Exercício Computacional 2
Métodos Numéricos para solução de EDOs
MAP3122 - Quadrimestral 2021

Gabriel Pereira de Carvalho NUSP: 11257668

Abril 2021

Contents

1 Exercicio 1: Testes **2**
1.1 Teste Runge Kutta 4 . . . 2
1.2 Teste Euler implícito . . . 7

2 Exercicio 2: Modelo presa-predador **10**
2.1 Solução com Método de Euler Explícito . . . 10
2.2 Solução com Método de Euler Implícito . . . 12
2.3 Comparação entre os Métodos Explícito e Implícito . . . 13
2.4 Solução com Método de Runge Kutta de ordem quatro . . . 16

3 Exercicio 3: Modelo duas presas-um predador **19**
3.1 alfa = 0.001 . . . 19
3.1.1 Método de Euler Explícito . . . 20
3.1.2 Método de Runge Kutta de ordem quatro . . . 22
3.2 alfa = 0.002 . . . 24
3.2.1 Método de Euler Explícito . . . 25
3.2.2 Método de Runge Kutta de ordem quatro . . . 27
3.3 alfa = 0.0033 . . . 29
3.3.1 Método de Euler Explícito . . . 30
3.3.2 Método de Runge Kutta de ordem quatro . . . 32
3.4 alfa = 0.0036 . . . 34
3.4.1 Método de Euler Explícito . . . 35
3.4.2 Método de Runge Kutta de ordem quatro . . . 37
3.5 alfa = 0.005 . . . 39
3.5.1 Método de Euler Explícito . . . 40
3.5.2 Método de Runge Kutta de ordem quatro . . . 42
3.6 alfa = 0.0055 . . . 44
3.6.1 Método de Euler Explícito . . . 45
3.6.2 Método de Runge Kutta de ordem quatro . . . 47
3.7 Teste de estabilidade . . . 49

4 Referências Bibliográficas **52**

1 Exercicio 1: Testes

Nesse exercício, foram resolvidos dois problemas de valor inicial onde a solução exata é conhecida para testar a corretude dos métodos numéricos implementados a partir da análise dos erros. Neste relatório, serão apresentadas as implementações utilizadas e os plots produzidos que comprovam sua corretude.

1.1 Teste Runge Kutta 4

```
def RungeKutta(n):#retorna vetor com aproximacoes obtidas
        h = (tf - to)/n #calculo do passo

        t = np.empty(n+1)
        t[0] = to #ponto inicial

        solucao = np.empty([n+1, xo.size])
        solucao[0] = xo #valor inicial

        for i in range(0, n):
            #calculo dos parametros k1,k2,k3,k4
            k1 = h*f(t[i], solucao[i])
            k2 = h*f(t[i] + h/2, solucao[i] + k1/2)
            k3 = h*f(t[i] + h/2, solucao[i] + k2/2)
            k4 = h*f(t[i] + h, solucao[i] + k3)
            #recorrencia
            t[i+1] = t[i] + h
            solucao[i+1] = solucao[i] + (k1+2*k2+2*k3+k4)/6

        return solucao
```

O método de Runge Kutta de quarta ordem foi implementado de forma iterativa. O valor inicial dado no problema constitui o caso base, e a cada iteração calculamos a partir dos resultados da iteração anterior os coeficientes necessários para calcular o valor da função no próximo ponto.

Para avaliar a qualidade da aproximação, foi utilizada a diferença absoluta entre $x_{numerica}$ e x_{exata}. Note que com esse método é possível garantir simultaneamente que o valor de cada uma das quatro componentes é suficientemente próximo da solução exata.

$$E_{1,n_i}(t) := máx_{1 \leqslant i \leqslant 4}\{x_{exata}(t) - x_{numérica}(t)\}$$

Como a qualidade da aproximação depende do parâmetro n de discretização do intervalo, foram utilizados seis valores distintos de $n = 20, 40, 80, 160, 320, 640$.

Em particular, estamos interessados no comportamento do erro relativo entre as diferentes execuções do algoritmo para analisar a rapidez do refinamento.

$$R_i := \frac{máx_{t \in [0,2]}\{E_{1,n_i}(t)\}}{máx_{t \in [0,2]}\{E_{1,n_{i+1}}(t)\}}$$

É importante observar que quanto maior o parâmetro de discretização, mais suave se torna a função gerada interpolando os pontos aproximados porque o passo é cada vez menor. Portanto ao aumentar n estamos aumentando a resolução da imagem.

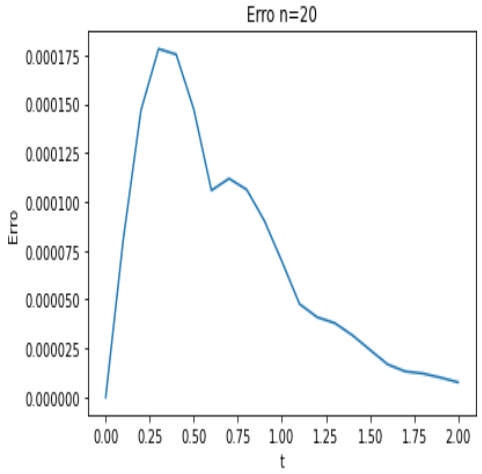

Figure 1: Plot da diferença absoluta para $n = 20$

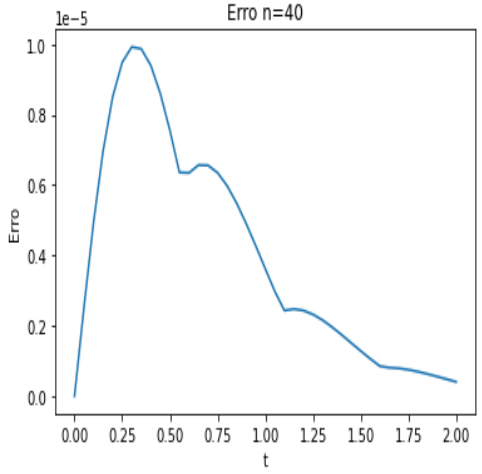

Figure 2: Plot da diferença absoluta para $n = 40$

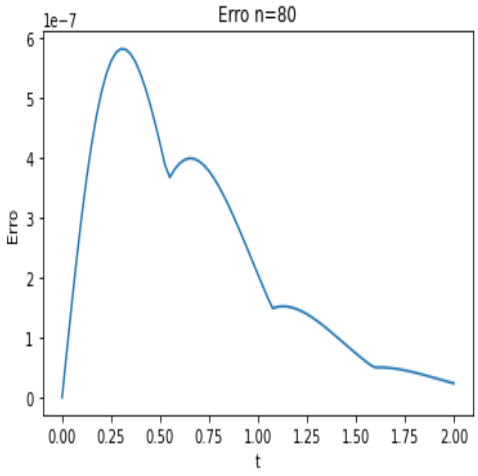

Figure 3: Plot da diferença absoluta para $n = 80$

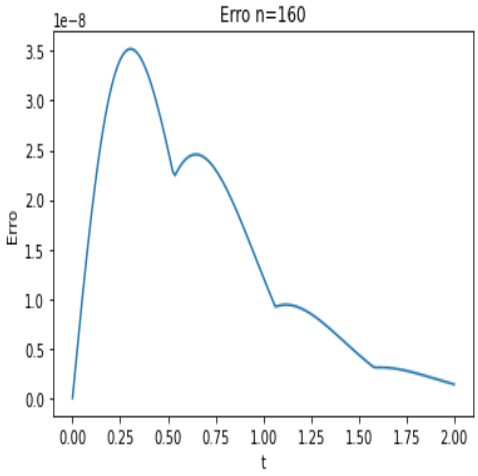

Figure 4: Plot da diferença absoluta para $n = 160$

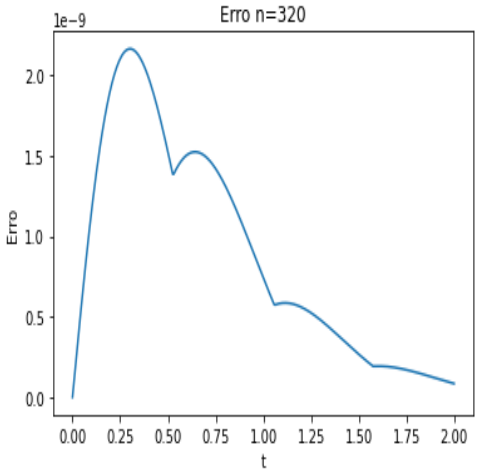

Figure 5: Plot da diferença absoluta para $n = 320$

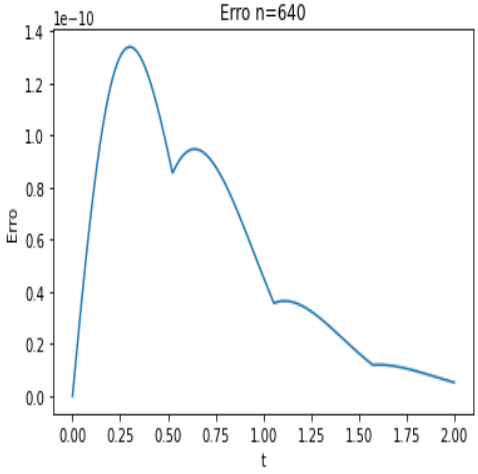

Figure 6: Plot da diferença absoluta para $n = 640$

Podemos observar ainda que o erro relativo é estritamente decrescente entre cada execução do programa e cai rapidamente. De fato, ao dobrar o parâmetro de discretização n observamos que o erro cai em mais de dez vezes.

i	R_i
1	17.960
2	17.055
3	16.543
4	16.275
5	16.138

Table 1: Cálculo do Erro Relativo para o Método de Runge Kutta de quarta ordem

1.2 Teste Euler implícito

O texto de Euler Explícito estudado em sala possui recorrência da forma

$$u_{k+1} = u_k + h * f(t_k, u_k)$$

já no método de Euler Implícito, a recorrência é da forma

$$u_{k+1} = u_k + h * f(t_{k+1}, u_{k+1})$$

Essa analogia sugere que as implementações são parecidas, no entanto é necessário um método numérico para resolver u_{k+1}. Foi utilizado o método de Newton nesse relatório.

