

RELATÓRIO SOBRE AS IMPLEMENTAÇÕES

*Relatório expondo as implementações dos métodos numéricos solicitadas pelo
Professor Doutor Ricardo Martins de Abreu Silva*

Vitor Matheus de Azevedo Martins

09.11.2017

IF816 - Métodos Numéricos Computacionais

INTRODUÇÃO

Na implementação dos métodos foram usadas as bibliotecas **matplotlib.pyplot** para fazer a plotagem dos dados no gráfico, assim como para exibi-lo, a biblioteca **math** foi utilizada para abranger funções como logarítmica, trigonométricas, entre outras e a biblioteca **decimal** para trabalhar com os números decimais, visto que possui uma precisão melhor que o float (inclusive erros de precisão no float que me levaram a usar esta biblioteca).

Na imagem ao lado temos cada termo das iterações no método de euler para uma devida função, onde o t_n deveria ser somado em $h = 0.025$ a cada iteração. O uso de float levou a esta imprecisão que acabaria trazendo resultados insatisfatórios para um número de iterações muito grande. O problema foi sanado ao utilizar `Decimal()` no lugar de float.

Entretanto operações entre float e Decimal não são suportadas, então se torna necessário que o usuário digite qualquer decimal ou uma função de **math** como argumento de `Decimal()`. Além disso, para as funções de **math** é necessário utilizar o prefixo `math..` É preciso que o usuário digite explicitamente os operadores. Para o número de Euler deve-se utilizar `math.exp()`. Qualquer divergência da entrada com o que foi citado neste documento poderá causar um erro.

Documentação:

<https://docs.python.org/2/library/math.html>

<https://docs.python.org/2/library/decimal.html>

<https://matplotlib.org/users/>

As implementações também podem ser encontradas no [meu repositório do Github](#).

```
Type the h: 0.025
Type the tf: 1.5
n = 60.0
y(0.025)= 1.125
y(0.05)= 1.261875
y(0.075000000000000001)= 1.4118125000000001
y(0.1)= 1.5761187500000002
y(0.125)= 1.7562306250000002
y(0.15)= 1.9537286875000002
y(0.175)= 2.17035155625
y(0.19999999999999998)= 2.408011711875
y(0.22499999999999998)= 2.6688128830625
y(0.24999999999999997)= 2.95506917136875
y(0.27499999999999997)= 3.2693260885056246
y(0.3)= 3.614383697356187
y(0.325)= 3.993322067091806
y(0.35000000000000003)= 4.409529273800986
y(0.37500000000000006)= 4.866732201181085
y(0.40000000000000001)= 5.369030421299193
y(0.42500000000000001)= 5.920933463429113
y(0.45000000000000001)= 6.527401899772024
y(0.47500000000000004)= 7.1938919907492265
y(0.50000000000000001)= 7.926496189824149
y(0.52500000000000001)= 8.731546898896565
y(0.55000000000000002)= 9.61657648968722
y(0.57500000000000002)= 10.589484138655942
y(0.60000000000000002)= 11.659057552521537
y(0.62500000000000002)= 12.834963307773691
y(0.65000000000000002)= 14.127834638551061
y(0.67500000000000003)= 15.549368102496167
y(0.70000000000000003)= 17.112429912646782
y(0.72500000000000003)= 18.831172903911458
y(0.75000000000000003)= 20.721165194302603
y(0.77500000000000004)= 22.799531713732865
y(0.80000000000000004)= 25.08510988510615
y(0.82500000000000004)= 27.598620873616763
y(0.85000000000000004)= 30.362857960978438
y(0.87500000000000004)= 33.402893757076285
y(0.90000000000000005)= 36.745308132783916
y(0.92500000000000005)= 40.423438946062305
y(0.95000000000000005)= 44.46765784066854
y(0.97500000000000005)= 48.91567362473539
y(1.00000000000000004)= 53.80786598720893
```

EXEMPLOS

Expressão	Input
$0.2y$	<code>Decimal(0.2)*y</code>
$\text{sen}(4t) + y$	<code>Decimal(math.sin(4*t)) + y</code>
$e^{3t} + 0.5y$	<code>Decimal(math.exp(3*t)) + Decimal(0.5)*y</code>
$1-t+4y$	<code>1 - t + 4*y</code>
$\ln(t+1)$	<code>Decimal(math.log(t+1))</code>

MÉTODO DE EULER

Para o método de Euler (euler.py), é pedido que o usuário digite inicialmente a função y' . Além disso é pedido um valor inicial de y , ou seja, t_0 e y_0 de forma que $y(t_0) = y_0$.

