatórias, como produto por um escalar, soma escalar, soma, produto e quociente de variáveis aleatórias continua sendo uma variável aleatória. Como consequência, temos que uma soma finita de variáveis aleatórias ainda é uma variável aleatória.

Definição 3.2.4. (Função distribuição de Probabilidade e Função densidade de Probabilidade) Seja X uma variável aleatória com valores em um domínio arbitrário Ω .

A função distribuição de Probabilidade é dada por

$$F_X(D) = \mathbb{P}\{X \in D\}, \quad \forall \text{ conjunto mensurável } D \subseteq \Omega. \quad (3.4)$$

A função densidade de Probabilidade $f_X(x)$ é definida pela derivada de Radon-Nikodym como

$$f_X(x) = \frac{dF_X}{d\lambda}(x) \text{ onde } \lambda \text{ é a medida.} \quad (3.5)$$

Da equação (3.5) temos

$$F_X(D) = \int_D f_X(x) d\lambda. \quad (3.6)$$

Definição 3.2.5 (Valor esperado, variância e desvio padrão). *Seja $Y = g(X)$ uma variável aleatória. O valor esperado é dado por*

$$\mathbb{E}[Y] = \int_{\Omega} g(X) f_X(x) d\lambda. \quad (3.7)$$

A variância é dada por

$$\sigma^2[Y] = \mathbb{E}[(Y - \mathbb{E}[Y])^2] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2. \quad (3.8)$$

O desvio padrão é dado por

$$\sigma[Y] = \sqrt{\sigma^2[Y]}. \quad (3.9)$$

Teorema 3.2.1 (Lei dos Grandes Números). *Sejam $X_i, i = 1, 2, \dots, N$ variáveis aleatórias independentes e identicamente distribuídas (i.i.d.), com $\mathbb{E}(X_i) = \mu < \infty$, então:*

i)

$$\mathbb{P} \left(\lim_{N \rightarrow \infty} \frac{X_1 + X_2 + \dots + X_N}{N} = \mu \right) = 1, \quad (3.10)$$

ou seja $\frac{\sum_i X_i}{N} \rightarrow \mu$ converge quase certamente quando $N \rightarrow \infty$, nesse caso, temos a Lei Forte dos Grandes Números.

ii)

$$\lim_{N \rightarrow \infty} \mathbb{P} \left(\frac{X_1 + X_2 + \dots + X_N}{N} = \mu \right) = 1, \quad (3.11)$$

ou seja $\frac{\sum_i X_i}{N} \rightarrow \mu$ converge em probabilidade quando $N \rightarrow \infty$, nesse caso, temos a Lei Fraca dos Grandes Números.

Teorema 3.2.2 (Teorema do Limite Central). *Sejam $X_i, i = 1, 2, \dots, N$ variáveis aleatórias independentes, cada uma com a sua função de densidade de probabilidade $f_i(x)$ (podendo ser distintas) com média μ_i e variância σ_i^2 . A variável aleatória $Z = \frac{\sum_i X_i}{N}$, ou seja, a média dos X_i , tem as seguintes propriedades:*

i) *O valor esperado é $\langle Z \rangle = \frac{\sum_i \mu_i}{N}$;*

ii) *A variância é $\sigma^2[Z] = \frac{\sum_i \sigma_i^2}{N^2}$;*

iii) *E quando $N \rightarrow \infty$ a função de densidade de probabilidade de Z tende a uma distribuição normal com a mesma média e variância, ou seja*

$$\lim_{N \rightarrow \infty} \mathbb{P} \left(\frac{|Z - \langle Z \rangle|}{\sigma[Z]/\sqrt{N}} \leq \beta \right) = \frac{1}{\sqrt{2\pi}} \int_{-\beta}^{\beta} e^{-t^2/2} dt. \quad (3.12)$$

A demonstração do Teorema 3.2.2 pode ser encontrada em [24].

Analisando a equação (3.12) para um valor de N grande, temos

$$\mathbb{P}\left(Z - \beta\sigma[Z]/\sqrt{N} \leq \langle Z \rangle \leq Z + \beta\sigma[Z]/\sqrt{N}\right) \approx \frac{1}{\sqrt{2\pi}} \int_{-\beta}^{\beta} e^{-t^2/2} dt. \quad (3.13)$$

O lado direito da equação acima pode ser calculado numericamente e podemos definir como sendo o Coeficiente de Confiança. Esses coeficientes, expressos por porcentagens e arredondados ou truncados, são chamados de limites de confiança. Os resultados para vários valores de β são mostrados na Tabela 1 de acordo com [24].

β	Coeficiente de Confiança	Intervalo de Confiança
0.25	0,1974	20%
0.50	0,3829	38%
1.00	0,6827	68%
1.50	0,8664	87%
2.00	0,9545	95%
3.00	0,9973	99.7%
4.00	0,9999	99.99%

Tabela 1: Limites de confiança para vários valores de acordo com [24]

Se consideramos $\beta = 2$ (dois erros padrão) na equação (3.13), temos que no intervalo $I_{2,N} = [Z - 2\sigma[Z]/\sqrt{N}, Z + 2\sigma[Z]/\sqrt{N}]$ tem 95.4% de chance de conter o valor verdadeiro da média. Considerando $I_{\beta,N} = [Z - \beta\sigma[Z]/\sqrt{N}, Z + \beta\sigma[Z]/\sqrt{N}]$ se N aumenta, o comprimento do Intervalo de Confiança ($comp(I_{\beta,N})$) é proporcional à $1/\sqrt{N}$, logo $comp(I_{\beta,N}) \rightarrow 0$, se $N \rightarrow \infty$.

3.3 O desenvolvimento do Método de Monte Carlo

Na sua autobiografia [78], *Adventures of a Mathematician*, o matemático polonês Stanislaw Marcin Ulam menciona como ele teve a brilhante ideia de aplicar números aleatórios para resolver vários problemas em que ele trabalhava. Ulam usou os computadores rápidos de Los Alamos para resolver uma ampla variedade de problemas de maneira estatística (jogo de cartas), usando números aleatórios.

A vantagem do método é que o computador pode ser eficientemente programado para executar cada passo em um jogo particular de acordo com probabilidades conhecidas, e o resultado pode ser determinado com qualquer precisão desejada, dependendo do número de jogos de amostra computados. O jogo de paciência é um exemplo de como o método pode ser usado para resolver problemas de outra forma intratáveis com computação bruta.

O método recebeu o nome de MMC em 1944 durante o projeto Manhattan na Segunda

Guerra Mundial, devido à similaridade das simulações estatísticas com os jogos de azar, já que Monte Carlo (capital do Principado de Mônaco) era um centro de cassinos, apostas e jogos, e também como forma de codinome, visto que eles estavam trabalhando em um projeto secreto. No projeto de construção da bomba atômica, Ulam, von Neumann e Fermi simularam o comportamento de nêutrons em um ambiente nuclear, onde cada interação do nêutron com outros núcleos é determinada aleatoriamente, considerando as probabilidades de espalhamento e fissão nuclear, assim, obtiveram informações sobre o comportamento dos nêutrons no sistema, tais como a quantidade de sua energia e a sua posição. A publicação do artigo de Metropolis e Ulam em 1949 [56] impulsionou a pesquisa e os desenvolvimentos da teoria e dos algoritmos do MMC.

O MMC é, em sua essência, uma técnica que usa números aleatórios na resolução de diversos problemas, especialmente em física, biologia, engenharias e finanças [71]. Formalmente, o MMC é baseado em dois resultados estatísticos fundamentais: a Lei dos Grandes Números (3.2.1) e o Teorema do Limite Central (3.2.2).

3.3.1 Cálculo de integrais definidas

Nessa subseção vamos analisar uma das principais aplicações do MMC, que consiste no cálculo de integrais definidas. Vamos apresentar como o MMC pode ser utilizado de forma simples para calcular integrais usando o Teorema do Valor Médio para integrais.

Teorema 3.3.1 (Método de Monte Carlo para Integrais). *Seja $f \in C([a, b])$, $[a, b] \subset \mathbb{R}$, tal que $a < b$ e um conjunto de N números $(x_1, x_2, \dots, x_N)$ aleatórios uniformemente distribuídos em $[a, b]$, então podemos ter uma aproximação para a integral da função $f(x)$ no domínio $[a, b]$ dada por*

$$\int_a^b f(x)dx \approx \frac{b-a}{N} \sum_{i=1}^N f(x_i). \quad (3.14)$$

Prova: Como a função $f(x)$ é contínua no intervalo, $[a, b]$ então ela é integrável no sentido de Riemann. Assim, pelo Teorema do Valor Médio para integrais, temos que

$$\int_a^b f(x)dx = (b-a)\langle f \rangle, \quad (3.15)$$

onde $\langle f \rangle$ é o valor médio da função em $[a, b]$. Neste ponto, uma amostra de números aleatórios pode ser utilizada para calcular aproximadamente $\langle f \rangle$. Dado um conjunto de N números $(x_1, x_2, \dots, x_N)$ aleatórios uniformemente distribuídos em $[a, b]$, um valor aproximado $\langle f \rangle_N$ para $\langle f \rangle$ é obtido simplesmente calculando $f(x)$ para cada número aleatório selecionado. Obtemos:

$$\langle f \rangle_N = \frac{1}{N} \sum_{i=1}^N f(x_i). \quad (3.16)$$

Pela Lei dos Grandes Números (Teorema 3.2.1) temos que

$$\mathbb{P}(\lim_{N \rightarrow \infty} \langle f \rangle_N = \langle f \rangle) = 1, \quad (3.17)$$

assim $\langle f \rangle_N$ converge estatisticamente para $\langle f \rangle$ no limite $N \rightarrow \infty$, ou seja, $\langle f \rangle = \lim_{N \rightarrow \infty} \langle f \rangle_N$ [24, 39]. Portanto, (3.16) nos fornece uma aproximação para a integral (3.15), e também converge estatisticamente para o valor exato da integral quando $N \rightarrow \infty$, ou seja,

$$\int_a^b f(x)dx = \lim_{N \rightarrow \infty} \frac{b-a}{N} \sum_{i=1}^N f(x_i). \quad \square \quad (3.18)$$

Teorema 3.3.2 (Convergência). *Supondo que $\sigma^2[f] < \infty$ temos as seguintes propriedades:*

i) *O Erro padrão ao realizar a aproximação (3.16) é dado por*

$$Err = \frac{\sqrt{\sigma^2}}{\sqrt{N}}. \quad (3.19)$$

ii) *A aproximação (3.16) converge para distribuição normal quando $N \rightarrow \infty$, ou seja,*

$$\lim_{N \rightarrow \infty} \mathbb{P} \left(\left| \frac{\frac{1}{N} \sum_{i=1}^N f(x_i) - \frac{1}{b-a} \int_a^b f(x)dx}{Err} \right| \leq \beta \right) = \frac{1}{\sqrt{2\pi}} \int_{-\beta}^{\beta} e^{-t^2/2} dt. \quad (3.20)$$

Prova (i): Na equação (3.16) faça $Y_i = f(x_i)$, assim temos

$$\langle f \rangle_N = \frac{1}{N} \sum_{i=1}^N Y_i. \quad (3.21)$$

Calculando a variância de Y_i

$$\sigma^2[Y_i] = \mathbb{E}[Y_i^2] - [Y_i]^2 = \int_a^b f^2(x_i)dx_i - \left(\int_a^b f(x_i)dx_i \right)^2. \quad (3.22)$$

Como $\sigma^2[Y_i]$ é finito, por hipótese e observando que $\sigma^2[\langle f \rangle_N]$ diminui linearmente com N , temos

$$\sigma^2[\langle f \rangle_N] = \sigma^2 \left[\frac{1}{N} \sum_{i=1}^N Y_i \right] = \frac{1}{N^2} \sigma^2 \left[\sum_{i=1}^N Y_i \right] = \frac{1}{N^2} \sum_{i=1}^N \sigma^2[Y_i] = \frac{1}{N} \sigma^2[Y] \quad (3.23)$$

onde consideramos que $\sigma^2[kY] = k^2\sigma^2[Y]$ e que os Y_i são amostras independentes. Assim, o Erro Padrão é dado conforme mostra a equação (3.19). $\square$

Prova (ii): Basta aplicar o Teorema do Limite Central (3.2.2) na distribuição (3.16). $\square$

Observação 3.3.1. *A integração usando o MMC, como vimos, converge a uma taxa de $O(N^{-1/2})$ em qualquer dimensão espacial, independente da suavidade da função a ser integrada. Portanto, o método não sofre o efeito dimensional, o que afetaria o seu desempenho à medida que a dimensão aumenta.*

Observação 3.3.2. *O MMC convergirá mesmo que a variância seja infinita, desde que o valor esperado exista; nesse caso, a convergência será mais lenta. Esse fato é devido à lei dos grandes números (3.17).*

Observação 3.3.3. *O Teorema do Limite Central dá suporte ao MMC, pois fornece uma estimativa da incerteza no valor esperado estimado. Além disso, ele mostra que a incerteza no valor esperado é proporcional a $\frac{1}{\sqrt{N}}$, onde N é o número de amostras de $f(x)$. Se o número de amostras for quadruplicado, a incerteza na estimativa da média amostral é reduzida à metade.*

Observação 3.3.4. *A aproximação (3.14) dá um resultado para a integral que difere por até um erro padrão do valor exato com probabilidade de 68.3% (e difere por até dois erros padrões do valor exato com probabilidade de 95.4%, veja tabela 1). Como o erro estimado é proporcional à variância e inversamente proporcional à raiz quadrada de N (erro diminui à taxa $O(N^{-1/2})$), existem dois métodos de reduzir o erro. O primeiro é aumentar N , e o segundo é reduzir a variância usando N números aleatórios $(x_1, x_2, \dots, x_N)$ não uniformemente distribuídos em $[a, b]$ [24, 39].*

Observação 3.3.5. *Existem técnicas de redução da variância do cálculo da integral definida [24] entre quais podemos destacar: Amostragem por importância, Amostragem estratificada, Amostragem correlacionada e Variáveis antiéticas.*

3.3.2 Algoritmo de Metropolis

Como vimos, o MMC utiliza números aleatórios e uma das técnicas para geração destes números aleatórios foi analisada em 1953 no artigo [55] e ficou conhecida como algoritmo de Metropolis. Sua construção foi motivada por uma analogia com sistemas da mecânica estatística que alcançam estados de equilíbrio; nesses estados, as propriedades estatísticas do sistema são determinadas exclusivamente por fatores termodinâmicos, independentes dos processos cinéticos específicos que levam ao equilíbrio [39]. A vantagem deste algoritmo está no fato de que, por meio do uso de números aleatórios, ele fornece uma maneira elegante e eficiente de obter respostas para problemas que são complicados de resolver analiticamente. Vamos descrever de forma simples a sua implementação.

Suponha que queremos gerar uma coleção de pontos x_i com $i = 1, \dots, n$, que seguem uma determinada função de densidade de probabilidade (3.5) $f(x)$. O algoritmo de Metropolis faz uma simulação aleatória que propõe iterativamente novos pontos e aceita ou rejeita-os com base em um critério probabilístico que garante a convergência para a distribuição alvo $f(x)$. Como exemplo, vamos gerar uma sequência aleatória:

1. Escolha um ponto inicial x_0 .
2. Escolha um tamanho máximo de passo fixo h .
3. Gere o próximo ponto x_1 .
4. Suponha que o caminhante esteja no ponto x_i .
5. Escolha uma nova configuração x_{novo} aleatoriamente e uniformemente no intervalo $[x_i - h/2, x_i + h/2]$.
6. Calcule a razão

$$r = \frac{f(x_{novo})}{f(x_i)}.$$
7. Se $r \geq 1$, o passo tentativa é aceito, ou seja, $x_{i+1} = x_{novo}$.
8. Se $r < 1$, o passo tentativa é aceito com uma probabilidade r . Escolha um número aleatório $\eta \in [0, 1]$ e a próxima configuração é gerada de acordo com

$$\begin{cases} x_{i+1} = x_{novo}, & \text{se } \eta < r, \\ x_{i+1} = x_i & \text{se } \eta \geq r. \end{cases}$$

O algoritmo de Metropolis é uma ferramenta extremamente útil, mas sua implementação eficaz requer a escolha cuidadosa do ponto de partida x_0 e do tamanho do passo h . Ao considerar esses fatores, é possível obter amostras precisas de distribuições de probabilidade complexas. Podemos adaptar a ideia desse algoritmo para trabalhar com otimização de funcionais, ou seja, empregar as funções como pontos em um espaço de funções e usar o Metropolis para explorar esse espaço, buscando o ponto de menor valor do funcional.

3.3.3 Monte Carlo Variacional

O Método de Monte Carlo Variacional (MCV) é uma técnica que faz uma combinação com a teoria variacional [82] e com a técnica de Monte Carlo para analisar os valores esperados em sistemas quânticos. Foi aplicada com sucesso pela primeira vez por McMillan [53] para estudar o estado fundamental do hélio-4.

O MCV emprega um conjunto de parâmetros ajustáveis para produzir uma função de onda de teste Ψ_T , cuja forma é escolhida após uma análise prévia do sistema físico em

investigação. A função de onda de teste Ψ_T , quando melhor otimizada, aproxima a função de onda exata. Se a função de onda de teste Ψ_T for idêntica à função de onda exata do estado fundamental, então o método MCV fornece a estimativa exata da energia do estado fundamental. O MCV também é chamado de método Monte Carlo Quântico [48].

4 O MÉTODO DE MONTE CARLO PARA INTEGRAIS E DERIVADAS FRACIONÁRIAS

Neste capítulo, mostraremos como se pode calcular integrais e derivadas fracionárias pelo MMC, tanto usando a definição de Riemann-Liouville quanto à definição de Caputo.

4.1 O MMC para integrais de Riemann-Liouville

Na presente seção, mostramos como podemos calcular integrais fracionárias pelo MMC para as integrais definidas apresentando na seção 3.3.1. Vamos considerar as integrais fracionárias de Riemann-Liouville (2.6) e (2.7). Podemos calcular essas integrais por um MMC usando a aproximação (3.14). Temos:

$${}_a J_x^\alpha f(x) \approx \frac{1}{\Gamma(\alpha)} \frac{x-a}{N} \sum_{i=1}^N \frac{f(x_i)}{(x-x_i)^{1-\alpha}}, \quad (4.1)$$

onde selecionamos aleatoriamente N números $(x_1, x_2, \dots, x_N)$ uniformemente distribuídos em $[a, x]$, e

$${}_b J_x^\alpha f(x) \approx \frac{1}{\Gamma(\alpha)} \frac{b-x}{N} \sum_{i=1}^N \frac{f(x_i)}{(x_i-x)^{1-\alpha}}, \quad (4.2)$$

para N números aleatórios $(x_1, x_2, \dots, x_N)$ uniformemente distribuídos em $[x, b]$. Como (2.6) e (2.7) são definidas para funções $L^1[a, b]$, o MMC (4.2) converge. A convergência seguirá o erro padrão Err (3.19) quando o integrando em (2.6) e (2.7) for $L^2[a, b]$, e a convergência será mais lenta que Err quando o integrando for $L^1[a, b]$.

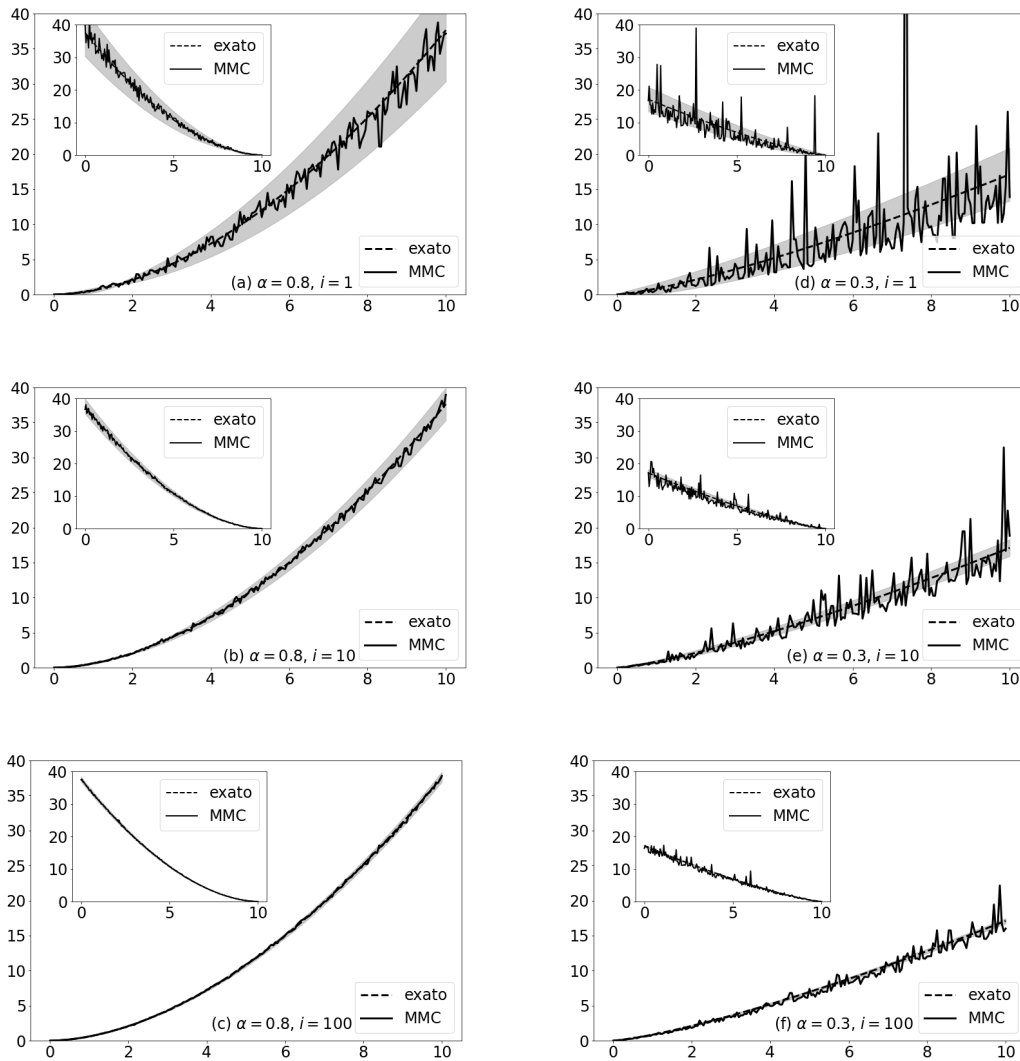


Figura 8: O MMC converge para as integrais fracionárias à esquerda de Riemann-Liouville de ordem $\alpha = 0.3, 0.8$ para $f(x) = x$. Em (a), (b) e (c) temos $\alpha = 0.8, i = 1, 10$, e 100, respectivamente. Os insertos nos painéis (a), (b) e (c) apresentam a convergência para as integrais fracionárias à direita de Riemann-Liouville para $f(x) = 10 - x$. Em (d), (e), e (f) temos $\alpha = 0.3, i = 1, 10$, e 100, respectivamente. Os insertos nos painéis (d), (e), e (f) apresentam a convergência para as derivadas fracionárias à direita de Riemann-Liouville para $f(x) = 10 - x$.

Para ilustrar o uso do MMC para calcular integrais fracionárias de Riemann-Liouville, na Figura 8 apresentamos o resultado de integrais fracionárias à esquerda (direita) com ordens $\alpha = 0.3, 0.8$ da função $f(x) = x$ ($f(x) = 10 - x$). Para obter as soluções pelo MMC apresentadas nos gráficos (a), (b), (c), (d), (e), e (f), o domínio $[a, b]$ ($a = 0$ e $b = 10$) foi dividido em $L = 200$ partes iguais, e foi realizada uma amostragem de i números aleatórios em cada uma dessas partes do domínio, ou seja, para a n -ésima parte do domínio temos uma amostra de i números aleatórios distribuídos uniformemente no

intervalo $[a + (n - 1)\Delta x, a + n\Delta x]$, onde $\Delta x = \frac{b-a}{L}$. Portanto, a amostragem total no intervalo $[a, x]$ é $N_x = in$ para $a + (n - 1)\Delta x \leq x < a + n\Delta x$. A região destacada em cinza contém os pontos do plano que diferem do valor exato da integral fracionária por até três erros padrão. Desses gráficos, podemos observar que o MMC converge estatisticamente para o valor exato da integral. É importante notar nos gráficos (a), (b), e (c) da Figura 8 que o MMC converge em concordância com o erro padrão Err (3.19) para $\alpha = 0.8$, como esperado, pois neste caso os integrandos em (2.6) e (2.7) são L^2 . Por outro lado, a convergência é mais lenta do que Err para $\alpha = 0.3$, como pode ser visto nos gráficos (d), (e), e (f), pois neste caso, os integrandos são L^1 . Finalmente, é importante reforçar que, por razões didáticas, consideramos valores pequenos para $N = 200i$ para todos os exemplos para enfatizar o caráter estatístico da solução obtida pelo MMC. Um valor maior do que os utilizados para N resultaria em gráficos visualmente indistinguíveis da solução exata.

4.2 O MMC para derivadas de Riemann-Liouville

O MMC também pode ser usado para calcular derivadas fracionárias de Riemann-Liouville usando as expressões (4.1), (4.2) e a aproximação $\frac{df(x)}{dx} \approx \frac{f(x+h)-f(x)}{h}$ para um pequeno $h \in \mathbb{R}$. Por simplicidade, vamos considerar $0 < \alpha \leq 1$. Seja agora f uma função $L^1[a, b]$, isto é, a integral fracionária de f existe para todo $x \in [a, b]$. Assim, pela Definição 2.2.1 e das equações (4.1) e (4.2), um MMC para calcular derivada fracionária de Riemann-Liouville de ordem $0 < \alpha \leq 1$ é dada por:

$$\begin{aligned} {}_a D_x^\alpha f(x) &= \frac{d}{dx} {}_a J_x^{1-\alpha} f(x) \approx \frac{1}{h} ({}_a J_{x+h}^{1-\alpha} f(x) - {}_a J_x^{1-\alpha} f(x)) \\ &\approx \frac{1}{\Gamma(1-\alpha)} \left(\frac{x+h-a}{h\hat{N}} \sum_{i=1}^{\hat{N}} \frac{f(\hat{x}_i)}{(x+h-\hat{x}_i)^\alpha} - \frac{x-a}{hN} \sum_{i=1}^N \frac{f(x_i)}{(x-x_i)^\alpha} \right), \end{aligned} \quad (4.3)$$

onde selecionamos aleatoriamente N números $(x_1, x_2, \dots, x_N)$ uniformemente distribuídos em $[a, x]$, e $\hat{N}$ números $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_{\hat{N}})$ uniformemente distribuídos em $[a, x+h]$, e

$$\begin{aligned} {}_x D_b^\alpha f(x) &= -\frac{d}{dx} {}_b J_x^{1-\alpha} f(x) \approx \frac{1}{h} ({}_b J_x^{1-\alpha} f(x) - {}_b J_{x+h}^{1-\alpha} f(x)) \\ &\approx \frac{1}{\Gamma(1-\alpha)} \left(\frac{b-x}{hN} \sum_{i=1}^N \frac{f(x_i)}{(x_i-x)^\alpha} - \frac{b-x-h}{h\hat{N}} \sum_{i=1}^{\hat{N}} \frac{f(\hat{x}_i)}{(\hat{x}_i-x-h)^\alpha} \right), \end{aligned} \quad (4.4)$$

para N números aleatórios $(x_1, x_2, \dots, x_N)$ uniformemente distribuídos em $[x, b]$, e $\hat{N}$ números $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_{\hat{N}})$ uniformemente distribuídos em $[x+h, b]$.

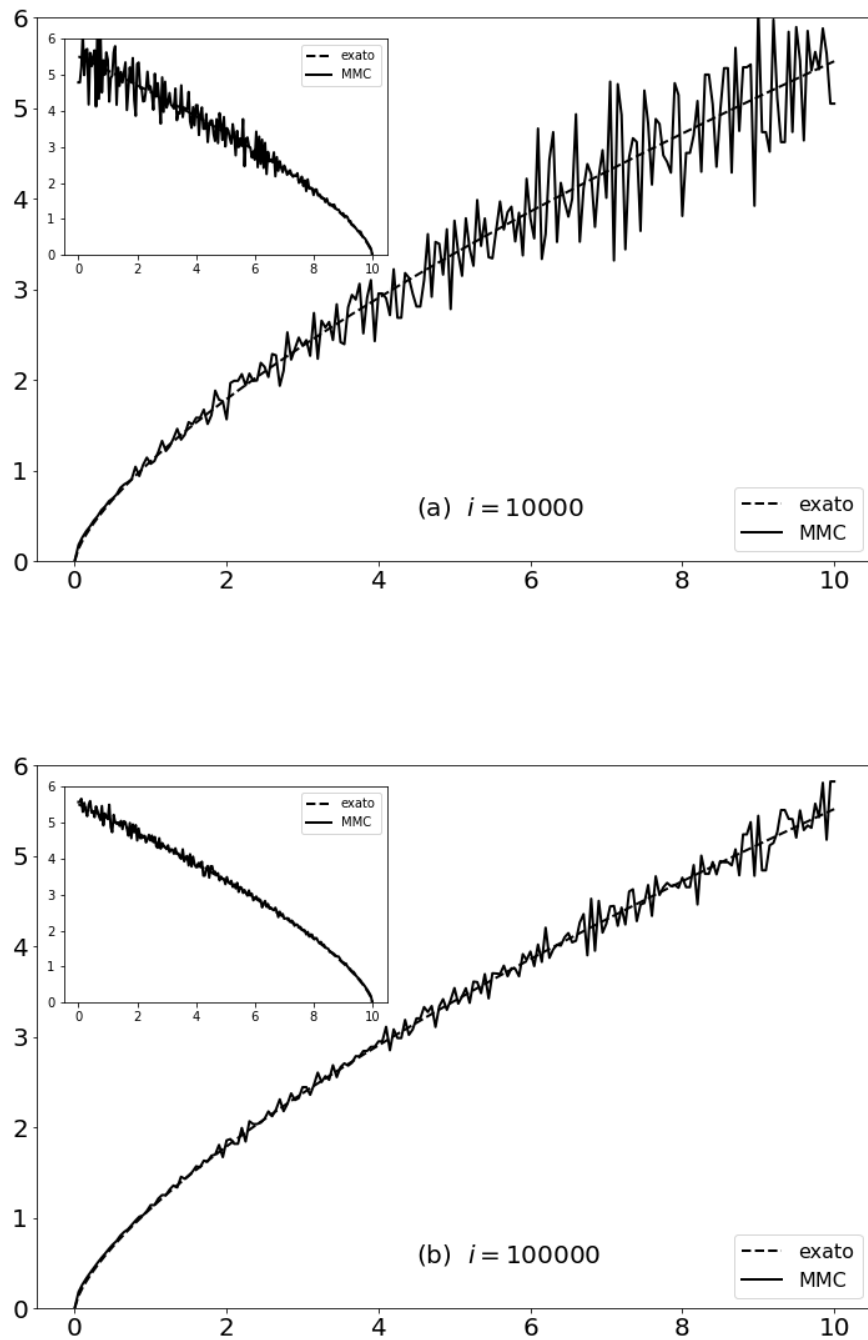


Figura 9: Convergência do MMC para a derivada à esquerda de Riemann-Liouville de ordem $\alpha = 0.3$ para a função $f(x) = x$. No gráfico (a) temos $i = 10000$, e no gráfico (b) temos $i = 100000$. Nas inserções dos gráficos (a) e (b) apresentamos a convergência para a derivada à direita de Riemann-Liouville da função $f(x) = 10 - x$.

Na Figura 9 comparamos a solução exata da derivada à esquerda de Riemann-Liouville de ordem $\alpha = 0.3$ da função $f(x) = x$ com a solução obtida pelo MMC. O domínio

foi novamente dividido em $L = 200$ partes e foi feita uma amostragem de $i = 10000$ e $i = 100000$ pontos aleatórios em cada uma dessas partes ($N_x = in$, para $a + (n-1)\Delta x \leq x < a + n\Delta x$ onde n é a n -ésima parte do domínio e $\Delta x = \frac{b-a}{L}$). Além disso, escolhemos $h = \Delta x$. Vemos que o MMC proposto (4.3) converge estatisticamente para a solução exata. No entanto, é importante observar que a convergência da solução é muito mais lenta para derivadas do que para integrais de Riemann-Liouville. Isto ocorre porque no cálculo das derivadas de Riemann-Liouville os erros estatísticos do cálculo das duas integrais fracionárias em (4.3) e (4.4) se somam e são amplificados pela divisão pelo fator h . No entanto, se a função $f(x)$ é diferenciável, uma forma mais conveniente de calcular as derivadas de Riemann-Liouville de $f(x)$, através de um MMC, é calcular derivadas de Caputo (como veremos a seguir) e usar as relações (2.42) e (2.43). Finalmente, como veremos no próximo capítulo, para a resolução de problemas de valor inicial de EDOs fracionárias só precisaremos calcular integrais fracionárias pelo MMC. Consequentemente, no problema de resolução de EDOs, a convergência do MMC será rápida.

4.3 O MMC para derivadas de Caputo

Assim como no caso das derivadas de Riemann-Liouville, as derivadas fracionárias de Caputo podem ser calculadas usando (4.1), (4.2) e aproximando a derivada de ordem inteira $\frac{df(x)}{dx} \approx \frac{f(x+h)-f(x)}{h}$ para um pequeno $h \in \mathbb{R}$. Para simplificar, consideremos novamente $0 < \alpha \leq 1$. Seja agora f uma função diferenciável fracionária em $[a, b]$, isto é, a derivada fracionária de Caputo de f existe para todo $x \in [a, b]$. Assim, pela Definição 2.3.1, (4.1) e (4.2), um MMC para calcular derivadas fracionárias de Caputo de ordem $0 < \alpha \leq 1$ é dada por:

$$\begin{aligned} {}^C D_x^\alpha f(x) &= {}_a J_x^{1-\alpha} \frac{d}{dx} f(x) \approx \frac{1}{h} {}_a J_x^{1-\alpha} (f(x+h) - f(x)) \\ &\approx \frac{1}{\Gamma(1-\alpha)} \frac{x-a}{hN} \sum_{i=1}^N \frac{f(x_i+h) - f(x_i)}{(x-x_i)^\alpha}, \end{aligned} \quad (4.5)$$

onde selecionamos aleatoriamente N números $(x_1, x_2, \dots, x_N)$ uniformemente distribuídos em $[a, x]$, e

$$\begin{aligned} {}^C D_b^\alpha f(x) &= -{}_b J_x^{1-\alpha} \frac{d}{dx} f(x) \approx -\frac{1}{h} {}_b J_x^{1-\alpha} (f(x+h) - f(x)) \\ &\approx \frac{1}{\Gamma(1-\alpha)} \frac{b-x}{hN} \sum_{i=1}^N \frac{f(x_i+h) - f(x_i)}{(x_i-x)^\alpha}, \end{aligned} \quad (4.6)$$

para N números aleatórios $(x_1, x_2, \dots, x_N)$ uniformemente distribuídos em $[x, b]$.