Outra observação importante é a escolha de aproximação inicial, uma vez que a convergência do método de Newton depende da escolha de uma boa aproximação inicial. Para isso foi utilizado o Método de Euler Explícito. Foi observado que a convergência é atingida com número muito baixo de iterações.

```
def f(t, x, x_anterior):
        return x - x_anterior - h*(2*t + (x - t**2)**2)

    def fp(t, x, x_anterior): #derivada de f
        return 1 - h*2*(x - t**2)

    def MetodoNewton(t, pz, lim, tol, x_anterior):
        i = 1
        while(i <= lim):
            p = pz - f(t, pz, x_anterior)/fp(t, pz, x_anterior)
            if(mt.fabs(p - pz) < tol): #encontramos raiz
                return p
            i = i+1
            pz = p
        return "Metodo de Newton falhou"

    def EulerImplicito():

        solucao = np.empty(n + 1)
        solucao[0] = xo

        for i in range(0, n):
            #usamos Euler Explicito para aproximacao inicial
            aprox_inicial = solucao[i] + h*(2*t[i] + (solucao[i] - t[i]**2)**2)
            #a partir da aproximacao inicial, calculamos raiz com metodo de Newton
            solucao[i+1] = MetodoNewton(t[i+1], aprox_inicial, 7, 10**-8, solucao[i])

        return solucao
```

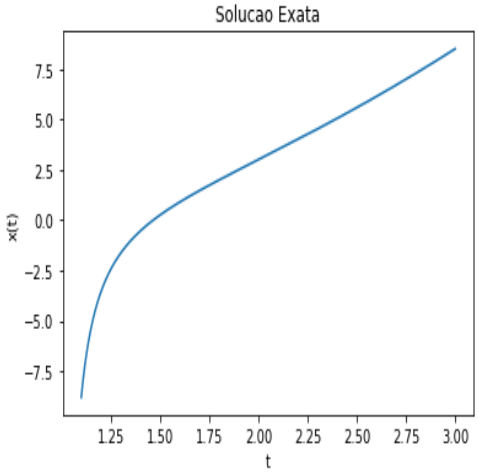

Figure 7: Plot da solução conhecida

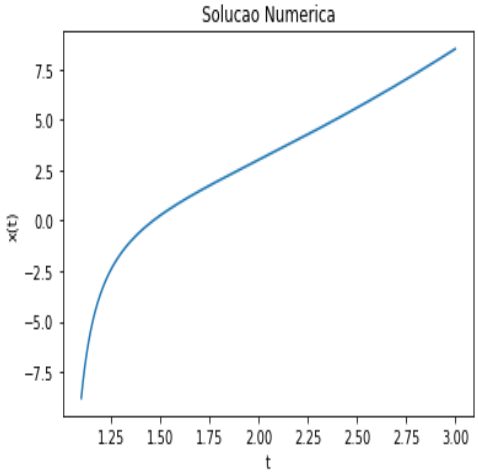

Figure 8: Plot da solução numérica com o método de Euler Implícito

Em primeira análise a forma das curvas é a mesma, o que é bom. Para analisar de forma quantitativa a qualidade da aproximação, utilizamos novamente a diferença absoluta entre as soluções.

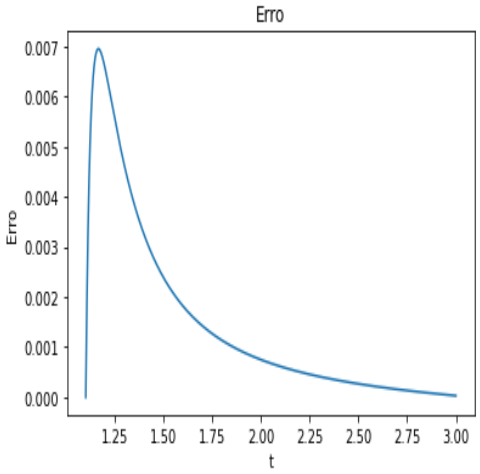

Figure 9: Plot da diferença absoluta entre as soluções

Observamos que os erros obtidos aqui são maiores do que os obtidos pelo método de Runge Kutta o que é esperado, porque o método de Euler é de ordem menor. Cabe observar que quando a função se aproxima de uma reta, a diferença absoluta é bastante pequena.

Isso ocorre porque o método de Euler, ao truncar a série de Taylor aproxima a função por uma reta. Quanto maior o parâmetro de discretização, melhor é essa aproximação.

2 Exercicio 2: Modelo presa-predador

Neste exercício, vamos resolver o seguinte sistema de equações

$$x'(t) = \lambda * x(t) - \alpha * x(t) * y(t)$$
$$y'(t) = \beta * x(t) * y(t) - \gamma * y(t)$$

Onde a função $x(t)$ representa a população de coelhos e $y(t)$ representa a população de raposas no modelo presa-predador.Para justificar o modelo de forma intuitiva, vale verificar alguns casos extremos.

Quando a população de raposas é nula, observamos que $y(t)$ é identicamente nula e que $x'(t) = \lambda * x(t)$ o que indica crescimento exponencial.

Quando a população de coelhos é nula, observamos que $x(t)$ é identicamente nula e que $y'(t) = -\gamma * x(t)$ o que indica queda exponencial. Logo, somos capazes de justificar as escolhas de sinal para λ e γ.

Quando a população de coelhos é muito grande, note que sua queda é proporcional ao número de raposas. O que faz sentido intuitivo, porque com zero raposas não há decréscimo e quanto maior a população mais rápida a queda esperada.

Analogamente, quando a população de raposas é grande seu crescimento é proporcional ao número de ovelhas que impulsiona esse crescimento. O sinal dos termos em cada equação são sempre opostos, porque um tenta balancear o efeito do outro.

2.1 Solução com Método de Euler Explícito

O passo do Método de Euler Explícito é

$$x_{k+1} = \lambda * x_k - \alpha * x_k * y_k$$
$$y_{k+1} = \beta * x_k * y_k - \gamma * y_k$$

a implementação pode ser feita de forma simples com um laço.

```
def EulerExplicito(n):#retorna vetor com aproximacoes obtidas

    h = (tf - to)/n #passo
    t = np.empty(n+1)
    t[0] = to

    solucao = np.empty([2, n+1])
    solucao[0][0] = xo #solucao[0] armazena solucao numerica de x(t)
    solucao[1][0] = yo #solucao[1] armazena solucao numerica de y(t)

    for i in range(0, n):
        #recorrencia
        t[i+1] = t[i] + h
        solucao[0][i+1] = solucao[0][i] + h*(lam*solucao[0][i] -
                                alf*solucao[0][i]*solucao[1][i])
        solucao[1][i+1] = solucao[1][i] +          h*(bet*solucao[0][i]*solucao[1][i]
                          - gam*solucao[1][i])

    return solucao
```

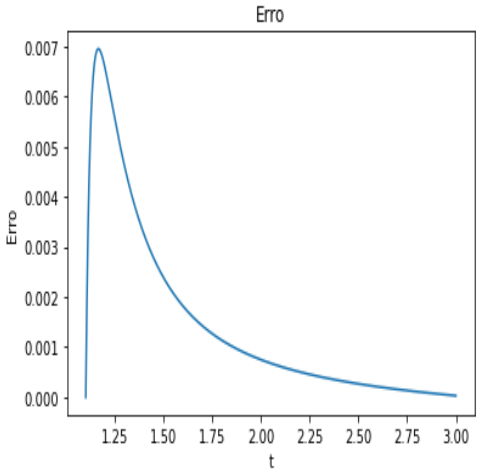

Figure 10: Plot do retrato de fase com Euler Explicito e n=5000

Observando o retrato de fase notamos que se trata de uma órbita periódica e estável. De certa forma, isso diz muito sobre a qualidade do modelo. Obtemos uma curva suave com equilíbrio estável que representa bem o equilíbrio observado no mundo natural.

Muito importante para a qualidade da aproximação do Método de Euler, como discutido no exercício anterior, é o grande parâmetro de discretização n o que justifica a escolha de 5000.

Podemos plotar cada população separadamente para observar as tendências discutidas nos casos extremos.

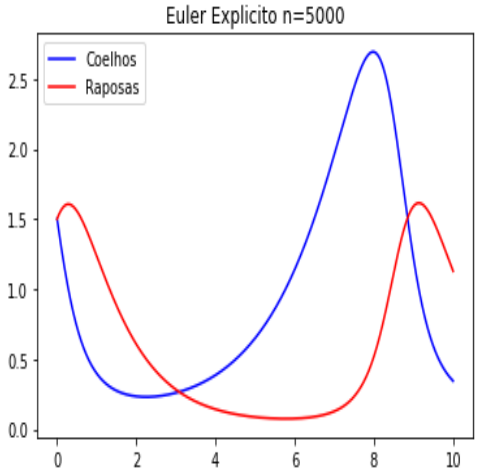

Figure 11: Plot das populações Euler Explícito

O aumento da população de raposas logo no início causa a queda da população de coelhos e, de forma análoga, o crescimento da população de raposas ao final resulta no aumento da população de raposas controlando esse crescimento.

2.2 Solução com Método de Euler Implícito

É esperado que a solução com Método de Euler Implícito seja uma aproximação melhor do que o Método de Euler Explícito dada a mesma discretização do intervalo. Porque a aproximação inicial do Método de Euler Implícito é o passo do Método de Euler Explícito e portanto a aproximação obtida no caso Implícito é um refinamento da aproximação obtida no caso Explícito.

Vale observar que o Método de Newton agora possui duas dimensões o que requer modificações sobre sua implementação. Foi utilizada uma abordagem incremental para solução do problema: primeiro decompomos as mudanças em diversas pequenas funções cuja implementação era intuitiva e depois integramos essas funções no fluxo de funcionamento do programa.

O código resultante é bastante fragmentado, porém fácil de entender e também de escrever.

```
def f1(t, x, y): #f = [f1, f2]
   return lam*x - alf*x*y

def f1delx(t, x, y): #derivada parcial de f1 com respeito a x
    return lam - alf*y

def f1dely(t, x, y): #derivada parcial de f1 com respeito a y
    return -alf*x

def f2(t, x, y): #f = [f1, f2]
    return bet*x*y - gam*y

def f2delx(t, x, y): #derivada parcial de f2 com respeito a x
    return bet*y

def f2dely(t, x, y): #derivada parcial de f2 com respeito a y
    return bet*x - gam

def f(t, u): #f = [f1, f2]
    return np.array([f1(t, u[0], u[1]), f2(t, u[0], u[1])])

def g(t, u, u_anterior): # g = u[k+1] - u[k] - h*f
    return u - u_anterior - h*f(t, u)

def jacobianoG(t, x, y): #jacobiano da funcao g
    return np.array([[1 - h*f1delx(t,x,y), -h*f1dely(t,x,y)],
                     [-h*f2delx(t,x,y), 1 - h*f2dely(t,x,y)]])

def inverteMatriz(mat): #inverte matrix 2x2
    #adjunta dividida pelo determinante
    determinante = mat[0][0]*mat[1][1] - mat[0][1]*mat[1][0]
    cof = np.array([[mat[1][1], -mat[1][0]],
                    [-mat[0][1], mat[0][0]]])
    adj = np.transpose(cof)
    return adj/determinante


def MetodoNewton(t, pz, lim, tol, u_anterior):
    i = 1
    while(i <= lim):
```

```
        p = pz - np.matmul(g(t, pz, u_anterior),inverteMatriz(jacobianoG(t, pz[0], pz[1
        if(max(mt.fabs(p[0] - pz[0]), mt.fabs(p[1] - pz[1])) < tol):
            return p
        i = i+1
        pz = p
    return "Metodo de Newton falhou"
```

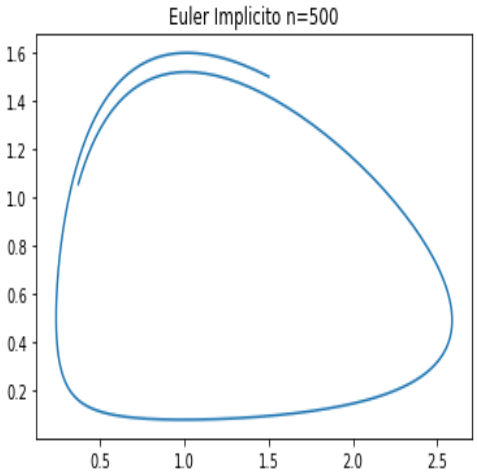

Figure 12: Plot do retrato de fase Euler Implícito com n=500

Observe que nesse retrato de fase, é possível visualizar as órbitas, pois elas não se sobrepõe.