É dada a opção do usuário escolher entre dar como entrada o **h** (height) e o **número de passos** (n_steps), dar o **h** e o t_f (tf) desejado, ou ainda o **número de passos** e o t_f desejado. A terceira variável que o usuário optar por não digitar será computada posteriormente.

Considere a função $y' = 1 - t + 4y$; $y(0) = 1$ como exemplo para o nosso caso teste. Usaremos também um $h = 0,001$ e iremos até $y(2)$, ou seja, $t_f = 2$.

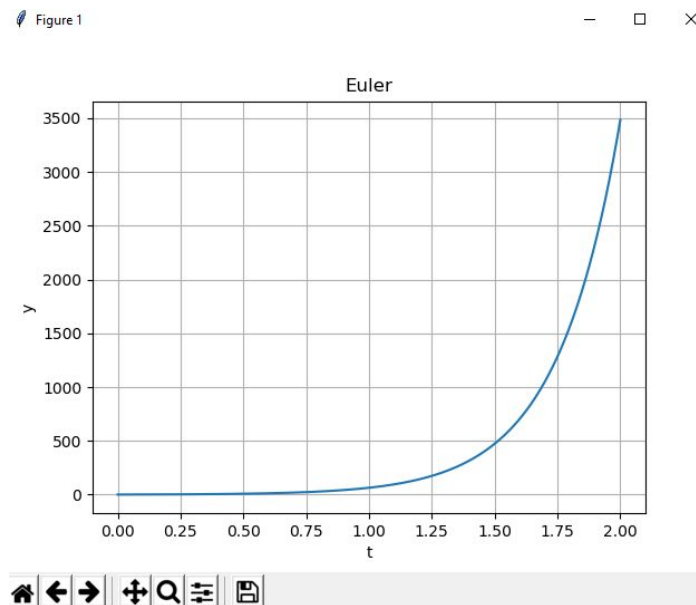
INPUT:

```
Type the function (y'): 1 - t + 4*y
Type the y0: 1
Type the t0: 0
1 - Enter height and number of steps(divisions)
2 - Enter height and the tf
3 - Enter number of steps(divisions) and the tf
2
Type the h: 0.001
Type the tf: 2
```

OUTPUT:

```
y(2) = 3484.160803076250053082217390
```

E a janela com o gráfico da plotagem é aberta:



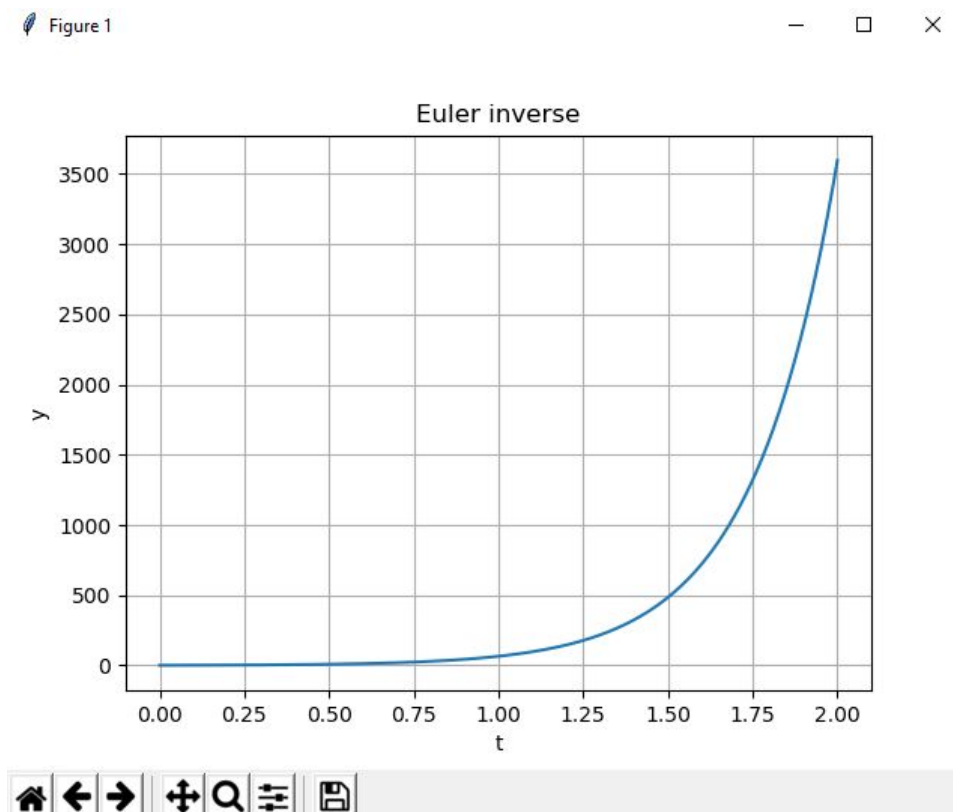
MÉTODO DE EULER INVERSO

O método de Euler inverso (euler_inverse.py) é feito de forma semelhante ao método de Euler. A entrada é dada da mesma forma. É necessário frisar que o y_{n+1} não foi feito implicitamente, e sim como uma aproximação do método de Euler.

Considere a função $y' = 1 - t + 4y$; $y(0) = 1$ como exemplo para o nosso caso teste. Usaremos também um $h = 0,001$ e iremos até $y(2)$, ou seja, $t_f = 2$.

```
Type the function (y'): 1 -t + 4*y
Type the y0: 1
Type the t0: 0
1 - Enter height and number of steps(divisions)
2 - Enter height and the tf
3 - Enter number of steps(divisions) and the tf
2
Type the h: 0.001
Type the tf: 2
y(2) = 3596.987375766046552649548241
```

E a janela com o gráfico da plotagem é aberta:



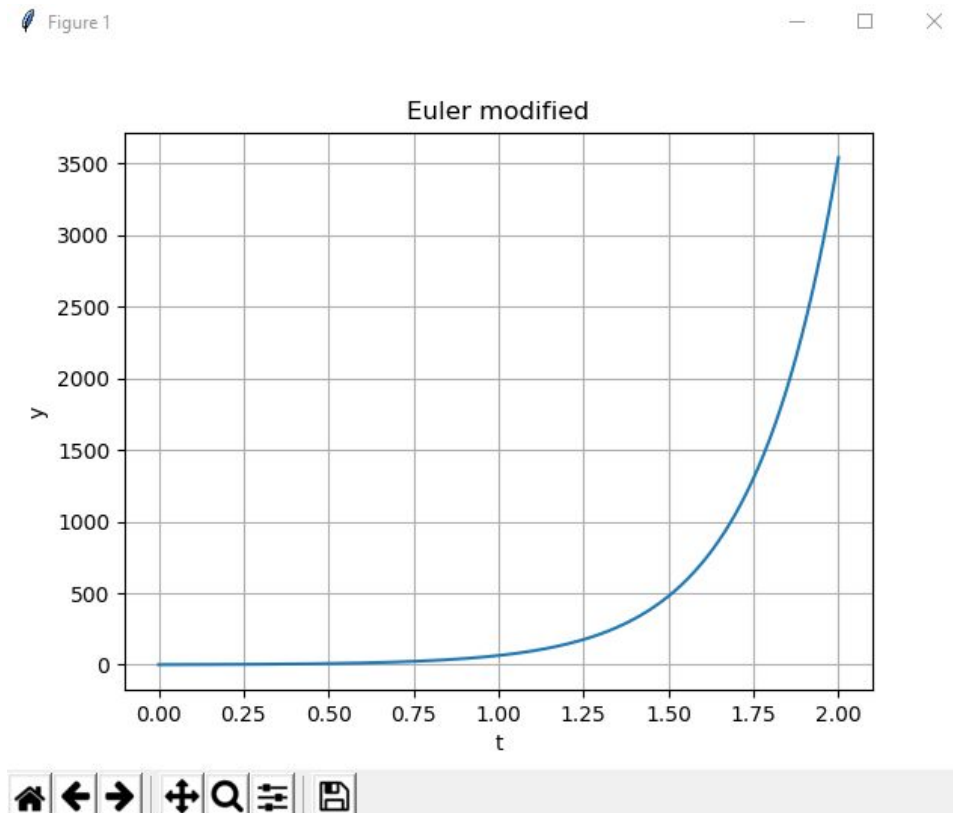
MÉTODO DE EULER MODIFICADO

O método de Euler modificado (euler_modified.py) também foi feito de forma semelhante ao método de Euler. A entrada se dá da mesma forma. Cada y_{n+1} é dado com base na média entre o y_n , encontrado anteriormente, e o y_{n+1} do método de Euler.