Na Figura 10 comparamos a solução obtida pelo MMC com a solução exata para derivadas de Caputo, à esquerda e à direita, com ordens $\alpha = 0.3, 0.5$ e 0.8 para a função

$f(x) = x$. Para obter a solução pelo MMC apresentada nos gráficos (a) e (b), o domínio foi novamente dividido em $L = 200$ partes iguais, e foi feita uma amostragem de i números aleatórios em cada uma dessas partes ($N_x = in$, para $a + (n - 1)\Delta x \leq x < a + n\Delta x$ onde n é a n -ésima parte do domínio e $h = \Delta x = \frac{b-a}{L}$). Vemos que o MMC proposto em (4.5) e (4.6) funciona, e a solução obtida converge estatisticamente para a solução exata muito mais rapidamente para derivadas de Caputo do que para derivadas de Riemann-Liouville. Em particular, vemos que a solução para a derivada de Caputo de ordem $\alpha = 0.3$, com $i = 100$, difere da solução exata muito menos do que a solução obtida para a derivada de Riemann-Liouville com $i = 100000$ (veja a Figura 9). Este fato era esperado, pois as derivadas de Caputo se reduzem ao cálculo de uma integral fracionária de ordem $1 - \alpha$ da derivada de ordem inteira da função e, como discutido na seção 4.1, o MMC converge bem para integrais fracionárias de ordem $1 - \alpha > 0.5$.

Outro aspecto importante que podemos observar na Figura 10 é que, conforme a ordem da derivada se aproxima de 1, a convergência vai ficando mais lenta. Para a ordem $\alpha = 0.3$, a solução obtida, com ambos $i = 100$ e $i = 1000$, está muito mais próxima da solução exata do que as soluções para $\alpha = 0.5$ e $\alpha = 0.8$ (de fato, o pior resultado ocorre para $\alpha = 0.8$). O resultado ser pior para $\alpha = 0.8$ era esperado, pois os integrandos em (2.40) e (2.41) são L^2 somente para $\alpha < 0.5$. Para $\alpha = 0.8$, além das flutuações estatísticas serem muito maiores, o próprio comportamento da solução obtida pelo MMC parece estar deslocado para valores menores do que a solução exata. Este último fato está relacionado à importância do polo no integrando (da definição da derivada de Caputo (2.40) e (2.41)) para o valor final da derivada. Quanto mais α está próximo de 1, maior será a contribuição do polo para a integral definindo a derivada de Caputo. Como o cálculo dessa integral é feito por uma amostragem aleatória uniformemente distribuída, para i pequeno, a maioria dos números aleatórios $(x_1, x_2, \dots, x_{N_x})$ usados no cálculo dessa integral estarão distantes do polo, contribuindo pouco para o cálculo da integral e, conseqüentemente, subestimando o valor da derivada. Além disso, as maiores flutuações estatísticas, em geral, superestimam a solução (em todos os casos). Isto também é uma consequência direta do polo no integrando da derivada de Caputo. Para a função $f(x) = x$, o integrando é uma função positiva, e, conseqüentemente, quando algum dos números aleatórios $(x_1, x_2, \dots, x_{N_x})$ usados no cálculo dessa integral estiver muito próximo do polo, teremos um valor positivo muito grande para a soma definindo o MMC (4.5) frente ao valor de N_x .

Para observarmos melhor o comportamento da solução obtida pelo MMC para a derivada de Caputo quando $\alpha > 0.5$, na Figura 11 observamos a convergência do MMC para a derivada de Caputo de $f(x) = x$ com $\alpha = 0.8$ ao aumentar o número de iterações, para $i = 1000, 10000$ e 100000 , respectivamente. Novamente, utilizamos $L = 200$ e $h = \Delta x = \frac{b-a}{L}$. Nos gráficos desta figura, com o aumento do tamanho i da amostragem utilizada no cálculo da derivada de Caputo pelo MMC, além da redução do erro estatístico

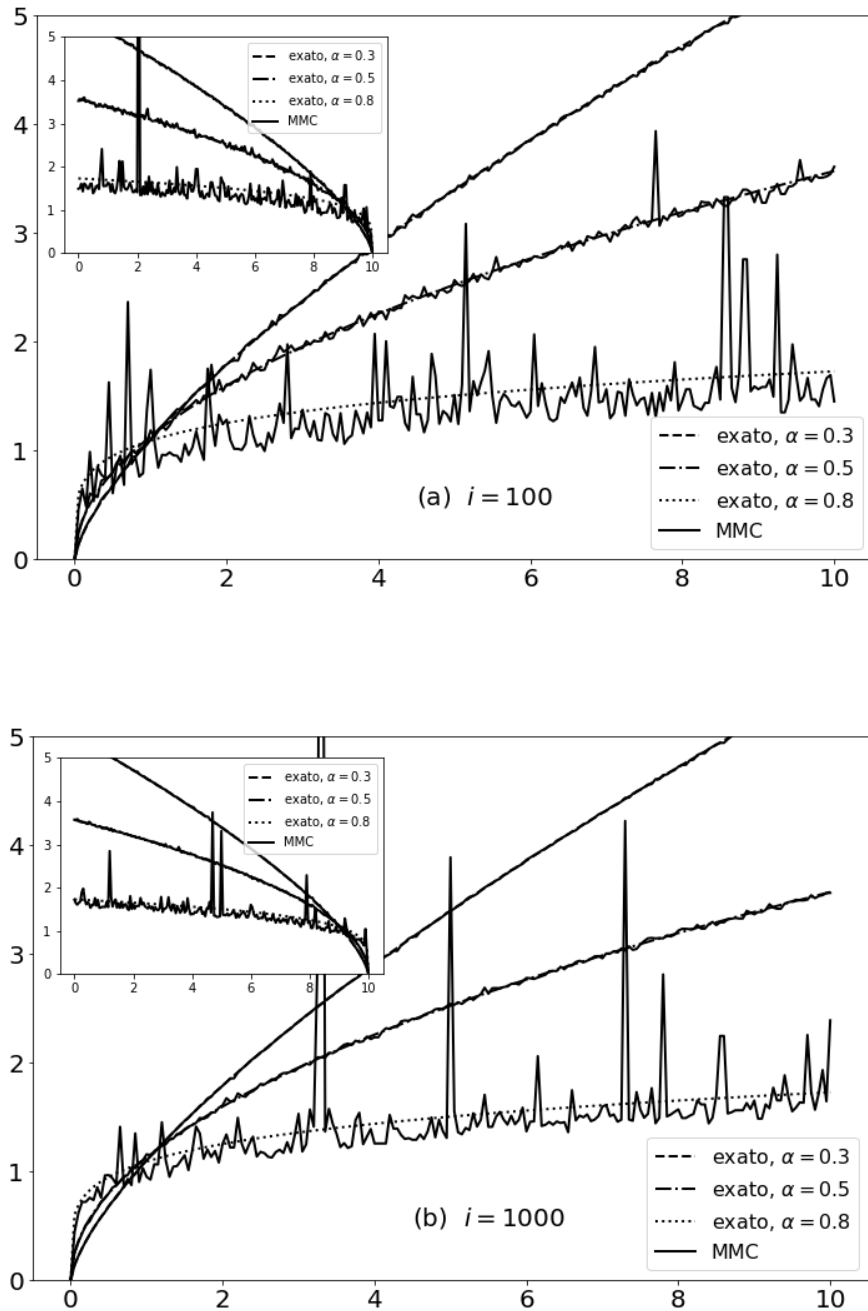


Figura 10: A convergência do MMC para derivadas de Caputo à esquerda com $\alpha = 0.3, 0.5, 0.8$ para $f(x) = x$. Em (a) temos $i = 100$, e em (b) temos $i = 1000$. Nos insertos dos gráficos (a) e (b) apresentamos a convergência para derivadas de Caputo à direita da função $f(x) = 10 - x$.

inerente ao MMC, observamos a redução da amplitude da subestimativa do valor da derivada devido aos polos nos integrandos em (2.40) e (2.41). Com o aumento de i , uma maior quantidade dos números aleatórios $(x_1, x_2, \dots, x_{N_x})$ usados no cálculo dessas integrais estará próxima do polo, contribuindo para o cálculo da integral e, consequentemente,

estimando melhor o valor da derivada.

Finalmente, na Figura 12 podemos ver que o MMC (4.5) e (4.6) também converge para funções não lineares. Nesta figura, apresentamos a derivada de Caputo à esquerda com ordem $\alpha = 0.4$ da função $f(x) = \text{sen}(x)$ (e a derivada de Caputo à direita da função $f(x) = \text{sen}(10 - x)$). A solução obtida pelo MMC é comparada com a solução obtida numericamente por quadratura. Nesta figura, usamos $L = 200$, $h = \Delta x = \frac{b-a}{L}$, e $i = 100$. Podemos verificar a partir do gráfico a concordância entre a solução obtida pelos dois métodos.

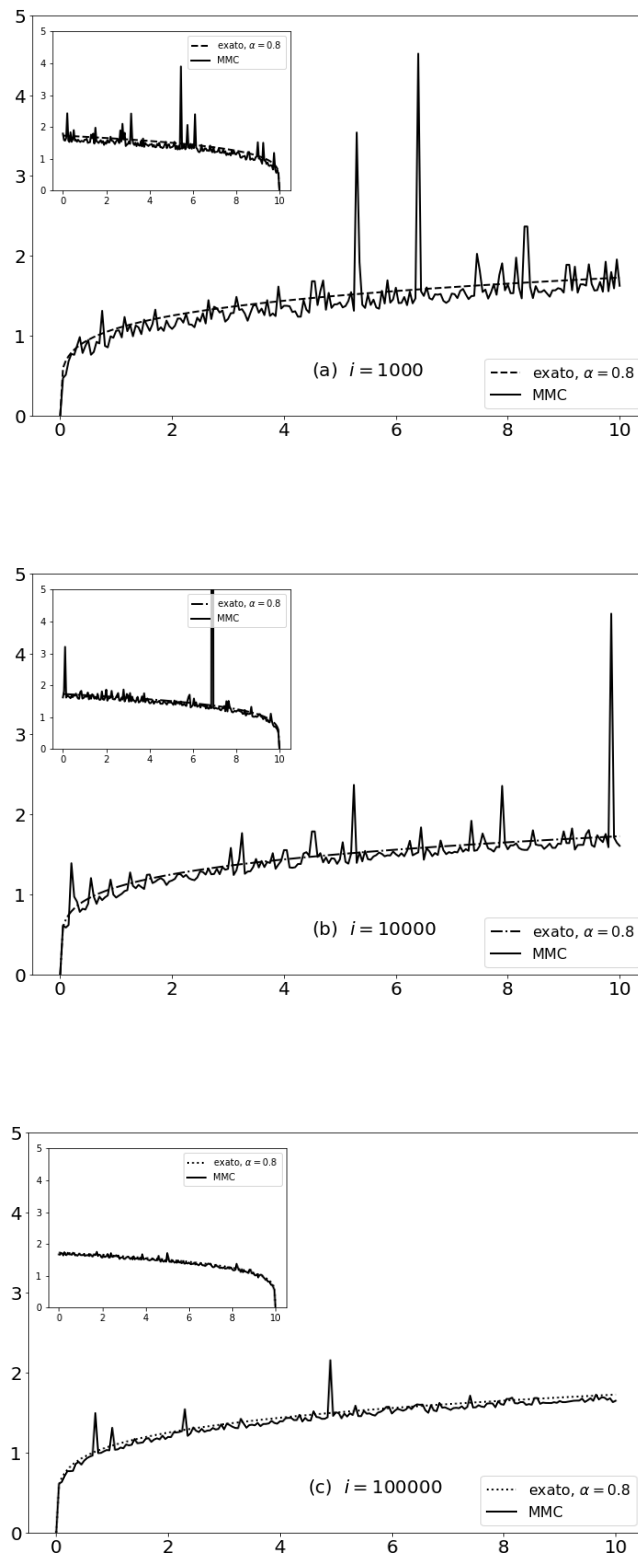


Figura 11: Convergência do MMC para a derivada de Caputo ordem $\alpha = 0.4$ para a função $f(x) = x$. Em (a), (b) e (c) temos $i = 1000, 10000$ e 100000 , respectivamente. Nos insertos dos gráficos (a), (b) e (c) apresentamos a convergência para derivadas de Caputo à direita da função $f(x) = 10 - x$.

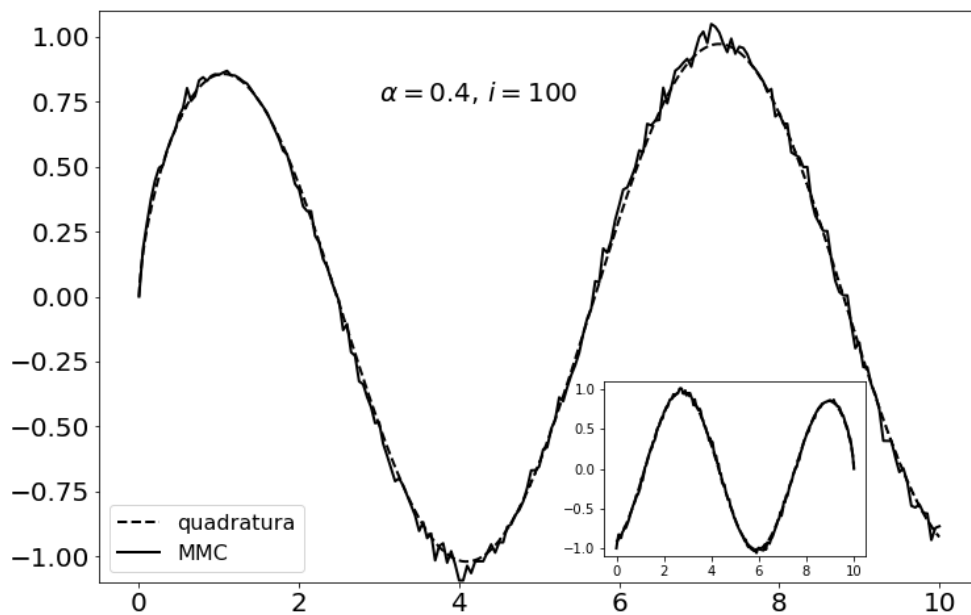


Figura 12: Convergência do MMC para derivadas de Caputo à esquerda com ordem $\alpha = 0.4$ para a função $f(x) = \sin(x)$. No inserto do gráfico, apresentamos a convergência para a derivada de Caputo à direita da função $f(x) = \sin(10 - x)$.

5 O MÉTODO DE MONTE CARLO PARA EQUAÇÕES DIFERENCIAIS ORDINÁRIAS FRACIONÁRIAS

Neste capítulo, iremos propor um MMC para resolver Problemas de Valor Inicial (PVI) para equações diferenciais ordinárias com derivadas fracionárias de Caputo e Riemann-Liouville. Daremos maior ênfase a problemas envolvendo derivadas de Caputo, por serem estas mais populares quanto às aplicações em ciência e engenharia. O método proposto pode ser facilmente estendido para outras formulações de derivadas fracionárias. Finalmente, alguns resultados parciais deste capítulo foram formalmente publicados no artigo [27].

5.1 Problema de valor inicial com Caputo

Para formular um MMC para equações diferenciais fracionárias ordinárias, vamos primeiro considerar problemas de valor inicial para derivadas fracionárias de Caputo. As equações diferenciais fracionárias de Caputo são mais populares na ciência aplicada por utilizarem condições iniciais usuais:

Problema 2 (Problema de valor inicial com Caputo). *O Problema de valor inicial com Caputo de ordem $0 < \alpha$ é dado pela equação diferencial fracionária*

$${}_a^C D_x^\alpha y = f(x, y), \quad (5.1)$$

com as seguintes condições iniciais

$$y^{(k)}(a) = y_a^{(k)} \quad (k = 0, 1, \dots, n-1), \quad (5.2)$$

onde $n = [\alpha] + 1$, $y^{(0)}(x) \equiv y(x)$, $e = y_0^{(k)}$ são constantes.

O primeiro passo para resolver o Problema 2 é integrar os dois lados de (5.1) com a integral fracionária de ordem α . Do Teorema Fundamental do Cálculo de Caputo, 2.3.2, temos:

$${}_a J_{x a}^{\alpha C} D_x^\alpha y(x) = y(x) - \sum_{k=0}^{n-1} \frac{y_a^{(k)}}{k!} (x-a)^k, \quad (5.3)$$

onde usamos as condições iniciais (5.2). Então, se $y(x) \in C^n$ no domínio, integrando ambos os lados de (5.1) com a integral fracionária RL (2.6) obtemos:

$$y(x) = \sum_{k=0}^{n-1} \frac{y_a^{(k)}}{k!} (x-a)^k + \frac{1}{\Gamma(\alpha)} \int_a^x (x-u)^{\alpha-1} f(u, y(u)) du. \quad (5.4)$$

O segundo e último passo para resolver o Problema 2 consiste no uso do MMC para obter a integral do lado direito de (5.4). No entanto, desde que $y(x)$ não é conhecida, não podemos aplicar o MMC direto na integral. Primeiramente, definimos uma função discretizada de $y(x)$. Para um inteiro positivo L , seja $x_n = x_0 + nh$ ($n = 0, 1, \dots, L$), onde $h = \frac{x_L - x_0}{L}$ e $x_0 = a$. Então, seja $y_n = y(x_n)$. A função discretizada é definida por:

$$y^L(x) = y_n \text{ se } x_n \leq x < x_{n+1}. \quad (5.5)$$

Desde que $y(x)$ é uma função diferenciável, temos $\lim_{L \rightarrow \infty} y^L(x) = y(x)$. Assim, a função discretizada $y^L(x)$ é uma aproximação para $y(x)$ quando L é grande. Podemos encontrar os valores y_n da seguinte relação de recorrência:

$$y_n = \sum_{k=0}^{n-1} \frac{y_a^{(k)}}{k!} (x_n - a)^k + \frac{1}{\Gamma(\alpha)} \int_a^{x_n} (x_n - u)^{\alpha-1} f(u, y^L(u)) du, \quad (n = 1, \dots, L). \quad (5.6)$$

Finalmente, calculando a integral em (5.6), usando um MCM (como em (3.14)), obtemos uma solução aproximada para o Problema 2:

$$y_n \approx \sum_{k=0}^{n-1} \frac{y_a^{(k)}}{k!} (x_n - a)^k + \frac{1}{\Gamma(\alpha)} \frac{x_n - a}{N} \sum_{i=1}^N (x_n - u_i)^{\alpha-1} f(u_i, y^L(u_i)), \quad (5.7)$$

onde N é um inteiro positivo, e $(u_1, \dots, u_N)$ são números aleatórios uniformemente dis-

tribuídos em $[a, x_n]$.

Observação 5.1.1. *É importante destacar que o MMC que propomos também pode ser usado para obter a solução de equações diferenciais ordinárias com derivadas de ordem inteira, pois a derivada fracionária de Caputo se reduz a derivada de ordem inteira quando α é um inteiro positivo.*

5.2 Problema de valor inicial com Riemann-Liouville

Consideremos agora os seguintes problemas de valor inicial com derivadas fracionárias de Riemann-Liouville:

Problema 3 (Problema de valor inicial com Riemann-Liouville). *O Problema de valor inicial com Riemann-Liouville de ordem $0 < \alpha$ é dado pela equação diferencial fracionária:*

$${}_a D_x^\alpha y = f(x, y), \quad (5.8)$$

com as seguintes condições iniciais:

$$\lim_{u \rightarrow a^+} \frac{d^{n-k-1}}{du^{n-k-1}} {}_a J_u^{n-\alpha} f(u) = b_a^{(k)} \quad (k = 0, 1, \dots, n-1), \quad (5.9)$$

onde $b_a^{(k)}$ são constantes.

Integrando ambos os lados de (5.8) com uma integral fracionária de Riemann-Liouville de ordem α , e pelo Teorema Fundamental do Cálculo de Riemann-Liouville (2.36) e a função discretizada (5.5), obtemos a solução aproximada por MMC para o Problema 3:

$$y_n \approx \sum_{k=0}^{n-1} \frac{b_a^{(k)}}{\Gamma(\alpha - k)} (x - a)^{\alpha-1-k} + \frac{1}{\Gamma(\alpha)} \frac{x_n - a}{N} \sum_{i=1}^N (x_n - u_i)^{\alpha-1} f(u_i, y^L(u_i)). \quad (5.10)$$

Observação 5.2.1. *Diferente do Problema 2 com a derivada fracionária de Caputo, o Problema de valor inicial com Riemann-Liouville requer condições iniciais não usuais (5.9). Apesar disso, quando α é um inteiro, a condição (5.9) reduz-se a usual e o Problema (3) reduz-se a um valor inicial de ordem inteira clássica.*

Observação 5.2.2. *É importante observar que a diferença no MMC para obtenção de solução de PVI para EDOs de Riemann-Liouville e Caputo é apenas a função potência em (5.10) e (5.7). O MMC na sua essência é utilizado para o cálculo dos somatórios que nessas expressões são idênticos. Portanto, do ponto de vista do MMC, não há diferença significativa se o PVI é de Riemann-Liouville ou de Caputo. Portanto, como os problemas de PVI com Caputo são os mais utilizados em aplicações, consideramos a seguir, sem perda de generalidade (para os propósitos do funcionamento do MMC), um número maior de exemplos para PVIs de Caputo.*

5.3 Resultados numéricos

Para ilustrar a utilização do MMC proposto neste capítulo para a resolução de PVIs fracionários, nesta seção comparamos a solução obtida pelo MMC com soluções exatas e com soluções obtidas pelo método de diferenças finitas (MDF) proposto em [73] para derivadas fracionárias. Por questões didáticas, para destacar o caráter estatístico do MMC, optou-se por um valor de N pequeno para que as divergências estatísticas entre a solução numérica e a solução analítica/MDF fiquem graficamente visíveis nas figuras. Consideramos 4 exemplos simples com derivadas fracionárias de Caputo e um com derivadas de Riemann-Liouville.

5.3.1 Exemplo 1: PVI de Caputo com $f(x, y) = x^\beta$

O primeiro exemplo é o Problema de Valor Inicial (PVI) mais simples possível:

$${}_a^C D_x^\alpha y = x^\beta, \quad y(0) = 0, \quad (5.11)$$

com $0 < \alpha \leq 1$, e onde $\beta > -\alpha$ é uma constante. A solução analítica do problema, que pode ser obtida por integração direta (usando a propriedade (2.51)), é dada por:

$$y(x) = \frac{\Gamma(\beta + 1)}{\Gamma(\alpha + \beta + 1)} x^{\alpha + \beta}. \quad (5.12)$$

Apesar de trivial, este problema é excelente para ilustrar a convergência estatística do

MMC proposto para a solução exata do problema. Na Figura 13, compara-se a solução exata (5.12) com a solução obtida pelo MMC proposto para diferentes valores de α e de $\beta = 0$. Nesta figura utilizou-se $L = 200$ e $i = 10$ ($N_x = in$, para $(n-1)\Delta x \leq x < n\Delta x$ onde $1 \leq n \leq 200$ é a n -ésima parte do domínio e $\Delta x = 0.01$) na equação (5.7) para o cálculo do valor aproximado de y_n . Na Figura 13 vemos uma boa concordância entre a solução exata e a solução pelo MMC para o caso usual de derivada inteira ($\alpha = 1$) e $\beta = 0$, mesmo para uma amostragem pequena ($i = 10$). Como discutido na seção anterior, o MMC proposto pode também ser aplicado para a resolução de equações diferenciais ordinárias de ordem inteira. As pequenas discrepâncias estatísticas (quase invisíveis visualmente nesta figura) entre a solução exata e a solução pelo MMC podem ser reduzidas aumentando-se o valor de i e/ou refinando a discretização da função (aumentando o valor de L). As soluções para os casos de derivadas fracionárias de ordem $\alpha = 0.75$ e $\alpha = 0.5$, com $\beta = 0$, também são apresentadas nesta figura. Nestes casos de derivadas fracionárias, existe também uma boa concordância entre a solução exata e a solução obtida pelo MMC, apesar de uma maior discrepância de natureza estatística entre a solução exata e a solução pelo MMC. Essa discrepância maior é devida ao polo da função $(x_n - u_i)^{\alpha-1}$ em (5.7). Quanto menor, α maior será a contribuição do polo. Portanto, o resultado para $\alpha = 0.5$ apresenta convergência pior que para $\alpha = 0.75$. As alternativas para se reduzir o erro estatístico são: 1) aumentar o valor de i (e também de L) ou 2) utilizar distribuições de probabilidades não uniformes [39, 24]. Optamos neste trabalho por utilizar um valor pequeno para i e L para facilitar a visualização das flutuações estatísticas na obtenção da solução. Utilizando valores grandes, visualmente não observaríamos a diferença entre a solução exata e a solução por MMC.

A convergência estatística da solução obtida pelo MMC quando aumentamos a amostragem i pode ser vista na Figura 14, onde consideramos $i = 100$ e $i = 1000$ para as derivadas de ordem $\alpha = 0.75$ e $\alpha = 0.5$. Novamente, utilizou-se $L = 200$ ($N_x = in$, para $(n-1)\Delta x \leq x < n\Delta x$ onde $1 \leq n \leq 200$ é a n -ésima parte do domínio e $\Delta x = 0.01$). Vemos, como esperado, que a convergência para $\alpha = 0.75$ é maior do que para $\alpha = 0.5$, como discutido no parágrafo anterior. A diferença entre a solução obtida pelo MMC para

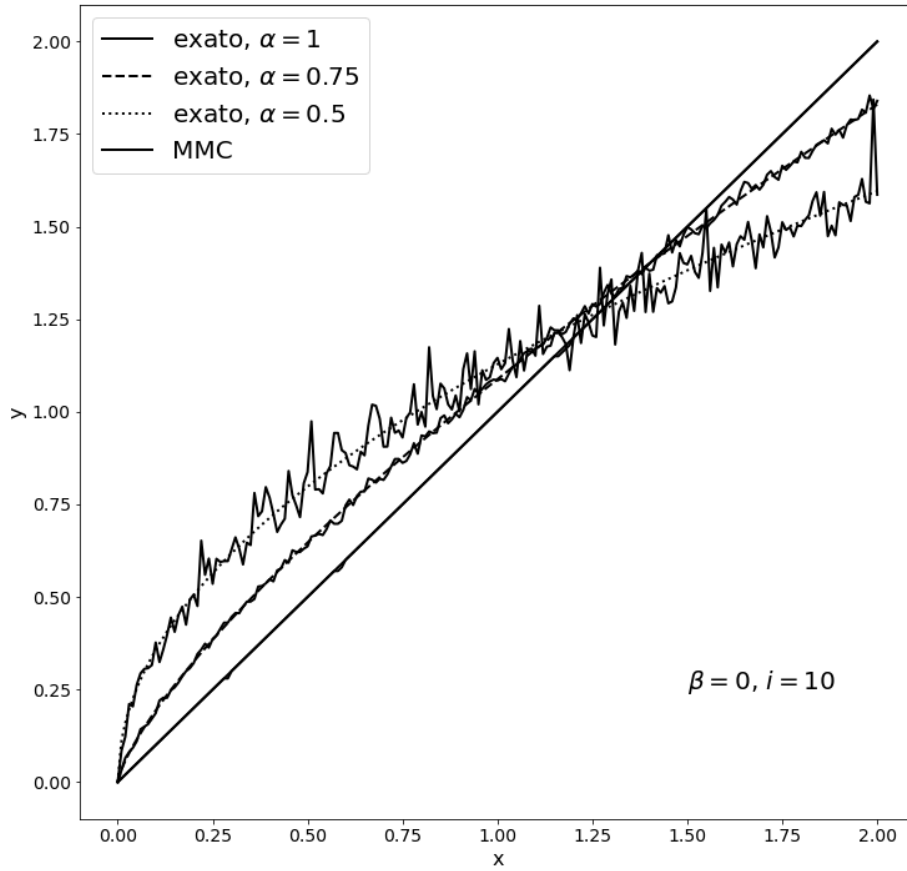


Figura 13: Comparação da solução obtida pelo MMC com a solução exata para $\alpha = 1, 0.75, 0.5$ e $\beta = 0$.

$\alpha = 0.75$, que é bem visível na Figura 13 para $i = 10$, praticamente desaparece visualmente para $i = 100$ (gráfico (a) da Figura 14). Para $i = 1000$ (gráfico (b)), o resultado é ainda melhor. Para o problema 5.11, quando $\alpha = 0.5$ o integrando em (5.7) deixa de ser L^2 no domínio, o que resulta em uma convergência muito mais lenta do que para $\alpha = 1$ e $\alpha = 0.75$. Comparando os resultados apresentados nas Figuras 13 e 14 (gráficos (a) e (b)) vemos que a discrepância entre a solução obtida pelo MMC em comparação à solução exata é bem maior para $\alpha = 0.5$ do que nos outros casos. Mesmo assim, com o aumento da amostragem i , observamos que o MMC converge estatisticamente para a solução exata mesmo que mais lentamente.

Na Figura 15, apresentam-se os resultados do PVI fracionário 5.11 para $\beta = -0.25, 0.5, 2$

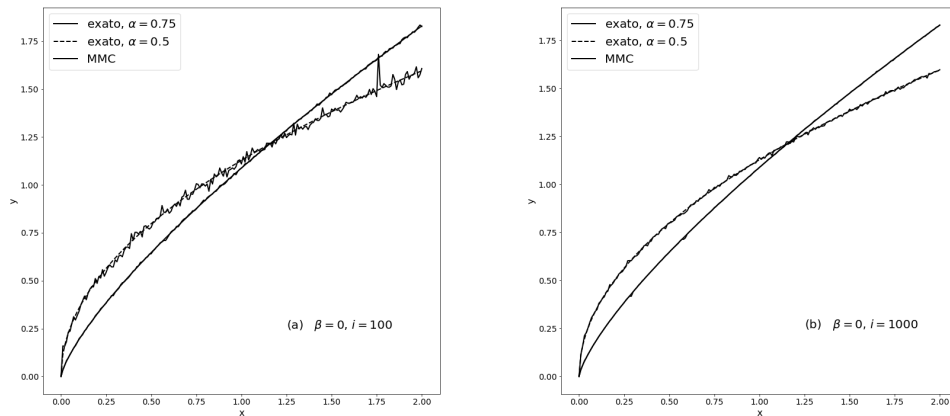


Figura 14: Convergência da solução obtida pelo MMC para $\alpha = 0.75$ e $\alpha = 0.5$ e $\beta = 0$. Na Figura (a) temos $i = 100$ e em (b) temos $i = 1000$.

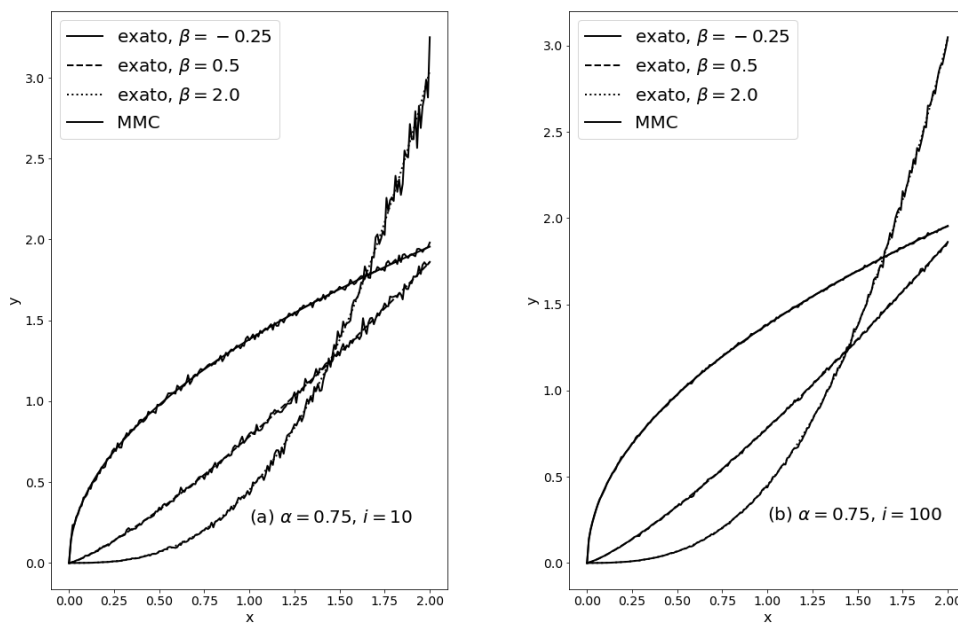


Figura 15: Comparação da solução obtida pelo MMC com a solução exata para $\alpha = 0.75$ e $\beta = -0.25, 0.5, 2$. Em (a) temos $i = 10$ e em (b) temos $i = 100$.

com derivada fracionária $\alpha = 0.75$. Observamos novamente, para todas as potências β , uma boa concordância entre a solução exata e a solução obtida pelo MMC. A convergência estatística do MMC proposto é evidente quando aumentamos a amostragem de $i = 10$ (gráfico (a)) para $i = 100$ (gráfico (b)).

5.3.2 Exemplo 2: PVI de Caputo com $f(x, y) = y$

O segundo exemplo é um caso particular da generalização fracionária do problema de crescimento exponencial. O PVI fracionário que consideramos é:

$${}_a^C D_x^\alpha y = y, \quad y(0) = 1, \quad (5.13)$$

com $0 < \alpha \leq 1$. A solução analítica do problema, que pode ser obtida usando a transformada de Laplace, é dada por [69]:

$$y(x) = \sum_{k=0}^{\infty} \frac{(x)^{\alpha k}}{\Gamma(\alpha k + 1)} = E_\alpha(x^\alpha), \quad (5.14)$$

onde $E_\alpha(x)$ é chamada de Função de Mittag-Leffler de um parâmetro [59].

Na Figura 16 apresentamos a solução do PVI de Caputo (5.13) com $\alpha = 1$ (derivada de ordem inteira), $\alpha = 0.75$, $\alpha = 0.5$ e condição inicial $y_0 = 1$. Nos gráficos da figura, a solução exata e a solução obtida pelo MMC são comparadas. Em todas essas figuras, observa-se uma boa concordância entre a solução exata e a solução pelo MMC. Novamente, observa-se ainda que, quanto menor a ordem da derivada (menor o valor de α), pior a convergência estatística do método. Para obter as soluções apresentadas nesses gráficos, o domínio foi dividido em $L = 500$ partes iguais, e foi feita uma amostragem de $i = 10$ (gráficos à esquerda) e $i = 100$ (gráficos à direita) pontos aleatórios em cada uma dessas partes. Esses valores foram escolhidos apenas para ilustrar o funcionamento do método. Como discutido antes, a convergência estatística da solução de Monte Carlo para a solução exata pode ser melhorada, reduzindo o erro estatístico em relação à solução exata, aumentando-se o número de amostragem (ou de forma alternativa, utilizando-se distribuições não uniformes), como pode ser visto nos gráficos (b), (d) e (f) da Figura 16, onde aumentamos o tamanho da amostragem de $i = 10$ (gráficos (a), (c) e (e)) para $i = 100$.