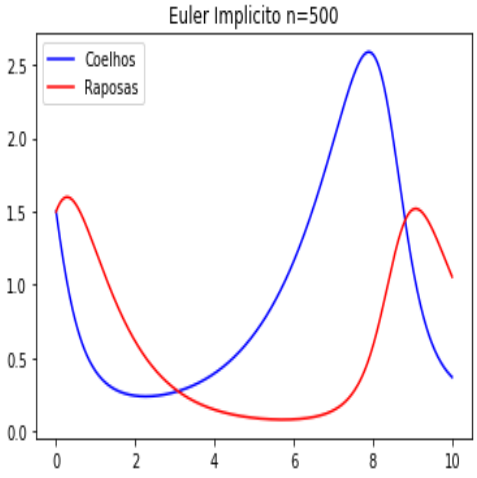

Figure 13: Plot das populações Euler Explícito

2.3 Comparação entre os Métodos Explícito e Implícito

Nessa seção iremos comparar os métodos de força quantitativa, utilizando a diferença entre a solução implícita e a solução explícita. Note que no Exercício 1 estávamos interessados na distância

da solução conhecida, portanto foi utilizado o módulo da diferença. Agora, como a solução não é conhecida, o sinal da diferença é importante para estabelecer uma relação de ordem.

Como a qualidade da aproximação depende da discretização do intervalo, foram fixados diferentes valores de $n = 250, 500, 1000, 2000, 4000$.

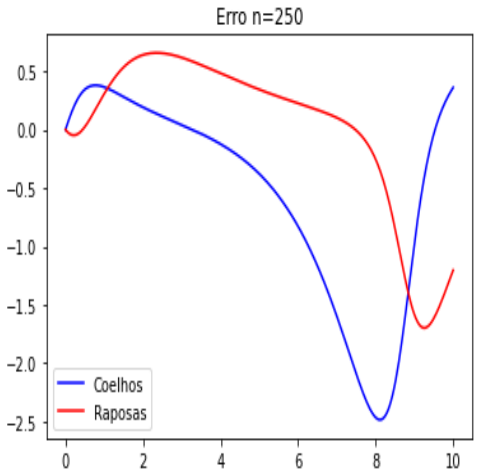

Figure 14: Diferença para n=250

Observe que a curva observada para $n = 250$ é bastante diferente do observado para n. O que é esperado, pois discretizações mais finais resultam na convergência de ambos os métodos.

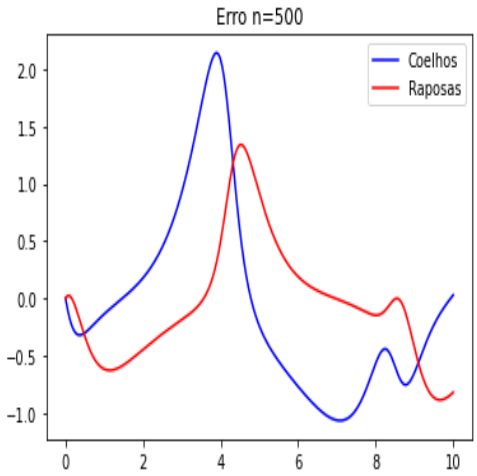

Figure 15: Diferença para n=500

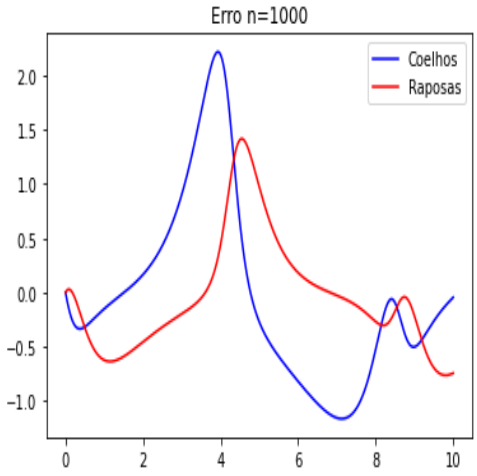

Figure 16: Diferença para n=1000

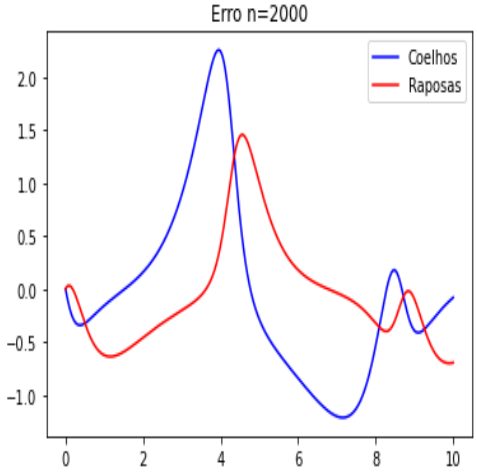

Figure 17: Diferença para n=2000

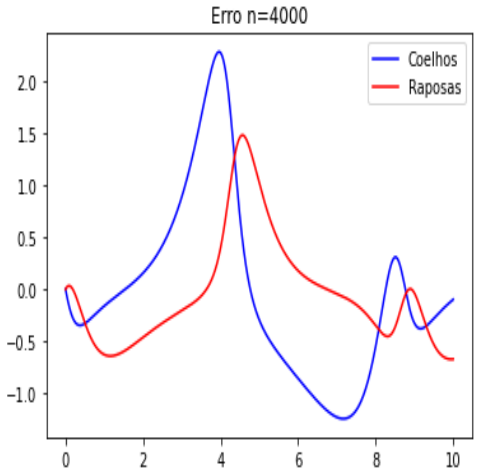

Figure 18: Diferença para n=4000

Vale observar que a ordem da diferença para $n = 4000$ não é desprezível. Como as discretizações usadas são diferentes, não é possível concluir sobre a qualidade de cada aproximação de forma comparativa. Para isso, vamos utilizar Runge Kutta como referência.

2.4 Solução com Método de Runge Kutta de ordem quatro

O método é análogo ao utilizado no Exercício 1, vale observar porém que os parâmetros K_1, K_2, K_3, K_4 são agora vetores.

```
def RungeKutta(n):#retorna vetor com aproximacoes obtidas
    h = (tf - to)/n #calculo do passo

    t = np.empty(n+1)
    t[0] = to #ponto inicial

    solucao = np.empty([2, n+1])
    solucao[0][0] = xo #valor inicial
    solucao[1][0] = yo

    for i in range(0, n):
        u = np.array([solucao[0][i], solucao[1][i]])
        #calculo dos parametros k1,k2,k3,k4
        k1 = h*f(t[i], u)
        k2 = h*f(t[i] + h/2, u + k1/2)
        k3 = h*f(t[i] + h/2, u + k2/2)
        k4 = h*f(t[i] + h, u + k3)
        #recorrencia
        t[i+1] = t[i] + h
        solucao[0][i+1] = u[0] + (k1[0]+2*k2[0]+2*k3[0]+k4[0])/6
        solucao[1][i+1] = u[1] + (k1[1]+2*k2[1]+2*k3[1]+k4[1])/6

    return solucao
```

Pela ordem do método, é esperado que ele seja a melhor aproximação da solução exata. Portanto, são esperadas pequenas diferenças com relação aos retratos de fase fornecidas pelo Método de Euler.

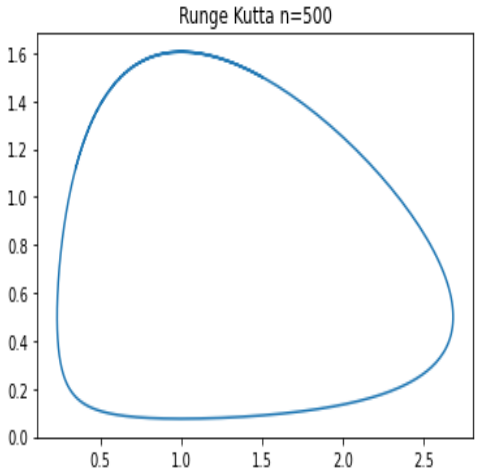

Figure 19: Plot do retrato de fase Runge Kutta n=500

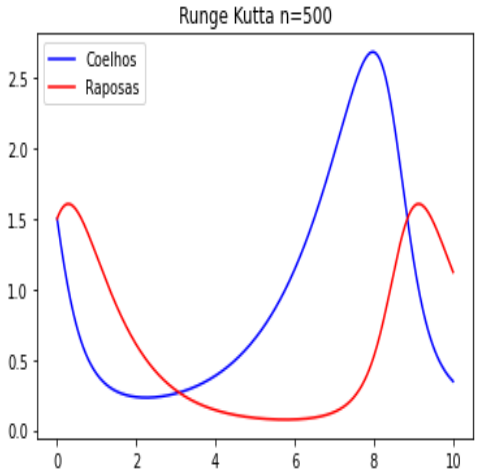

Figure 20: Plot das populações Runge Kutta n=500

Ainda nessa seção, vamos observar os retratos de fase obtidos com cada método sobrepostos.

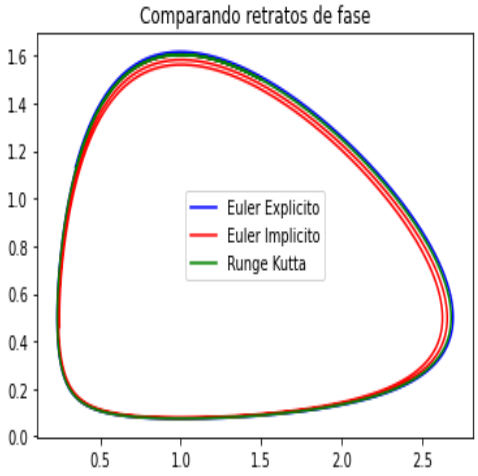

Figure 21: Retratos de fase sobrepostos

Vale observar que o Método de Runge Kutta de quarta ordem e o método de Euler Explícito praticamente se sobrepõe. Já o método de Euler Implícito apresenta órbitas menos estáveis e não se sobrepõem com os demais.

A conclusão aqui é que a discretização é um fator fundamental para a qualidade do Método de Euler. O refinamento do método implícito não foi capaz de compensar o parâmetro de discretização n menor.

3 Exercicio 3: Modelo duas presas-um predador

Neste exercício, vamos resolver o seguinte sistema de equações:

$$x'(t) = x(t) * (B_1 - A_{1,1} * x(t) - A_{1,2} * y(t) - A_{1,3} * z(t)$$

$$y'(t) = y(t) * (B_2 - A_{2,1} * x(t) - A_{1,2} * y(t) - A_{1,3} * z(t)$$

$$z'(t) = z(t) * (B_3 - A_{3,1} * x(t) - A_{3,2} * y(t)$$

Primeiramente, observamos que o crescimento ou queda de cada população é sempre proporcional ao seu tamanho. Afinal, as tendências são acentuadas pelo tamanho da população o que é esperado.