Considere a função $y' = 1 - t + 4y$; $y(0) = 1$ como exemplo para o nosso caso teste. Usaremos também um $h = 0,001$ e iremos até $y(2)$, ou seja, $t_f = 2$.

```
Type the function (y'): 1-t+4*y
Type the y0: 1
Type the t0: 0
Choose one:
1 - Enter height and number of steps(divisions)
2 - Enter height and the tf
3 - Enter number of steps(divisions) and the tf
2
Type the h: 0.001
Type the tf: 2
y(2) = 3540.124819000735771583460563
```

E a janela com o gráfico da plotagem é aberta:



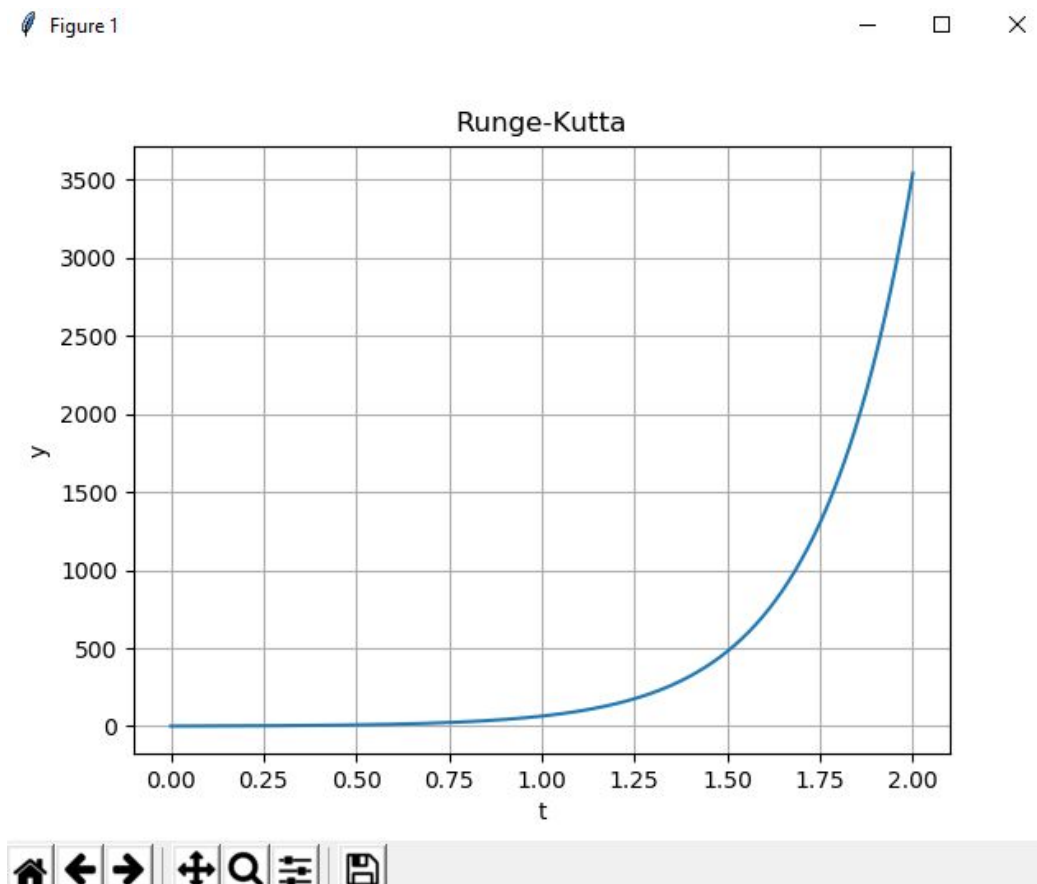
RUNGE-KUTTA

A entrada para o método de Runge-Kutta (runge_kutta.py) também é feita da mesma forma que o método de Euler.