Finalmente, podemos ainda observar na Figura 16, com um pouco de atenção, que a solução obtida pelo MMC proposto em (5.7) subestima ligeiramente a solução do PVI de Caputo (5.13) em relação à solução exata. Este fato não está relacionado à con-

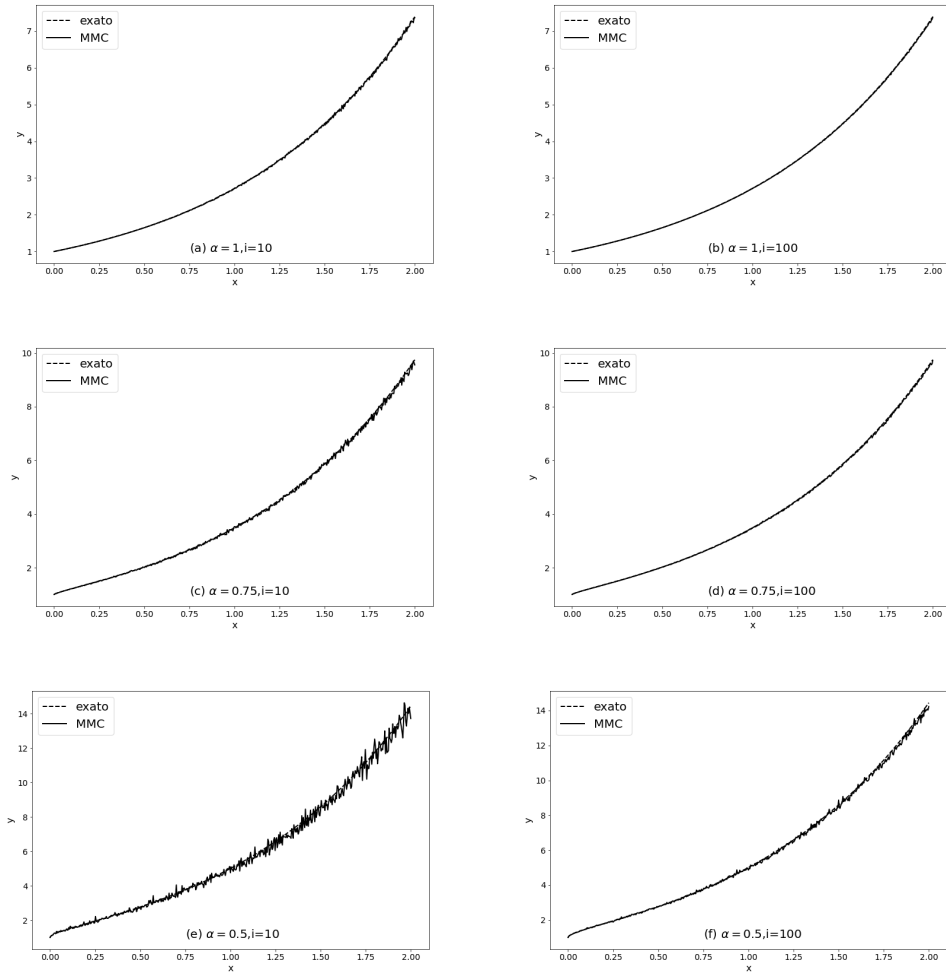


Figura 16: A convergência da solução do MMC para o PVI (5.13), e com derivadas de ordens $\alpha = 1, 0.75$ e 0.5 , respectivamente. Em (a), (c), e (e), temos $i = 10$. Em (b), (d), e (f), temos $i = 100$.

vergência estatística do MMC, mas ao erro numérico devido à discretização da função y . Aumentando-se o valor de L , essa diferença (subestimação) é reduzida. Observamos melhor esta subestimação na Figura 17, onde consideramos $L = 50, 100, 250$. Nos gráficos desta figura, para manter um mesmo tamanho de amostragem de números aleatórios em todo o domínio (temos iL números aleatórios em cada amostragem no domínio $[0, 2]$), fizemos respectivamente $i = 1000, 500, 200$. O caso $L = 500$ com $i = 100$ pode ainda ser visto no gráfico (d) da Figura 16. Vemos que a subestimação do MMC proposto em (5.7) em relação à solução exata reduz-se monotonamente com o aumento de L . Para $L = 50$, a subestimação é bem visível no gráfico (a) da Figura 17, reduzindo-se para $L = 100$ no gráfico (b). Para $L = 250$, a subestimação começa a ser difícil de observar visualmente

na escala apresentada do gráfico (c).

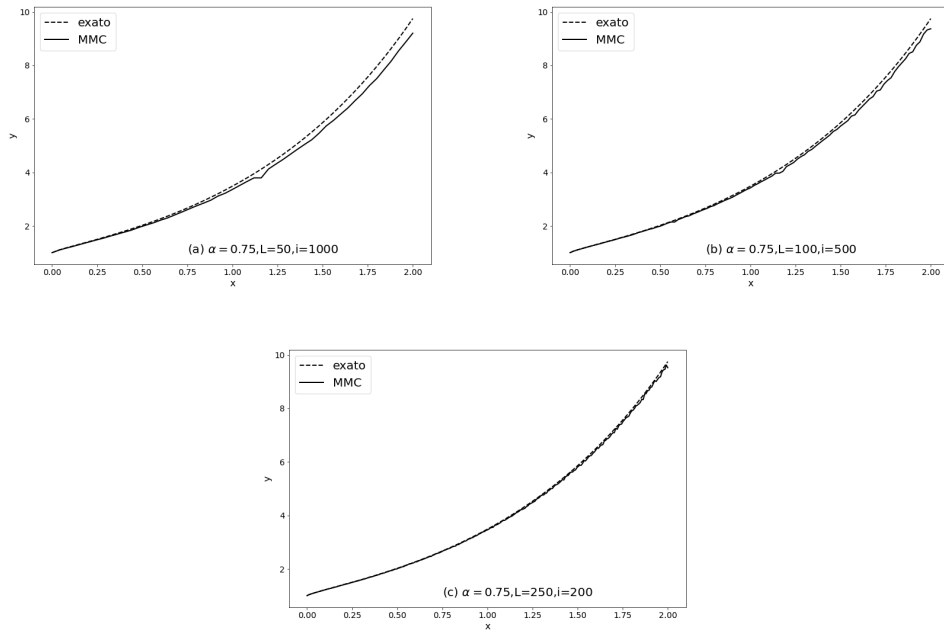


Figura 17: A convergência da solução do MMC para o PVI (5.13) com derivadas de ordens $\alpha = 0.75$, para $L = 50, 100, 250$ e com $i = 1000, 500, 200$, respectivamente.

5.3.3 Exemplo 3: PVI de Caputo com $f(x, y) = -y$

Neste terceiro exemplo verificamos que o MMC (5.7) também pode ser utilizado para PVI de ordem maior. Primeiro consideraremos o PVI:

$${}_a^C D_x^\alpha y = -y, \quad y(0) = 1, \quad y'(0) = 0, \quad (5.15)$$

com $1 < \alpha \leq 2$. A solução analítica do problema, que também pode ser obtida usando a transformada de Laplace, é dada por [69]:

$$y(x) = \sum_{k=0}^{\infty} \frac{(-x^\alpha)^k}{\Gamma(\alpha k + 1)} = E_\alpha(-x^\alpha), \quad (5.16)$$

onde $E_\alpha(x)$ é a Função de Mittag-Leffler de um parâmetro [59].

Na Figura 18 apresentamos a solução obtida pelo MMC (5.7) para $\alpha = 2, \alpha = 1.75, \alpha = 1.5$ e $\alpha = 1.25$, respectivamente. Nestes gráficos, o domínio foi dividido em $L = 500$ partes iguais, e foi feita uma amostragem de $i = 100$ pontos aleatórios em cada uma

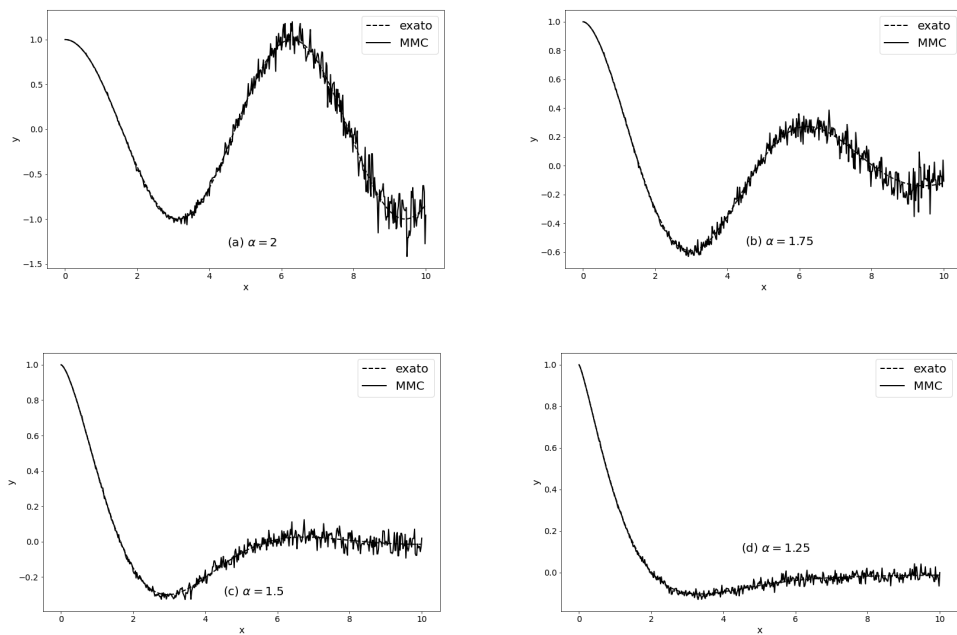


Figura 18: Convergência da solução do MMC para o PVI (5.15) com derivadas de ordens $\alpha = 2, 1.75, 1.5$ e 1.25 , respectivamente. Em (a), (b), (c), e (d), temos $L = 500$ e $i = 100$.

dessas partes. Esses valores foram escolhidos apenas para ilustrar o funcionamento do método, o valor baixo de i foi escolhido para manter evidente o caráter estatístico do MMC. Novamente, observamos uma boa concordância entre a solução pelo MMC e a solução exata para todas as ordens observadas, incluindo o caso de derivada de ordem inteira $\alpha = 2$ no gráfico (a).

Finalmente, para evidenciar a convergência do MMC para a solução exata, apresentamos na Figura 19 a solução do MMC para $\alpha = 1.75$, e com $i = 1000$ e $i = 10000$.

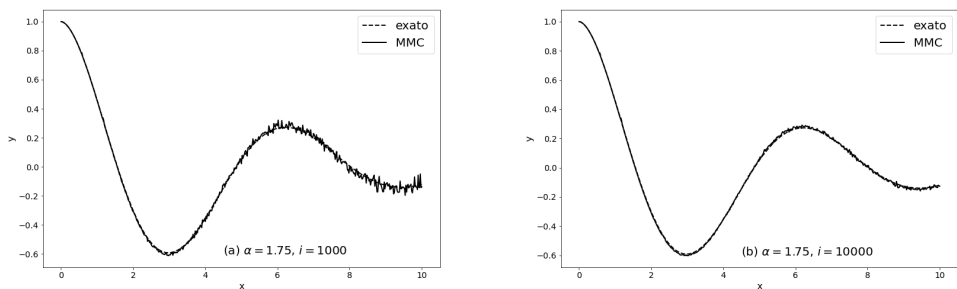


Figura 19: Convergência da solução do MMC para o PVI (5.15) com derivada de ordem $\alpha = 1.75$ e $L = 500$. Em (a) temos $i = 1000$, e em (b) temos $i = 10000$.

5.3.4 Exemplo 4: PVI de Caputo com ordem grande e $f(x, y) = y$

Finalmente, vamos verificar agora que o MMC proposto pode ser útil não apenas para EDOs fracionárias, mas também para EDOs de ordem inteira quando a ordem da derivada é muito grande (neste caso, o método de Runge-Kutta é trabalhoso de ser implementado). Consideramos o seguinte PVI:

$${}_a^C D_x^\alpha y = y, \quad y(0) = 1, \quad y^{(j)}(0) = 0, \quad (j = 1, 2, \dots, 20), \quad (5.17)$$

com $20 < \alpha \leq 21$. A solução analítica do problema, que também pode ser obtida usando a transformada de Laplace, é dada por:

$$y(x) = \sum_{k=0}^{\infty} \frac{(x^\alpha)^k}{\Gamma(\alpha k + 1)} = E_\alpha(x^\alpha). \quad (5.18)$$

onde $E_\alpha(x)$ é novamente a Função de Mittag-Leffler de um parâmetro [59].

Nas Figuras 20, 21 e 22 apresentamos a solução obtida pelo MMC (5.7) para $\alpha = 21$, $\alpha = 20.75$ e $\alpha = 20.5$, respectivamente. Nestes gráficos utilizamos novamente $L = 500$ e $i = 100$. Novamente observamos uma boa concordância entre a solução pelo MMC e a solução exata, mostrando a utilidade do MMC (5.7) que propomos para obter a solução de PVI, fracionárias ou não, de ordem superior.

5.3.5 Exemplo 5: PVI de Riemann-Liouville com $f(x, y) = y$

Como exemplo final, vamos mostrar que o MMC proposto em (5.10) pode ser utilizado para resolver PVI de Riemann-Liouville. Consideramos o seguinte PVI:

$${}_0 D_x^\alpha y = y, \quad \lim_{u \rightarrow 0^+} {}_a J_u^{1-\alpha} y(u) = 1, \quad (5.19)$$

com $0 < \alpha \leq 1$.

A solução exata do PVI (5.19) pode ser obtida, por exemplo, por transformada de

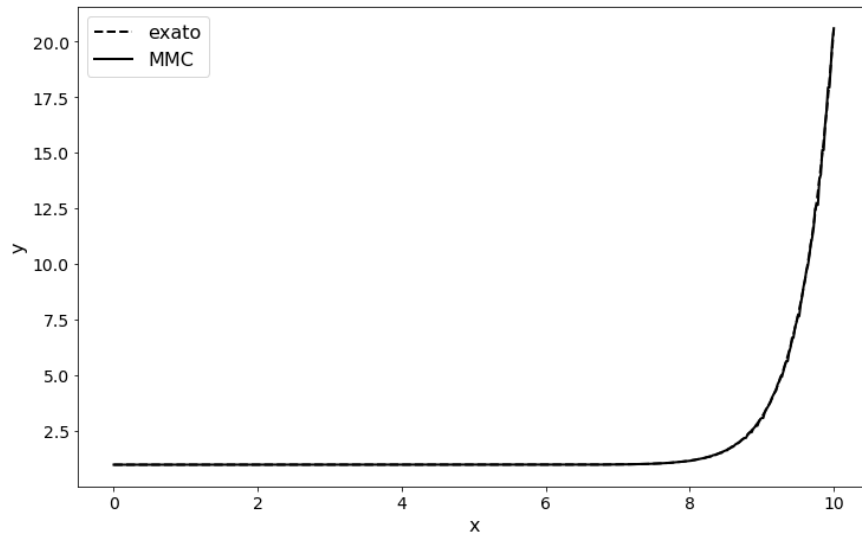


Figura 20: Comparação da solução exata com a obtida pelo MMC para $\alpha = 21$ da equação 5.17.

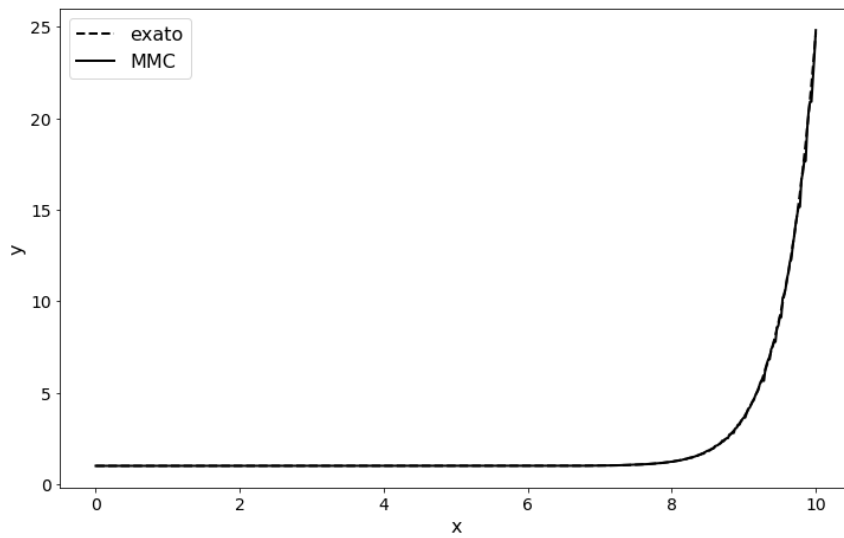


Figura 21: Comparação da solução exata com a obtida pelo MMC para $\alpha = 20.75$ da equação 5.17.

Laplace ou por Frobenius. A solução é [69]:

$$y(x) = x^{\alpha-1} \sum_{k=0}^{\infty} \frac{(x)^{\alpha k}}{\Gamma(\alpha k + \alpha)} = x^{\alpha-1} E_{\alpha,\alpha}(x^{\alpha}), \quad (5.20)$$

onde $E_{\alpha,\alpha}(\cdot)$ é a função de Mittag-Leffler de dois parâmetros [69].

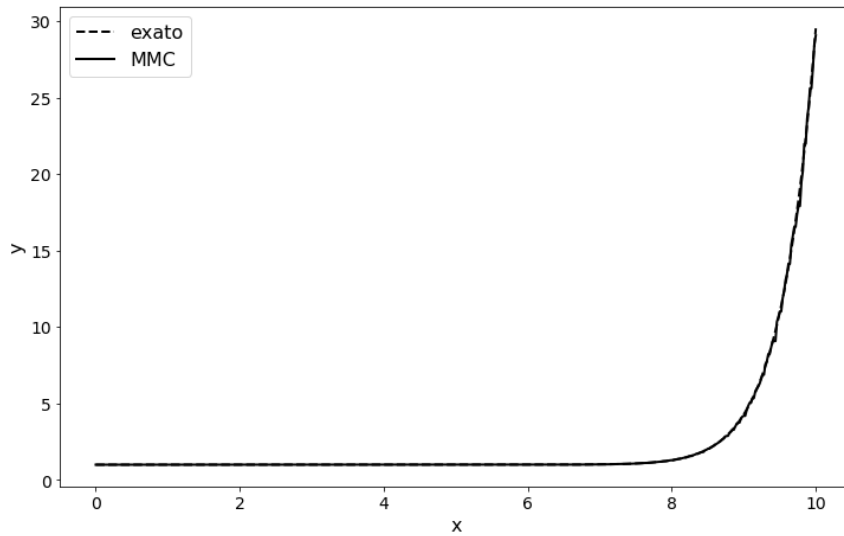


Figura 22: Comparação da solução exata com a obtida pelo MMC para $\alpha = 20.5$ da equação 5.17.

Na Figura 23 apresentamos a solução obtida pelo MMC (5.10) para o PVI (5.19) com $\alpha = 0.75$. Nestes gráficos, usamos $L = 1000$, $i = 10$, e $i = 100$. A solução pelo MMC é comparada com a solução exata (5.20), mostrando visualmente uma boa concordância. Pode ser observado que a solução numérica subestima a solução quando comparamos com o resultado exato. Este fato não é devido à convergência estatística do MMC, mas sim à discretização da função y . Aumentando o valor de L , essa diferença é reduzida.

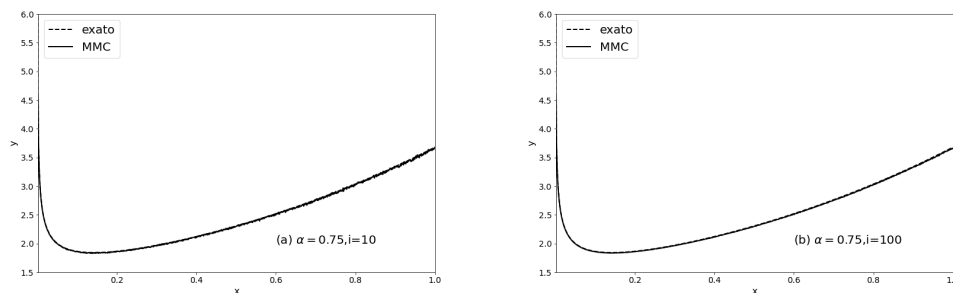


Figura 23: Convergência da solução do MMC para o PVI (5.19) com derivada de ordem $\alpha = 0.75$. Em (a) temos $i = 10$, e em (b) temos $i = 100$.

6 O MÉTODO DE MONTE CARLO PARA SISTEMAS DINÂMICOS FRACIONÁRIOS

Neste capítulo, mostraremos que o MMC pode ser utilizado para resolver sistemas de equações diferenciais fracionárias. O algoritmo proposto aqui resultou na publicação do artigo [28]. Como exemplos de aplicação, resolvemos o modelo epidemiológico SIR fracionário e o sistema Lotka-Volterra fracionário. Para analisar a convergência estatística do MMC proposto, é feita uma comparação dos resultados encontrados com a solução obtida pela variante do Método de Diferenças Finitas (MDF) utilizada em [73].

6.1 Modelo SIR fracionário

O modelo SIR e suas diversas generalizações são muito utilizados na modelagem de sistemas que visam entender a dinâmica de uma doença em uma população. Como exemplo recente de aplicação, podemos citar a modelagem da pandemia da COVID-19 para explicar a disseminação e definir estratégias de vacinação conforme temos em [11] e em [73].

O modelo SIR foi proposto no trabalho seminal de Kermack e McKendrick publicado em 1927 [40]. O modelo introduzido por eles descreve a disseminação de doenças infecciosas em uma população dividida em subgrupos de indivíduos, chamados de suscetíveis (com fração $0 \leq S \leq 1$), infectados (com fração $0 \leq I \leq 1$) e recuperados (com fração $0 \leq R \leq 1$). Cada um desses subgrupos tem sua dinâmica descrita por uma EDO. No trabalho [73], de onde utilizamos o MDF para comparação com o MMC, analisou-se o modelo fracionário SIR tendo duas populações interagindo. Neste capítulo vamos analisar

o caso de apenas uma população. O modelo que analisaremos é dado por:

$$\begin{cases} {}^C_0 D^\alpha S(t) = -\beta SI - \mu(S - 1) \\ {}^C_0 D^\alpha I(t) = \beta SI - (\mu + \gamma)I \\ {}^C_0 D^\alpha R(t) = \gamma I - \mu R \end{cases}, \quad (6.1)$$

onde $0 < \alpha \leq 1$ é a ordem da derivada, β é a taxa de transmissão fracionária da doença (relacionada à taxa média de contato dentro da população), γ é o inverso fracionário de período infeccioso médio. Além disso, as taxas de mortalidade fracionárias μ são consideradas iguais às taxas de natalidade, de modo que o total de N da população é constante durante a doença. Portanto, tem-se $S(t) + I(t) + R(t) = 1$. Além disso, em relação à dimensão dos parâmetros no modelo, é importante notar que, como S , I e R são adimensionais, e o operador de derivada fracionária ${}^C_0 D_t^\alpha$ tem dimensão de tempo $^{-\alpha}$ (como a derivada de ordem inteira $\frac{d}{dt}$ tem dimensão de tempo $^{-1}$), as taxas fracionárias β , γ e μ devem ter dimensão de tempo $^{-\alpha}$. Para uma discussão sobre a questão dimensional da modelagem fracionária, recomendam-se os artigos [31, 45] e o livro [30].

Os pontos de equilíbrio $P(S^*, I^*, R^*)$ do sistema (6.1), podem ser obtidos fazendo ${}^C_0 D^\alpha S(t) = 0$, ${}^C_0 D^\alpha I(t) = 0$, e considerando $R = 1 - S - I$. Eles são dados por (veja [73]):

- i) $P_0(1, 0, 0)$ que é o ponto de equilíbrio livre da infecção;
- ii) $P_1\left(\frac{\mu + \gamma}{\beta}, \frac{\mu(\mu + \gamma - \beta)}{\beta(\mu + \gamma)}, \frac{2\beta\mu + \beta\gamma - \gamma\mu - \gamma^2}{\beta(\mu + \gamma)}\right)$ que é o ponto de equilíbrio endêmico.

Observação 6.1.1. *Os pontos de equilíbrio do modelo SIR fracionário são idênticos aos pontos de equilíbrio do modelo SIR usual, pois a derivada de Caputo de uma constante é idênticamente zero.*

6.1.1 O MMC para o modelo SIR fracionário

Nesta seção, um MMC é proposto para obter a solução numérica do sistema (6.1) considerando as derivadas de Caputo. O primeiro passo consiste no uso do Teorema Funda-

mental do Cálculo de Caputo (2.3.2). Assumindo que $S(t)$, $R(t)$ e $I(t)$ são diferenciáveis no domínio, e integrando os dois lados do sistema (6.1) com uma integral fracionária de Riemann-Liouville (2.6) de ordem $0 < \alpha \leq 1$, temos:

$$\begin{cases} S(t) = S(0) - \beta_0 J_t^\alpha SI - \mu_0 J_t^\alpha (S - 1) \\ I(t) = I(0) + \beta_0 J_t^\alpha SI - (\mu + \gamma)_0 J_t^\alpha I \\ R(t) = R(0) + \gamma_0 J_t^\alpha I - \mu_0 J_t^\alpha R. \end{cases} \quad (6.2)$$

O segundo passo consiste no uso do MMC para obter a solução das integrais em (6.2). No entanto, como $S(t)$, $R(t)$ e $I(t)$ não são conhecidas, não se pode aplicar o MMC diretamente para o cálculo dessas integrais. Então, define-se primeiro uma discretização destas funções, como fizemos no capítulo anterior para o PVI. Para um inteiro positivo L , temos $t_n = t_0 + nh$ ($n = 0, 1, \dots, L$), onde $h = \frac{t_L - t_0}{L}$ e $t_0 = 0$. Seja então $X^{(n)} = X(t_n)$ (onde X representa cada uma das funções S, I e R), a discretização das funções é definida por:

$$X^L(t) = X^{(n)} \text{ se } t_n \leq t < t_{n+1}. \quad (6.3)$$

Como foi assumido que $X(t)$ é diferenciável, temos $\lim_{L \rightarrow \infty} X^L(t) = X(t)$. Portanto, a função discretizada $X^L(t)$ é uma boa aproximação da função $X(t)$ para L suficientemente grande. Pode-se então encontrar os valores $X(t_n)$ a partir da seguinte relação de recorrência:

$$\begin{cases} S^{(n)} = S(0) - \beta \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{S^L I^L}{(t_n - t)^{1-\alpha}} dt - \mu \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{S^L - 1}{(t_n - t)^{1-\alpha}} dt \\ I^{(n)} = I(0) + \beta \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{S^L I^L}{(t_n - t)^{1-\alpha}} dt - (\mu + \gamma) \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{I^L}{(t_n - t)^{1-\alpha}} dt \\ R^{(n)} = R(0) + \gamma \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{I^L}{(t_n - t)^{1-\alpha}} dt - \mu \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{R^L}{(t_n - t)^{1-\alpha}} dt. \end{cases} \quad (6.4)$$

Finalmente, calculando as integrais em (6.4) usando o MMC (como feito na (3.14)),

obtem-se a soluão aproximada do sistema SIR fracionário (6.1):

$$\begin{cases} S^{(n)} = S(0) - \frac{\beta t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} S^L(t_j^*) I^L(t_j^*) - \frac{\mu t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} (S^L(t_j^*) - 1) \\ I^{(n)} = I(0) + \frac{\beta t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} S^L(t_j^*) I^L(t_j^*) - \frac{(\mu + \gamma) t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} I^L(t_j^*) \\ R^{(n)} = R(0) + \frac{\gamma t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} I^L(t_j^*) - \frac{\mu t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} R^L(t_j^*), \end{cases} \quad (6.5)$$

onde N é um inteiro positivo e $(t_1^*, \dots, t_N^*)$ são números aleatórios uniformemente distribuídos em $[0, t_n]$. Finalmente, é importante ressaltar que o MMC proposto pode ser utilizado também para obter a soluão do sistema SIR com derivada inteira fazendo-se $\alpha = 1$ em (6.5).

6.1.2 Resultados numéricos

Neste subcapítulo, apresentamos a soluão obtida pelo MMC (6.5) para o modelo SIR fracionário com derivadas de Caputo de ordem $\alpha = 0.75$. Para os parâmetros definindo o modelo, foram considerados os seguintes valores: $\beta = 0.7$, $\gamma = 0.1$ e $\mu = 0$. Além disso, escolhemos uma população total e constante de $N_p = 100000$ indivíduos. Para obter a soluão do problema, consideramos ainda a seguinte condião inicial: $S(0) = 1 - \frac{1}{N_p}$, $I(0) = \frac{1}{N_p}$ e $R(0) = 0$, ou seja, inicialmente havia apenas um indivíduo doente. Finalmente, como forma de comparar a soluão encontrada usando o MMC, foi também obtida a soluão do sistema (6.5) pelo MDF proposto em [73].

Na Figura 24 apresentamos a soluão do sistema (6.1) obtida pelo MMC e pelo MDF. Nestas soluões, foi utilizado um passo de tempo de $h = t_n - t_{n-1} = 0.5$ dias. Para a soluão pelo MMC, escolhemos um tamanho de amostragem $N = in = 5n$ pequeno (ou seja $i = 5$ números aleatórios em cada subintervalo de tempo) como forma de mostrar as flutuaões estatísticas que o MMC gera.

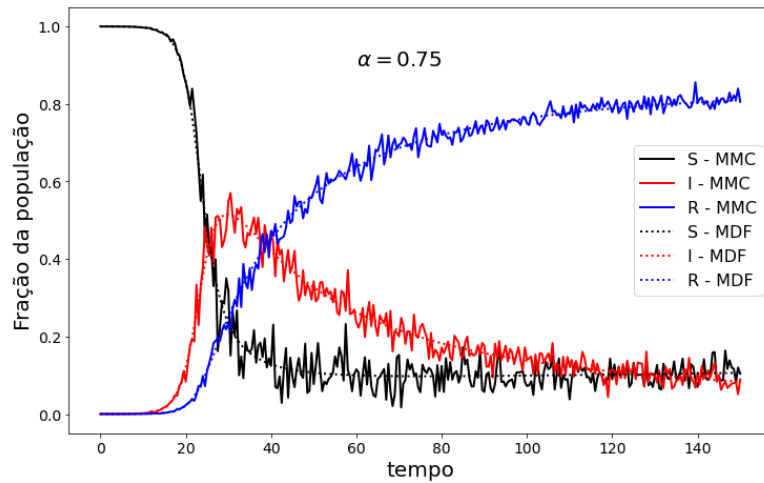


Figura 24: Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 5$.

Para verificar como ocorre a convergência estatística da solução, na Figura 25 dobrou-se o valor de N . Percebe-se neste caso uma melhor convergência, apesar de uma pequena mudança em N , quando comparamos com a solução obtida anteriormente.

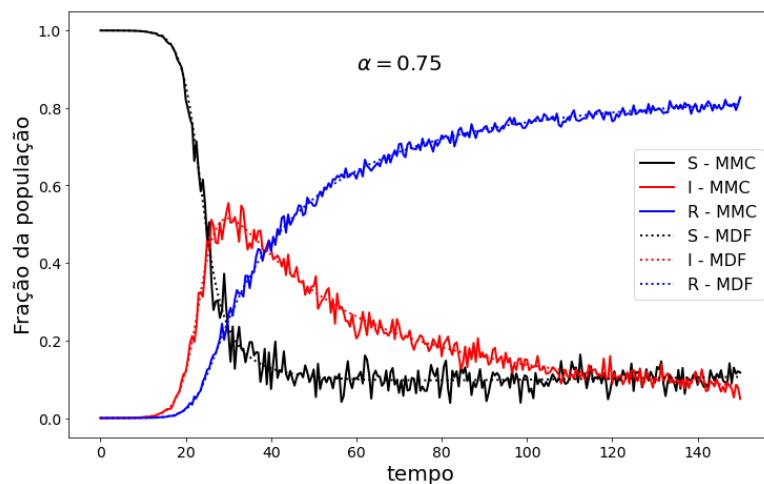


Figura 25: Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 10$.

Utilizando $N = 100n$ e $N = 1000n$, nas Figuras 26 e 27 obtemos soluções por MMC bem mais próximas das obtidas por MDF. Estes resultados ilustram como ocorre a

convergência estatística da solução do sistema pelo aumento do valor do N . Isto se deve, conforme vimos em (3.19), à redução do erro padrão.

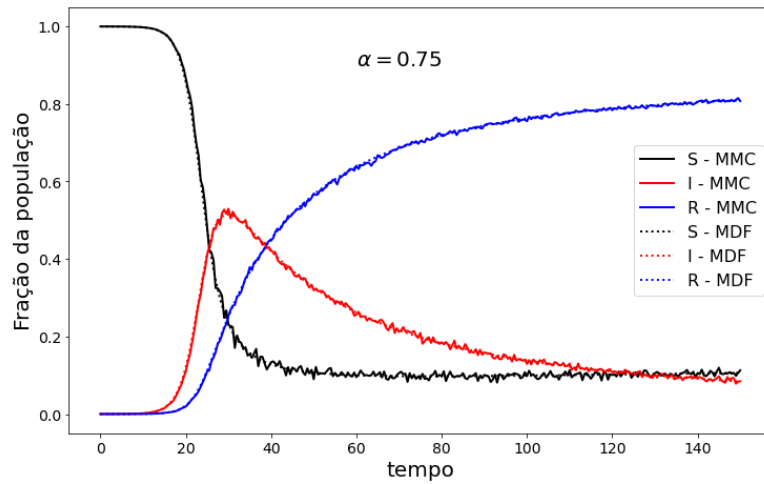


Figura 26: Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 100$.

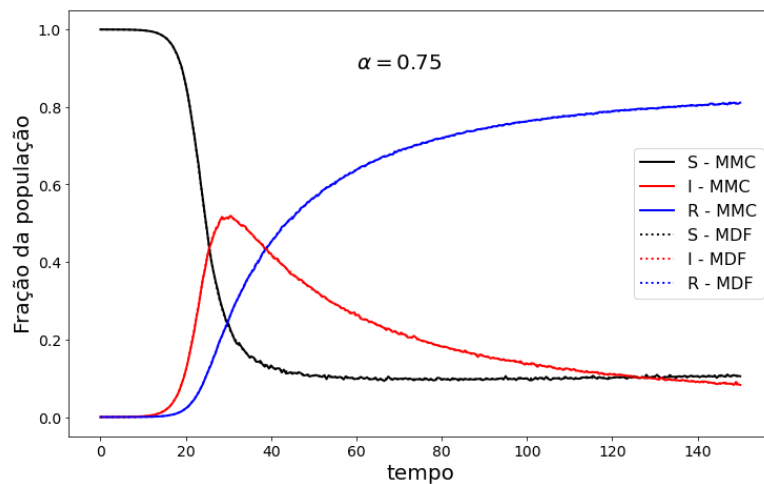


Figura 27: Comparação da solução do sistema SIR-fracionário por MDF e por MMC com $i = 1000$.