De forma similar ao que foi feito no item 2, vamos utilizar casos extremos para tentar entender a forma das equações com argumentos intuitivos.

Caso não haja presas, isto é, $x(t) = y(t) = 0$ note que a população de raposas decae exponencialmente. Isso explica porque B_3 é negativo. A presença de presas incentiva o crescimento da população de raposas, portanto $A_{3,1}, A_{3,2}$ são negativos resultando numa contribuição positiva.

Caso haja apenas coelhos ou apenas lebres, é esperado crescimento da população da presa solitária o que explica porque B_1, B_2 são ambos positivos.Porém note que caso a população seja muito grande, o modelo nos permite balancear seu crescimento o que explica porque $A_{1,1}, A_{2,2}$ são ambos positivos.

Nesse ponto, o modelo é diferente do estudado no exercício 2. Anteriormente, presas solitárias teriam crescimento exponencial. O modelo atual porém retrata a competição por recursos entre presas sejam elas da mesma espécie ou de espécies diferentes.

Portanto, caso haja ambos coelhos e lebres é esperado competição e $A_{1,2}, A_{2,1}$ são ambos positivos.

Ao aumentar o parâmetro , estamos aumentando a competição das raposas entre si o que é acentuado na ausência de presas. Portanto, quanto maior mais difícil é atingir grandes populações de raposas o que pode causar uma explosão no número de presas. Portanto é necessário evitar grandes.

3.1 alfa = 0.001

Iremos analisar plots das populações, retratos de fase $2d$ e retratos de fase $3d$ utilizando dois métodos em todos os casos: o Método de Euler Explícito e o Método de Runge Kutta de ordem quatro.

Os retratos de fase observados aqui são característicos de órbitas não períodicas com equilíbrio instável.

3.1.1 Método de Euler Explícito

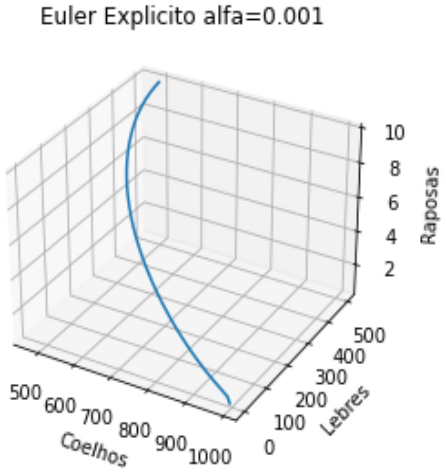

Figure 22: Retrato de fase 3d

Para facilitar a interpretação, vamos focar a análise dos retratos de duas dimensões.

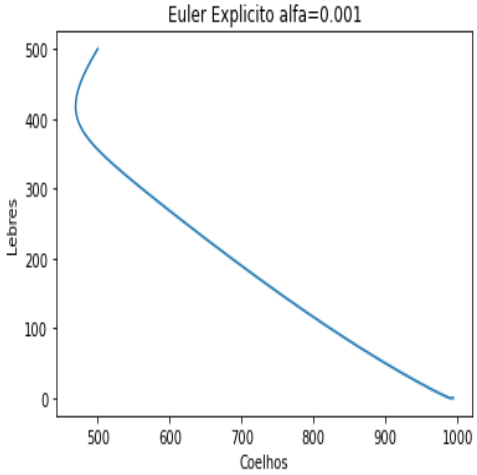

Figure 23: Retrato de fase CoelhosxLebres

Vale observar como o número alto de presas causou o decréscimo de ambas as populações. Um exemplo de competição.

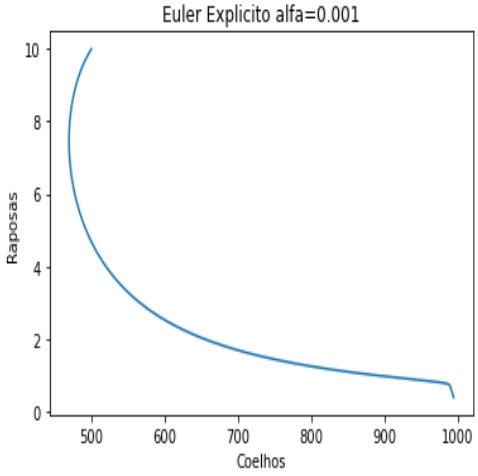

Figure 24: Retrato de fase CoelhosxRaposas

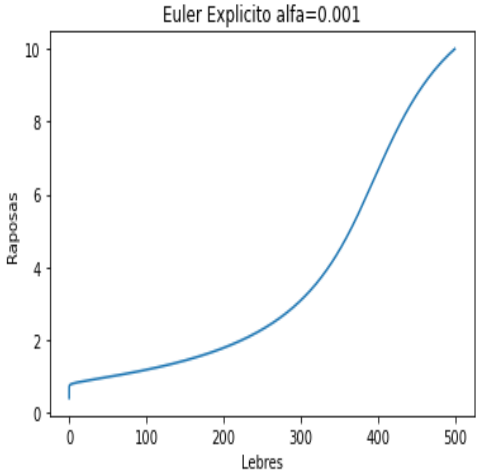

Figure 25: Retrato de fase LebresxRaposas

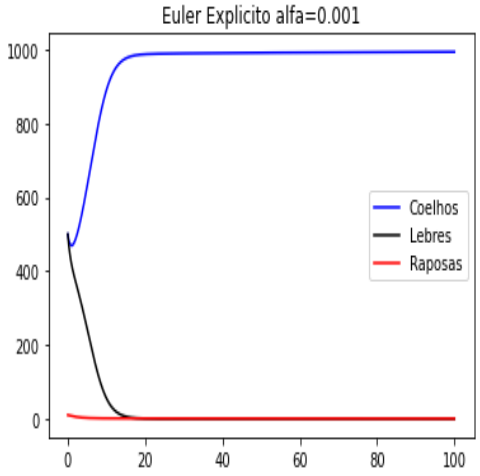

Figure 26: Evolução das três espécies

Observe que $A_{1,2} < A_{2,1}$ portanto na ausência de presas, as lebras competindo contra coelhos enquanto o efeito de competição interno de cada espécie possue o mesmo peso. Isso explica porque coelhos têm um aumento enquanto lebres tem uma queda bastante acentuada.

O efeito predominante neste plot é a competição entre coelhos e lebres.

3.1.2 Método de Runge Kutta de ordem quatro

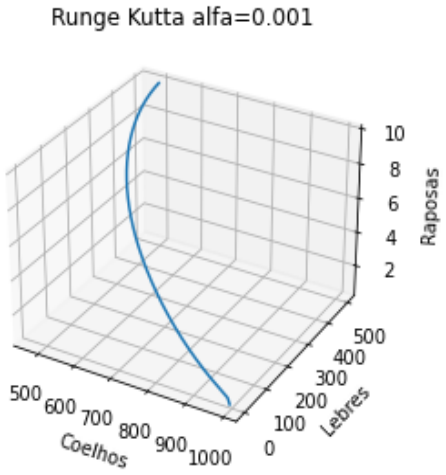

Figure 27: Retrato de fase 3d

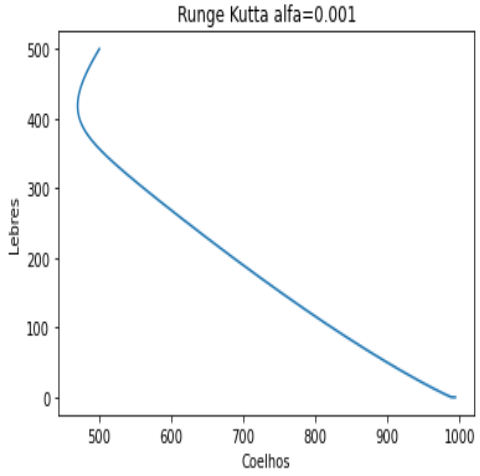

Figure 28: Retrato de fase CoelhosxLebres

Vale observar que há grande semelhança entre os dois métodos neste primeiro caso.

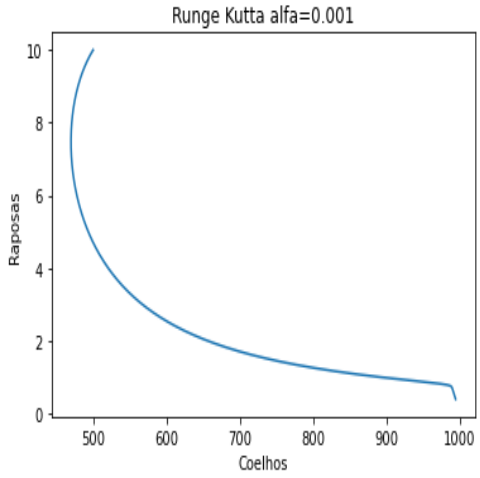

Figure 29: Retrato de fase CoelhosxRaposas

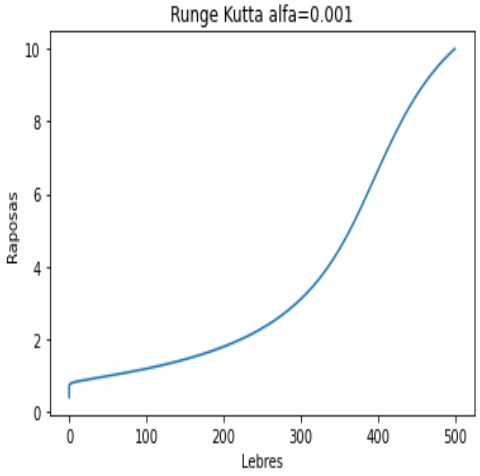

Figure 30: Retrato de fase LebresxRaposas

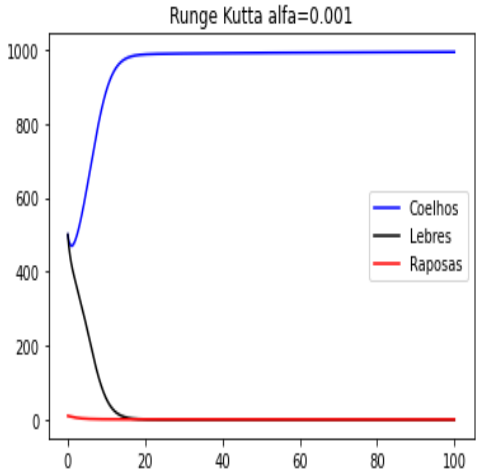

Figure 31: Evolução das três espécies

3.2 alfa = 0.002

Nesta seção o efeito predominante é a preferência das raposas por coelhos. Enquanto na seção anterior, coelhos dominaram as lebres na competição, nessa seção ambas terão populações similares porque a preferências das raposas por coelhos foi dobrada.

Ao aumentar alfa, a população de lebres é muito favorecida.

Ainda não há grandes discrepâncias entre os dois métodos numéricos. Observamos órbitas periódicas de equilíbrio não estável. Pequenas mudanças nas populações nos retiram da região de equilíbrio do retrato de fase.

