



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



## 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

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

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

onde:

- $x$  é a população de coelhos;
- $y$  é a população de raposas;
- $\alpha$  é a taxa de crescimento dos coelhos na ausência de predadores;
- $\beta$  é a taxa de predação dos coelhos pelas raposas;
- $\gamma$  é a taxa de morte das raposas na ausência de presas;
- $\delta$  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;

$\Delta t$  é o intervalo de tempo;

$x_i$ ,  $y_i$  são as populações de coelhos e raposas no passo  $i$ , respectivamente.

# Simulação Computacional em Python:

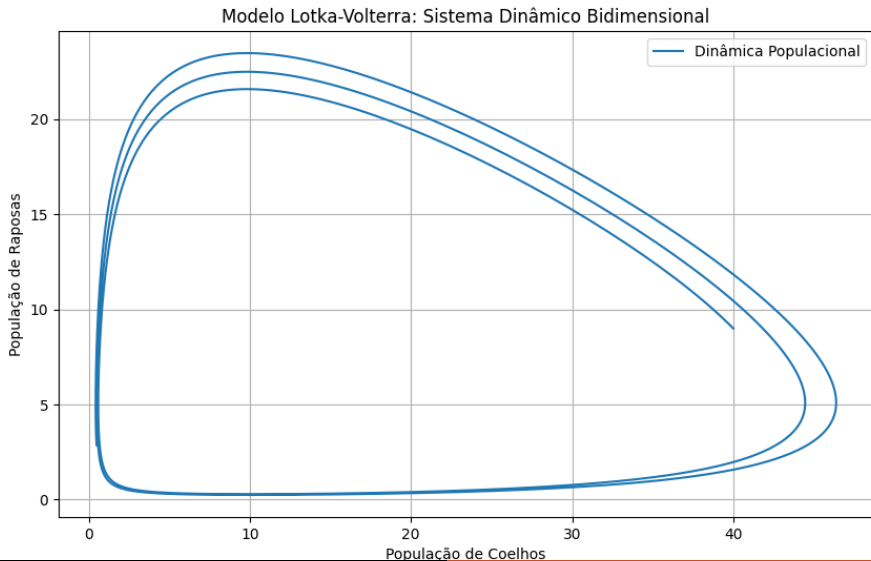
```
Lotka-VolterraPython > 1-ModeloPresaPredadorLVGraficoBidimensional.py > ...
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  # Parâmetros do modelo Lotka-Volterra
5  alpha = 0.1   # Taxa de crescimento dos coelhos na ausência de predadores
6  beta = 0.02   # Taxa de predação dos coelhos pelas raposas
7  gamma = 0.1   # Taxa de morte das raposas na ausência de presas
8  delta = 0.01  # Taxa de crescimento das raposas devido à predação dos coelhos
9
10 # Condições iniciais
11 x0 = 40  # População inicial de coelhos
12 y0 = 9   # População inicial de raposas
13
14 # Configuração do tempo
15 T = 200
16 dt = 0.1
17 num_steps = int(T / dt)
```

Figura: Enter Caption

# Simulação Computacional em Python:

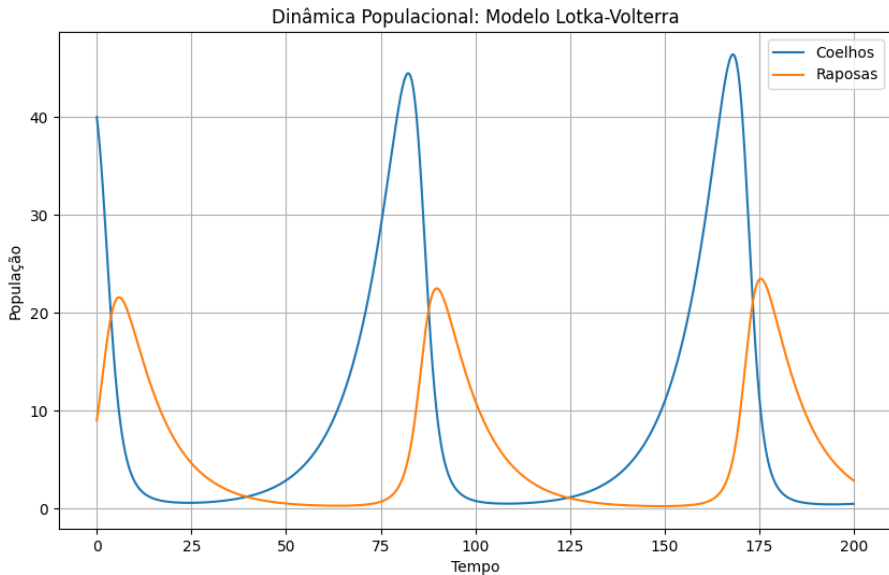
```
19 # Arrays para armazenar resultados
20 x_values = np.zeros(num_steps)
21 y_values = np.zeros(num_steps)
22 time_values = np.zeros(num_steps)
23
24 # Inicialização das populações iniciais
25 x = x0
26 y = y0
27
28 # Simulação do modelo Lotka-Volterra usando o método de Euler
29 for i in range(num_steps):
30     x_values[i] = x
31     y_values[i] = y
32     time_values[i] = i * dt
33     # Equações de Lotka-Volterra usando o método de Euler
34     dx = dt * (alpha * x - beta * x * y)
35     dy = dt * (delta * x * y - gamma * y)
36     # Atualização das populações
37     x += dx
38     y += dy
```