6.2 Sistema Lotka-Volterra

O sistema Lotka-Volterra, também conhecido como sistema predador-presa, foi proposto em 1925 de forma independente pelo matemático Volterra e pelo biofísico Lotka [4]. Este modelo descreve como evolui temporalmente a população de predadores e presas em um dado ecossistema. A versão com derivadas fracionárias do sistema Lotka-Volterra é dada por:

$$\begin{cases} {}^C_0 D_t^\alpha x = ax - bxy \\ {}^C_0 D_t^\beta y = -cy + dxy \end{cases}, \quad (6.6)$$

onde $x = x(t)$ representa o número de presas, $y = y(t)$ representa o número de predadores, a é a taxa de crescimento de presas sem predadores, b é a taxa de decrescimento de presas devido à predação, c é taxa de mortalidade da população de predadores sem presas e d é a taxa de crescimento de predadores devido à predação, como pode ser vista em [18, 19]. Além disso, consideramos derivadas de Caputo de ordens $0 < \alpha, \beta \leq 1$.

A solução para o problema (6.6) pelo MMC pode ser obtida de forma análoga ao do modelo SIR. Seguindo os mesmos passos utilizados para obter-se a equação (6.5), temos para o MMC para o sistema Lotka-Volterra (6.6):

$$\begin{cases} x^{(n)} = x(0) + \frac{t_n}{\Gamma(\alpha)N} \sum_{j=1}^N (t_n - t_j^*)^{\alpha-1} \left(ax^L(t_j^*) - bx^L(t_j^*)y^L(t_j^*) \right) \\ y^{(n)} = y(0) + \frac{t_n}{\Gamma(\beta)N} \sum_{j=1}^N (t_n - t_j^*)^{\beta-1} \left(-cy^L(t_j^*) + dy^L(t_j^*)x^L(t_j^*) \right) \end{cases}, \quad (6.7)$$

onde usamos a discretização (6.3) das funções $x(t)$ e $y(t)$, N é um inteiro positivo e $(t_1^*, \dots, t_N^*)$ são números aleatórios uniformemente distribuídos em $[0, t_n]$.

Na Figura 28 apresentamos a solução do sistema (6.6) com os seguintes parâmetros: $a, b, c, d = 1$ e $\alpha, \beta = 0.8$, $L = 200$. Nos gráficos (a), (b), (c) e (d), temos as soluções para $i = 10, 100, 1000$ e 10000 , respectivamente. Nesta Figura também apresentamos a solução do sistema Lotka-Volterra fracionário pelo MDF, para efeito de comparação. Se tivermos $\alpha, \beta = 1$, teremos a solução clássica do sistema Lotka-Volterra em cada caso.

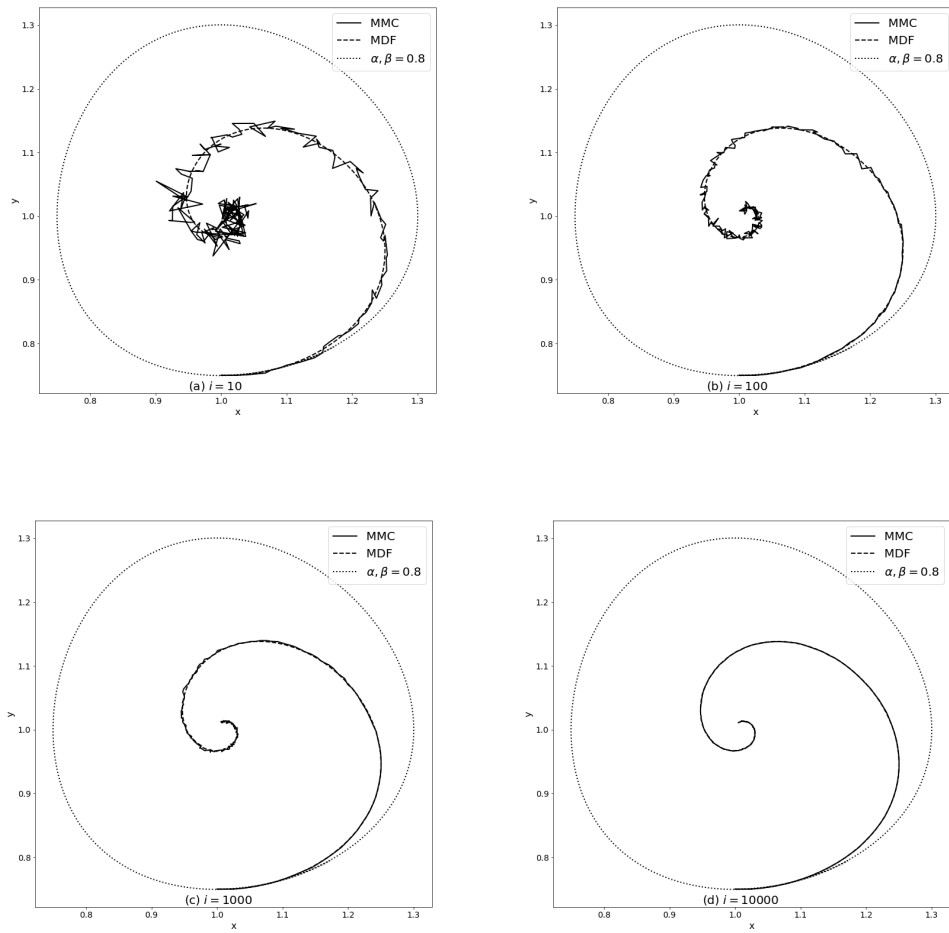


Figura 28: Solução do sistema (6.6) com os parâmetros, $a, b, c, d = 1$, $\alpha, \beta = 0.8$ e as condições iniciais dadas por $x_0 = 1$ e $y_0 = 0.75$. Em (a), (b), (c) e (d), temos $i = 10$, $i = 100$, $i = 1000$ e $i = 10000$, respectivamente.

7 MÉTODO DE MONTE CARLO VARIACIONAL: PROBLEMAS DE FRONTEIRA PARA EQUAÇÕES DIFERENCIAIS FRACIONÁRIAS E EXTREMANTES DO CÁLCULO VARIACIONAL FRACIONÁRIO

Neste último capítulo, formulamos algoritmos estatísticos para resolver problemas de fronteira para equações diferenciais fracionárias, e também para obter as soluções diretas de problemas do cálculo variacional fracionário. Os algoritmos estatísticos que propomos são baseados no método de Monte Carlo Variacional (MMCV) [53, 48]. O MMCV, também conhecido como método de Monte Carlo Quântico, é historicamente utilizado na Física para obter-se o estado fundamental de sistemas quânticos através da minimização de um funcional. Enquanto o MMC baseia-se na construção de uma sequência de números cuja média converge estatisticamente para o resultado exato de um dado problema matemático, o MMCV baseia-se na construção de uma sequência de funções que converge estatisticamente (através do uso do chamado algoritmo de Metropolis) para a solução exata de um problema.

7.1 Problemas de fronteira para equações diferenciais fracionárias

Nesta seção, propomos um algoritmo baseado no MMCV para obter a solução aproximada de problemas de fronteira para equações diferenciais fracionárias. Por simplicidade, consideraremos apenas equações diferenciais envolvendo derivadas de Caputo. No entanto, é importante salientar que o algoritmo proposto pode ser facilmente adaptado para outras formulações de derivadas fracionárias. Consideramos problemas de fronteira do tipo:

Problema 4. *Sejam $1 < \alpha \leq 2$ e $y \in C^2[a, b]$. O problema de fronteira do Cálculo Fracionário de Caputo que consideramos consiste em obter a solução do problema:*

$$\begin{cases} {}^C D_x^\alpha y(x) = f(x, y(x)) \text{ com } 1 < \alpha \leq 2 \text{ e } a < x < b \\ \text{e a condição de fronteira } y(a) = y_a, \ y(b) = y_b. \end{cases} \quad (7.1)$$

Nesta seção propomos dois algoritmos do MMCV para resolver problemas de fronteira do tipo (7.1).

7.1.1 Primeiro algoritmo

Antes de formular um algoritmo estatístico baseado no MMCV, observe que podemos reescrever a equação diferencial em (7.1) como um funcional:

$${}_a^C D_x^\alpha y(x) - f(x, y(x)) = 0 \implies J[y] = \int_a^b ({}_a^C D_x^\alpha y(x) - f(x, y(x)))^2 dx = 0, \quad (7.2)$$

sujeito às condições de fronteira $y(a) = y_a$ e $y(b) = y_b$. Note que o funcional $J[y]$ tem valor mínimo igual a zero quando $y(x)$ é a solução do problema de fronteira (7.1).

O algoritmo estatístico do MMCV que propomos é então construído a partir da função discretizada $y^L(x) = y_l$, para $x_l \leq x < x_{l+1}$, definida na equação (5.5) do Capítulo 5. Nesta função L é um inteiro positivo e $y_l = y(x_l)$, onde $x_l = a + lh$ ($l = 0, 1, \dots, L$) e $h = \frac{b-a}{L}$. Como a função discretizada deve satisfazer a condição de fronteira, temos que $y_0 = y_a$ e $y_L = y_b$. Os passos subsequentes do algoritmo do MMCV são:

1. Inicialização: escolhe-se uma função inicial y^L e calcula-se o valor do funcional $J[y^L]$, definido em (7.2), para esta função.
2. Iterando:
 - (a) Gerando uma nova função z^L candidata à solução do problema (7.1): sorteia-se aleatoriamente um valor inteiro i ($i = 1, \dots, L-1$) seguindo-se uma distribuição uniforme. Dado um número real positivo δ , sorteia-se então um número real $\Delta y \in [-\delta, \delta]$, seguindo uma distribuição uniforme, e define-se o novo valor de $y_i \rightarrow y_i + \Delta y$. Com o novo valor de y_i , temos uma nova função discretizada $z^L(x) = y_l$, para $x_l \leq x < x_{l+1}$, que difere da anterior apenas para $x_i \leq x < x_{i+1}$.
 - (b) Verificando se a nova função será aceita (algoritmo de Metrópoles): com a nova função z^L construída no passo (a), calcula-se o novo valor do funcional $J[z^L]$. Se $J[z^L] < J[y^L]$, aceita-se a mudança atualizando o valor de y^L fazendo $y^L = z^L$. Atualiza-se também o valor de $J[y^L]$ fazendo $J[y^L] = J[z^L]$. Por outro lado, se $J[z^L] \geq J[y^L]$, rejeita-se a nova função z^L e mantêm-se os valores anteriores de y^L e $J[y^L]$.
 - (c) Retornando: com os valores finais de y^L e $J[y^L]$ obtidos após o passo (b), retorna-se ao passo (a) um número n (número de iterações) de vezes.

Esse algoritmo iterativo tem como objetivo ir transformando/deformando uma função inicial aleatoriamente ao modificar aleatoriamente o valor da função discretizada em apenas um ponto y_i . Se a transformação resulta em uma nova função que reduz o valor do

funcional, então esta nova função é aceita. Caso contrário, a nova função é rejeitada. Ao repetir o procedimento n vezes, teremos uma sequência de novas funções aceitas. Como o integrando que define o funcional é convexo, essa sequência converge estatisticamente para a solução do problema de valor de fronteira (7.1).

Finalmente, para implementar numericamente o algoritmo proposto, precisamos escolher um método numérico para calcular a derivada fracionária e para calcular a integral definindo o funcional $J[y^L]$ definido em (7.2). Para o cálculo da integral definindo o funcional $J[y^L]$ usamos o método do trapézio dividindo o intervalo $[a, b]$ em L partes iguais de comprimento h . Por outro lado, da Definição 2.3.1 das derivadas de Caputo temos, para $1 < \alpha \leq 2$,

$${}_a^C D_x^\alpha y(x) = \frac{1}{\Gamma(2-\alpha)} \int_a^x \frac{y''(u)}{(x-u)^{\alpha-1}} du, \quad (7.3)$$

onde y'' representa a derivada de segunda ordem da função y . Aproximando a função y pela função discretizada y^L , e aproximando a derivada de segunda ordem y'' por diferenças finitas, ou seja, $y''^L = y_l''$, para $x_l \leq x < x_{l+1}$, onde

$$y_l'' = \begin{cases} \frac{y_0 - 2y_1 + y_2}{h^2} & \text{se } l = 0, \\ \frac{y_{l-1} - 2y_l + y_{l+1}}{h^2} & \text{se } 1 \leq l \leq L-1, \\ \frac{y_{L-2} - 2y_{L-1} + y_L}{h^2} & \text{se } l = L, \end{cases} \quad (7.4)$$

temos:

$$\begin{aligned} {}_a^C D_{a+lh}^\alpha y^L(a+lh) &= \frac{1}{\Gamma(2-\alpha)} \int_a^{a+lh} \frac{y''^L(u)}{(a+lh-u)^{\alpha-1}} du \\ &= \frac{1}{\Gamma(2-\alpha)} \sum_{j=0}^{l-1} y_j'' \int_{a+jh}^{a+(j+1)h} \frac{1}{(a+lh-u)^{\alpha-1}} du \\ &= \frac{h^{2-\alpha}}{\Gamma(3-\alpha)} \sum_{j=0}^{l-1} y_j'' [(l-j)^{2-\alpha} - (l-j-1)^{2-\alpha}], \end{aligned} \quad (7.5)$$

onde usamos $x = a + lh$.

7.1.2 Segundo algoritmo

O segundo algoritmo difere do primeiro apenas pelo funcional. Aplicando a integral de Riemann-Liouville na equação diferencial em (7.1), e usando o Teorema Fundamental do Cálculo de Caputo 2.3.2 obtemos para $1 < \alpha \leq 2$:

$${}_a J_x^\alpha ({}_a^C D_x^\alpha y(x) - f(x, y(x))) = y(x) - y(a) - y'(a)(x-a) - {}_a J_x^\alpha f(x, y(x)) = 0. \quad (7.6)$$

Portanto, podemos definir o funcional convexo:

$$J[y] = \int_a^b \left(y(x) - y(a) - y'(a)(x - a) - {}_a J_x^\alpha f(x, y(x)) \right)^2 dx, \quad (7.7)$$

sujeito às condições de fronteira $y(a) = y_a$ e $y(b) = y_b$, e que tem valor mínimo igual a zero quando $y(x)$ é a solução do problema de fronteira (7.1).

Dado o funcional (7.7), o algoritmo do MMCV que propomos segue os mesmos passos 1 e 2 do primeiro algoritmo. Para o cálculo do funcional (7.7), utilizamos novamente o método do trapézio dividindo o intervalo $[a, b]$ em L partes iguais de comprimento h . Também usamos novamente a função discretizada $y^L(x) = y_l$, para $x_l \leq x < x_{l+1}$, definida na equação (5.5) do Capítulo 5. Com a função discretizada, temos $y(a) = y_0$ e $y'(a) = y'_0$ onde aproximamos $y'_0 = \frac{y_1 - y_0}{h}$. Para a integral Riemann-Liouville no integrando do funcional (7.7) temos a aproximação, usando a função discretizada,

$$\begin{aligned} {}_a J_{a+lh}^\alpha y^L(a + lh) &= \frac{1}{\Gamma(\alpha)} \int_a^{a+lh} \frac{y^L(u)}{(a + lh - u)^{1-\alpha}} du \\ &= \frac{1}{\Gamma(\alpha)} \sum_{j=0}^{l-1} y_j \int_{a+jh}^{a+(j+1)h} (a + lh - u)^{\alpha-1} du \\ &= \frac{h^\alpha}{\Gamma(\alpha + 1)} \sum_{j=0}^{l-1} y_j [(l - j)^\alpha - (l - j - 1)^\alpha], \end{aligned} \quad (7.8)$$

onde usamos $x = a + lh$.

Finalmente, o segundo algoritmo tem a vantagem de convergir estatisticamente mais rápido que o primeiro algoritmo, e com um erro numérico menor devido à presença da aproximação da derivada segunda no primeiro algoritmo. No entanto, como não temos derivadas da função y no integrando definindo o funcional (7.7) (com exceção da derivada primeira em $x = a$), não é possível garantir que o segundo algoritmo irá convergir para soluções contínuas e diferenciáveis do problema de fronteira (7.1). O algoritmo pode acabar convergindo para um extremante descontínuo do funcional (7.7).

7.1.3 Exemplo 1

Para ilustrar o funcionamento dos algoritmos 1 e 2, vamos considerar neste exemplo o problema de fronteira dado por:

$$\begin{cases} {}^C D_x^\alpha y(x) = y(x) + 1 & \text{com } 1 < \alpha \leq 2 \text{ e } 0 < x < 2 \\ \text{e a condição de fronteira } y(0) = 0, \quad y(2) = E_\alpha(2^\alpha) - 1. \end{cases} \quad (7.9)$$

A solução analítica do problema pode ser obtida usando a transformada de Laplace ou também pelo método de Frobenius, e é dada por:

$$y(x) = \sum_{k=0}^{\infty} \frac{(x^\alpha)^k}{\Gamma(\alpha k + 1)} - 1 = E_\alpha(x^\alpha) - 1, \quad (7.10)$$

onde $E_\alpha(x)$ é a Função de Mittag-Leffler de um parâmetro [59].

Na Figura 29 vemos a convergência da solução obtida pelo primeiro algoritmo para o problema de fronteira (7.9) com derivada de Caputo de ordem $\alpha = 1.5$. Comparamos nesta figura as soluções obtidas com o primeiro algoritmo, para $n = 200000, 1000000$ e 5000000 passos de iteração, com a solução exata. Para os resultados desta figura utilizamos $L = 50$ (o domínio $[0,2]$ foi dividido em 50 partes iguais, e a função discretizada tem 51 pontos). O valor utilizado para Δy_i em cada iteração foi um número aleatório no intervalo $[-0.02, 0.02]$, ou seja, usamos $\delta = 0.02$. Nesta figura vemos que para $\delta = 0.02$ são necessários $n = 5000000$ passos de iteração para o primeiro algoritmo convergir para a solução exata. Aumentar ainda mais o número de passos n não resulta em uma solução mais próxima da exata, pois o erro da solução obtida pelo primeiro algoritmo (graficamente quase imperceptíveis na Figura 29) resulta majoritariamente da discretização da função $y(x)$, que acarreta no erro numérico do cálculo da derivada de segunda ordem e da derivada de Caputo, além do erro no cálculo da integral definindo o funcional (7.6). Para obter uma solução mais próxima da exata é necessário aumentar o número L de intervalos na discretização da função $y(x)$. Optamos por usar um número $L = 50$ (pequeno) para facilitar a visualização da solução numérica no gráfico da Figura 29.

O efeito da ordem da derivada no erro numérico do cálculo da derivada de segunda ordem e da derivada de Caputo, além do erro no cálculo da integral definindo o funcional (7.6) diminui quando α se aproxima de 1 e aumenta quando α se aproxima de 2, como pode ser observado na Figura 30. Para $\alpha = 1.25$ (gráfico (a)), a solução obtida pelo primeiro algoritmo apresenta melhor concordância com a solução exata do que a solução para $\alpha = 1.5$ apresentada na Figura 29. Já para $\alpha = 1.75$, o erro numérico devido à discretização da solução torna-se inaceitável, como pode ser visto no gráfico (b). O aumento do erro numérico com o aumento da ordem da derivada fracionária, devido à discretização da função, é esperado, pois quanto mais α está próximo de 2, maior será o efeito do polo no integrando da definição da derivada de Caputo (7.3), amplificando o erro numérico na aproximação da derivada de segunda ordem da função discretizada e aumentando também o erro numérico na integral definindo o funcional $J[y]$. Nesta figura utilizamos novamente $L = 50$ e $\delta = 0.02$.

O segundo algoritmo converge muito mais rápido para a solução exata, e com um erro numérico muito menor devido à discretização, do que o primeiro algoritmo. Como pode ser visto na Figura 31 para $\alpha = 1.75$ (no gráfico (a)) e $\alpha = 1.95$ (no gráfico (b)). Para $L = 50$ foram necessárias apenas $n = 200000$ iterações (25 vezes menos iterações) para obter

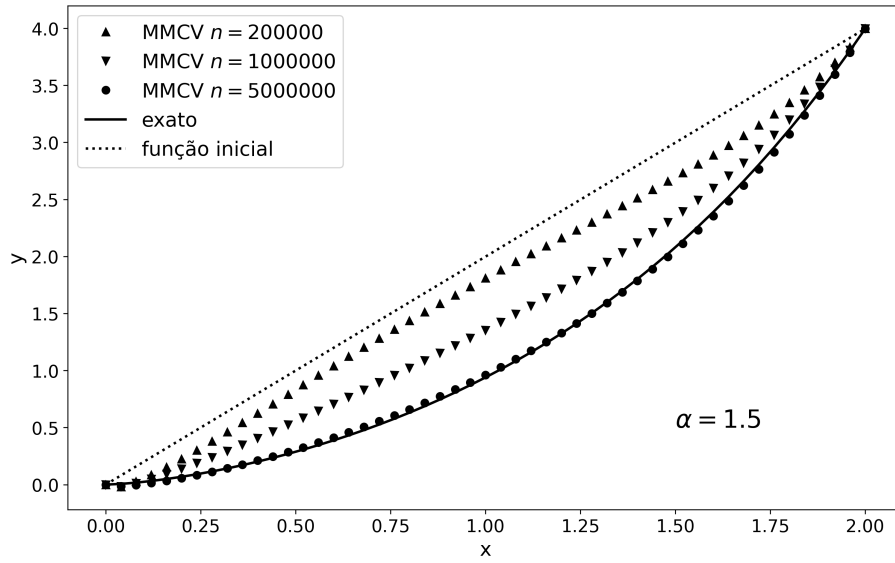


Figura 29: Convergência do MMCV para o primeiro algoritmo com $\alpha = 1.5$ e 200000, 1000000 e 5000000 passos.

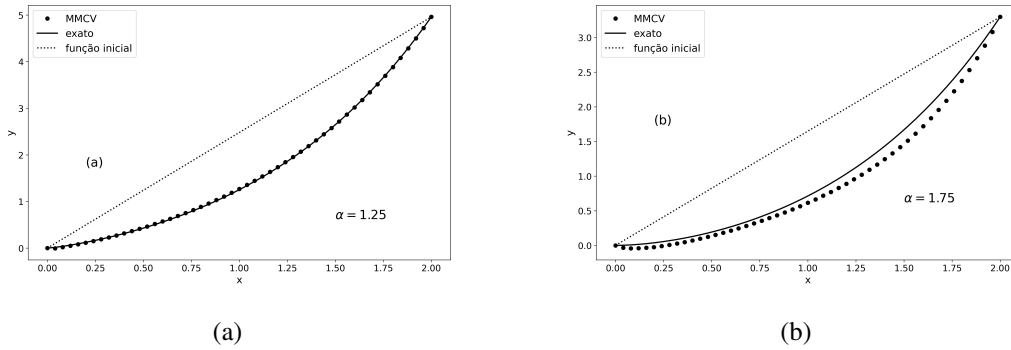


Figura 30: Convergência do MMCV para o primeiro algoritmo. Em (a) temos $\alpha = 1.25$ e 5000000 passos, e em (b) temos $\alpha = 1.75$ e 5000000 passos.

uma boa concordância com a solução analítica. A redução no erro numérico da solução se deve à ausência da necessidade de calcular a derivada numérica (7.4), eliminando, portanto, esta fonte de erro, e pela ausência de polos no integrando definindo a integral de Riemann-Liouville (7.8). Esta ausência de polos tem como consequência a redução do erro no cálculo numérico da integral definindo o funcional $J[y]$ (7.7). No entanto, veremos no próximo exemplo que o segundo algoritmo pode convergir para extremantes não contínuos do funcional (7.7), portanto, pode convergir para uma função que não é a solução desejada do problema de fronteira.

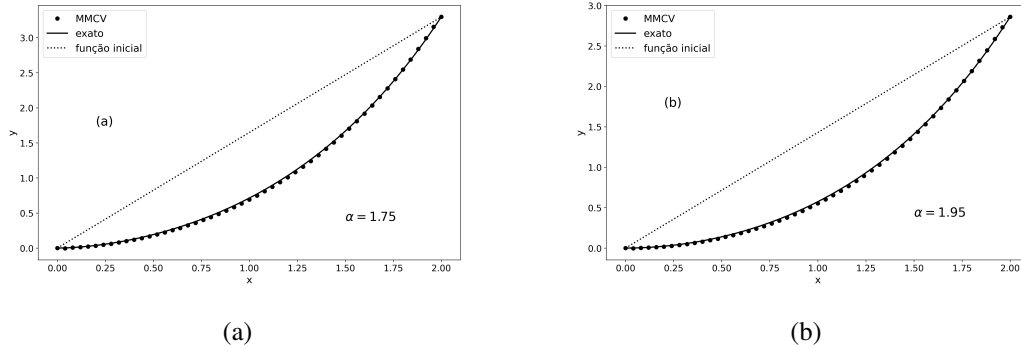


Figura 31: Convergência do MMCV para o segundo algoritmo. Em (a) temos $\alpha = 1.75$ e $n = 200000$ passos, e em (b) temos $\alpha = 1.95$ e $n = 200000$ passos.

7.1.4 Exemplo 2

Neste segundo exemplo, vamos considerar um problema de fronteira não linear:

$$\begin{cases} {}^C D_x^\alpha y(x) = -xy(x)^2 + \frac{1}{2} & \text{com } 1 < \alpha \leq 2 \text{ e } 0 < x < 2 \\ \text{e a condição de fronteira } y(0) = 0, y(2) = 0.6938494002146437. \end{cases} \quad (7.11)$$

Como este problema não tem solução exata, compararemos as soluções obtidas pelos primeiros e segundo algoritmos com a solução obtida numericamente pelo método das diferenças finitas (DF) apresentado em [73].

Na Figura 32 vemos a comparação da solução obtida pelo primeiro algoritmo com a solução obtida pelo método de DF. Consideramos neste exemplo derivadas de Caputo de ordem $\alpha = 1.5$. Para a solução pelo primeiro algoritmo utilizamos $L = 80$ (o domínio $[0, 2]$ foi dividido em 80 partes iguais, e a função discretizada tem 81 pontos). O valor utilizado para Δy_i em cada iteração foi um número aleatório no intervalo $[-0.001, 0.001]$, ou seja, usamos $\delta = 0.001$. Para a solução pelo método de DF [73], utilizamos um passo $\Delta x = h/10 = 1/400$. A convergência neste problema de fronteira não linear foi mais lenta, em comparação ao exemplo anterior, pois foi necessário aumentar o valor de L (reduzindo o erro da discretização da função) e reduzir o valor de δ para diminuir a rejeição à nova função z^L definida no passo (b) do primeiro algoritmo. Como consequência do menor valor de δ , foi necessário aumentar o número de iterações n para 25000000.

Apesar da convergência mais lenta, vemos na Figura 32 que a solução obtida pelo primeiro algoritmo converge bem para a solução obtida pelo método da DF. O mesmo não é verdade quando utilizamos o segundo algoritmo, como pode ser visto na Figura 33. O segundo algoritmo pode convergir para um extremante do funcional (7.7) que é descontínuo. No caso deste exemplo, o algoritmo convergiu para uma função descontínua em $x = 2$. O valor obtido para o funcional para esta função descontínua foi de $J[y^L] = 7.839002350429781 \times 10^{-6}$. O que significa que o valor do funcional é tão

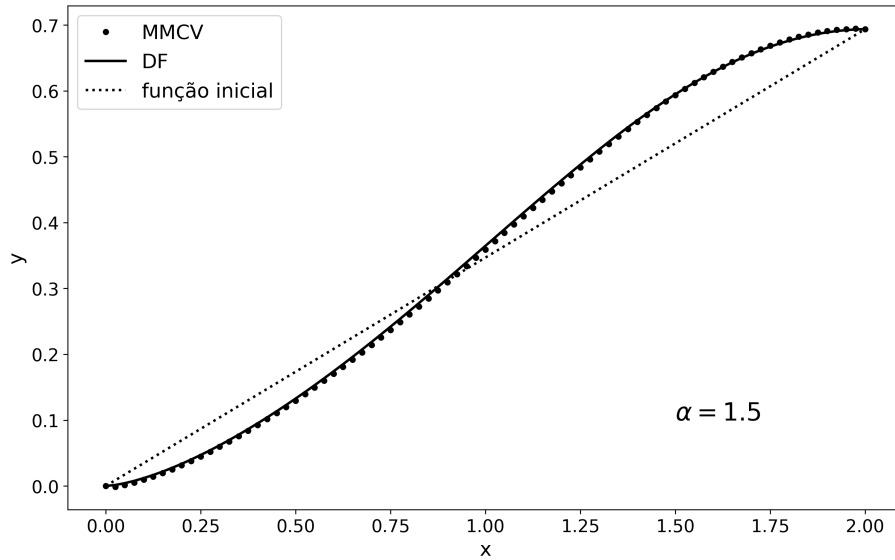


Figura 32: Convergência do MMCV para o primeiro algoritmo com $\alpha = 1.5$ e $n = 25000000$ passos.

próximo de zero quanto possível, dado o erro da discretização da função. O segundo algoritmo convergiu para uma função descontínua, pois o funcional (7.7) admite extremantes descontínuos por ser definido por um integrando sem a presença de derivadas para $0 < x \leq 2$. Como o integrando não contém derivadas, descontinuidades por salto da função não afetam o valor final do funcional, resultando na existência de extremantes descontínuos.

7.2 O Cálculo variacional fracionário

O cálculo variacional tem o seu início no estudo do problema da braquistócrona [42], surgindo em paralelo ao desenvolvimento do cálculo diferencial e integral. O problema da braquistócrona, proposto por Johann Bernoulli, consistia em determinar a curva em que um objeto deslizaria sob a ação da gravidade, entre dois pontos fixos, no menor tempo possível. Outros problemas clássicos como o isoperimétrico, da geodésia e da catenária foram estudados [68]. A análise desses problemas culminou com o desenvolvimento de uma nova área da matemática, o cálculo variacional, cujo objetivo é encontrar os máximos ou mínimos valores de um funcional, ou seja, encontrar os extremos de um funcional. Atualmente, o cálculo variacional continua a ser uma área de pesquisa ativa, com aplicações em mecânica clássica, teoria do controle, otimização de processos, física de partículas, entre outras [26, 44, 81].

O cálculo variacional e o cálculo fracionário estão conectados desde o século XIX. De fato, em 1823, Niels Henrik Abel aplicou o cálculo fracionário para obter a solução

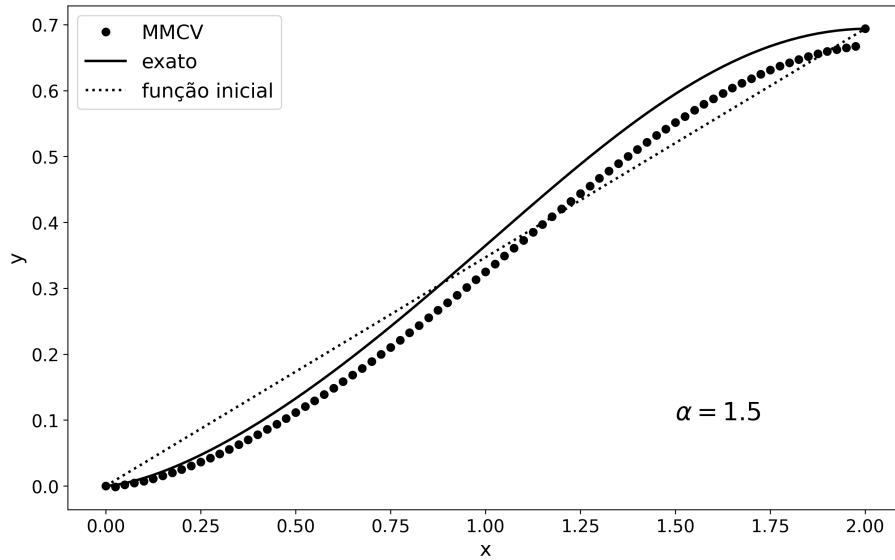


Figura 33: Convergência do MMCV para o segundo algoritmo com $\alpha = 1.5$ e $n = 200000$ passos.

de uma equação integral que surge na formulação do problema da tautócrona [19]. Desde então, o cálculo de variações tem sido uma ferramenta fundamental na análise de equações de movimento em física. No entanto, sistemas dissipativos, como aqueles com amortecimento, representam um desafio para essa abordagem tradicional [64]. Em 1996, Riewe [63] propôs uma nova abordagem ao introduzir derivadas fracionárias em funcionais. Essa inovação permite modelar de forma mais precisa e abrangente uma grande variedade de sistemas físicos. Essa abordagem de Riewe marca o nascimento do estudo do cálculo variacional fracionário. Surgiram várias pesquisas sobre as equações de Euler-Lagrange fracionárias [2, 1, 8], e sobre a teoria do cálculo variacional fracionário [5, 6, 77]

Para reforçar a importância do MMCV que propomos nesta tese, para obter o extremante de funcionais, vamos antes revisar alguns resultados básicos sobre o problema fundamental do cálculo variacional fracionário de Caputo. O problema fundamental consiste em determinar os máximos ou mínimos valores de funcionais dados por:

Problema 5 (Problema Fundamental). *Seja $y \in C^1([a, b], \mathbb{R})$, e considere o problema*

$$\begin{cases} S[y] = \int_a^b F(x, y, {}^C D_x^\alpha y) dx \rightarrow \text{mínimo}, \\ y(a) = y_a, \quad y(b) = y_b \end{cases} \quad (7.12)$$

Tendo as seguintes condições:

- $F \in C^1([a, b] \times \mathbb{R}^2, \mathbb{R})$.

- Para qualquer y , a função $x \rightarrow {}_x D_b^\alpha \left(\frac{\partial F}{\partial {}_a^C D_x^\alpha y} \right)$ é contínua.

Para encontrar a função y^* que minimiza o funcional S em (7.12), definimos a família de funções (variações fracas)

$$y(x) = y^*(x) + \epsilon \eta(x), \quad (7.13)$$

onde ϵ é uma constante ($|\epsilon| \ll 1$) e $\eta \in C^1([a, b], \mathbb{R})$ é uma função arbitrária continuamente diferenciável que satisfaz às condições de fronteira $\eta(a) = \eta(b) = 0$. Da equação (7.13) e pelas condições de fronteira de η , temos que $y^*(a) = y_a$, e $y^*(b) = y_b$, portanto, a função $y \in C^1([a, b])$ com $y(a) = y_a$ e $y(b) = y_b$ é admissível. Como y^* é um extremante para o funcional S , a derivada de Gateaux $\delta S[y^*]$ se torna identicamente nula [2, 1, 8]. Assim, para o funcional em (7.12),

$$\begin{aligned} \delta S[y^*] &= \lim_{\epsilon \rightarrow 0} \left(\int_a^b F(x, y, {}_a^C D_x^\alpha y) dx - \int_a^b F(x, y^*, {}_a^C D_x^\alpha y^*) dx \right) \\ &= \int_a^b \left(\frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial y^*} \eta(x) + \frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} {}_a^C D_x^\alpha \eta(x) \right) dx = 0. \end{aligned} \quad (7.14)$$

Usando o Teorema 2.3.3 sobre integração por partes no segundo termo da equação (7.14) temos

$$\begin{aligned} \int_a^b \left(\frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} {}_a^C D_x^\alpha \eta(x) \right) dx &= \int_a^b {}_x D_b^\alpha \frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} \eta(x) dx \\ &+ \left[\eta(x) {}_x D_b^{1-\alpha} \frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} \right]_{x=a}^{x=b}. \end{aligned} \quad (7.15)$$

Substituindo a equação (7.15) em (7.14) obtemos

$$\begin{aligned} \int_a^b \left[\frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial y^*} + {}_x D_b^\alpha \frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} \right] \eta(x) dx \\ + \left[\eta(x) {}_x D_b^{1-\alpha} \frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} \right]_{x=a}^{x=b} = 0. \end{aligned} \quad (7.16)$$

Considerando que $y(a) = y_a$ e $y(b) = y_b$, devemos ter que $\eta(a) = \eta(b) = 0$, assim da equação (7.16) temos:

$$\int_a^b \left[\frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial y^*} + {}_x D_b^\alpha \frac{\partial F(x, y^*, {}_a^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} \right] \eta(x) dx = 0. \quad (7.17)$$

Considerando que a função $\eta(x)$ é arbitrária, e usando o Lema fundamental do cálculo variacional [29], obtemos a equação diferencial fracionária

$$\frac{\partial F(x, y^*, {}^C D_x^\alpha y^*)}{\partial y^*} + {}_x D_b^\alpha \frac{\partial F(x, y^*, {}^C D_x^\alpha y^*)}{\partial ({}^C D_x^\alpha y^*)} = 0, \quad (7.18)$$

conhecida como equação fracionária de Euler-Lagrange [2, 1, 8].

Finalmente, podemos formular uma condição necessária para uma função y ser o extremante associado ao problema fundamental (5) [2, 1, 8]:

Teorema 7.2.1. *Se y é um minimizante para o funcional S em (7.12) com $y \in C^1([a, b], \mathbb{R})$, $y(a) = y_a$ e $y(b) = y_b$ para $y_a, y_b \in \mathbb{R}$ fixos. Então y é a solução da equação fracionária de Euler-Lagrange*

$$\frac{\partial F(x, y, {}^C D_x^\alpha y)}{\partial y} + {}_x D_b^\alpha \frac{\partial F(x, y, {}^C D_x^\alpha y)}{\partial ({}^C D_x^\alpha y)} = 0. \quad (7.19)$$

Da equação de Euler-Lagrange fracionária (7.19), podemos observar a importância de desenvolver métodos diretos para obter o extremante de funcionais fracionários, como o MMCV que propomos. Em raros casos (muitas vezes para problemas sem interesse prático) a equação de Euler-Lagrange fracionária (7.19) terá solução analítica. Isto porque, para se obter o extremante do funcional, é necessário resolver um problema de fronteira (em geral não linear) envolvendo ao mesmo tempo, derivadas de Caputo à esquerda e derivadas de Riemann-Liouville à direita. Mesmo soluções numéricas para resolver o problema de fronteira para a (7.19) são difíceis de serem implementadas. Por outro lado, o MMCV que propomos permite obter o extremante diretamente do funcional S sem precisarmos resolver a (7.19). Além disso, o MMCV é útil mesmo para problemas de funcionais envolvendo apenas derivadas inteiras, pois, muitas vezes, a função que extremiza o funcional não é solução da equação de Euler-Lagrange.

Observação 7.2.1. *A função $F(x, y, {}^C D_x^\alpha y)$ da equação (7.19) é chamada de Lagrangiano fracionário.*

Observação 7.2.2. *Não vamos discutir a condição de existência para a equação fracionária de Euler-Lagrange neste texto. A existência de soluções fracas é discutida em [14] no qual se prova um teorema apresentando 5 hipóteses sobre o lagrangiano F que garantem para que tenhamos uma solução fraca.*

Finalmente, o cálculo variacional fracionário também pode ser utilizado como ferramenta para resolver alguns tipos de problemas de fronteira. Vamos definir a seguinte classe de problemas de fronteira.

Problema 6 (Problema de Fronteira Variacional). *Seja $y \in C^1([a, b], \mathbb{R})$, e considere o problema de fronteira*

$$\begin{cases} {}_b D_x^\alpha {}^C D_x^\alpha y(x) = f(x, y(x)) \text{ com } 0 < \alpha \leq 1 \text{ e } a < x < b \\ \text{e a condição de fronteira } y(a) = y_a, \quad y(b) = y_b. \end{cases} \quad (7.20)$$

Para encontrar uma possível solução para este problema, vamos encontrar o Lagrangiano fracionário L

$$L := L(x, y, {}^C D_x^\alpha y), \quad 0 < \alpha \leq 1, \quad (7.21)$$

tal que

$$\frac{\partial L}{\partial y} + {}_x D_b^\alpha \frac{\partial L}{\partial ({}^C D_x^\alpha y)} = {}_b D_x^\alpha {}^C D_x^\alpha y(x) - f(x, y(x)). \quad (7.22)$$

Supondo que o Lagrangiano fracionário associado a esse problema é dado por

$$L(x, y, {}^C D_x^\alpha y) = \frac{1}{2} ({}^C D_x^\alpha y)^2 + g(x, y(x)). \quad (7.23)$$

Assim, aplicando (7.23) no lado esquerdo da equação (7.22) obtemos

$$({}_b D_x^\alpha ({}^C D_x^\alpha y(x)) + \frac{\partial g(x, y(x))}{\partial y}). \quad (7.24)$$

Usando (7.23) obtemos

$$\frac{\partial g(x, y(x))}{\partial y} = -f(x, y(x)). \quad (7.25)$$

Portanto, podemos encontrar o Lagrangiano fracionário (7.23) associado ao Problema (7.20) calculando a função $g(x, y(x))$ com a condição (7.25). Com esse Lagrangiano fracionário, podemos o MMCV que propomos para obter uma possível solução do problema de fronteira (7.20).

7.2.1 Um algoritmo para o extremante de funcionais

O algoritmo que propomos para obter o extremante de funcionais difere do primeiro algoritmo para problemas de fronteira apenas pelo funcional. Ao invés de utilizar o funcional $J[y]$ definido em (7.2), para o problema de extremantes utilizamos diretamente o funcional $S[y]$ definindo o Problema Fundamental (7.12). Dado o funcional (7.12), o algoritmo do MMCV que propomos segue os mesmos passos 1 e 2 do primeiro algoritmo. Para o cálculo do funcional (7.12), utilizamos novamente o método do trapézio dividindo o intervalo $[a, b]$ em L partes iguais de comprimento h . Também usamos novamente a função discretizada $y^L(x) = y_l$, para $x_l \leq x < x_{l+1}$, definida na (5.5) do Capítulo 5. Com a função discretizada, temos $y(a) = y_0$ e $y'(a) = y'_0$ onde aproximamos $y'_0 = \frac{y_1 - y_0}{h}$. Para a derivada de Caputo no integrando do funcional (7.12) temos a aproximação, usando

a função discretizada,

$$\begin{aligned}
 {}^C D_{a+lh}^\alpha y^L(a+lh) &= \frac{1}{\Gamma(1-\alpha)} \int_a^{a+lh} \frac{y'^L(u)}{(a+lh-u)^\alpha} du \\
 &= \frac{1}{\Gamma(1-\alpha)} \sum_{j=0}^{l-1} y'_j \int_{a+jh}^{a+(j+1)h} \frac{1}{(a+lh-u)^\alpha} du \\
 &= \frac{h^{1-\alpha}}{\Gamma(2-\alpha)} \sum_{j=0}^{l-1} y'_j [(l-j)^{1-\alpha} - (l-j-1)^{1-\alpha}],
 \end{aligned} \tag{7.26}$$

onde $0 < \alpha \leq 1$, e usamos $x = a + lh$.

7.2.2 Exemplo 1

Neste primeiro exemplo, consideramos o seguinte funcional:

$$S[y] = \int_0^2 \left(({}^C D_x^\alpha y)^2 + y^2 \right) dx, \tag{7.27}$$

onde y satisfaz a condição de fronteira $y(0) = 1$ e $y(2) = e^2$.

Como o integrando definindo o funcional (7.27) é positivo, é evidente que deve existir uma função que minimiza o funcional. Essa função que minimiza o funcional (extremante) é a solução da equação de Euler-Lagrange fracionária

$${}_2 D_{x0}^{\alpha C} D_x^\alpha y(x) = -y(x), \tag{7.28}$$

que satisfaz a condição de fronteira $y(0) = 1$ e $y(2) = e^2$. Portanto, para obter o extremante do funcional (7.27) precisamos resolver um problema de fronteira bem complicado, envolvendo ao mesmo tempo, derivadas de Riemann-Liouville e Caputo, tendo ainda derivadas à direita e à esquerda.

Antes de obter o extremante do funcional (7.27) para derivadas fracionárias, vamos mostrar que o MMCV que propomos permite obter também o extremante de funcionais com derivadas de ordem inteira ($\alpha = 1$). Note que neste caso, com derivada de ordem inteira, a equação de Euler-Lagrange se reduz ao problema de fronteira

$$\frac{d^2 y}{dx^2} = -y, \tag{7.29}$$

com $y(0) = 1$ e $y(2) = e^2$, e que tem como solução exata o extremante $y(x) = e^x$.

Na Figura 34 apresentamos a solução obtida pelo MMCV para o problema variacional (7.27) com derivada de ordem $\alpha = 1$. Mostramos também no gráfico a solução exata $\alpha = 1$. Para obter a solução pelo MMCV, utilizamos $L = 120$ (o domínio $[0, 2]$ foi dividido em 120 partes iguais, e a função discretizada tem 121 pontos). O valor utilizado para Δy_i em cada iteração foi um número aleatório no intervalo $[-0.005, 0.005]$, ou seja,

usamos $\delta = 0.005$. Utilizamos ainda um número de iterações $n = 10000000$. Finalmente, a função usada como ponto de partida foi uma reta. Vemos neste gráfico uma excelente concordância entre o resultado obtido pelo MMCV com o resultado exato.

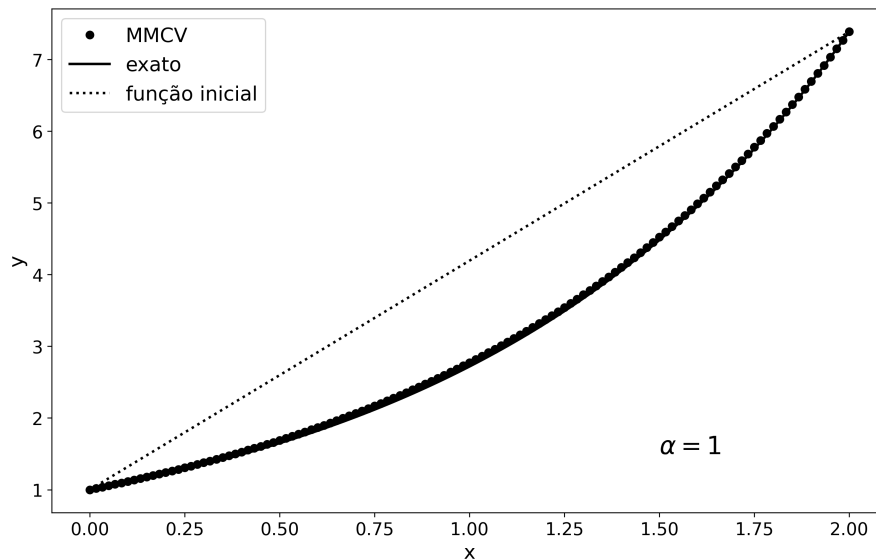


Figura 34: Convergência do MMCV para o primeiro algoritmo com $\alpha = 1$ e $n = 10000000$ iterações.

Para efeito de estimativa de erro para a solução obtida pelo MMCV, quando não tivermos uma solução exata para comparação, vamos definir uma função que chamaremos de “função desvio”:

Definição 7.2.1. A função desvio para o funcional (7.27) é definida como:

$$desvio(y) = \frac{\partial F(x, y^*, {}^C D_x^\alpha y^*)}{\partial y^*} + {}_x D_b^\alpha \frac{\partial F(x, y^*, {}^C D_x^\alpha y^*)}{\partial ({}_a^C D_x^\alpha y^*)} = {}_b D_x^\alpha {}_a^C D_x^\alpha y + y. \quad (7.30)$$

Da equação (7.30), vemos que se y for a solução da equação de Euler-Lagrange a função desvio será nula. Por outro lado, se y diferir pouco, e suavemente, da solução da equação de Euler-Lagrange, então a função desvio será uma função com valores próximos de zero. Desta forma, a função desvio serve como uma medida para o quanto a função y difere (desvio) do extremante. No entanto, é importante reforçar que, apesar de servir para estimar qualitativamente se a função y está próxima do extremante exato do funcional, a função desvio não fornece o erro ao utilizar a função y como uma aproximação para o extremante. De fato, o valor da função desvio será, em geral, muito maior que o erro. Isto, pois a derivada contida na definição da função desvio tem o efeito de amplificar o erro da função.

Para o problema com derivada de ordem inteira ($\alpha = 1$), a função desvio reduz-se à

$$desvio(y) = -\frac{d^2y}{dx^2} + y. \quad (7.31)$$

Vemos na Figura 35 a função desvio, assim como o erro absoluto $y - e^x$ e o erro relativo $\frac{y-e^x}{e^x}$ para a solução obtida pelo MMC para o problema (7.27). No gráfico (a) podemos observar que o erro absoluto e o erro relativo são muito menores do que o valor da função desvio. Além disso, a função desvio apresenta um caráter oscilatório devido às irregularidades da função discretizada. Essas irregularidades são ampliadas pela derivada de segunda ordem definindo a função desvio. No entanto, eliminando-se essas irregularidades, pode-se observar que, em média, a função desvio está próxima de zero. No gráfico (b) mudamos a escala para facilitar a visualização dos erros absoluto e relativo. Finalmente, uma forma de reduzir o erro é aumentar o valor de L , refinando a discretização.

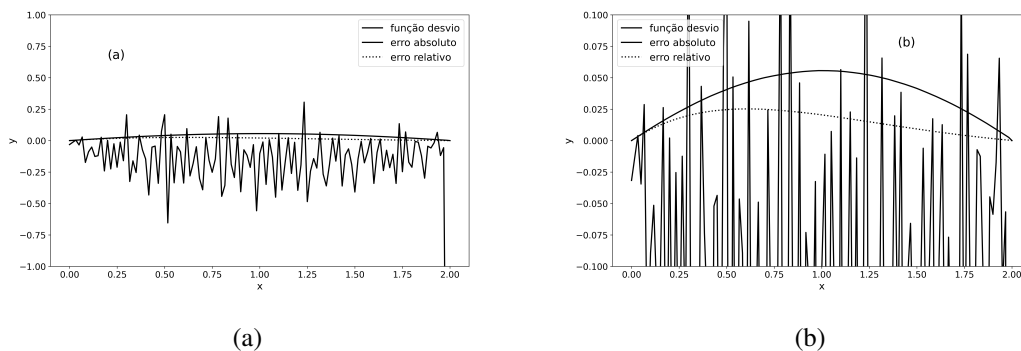


Figura 35: Função desvio, erro absoluto e erro relativo.

Vamos considerar agora o problema (7.27) com derivada fracionária. Na Figura 36 apresentamos a solução obtida pelo MMCV para o caso onde $\alpha = 0.75$. Utilizamos novamente $L = 120$ (o domínio $[0, 2]$ foi dividido em 120 partes iguais, e a função discretizada tem 121 pontos). Agora, o valor utilizado para Δy_i em cada iteração foi um número aleatório no intervalo $[-0.002, 0.002]$, ou seja, usamos $\delta = 0.002$. Utilizamos ainda um número de iterações $n = 10000000$, a função inicial utilizada foi novamente uma reta. Como neste caso não temos uma solução exata simples para a equação de Euler-Lagrange, neste gráfico optamos por incluir a solução do problema com derivada de ordem inteira ($\alpha = 1$) a título de comparação. Vemos neste gráfico que o extremante obtido pelo MMCV é uma função que começa com uma taxa de crescimento menor do que a da função exponencial, mas termina com uma taxa de crescimento maior.

Na Figura 37, vemos a função desvio (7.2.1) para o caso $\alpha = 0.75$. No gráfico (a) vemos que a função desvio está próxima de zero na maior parte do domínio, indicando que a solução obtida pelo MMCV é uma boa aproximação para o extremante do funcional.

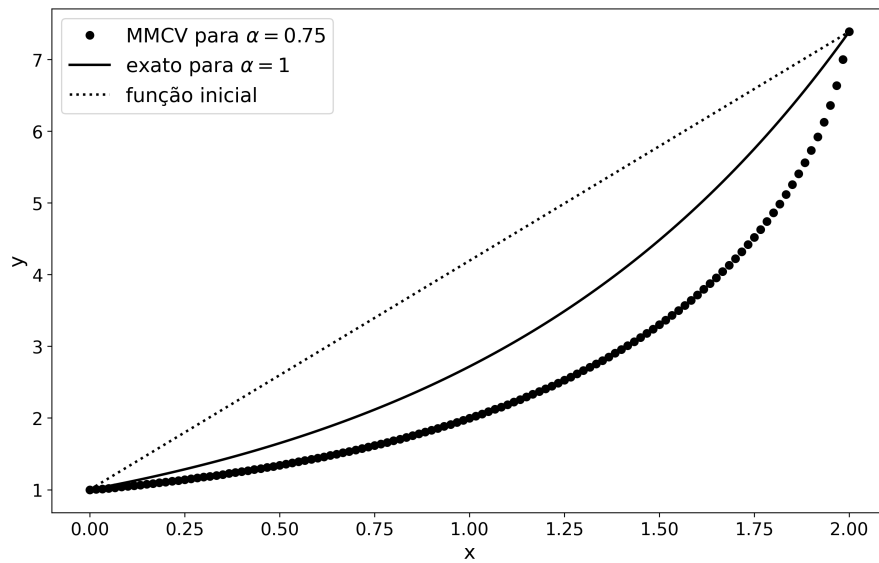


Figura 36: Convergência do MMCV para o primeiro algoritmo com $\alpha = 0.75$ e $n = 10000000$ passos.

No entanto, próximo à fronteira à direita do domínio, a função erro afasta-se de zero. Isto ocorre devido ao polo no integrando definindo a derivada de Riemann-Liouville em (7.2.1), que amplifica o erro numérico na aproximação da derivada fracionária. No gráfico (b) mudamos a escala para facilitar a visualização do desvio.

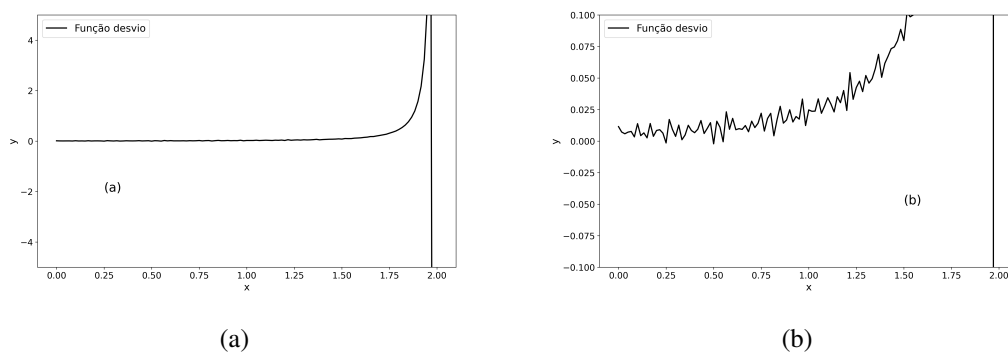


Figura 37: Função desvio.

7.2.3 Exemplo 2

Neste exemplo, vamos considerar um problema que ilustra a importância do MMCV proposto mesmo para o caso onde a derivada é de ordem inteira. Vamos considerar o

seguinte problema:

$$S[y] = \int_1^2 \left((1 + {}^C D_x^\alpha y)^2 (1 - {}^C D_x^\alpha y)^2 \right) dx \longrightarrow \text{mínimo}, \quad (7.32)$$

onde y satisfaz a condição de fronteira $y(1) = 1$ e $y(2) = \frac{1}{2}$.

Como o integrando definindo o funcional (7.32) é positivo, é evidente que deve existir uma função que minimiza o funcional. No entanto, mesmo para derivada de ordem inteira ($\alpha = 1$), a solução da equação de Euler-Lagrange para o problema (7.32) não é a função que minimiza o funcional (7.32). De fato, a solução da equação de Euler-Lagrange é a função de classe $C^2[1, 2]$ que maximiza o funcional [54]. Isso ocorre, pois as funções que minimizam o funcional (7.32), para derivadas de ordem inteira ($\alpha = 1$), são não diferenciáveis. Portanto, não são solução da equação de Euler-Lagrange. Neste caso, para obter os extremantes, precisamos utilizar métodos diretos, como o MMCV que propomos nesta tese. De fato, essas funções (e existem infinitas delas) que minimizam o funcional (7.32) são funções definidas por partes, aonde cada parte é constituída de um segmento de reta com inclinação ± 1 [54]. Vemos na Figura 38 que o MMCV proposto convergiu corretamente para uma dessas funções que minimizam o funcional. Não apresentamos a função desvio para este problema, pois, como o extremante é não diferenciável, ele não satisfaz a equação de Euler-Lagrange usada para definir a função desvio. Para a Figura 38, utilizamos $L = 120$, $\delta = 0.005$, e um número de iterações $n = 5000000$.

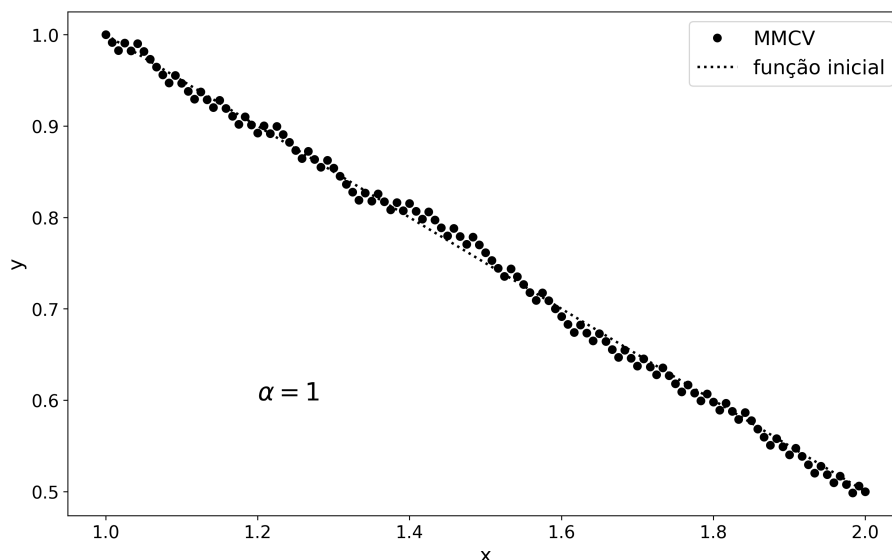


Figura 38: Convergência do MMCV para o primeiro algoritmo com $\alpha = 1$ e $n = 5000000$ passos.

Para o problema com derivada fracionária de ordem $\alpha = 0.75$, vemos novamente na

Figura 39 que o extremante é não diferenciável (função definida por partes). Portanto, o extremante não é solução da equação de Euler-Lagrange fracionária. Para a Figura 38, utilizamos também $L = 120$, $\delta = 0.005$, e um número de iterações $n = 5000000$. Devido à baixa resolução (valor pequeno de L) não é possível determinar se as partes dessa função são segmentos de reta ou se são outra curva. No entanto, podemos ver que a inclinação média dessas partes é, em módulo, maior que 1.

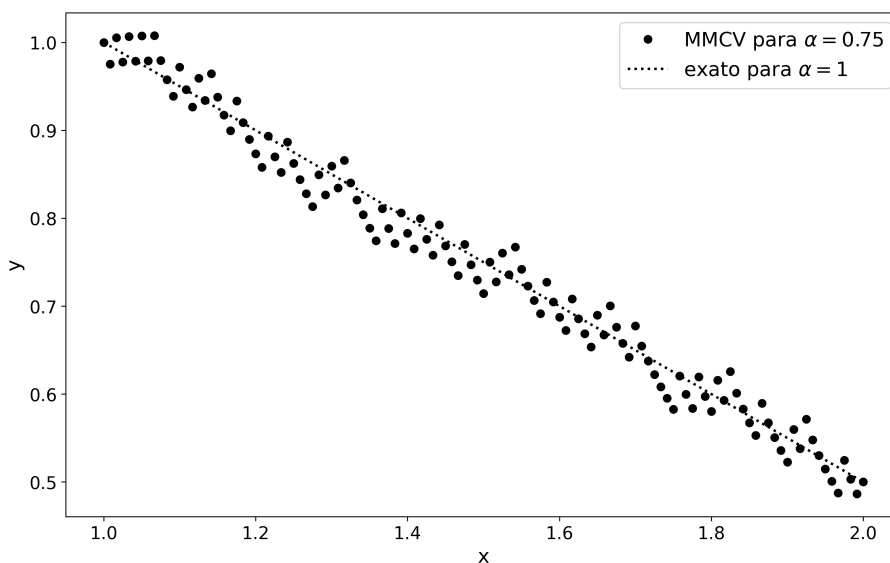


Figura 39: Convergência do MMCV para o primeiro algoritmo com $\alpha = 0.75$ e $n = 5000000$ passos.

7.2.4 Exemplo 3

No último exemplo deste subcapítulo, vamos obter o extremante do funcional associado ao problema da braquistócrona na sua versão fracionária. O problema da braquistócrona é um dos mais famosos do cálculo variacional. Foi proposto em 1696 por Johann Bernoulli, e consiste em encontrar a curva que uma partícula P deve seguir ao se deslocar de um ponto A até o ponto B , situados no mesmo plano vertical, no menor tempo possível, apenas sob a ação da gravidade.

Supondo que o ponto A está na posição $A(0, 1)$ sobre o eixo y , e que o ponto B está na posição $B(1, 0)$ sobre o eixo x (veja a Figura 40), e seja $y = y(x)$ a função que descreve a trajetória da partícula em função da posição na horizontal. Então, a função y deve satisfazer a condição de fronteira $y(0) = 1$ e $y(1) = 0$. Além disso, vamos considerar que a partícula começa o percurso com velocidade inicial nula, na posição $A(0, 1)$, no instante de tempo $t_i = 0$, e chega no ponto final $B(1, 0)$ no instante $t_f > t_i$. Se s_f é o comprimento da curva y , o tempo $\Delta t = t_f - t_i$ que a partícula leva para percorrer o trajeto

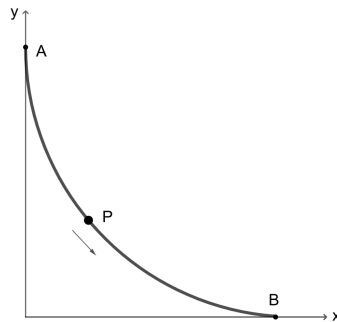


Figura 40: Partícula P deslizando numa curva $y(x)$ sem atrito do ponto A até o ponto B .

é dado por:

$$\Delta t = \int_{t_i}^{t_f} dt = \int_0^{s_f} \frac{ds}{v} = \int_0^1 \frac{\sqrt{1 + (y')^2}}{v}, \quad (7.33)$$

onde s denota o comprimento do arco da curva, $v = \frac{ds}{dt}$ é a velocidade escalar da partícula, e $y' = \frac{dy}{dx}$. Supondo que não há atrito, então a soma da energia cinética com a energia potencial (energia total) deve ser uma constante, ou seja $\frac{1}{2}mv^2 + mgy = mg$, onde m é a massa da partícula P e g é a constante da gravidade. Logo, $v = \sqrt{2g(1 - y)}$ e podemos reescrever a equação (7.33) como um funcional

$$S[y] = \int_0^1 \sqrt{\frac{1 + (y')^2}{2g(1 - y)}} dx, \quad (7.34)$$

onde y satisfaz a condição de fronteira $y(0) = 1$ e $y(1) = 0$.

O problema da braquistócrona é então equivalente a encontrar a função y que minimize o funcional $S[y]$. A solução para esse funcional é bem conhecida e é dada por um arco de cicloide invertido [54].

Vamos agora considerar a generalização do problema da braquistócrona para um funcional fracionário utilizando a derivada de Caputo. Temos o problema de encontrar a função y que minimiza o funcional:

$$\begin{cases} S[y] = \int_0^1 \sqrt{\frac{1 + ({}_0^C D^\alpha y)^2}{2g(1 - y)}} dx \rightarrow \text{mínimo}, \\ \text{com a condição de fronteira } y(0) = 1, \ y(1) = 0 \text{ e } 0 < \alpha \leq 1. \end{cases} \quad (7.35)$$

Como veremos nos resultados obtidos pelo MMCV, a introdução da derivada fracionária no problema da braquistócrona introduz efeitos que qualitativamente podem ser interpretados como efeitos de uma força de atrito viscoso não trivial. Isto está de acordo com o esperado para problemas de mecânica, onde a derivada fracionária pode ser utilizada para modelar problemas envolvendo forças de atrito viscoso [34].

A solução analítica para o problema (7.35) não é conhecida. A equação de Euler-

Lagrange fracionária associada a esse funcional é bem complexa. Antes de aplicar o MMCV para o problema fracionário, vamos utilizar o algoritmo desenvolvido nesse subcapítulo para recuperar a solução analítica quando $\alpha = 1$. Na Figura 41 apresentamos a solução obtida pelo MMCV para o caso onde $\alpha = 1$. Utilizamos $L = 80$ (o domínio $[0, 1]$ foi dividido em 80 partes iguais, e a função discretizada tem 81 pontos). O valor utilizado para Δy_i em cada iteração foi um número aleatório no intervalo $[-0.005, 0.005]$, ou seja, usamos $\delta = 0.005$. Utilizamos ainda um número de iterações $n = 10000000$, e a função inicial utilizada foi novamente uma reta. Para fins de comparação, nesta Figura também temos a solução exata do problema. Podemos observar graficamente uma boa concordância entre a solução obtida pelo MMC com a solução analítica.

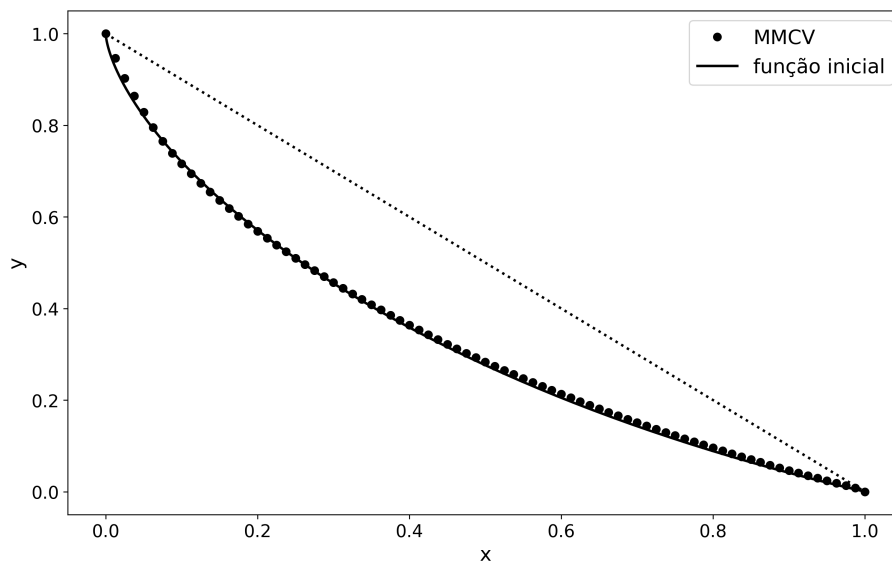


Figura 41: Convergência do MMCV para a braquistócrona com $\alpha = 1$ e $n = 10000000$ passos.

Na Figura 42 podemos ver a solução obtida pelo MMCV para o problema da braquistócrona fracionária com derivada de Caputo de ordem $\alpha = 0.5$. Também utilizamos $L = 80$, $\delta = 0.005$, e um número de iterações $n = 10000000$. A função inicial utilizada foi novamente uma reta. Vemos neste gráfico que a solução obtida pelo MMCV para o problema fracionário é muito diferente do caso com derivada de ordem inteira. O movimento da partícula P inicia-se quase em queda livre, mudando após para uma trajetória quase horizontal antes de atingir o ponto final. Esse início em queda livre é esperado quando temos um problema de atrito viscoso, pois neste caso, para ganhar velocidade rapidamente (contra a força de atrito), é necessária uma trajetória que maximize o efeito da gravidade. Portanto, a trajetória encontrada está de acordo com o esperado para um problema com atrito viscoso não trivial introduzido pela derivada fracionária, como comentamos anteriormente.

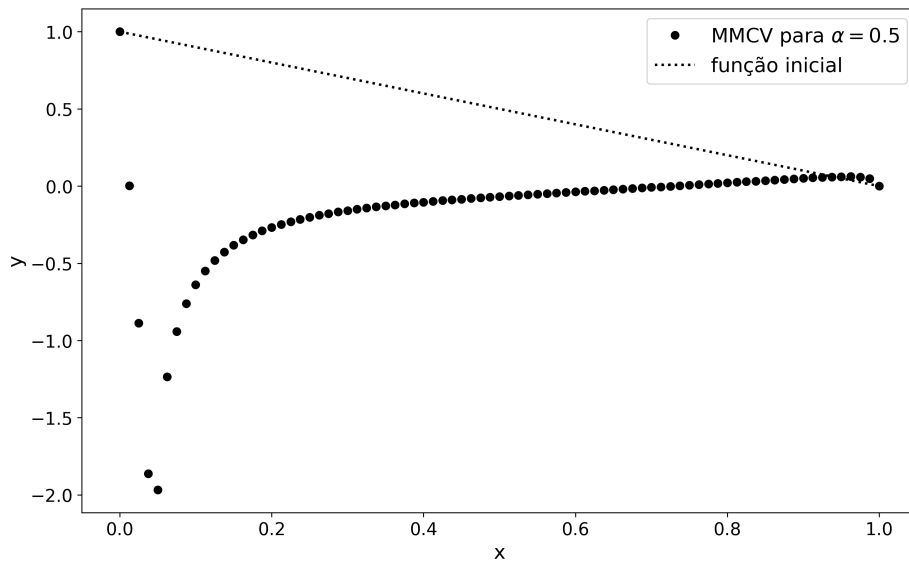


Figura 42: Convergência do MMCV para a braquistócrona com $\alpha = 0.5$ e $n = 10000000$ passos.