3.2.1 Método de Euler Explícito

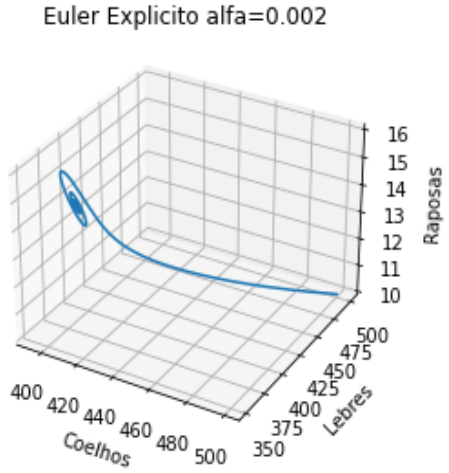

Figure 32: Retrato de fase 3d

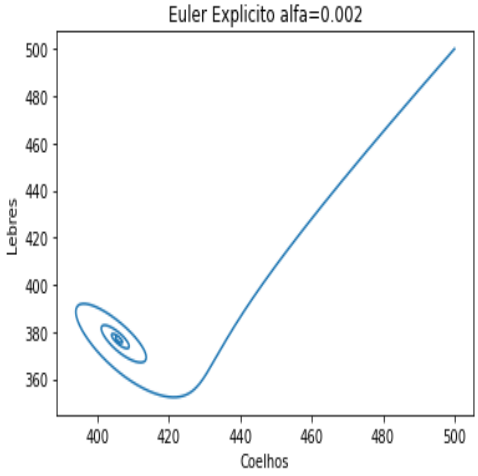

Figure 33: Retrato de fase CoelhosxLebres

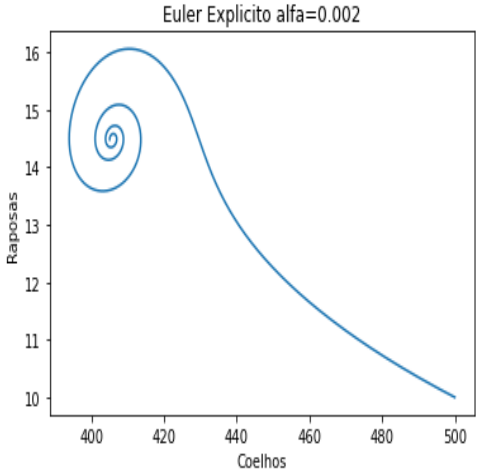

Figure 34: Retrato de fase CoelhosxRaposas

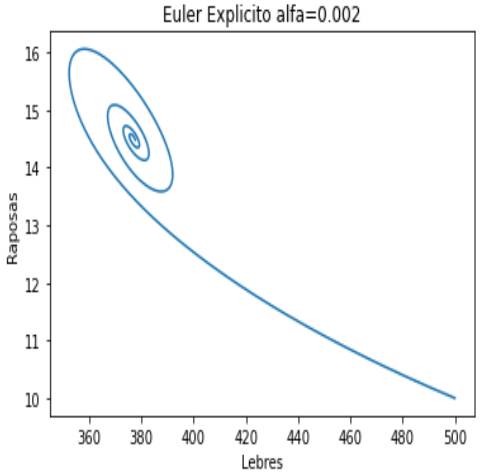

Figure 35: Retrato de fase LebresxRaposas

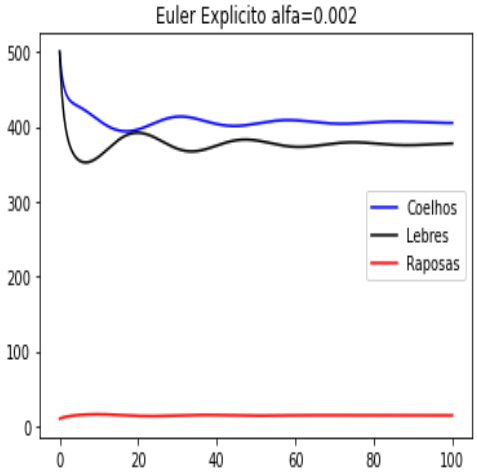

Figure 36: Evolução das três espécies

3.2.2 Método de Runge Kutta de ordem quatro

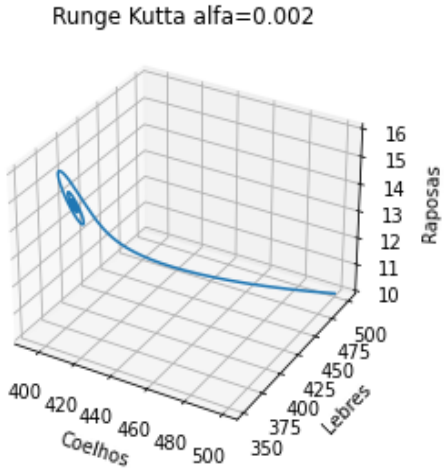

Figure 37: Retrato de fase 3d

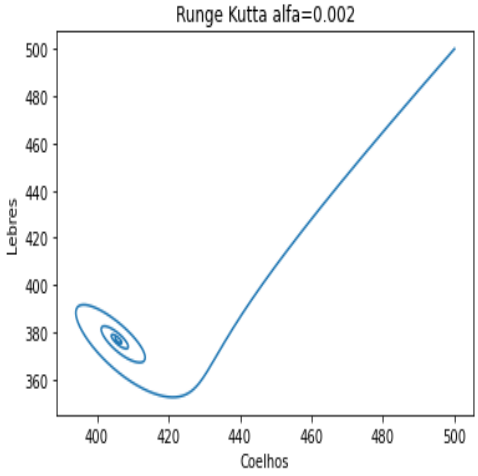

Figure 38: Retrato de fase CoelhosxLebres

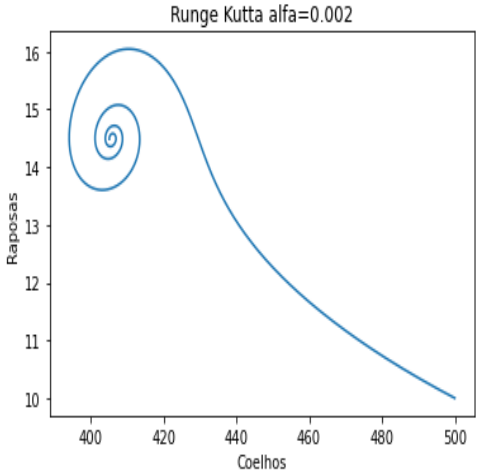

Figure 39: Retrato de fase CoelhosxRaposas

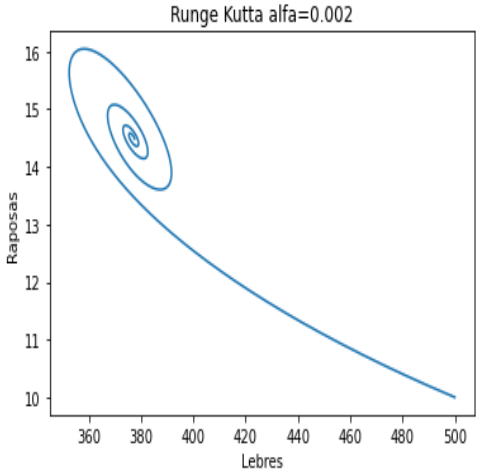

Figure 40: Retrato de fase LebresxRaposas

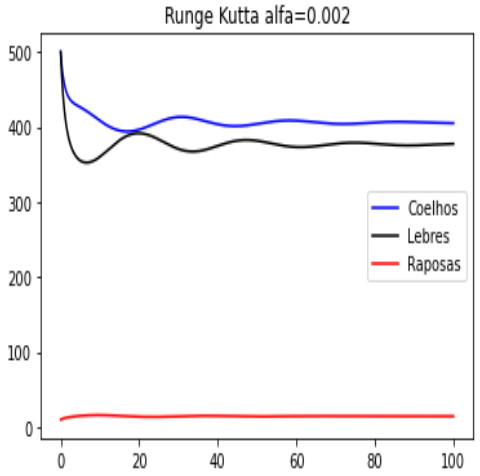

Figure 41: Evolução das três espécies

3.3 alfa = 0.0033

Aqui já podemos observar discrepâncias entre os métodos numéricos. Enquanto em Runge Kutta, as órbitas são períodicas e estáveis se sobrepondo quase perfeitamente no Método de Euler Explícito os erros de aproximação levam as desvios que impedem a sobreposição das órbitas, temos equilíbrio instável.

Aqui também está claro a vantagem das lebres que atingem máximos muito acima dos coelhos. Vale observar que esses patamares são cada vez mais largos e achatados com o aumento do parâmetro .

