Relatório do Projeto 2 de Cálculo Numérico

Gabriel Volpato Giliotti - 197569
Erick Kussumoto do Nascimento - 196492

Problema:

A equação logística, estudada primeiramente por Pierre-François Verhulst em 1838, modela um cenário em que a taxa de crescimento populacional é proporcional à população existente e à quantidade de recursos disponíveis no meio. Desta forma, o crescimento da população não se dá de forma indefinida, mas está limitado a certas condições ambientais, ou seja, os indivíduos competem entre si por alimento e recursos necessários para a sobrevivência. Seja y(t) o número de indivíduos no instante t , então o crescimento populacional é representado pela equação diferencial ordinária:

$$y'(t) = r \cdot y(t) \cdot \left(1 - \frac{y(t)}{K} \right)$$

em que r é a taxa de reprodução da população e K é a constante de capacidade do meio, que é o maior valor atingido pela população para o tempo tendendo ao infinito. A solução analítica para este problema de valor inicial é dada por:

$$y(t) = \frac{K \cdot y0 \cdot e^{r \cdot t}}{K + y0 \cdot (e^{r \cdot t} - 1)}$$

na qual y0 é o número de indivíduos no instante inicial.

Introdução:

No projeto utilizamos a linguagem de programação python para desenvolver as devidas soluções para o problema. No código, procuramos dividir cada questão através de uma linha de “#” e ao chamar cada método, uma lista de resultados são gerados e são utilizados para fundamentar o relatório.

Podemos notar que a equação logística dada no enunciado é uma EDO de primeira ordem, já que a função possui somente uma derivada em função de duas variáveis, sendo “y” a variável dependente e “t” a variável independente.

O objetivo do relatório é resolver a EDO dada através de métodos numéricos ensinados em sala e comparar seus resultados com a solução analítica de forma a tirar conclusões sobre a precisão e confiabilidade dos métodos.

Questões:

a) ***Método de Euler:***

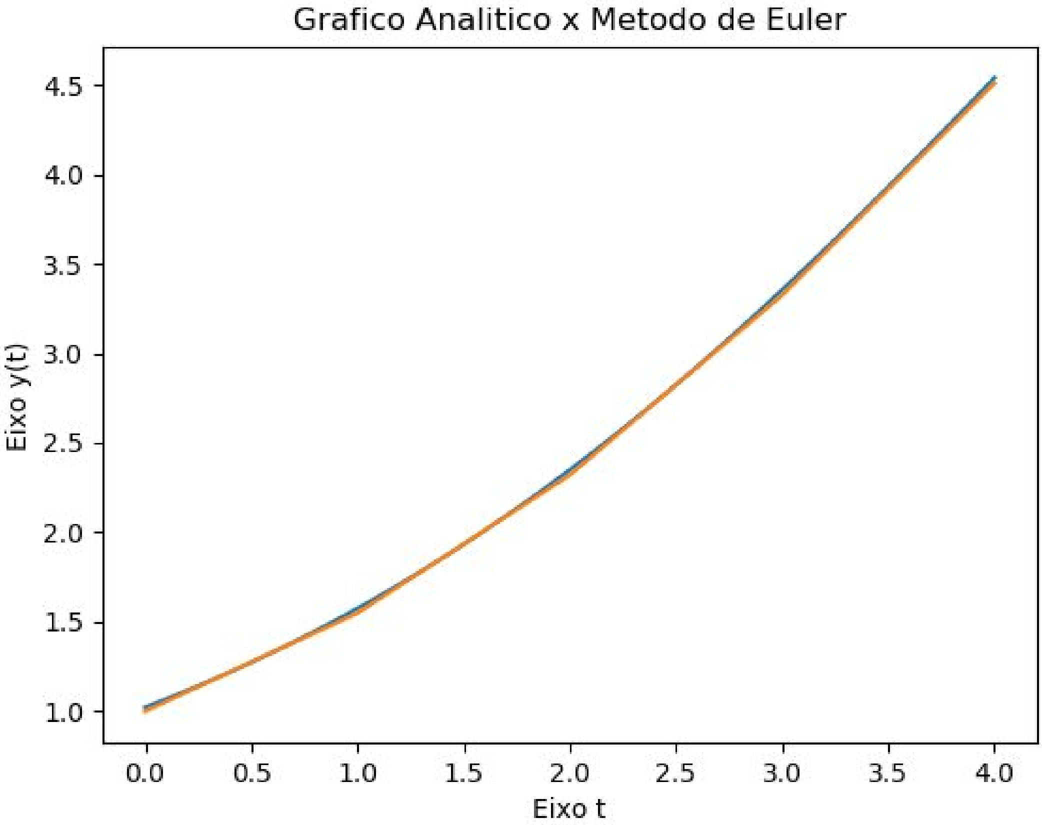
No método de Euler implementamos o seguinte código para resolver a EDO em questão:

```
def metodoEuler(ini,fim,r,K,h,y0):
        t = ini
        y_de_t = y0
        it = 1

        while(t < fim):
                y_n_mais_um = y_de_t + (h * (r * y_de_t * (1.0 - (y_de_t/K))))
                print("Valor na iteracao " + str(it) + " = " + str(y_n_mais_um) + " e Valor de t = " + str(t))

                it = it + 1
                t = t + h
                y_de_t = y_n_mais_um

        return y_n_mais_um
```

(OBS: Para ter acesso ao código comepleto com comentarios acesse *https://github.com/GabrielGiliotti/MS211/blob/master/projeto2.py*)

Obtivemos então, o resultado “Valor na iteração 81 = 4.53907692437 e Valor de t = 4.0” que representa y(4). Valor na iteração aqui é apenas um contador que foi acrescentado ao código para monitorarmos a quantidade de informação. Ou seja, y(4) = 4.53907692437.

Na solução analítica, substituímos o parâmetro t = 4 para comparar seu resultado com os dados obtidos pelo método, encontrando o valor de y(4) = 4.508530604 . Através dos cálculos, podemos concluir que o método implementado possui uma precisão de apenas uma casa decimal levando em consideração os parâmetros dados no enunciado.



b) Para o item b) o primeiro parâmetro alterado foi o passo “h” e estes foram os seguintes resultados correspondentes:

Para h = 0.01 , encontramos y(4) = 4.51464500
Para h = 0.005 , encontramos y(4) = 4.51158813
Para h = 0.001 , encontramos y(4) = 4.50914216
Para h = 0.0005 , encontramos y(4) = 4.50821743
Para h = 0.0001 , encontramos y(4) = 4.50846797

Esses dados indicam como a redução do valor do passo (valor de h) influencia diretamente na precisão do método em relação ao obtido na solução analítica. Ao incluir um valor menor de “h”, será obtido um valor mais próximo, enquanto o contrário gera um resultado menos preciso sobre a EDO.

O segundo parâmetro estudado foi a constante “r” que caracteriza a taxa de reprodução da população:

r = 60, encontramos y(4) = 1.61329457
r = 50, encontramos y(4) = 12.24996169
r = 40, encontramos y(4) = 10.36179155

r = 10, encontramos y(4) = 10.00000000
r = 5, encontramos y(4) = 9.99999999
r = 2, encontramos y(4) = 9.97738886
r = 1, encontramos y(4) = 8.63649962
r = 0.5, encontramos y(4) = 4.53907692
r = 0.05 encontramos y(4) = 1.19737114
r = 0.00000005, encontramos y(4) = 1.00000018

Observando os dados podemos verificar que a taxa de reprodução faz com que a população cresça até determinados níveis, e, ao chegar ao limite, tenha uma queda brusca no crescimento populacional pois, dado o esgotamento de recursos, podemos dizer que a população diminuirá e começará a crescer novamente após um tempo. Repare que quando r = 50 e quando r = 60: em r = 60 o limite foi excedido e em t = 4 a população cresce em uma taxa de 1.61329, já com uma taxa de reprodução menor r = 50, o crescimento populacional está em uma taxa de crescimento de 12.24996 para o mesmo t = 4 . Além disso, podemos afirmar que, ao adotar pequenas taxas de reprodução, a população mantém-se em torno de um máximo de valor baixo devido à sobra de recursos. Observe que, quando r = 0.5 temos uma taxa de crescimento em 1.19737 e quando r = 0.00000005 temos uma taxa de crescimento 1.00000.

Por último, alteramos o valor de “K” que representa a constante de capacidade do meio. Para melhor verificar as consequências da alteração da capacidade do meio, também variamos o tempo e obtivemos os seguintes dados:

K = 10000000, encontramos y(4) = 7.38980240
K = 10000000, encontramos y(10) = 9999999.99999996
K = 0.75, encontramos y(4) = 0.77475305
K = 0.75, encontramos y(10) = 0.75117873

Através desses dados, podemos afirmar que o aumento do valor de K pode permitir o aumento da taxa de crescimento populacional de forma infinita, de acordo com o decorrer do tempo. Já a redução faz com que o valor de y se mantenha em um valor baixo. Esse comportamento pode ser explicado simplesmente pelo fato de que a população não consegue aumentar em uma escassez de recursos e que, sob uma elevada, mas finita, disponibilidade de recursos, pode haver um limite máximo de capacidade do meio que a população consegue utilizar. Essa utilização vai de acordo com a taxa de reprodução.

Sendo assim, podemos afirmar que o método de Euler continua sendo confiável, já que retorna valores bem próximos à solução analítica. O único porém é que sua precisão depende do valor do passo “h” que altera o número de iterações e cálculos realizados.

c) *Método de Runge-Kutta de Ordem 4:*

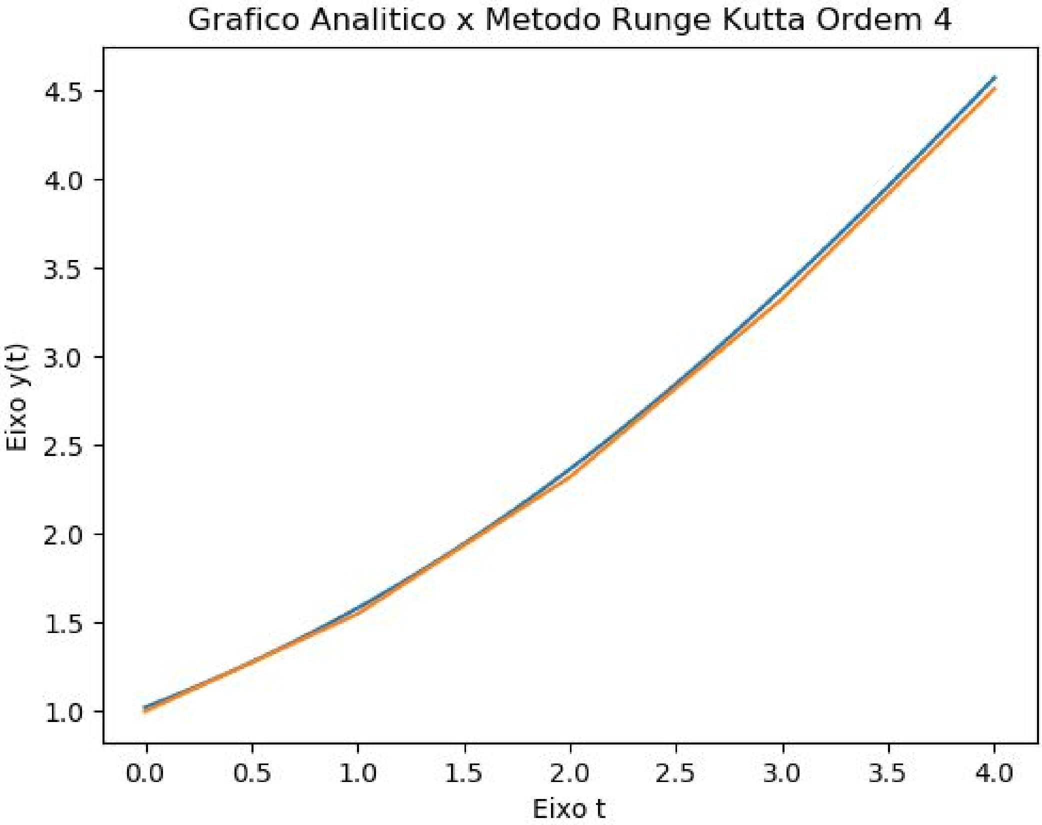
No método de Runge-Kutta de Ordem 4, implementamos os seguinte código para resolver a EDO em questão:

```
def RungeKuttaOrdem4(ini, fim , r, K, h, y0):
    tk = ini
    yk = y0
    it = 1
    while( tk < fim ):

        k1 = h*(r * yk *(1 - (yk/K)))

        tkparak2 = tk + (h/2.0)
        ykparak2 = yk + (k1/2.0)
        k2 = h*(r * ykparak2 * (1 - (ykparak2/K)))

        tkparak3 = tk + (h/2.0)
        ykparak3 = yk + (k2/2.0)
        k3 = h*(r * ykparak3 * (1 - (ykparak3/K)))

        tkparak4 = tk + h
        ykparak4 = yk + k3
        k4 = h*(r * ykparak4 * (1 - (ykparak4/K)))

        yk_mais_um = yk + ((1/6.0)*(k1+(2*k2)+(2*k3)+k4))
        tk = tk + h
        yk = yk_mais_um

        print("Valor na iteracao " + str(it) + " = " + str(yk_mais_um) + " e Valor de t = " + str(tk) + "\n")
        it = it + 1
```

(OBS: Para ter acesso ao código comepleto com comentarios acesse *https://github.com/GabrielGiliotti/MS211/blob/master/projeto2.py*)

Utilizando os mesmo parâmetros do item a) para o método de Runge-Kutta percebemos uma melhor aproximação para um mesmo intervalo, isto é, enquanto o método de Euler aproximou com precisão de uma casa decimal, o método de Runge-Kutta de ordem 4 aproximou com precisão de 6 casas decimais, mostrando-se mais preciso para um mesmo número de iterações.

Analisando em seguida, para o intervalo [0,10], obtivemos uma maior precisão com os mesmos outros parâmetros da entrada, obtendo y(10) = 9.428256186 para a solução analitica e y(10) = 9.42825618294 utilizando o método, mostrando que o método converge em até 8 casa decimais, com um maior número de iterações.



d) O método que se mostrou mais condizente com a solução analítica foi o método de Runge-Kutta, sendo que o método de Euler possui precisão de somente uma casa decimal e o de Runge-Kutta de 6 casas decimais no exemplo dado.

Conclusão:

Após a implementação de ambos os métodos, pudemos analisar através dos dados obtidos que o método de Runge-Kutta de Ordem 4 é o método mais preciso. O método retorna com uma precisão de 6 casas decimais em um mesmo número de iterações, porém isso não exclui o fato de que o método de Euler ainda é bastante confiável e também gera uma boa aproximação.

Como afirmado anteriormente, a mudança de parâmetros afeta a população em um comportamento esperado, porém não gera resultados inesperados usando o método de Euler. Esse fato confirma que o método numérico é bom para resolver EDOs, mesmo sendo menos preciso.

Referências:

[1] CHENEY, W. , KINGAID, D. Numerical Mathematics and Computing. 6th Edition. Thomson Brooks/Cole: 2008.

[2] QUARTERONI, A., SALERI, F., GERVASIO, P. Scientific Computing with Matlab and Octave. Third Edition. Springer: 2010.

[3] WOODFORD, C., PHILLIPS, C. Numerical Methods with Worked Examples: Matlab Edition. Second Edition. Springer: 2012.

[4] YANG, W.Y., CAO, W., CHUNG, T., MORRIS, J. Applied Numerical Methods Using Matlab. First Edition. John Wiley and Sons: 2005.

[5] DRISCOLL, T.A. Learning Matlab. First Edition. SIAM: 2009.