Universidade Estadual de Campinas
Mestrado Profissional em Matemática Aplicada
PM009-Tópicos em Matemática I
Introdução ao Cálculo Fracionário e Aplicações

Trabalho Disponível em

https://github.com/Diogotb/ProjetoPython/tree/main/Lotka-VolterraPython

Cálculo Fracionário aplicado no Sistema de Lotka-Volterra: Uma análise didática e computacional -

Aluno(s): Diogo Takamori Barbosa - RA 037382 e Framilson José Ferreira Carneiro - RA 230113

9 de dezembro de 2023

Introdução

O trabalho analisado busca construir uma extensão do sistema de Lotka-Volterra a fim de incorporar derivadas de ordem não inteira. O modelo clássico que descreve as interações entre presa e predador, conhecido como "sistema de Lotka-Volterra", é abordado com duas derivadas de ordem inteira. Por meio de uma técnica de linearização, intenta-se obter uma solução em termos dos parâmetros constantes. Além disso, apresenta-se uma solução para o sistema assim denominado "Lotka-Volterra fracionário", que consiste em duas equações diferenciais não lineares com derivadas de ordem menor que a unidade. Tal solução é expressa em termos da função de Mittag-Leffler, por intermédio da aplicação do conceito de diferenças finitas, paralelamente à técnica de linearização utilizada.

Modelagem Sistema Clássico Lotka-Volterra em Python

O modelo Lotka-Volterra é um exemplo clássico de um sistema dinâmico que pode ser aplicado em uma ampla gama de contextos, não apenas na ecologia, mas também em campos como economia, epidemiologia e engenharia. Essa versatilidade torna o entendimento do modelo e sua implementação prática em linguagens de programação, como Python, uma habilidade valiosa.

Modelo Matemático Lotka-Volterra para Dinâmica de Populações de Presas e Predadores

$$\frac{dx}{dt} = \alpha x - \beta xy$$
$$\frac{dy}{dt} = \delta xy - \gamma y$$

onde:

- x é a população de coelhos;
- y é a população de raposas;
- α é a taxa de crescimento dos coelhos na ausência de predadores;
- β é a taxa de predação dos coelhos pelas raposas;
- γ é a taxa de morte das raposas na ausência de presas;
- δ é a taxa de crescimento das raposas devido à predação dos coelhos.

Modelo Matemático Lotka-Volterra para Dinâmica de Populações de Presas e Predadores

A simulação numérica do modelo usando o método de Euler é feita pelas seguintes equações de atualização:

$$x_{i+1} = x_i + \Delta t \cdot (\alpha x_i - \beta x_i y_i)$$
$$y_{i+1} = y_i + \Delta t \cdot (\delta x_i y_i - \gamma y_i)$$

onde: i é o índice da iteração;
Δt é o intervalo de tempo;
x_i , y_i são as populações de coelhos e raposas no passo i, respectivamente.

Simulação Computacional em Python:

Lotka-VolterraPython > 1-ModeloPresaPredadorLVGraficoBidimensional.py > ...

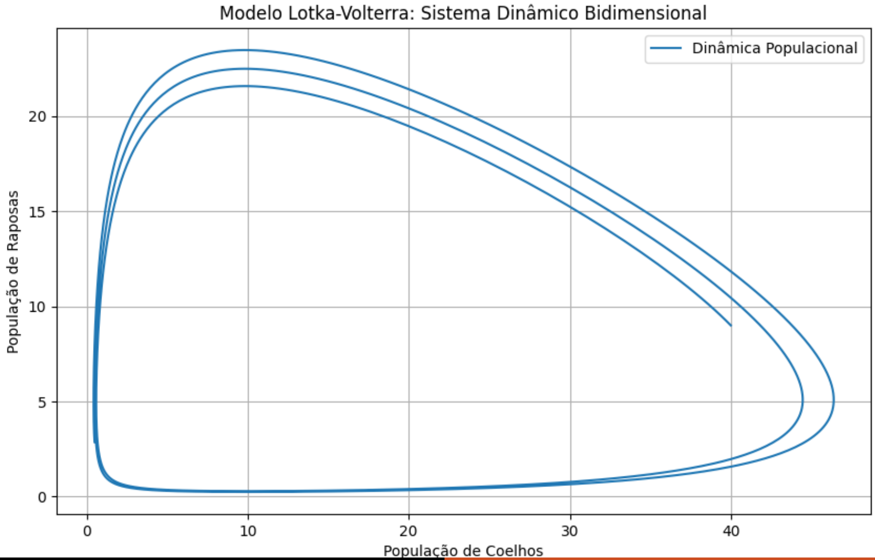
```python
import numpy as np
import matplotlib.pyplot as plt

# Parâmetros do modelo Lotka-Volterra
alpha = 0.1   # Taxa de crescimento dos coelhos na ausência de predadores
beta = 0.02   # Taxa de predação dos coelhos pelas raposas
gamma = 0.1   # Taxa de morte das raposas na ausência de presas
delta = 0.01  # Taxa de crescimento das raposas devido à predação dos coelhos

# Condições iniciais
x0 = 40  # População inicial de coelhos
y0 = 9   # População inicial de raposas

# Configuração do tempo
T = 200
dt = 0.1
num_steps = int(T / dt)
```