3.3.1 Método de Euler Explícito

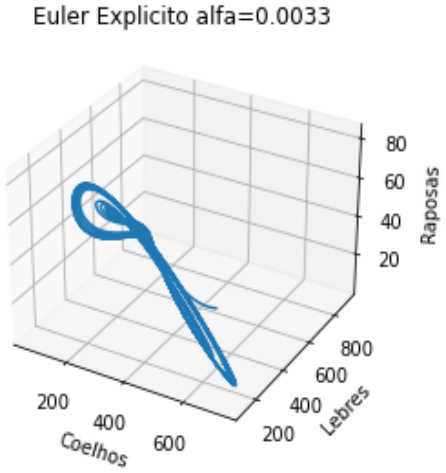

Figure 42: Retrato de fase 3d

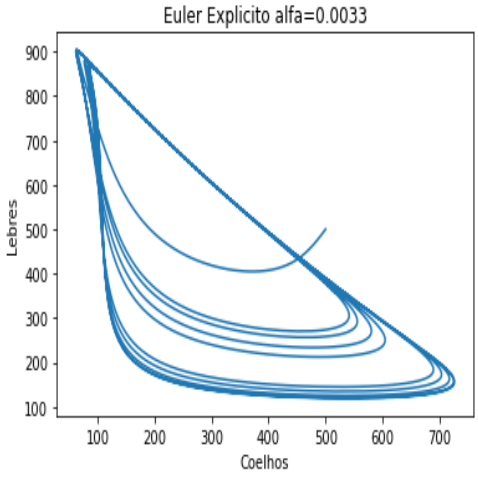

Figure 43: Retrato de fase CoelhosxLebres

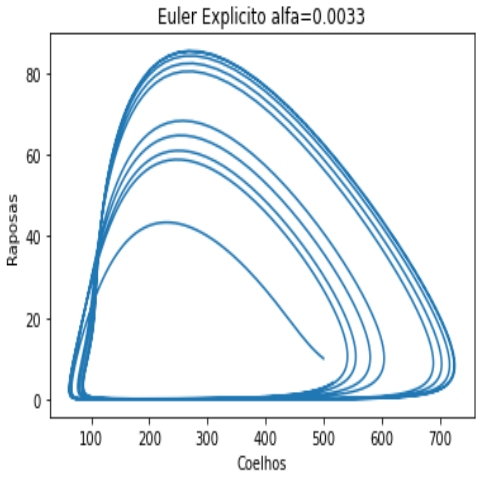

Figure 44: Retrato de fase CoelhosxRaposas

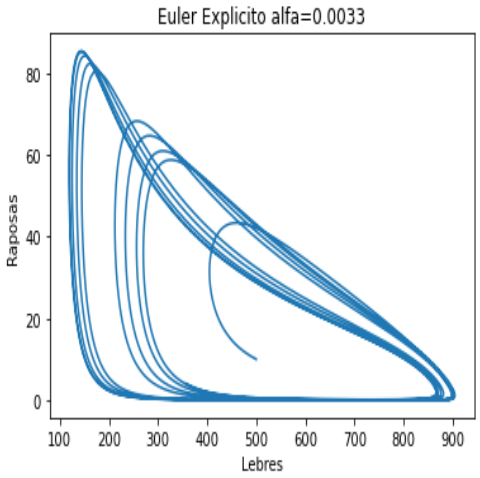

Figure 45: Retrato de fase LebresxRaposas

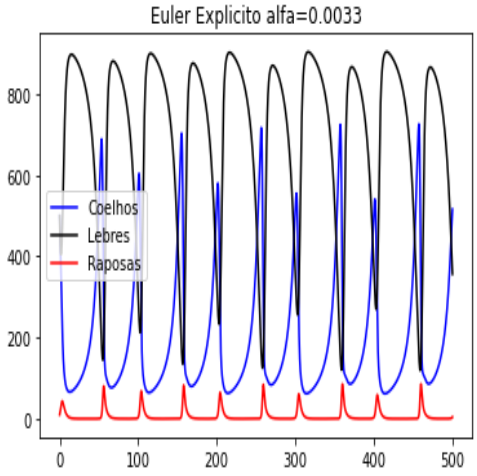

Figure 46: Evolução das três espécies

3.3.2 Método de Runge Kutta de ordem quatro

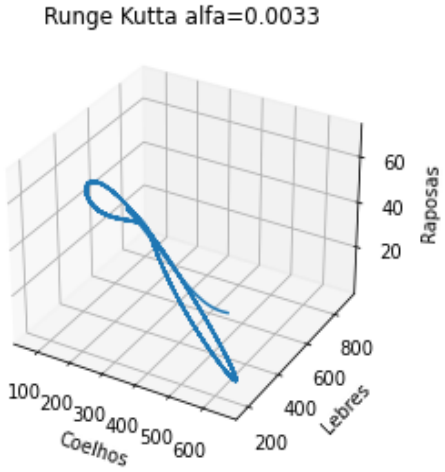

Figure 47: Retrato de fase 3d

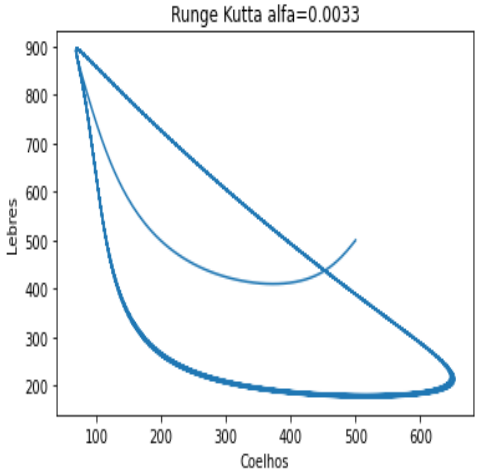

Figure 48: Retrato de fase CoelhosxLebres

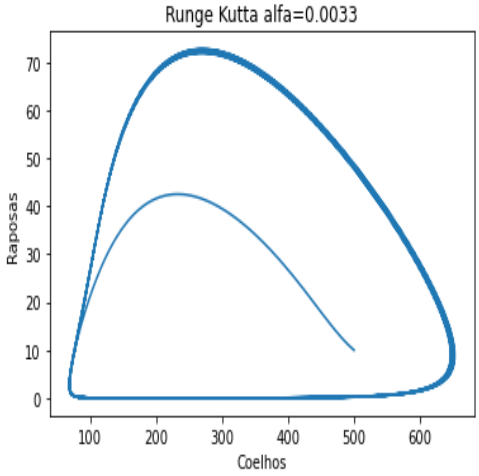

Figure 49: Retrato de fase CoelhosxRaposas

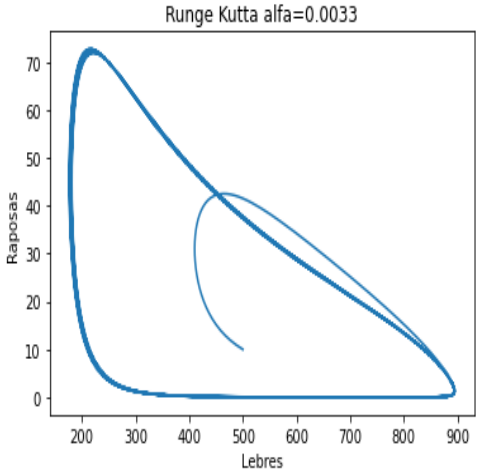

Figure 50: Retrato de fase LebresxRaposas

Figure 51: Evolução das três espécies

3.4 alfa = 0.0036

Aqui observamos o Método de Runge Kutta perder a estabilidade, as órbitas não se sobrepõe mais. Vale observar no entanto que o Método de Euler produz soluções muito mais instáveis.