Para estimar a validade do resultado obtido pelo MMC para o problema da braquistócrona fracionária, já que não temos solução exata para comparação, e tampouco temos soluções numéricas por outros métodos, apresentamos na Figura 43 a função desvio para o problema fracionário. Vemos dos gráficos desta Figura que a função desvio é muito próxima de zero para a maior parte do domínio, sendo necessário ampliar a escala em cerca de 1000 vezes (gráfico (c)) para visualizar a flutuação da função desvio em torno de zero. Como consequência, a solução obtida pelo MMCV está muito próxima da solução exata do problema na maior parte do domínio. A função desvio só difere significativamente de zero para valores de x próximos de zero, onde a função lagrangiana e a derivada fracionária têm polo, e para x próximo de 1. Neste último caso, o gráfico da solução apresentado na Figura 42 sugere que a solução obtida apresenta uma pequena descontinuidade em $x = 1$. Acreditamos que isto seja efeito do erro numérico da discretização da função próximo ao polo do integrando (em $x = 0$), que afeta o valor da integral definindo o funcional, deslocando a solução ligeiramente para cima.

7.3 Um algoritmo de MMCV usando aproximação por funções potência

Neste último subcapítulo da tese, propomos um algoritmo alternativo de MMCV que é construído a partir da aproximação da função y (solução de um problema de fronteira ou solução de um problema de extremante) por uma função potência. Chamaremos esta função potência de função teste $y^T(x) = y(x)$. Esta função teste deve satis-

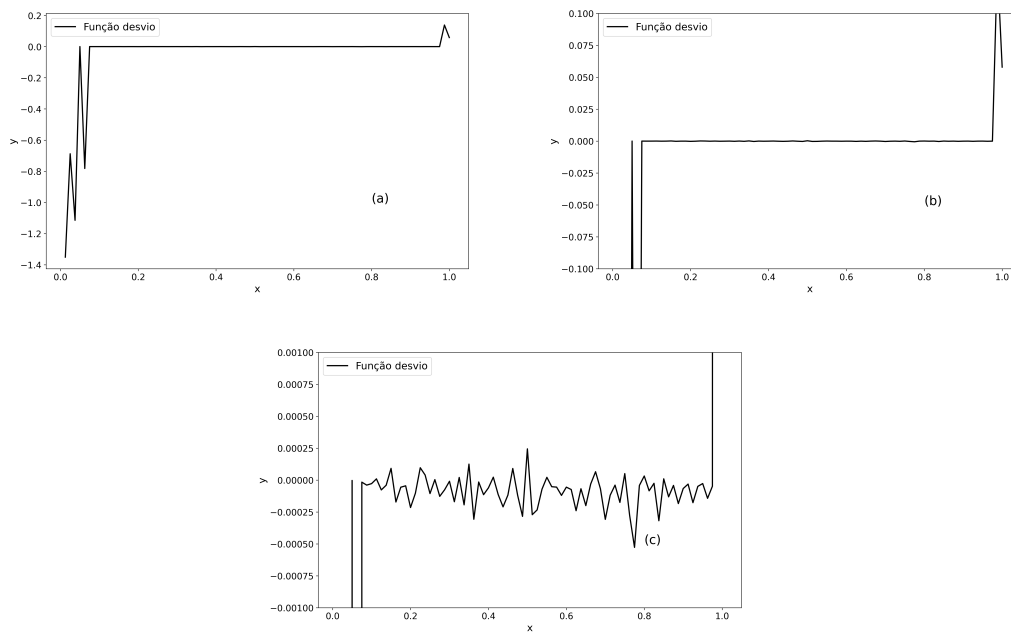


Figura 43: Função desvio para a braquistócrona com $\alpha = 0.5$ e $n = 10000000$ passos.

fazer a condição de fronteira do problema em questão. Um exemplo de função teste seria a função potência $y^T = y^T(\mu, \beta, x) = \mu x^\beta$ com o coeficiente $\mu \in \mathbb{R}$ e o expoente $\beta \in \mathbb{R}$. O MMCV que propomos é obtido seguindo os passos (usaremos a função potência $y^T = \mu x^\beta$ para ilustrar o método):

1. Inicialização: escolhe-se uma função inicial $y_0^T = \mu_0 x^{\beta_0}$ que satisfaz a condição de fronteira e calcula-se o valor do funcional $S[y_0^T]$ (definido em (7.2) para um problema de fronteira, e definido em (7.12) para um problema de extremante de funcionais).
2. Iterando:
 - (a) Gerando uma nova função $y_1^T = \mu_1 x^{\beta_1}$ candidata à solução do problema: dados números reais positivos δ e ϵ , sorteia-se seguindo-se uma distribuição uniforme, um número real $\Delta\mu \in [-\delta, \delta]$ e $\beta_1 \in [-\epsilon, \epsilon]$ e define-se o novo valor do coeficiente $\mu_1 \rightarrow \mu_0 + \Delta\mu$ e do expoente $\beta_0 \rightarrow \beta_1$.
 - (b) Verificando se a nova função será aceita: com a nova função y_1^T construída no passo (a), calcula-se o novo valor do funcional $S[y_1^T]$. Se $S[y_1^T] < S[y_0^T]$, aceita-se a mudança atualizando o valor de y_0^T fazendo $y_0^T = y_1^T$. Atualiza-se também o valor de $S[y_0^T]$ fazendo $S[y_0^T] = S[y_1^T]$. Por outro lado, se $S[y_1^T] \geq S[y_0^T]$, rejeita-se a nova função y_1^T e mantêm-se os valores anteriores de y_0^T e $S[y_0^T]$.
 - (c) Retornando: com os valores finais de y^T e $S[y^T]$ obtidos após o passo (b), retorna-se ao passo (a) um número n (número de iterações) de vezes.

Esse algoritmo iterativo tem como objetivo construir uma sequência de funções a partir da função inicial ao modificar aleatoriamente os valores das novas funções no seu coeficiente μ e no expoente β . Se essa nova escolha resulta em uma nova função que reduz o valor do funcional, então essa nova função é aceita. Caso contrário, a nova função é rejeitada. Ao repetir o procedimento n vezes, teremos uma sequência de novas funções aceitas que satisfazem às condições de fronteira e convergem estatisticamente para a solução do problema (7.12).

Observação 7.3.1. *Os passos apresentados acima podem ser facilmente estendidos para funções testes contendo mais termos, como, por exemplo, para $y^T(x) = c_0 + c_1x^\beta + c_2x^{2\beta} + c_3x^{3\beta} + c_4x^{4\beta} + c_5x^{5\beta}$. Neste caso, no passo (a) sorteia-se aleatoriamente um dos coeficientes c_i para ser modificado.*

Observação 7.3.2. *O algoritmo proposto pode ser facilmente modificado para se utilizar como função teste qualquer combinação linear de funções linearmente independentes. Neste caso poderíamos, por exemplo, aproximar a função y por uma função teste que seja os primeiros termos de uma série de Fourier.*

Observação 7.3.3. *Finalmente, o MMCV proposto aqui converge geralmente mais rápido que os algoritmos propostos nos subcapítulos anteriores. No entanto, os algoritmos anteriores apresentam a vantagem de convergirem mesmo quando a solução do problema é uma função não diferenciável.*

7.3.1 Exemplo 1

Para ilustrar o funcionamento desse algoritmo, vamos considerar o mesmo problema analisando em 7.1.3 de fronteira dada por:

$$\begin{cases} {}^C D_x^\alpha y(x) = y(x) + 1 & \text{com } 1 < \alpha \leq 2 \text{ e } 0 < x < 2 \\ \text{e a condição de fronteira } y(0) = 0, \quad y(2) = E_\alpha(2^\alpha) - 1. \end{cases} \quad (7.36)$$

A solução analítica do problema é a equação (7.10).

Como feito em 7.1.3, para construir o algoritmo do MMCV proposto, reescreve-se a equação diferencial em (7.36) como um funcional:

$${}^C D_x^\alpha y(x) - y(x) - 1 = 0 \implies S[y] = \int_a^b ({}^C D_x^\alpha y(x) - y(x) - 1)^2 dx = 0, \quad (7.37)$$

sujeito às condições de fronteira $y(0) = 0$ e $y(2) = E_\alpha(2^\alpha) - 1$. Note que o funcional $S[y]$ tem valor mínimo igual a zero quando $y(x)$ é a solução do problema de fronteira (7.36).

Vamos ainda considerar a seguinte função teste:

$$\left\{ y^T(x) = c_0 + c_1x^\beta + c_2x^{2\beta} + c_3x^{3\beta} + c_4x^{4\beta} + c_5x^{5\beta}. \right. \quad (7.38)$$

Como $y(0) = 0$ temos que $c_0 = 0$, e sabendo que $y(2) = E_\alpha(2^\alpha) - 1$, obtemos a função potência simplificada candidata à função teste do problema (7.36)

$$\left\{ \begin{array}{l} y^T(x) = a_1x^\beta + a_2x^{2\beta} + a_3x^{3\beta} + a_4x^{4\beta} + a_5x^{5\beta} \\ \text{com } a_5 = (E_\alpha(2^\alpha) - 1 - a_12^\beta - a_22^{2\beta} - a_32^{3\beta} - a_42^{4\beta}) / (2^{5\beta}) \\ \text{que satisfaz a condição de fronteira } y(0) = 0, \quad y(1) = E_\alpha(2^\alpha) - 1. \end{array} \right. \quad (7.39)$$

A função teste tem 5 coeficientes $(a_1, a_2, a_3, a_4, a_5)$, porém o último coeficiente a_5 depende dos outros coeficientes, assim observa-se que a função teste pode ser representada por $y^T(a_1, a_2, a_3, a_4, \beta, x)$. Iniciamos com $a_1 = a_2 = a_3 = a_4 = 0.1$ e $\beta = 0.5$, assim, com a função teste inicial $y_0^T = y^T(0.1, 0.1, 0.1, 0.1, 0.5, x)$, calculamos o valor do funcional (7.37) $S[y_0^T]$.

Para gerar uma nova função teste, não alteramos todos os coeficientes de uma vez, sendo assim escolhemos um dos coeficientes (a_1, a_2, a_3, a_4) de forma aleatória e o alteramos para $a_i = a_i + \epsilon h$ com $h = 0.02$ e ϵ um número aleatório no intervalo uniformemente distribuído $[-1, 1]$. Além disso, escolhemos aleatoriamente o novo valor para o expoente β no intervalo $[0.3, 0.6]$. Calculamos então o valor do funcional desta nova função teste gerada $S[y_1^T]$. Faz-se o processo conforme descrito no algoritmo em n passos e obtém-se uma sequência de polinômios que converge para a aproximação da solução do problema.

Na Figura 44 temos a solução do problema de fronteira (7.36) em 10000 passos. A função potência solução encontrada foi:

$$\begin{aligned} y^T(x) = & 0.00000341654989786409x^{0.473135652476533} + 0.1x^{1.89254260990613} \\ & + 0.11031819965133x^{0.946271304953067} + 0.0921240475589665x^{1.4194069574296} \\ & + 0.614358758331967x^{2.36567826238267} \end{aligned}$$

Vemos ainda na Figura 44(a) que a solução encontrada tem uma boa concordância com a solução analítica do problema, visto que utilizamos apenas 10000 passos e usamos uma função teste com apenas 5 coeficientes. Para analisar a convergência da sequência de funções teste geradas pelo processo definindo o MMCV, na Figura 44(b) temos o gráfico de todas as funções teste que foram aceitas e depois descartadas no algoritmo. Essas soluções criaram uma sequência de funções teste que evoluem para a solução que melhor aproxima a solução exata deste problema. Observando que o resultado foi obtido com apenas 10000 passos, apresentando uma vantagem de custo computacional em relação aos algoritmos propostos nos subcapítulos anteriores.

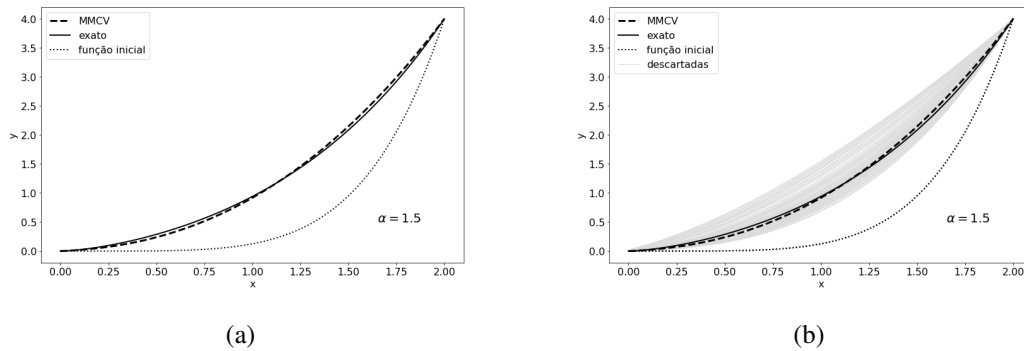


Figura 44: Monte Carlo Variacional para o PVF 7.36 com 10000 passos com $\alpha = 1.5$.

7.3.2 Exemplo 2

Neste exemplo, vamos considerar o PFV que envolve as derivadas de Riemann-Liouville à direita e de Caputo à esquerda.

$$\begin{cases} \frac{\pi}{2} D_x^\alpha {}^C D_x^\alpha y(x) = y(x) & \text{com } 0 < \alpha \leq 1 \text{ e } 0 < x < \frac{\pi}{2} \\ \text{e a condição de fronteira } y(0) = 0, \quad y(\frac{\pi}{2}) = 1. \end{cases} \quad (7.40)$$

Com $\alpha = 1$ temos uma solução exata, que é $y(x) = \sin(x)$, porém, para $0 < \alpha < 1$ as soluções analíticas são bem complicadas de serem descritas e implementadas [9]. Em [9] há uma discussão e análise de casos gerais de sua solução. Este problema pertence à classe dos osciladores fracionários unidimensionais.

Para construir um algoritmo do MMCV para obter uma solução, vamos associar o problema (7.40) a um funcional através de uma função lagrangiana fracionária. Para obter a solução do problema (7.40), basta então minimizar esse funcional usando os passos definidos no MMCV proposto neste subcapítulo.

Portanto, de acordo com (7.23), (7.24) e (7.25) o funcional a ser minimizado é dado por

$$\begin{cases} S[y] = \int_0^1 \left(\frac{1}{2} ({}^C D_x^\alpha y)^2 - \frac{1}{2} y^2 \right) dx \rightarrow \text{mínimo}, \\ y(0) = 0, \quad y(\frac{\pi}{2}) = 1 \end{cases} \quad (7.41)$$

Pelo teorema 7.2.1, se $y(x)$ é um minimizante do funcional (7.41) então $y(x)$ será solução do exemplo (7.40).

Vamos utilizar a mesma classe de funções teste (7.38) do exemplo 7.3.1 adaptando para os novos valores fixos da fronteira. Assim, temos a seguinte função teste do problema

(7.41)

$$\begin{cases} y^T(x) = a_0 + a_1x^\beta + a_2x^{2\beta} + a_3x^{3\beta} + a_4x^{4\beta} + a_5x^{5\beta}, \\ \text{com } a_0 = 0 \text{ e } a_5 = \left(\frac{2}{\pi}\right)^{5\beta} - a_1\left(\frac{2}{\pi}\right)^{4\beta} - a_2\left(\frac{2}{\pi}\right)^{3\beta} - a_3\left(\frac{2}{\pi}\right)^{2\beta} - a_4\left(\frac{2}{\pi}\right)^\beta, \\ \text{que satisfaz a condição de fronteira } y(0) = 0, \quad y(1) = \frac{\pi}{2}. \end{cases} \quad (7.42)$$

A função teste $y^T(x)$ é uma soma de funções potência com números reais nos expoentes. Poderíamos ter escolhido outra classe de função teste, porém escolhemos (7.42) para facilitar o cálculo da derivada de Caputo, visto que a derivada de Caputo para funções potências é dada por:

$${}_0^C D_x^\alpha x^\beta = \frac{\Gamma(\beta + 1)}{\Gamma(\beta + 1 - \alpha)} x^{\beta - \alpha} \quad \text{se } \alpha - \beta \notin \mathbb{N}. \quad (7.43)$$

Assim, a derivada de Caputo da função teste (7.42) considerando $a_0 = 0$ é

$${}_0^C D_x^\alpha y^T(x) = \sum_{k=1}^5 a_k \frac{\Gamma(k\beta + 1)}{\Gamma(k\beta + 1 - \alpha)} x^{k\beta - \alpha} \quad \text{se } \alpha - k\beta \notin \mathbb{N}. \quad (7.44)$$

Observando ainda que o coeficiente a_5 depende dos coeficientes (a_1, a_2, a_3, a_4) a função teste pode ser representada por $y^T(a_1, a_2, a_3, a_4, \beta, x)$. Iniciamos com $a_1 = a_2 = a_3 = a_4 = 0$ e $\beta = 1$, assim, com a função teste inicial $y_0^T = y^T(0, 0, 0, 0, 1, x) = \left(\frac{2}{\pi}\right)^5 x^5$ e com ${}_0^C D_x^\alpha y_0^T = \left(\frac{2}{\pi}\right)^5 \frac{\Gamma(6)}{\Gamma(6 - \alpha)} x^{5 - \alpha}$ calculamos o valor do funcional

$$S[y_0^T] = \int_0^1 \left(\frac{1}{2} ({}_0^C D_x^\alpha y_0^T)^2 - \frac{1}{2} (y_0^T)^2 \right) dx. \quad (7.45)$$

Para gerar uma nova função teste, não alteramos todos os coeficientes de uma vez, sendo assim escolhemos um dos coeficientes (a_1, a_2, a_3, a_4) de forma aleatória e o alteramos para $a_i = a_i + \epsilon h$ com $h = 0.3$, onde ϵ é um número aleatório no intervalo uniformemente distribuído $[-1, 1]$. Além disso, escolhemos aleatoriamente o novo valor para o expoente β no intervalo $[0.5, 3]$. Calculamos o valor do funcional desta nova função teste gerada $S[y_1^T]$. Faz-se o processo conforme descrito no algoritmo em n passos e obtém-se a solução aproximada para o problema de fronteira. Para $\alpha = 1$ obtêm-se as soluções para o caso inteiro, a Figura 45 apresenta a solução obtida pelo MMCV, a solução exata $y(x) = \sin(x)$, e a função inicial. A solução encontrada apresenta uma boa concordância com a solução analítica com 10000 passos. A solução encontrada foi:

$$\begin{aligned} y^T(x) = & 0.742027764214533x^{0.844520231028307} + 0.383019938287862x^{1.68904046205661} \\ & - 0.263441500653144x^{2.53356069308492} - 0.040042870589724x^{3.37808092411323} \\ & + 0.015361481534537x^{4.22260115514154} \end{aligned}$$

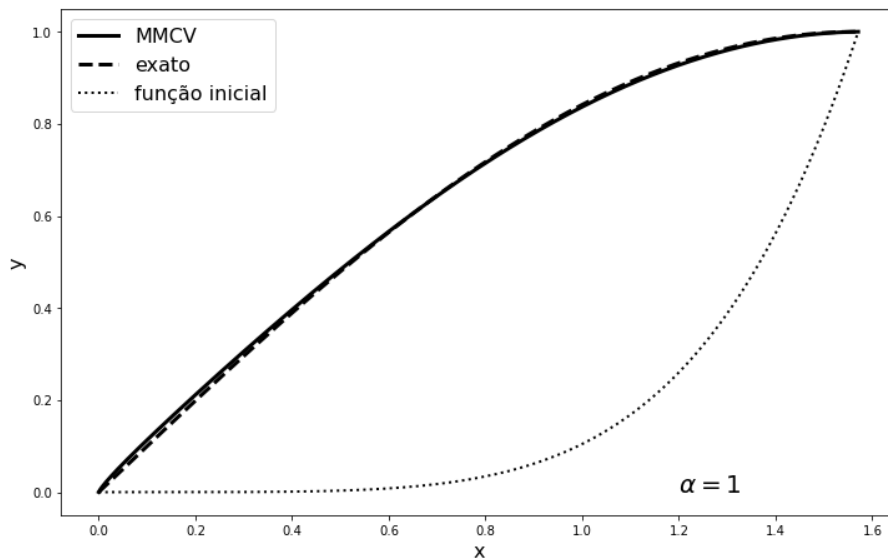


Figura 45: Monte Carlo Variacional para o PVF 7.3.2 com 10000 passos considerando $\alpha = 1$.

Para o caso de derivadas fracionárias, usaremos a ideia da função desvio introduzida anteriormente para verificar o quanto a solução $y^T(x)$ encontrada pelo MMCV difere da solução exata do problema. A solução desvio neste exemplo é dada por:

Definição 7.3.1.

$$desvio(y) = {}_{\frac{\pi}{2}}D_x^\alpha {}_0^CD_x^\alpha y - y. \quad (7.46)$$

A construção dessa função busca criar uma forma de analisar as soluções numéricas encontradas, principalmente quando não temos a solução analítica. Assim, quanto mais a função desvio se aproxima do zero, mais as soluções encontradas tendem a satisfazer a equação à qual se originou o problema.

A Figura 46 apresenta as soluções obtidas pelo MMCV para derivadas de ordem $\alpha = 0.3, 0.5$ e 0.8 , respectivamente. Como no caso de derivadas fracionárias não temos a solução exata para comparar, incluímos nesta figura os gráficos das respectivas funções desvio. Percebe-se uma evolução da solução à medida que o $\alpha \rightarrow 1$ e as funções desvio apresentam comportamentos próximos de zero na maior parte do domínio, com exceção no extremo do intervalo, onde temos o problema do polo da derivada de Riemann-Liouville. É importante lembrar que, como discutimos anteriormente, a função desvio difere de zero, em geral, muito mais do que o erro absoluto e do que o erro relativo. Isto porque a derivada definindo a função erro amplifica qualquer erro presente na solução obtida. No entanto, essa função erro nos permite avaliar para que regiões do domínio a função obtida pelo MMCV mais se aproxima da solução exata do problema.

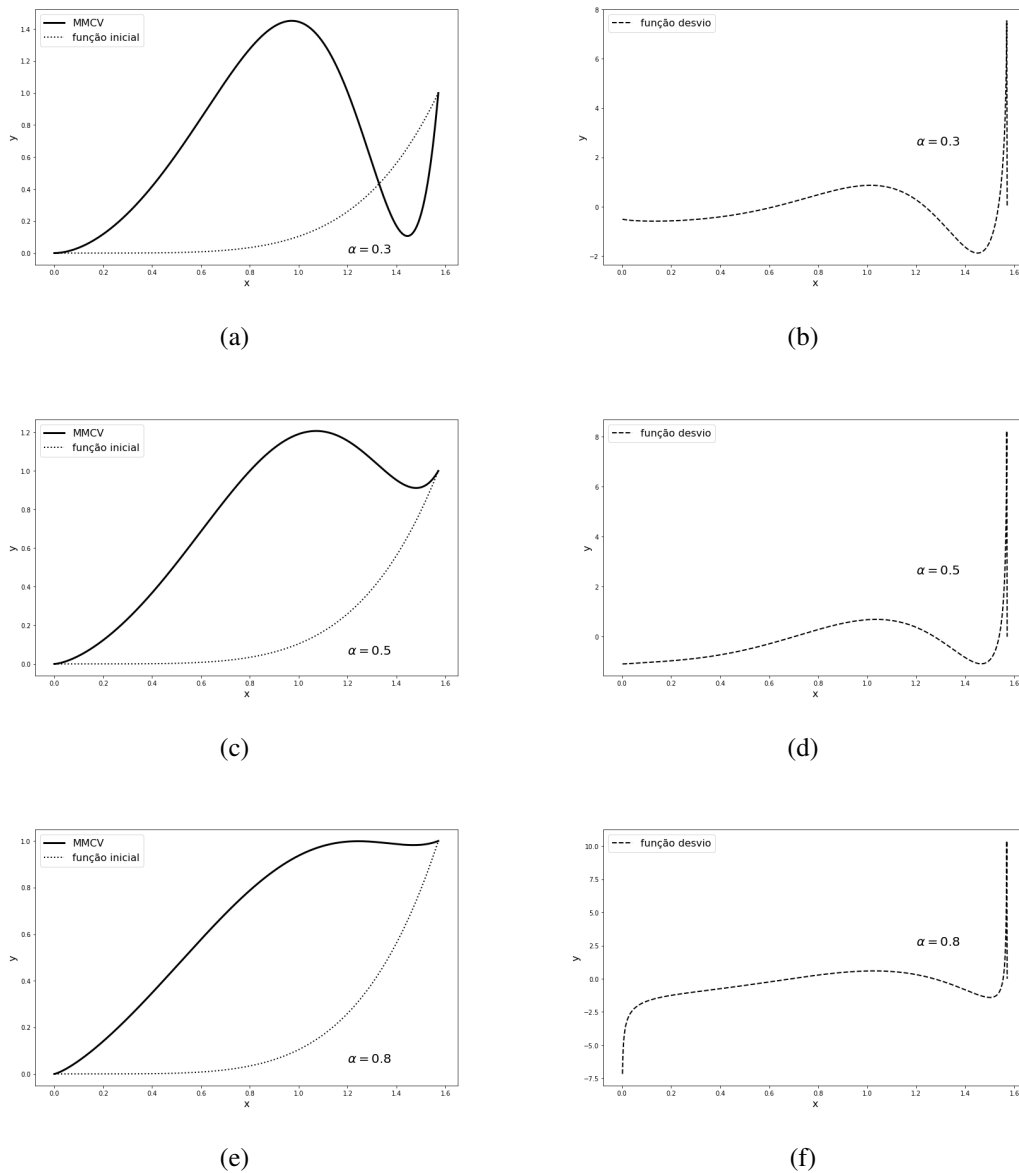


Figura 46: Convergência do MMCV para o PVF 7.3.2 com 10000 passos considerando $\alpha = 0.3$, $\alpha = 0.50$ e $\alpha = 1$.

7.3.3 Exemplo 3

Neste último exemplo, vamos resolver novamente o problema da braquistócrona fracionária, apresentado no subcapítulo anterior, utilizando o novo MMCV. Temos o problema:

$$\begin{cases} S[y] = \int_0^1 \sqrt{\frac{1 + ({}_0^C D^\alpha y)^2}{2g(1-y)}} dx \rightarrow \text{mínimo}, \\ \text{com a condição de fronteira } y(0) = 0, y(1) = 0 \text{ e } 0 < \alpha \leq 1. \end{cases} \quad (7.47)$$

Como discutido anteriormente, a solução analítica para o problema (7.47) quando $\alpha \neq$

1 não é conhecida. A equação de Euler-Lagrange fracionária associada a esse funcional é bem complexa. Vamos primeiro aplicar o algoritmo desenvolvido nesse subcapítulo para obter uma aproximação para a solução analítica (quando $\alpha = 1$). Após, vamos também obter aproximações para a solução do problema quando $\alpha = 0.3, 0.5$ e 0.8 , respectivamente.

Para implementar o MMCV proposto, vamos utilizar a mesma classe de funções teste (7.38) do exemplo 7.3.1, adaptando para os valores fixos da fronteira do problema da braquistócrona. Assim, temos a seguinte função teste para o problema variacional definido em (7.47):

$$\begin{cases} y^T(x) = a_0 + a_1x^\beta + a_2x^{2\beta} + a_3x^{3\beta} + a_4x^{4\beta} + a_5x^{5\beta}, \\ \text{com } a_0 = 0 \text{ e } a_5 = -1 - a_1 - a_2 - a_3 - a_4, \\ \text{que satisfaz a condição de fronteira } y(0) = 1, \quad y(1) = 0. \end{cases} \quad (7.48)$$

Iniciamos com $a_1 = a_2 = a_3 = a_4 = 0$ e $\beta = 1$, assim a função teste inicial é $y_0^T(x) = 1 - x^5$, e tem derivada de Caputo ${}_0^C D_x^\alpha y_0^T = -\frac{\Gamma(6)}{\Gamma(6-\alpha)}x^{5-\alpha}$. Calculamos então o valor do funcional

$$S[y_0^T] = \int_0^1 \sqrt{\frac{1 + ({}_0^C D_x^\alpha y_0^T)^2}{2g(1 - y_0^T)}} dx. \quad (7.49)$$

Para gerar uma nova função teste, não alteramos todos os coeficientes de uma vez, sendo assim escolhemos um dos coeficientes (a_1, a_2, a_3, a_4) de forma aleatória e o alteramos para $a_i = a_i + \epsilon h$ com $h = 1$, e onde ϵ um número aleatório escolhido no intervalo uniformemente distribuído $[-1, 1]$. Também escolhemos aleatoriamente o novo valor para o expoente β no intervalo $[0.1, 0.3]$. Calculamos o valor do funcional desta nova função teste gerada $S[y_1^T]$. Faz-se o processo conforme descrito no algoritmo em n passos e obtém-se uma aproximação para a solução do problema.

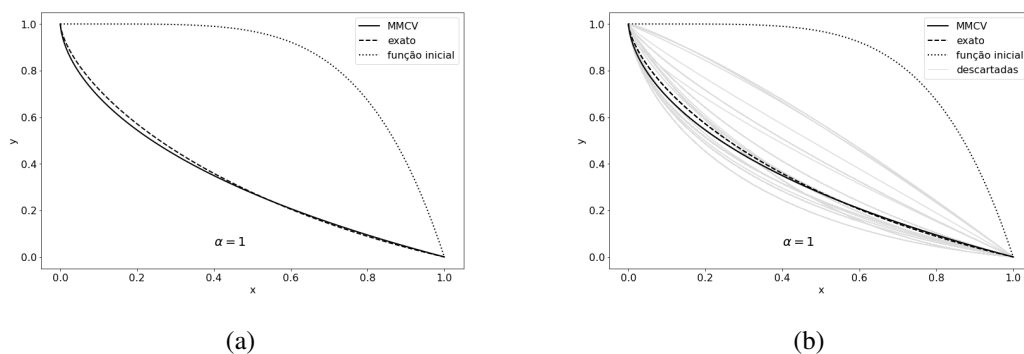


Figura 47: Convergência do MMCV para a braquistócrona com 10000 passos e considerando $\alpha = 1$.

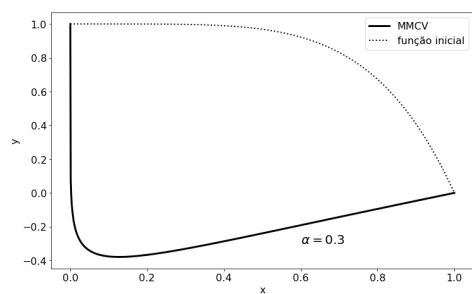
Para $\alpha = 1$ obtém-se uma aproximação para a solução com derivada de ordem inteira.

A Figura 47(a) apresenta a solução obtida pelo MMCV, a solução exata e a função teste inicial. A solução encontrada apresenta uma boa concordância com a solução analítica com 10000 passos. A solução encontrada foi:

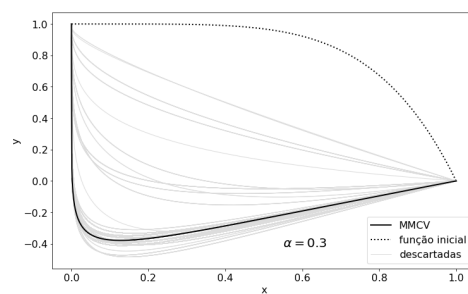
$$y^T(x) = -0.0151396009886435x^{0.18561173523476} + 0.298070500512996x^{0.37122347046952} \\ - 1.73270945361161x^{0.556835205704281} - 0.0288275427819384x^{0.742446940939041} \\ + 0.478606096869201x^{0.928058676173801} + 1.0$$

A Figura 47(b) apresenta o gráfico de todas as funções testes que foram aceitas e depois descartadas no algoritmo. Essas soluções oscilam em torno da solução exata. Finalmente, é importante ressaltar que podemos obter uma solução aproximada ainda melhor aumentando o número de termos na função teste.

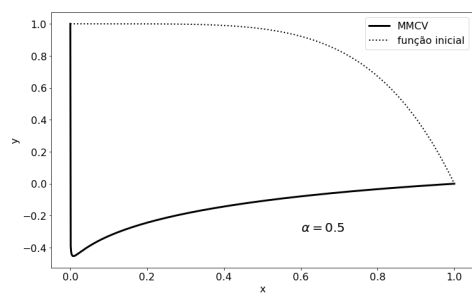
A Figura 48 apresenta as soluções aproximadas para $\alpha = 0.3, 0.5$ e 0.8 , respectivamente. Não temos a solução exata para comparar, mas podemos analisar a evolução das soluções descartadas pelo algoritmo. Note que em todos os casos, as soluções do problema fracionário iniciam com um movimento da partícula de quase queda livre. Este resultado é consistente com a interpretação do efeito de atrito viscoso introduzido pela derivada fracionária [34], conforme discutimos anteriormente. Note ainda o comportamento diferente da solução para $\alpha = 0.5$ (similar ao que obtivemos no subcapítulo anterior), apresentando um ponto de quase quina (sugerindo que a solução pode não ser diferenciável neste ponto) seguido por uma inversão na concavidade da função. É importante ressaltar que estamos aproximando a solução do problema por uma função potência contendo apenas 6 termos. Utilizando mais termos, esperamos obter um resultado mais próximo do obtido no subcapítulo anterior. No entanto, qualitativamente obtemos o mesmo comportamento.