Figura: Enter Caption

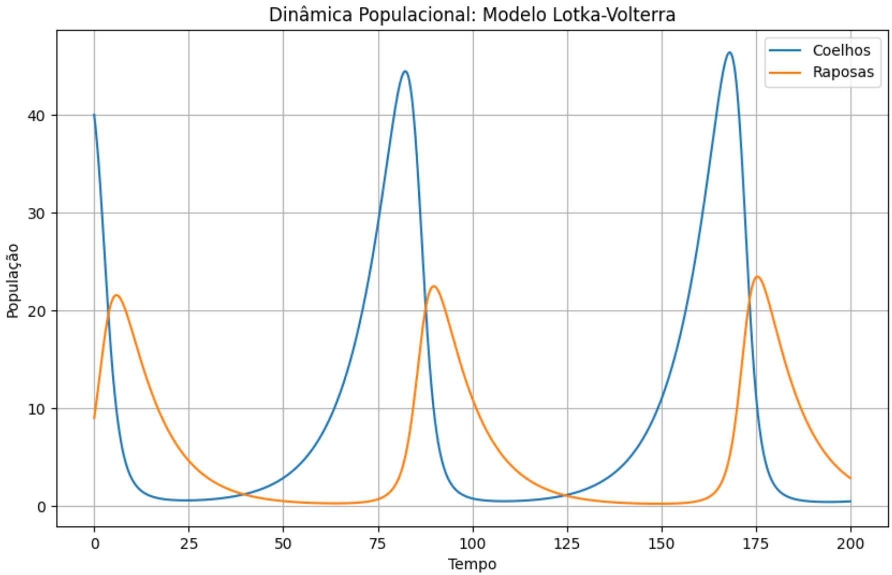
Simulação Computacional em Python:

```python
# Arrays para armazenar resultados
x_values = np.zeros(num_steps)
y_values = np.zeros(num_steps)
time_values = np.zeros(num_steps)

# Inicialização das populações iniciais
x = x0
y = y0

# Simulação do modelo Lotka-Volterra usando o método de Euler
for i in range(num_steps):
    x_values[i] = x
    y_values[i] = y
    time_values[i] = i * dt
    # Equações de Lotka-Volterra usando o método de Euler
    dx = dt * (alpha * x - beta * x * y)
    dy = dt * (delta * x * y - gamma * y)
    # Atualização das populações
    x += dx
    y += dy
```

Plotagem do Gráfico para a modelagem - Bidimensional



Plotagem do Gráfico para a modelagem - População x Tempo



Equações Diferenciais Fracionárias (EDFs) são uma generalização das Equações Diferenciais Ordinárias (EDOs), onde a ordem da derivada ou integral é um número não inteiro. Em outras palavras, elas envolvem derivadas ou integrais de ordens fracionárias. A EDF mais comum é a Equação Diferencial Fracionária (EDF) de Caputo, que é uma generalização da derivada comum para ordens não inteiras.

A forma geral de uma EDF de Caputo de ordem q para uma função $y(t)$ é dada por:

$$D_t^q y(t) = f(t, y(t), D_t^{\lceil q \rceil -1} y(t), D_t^{\lceil q \rceil -2} y(t), \ldots, y'(t), y(t))$$

onde D_t^q representa a derivada de ordem fracionária de Caputo, f é uma função que descreve a dinâmica do sistema, e $\lceil q \rceil$ denota o arredondamento para cima de q para o inteiro mais próximo.

Modelo Matemático Lotka-Volterra para Dinâmica de Populações de Presas e Predadores com Cálculo Fracionário

O modelo Lotka-Volterra com derivadas fracionárias é representado por:

$$\frac{dR}{dt} = \alpha R - \beta RF + \delta \mathscr{D}_q[R]$$
$$\frac{dF}{dt} = -\gamma F + \delta RF + \delta \mathscr{D}_q[F]$$

onde $\mathscr{D}_q$ representa a derivada fracionária e q é a ordem da derivada fracionária.

A derivada fracionária $\mathscr{D}_q$ é aproximada usando o método de diferenças finitas fracionárias, como será demonstrado no código pela função '$fractional_difference$':

$$\mathscr{D}_q[Y] = (1 - q) \cdot Y_i + q \cdot Y_{i-1}$$

onde Y pode ser R ou F.

Modelo Matemático Lotka-Volterra para Dinâmica de Populações de Presas e Predadores com Cálculo Fracionário

Para realizar os cálculos dentro do loop de atualização populacional:
Cálculo de $\frac{dR}{dt}$:

$$\frac{dR}{dt} = \alpha R - \beta RF + \delta \mathscr{D}_q[R]$$

Substituindo a expressão para $\mathscr{D}_q[R]$:

$$\frac{dR}{dt} = \alpha R - \beta RF + \delta \left((1-q) \cdot R_i + q \cdot R_{i-1}\right)$$

Cálculo de $\frac{dF}{dt}$:

$$\frac{dF}{dt} = -\gamma F + \delta RF + \delta \mathscr{D}_q[F]$$

Substituindo a expressão para $\mathscr{D}_q[F]$:

$$\frac{dF}{dt} = -\gamma F + \delta RF + \delta \left((1-q) \cdot F_i + q \cdot F_{i-1}\right)$$

Lotka-VolterraPython > 4-ModeloLVPresaPredFracDiff.py > ...

```python
import numpy as np
import matplotlib.pyplot as plt

def fractional_difference(y, alpha, dt):
    """
    Calcula a diferença fracionária usando o método de diferenças finitas fracionárias.

    Parâmetros:
    - y: Lista ou array contendo os valores da série temporal.
    - alpha: Ordem fracionária para a derivada.
    - dt: Incremento de tempo.

    Retorna:
    - Valor fracionário calculado.
    """
    if len(y) >= 2:
        # Se houver pelo menos dois elementos em y, calcula a diferença fracionária.
        return (1 - alpha) * y[-1] + alpha * y[-2]
    else:
        # Se houver menos de dois elementos, retorna o último elemento.
        return y[-1]
```

Modelo em Python Para Simulação da Dinâmica Populacional

```python
# Parâmetros do modelo Lotka-Volterra
alpha = 0.1   # Taxa de crescimento das presas na ausência de predadores
beta = 0.02   # Taxa de predação (interação entre presas e predadores)
gamma = 0.1   # Taxa de diminuição dos predadores na ausência de presas
delta = 0.01  # Taxa de crescimento dos predadores em função das presas
q = 0.5       # Ordem fracionária para a derivada (pode ser ajustada conforme necessário)

# Condições iniciais
R0 = 40  # População inicial de coelhos (presas)
F0 = 9   # População inicial de raposas (predadores)

# Configuração do tempo
t_max = 200  # Tempo máximo de simulação
dt = 0.1     # Incremento de tempo (passo)
time_points = np.arange(0, t_max, dt)  # Lista de pontos temporais

# Inicialização das populações
R = np.zeros(len(time_points))  # Lista para armazenar a população de coelhos
F = np.zeros(len(time_points))  # Lista para armazenar a população de raposas
R[0] = R0  # População inicial de coelhos
F[0] = F0  # População inicial de raposas
```

Figura: Enter Caption

Modelo em Python Para Simulação da Dinâmica Populacional

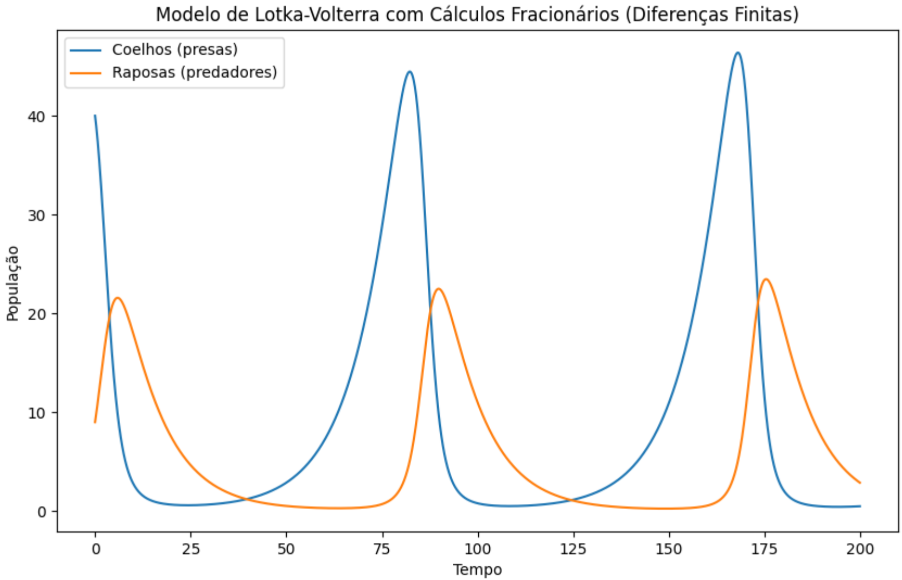
```python
# Método de diferenças finitas fracionárias para resolver EDFs
for i in range(1, len(time_points)):
    # Equações Lotka-Volterra discretizadas com derivadas fracionárias
    dRdt = alpha * R[i-1] - beta * R[i-1] * F[i-1] + delta *
    fractional_difference(R[:i], q, dt)
    dFdt = -gamma * F[i-1] + delta * R[i-1] * F[i-1] + delta *
    fractional_difference(F[:i], q, dt)

    # Atualização das populações usando o método de diferenças finitas
    fracionárias
    R[i] = R[i-1] + dt * dRdt
    F[i] = F[i-1] + dt * dFdt

# Plotagem dos resultados
plt.figure(figsize=(10, 6))
plt.plot(time_points, R, label='Coelhos (presas)')
plt.plot(time_points, F, label='Raposas (predadores)')
plt.xlabel('Tempo')
plt.ylabel('População')
plt.title('Modelo de Lotka-Volterra com Derivadas Fracionárias (Diferenças
Finitas)')
```

Neste código, as equações do modelo Lotka-Volterra são discretizadas usando o método de diferenças finitas para aproximar as derivadas. O resultado é uma simulação temporal das populações de coelhos e raposas ao longo do tempo. O gráfico final mostra como as populações evoluem de acordo com o modelo. O código fornecido implementa o modelo Lotka-Volterra com a adição de cálculos fracionários, representados pela ordem fracionária q . As equações diferenciais discretizadas usando o método de diferenças finitas são:

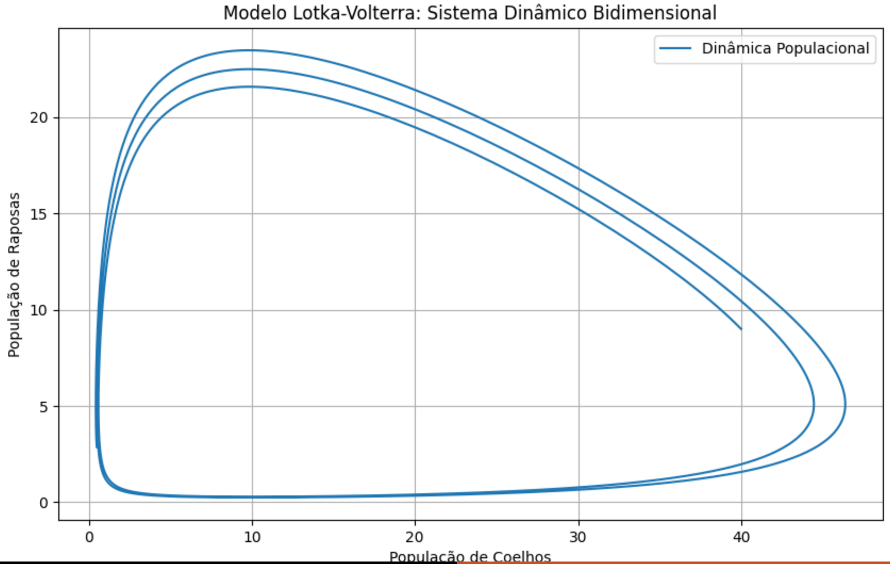
$$
\begin{aligned}
R_{i+1} &= R_i + \Delta t \cdot (\alpha R_i - \beta R_i F_i) \\
F_{i+1} &= F_i + \Delta t \cdot (-\gamma F_i + \delta R_i F_i)
\end{aligned}
$$

onde i é o índice da iteração, Δt é o intervalo de tempo, e R_i, F_i são as populações de coelhos e raposas no passo i, respectivamente.

Plotagem dos Resultados



Plotagem dos Resultados



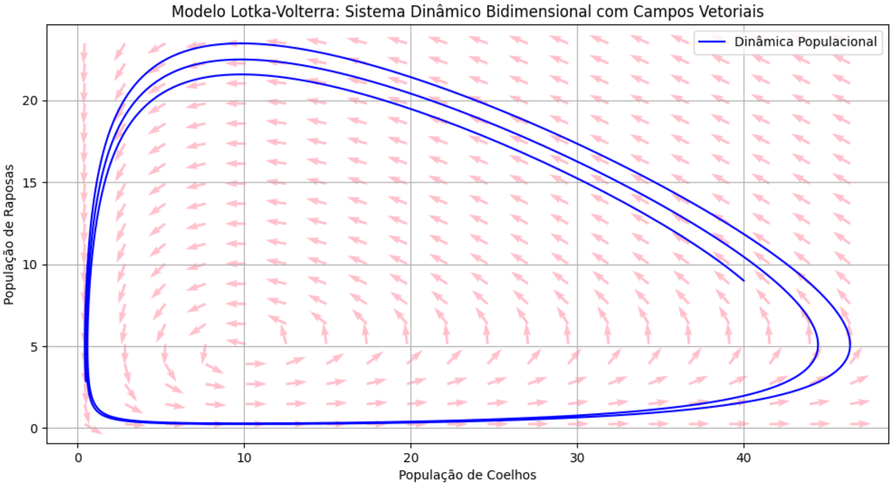


Figura: Enter Caption

Conclusão

A introdução da função fractional_difference(código Python), ou Calculo Fracionário, no cálculo das derivadas fracionárias representa uma abordagem alternativa para incorporar efeitos fracionários no modelo Lotka-Volterra em comparação com o cálculo por Equações Diferenciais Ordinárias (EDOs) padrão. A diferença fundamental está na forma como as derivadas fracionárias são tratadas. O método de diferenças finitas fracionárias é uma técnica numérica que discretiza a derivada fracionária usando diferenças finitas, levando em consideração a ordem fracionária q. Isso pode ser especialmente útil, quando se lida com sistemas complexos ou comportamentos não lineares que podem ser capturados de maneira mais precisa por derivadas fracionárias.

Bibliografia

1 - CAPUTO, M. Linear Model of Dissipation Whose Q is Almost Frequency Independent – II, Geophys. J. R. Astron. Soc., 13, 529-539, (1967).

2 - CAMARGO, R. F. Cálculo Integro diferencial de Ordem Arbitrária. Tese de doutorado em Matemática, 2008.

3 - LORENZO, C. F., HARTLEY, T. T. Initialized Fractional Calculus, NASA/PT-2000- 209943, 2000.

4 - OLIVEIRA, E. C. The Green's Function and the Lotka-Volterra System, 2007.

5 - OLIVEIRA, E. C., RODRIGUES, W. A. Funções Analíticas com Aplicações. São Paulo: Editora Livraria da Física, 2005.

6 - Python Software Foundation. (https://www.python.org/).

Muito obrigado!

Trabalho Disponível em https://github.com/Diogotb/ProjetoPython/tree/main/Lotka-VolterraPython