3.4.1 Método de Euler Explícito

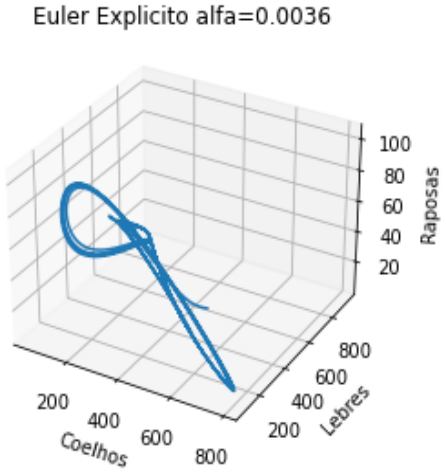

Figure 52: Retrato de fase 3d

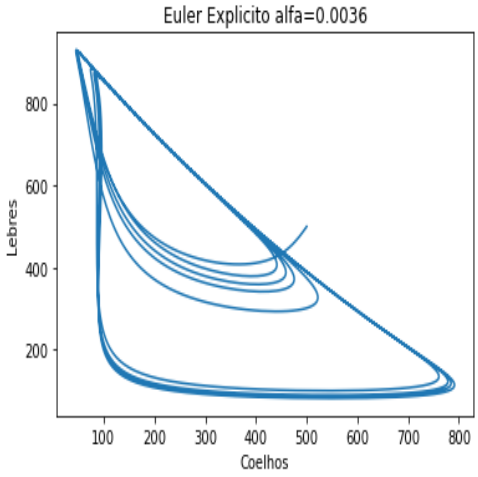

Figure 53: Retrato de fase CoelhosxLebres

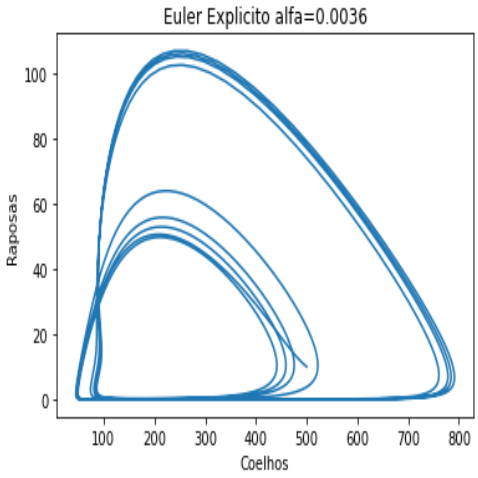

Figure 54: Retrato de fase CoelhosxRaposas

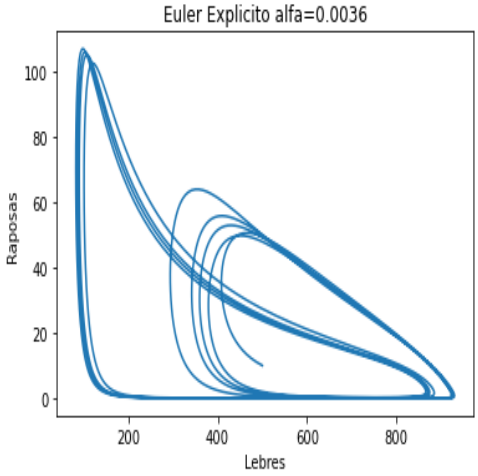

Figure 55: Retrato de fase LebresxRaposas

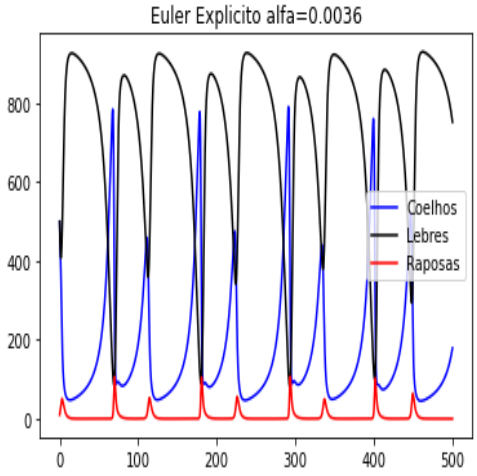

Figure 56: Evolução das três espécies

3.4.2 Método de Runge Kutta de ordem quatro

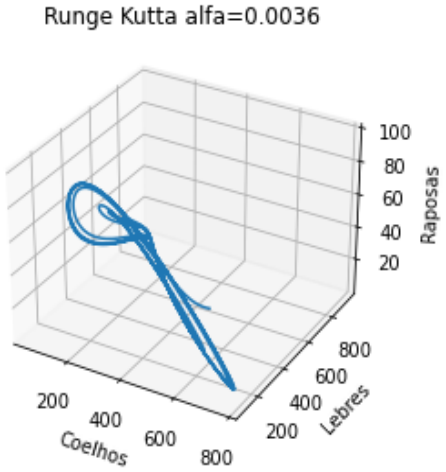

Figure 57: Retrato de fase 3d

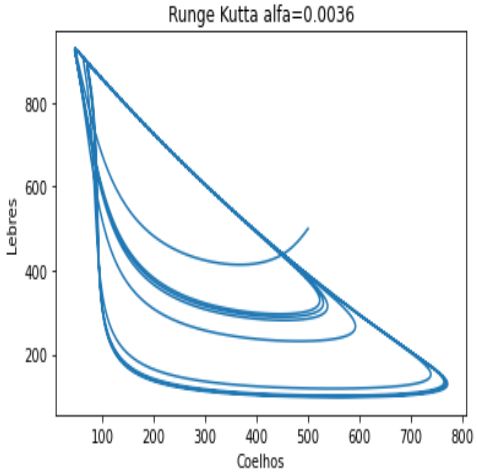

Figure 58: Retrato de fase CoelhosxLebres

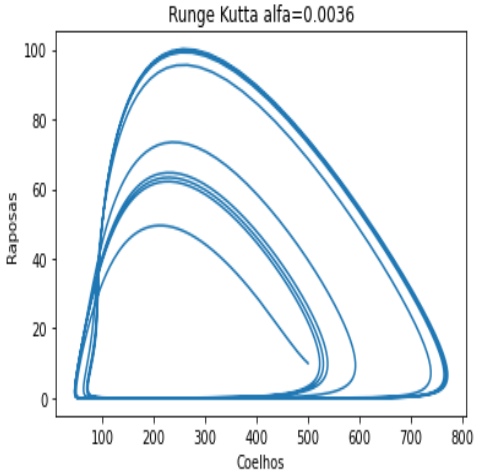

Figure 59: Retrato de fase CoelhosxRaposas

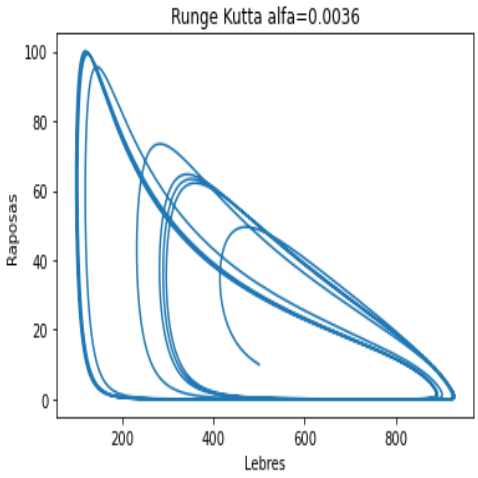

Figure 60: Retrato de fase LebresxRaposas

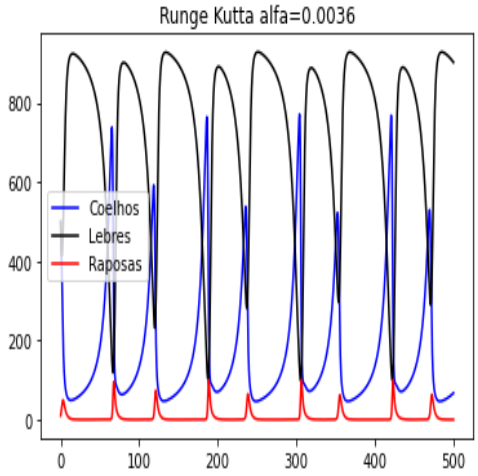

Figure 61: Evolução das três espécies

3.5 alfa = 0.005

Aqui observamos grandes discrepâncias entre os modelos. A população de lebres explode no Método de Euler como previsto.

O Método de Euler não é mais confiável para tirar inferências sobre a solução real do sistema.

3.5.1 Método de Euler Explícito

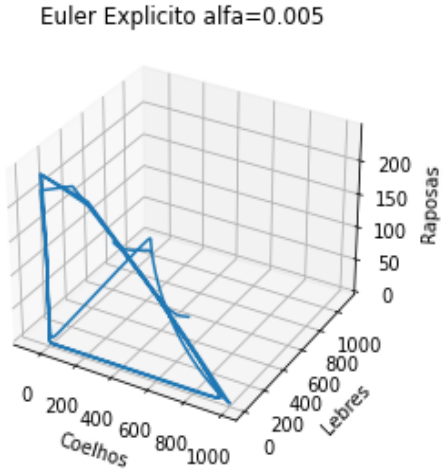

Figure 62: Retrato de fase 3d

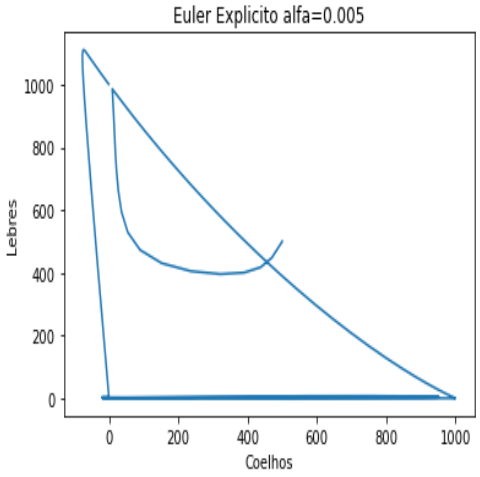

Figure 63: Retrato de fase CoelhosxLebres

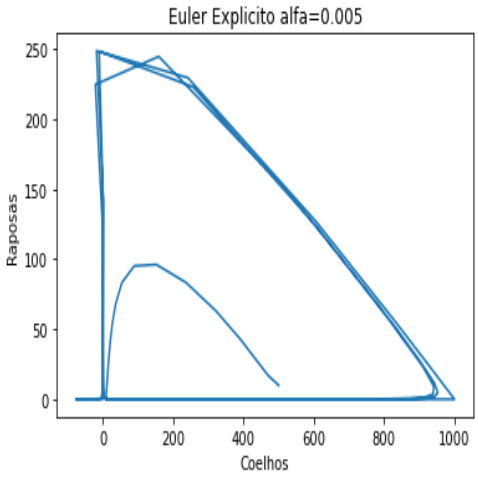

Figure 64: Retrato de fase CoelhosxRaposas

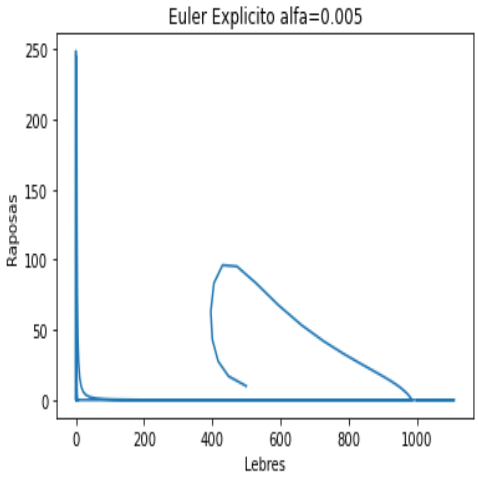

Figure 65: Retrato de fase LebresxRaposas

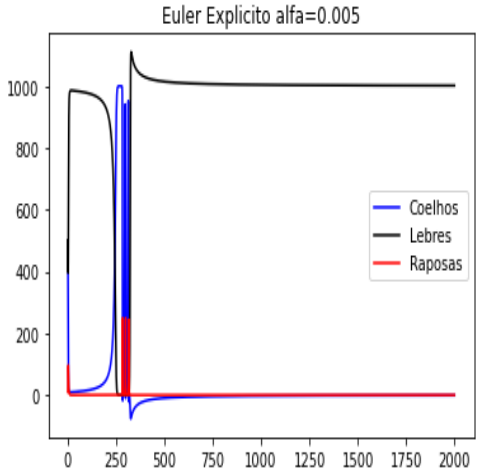

Figure 66: Evolução das três espécies

3.5.2 Método de Runge Kutta de ordem quatro

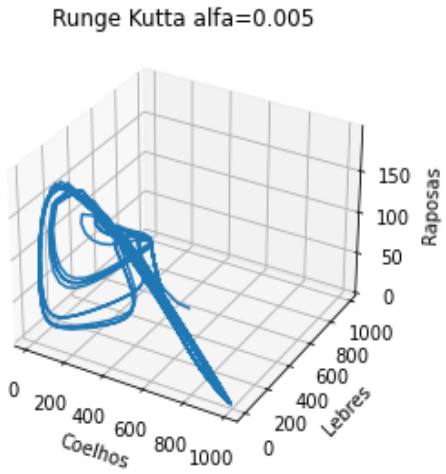

Figure 67: Retrato de fase 3d

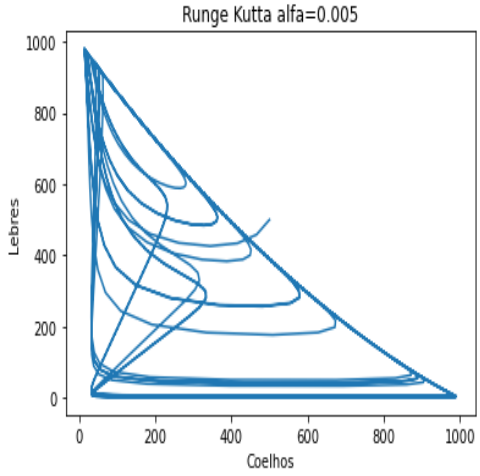

Figure 68: Retrato de fase CoelhosxLebres

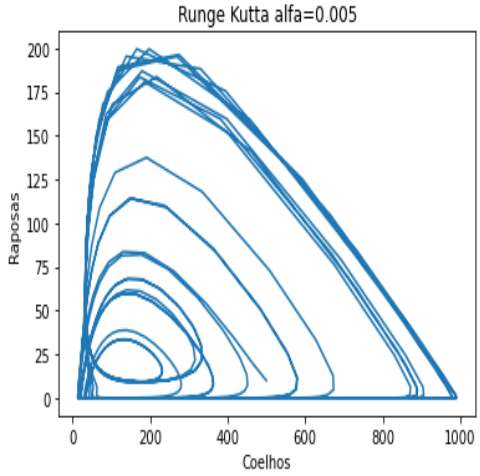

Figure 69: Retrato de fase CoelhosxRaposas

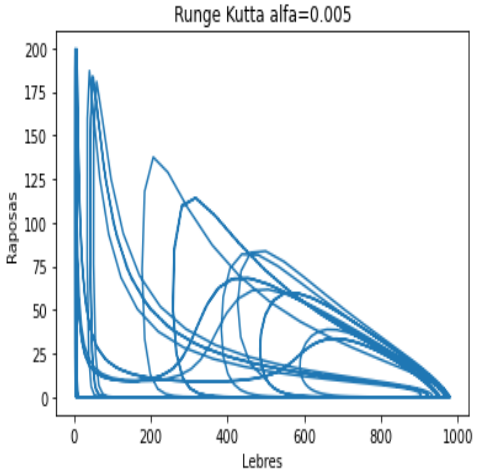

Figure 70: Retrato de fase LebresxRaposas

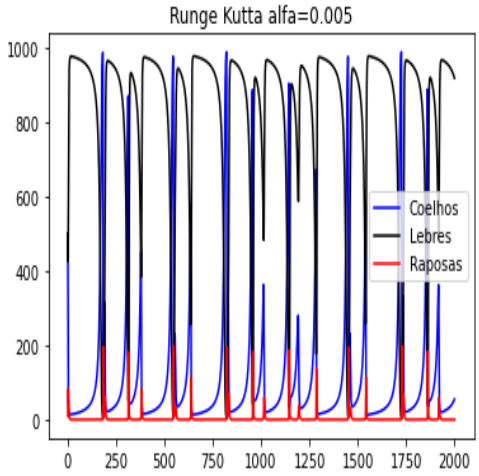

Figure 71: Evolução das três espécies

3.6 alfa = 0.0055

Aqui observamos overflow no método de Euler, confirmando que de fato a população de lebres explodiu por conta da preferência das raposas por coelhos.

As órbitas do método de Runge Kutta já são também bastante instáveis mas ainda periódicas, mostrando que estamos no limite do poder de aproximação deste robusto método numérico.