# Plotagem do Gráfico para a modelagem - Bidimensional





# Plotagem do Gráfico para a modelagem - População x Tempo



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

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.

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

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

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

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:

$$\begin{aligned}\frac{dR}{dt} &= \alpha R - \beta RF + \delta \mathcal{D}_q[R] \\ \frac{dF}{dt} &= -\gamma F + \delta RF + \delta \mathcal{D}_q[F]\end{aligned}$$

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

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

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

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

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  $\frac{dR}{dt}$ :

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

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

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

Cálculo de  $\frac{dF}{dt}$ :

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

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

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

# Modelo em Python Para Simulação da Dinâmica Populacional

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

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3
4  def fractional_difference(y, alpha, dt):
5      """
6      Calcula a diferença fracionária usando o método de diferenças finitas fracionárias.
7
8      Parâmetros:
9      - y: Lista ou array contendo os valores da série temporal.
10     - alpha: Ordem fracionária para a derivada.
11     - dt: Incremento de tempo.
12
13     Retorna:
14     - Valor fracionário calculado.
15     """
16     if len(y) >= 2:
17         # Se houver pelo menos dois elementos em y, calcula a diferença fracionária.
18         return (1 - alpha) * y[-1] + alpha * y[-2]
19     else:
20         # Se houver menos de dois elementos, retorna o último elemento.
21         return y[-1]
```

# Modelo em Python Para Simulação da Dinâmica Populacional

```
23 # Parâmetros do modelo Lotka-Volterra
24 alpha = 0.1 # Taxa de crescimento das presas na ausência de predadores
25 beta = 0.02 # Taxa de predação (interação entre presas e predadores)
26 gamma = 0.1 # Taxa de diminuição dos predadores na ausência de presas
27 delta = 0.01 # Taxa de crescimento dos predadores em função das presas
28 q = 0.5 # Ordem fracionária para a derivada (pode ser ajustada conforme necessário)
29
30 # Condições iniciais
31 R0 = 40 # População inicial de coelhos (presas)
32 F0 = 9 # População inicial de raposas (predadores)
33
34 # Configuração do tempo
35 t_max = 200 # Tempo máximo de simulação
36 dt = 0.1 # Incremento de tempo (passo)
37 time_points = np.arange(0, t_max, dt) # Lista de pontos temporais
38
39 # Inicialização das populações
40 R = np.zeros(len(time_points)) # Lista para armazenar a população de coelhos
41 F = np.zeros(len(time_points)) # Lista para armazenar a população de raposas
42 R[0] = R0 # População inicial de coelhos
43 F[0] = F0 # População inicial de raposas
```

Figura: Enter Caption



# Modelo em Python Para Simulação da Dinâmica Populacional

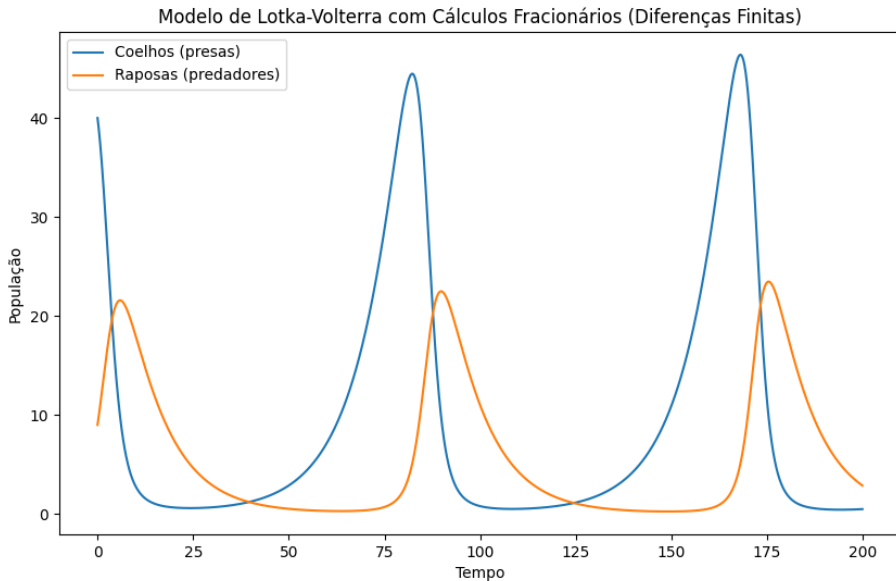
```
45 # Método de diferenças finitas fracionárias para resolver EDFs
46 for i in range(1, len(time_points)):
47     # Equações Lotka-Volterra discretizadas com derivadas fracionárias
48     dRdt = alpha * R[i-1] - beta * R[i-1] * F[i-1] + delta *
         fractional_difference(R[:i], q, dt)
49     dFdt = -gamma * F[i-1] + delta * R[i-1] * F[i-1] + delta *
         fractional_difference(F[:i], q, dt)
50
51     # Atualização das populações usando o método de diferenças finitas
         fracionárias
52     R[i] = R[i-1] + dt * dRdt
53     F[i] = F[i-1] + dt * dFdt
54
55 # Plotagem dos resultados
56 plt.figure(figsize=(10, 6))
57 plt.plot(time_points, R, label='Coelhos (presas)')
58 plt.plot(time_points, F, label='Raposas (predadores)')
59 plt.xlabel('Tempo')
60 plt.ylabel('População')
61 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:

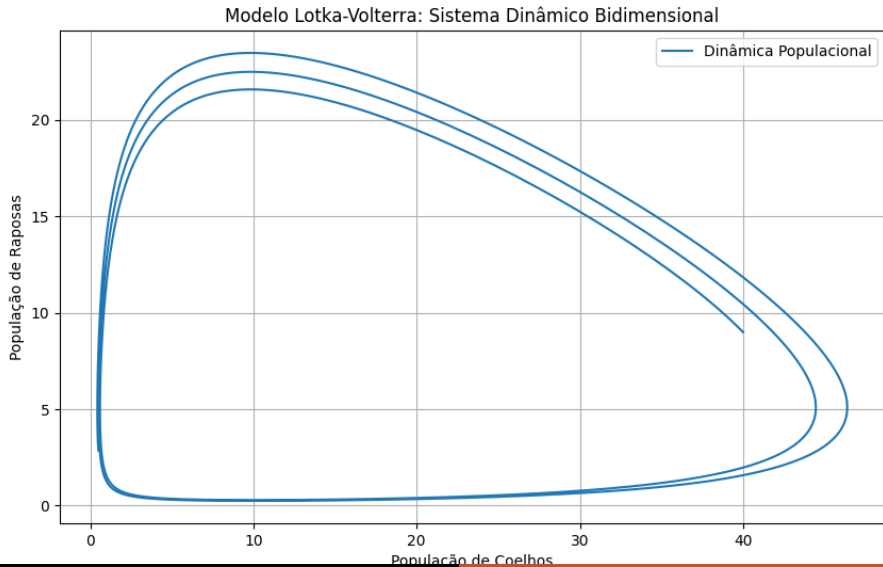
$$\begin{aligned}R_{i+1} &= R_i + \Delta t \cdot (\alpha R_i - \beta R_i F_i) \\F_{i+1} &= F_i + \Delta t \cdot (-\gamma F_i + \delta R_i F_i)\end{aligned}$$

onde  $i$  é o índice da iteração,  $\Delta t$  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



# Plotagem dos Resultados

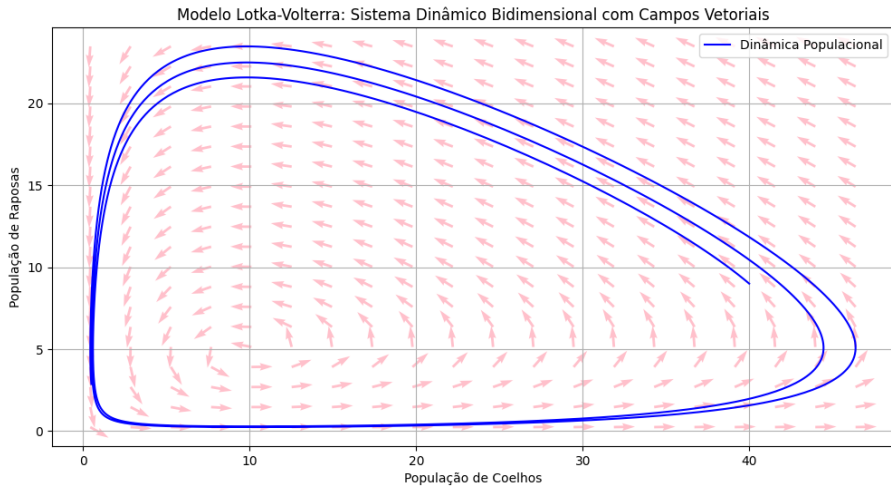


Figura: Enter Caption

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.

Muito obrigado!