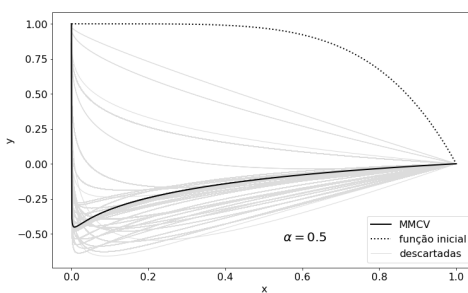
(a)



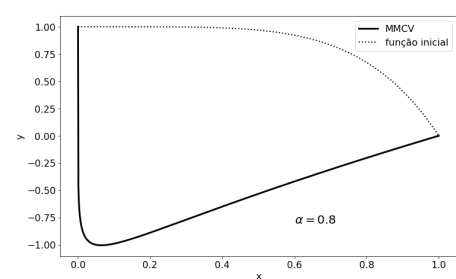
(b)



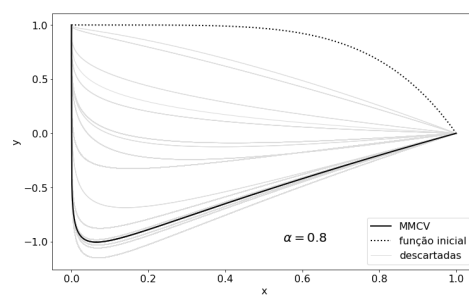
(c)



(d)



(e)



(f)

Figura 48: Monte Carlo Variacional para a braquistócrona com 100000 passos e considerando $\alpha = 0.3$, $\alpha = 0.5$ e $\alpha = 0.8$

8 CONSIDERAÇÕES FINAIS

O Cálculo Fracionário é empregado com sucesso na modelagem de sistemas complexos, descrevendo muitos problemas do mundo real, melhor do que modelos com derivadas de ordem inteira [35, 51, 57, 65, 67, 50]. Baseado nesses resultados, observa-se que equações diferenciais fracionárias podem ser usadas na modelagem matemática de diversos problemas. Neste sentido, faz-se necessário a construção de métodos numéricos eficientes para e determinar a solução de equações fracionárias. Nesta tese estudamos o Método de Monte Carlo [3, 12, 36]. Este método é importante pelo seu caráter universal (pode ser facilmente adaptado para definições diferentes de derivadas fracionárias e para ordens grandes de derivada), e pelo potencial de ser utilizado na resolução de problemas de fronteira, e para obter extremantes de funcionais. No entanto, não foi encontrado na literatura nenhum trabalho que utiliza o Método de Monte Carlo, como o utilizado neste estudo, para resolver equações diferenciais ordinárias fracionárias, tanto em problemas de valor inicial (PVI), como em problemas de valor de fronteira (PVF). Além disso, a aplicação do Método de Monte Carlo no Cálculo Variacional fracionário também não foi explorada ainda na literatura. Mostramos que os algoritmos de Monte Carlo propostos nesta tese permitem resolver todos esses tipos de problemas, fornecendo um método numérico de fácil implementação para obter a solução de problemas não triviais.

Para estudos futuros, pretende-se implementar o Método de Monte Carlo para equações diferenciais parciais fracionárias, e para estudar o problema de fronteiras não retangulares. Melhorar a convergência dos métodos apresentados, por exemplo, introduzindo a noção de análise de sinais com o Python [33]. Em [70] Savitzky e Golay propuseram um método de suavização de dados baseado na aproximação polinomial de mínimos quadra-

dos local. A ideia é usar esse filtro para tratar as soluções encontradas pelo Método de Monte Carlo.

REFERÊNCIAS

- [1] AGRAWAL, O. P. Formulation of Euler–Lagrange equations for fractional variational problems. **Journal of Mathematical Analysis and Applications**, v.272, n.1, p.368–379, 2002.
- [2] AGRAWAL, O. P. Fractional variational calculus and the transversality conditions. **Journal of Physics A: Mathematical and General**, v.39, n.33, p.10375, 2006.
- [3] AKHTAR, M. N.; DURAD, M. H.; AHMED, A. Solving initial value ordinary differential equations by Monte Carlo method. **Proc. IAM**, v.4, p.149–174, 2015.
- [4] ALLEN, L. J. S. **An Introduction to Mathematical Biology**. Pearson/Prentice Hall, 2007.
- [5] ALMEIDA, R.; POOSEH, S.; TORRES, D. F. **Computational methods in the fractional calculus of variations**. Singapore: World Scientific Publishing Company, 2015.
- [6] ALMEIDA, R.; TAVARES, D.; TORRES, D. F. **The variable-order fractional calculus of variations**. Springer, 2019.
- [7] AMES, W. F. **Numerical methods for partial differential equations**. San Diego: Academic press, 2014.
- [8] ATANACKOVIĆ, T.; KONJIK, S.; PILIPOVIĆ, S. Variational problems with fractional derivatives: Euler–Lagrange equations. **Journal of Physics A: Mathematical and Theoretical**, v.41, n.9, p.095201, 2008.
- [9] BALEANU, D.; TRUJILLO, J. J. On exact solutions of a class of fractional Euler–Lagrange equations. **Nonlinear Dynamics**, v.52, p.331–335, 2008.
- [10] BARON, M. E. **The origins of infinitesimal calculus**. Hungary: Elsevier, 2014.
- [11] BARROS, L. C. d.; LOPES, M. M.; PEDRO, F. S.; ESMI, E.; SANTOS, J. P. C. d.; SÁNCHEZ, D. E. The memory effect on fractional calculus: an application in the spread of COVID-19. **Computational and Applied Mathematics**, v.40, n.3, p.1–21, 2021.

- [12] BAUER, W. The monte carlo method. **Journal of the Society for Industrial and Applied Mathematics**, v.6, n.4, p.438–451, 1958.
- [13] **Traité du calcul différentiel et de calcul intégral, avec un traité des différences et des séries**, author=Lacroix, Silvestre-François. Courcier, 1819. 409–410p.
- [14] BOURDIN, L. Existence of a weak solution for fractional Euler–Lagrange equations. **Journal of Mathematical Analysis and Applications**, v.399, n.1, p.239–251, 2013.
- [15] BOYER, C. B. **The history of the calculus and its conceptual development:(The concepts of the calculus)**. United States: Courier Corporation, 1959.
- [16] BOYER, C. B.; MERZBACH, U. C. **História da matemática**. São Paulo: Editora Blucher, 2019.
- [17] BUFFON, G. **Histoire naturelle, generale er particuliere, Supplément 4. Essai d’arithmetique morale**. McGraw-Hill Book Company, Inc, 1777.
- [18] CAMARGO, R. F. **Cálculo fracionário e aplicações**. 2009. Tese — Universidade Estadual de Campinas.
- [19] CAMARGO, R. F.; OLIVEIRA, E. C. d. **Cálculo fracionário**. São Paulo: Livraria da Física, 2015.
- [20] DAVIS, P. J. Leonhard euler’s integral: A historical profile of the gamma function: In memoriam: Milton abramowitz. **The American Mathematical Monthly**, v.66, n.10, p.849–869, 1959.
- [21] DEBNATH, L. A brief historical introduction to fractional calculus. **International Journal of Mathematical Education in Science and Technology**, v.35, n.4, p.487–501, 2004.
- [22] DIETHELM, K. **The analysis of fractional differential equations: An application-oriented exposition using differential operators of Caputo type**. Springer Science & Business Media, 2010.
- [23] DIETHELM, K.; FORD, N. J.; FREED, A. D.; LUCHKO, Y. Algorithms for the fractional calculus: a selection of numerical methods. **Computer methods in applied mechanics and engineering**, v.194, n.6-8, p.743–773, 2005.
- [24] DUNN, W. L.; SHULTIS, J. K. **Exploring monte carlo methods**. Amsterdam: Elsevier, 2011.

- [25] EVANS, G.; BLACKLEDGE, J.; YARDLEY, P. **Numerical methods for partial differential equations**. London: Springer Science & Business Media, 2012.
- [26] EWING, G. M. **Calculus of variations with applications**. Courier Corporation, 1985.
- [27] FERREIRA, L. N.; LAZO, M. J. Método de Monte Carlo Aplicado ao Cálculo Fracionário. **Trends in Computational and Applied Mathematics**, v.23, p.243–255, 2022.
- [28] FERREIRA, L. N.; LAZO, M. J. The Monte Carlo method applied to the fractional SIR model. **REVISTA CEREUS**, v.15, p.92–104, 2023.
- [29] GELFAND, I. M.; SILVERMAN, R. A. et al. **Calculus of variations**. Courier Corporation, 2000.
- [30] GEORGIEV, S. G. **Fractional dynamic calculus and fractional dynamic equations on time scales**. Springer, 2018. v.196.
- [31] GÓMEZ-AGUILAR, J.; ROSALES-GARCÍA, J.; BERNAL-ALVARADO, J.; CÓRDOVA-FRAGA, T.; GUZMÁN-CABRERA, R. Fractional mechanical oscillators. **Revista mexicana de física**, v.58, n.4, p.348–352, 2012.
- [32] GRÜNWALD, A. K. Ueber begrenzte derivationen und deren anwendung. **Z. Math. Phys**, v.12, n.6, p.441–480, 1867.
- [33] HASLWANTER, T. **Hands-on signal analysis with python**. Springer, 2021.
- [34] HERRMANN, R. **Fractional calculus : an introduction for physicists**. Singapore: World Scientific Publishing Co. Pte. Ltd., 2014.
- [35] HILFER, R. **Applications of fractional calculus in physics**. Singapore: World scientific, 2000.
- [36] JE, G. **Random Number Generation and Monte Carlo Methods**. New York: Springer-Verlag, 2003.
- [37] JOHANSSON, R. **Numerical Python : Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib**. Apress Berkeley, CA, 2018.
- [38] JOHNSON, W. P. The curious history of Faà di Bruno's formula. **The American mathematical monthly**, v.109, n.3, p.217–234, 2002.
- [39] KALOS, M. H.; WHITLOCK, P. A. **Monte carlo Methods**. Germany: John Wiley & Sons, 2009.

- [40] KERMACK, W. O.; MCKENDRICK, A. G. A contribution to the mathematical theory of epidemics. **Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character**, v.115, n.772, p.700–721, 1927.
- [41] KILBAS, A. A.; SRIVASTAVA, H. M.; TRUJILLO, J. J. **heory and applications of fractional differential equations**. elsevier, 2006. v.204.
- [42] KIMBALL, W. S. A HISTORY OF THE BRACHISTOCHRONE. **Pi Mu Epsilon Journal**, v.2, n.2, p.57–77, 1955.
- [43] KOLWANKAR, K. M.; GANGAL, A. D. Fractional differentiability of nowhere differentiable functions and dimensions. **Chaos: An Interdisciplinary Journal of Non-linear Science**, v.6, n.4, p.505–513, 1996.
- [44] KOMZSIK, L. **Applied calculus of variations for engineers**. CRC Press, 2019.
- [45] KURODA, L. K. B.; FIGUEIREDO CAMARGO, R. de. Generalização da Modelagem Fracionária. **Proceeding Series of the Brazilian Society of Computational and Applied Mathematics**, v.8, n.1, 2021.
- [46] LAPLACE, P. S. Théorie analytique des probabilités Suppl. **Théorie analytique des probabilités**, 1812.
- [47] LEIBNIZ, G. W. **Sechster Band: 1694–Juni 1696**. Akademie Verlag, 2004.
- [48] LÜCHOW, A. Quantum monte carlo methods. **Wiley Interdisciplinary Reviews: Computational Molecular Science**, v.1, n.3, p.388–402, 2011.
- [49] MACHADO, J. T.; GALHANO, A. M.; TRUJILLO, J. J. On development of fractional calculus during the last fifty years. **Scientometrics**, v.98, p.577–582, 2014.
- [50] MAGIN, R. L. **Fractional calculus in bioengineering**. Begell House Redding, 2006. v.2.
- [51] MAINARDI, F. **Fractional calculus and waves in linear viscoelasticity: an introduction to mathematical models**. Singapore: World Scientific, 2010.
- [52] MAINARDI, F. Why the Mittag-Leffler function can be considered the queen function of the fractional calculus? **Entropy**, v.22, n.12, p.1359, 2020.
- [53] MCMILLAN, W. L. Ground state of liquid He 4. **Physical Review**, v.138, n.2A, p.A442, 1965.
- [54] MESTERTON-GIBBONS, M. **A primer on the calculus of variations and optimal control theory**. American Mathematical Soc., 2009. v.50.

- [55] METROPOLIS, N.; ROSENBLUTH, A. W.; ROSENBLUTH, M. N.; TELLER, A. H.; TELLER, E. Equation of state calculations by fast computing machines. **The journal of chemical physics**, v.21, n.6, p.1087–1092, 1953.
- [56] METROPOLIS, N.; ULAM, S. The monte carlo method. **Journal of the American statistical association**, v.44, n.247, p.335–341, 1949.
- [57] METZLER, R.; KLAFTER, J. The random walk's guide to anomalous diffusion: a fractional dynamics approach. **Physics reports**, v.339, n.1, p.1–77, 2000.
- [58] MILLER, K. S.; ROSS, B. **An introduction to the fractional calculus and fractional differential equations**. Wiley, 1993.
- [59] MITTAG-LEFFLER, G. M. Sur la nouvelle fonction $E_\alpha(x)$. **CR Acad. Sci. Paris**, v.137, n.2, p.554–558, 1903.
- [60] OLDHAM, K.; SPANIER, J. **The fractional calculus theory and applications of differentiation and integration to arbitrary order**. New York: Elsevier, 1974.
- [61] ORTIGUEIRA, M. D.; Tenreiro Machado, J. What is a fractional derivative? **Journal of Computational Physics**, v.293, p.4–13, 2015. Fractional PDEs.
- [62] PODLUBNY, I. Geometric and physical interpretation of fractional integration and fractional differentiation. **Fractional Calculus and Applied Analysis**, v.5, p.367–386, 2002.
- [63] RIEWE, F. Nonconservative lagrangian and hamiltonian mechanics. **Physical Review E**, v.53, n.2, p.1890, 1996.
- [64] RIEWE, F. Mechanics with fractional derivatives. **Physical Review E**, v.55, n.3, p.3581, 1997.
- [65] RIVERO, M.; TRUJILLO, J. J.; VÁZQUEZ, L.; VELASCO, M. P. Fractional dynamics of populations. **Applied Mathematics and Computation**, v.218, n.3, p.1089–1095, 2011.
- [66] ROSS, B. The development of fractional calculus 1695–1900. **Historia mathematica**, v.4, n.1, p.75–89, 1977.
- [67] SABATIER, J.; AGRAWAL, O. P.; MACHADO, J. T. **Advances in fractional calculus**. The Netherlands: Springer, 2007. v.4.
- [68] SAGAN, H. **Introduction to the Calculus of Variations**. Courier Corporation, 2012.

- [69] SAMKO, S. G.; KILBAS, A. A.; MARICHEV, O. I. et al. **Fractional integrals and derivatives**. Yverdon: Gordon and Breach Science Publishers, 1993. v.1.
- [70] SCHAFER, R. W. What is a Savitzky-Golay filter?[lecture notes]. **IEEE Signal processing magazine**, v.28, n.4, p.111–117, 2011.
- [71] SHLOMO M, M. S. **Applications of Monte Carlo Method in Science and Engineering**. London: IntechOpen, 2011.
- [72] SOLOMON, H. **Geometric probability**. Philadelphia: SIAM, 1978.
- [73] TAVARES, C.; LAZO, M. Dynamic Systems with Fractional Derivatives Applied to Interagent Populations Problems. **Trends in Computational and Applied Mathematics**, 2022.
- [74] TEODORO, G. S. **Derivadas fracionárias: tipos e critérios de validade**. 2019. Tese — Universidade Estadual de Campinas.
- [75] TEODORO, G. S.; MACHADO, J. T.; DE OLIVEIRA, E. C. A review of definitions of fractional derivatives and other operators. **Journal of Computational Physics**, v.388, p.195–208, 2019.
- [76] THOMAS, J. W. **Numerical partial differential equations: finite difference methods**. New York: Springer Science & Business Media, 2013. v.22.
- [77] TORRES, D. F.; MALINOWSKA, A. B. **Introduction to the fractional calculus of variations**. World Scientific Publishing Company, 2012.
- [78] ULAM, S. M. **Adventures of a Mathematician**. Berkeley: Univ of California Press, 1991.
- [79] VANDERPLAS, J. **Python data science handbook: Essential tools for working with data**. O'Reilly Media, Inc., 2016.
- [80] WEILBEER, M. **Efficient numerical methods for fractional differential equations and their analytical background**. Germany: Papierflieger Clausthal-Zellerfeld, 2006.
- [81] WEINSTOCK, R. **Calculus of variations**. New York: Courier Corporation, 2012.
- [82] ZETTLI, N. **Quantum mechanics: concepts and applications**. John Wiley & Sons, 2009.

A FUNÇÕES ESPECIAIS

Neste anexo, apresentam-se as definições de algumas funções especiais que são mencionadas neste texto, importantes na análise e desenvolvimento do CF.

A.1 Função Gamma

A função Gamma começou a ser desenvolvida em 1729 pelo matemático Leonhard Euler (1707-1783) de acordo com [20]. A definição da função $\Gamma(z)$ é dada por

$$\Gamma(z) = \int_0^{\infty} e^{-t} t^{z-1} dt \quad (\text{A.1})$$

onde $\Re(z) > 0$ generaliza a função fatorial para números não inteiros (para $z = m + 1$ inteiros, temos $\Gamma(m + 1) = m!$).

A.2 Função Beta

A função Beta é definida por:

$$B(z, w) = \int_0^1 t^{z-1} (1 - t)^{w-1} dt, \quad (\text{A.2})$$

onde $\Re(z) > 0$ e $\Re(w) > 0$. A função Gamma e a função Beta podem ser relacionadas por:

$$B(z, w) = \frac{\Gamma(z)\Gamma(w)}{\Gamma(z + w)}. \quad (\text{A.3})$$

Para provar essa relação, basta fazer o produto de duas funções Gamma, fazer uma mudança de variável na integral dupla resultante e utilizar as coordenadas polares.

A.3 Função de Gel'fand-Shilov

A função de Gel'fand-Shilov tem sua importância no CF por sua aplicação, juntamente com a convolução, na definição de integral fracionária. A função é dada por:

$$\Phi_{\alpha}(x) = \begin{cases} \frac{x^{\alpha-1}}{\Gamma(\alpha)} & \text{se } x > 0 \\ 0 & \text{se } x \leq 0. \end{cases} \quad (\text{A.4})$$

Usando a convolução de Laplace obtemos uma importante propriedade

$$\Phi_{\alpha}(x) * \Phi_{\beta}(x) = \Phi_{\alpha+\beta}(x). \quad (\text{A.5})$$

B O TEOREMA DE CAUCHY PARA INTEGRAIS REPE- TIDAS

Neste anexo, vamos analisar a demonstração do teorema de Cauchy para integrais repetidas que pode ser encontrada em diversos livros-texto de Cálculo. Apesar disso, vamos apresentar aqui uma demonstração desse teorema, pois ele desempenha um papel fundamental em diversas formulações do Cálculo Fracionário. Precisamos, antes, relembrar outro importante teorema proposto por Leibniz:

Teorema B.0.1 (Teorema de Leibniz para a derivada de uma integral). *Seja $f(x, t)$ uma função de duas variáveis diferenciável em x , e seja $g(x)$ e $h(x)$ funções diferenciáveis. Então temos:*

$$\frac{d}{dx} \int_{g(x)}^{h(x)} f(x, t) dt = \int_{g(x)}^{h(x)} \frac{\partial}{\partial x} f(x, t) dt + f(x, h(x)) \frac{dh(x)}{dx} - f(x, g(x)) \frac{dg(x)}{dx}. \quad (\text{B.1})$$

Prova: Uma possível demonstração desse teorema segue de identificar a integral com uma função de 3 variáveis $u(x, h, g) = \int_{g(x)}^{h(x)} f(x, t) dt$. Tomando a derivada total da função $u(x, h, g)$ em relação à x , obtemos diretamente (B.1) $\square$.

Para demonstrar o Teorema de Cauchy para integrais repetidas vamos utilizar um caso particular da (B.1) quando $h(x) = x$ e $g(x) = a$. Neste caso a (B.1) se reduz à:

$$\frac{d}{dx} \int_a^x g(x, t) dt = \int_a^x \frac{\partial}{\partial x} g(x, t) dt + g(x, x). \quad (\text{B.2})$$

Vamos enunciar agora o Teorema de Cauchy para integrais repetidas:

Teorema B.0.2 (Teorema de Cauchy para integrais repetidas). *Seja f uma função integrável $n \in \mathbb{N}^*$ vezes no intervalo $[a, x] \in \mathbb{R}$. Então vale a relação:*

$$\int_a^x f(t) (dt)^n \equiv \int_a^x dt_n \int_a^{t_n} \cdots \int_a^{t_2} f(t_1) dt_1 = \frac{1}{\Gamma(n)} \int_a^x (x-t)^{n-1} f(t) dt, \quad (\text{B.3})$$

onde Γ é a função Gamma de Euler.

Prova: Para provar (B.3) vamos primeiro mostrar que a primeira integral da função f

pode ser escrita na forma

$$\int_a^x f(t)dt = \frac{1}{(n-1)!} \frac{d^{n-1}}{dx^{n-1}} \int_a^x (x-t)^{n-1} f(t)dt. \quad (\text{B.4})$$

Calculando sucessivamente as derivadas da integral do lado direito da (B.4) obtemos:

$$\begin{aligned} \int_a^x f(t)dt &= \frac{1}{(n-1)!} \frac{d^{n-1}}{dx^{n-1}} \int_a^x (x-t)^{n-1} f(t)dt \\ &= \frac{1}{(n-1)!} \frac{d^{n-2}}{dx^{n-2}} \int_a^x \frac{\partial}{\partial x} (x-t)^{n-1} f(t)dt + (x-x)^{n-1} f(x) \\ &= \frac{1}{(n-2)!} \frac{d^{n-2}}{dx^{n-2}} \int_a^x (x-t)^{n-2} f(t)dt \\ &= \frac{1}{(n-2)!} \frac{d^{n-3}}{dx^{n-3}} \int_a^x \frac{\partial}{\partial x} (x-t)^{n-2} f(t)dt + (x-x)^{n-2} f(x) \\ &= \frac{1}{(n-3)!} \frac{d^{n-3}}{dx^{n-3}} \int_a^x (x-t)^{n-3} f(t)dt = \dots \\ &= \frac{1}{(n-j)!} \frac{d^{n-j}}{dx^{n-j}} \int_a^x (x-t)^{n-j} f(t)dt, \end{aligned} \quad (\text{B.5})$$

onde $1 \leq j \leq n$ é um inteiro qualquer. Escolhendo $j = n$ e lembrando que $0! = 1$ e usando a definição padrão $\frac{d^0}{dx^0} \equiv 1$ obtemos a (B.4).

Para provar a (B.3) basta integrarmos repetidas vezes a (B.4). Integrando uma vez obtemos:

$$\begin{aligned} \int_a^x dt_2 \int_a^{t_2} f(t_1)dt_1 &= \int_a^x \frac{1}{(n-1)!} \frac{d^{n-1}}{dt_2^{n-1}} \int_a^{t_2} (t_2-t)^{n-1} f(t)dt dt_2 \\ &= \frac{1}{(n-1)!} \left(\frac{d^{n-2}}{dx^{n-2}} \int_a^x (x-t)^{n-1} f(t)dt - \frac{d^{n-2}}{da^{n-2}} \int_a^a (a-t)^{n-1} f(t)dt \right) \\ &= \frac{1}{(n-1)!} \frac{d^{n-2}}{dx^{n-2}} \int_a^x (x-t)^{n-1} f(t)dt, \end{aligned} \quad (\text{B.6})$$

integrando duas vezes obtemos:

$$\begin{aligned} \int_a^x dt_3 \int_a^{t_3} dt_2 \int_a^{t_2} f(t_1)dt_1 &= \int_a^x \frac{1}{(n-1)!} \frac{d^{n-2}}{dt_3^{n-2}} \int_a^{t_3} (t_3-t)^{n-1} f(t)dt dt_3 \\ &= \frac{1}{(n-1)!} \left(\frac{d^{n-3}}{dx^{n-3}} \int_a^x (x-t)^{n-1} f(t)dt - \frac{d^{n-3}}{da^{n-3}} \int_a^a (a-t)^{n-1} f(t)dt \right) \\ &= \frac{1}{(n-1)!} \frac{d^{n-3}}{dx^{n-3}} \int_a^x (x-t)^{n-1} f(t)dt, \end{aligned} \quad (\text{B.7})$$

e finalmente, após integrar $n-1$ vezes a (B.4) e identificar $\Gamma(n) = (n-1)!$ obtemos a

(B.3) $\square$.

C ALGORITMOS

Neste anexo, vamos disponibilizar alguns algoritmos resumidos que foram construídos durante a elaboração desta tese. A linguagem e as ferramentas utilizadas podem ser encontradas em [79, 37].

C.1 Integral Fracionária

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Tue Jan 14 11:44:56 2020
5 @author: luverci
6 """
7 import random as random_number
8 import matplotlib.pyplot as plt
9 import numpy as np
10 from scipy.special import gamma
11 ##### MMC PARA INTEGRAIS DE RIEMANN-LIOUVILE (Esquerda e
12     Direita)
13 alpha = 0.3 # ordem da derivada
14 a = 0 # limite inferior do domínio
15 b = 10.0 # limite superior do domínio
16 m= 200 # número de partições iguais do intervalo [a,b]
17 n = 10 # número médio de realizações em cada partição
18 D = float((b-a)/m) # comprimento das partições
19 #####
20 # Rotina para calcular integria no intervalo [a,b] usando MC, onde 0 <
21     c < m é a c-ésima partição do intervalo [a,b_máximo],
22 # D=float((b_máximo-a)/m) é comprimento da partição, f é a função, e n
23     é o número médio de realizações em cada partição.
24 def MCint(f, a, b, c, n): #integral à esquerda
25     s = 0
26     ran=c*n # número total de realizações no intervalo de integração [a
27     ,b]
28     for i in range(ran):
```

```

25     x = random_number.uniform(a, b)
26     j=int((x-a)/D) # mudança da variável real x para a variável
    inteira j
27     s += f(b,x,y[j])
28     I = (float(b-a)/(ran))*s # Valor da integração por MC
29     return I
30 def MCintD(f, a, b, c, n): #integral à direita
31     s = 0
32     ran=c*n # número total de realizações no intervalo de integração [a
    ,b]
33     for i in range(ran):
34         x = random_number.uniform(a, b)
35         j=int((b-x)/D) # mudança da variável real x para a variável
    inteira j
36         s += f(a,x,y[j])
37         I = (float(b-a)/(ran))*s # Valor da integração por MC
38         return I
39 #####
40 # Definição da função a ser integrada junto com o kernel da integral
    fracionária
41 beta = 1.0
42 def f(xb,x,y): # Esquerda
43     return (1/gamma(alpha))*x**beta*(xb-x)**(alpha-1)
44
45 def fD(xb,x,y): #Direita
46     return (1/gamma(alpha))*(b-x)**beta*(x-xb)**(alpha-1)
47 #####
48 # Cálculo da integral
49 y=np.zeros(m+1) # função discretizada, com domínio no conjunto dos
    inteiros, que receberá o resultado da integral
50 for k in range(1,m+1): # rotina que controli a função y (integral)
51     auxsol = MCint(f,a,a+k*D,k,n)
52     y[k]= auxsol
53 # Cálculo da integral
54 yD=np.zeros(m+1) # função discretizada, com domínio no conjunto dos
    inteiros, que receberá o resultado da integral
55 for k in range(1,m+1): # rotina que constrói a função y (integral)
56     auxsol = MCintD(fD,b-k*D,b,k,n)
57     yD[k]= auxsol
58 #####
59 # Rotina que cria a função discretizada h(x) com domínio nos reais (
    intervalo [a,b]) à partir da função y
60 def h(x):
61     aux=int((x-a)/D)#leva x do intervalo [a,b] ao número inteiro aux no
    intervalo de 0 até m
62     return y[aux]

```

```

63 # Rotina que cria a função discretizada h(x) com domínio nos reais (
    intervalo [a,b]) à partir da função y
64 def hD(x):
65     aux=int((x-a)/D) #leva x do intervalo [a,b] ao número inteiro aux no
        intervalo de 0 até m
66     return yD[m-aux] #montamos a função do fim para o início
67 #####
68 # Rotina que constrói vetores para fazer o gráfico da função integrada
69 xs = [a+D*(i) for i in range(m+1)] #discretizando o intervalo [a,b]
70 ys = [h(x) for x in xs] # vetor criado à partir da função integrada
71 ysD = [hD(x) for x in xs] # vetor criado à partir da função integrada
72 xf = [gamma(beta+1)*x**(alpha+beta)/gamma(alpha+beta+1) for x in xs]
73 xfD = [gamma(beta+1)*(b-x)**(alpha+beta)/gamma(alpha+beta+1) for x in
        xs]
74 #### Rotina com os erros (desvios quadráticos)
75 er1 = [gamma(beta+1)*x**(alpha+beta)/gamma(alpha+beta+1)-3*(abs(gamma
        (2*alpha-1)*gamma(2*beta+1)*x**(2*alpha+2*beta-1)/(gamma(alpha)**2*
        gamma(2*alpha+2*beta))- (gamma(beta+1)*x**(alpha+beta)/gamma(alpha+
        beta+1))**2)/(n*int((x-a)/D))**0.5) for x in xs]
76 er2 = [gamma(beta+1)*x**(alpha+beta)/gamma(alpha+beta+1)+3*(abs(gamma
        (2*alpha-1)*gamma(2*beta+1)*x**(2*alpha+2*beta-1)/(gamma(alpha)**2*
        gamma(2*alpha+2*beta))- (gamma(beta+1)*x**(alpha+beta)/gamma(alpha+
        beta+1))**2)/(n*int((x-a)/D))**0.5) for x in xs]
77 er1D = [gamma(beta+1)*(b-x)**(alpha+beta)/gamma(alpha+beta+1)-3*(abs(
        gamma(2*alpha-1)*gamma(2*beta+1)*(b-x)**(2*alpha+2*beta-1)/(gamma(
        alpha)**2*gamma(2*alpha+2*beta))- (gamma(beta+1)*(b-x)**(alpha+beta)
        /gamma(alpha+beta+1))**2)/(n*int((b-x)/D))**0.5) for x in xs]
78 er2D = [gamma(beta+1)*(b-x)**(alpha+beta)/gamma(alpha+beta+1)+3*(abs(
        gamma(2*alpha-1)*gamma(2*beta+1)*(b-x)**(2*alpha+2*beta-1)/(gamma(
        alpha)**2*gamma(2*alpha+2*beta))- (gamma(beta+1)*(b-x)**(alpha+beta)
        /gamma(alpha+beta+1))**2)/(n*int((b-x)/D))**0.5) for x in xs]
79 ##### Gráficos#####
80 ####Dimensões das Figuras
81 fig, ax1 = plt.subplots(figsize=(12, 8))
82 left, bottom, width, height = [0.2, 0.5, 0.35, 0.35]
83 ax2 = fig.add_axes([left, bottom, width, height])
84 ax1.plot(xs,xf,'k--', linewidth=3)
85 ax1.plot(xs,ys,'k', linewidth=3)
86 ax1.legend(['exato', 'MMC'],loc='lower right', fontsize=24)
87 ax1.text(4.5,1,f'$\\alpha=${alpha:.1f}, i={n:.0f}', fontsize=24)
88 ax1.tick_params(axis='both', which='major', labelsize=24)
89 ax1.tick_params(axis='both', which='minor', labelsize=24)
90 ax1.fill_between(xs, er1, er2, alpha=.4, color = (.5, .5, .5), label='
        MC uncertainty')
91 ax1.set_ylim([0, 40])
92 ax2.plot(xs,xfD,'k--', linewidth=2)
93 ax2.plot(xs,ysD,'k', linewidth=2)

```

```

94 ax2.legend(['exato', 'MMC'],loc='upper right', fontsize=24)
95 ax2.fill_between(xs, er1D, er2D, alpha=.4, color = (.5, .5, .5), label
    ='MC uncertainty')
96 ax2.tick_params(axis='both', which='major', labelsize=24)
97 ax2.tick_params(axis='both', which='minor', labelsize=24)
98 ax2.set_ylim([0, 40])
99 plt.show()

```

Listing C.1: Integral fracionária

C.2 Derivada de Riemann-Liouville

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  """
4  Created on Thu Jan 16 09:47:53 2020
5  @author: luverci
6  """
7  import random as random_number
8  import matplotlib.pyplot as plt
9  import numpy as np
10 from scipy.special import gamma
11 alpha = 0.3 # ordem da derivada
12 a = 0 # limite inferior do domínio
13 b = 10.0 # limite superior do domínio
14 m = 200 # número de partições iguais do intervalo [a,b]
15 n = 100 # número médio de realizações em cada partição
16 D=float((b-a)/m) # comprimento das partições
17 # Rotina para calcular integria no intervalo [a,b] usando MC, onde 0 <
    c < m é a c-ésima partição do intervalo [a,b_máximo],
18 # D=float((b_máximo-a)/m) é comprimento da partição, f é a função, e n
    é o número médio de realizações em cada partição.
19 def MCInt(f, a, b, c, n): #integral à esquerda
20     s = 0
21     ran=c*n # número total de realizações no intervalo de integração [a
    ,b]
22     for i in range(ran):
23         x = random_number.uniform(a, b)
24         s += f(b,x)
25     I = (float(b-a)/(ran))*s # Valor da integração por MC
26     return I
27 def MCIntD(f, a, b, c, n): #integral à direita
28     s = 0
29     ran=c*n # número total de realizações no intervalo de integração [a
    ,b]
30     for i in range(ran):
31         x = random_number.uniform(a, b)
32         s += f(a,x)

```

```

33     I = (float(b-a)/(ran))*s # Valor da integração por MC
34     return I
35 # Definição da função a ser integrada junto com o kernel da integral
    fracionária
36 beta = 1.0
37 def f(xb,x):
38     return (1/gamma(1-alpha))*(x**beta)*(xb-x)**(-alpha)
39 def fD(xa,x):
40     # return (1/gamma(1-alpha))*(cos(x))*(x-xa)**(-alpha)
41     return (1/gamma(1-alpha))*((b-x)**(beta))*(x-xa)**(-alpha)
42 # Definição da função a ser integrada junto com o kernel da integral
    fracionária
43 beta2 = 1.0
44 def f2D(xa,x):
45     # return (1/gamma(1-alpha))*(cos(x))*(x-xa)**(-alpha)
46     return (1/gamma(1-alpha))*((b-x)**(beta2))*(x-xa)**(-alpha)
47 # Cálculo da integral
48 y=np.zeros(m+1) # função discretizada, com domínio no conjunto dos
    inteiros, que receberá o resultado da derivada
49 for k in range(1,m+1): # rotina que controli a função y (integral)
50     auxsol = MCint(f,a,a+k*D,k,n)
51     y[k]= auxsol
52 # Cálculo da integral
53 yD=np.zeros(m+1) # função discretizada, com domínio no conjunto dos
    inteiros, que receberá o resultado da integral
54 for k in range(1,m+1): # rotina que controli a função y (integral)
55     auxsol = MCintD(fD,b-k*D,b,k,n)
56     yD[k]= auxsol
57 dy=np.zeros(m+1)
58 dyD=np.zeros(m+1)
59 for k in range(1,m):
60     dy[k]=-(y[k]-y[k+1])/(D)
61     dyD[k]=-(yD[k]-yD[k+1])/(D)
62 dy[m]=dy[m-1]
63 dyD[m]=dyD[m-1]
64 # Rotina que cria a função discretizada h(x) com domínio nos reais (
    intervalo [a,b]) à partir da função y
65 def h(x):
66     aux=int((x-a)/(D)) #leva x do intervalo [a,b] ao número inteiro aux
    no intervalo de 0 até m
67     return dy[aux]
68 # Rotina que cria a função discretizada h(x) com domínio nos reais (
    intervalo [a,b]) à partir da função y
69 def hD(x):
70     aux=int((x-a)/(D)) #leva x do intervalo [a,b] ao número inteiro aux
    no intervalo de 0 até m
71     return dyD[m-aux] #montamos a função do fim para o início

```

```

72 # Rotina que constrói vetores para fazer o gráfico da função integrada
73 xs = [a+D*(i) for i in range(m+1)] #discretizando o intervalo [a,b]
74 ys = [h(x) for x in xs] # vetor criado à partir da função integrada
75 ysD = [hD(x) for x in xs] # vetor criado à partir da função integrada
76 # Rotina com as soluções exatas
77 xf = [gamma(beta+1)*x**(-alpha+beta)/gamma(-alpha+beta+1) for x in xs]
78 xfD = [gamma(beta+1)*(b-x)**(-alpha+beta)/gamma(-alpha+beta+1) for x in
        xs]
79 ##### Gráficos#####
80 ###Dimensões das Figuras
81 fig, ax1 = plt.subplots(figsize=(12, 8))
82 left, bottom, width, height = [0.15, 0.55, 0.3, 0.3]
83 ax2 = fig.add_axes([left, bottom, width, height])
84 ax1.plot(xs,xf,'k--', linewidth=2)
85 ax1.plot(xs,ys,'k', linewidth=2)
86 ax1.legend(['exato', 'MMC'],loc='lower right', fontsize=16)
87 ax1.set_ylim([0, 6])
88 ax1.text(4.5,0.5,f'$\\alpha=${alpha:.1f}, i={n:.0f}', fontsize=20)
89 ax1.tick_params(axis='both', which='major', labelsize=20)
90 ax1.tick_params(axis='both', which='minor', labelsize=20)
91 ax2.plot(xs,xfD,'k--', linewidth=2)
92 ax2.plot(xs,ysD,'k', linewidth=2)
93 ax2.set_ylim([0, 6])
94 ax2.legend(['exato', 'MMC'],loc='upper right', fontsize=10)
95 #plt.savefig('RL10000.png', format='png')
96 plt.show()

```

Listing C.2: Derivada de Riemann-Liouville

C.3 Derivada de Caputo

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Thu Jan 16 09:47:53 2020
5
6 @author: luverci
7 """
8 import random as random_number
9 import matplotlib.pyplot as plt
10 import numpy as np
11 from scipy.special import gamma
12 alpha = 0.3 # ordem da derivada
13 a = 0.0 # limite inferior do domínio
14 b = 10.0 # limite superior do domínio
15 m=200 # número de partições iguais do intervalo [a,b]
16 n = 100 # número médio de realizações em cada partição
17 D=float((b-a)/m) # comprimento das partições

```

```

18 # Rotina para calcular integria no intervalo [a,b] usando MC, onde  $0 < c < m$  é a c-ésima partição do intervalo [a,b_máximo],
19 #  $D=\text{float}((b\_máximo-a)/m)$  é comprimento da partição, f é a função, e n é o número médio de realizações em cada partição.
20 def MCint(f, a, b, c, n): #integral à esquerda
21     s = 0
22     ran=c*n # número total de realizações no intervalo de integração [a,b]
23     for i in range(ran):
24         x = random_number.uniform(a, b)
25         s += f(b,x)
26     I = (float(b-a)/(ran))*s # Valor da integração por MC
27     return I
28 def MCintD(f, a, b, c, n): #integral à direita
29     s = 0
30     ran=c*n # número total de realizações no intervalo de integração [a,b]
31     for i in range(ran):
32         x = random_number.uniform(a, b)
33         s += f(a,x)
34     I = (float(b-a)/(ran))*s # Valor da integração por MC
35     return I
36 # Definição da função a ser integrada junto com o kernel da integral fracionária
37 beta = 1.0
38 def f(xb,x):
39     # return (1/gamma(1-alpha))*(cos(x))*(xb-x)**(-alpha)
40     return (1/gamma(1-alpha))*(beta*x**(beta-1))*(xb-x)**(-alpha)
41
42 def fD(xa,x):
43     # return (1/gamma(1-alpha))*(cos(x))*(x-xa)**(-alpha)
44     return (-1/gamma(1-alpha))*(-beta*(b-x)**(beta-1))*(x-xa)**(-alpha)
45 # Cálculo da integral
46 y=np.zeros(m+1) # função discretizada, com domínio no conjunto dos inteiros, que receberá o resultado da integral
47 for k in range(1,m+1): # rotina que controí a função y (integral)
48     auxsol = MCint(f,a,a+k*D,k,n)
49     y[k]= auxsol
50     # Cálculo da integral
51 yD=np.zeros(m+1) # função discretizada, com domínio no conjunto dos inteiros, que receberá o resultado da integral
52 for k in range(1,m+1): # rotina que controí a função y (integral)
53     auxsol = MCintD(fD,b-k*D,b,k,n)
54     yD[k]= auxsol
55 # Rotina que cria a função discretizada h(x) com domínio nos reais ( intervalo [a,b]) à partir da função y
56 def h(x):

```

```

57     aux=int((x-a)/D)#leva x do intervalo [a,b] ao número inteiro aux no
        intervalo de 0 até m
58     return y[aux]
59 # Rotina que cria a função discretizada h(x) com domínio nos reais (
        intervalo [a,b]) à partir da função y
60 def hD(x):
61     aux=int((x-a)/D)#leva x do intervalo [a,b] ao número inteiro aux no
        intervalo de 0 até m
62     return yD[m-aux] #montamos a função do fim para o início
63 # Rotina que controli vetores para fazer o gráfico da função integrada
64 xs = [a+D*(i) for i in range(m+1)] #discretizando o intervalo [a,b]
65 ys = [h(x) for x in xs] # vetor criado à partir da função integrada
66 ysD = [hD(x) for x in xs] # vetor criado à partir da função integrada
67 # Rotina com as soluções exatas
68 xf = [gamma(beta+1)*x**(-alpha+beta)/gamma(-alpha+beta+1) for x in xs]
69 xfD = [gamma(beta+1)*(b-x)**(-alpha+beta)/gamma(-alpha+beta+1) for x in
        xs]
70 ##### Graficos#####
71 ###Dimensoes das Figuras
72 fig, ax1 = plt.subplots(figsize=(12, 8))
73 left, bottom, width, height = [0.15, 0.55, 0.3, 0.3]
74 ax2 = fig.add_axes([left, bottom, width, height])
75 ax1.plot(xs,xf,'k--', linewidth=2)
76 ax1.plot(xs,ys,'k', linewidth=2)
77 ax1.legend(['exato', 'MMC'],loc='lower right', fontsize=16)
78 ax1.text(4.5,0.5,f'$\\alpha=${alpha:.1f}, i={n:.0f}', fontsize=20)
79 ax1.tick_params(axis='both', which='major', labelsize=20)
80 ax1.tick_params(axis='both', which='minor', labelsize=20)
81 ax2.plot(xs,xfD,'k--', linewidth=2)
82 ax2.plot(xs,ysD,'k', linewidth=2)
83 ax2.set_ylim([0, 5])
84 ax2.legend(['exato', 'MMC'],loc='upper right', fontsize=10)
85 #plt.savefig('IntCapOrdens1.png', format='png')
86 plt.show()

```

Listing C.3: Derivada de Caputo

C.4 PVI de Caputo com $f(x, y) = x^\beta$

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Tue Nov 22 11:20:13 2022
5 @author: luverci
6 """
7 #Progrma para calcular o PVI Dy**alpha=x**beta com y[0]=0.
8 import random as random_number
9 import matplotlib.pyplot as plt

```

```

10 import numpy as np
11 from scipy.special import gamma
12 #####
13 a =0; b = 2; m=200; n = 100
14 #####
15 def MCint1(f, a, b, c, n):
16     s = 0
17     ran=c*n
18     for i in range(ran):
19         x = random_number.uniform(a, b)
20         j=int((x-a)/D)
21         s += f1(b,x,y1[j])
22     I = (float(b-a)/(ran))*s
23     return I
24 #####
25 alpha =0.3
26 beta=2
27 #####
28 def f1(xb,x,y): #kernel
29     return (1/gamma(alpha))*x**beta*(xb-x)**(alpha-1)
30 y1=np.zeros(m+1)
31 D=float((b-a)/m)
32 y1[0]=0 #condição inicial
33 for k in range(1,m+1):
34     auxsol = MCint1(f1,a,a+k*D,k,n)
35     y1[k]=y1[0] + auxsol
36 #####
37 def h1(x):
38     aux=int((x-a)/D)#comentário leva o intervalo de a até b , números
        inteiros 0 até m
39     return y1[aux] #deslocamento na horizontal
40 xs = [a+D*(i) for i in range(m+1)] #discretizando o intervalo [a,b]
41 ys1 = [h1(x) for x in xs]
42 xf1 = [x**(alpha+beta)/gamma(alpha+beta+1) for x in xs] #Valor exato da
        equação fracionária.
43 ##### Gráficos#####
44 ###Dimensões das Figuras
45 fig = plt.figure(figsize=(12, 12))
46 plt.plot(xs,xf1,'k-', linewidth=2)
47 plt.plot(xs,ys1,'k', linewidth=2)
48 plt.legend([f'exato,  $\alpha={\alpha:.1f}$ ','MMC'],loc='upper left',
        fontsize=20)
49 plt.xlabel('x', fontsize=16)
50 plt.ylabel('y', fontsize=16)
51 plt.text(1.5,0.25, f' $\beta={\beta:.1f}$ , i={n:.0f}', fontsize=20)
52 plt.tick_params(axis='both', which='major', labelsize=14)
53 plt.tick_params(axis='both', which='minor', labelsize=14)

```

```
54 plt.show()
```

Listing C.4: PVI CAPUTO

C.5 PVI de RL com $f(x, y) = y$

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  """
4  Created on Tue Nov 22 12:24:34 2022
5  @author: luverci
6  """
7  #Programa para calcular D**alpha=y e D**alpha=-y
8  import math
9  import random as random_number
10 import matplotlib.pyplot as plt
11 from numpy import *
12 from scipy.special import gamma, factorial
13 #####
14 def mlrl(z,a):##### Função de Mittag-Leffler de um parâ
    metro
15     s = 0
16     for i in range(50):
17         s += z**i/math.gamma(i*a+alpha)
18     return s
19 #####
20 def MCint(f, a, b, c, n):
21     s = 0
22     s2 = 0
23     ran=c*n
24     for i in range(ran):
25         x = random_number.uniform(a, b)
26         j=int((x-a)/D)
27         s += f(b,x,y[j])
28         s2 += f(b,x,y[j])*f(b,x,y[j])
29     I = (float(b-a)/(ran))*s
30     Er = sqrt(abs((float(b-a)/(ran))*s2-I**2))/sqrt(ran)
31     return [I,Er]
32 #####
33 alpha = 0.75
34 #####
35 def f(xb,x,y): #kernel, agora basta colocar a função gamma.
36     return (1/gamma(alpha))*(y)*(xb-x)**(alpha-1)#####colocamos como n
    úcleo -y ou y
37 #a =0; b = 10; m=500; n = 100
38 a =0; b = 1; m=1000; n =10
39 #caso m=L, o n=i, onde o domínio é dividido em L partes e em cada
40 #uma dessas partes tiramos uma amostra de i pontos.
```

[illegible]

Listing C.5: PVI RL

C.6 Sistemas Dinâmicos Fracionários

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Thu Jan 16 09:47:53 2020
5
6 @author: luverci
```

```

7 """
8 #Programa para calcular o Sistema Lotka-Volterra fracionário.
9 import random as random_number
10 import matplotlib.pyplot as plt
11 import numpy as np
12 from scipy.special import gamma
13 ##### Daddos Iniciais#####
14 alpha = 0.8 # ordem da derivada
15 a = 0 # limite inferior do domínio
16 b = 10.0 # limite superior do domínio
17 m = 200 # número de partições iguais do intervalo [a,b]
18 n = 100 # número médio de realizações em cada partição
19 D=float((b-a)/m) # comprimento das partições
20 agamma=gamma(alpha)
21 # Rotina para calcular integria no intervalo [a,b] usando MC, onde 0 <
    c < m é a c-ésima partição do intervalo [a,b_máximo],
22 # D=float((b_máximo-a)/m) é comprimento da partição, f é a função, e n
    é o número médio de realizações em cada partição.
23 def MCint(f, a, xb, c, n):
24     s = 0
25     ran=c*n # número total de realizações no intervalo de integração [a
    ,b]
26     for i in range(ran):
27         x = random_number.uniform(a, xb)
28         j=int((x-a)/D)
29         s += f(xb,x,y[j],y2[j])
30     I = (float(xb-a)/(ran))*s # Valor da integração por MC
31     return I
32 def MCint2(f2, a, xb, c, n):
33     s = 0
34     ran=c*n # número total de realizações no intervalo de integração [a
    ,b]
35     for i in range(ran):
36         x = random_number.uniform(a, xb)
37         j=int((x-a)/D)
38         s += f2(xb,x,y[j],y2[j])
39     I = (float(xb-a)/(ran))*s # Valor da integração por MC
40     return I
41 #####
42 # Definição da função a ser integrada junto com o kernel da integral
    fracionária
43 def f(xb,x,y,y2):
44     return (1/gamma(alpha))*(y-y*y2)*(xb-x)**(alpha-1)
45 # Definição da função a ser integrada junto com o kernel da integral
    fracionária
46 def f2(xb,x,y,y2):
47     return (1/gamma(alpha))*(y*y2-y2)*(xb-x)**(alpha-1)

```

```

48 # Cálculo da integral
49 y=np.zeros(m+1) # função discretizada, com domínio no conjunto dos
    inteiros, que receberá o resultado da derivada
50 y2=np.zeros(m+1)
51 y[0]=1 # Condições iniciais
52 y2[0]=0.75
53 for k in range(1,m+1): # rotina que controí a função y (integral)
54     auxsol = MCint(f,a,a+k*D,k,n)
55     y[k]=y[0] + auxsol
56     auxsol2 = MCint2(f2,a,a+k*D,k,n)
57     y2[k]=y2[0] + auxsol2
58 #####
59 # Rotina que cria a função discretizada h(x) com domínio nos reais (
    intervalo [a,b]) à partir da função y
60 def h(x):
61     aux=int((x-a)/D) #leva x do intervalo [a,b] ao número inteiro aux no
    intervalo de 0 até m
62     return y[aux]
63 # Rotina que cria a função discretizada h2(x) com domínio nos reais (
    intervalo [a,b]) à partir da função y
64 def h2(x):
65     aux=int((x-a)/D) #leva x do intervalo [a,b] ao número inteiro aux no
    intervalo de 0 até m
66     return y2[aux]
67 # Rotina que constrói vetores para fazer o gráfico da função integrada
68 xs = [a+D*(i) for i in range(m+1)] #discretizando o intervalo [a,b]
69 ys = [h(x) for x in xs] # vetor criado à partir da função integrada
70 # Rotina que controí vetores para fazer o gráfico da função integrada
71 ys2 = [h2(x) for x in xs] # vetor criado à partir da função integrada
72 #####
73 #Rotina para a solução com derivada inteira
74 m2=10000
75 z=np.zeros(m2)
76 w=np.zeros(m2)
77 b2=7.0
78 D2=float((b2-a)/m2)
79 z[0]=1 # Condições iniciais
80 w[0]=0.75
81 for j in range(m2-1):
82     z[j+1] = z[j] + (z[j]-z[j]*w[j])*D2
83     w[j+1] = w[j] + (z[j]*w[j]-w[j])*D2
84 #Rotina para a solução com derivada fracionária e solução por diferenç
    as finitas (MDF)
85 m3=200
86 z2=np.zeros(m3)
87 w2=np.zeros(m3)
88 D3=float((b-a)/m3)

```

```

89 t = np.arange(a, b, D3)
90 z2[0]=1 # Condições iniciais
91 w2[0]=0.75
92 for j in range(m3-1):
93     z2[j+1]=z2[0]+(z2[j]-z2[j]*w2[j])*D3**alpha/(alpha*agamma)
94     w2[j+1]=w2[0]+(z2[j]*w2[j]-w2[j])*D3**alpha/(alpha*agamma)
95     for i in range(j):
96         aD3=((t[j+1]-t[i])**alpha)-(t[j+1]-t[i]-D3)**alpha)/(alpha*
            agamma)
97         z2[j+1] += (z2[i]-z2[i]*w2[i])*aD3
98         w2[j+1] += (z2[i]*w2[i]-w2[i])*aD3
99 ##### Gráficos#####
100 ###Dimensões das Figuras
101 fig = plt.figure(figsize=(12, 12))
102 plt.plot(y,y2,'k', linewidth=2)
103 plt.plot(z2,w2,'k--', linewidth=2)
104 plt.plot(z,w,'k:', linewidth=2)
105 plt.legend(['MMC', 'MDF', '$\\alpha$, $\\beta = 1$'],loc='upper right',
            fontsize=20)
106 plt.xlabel('x', fontsize=16)
107 plt.ylabel('y', fontsize=16)
108 plt.text(1.15,0.75,f'$\\alpha=${alpha:.1f}, i={n:.0f}',fontsize=20)
109 plt.tick_params(axis='both', which='major', labelsize=14)
110 plt.tick_params(axis='both', which='minor', labelsize=14)
111 plt.show()

```

Listing C.6: Sistemas Dinâmicos Fracionários

C.7 PVF com o MMCV ponto a ponto

```

1 """
2 Created on Tue Jan 14 11:44:56 2023
3
4 @author: luverci
5 """
6 #Programa para calcular o PVF  $Dy^{\alpha}-y-1=0$  de forma varicional
   ponto a ponto
7 import random as random_number
8 import matplotlib.pyplot as plt
9 import numpy as np
10 from scipy.special import gamma
11 #####
12 alpha = 1.75
13 #####Definição do domínio#####
14 a = 0; b = 2.0; m=100; n = 1
15 D=float((b-a)/m)
16 xs = [a+D*(i) for i in range(m+1)]
17 #####Definição da função inicial e suas derivadas#####

```

```

18 def ml(z,a): ##### Mittag-Leffler um parâmetro
19     s = 0
20     if z != 0:
21         for i in range(50):
22             s += (z/abs(z))**i*abs(z)**(a*i)/gamma(i*a+1)
23     else:
24         s=1
25     return s
26 #####
27 y=np.zeros(m+1)
28 y0=np.zeros(m+1)
29 y[0]=0 #condição inicial (fronteira 1)
30 y[m]=ml(2,alpha)-1#condição inicial (fronteira 2)
31 print(y[m])
32 y0[0]=y[0]
33 y0[m]=y[m]
34 for l in range(m):
35     y[l+1] = (y[m]-y[0])*(l+1)/m+y[0] #reta que é a função inicial
36     y0[l+1] = y[l+1]
37
38
39 dy0=(-3*y[0]+4*y[1]-1*y[2])/(2*1.0*D)
40 #####Rotina para o cálculo do funcional associado ao problema
41 Jfy=np.zeros(m+1)
42 agamma=gamma(alpha)
43 aDf=np.zeros((m+1,m+1))
44 for j in range(m+1):
45     for i in range(j+1):
46         aDf[j,i]=((D*(j+1-i))**(alpha)-(D*(j-i))**(alpha))/(agamma*
            alpha)
47 for j in range(m+1):
48     for i in range(j):
49         Jfy[j] += (y[i+1]+1)*aDf[j,i]
50 #####Implementação do Metropolis#####
51 aint=0.0
52 for i in range(m+1):
53     aint+=(dy0*i*D+Jfy[i]-y[i])**2*D
54 aint=1000
55 auxc=0
56 NN=10000 ##### número de passos (NN deve ser muito grande para a convergê
    ncia)
57 for i in range(NN):
58     ax = random_number.uniform(1, m) #escolhe um ponto para modificar
59     aax=int(ax)
60     x = random_number.uniform(-0.01, 0.01)
61     ay=y[aax]
62     y[aax]=y[aax]+x #ponto modificado

```

```

63     ady0 = dy0
64     dy0=(y[1]-y[0])/D #Derivada primeira numérica
65     aJfy=np.zeros(m+1)
66     for j in range(m+1):
67         for l in range(j):
68             aJfy[j] += (y[l+1]-y[l])*aDf[j,l]
69     aaint=0
70     for i in range(m+1):
71         aaint+=(dy0*i*D+Jfy[i]-y[i])**2*D
72     if aaint > aint:#### aceita ou não o ponto escolhido
73         y[aax]=ay
74         dy0=ady0
75     else:
76         auxc=auxc+1
77         aint=aaint
78         for j in range(m+1):
79             Jfy[j] = aJfy[j]
80     print(aint)
81 print(auxc)
82 #####Solução exata
83 ey=np.zeros(m+1)
84 for i in range(m+1):
85     #ey[i]=exp(i*D)
86     ey[i]=ml(i*D,alpha)-1
87 ##### Gráficos#####
88 ###Dimensões das Figuras
89 width = 12
90 height = width / 1.618
91 fig = plt.figure(figsize=(width, height))
92 plt.plot(xs,y,'ko', linewidth=2)
93 plt.plot(xs,ey,'k', linewidth=2)
94 plt.plot(xs,y0,'k:', linewidth=2)
95 plt.legend(['MMCV', 'exato', 'função inicial'], fontsize=16)
96 plt.xlabel('x', fontsize=16)
97 plt.ylabel('y', fontsize=16)
98 plt.text(1.5,0.39,f'$\\alpha=${alpha:.1f}, i={NN:.0f}', fontsize=20)
99 plt.tick_params(axis='both', which='major', labelsize=14)
100 plt.tick_params(axis='both', which='minor', labelsize=14)
101 plt.show()

```

Listing C.7: Sistemas Dinâmicos Fracionários

C.8 PVF com MMCV usando a função potência

```

1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Created on Sun Nov 17 09:07:53 2024

```

```

5
6 @author: luverci
7 """
8 #Programa para calcular o PVF  $b_{Dy}^{\alpha} aD^{\alpha} y$  de forma
    variacional
9 # usando funções potência
10 import random
11 import sympy as sp
12 import sympy
13 import numpy as np
14 import math
15 import matplotlib.pyplot as plt
16 from scipy.special import gamma
17 import time
18 t1 = time.time()
19 #
    #####

20 x = sp.symbols('x'); e = math.exp(1); pi= math.pi
21 #####MÉTODO DE SIMPSON
22 def simpson(f,a,b,n):
23     if n % 2 == 1:
24         raise ValueError("N deve ser um inteiro par.")
25     h = (b-a)/n
26     x = np.linspace(a,b,n+1)
27     y = f(x)
28     S = (h/3)*np.sum(y[0:-1:2] + 4*y[1::2] + y[2::2])
29     return S
30 ##### Rotina para a DERIVADA CAPUTO do termo x
    ^{beta}
31 def Caputob(f,alpha,b,x):
32     s=0
33     nn=6
34     for k in range(nn):
35         c = b*k
36         s += ((f[k])*gamma(c+1)/(gamma(c+1-alpha)))*x**(c-alpha)
37     return s
38 #####
39 def poli4be(a1, a2, a3, a4, b, x): #Definindo o Polinômio que iremos
    utilizar
40 #####poli4be(0)=0 e poli4be(1)=pi/2, satisfaz a condição de
    fronteira
41 a0 = 0
42 a5 = 2**(5*b)*pi**(-5*b)-a1*2**(4*b)*pi**(-4*b)-a2*2**(3*b)*pi
    **(-3*b)-a3*2**(2*b)*pi**(-2*b)-a4*2**(b)*pi**(-b)
43 p = pi/2
44 a5 = p**(-5*b)-a1*(p)**(-4*b)-a2*(p)**(-3*b)-a3*(p)**(-2*b)-a4*(p)

```

```

    **(-b)
45     a  = np.array([a0,a1,a2,a3,a4,a5])
46     y  = a[0]+(a[1])*x**(1*b) + ((a[2])*x**(2*b)) + ((a[3])*x**(3*b))
    +(a[4])*x**(4*b))+ ((a[5])*x**(5*b))
47     bx = np.array([b,2*b,3*b,4*b,5*b])
48     Yp = [y, a,bx]
49     return Yp
50 ##### Polinômio auxiliar
51 def poli5be(c1,c2,c3,c4,c5,cc,x):
52     c0=0
53     cr  = np.array([c0,c1,c2,c3,c4,c5])
54     y  = cr[0]+(cr[1])*x**(cc*1) + ((cr[2])*x**(cc*2)) + ((cr[3])*x
    **(cc*3)) + ((cr[4])*x**(cc*4))+ ((cr[5])*x**(cc*5))
55     Yp = [y, cr]
56     return Yp
57 #
    #####

58 #Valores iniciais
59 Alpha=1; p=pi/2; b=1
60 a1=0; a2=0; a3=0; a4=0; a5= p**(-5*b)
61 ##### Rotinas para a função inicial
62 p_old0=poli5be(a1, a2, a3, a4,a5,b,x)[0]
63 p_old1=poli5be(a1, a2, a3, a4,a5,b,x)[1]
64 p_caputo1=Caputob(p_old1,Alpha,b,x)
65 y1=(p_caputo1**2-p_old0**2)*1/2
66 def g(x):
67     y=y1
68     return y
69 def gteste(a1,a2,a3,a4,a5,b,x):
70     p_old0=poli5be(a1, a2, a3, a4,a5,b,x)[0]
71     p_old1=poli5be(a1, a2, a3, a4,a5,b,x)[1]
72     p_caputo1=Caputob(p_old1,Alpha,b,x)
73     y1=(1/2)*(p_caputo1**2-p_old0**2)#### MULTIPLIQUEI POR 1/2
74     y=y1
75     return y
76 def gt(x):
77     y= gteste(a1,a2,a3,a4,a5,b,x)
78     return y
79 Ip_old = simpson(gt,1e-20,pi/2,1000)
80 print(Ip_old)
81 print(p_old0)
82 ##### Monta a equação para usar o
    variacional
83 #####
84 nsteps = 10000 #número de passos
85 h = 0.3 #tamanho dos passos

```

```

86 #vetores para guardar dados
87 aa1=[];aa2=[];aa3=[];aa4=[];aa5=[];bbr=[];pp0 = [];pp1 = []
88 Ipp = [];pp_descartados = [];semente=[]
89 naccept = 0
90 print('p_old0:', p_old0)
91 print('Ip_old:', Ip_old)
92 prob_old = Ip_old # Valor da integral para a função inicial
93 #Implementação de Metropolis modificando os coeficientes MMCV função
    potência
94 for i in range(nsteps): # Implementação de Metropolis modificando todos
    os coeficientes
95     a=[a1,a2,a3,a4]
96     j=random.choice(a) #escolhe um dos 4 coeficientes para modificar
97     k=a.index(j)
98     a[k]=a[k]+random.uniform(-1,1)*h #modifica o coeficiente escolhido
99     #####
100    a1_new=a[0]
101    a2_new=a[1]
102    a3_new=a[2]
103    a4_new=a[3]
104    a_new = [a1_new,a2_new,a3_new,a4_new]
105    #####
106    b_new =np.random.uniform(0.5,3) #intervalo para escolha do beta
107    bb=b_new
108    b1=a1_new
109    b2=a2_new
110    b3=a3_new
111    b4=a4_new
112    #
113    #####
114    p=pi/2
115    a5_new = p**(-5*bb)-b1*(p)**(-4*bb)-b2*(p)**(-3*bb)-b3*(p)**(-2*bb)
116    )-b4*(p)**(-bb)
117    b5=a5_new
118    #
119    #####
120
121    def poli5be(b1,b2,b3,b4,b5,bb,x):
122        b0=0
123        br = np.array([b0,b1,b2,b3,b4,b5])
124        y = br[0]+(br[1])*x**(bb*1) + ((br[2])*x**(bb*2)) + ((br[3])
125        *x**(bb*3)) + ((br[4])*x**(bb*4)) + ((br[5])*x**(bb*5))
126        Yp = [y, br]
127        return Yp
128    #
129    #####

```

```

124     def gte(x):
125         p_new0 = poli5be(b1,b2,b3,b4,b5,bb,x)[0]
126         p_new1 = poli5be(b1,b2,b3,b4,b5,bb,x)[1]
127         p_caputo1=Caputob(p_new1,Alpha,bb,x)
128         y1=(1/2)*(p_caputo1**2-p_new0**2)##### MULTIPLIQUEI POR 1/2
129         y= y1
130         return y
131     #####Método de Simpson calcula o valor da integral do
funcional
132     Ip_new = simpson(gte,1e-20,pi/2,1000)
133     prob_new = Ip_new
134     print('passo número:',i) #acompanha a evolução das iterações
135     print('Valor da Integral:',Ip_new)
136     ##### Aceita ou rejeita a amostra
137     if prob_new < prob_old: #atualiza os novos valores
138         a1 = a1_new
139         a2 = a2_new
140         a3 = a3_new
141         a4 = a4_new
142         bb = b_new
143         Ip_old = Ip_new
144         prob_old = prob_new
145         naccept +=1
146     else: ##### mantém os valores antigos
147         a1_new = a1
148         a2_new = a2
149         a3_new = a3
150         a4_new = a4
151         b_new = bb
152         Ip_new = Ip_old
153         prob_new = prob_old
154         pp_descartados.append(p_old0)
155         naccept +=1
156         # Armazena as soluções descartadas
157         pass
158     #####
159     # Rotinas que guardam os dados obtidos nas iterações
160     aa1.append(a1)
161     aa2.append(a2)
162     aa3.append(a3)
163     aa4.append(a4)
164     bbr.append(bb)
165     pp0.append(p_old0)#soluções descartadas
166     pp1.append(p_old1)
167     Ipp.append(Ip_old)
168     print ('quantidade de passo para precisão:', len(pp0))

```

```

169 ##### Definição de funções que irão construir os grá
    ficos
170 def reta(x):
171     y=1-x
172     return y
173 def p00e(x):
174     y=poli4be(0,0,0,0,1,x)[0]
175     return y
176 def p11e(a1,a2,a3,a4,bb,x):
177     y= poli4be(a1,a2,a3,a4,bb,x)[0]
178     return y
179 def polysol(x):
180     y = np.sin(x)
181     return y
182 #####Construção da Função
183 ##### bDx _aD_x - y =0
184 #Dados para fazer os gráficos
185 m=10000
186 xs = np.linspace(0,pi/2,m+1)
187 a =0; b = pi/2;
188 D=float((b-a)/m) # comprimento das partições
189 xss = [a+D*(l) for l in range(m+1)] #discretizando o intervalo [a,b]
190 ##### Construção da função
    desvio.
191 # ####Discretização da solução encontrada
192 def ysol(x):
193     y= poli4be(a1_new,a2_new,a3_new,a4_new,b_new,x)[0]
194     return y
195 ysol_discreta=[ysol(x) for x in xs]
196 ##### Cálculo da Derivada Caputo da solução encontrada ysol
197 ysol=poli4be(a1_new,a2_new,a3_new,a4_new,b_new,x)[1]
198 Caputo_ysol=Caputob(ysol,Alpha,b_new,x)
199 ##### Discretização de Caputo_ysol
200 def Dcaputo(x):
201     y=Caputob(ysol,Alpha,b_new,x)
202     return y
203 ysol_d=[Dcaputo(x) for x in xs]
204 #####
205 ##### Cálculo da Derivada de Riemann-Liouville de ysol_d
206 xJb=np.zeros(m+1)
207 for l in range(m+1):
208     for j in range(l,m+1):
209         xJb[l]+=ysol_d[j]*((j+1-l)**(1-Alpha)-(j-l)**(1-Alpha))*D** (1-
            Alpha)/gamma(2-Alpha)
210 dRL=np.zeros(m+1)
211 for l in range(m):
212     dRL[l]=-(xJb[l+1]-xJb[l])/D

```

```

213 #####
214 ##### Vetor da Função Desvio.
215 desvio=np.zeros(m+1)
216 for j in range(m):
217     desvio[j]=dRL[j]-ysol_discreta[j]
218 ##### Função Desvio na solução
219     exata para alpha=1
219 yexata=[polsol(x) for x in xs] #solução exata discretizada
220 #####
221 #Rotina para construção da função desvio discretizando y.
222 xJbe=np.zeros(m+1)
223 for l in range(m+1):
224     for j in range(l,m+1):
225         xJbe[l]+=yexata[j]*((j+1-l)**(1-Alpha)-(j-l)**(1-Alpha))*D** (1-
226             Alpha)/gamma(2-Alpha)
226 dRLe=np.zeros(m+1)
227 for l in range(m):
228     dRLe[l]=-(xJbe[l+1]-xJbe[l])/D
229 #####Vetor da Função Desvio.
230 desvioe=np.zeros(m+1)
231 for j in range(m):
232     desvioe[j]=dRLe[j]-yexata[j]
233 #####
234 y_aprox1 = p00e(xs) # Função Inicial
235 y_aprox2 = [p11e(a1_new,a2_new,a3_new,a4_new,bb,x) for x in xs]#Solução
236     MMCV
236 y_exata=[polsol(x) for x in xs]#Solução exata apenas para o alpha=1.
237 ##### Construção dos Gráficos
238 ##### Gráfico 1 Solução
239 width = 12
240 height = width / 1.61
241 fig = plt.figure(figsize=(width, height))
242 plt.plot(xs,y_aprox2,'k', linewidth=3)
243 plt.plot(xs,y_exata,'k--',linewidth=3)
244 plt.plot(xs,y_aprox1,'k:', linewidth=2)
245 plt.legend(['MMCV', 'exato, para  $\alpha=1$ ', 'função inicial'],loc='
246     upper left', fontsize=16)
246 plt.text(1.3,0.2,f' $\alpha={Alpha:.1f}$ ', fontsize=20)
247 plt.text(0.9,0.0,f'passos={nsteps:.0f}', fontsize=20)
248 plt.xlabel('x', fontsize=16)
249 plt.ylabel('y', fontsize=16)
250 plt.tick_params(axis='both', which='major', labelsize=16)
251 plt.tick_params(axis='both', which='minor', labelsize=16)
252 plt.grid
253 plt.show()
254 #####Gráfico 2 Função Desvio
255 width = 12

```

```

256 height = width / 1.61
257 fig = plt.figure(figsize=(width, height))
258 plt.plot(xs, desvio, 'k--', linewidth=2)
259 plt.xlabel('x', fontsize=16)
260 plt.ylabel('y', fontsize=16)
261 plt.legend(['função desvio', ], loc='best', fontsize=16)
262 plt.text(0.8, -0.5, f'$\\alpha=${Alpha:.1f}', fontsize=20)
263 plt.text(0.5, -1, f'passos={nsteps:.0f}', fontsize=20)
264 plt.tick_params(axis='both', which='major', labelsize=16)
265 plt.tick_params(axis='both', which='minor', labelsize=16)
266 plt.grid
267 plt.show()
268 #####
269 tempoExec = time.time() - t1
270 print("Tempo de execução: {} segundos".format(tempoExec))

```

Listing C.8: Sistemas Dinâmicos Fracionários