3.6.1 Método de Euler Explícito

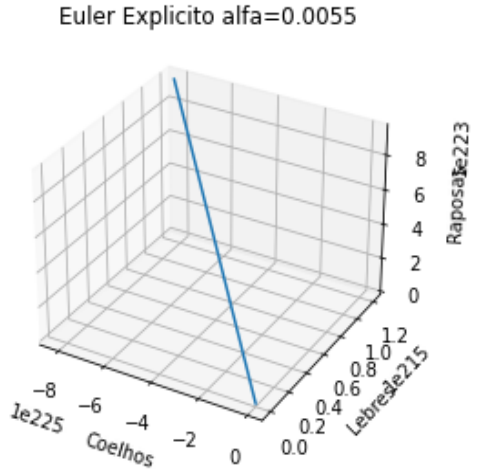

Figure 72: Retrato de fase 3d

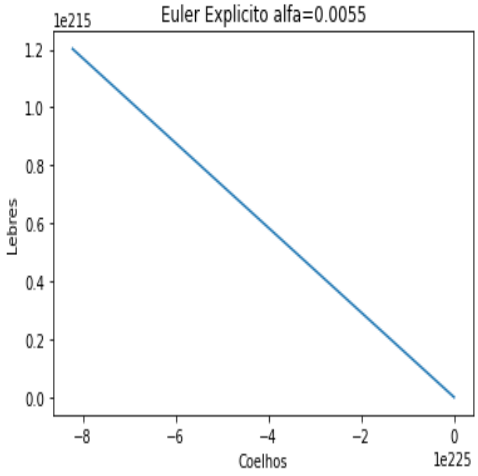

Figure 73: Retrato de fase CoelhosxLebres

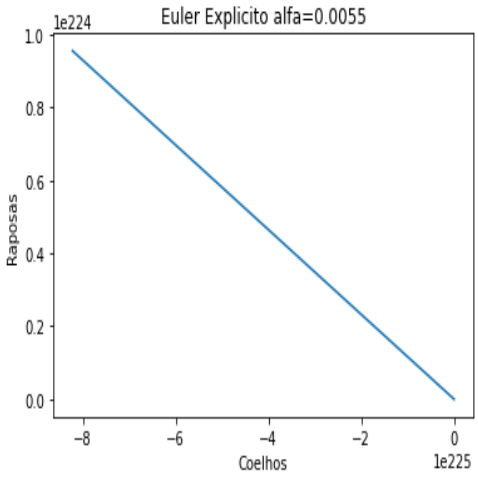

Figure 74: Retrato de fase CoelhosxRaposas

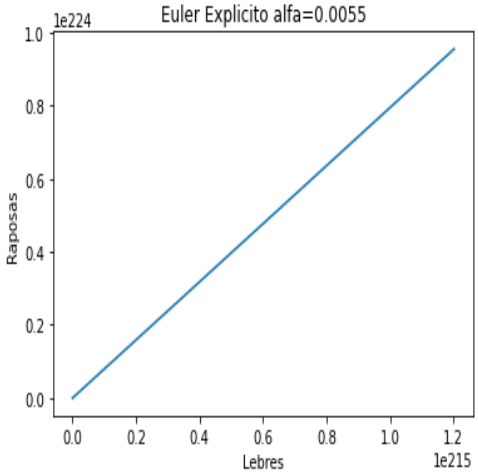

Figure 75: Retrato de fase LebresxRaposas

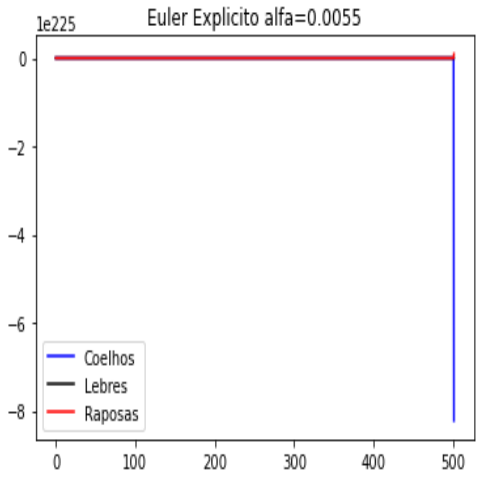

Figure 76: Evolução das três espécies

3.6.2 Método de Runge Kutta de ordem quatro

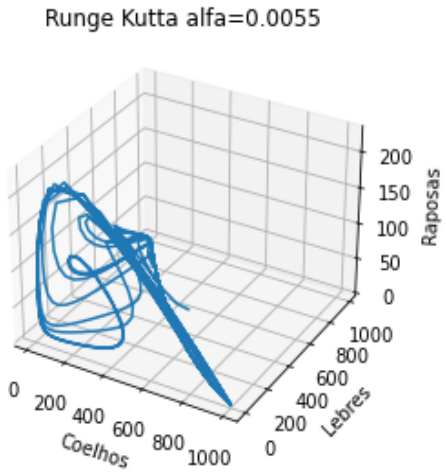

Figure 77: Retrato de fase 3d

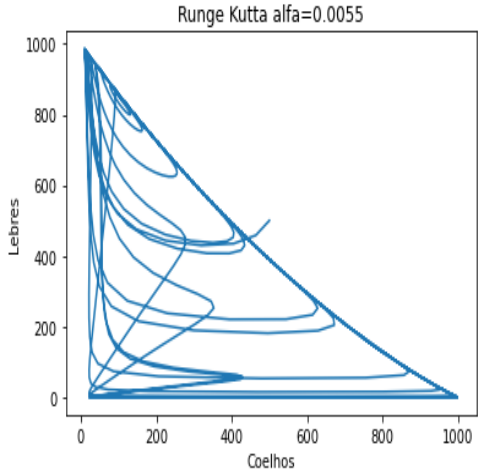

Figure 78: Retrato de fase CoelhosxLebres

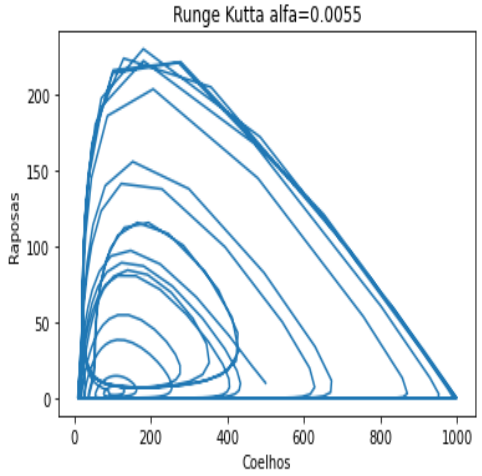

Figure 79: Retrato de fase CoelhosxRaposas

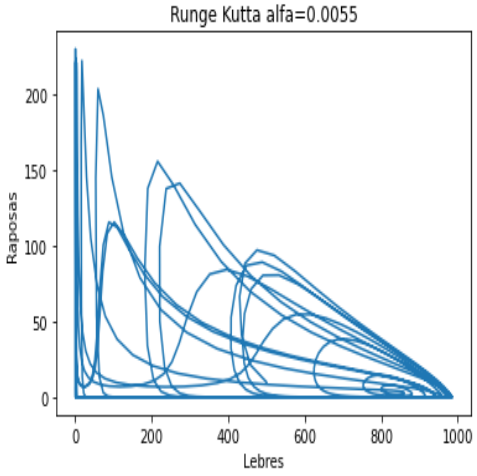

Figure 80: Retrato de fase LebresxRaposas

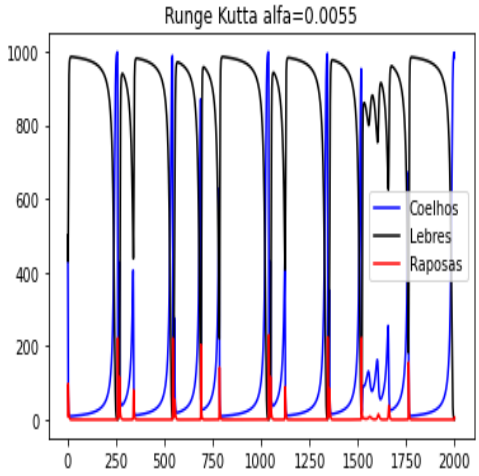

Figure 81: Evolução das três espécies

3.7 Teste de estabilidade

Ao realizar o teste de estabilidade me lembrei de um artigo que li no livro *An Invitation to Mathematics* sobre a estabilidade de sistemas dinâmicos. O artigo abordava sistemas de dois ou mais corpos orbitando o Sol.

Caso a razão entre os períodos de rotação fosse racional, os planetas estavam em ressonância e sua interação tornava a dinâmica do sistema instável.

É um exemplo de como a estabilidade de órbitas em sistemas dinâmicos pode depender de relações bastante delicadas.

No nosso caso, observamos como mudanças no parâmetro alfa podem levar a explosão na população de lebres prejudicando o equílibrio estável que inicialmente víamos no sistema.

Neste teste de instabilidade vamos verificar duas coisas

1. a explosão da população de lebres

2. a instabilidade do Modelo de Euler comparado ao Modelo de Runge Kutta. Note que para o valor de alfa utilizado Runge Kutta ainda é bastante confiável.

População	Euler $n = 75$	Runge Kutta $n = 75$	Euler $n = 74$	Runge Kutta $n = 74$
Coelhos	47.84	41.67	74.81	38.07
Lebres	929.86	938.72	890	943.94
Raposas	0.01	0.01	0.32	0.0

Table 2: Valores finais do teste de estabilidade

Note que as mudanças são muito menores no método de Runge Kutta. Vale observar que erros de aproximação são acentuadas quando utilizamos um intervalo grande, uma vez que erros são propagados através de um grande número de passos.

Ao analisar a evolução das populações, vale observar como os patamares são mais largos no Método de Euler com $n = 75$. Existe uma resistência maior à queda da população de lebres que eventualmente leva à sua explosão.

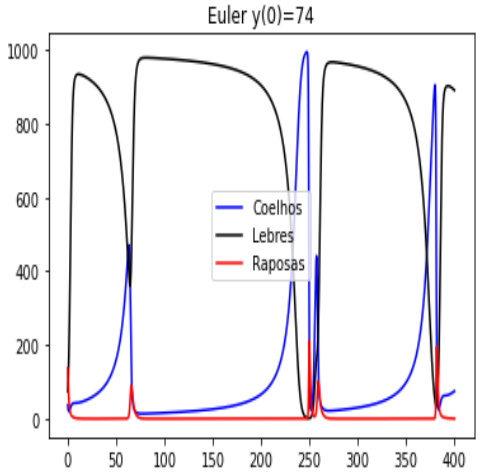

Figure 82: Evolução das três espécies Euler $n = 74$

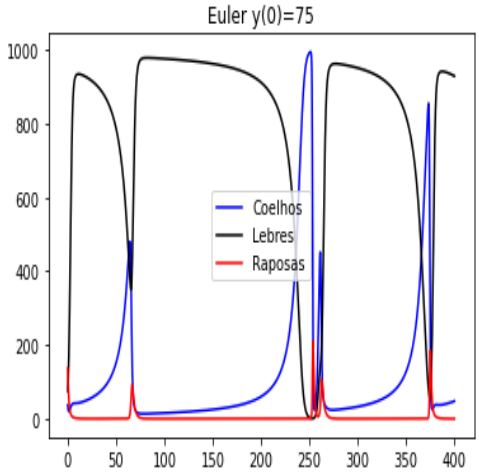

Figure 83: Evolução das três espécies Euler $n = 75$

4 Referências Bibliográficas

1. web.cs.ucdavis.edu **A guide to writing Mathematics**
2. oeis.org **List of LaTeX mathematical symbols**
3. courses.csail.mit.edu **Basic Plotting with Python and Matplotlib**
4. matplotlib.org **Parametric Curve**
5. LACKMANN, Malte; SCHLEICHER, Dierk **An Invitation to Mathematics**