Considere a função $y' = 1 - t + 4y$; $y(0) = 1$ como exemplo para o nosso caso teste. Usaremos também um $h = 0,001$ e iremos até $y(2)$, ou seja, $t_f = 2$.

```
Type the function (y'): 1-t+4*y
Type the y0: 1
Type the t0: 0
1 - Enter height and number of steps(divisions)
2 - Enter height and the tf
3 - Enter number of steps(divisions) and the tf
2
Type the h: 0.001
Type the tf: 2
y(2) = 3540.200109551839279839795312
```

E a janela com o gráfico da plotagem é aberta:



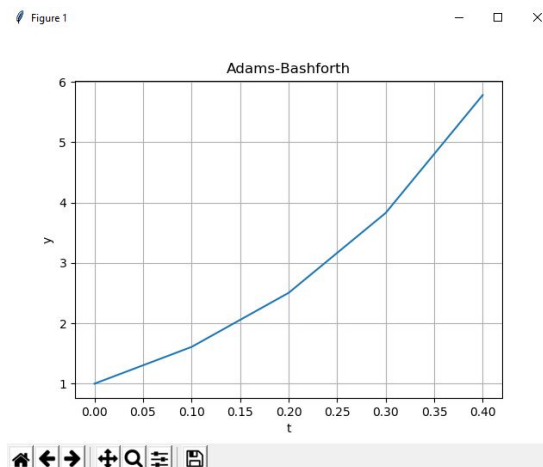
MÉTODO DE ADAMS-BASHFORTH

Para o método de Adams-Bashforth, além da função y' , do ponto inicial e das variáveis **height**, **n_steps** e/ou **tf**, é pedido que o usuário entre com mais alguns pontos, com a quantidade dependendo da ordem escolhida para o método. Por exemplo, para o método na segunda ordem, é necessário a entrada de mais um ponto, totalizando dois pontos. Para o método na sexta ordem, é necessário a entrada de mais cinco pontos, totalizando seis pontos. É dada ao usuário a opção de escolher qual ordem deseja.

Considere a função $y' = 1 - t + 4y$; $y(0) = 1$ como exemplo para o nosso caso teste. Usaremos também um $h = 0,1$ e iremos até $y(0,4)$, ou seja, $t_f = 0,4$.

```
Type the function (y'): 1-t+4'y
Type the y0: 1
Type the t0: 0
1 - Enter height and number of steps(divisions)
2 - Enter height and the tf
3 - Enter number of steps(divisions) and the tf
2
Type the h: 0.1
Type the tf: 0.4
Choose the order of Adams-Bashforth
1 - First order
2 - Second order
3 - Third order
4 - Fourth order
5 - Fifth order
6 - Sixth order
4
About the point (y1, f1):
Type the 'y' of the point: 1.6089333
About the point (y2, f2):
Type the 'y' of the point: 2.5050062
About the point (y3, f3):
Type the 'y' of the point: 3.8294145
y(0.4) = 5.78363056333333333333333333333333
```

E a janela com o gráfico da plotagem é aberta:



MÉTODO DE ADAMS-MOULTON

Para o método de Adams-Bashforth, além da função y' , do ponto inicial e das variáveis **height**, **n_steps** e/ou **tf**, é pedido que o usuário entre com mais alguns pontos, com a quantidade dependendo da ordem escolhida para o método. Por exemplo, para o método na terceira ordem, é necessário a entrada de mais um ponto, totalizando dois pontos, o terceiro ponto é calculado utilizando o método de Euler. Para o método na sexta ordem, é necessário a entrada de mais quatro pontos, totalizando cinco pontos, da mesma forma, o sexto ponto será calculado pelo método de Euler. É dada ao usuário a opção de escolher qual ordem deseja.

```
Type the function (y'): 1-t+4*y
Type the y0: 1
Type the t0: 0
1 - Enter height and number of steps(divisions)
2 - Enter height and the tf
3 - Enter number of steps(divisions) and the tf
2
Type the h: 0.1
Type the tf: 0.4
Choose the order of Adams-Moulton
1 - First order
2 - Second order
3 - Third order
4 - Fourth order
5 - Fifth order
6 - Sixth order
5
About the point (y1, f1):
Type the 'y' of the point: 1.6089333
About the point (y2, f2):
Type the 'y' of the point: 2.5050062
About the point (y3, f3):
Type the 'y' of the point: 3.8294145
y(0.4) = 5.742888564055555555555555555556
```

E a janela com o gráfico da plotagem é aberta:

