

Introdução à Física Computacional II (4300318)

Prof. André Vieira
apvieira@if.usp.br
Sala 3120 – Edifício Principal

Aula 6

Método do ponto médio modificado
Método de Bulirsch-Stoer

O algoritmo *leapfrog*

- Na aula passada, vimos que as equações de iteração (embora não o passo inicial!) do algoritmo *leapfrog* são **invariantes por reversão temporal**, e que o **erro em um passo é cúbico** no tamanho h do passo de integração.
- A invariância por reversão temporal implica que, em um passo (diferente do inicial!), a variação da variável dependente em direção ao “passado” é igual ao negativo da variação em direção ao “futuro”:

$$\Delta x_- = -\Delta x_+$$

O algoritmo *leapfrog*

- Na aula passada, vimos que as equações de iteração (embora não o passo inicial!) do algoritmo *leapfrog* são **invariantes por reversão temporal**, e que o **erro em um passo é cúbico** no tamanho h do passo de integração.
- Mas isso implica que o erro que cometemos na integração para o “passado” tem que ser o negativo do erro que cometemos na integração para o “futuro”, ou seja, o erro tem que ser uma função ímpar de h :

$$\Delta x_- = -\Delta x_+ \Rightarrow \epsilon(-h) = -\epsilon(h)$$

Para voltar ao valor anterior, um erro cometido “na ida” tem que ser revertido “na volta”!

O algoritmo *leapfrog*

- Na aula passada, vimos que as equações de iteração (embora não o passo inicial!) do algoritmo *leapfrog* são **invariantes por reversão temporal**, e que o **erro em um passo é cúbico** no tamanho h do passo de integração.
- O erro ser uma função ímpar de h , juntamente com o fato de que o erro é de ordem cúbica em h , implica que se pode escrever, para o erro em um único passo,

$$\epsilon(h) = c_3 h^3 + c_5 h^5 + c_7 h^7 + \dots$$

O algoritmo *leapfrog*

- Portanto, o erro **acumulado** em uma integração com o algoritmo *leapfrog* deveria ter uma expansão em h que contém apenas potências pares. Isso não ocorre porque o passo inicial, não sendo invariante por reversão temporal, tem uma expansão em h com potências tanto pares quanto ímpares.
- No entanto, há um esquema (W. Gragg, 1965) que elimina da expansão do erro acumulado as potências ímpares de h .

O método do ponto médio modificado

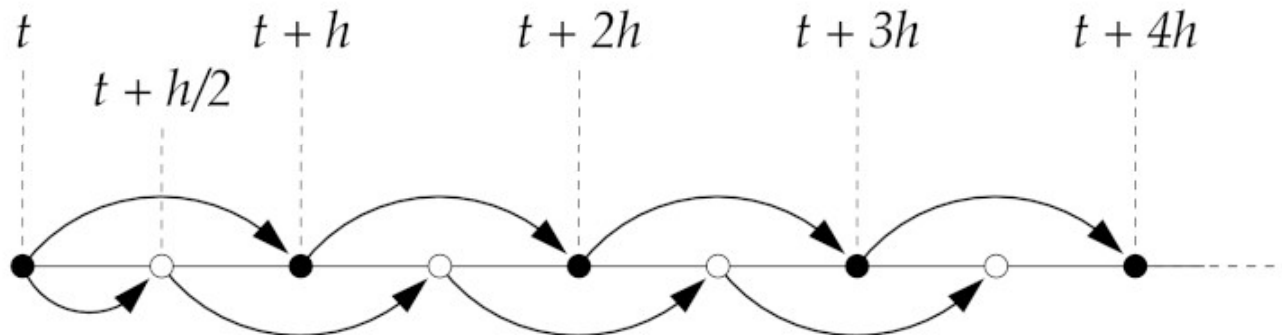
- Do algoritmo *leapfrog* (no caso 1D):

$$x(t+h) = x(t) + hf\left(x\left(t+\frac{1}{2}h\right), t+\frac{1}{2}h\right)$$

$$x\left(t+\frac{3}{2}h\right) = x\left(t+\frac{1}{2}h\right) + hf(x(t+h), t+h)$$

com o “pontapé” inicial

$$x\left(t+\frac{1}{2}h\right) = x(t) + \frac{1}{2}hf(x(t), t)$$



O método do ponto médio modificado

- Definindo

$$x(t+mh) \equiv x_m \quad x\left(t+\left(m+\frac{1}{2}\right)h\right) \equiv y_{m+1}$$

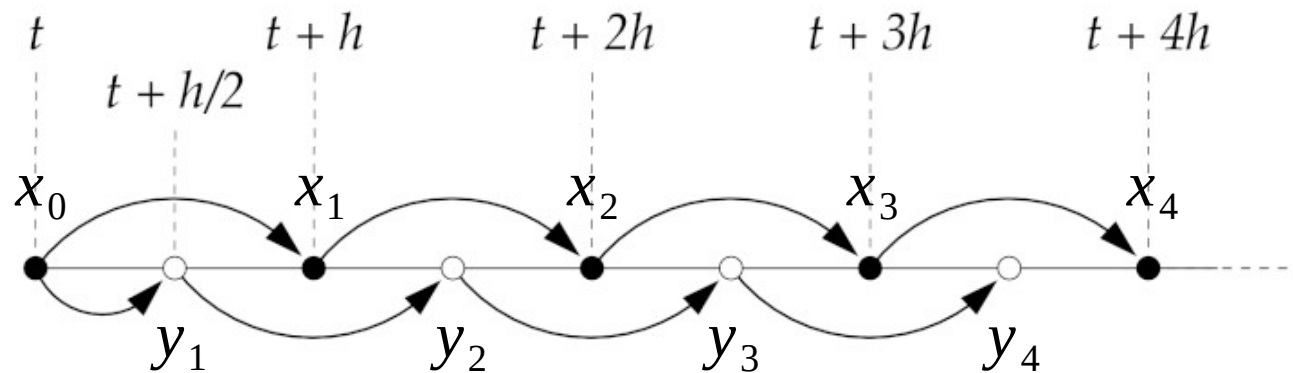
vem

$$y_{m+1} = y_m + hf(x_m, t+mh)$$

$$x_{m+1} = x_m + hf\left(y_{m+1}, t+\left(m+\frac{1}{2}h\right)\right)$$

com

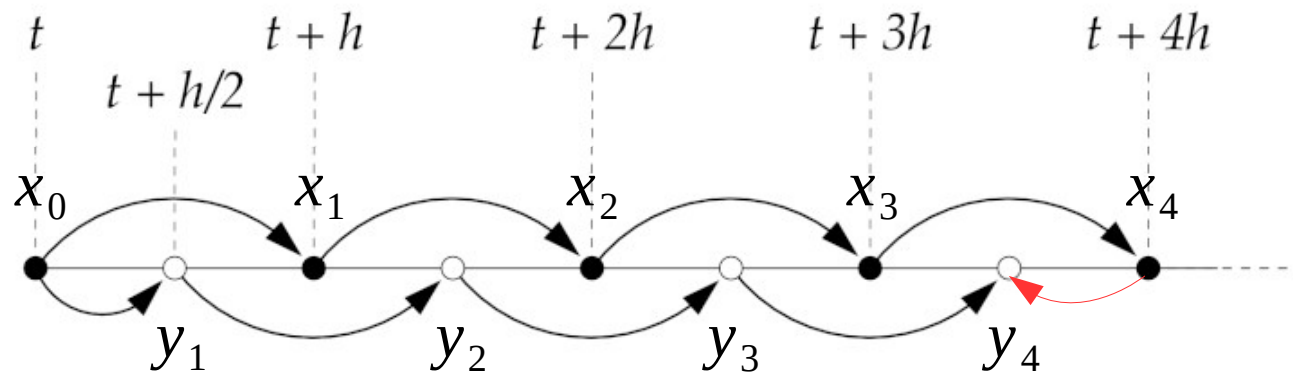
$$y_1 = x_0 + \frac{1}{2}hf(x_0, t)$$



O método do ponto médio modificado

- No algoritmo *leapfrog*, o valor $x(t+nh)$ da variável dependente ao final da integração seria estimado como o valor no último ponto inteiro, x_n (com $n=4$ na figura).
- A prescrição de Gragg envolve “espelhar” o passo inicial. Note que, idealmente, teríamos

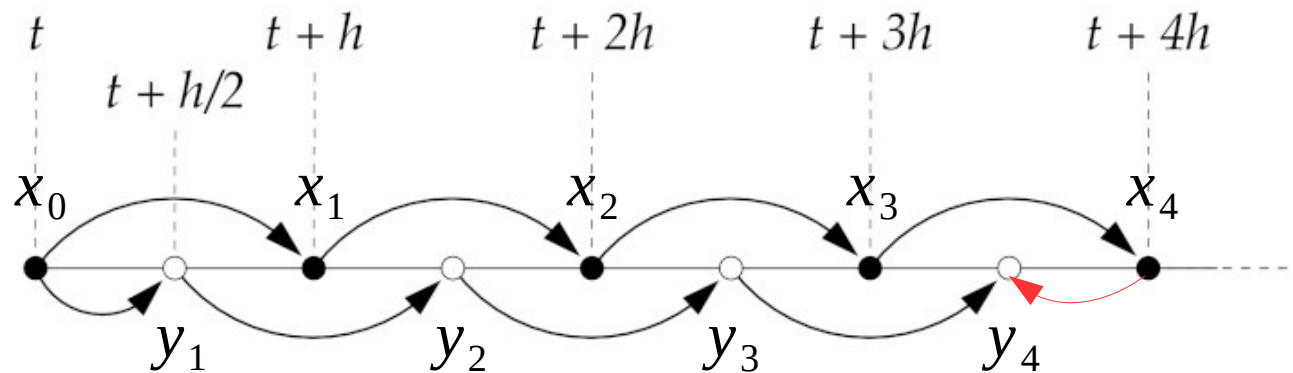
$$y_n = x(t+nh) - \frac{1}{2}h f(x(t+nh), t+nh)$$



O método do ponto médio modificado

- No algoritmo *leapfrog*, o valor $x(t+nh)$ da variável dependente ao final da integração seria estimado como o valor no último ponto inteiro, x_n (com $n=4$ na figura).
- Mas como não conhecemos o ponto final exato $x(t+nh)$, usamos x_n como aproximação em f :

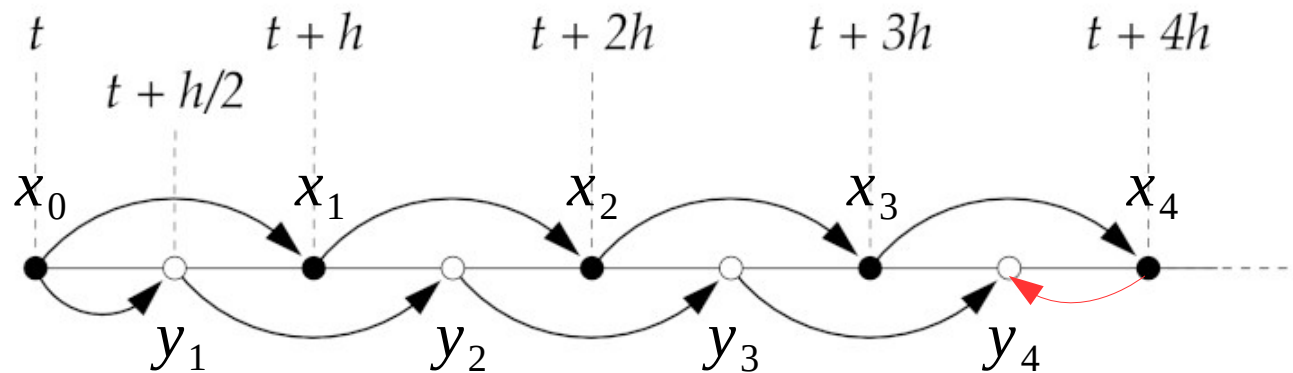
$$y_n = x(t+nh) - \frac{1}{2} h f(x_n, t+nh)$$



O método do ponto médio modificado

- No algoritmo *leapfrog*, o valor $x(t+nh)$ da variável dependente ao final da integração seria estimado como o valor no último ponto inteiro, x_n (com $n=4$ na figura).
- Como y_n também já havia sido calculado antes, uma segunda estimativa para $x(t+nh)$ é

$$x(t+nh) = y_n + \frac{1}{2} h f(x_n, t+nh)$$



O método do ponto médio modificado

- No algoritmo *leapfrog*, o valor $x(t+nh)$ da variável dependente ao final da integração seria estimado como o valor no último ponto inteiro, x_n (com $n=4$ na figura).
- Como y_n também já havia sido calculado antes, uma segunda estimativa para $x(t+nh)$ é

$$x(t+nh) = y_n + \frac{1}{2} h f(x_n, t+nh)$$

- Finalmente, a prescrição de Gragg é tomar a média entre essas duas estimativas:

$$x(t+nh) = \frac{1}{2} \left[x_n + y_n + \frac{1}{2} h f(x_n, t+nh) \right]$$

O método do ponto médio modificado

- Mas qual é a utilidade disso? Afinal, fazer uma pequena correção no último ponto calculado não passa de um preciosismo.

O método do ponto médio modificado

- Mas qual é a utilidade disso? Afinal, fazer uma pequena correção no último ponto calculado não passa de um preciosismo.
- Ocorre que ter um erro acumulado escrito apenas em termos de potências pares permite formular um esquema para eliminar esses erros sistematicamente, ganhando duas ordens de precisão em h a cada eliminação. Essa ideia é a base do **método de Bulirsch–Stoer**, cujo espírito é o mesmo do método de Romberg para o cálculo de integrais, estudado em Introdução à Física Computacional I.

O método de Bulirsch–Stoer

- Suponha que queiramos integrar uma equação diferencial ao longo de um intervalo, e que o passo de integração a partir de um certo valor t da variável independente seja H .
- Se utilizarmos o método do ponto médio modificado, com passo $h_1=H$, nossa estimativa para $x(t+H)$ a partir de $x(t)$ será $R_{1,1}$ tal que

$$x(t+H) = R_{1,1} + c_2 h_1^2 + O(h_1^4),$$

sendo c_2 uma constante, enquanto usando um passo $h_2=H/2$ nossa estimativa será $R_{2,1}$ tal que

$$x(t+H) = R_{2,1} + c_2 h_2^2 + O(h_2^4).$$

O método de Bulirsch–Stoer

- Mas $h_1=2h_2$, e portanto, até ordem quadrática,

$$x(t+H)=R_{2,1}+c_2 h_2^2=R_{1,1}+c_2 \times 4 h_2^2,$$

de onde segue que

$$c_2 h_2^2=\frac{1}{3}\left(R_{2,1}-R_{1,1}\right),$$

ou seja,

$$x(t+H)=R_{2,1}+\frac{1}{3}\left(R_{2,1}-R_{1,1}\right)+O\left(h_2^4\right),$$

o que fornece uma estimativa $R_{2,2}$ para $x(t+H)$ com erro de quarta ordem,

$$x(t+H)\simeq R_{2,2}=R_{2,1}+\frac{1}{3}\left(R_{2,1}-R_{1,1}\right).$$

O método de Bulirsch–Stoer

- Podemos ir adiante, utilizando um passo $h_3=H/3$ para obter uma estimativa $R_{3,1}$ tal que

$$x(t+H) = R_{3,1} + c_2 h_3^2 + O(h_3^4),$$

e a partir de

$$x(t+H) = R_{2,1} + c_2 h_2^2 + O(h_2^4),$$

com $h_2=3h_3/2$, segue, até ordem quadrática,

$$c_2 h_3^2 = \frac{4}{5} (R_{3,1} - R_{2,1}),$$

e portanto

$$x(t+H) = R_{3,1} + \frac{4}{5} (R_{3,1} - R_{2,1}) + O(h_3^4).$$

O método de Bulirsch–Stoer

- Escrevendo

$$R_{3,2} \equiv R_{3,1} + \frac{4}{5} (R_{3,1} - R_{2,1})$$

temos duas estimativas $R_{2,2}$ e $R_{3,2}$ para $x(t+H)$ associadas a um erro de quarta ordem,

$$x(t+H) = R_{2,2} + c_4 h_2^4 + O(h_2^6) = R_{3,2} + c_4 h_3^4 + O(h_3^6),$$

sendo c_4 uma constante, e usando $h_2 = 3h_3/2$ eliminamos os termos quárticos, obtendo

$$c_4 h_3^4 = \frac{16}{65} (R_{3,2} - R_{2,2}),$$

o que nos leva a um erro de sexta ordem:

$$x(t+H) = R_{3,2} + \frac{16}{65} (R_{3,2} - R_{2,2}) + O(h_3^6) \equiv R_{3,3} + O(h_3^6).$$

O método de Bulirsch–Stoer

- Isso pode ser levado a ordens arbitrárias. De

$$x(t+H) = R_{n,m} + c_{2m} h_n^{2m} + O(h_n^{2m+2}),$$

$$x(t+H) = R_{n-1,m} + c_{2m} h_{n-1}^{2m} + O(h_{n-1}^{2m+2}),$$

com c_{2m} constante, e de $h_{n-1} = nh_n/(n-1)$, obtemos

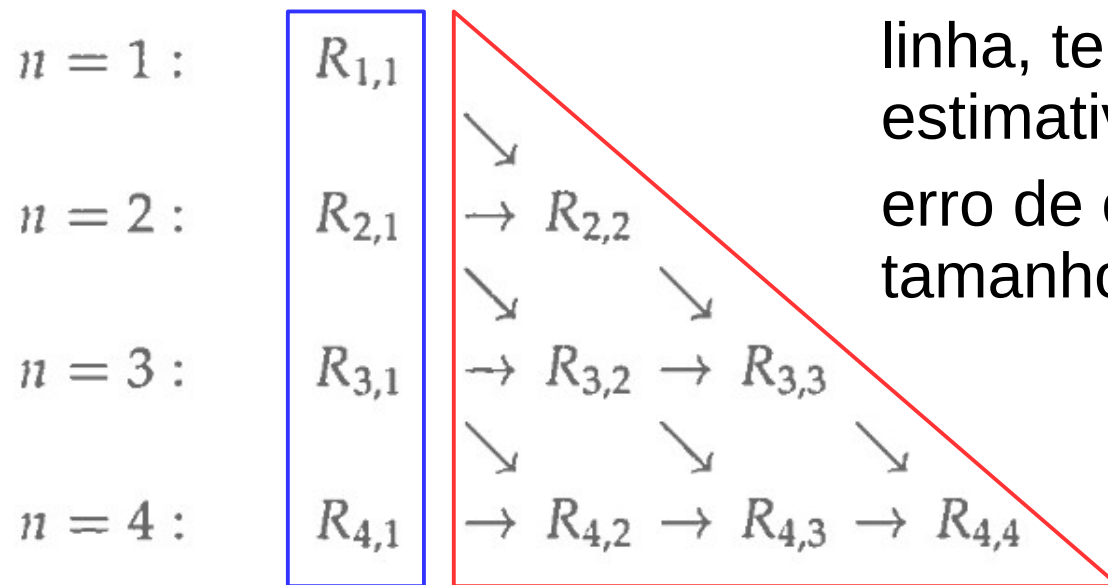
$$c_{2m} h_n^{2m} = \frac{R_{n,m} - R_{n-1,m}}{\left[n/(n-1)\right]^{2m} - 1},$$

que nos leva a uma estimativa para $x(t+H)$ com erro de ordem $2m+2$ no tamanho do passo:

$$x(t+H) = R_{n,m} + \frac{R_{n,m} - R_{n-1,m}}{\left[n/(n-1)\right]^{2m} - 1} + O(h_n^{2m+2}).$$

O método de Bulirsch–Stoer

- O esquema pode então ser sintetizado pela figura abaixo.



Ao final da n -ésima linha, temos uma estimativa $R_{n,n}$ com um erro de ordem $2n$ no tamanho do passo.

Método do ponto médio modificado

$$R_{n,m+1} = R_{n,m} + \frac{R_{n,m} - R_{n-1,m}}{\left[n/(n-1) \right]^{2m} - 1}$$

O método de Bulirsch–Stoer

- Como decidir quantos passos realizar?
- Comparamos a estimativa do erro $\epsilon \equiv R_{n,n} - R_{n,n-1}$ associado ao termo $R_{n,n}$ com uma precisão δ por unidade de t e ficamos satisfeitos quando ϵ for menor do que $H\delta$.
- Note que da relação de recorrência

$$R_{n,m+1} = R_{n,m} + \frac{R_{n,m} - R_{n-1,m}}{\left[n/(n-1)\right]^{2m} - 1}$$

fica claro que não precisamos armazenar toda a tabela de valores de $R_{n,m}$, mas apenas as duas linhas mais recentes.

Exemplo 1

$$\frac{d^2 \theta}{dt^2} = -\frac{g}{L} \sin \theta$$

$$\theta(0) = \frac{179\pi}{180},$$

$$\dot{\theta}(0) = 0$$

$$m=1, \quad L=0.1$$

$$g=9.8$$

$$H=0.1$$

$$\delta = 10^{-8}$$

note a definição
do erro

```
def passo_mbs(f,r,t,H,prec): # Calcula um passo no método de Bulirsch-Stoer
# Inicializamos com um passo do método do ponto médio modificado
# A matriz R1 armazena a primeira linha da tabela de extrapolação.
# Por agora, essa linha contém apenas a estimativa do método do
# ponto médio modificado para a solução no final do intervalo.
n = 1
y = r + 0.5*H*f(r,t)
x = r + H*f(y,t+0.5*H)
R1 = empty([1,r.shape[0]],float)
R1[0] = 0.5*(y + x + 0.5*H*f(x,t+H))
# Agora fazemos um laço aumentando o valor de n até que a precisão
# seja atingida.
erro = 2*H*prec # Garantindo que o laço seja executado ao menos 1 vez
while erro > H*prec:
    n += 1
    h = H/n
    # Método do ponto médio modificado
    y = r + 0.5*h*f(r,t)
    x = r + h*f(y,t+0.5*h)
    for i in range(n-1):
        y += h*f(x,t+(i+1.0)*h)
        x += h*f(y,t+(i+1.5)*h)
    # Calculando as estimativas por extrapolação.
    # As matrizes R1 e R2 armazenam a penúltima e a última
    # linhas mais recentes da tabela
    R2 = empty([n,r.shape[0]],float)
    R2[0] = 0.5*(y + x + 0.5*h*f(x,t+h))
    for m in range(1,n):
        epsilon = (R2[m-1]-R1[m-1])/((n/(n-1))**(2*m)-1)
        R2[m] = R2[m-1] + epsilon
    erro = abs(epsilon[0])
    R1 = R2
    # Fazemos r igual à estimativa mais precisa de que dispomos
    r = R2[n-1]
    return r # Retornamos o NOVO VALOR de r
```

calcula $R_{1,1}$

precisão foi atingida?

executado ao menos 1 vez

calcula $R_{n,1}$

calcula $R_{n,m+1}$

Exemplo 1

$$\frac{d^2 \theta}{dt^2} = -\frac{g}{L} \sin \theta$$

$$\theta(0) = \frac{179\pi}{180},$$

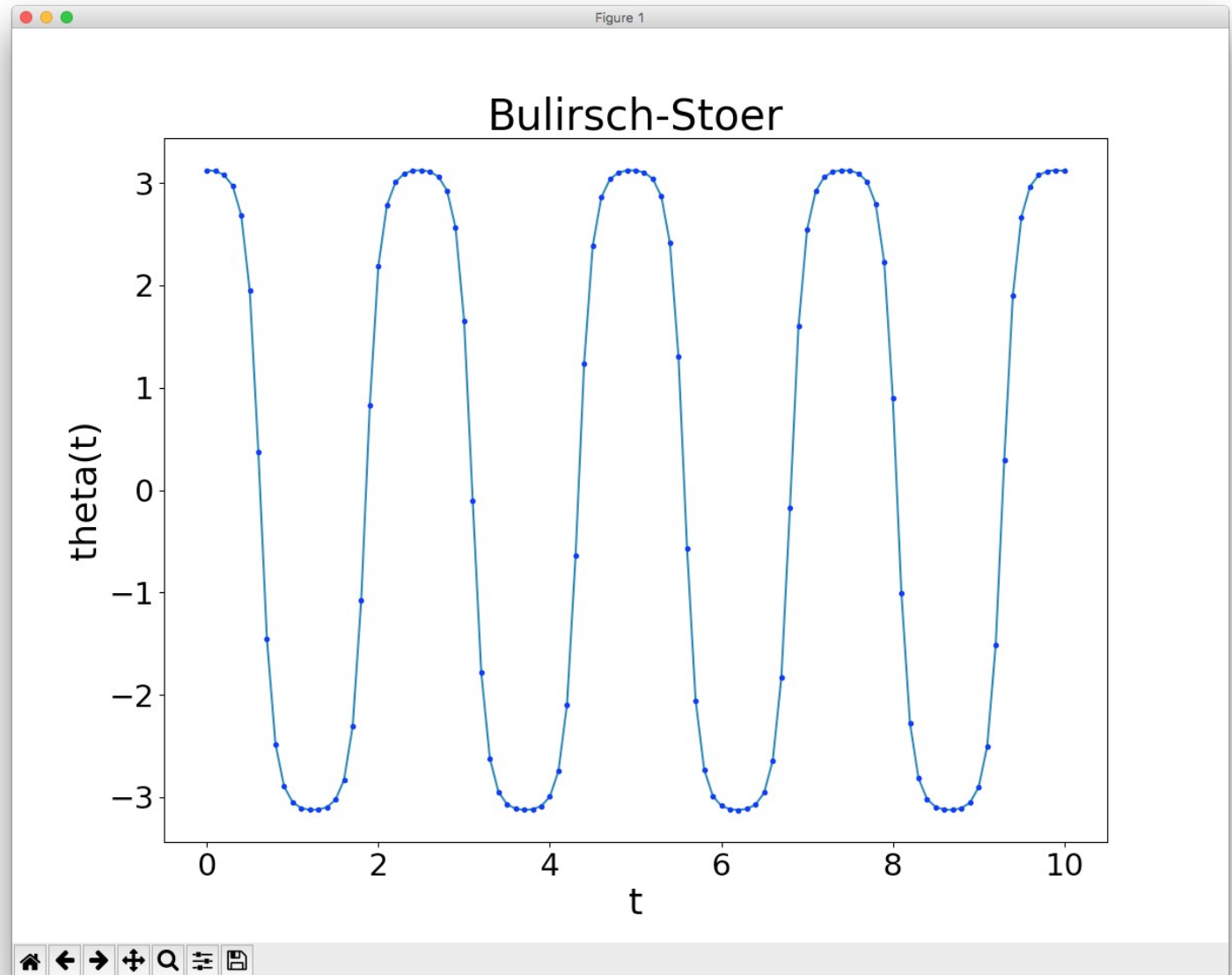
$$\dot{\theta}(0) = 0$$

$$m=1, \quad L=0.1$$

$$g=9.8$$

$$H=0.1$$

$$\delta=10^{-8}$$



O tempo de execução com o método de Bulirsch-Stoer é de 0.1 segundo.

Exemplo 1

$$\frac{d^2 \theta}{dt^2} = -\frac{g}{L} \sin \theta$$

$$\theta(0) = \frac{179\pi}{180},$$

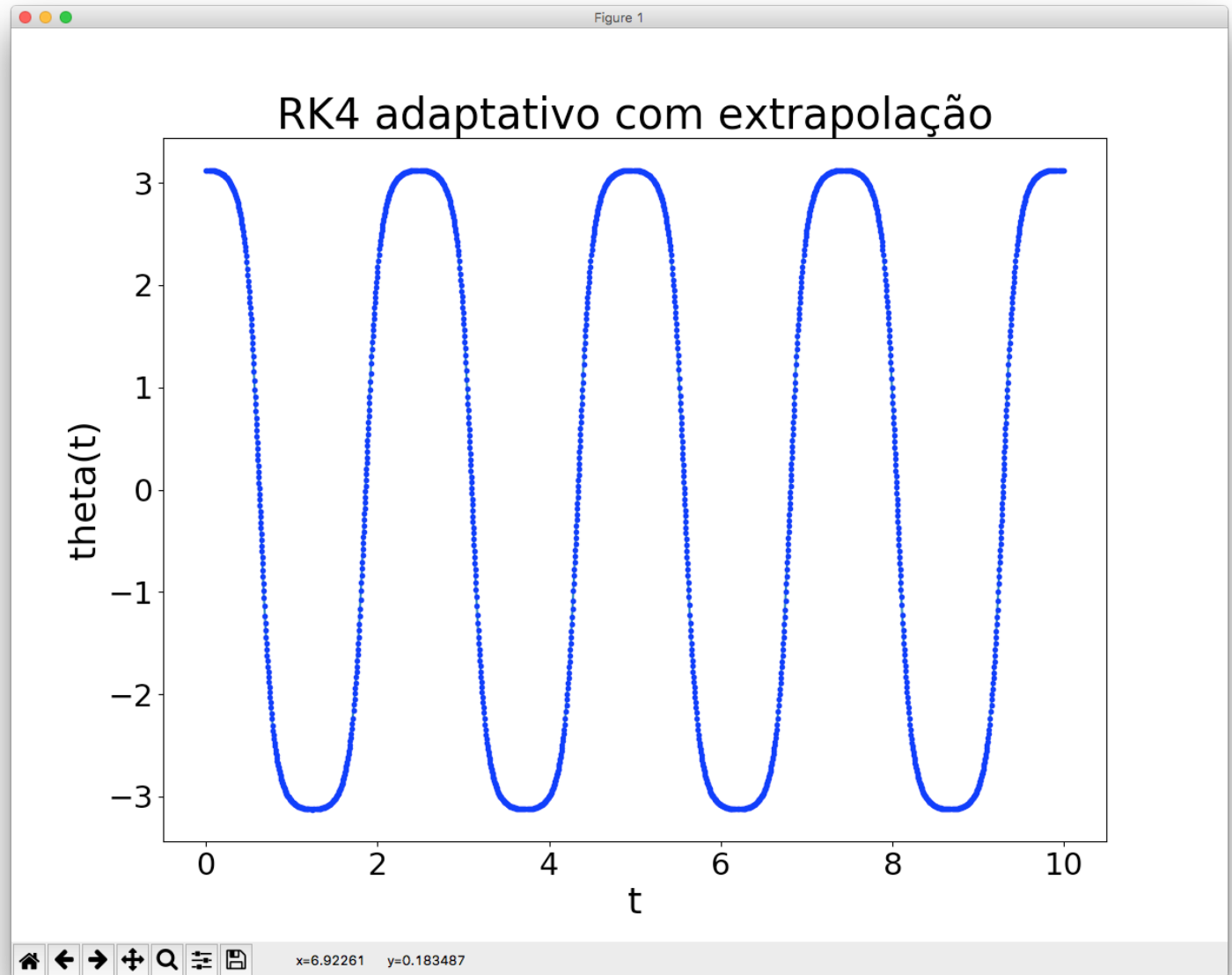
$$\dot{\theta}(0) = 0$$

$$m = 1, \quad L = 0.1$$

$$g = 9.8$$

$$h(0) = 0.1$$

$$\delta = 10^{-8}$$



Utilizando o método adaptativo com extrapolação local baseado no algoritmo de Runge-Kutta de quarta ordem, exigindo a mesma precisão, o tempo de execução aumenta para 0.5 segundo.

O método de Bulirsch–Stoer

- A precisão do método está associada aos valores calculados no final de cada passo. Os erros associados a valores calculados em pontos intermediários de um intervalo não são estimados.
- Usar um tamanho de passo muito grande pode exigir muitas iterações do algoritmo para se atingir a precisão exigida, levando a erros de arredondamento e a problemas com “overflow”. Tente executar o problema do exemplo 1 com tamanho de passo 0.4, sem o comentário da linha 62, e verifique.

Método de Bulirsch–Stoer adaptativo

- A escolha do tamanho ideal de passo pode ser automatizada limitando a n_{\max} o número de iterações do algoritmo em um mesmo passo. Se esse número máximo for atingido sem que a precisão desejada seja atingida, o tamanho de um certo passo é dividido por 2 e uma nova tentativa.
- Uma implementação possível dessa ideia é oferecida pelo exemplo seguinte.

Exemplo 2

$$\frac{d^2 \theta}{dt^2} = -\frac{g}{L} \sin \theta$$

$$\theta(0) = \frac{179\pi}{180},$$

$$\dot{\theta}(0) = 0$$

$$m=1, \quad L=0.1$$

$$g=9.8$$

$$H=0.4$$

$$\delta=10^{-8}$$

$$n_{\max}=10$$

informamos sobre a convergência

```
def passo_mbs_indiv(f, r, t, H, prec, nmax):  
    # Calcula um passo no método de Bulirsch-Stoer. Caso se atinjam  
    # 'nmax' iterações sem convergência, retorna a informação  
    # Inicializamos com um passo do método do ponto médio modificado  
    # A matriz R1 armazena a primeira linha da tabela de extrapolação.  
    # Por agora, essa linha contém apenas a estimativa do método do  
    # ponto médio modificado para a solução no final do intervalo.  
    converge = False  
    n = 1  
    y = r + 0.5*H*f(r, t)  
    x = r + H*f(y, t+0.5*H)  
    R1 = empty([1, r.shape[0]], float)  
    R1[0] = 0.5*(y + x + 0.5*H*f(x, t+H))  
    # Agora fazemos um laço aumentando o valor de n até que a precisão  
    # seja atingida.  
    for n in range(2, nmax+1):  
        h = H/n  
        # Método do ponto médio modificado  
        y = r + 0.5*h*f(r, t)  
        x = r + h*f(y, t+0.5*h)  
        for i in range(n-1):  
            y += h*f(x, t+(i+1.0)*h)  
            x += h*f(y, t+(i+1.5)*h)  
        # Calculando as estimativas por extrapolação.  
        # As matrizes R1 e R2 armazenam a penúltima e a última  
        # linhas mais recentes da tabela  
        R2 = empty([n, r.shape[0]], float)  
        R2[0] = 0.5*(y + x + 0.5*h*f(x, t+h))  
        for m in range(1, n):  
            epsilon = (R2[m-1]-R1[m-1])/((n/(n-1))**(2*m)-1)  
            R2[m] = R2[m-1] + epsilon  
            erro = abs(epsilon[0])  
            if erro <= H*prec:  
                converge = True  
                break  
        R1 = R2  
    # Fazemos r igual à estimativa mais precisa de que dispomos  
    r = R2[n-1]  
    return converge, r # Retornamos o NOVO VALOR de r
```

criamos um teste de convergência

fazemos no máximo 'nmax' iterações

se a convergência for atingida, saímos do laço

Exemplo 2

$$\frac{d^2 \theta}{dt^2} = -\frac{g}{L} \sin \theta$$

$$\theta(0) = \frac{179\pi}{180},$$

$$\dot{\theta}(0) = 0$$

$$m = 1, \quad L = 0.1$$


$$g = 9.8$$

$$H = 0.4$$

$$\delta = 10^{-8}$$

$$n_{\max} = 10$$

```
def passo_mbs_adapt(f,H,prec,nmax,r_lista,t_lista):  
    # Calcula um passo no método de Bulirsch-Stoer adaptativo.  
    # Esta função não retorna nenhum valor, mas apenas atualiza as listas  
    # de t e de r ao atingir convergência em até 'nmax' iterações  
    r = r_lista[-1]  
    t = t_lista[-1]  
    converge, r = passo_mbs_indiv(f,r,t,H,prec,nmax)  
    if converge == False: # Se não houve convergência, divida o passo por 2  
        passo_mbs_adapt(f,H/2,prec,nmax,r_lista,t_lista)  
    else:  
        t_lista.append(t+H)  
        r_lista.append(r)
```



note o uso recursivo
da função

```
def integ_mbs_adapt(f,r_a,a,b,H,prec,nmax,r_lista,t_lista):  
    # Esta função percorre o intervalo de integração, determinando os  
    # valores de r e t com passo máximo de tamanho H, que é subdividido  
    # caso não se atinja a precisão requerida em até 'nmax' iterações  
    # do algoritmo de Bulirsch-Stoer. A função não retorna um valor,  
    # mas atualiza as listas de r e t.  
    t = a  
    t_lista.append(t) # Registramos o valor inicial de t  
    r_lista.append(r_a) # Registramos o valor inicial de r  
    while t < b:  
        passo_mbs_adapt(f,H,prec,nmax,r_lista,t_lista)  
        t = t_lista[-1] # Atualizamos t para o último valor calculado  
  
    # Invocando a integração e criando as listas para traçar os gráficos  
    r_a = array([theta_a,omega_a],float) # Condição inicial  
    r_lista, t_lista = [], []  
    integ_mbs_adapt(f,r_a,a,b,H,prec,nmax,r_lista,t_lista)  
  
    theta_mbs = []  
    h_lista = []  
    for i in range(len(t_lista)):  
        theta_mbs.append(r_lista[i][0])  
        h_lista.append(t_lista[i]-t_lista[i-1])  
    h_lista[0] = h_lista[1]
```

Exemplo 2

$$\frac{d^2 \theta}{dt^2} = -\frac{g}{L} \sin \theta$$

$$\theta(0) = \frac{179\pi}{180},$$

$$\dot{\theta}(0) = 0$$

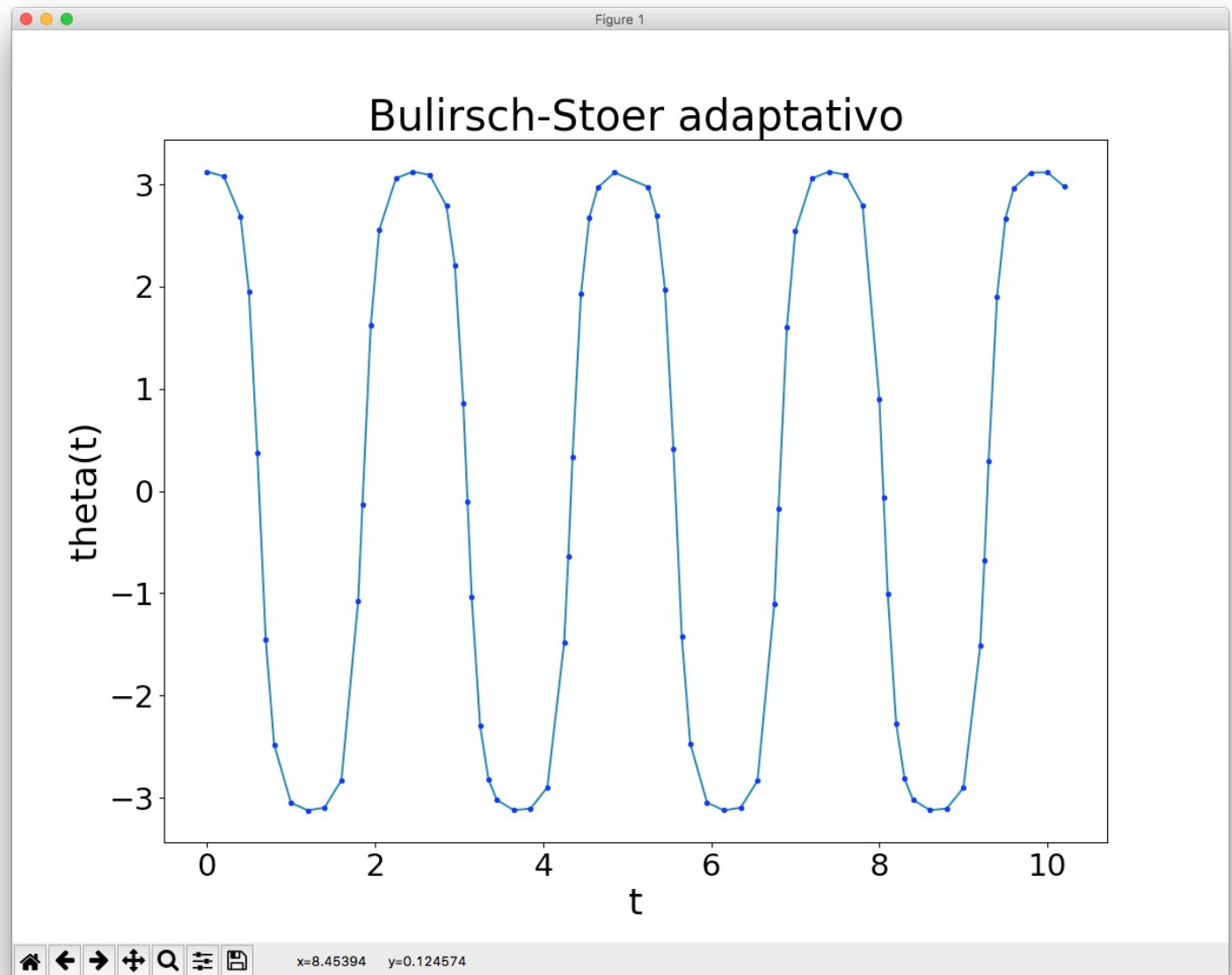
$$m=1, \quad L=0.1$$

$$g=9.8$$

$$H=0.4$$

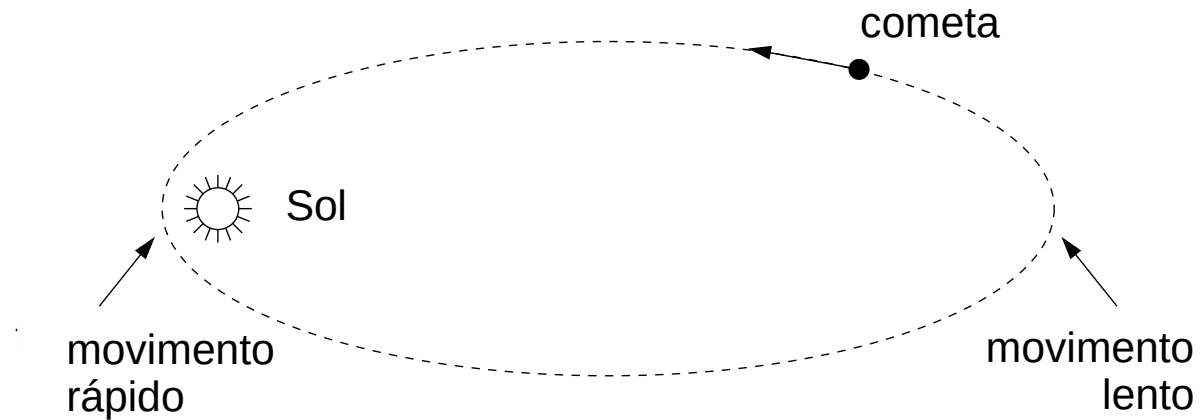
$$\delta=10^{-8}$$

$$n_{\max}=10$$



Exemplo 3: órbita de um cometa

Vamos considerar um cometa de massa m em órbita elíptica em torno do Sol, de massa M . Descrever esse movimento de forma eficiente requer um método adaptativo, para que o erro seja pequeno nas regiões em que o cometa se move rapidamente e não precisarmos executar muitos passos à toa nas regiões em que o cometa se move muito lentamente.



$$m \frac{d^2 \vec{r}}{dt^2} = - \frac{G m M}{|\vec{r}|^2} \hat{r}$$

Movimento no plano xy :

$$\frac{d^2 x}{dt^2} = -GM \frac{x}{r^3}, \quad \frac{d^2 y}{dt^2} = -GM \frac{y}{r^3}$$

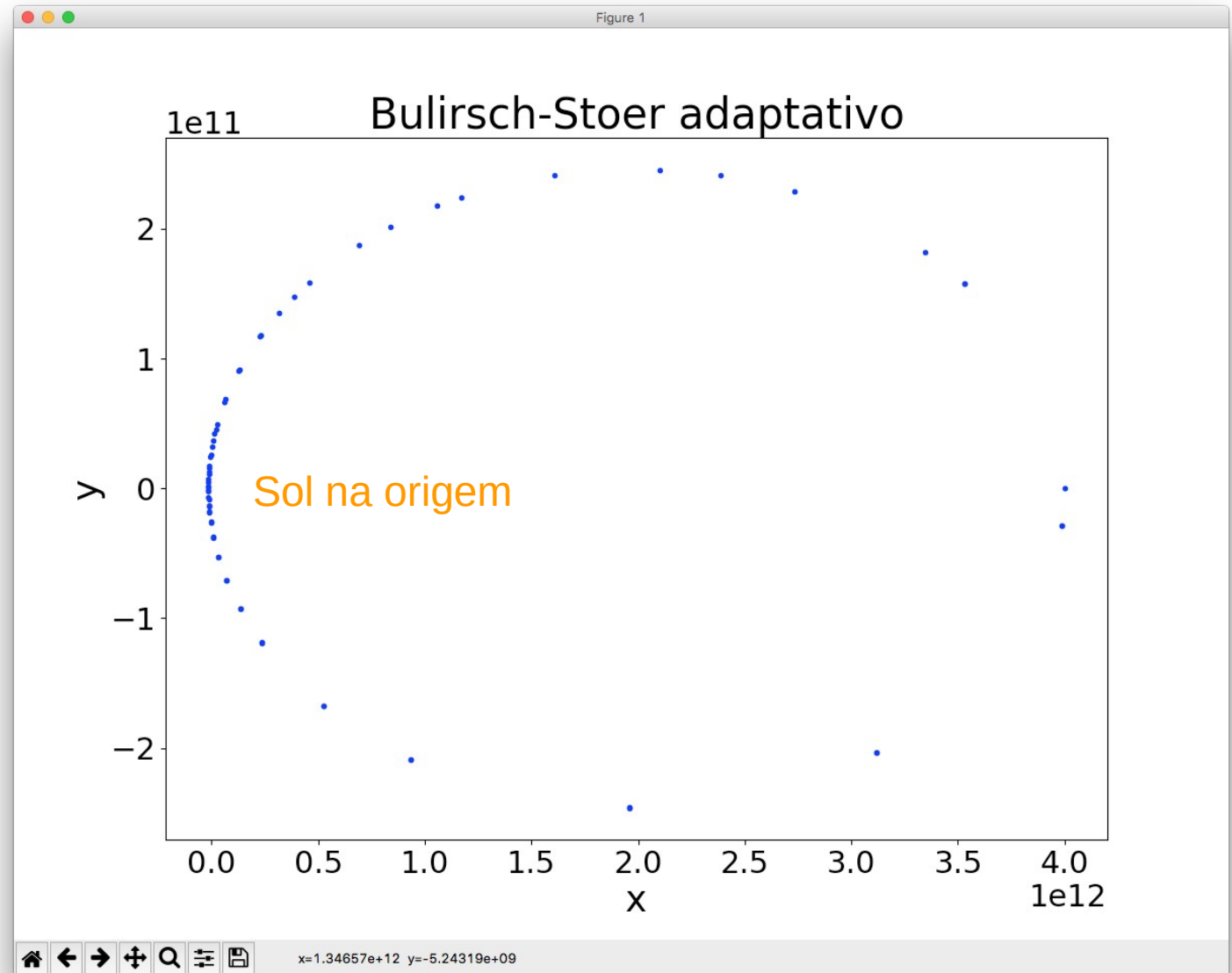
$$r = \sqrt{x^2 + y^2}$$

Exemplo 3: órbita de um cometa

$$\begin{aligned}\frac{dx}{dt} &= v_x \\ \frac{dv_x}{dt} &= -GM \frac{x}{r^3} \\ \frac{dy}{dt} &= v_y \\ \frac{dv_y}{dt} &= -GM \frac{y}{r^3} \\ r &= \sqrt{x^2 + y^2}\end{aligned}$$

$$\delta = 1 \text{ km/ano}$$

$$H(0) = 49 \text{ anos}$$



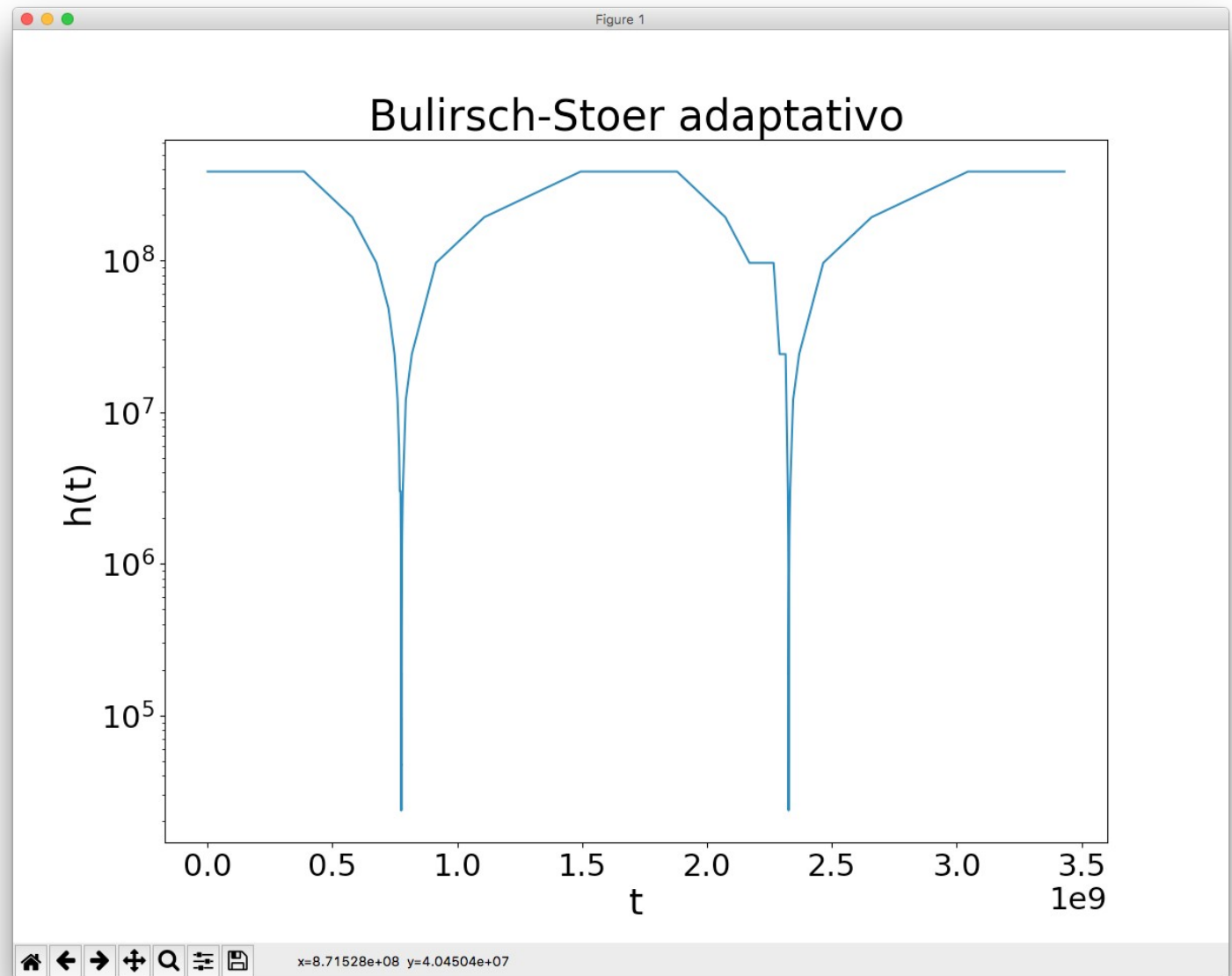
Como deve ser definido o erro a comparar com a precisão?

A simulação percorre **dois** períodos, com tempo de execução de **1** segundo.

Exemplo 3: órbita de um cometa

$$\begin{aligned}\frac{dx}{dt} &= v_x \\ \frac{dv_x}{dt} &= -GM \frac{x}{r^3} \\ \frac{dy}{dt} &= v_y \\ \frac{dv_y}{dt} &= -GM \frac{y}{r^3} \\ r &= \sqrt{x^2 + y^2}\end{aligned}$$

$$\begin{aligned}\delta &= 1 \text{ km/ano} \\ H(0) &= 49 \text{ anos}\end{aligned}$$

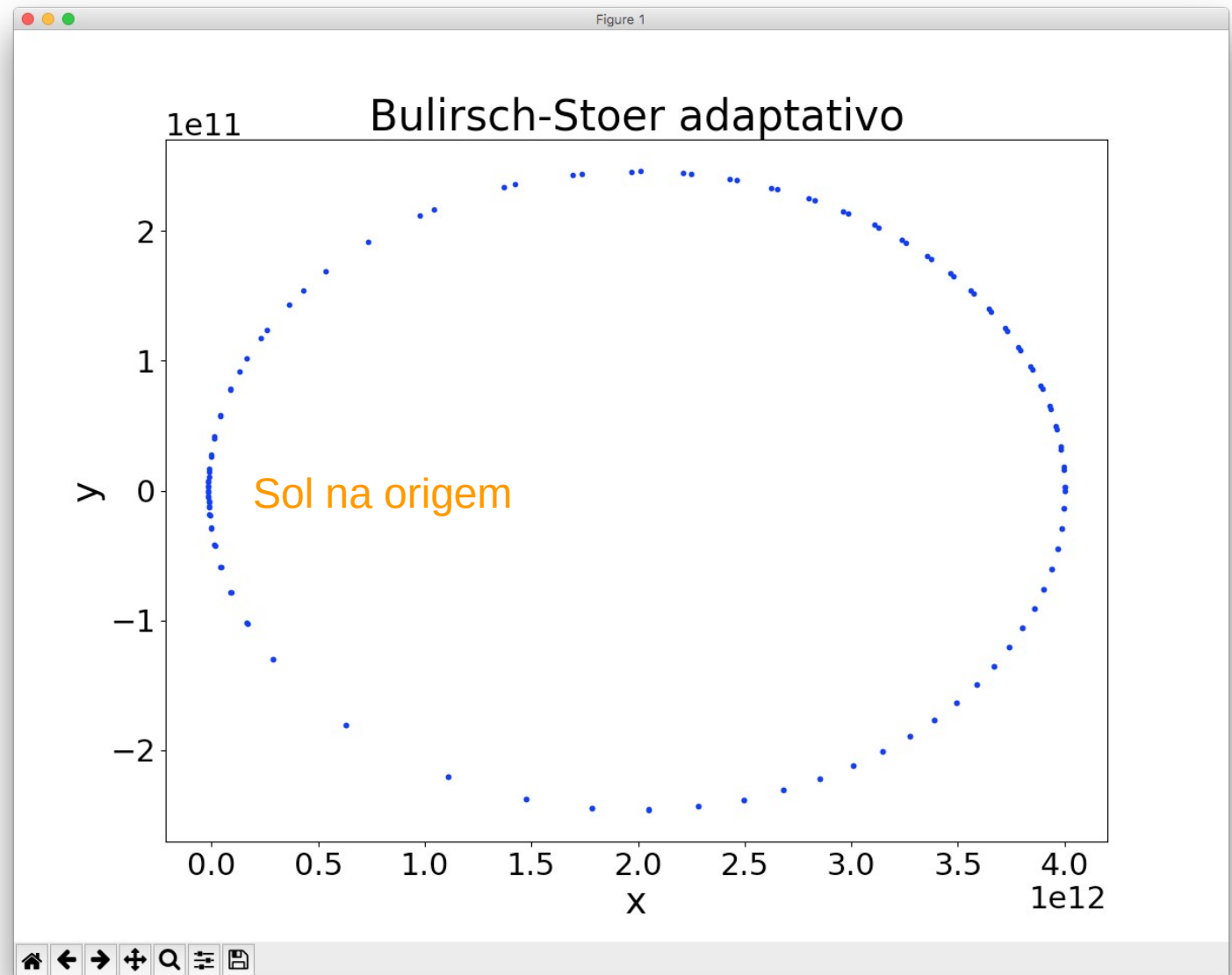


A simulação percorre **dois** períodos, com tempo de execução de **1** segundo.

Exemplo 3: órbita de um cometa

$$\begin{aligned}\frac{dx}{dt} &= v_x \\ \frac{dv_x}{dt} &= -GM \frac{x}{r^3} \\ \frac{dy}{dt} &= v_y \\ \frac{dv_y}{dt} &= -GM \frac{y}{r^3} \\ r &= \sqrt{x^2 + y^2}\end{aligned}$$

$$\begin{aligned}\delta &= 1 \text{ km/ano} \\ H(0) &= 1 \text{ ano}\end{aligned}$$

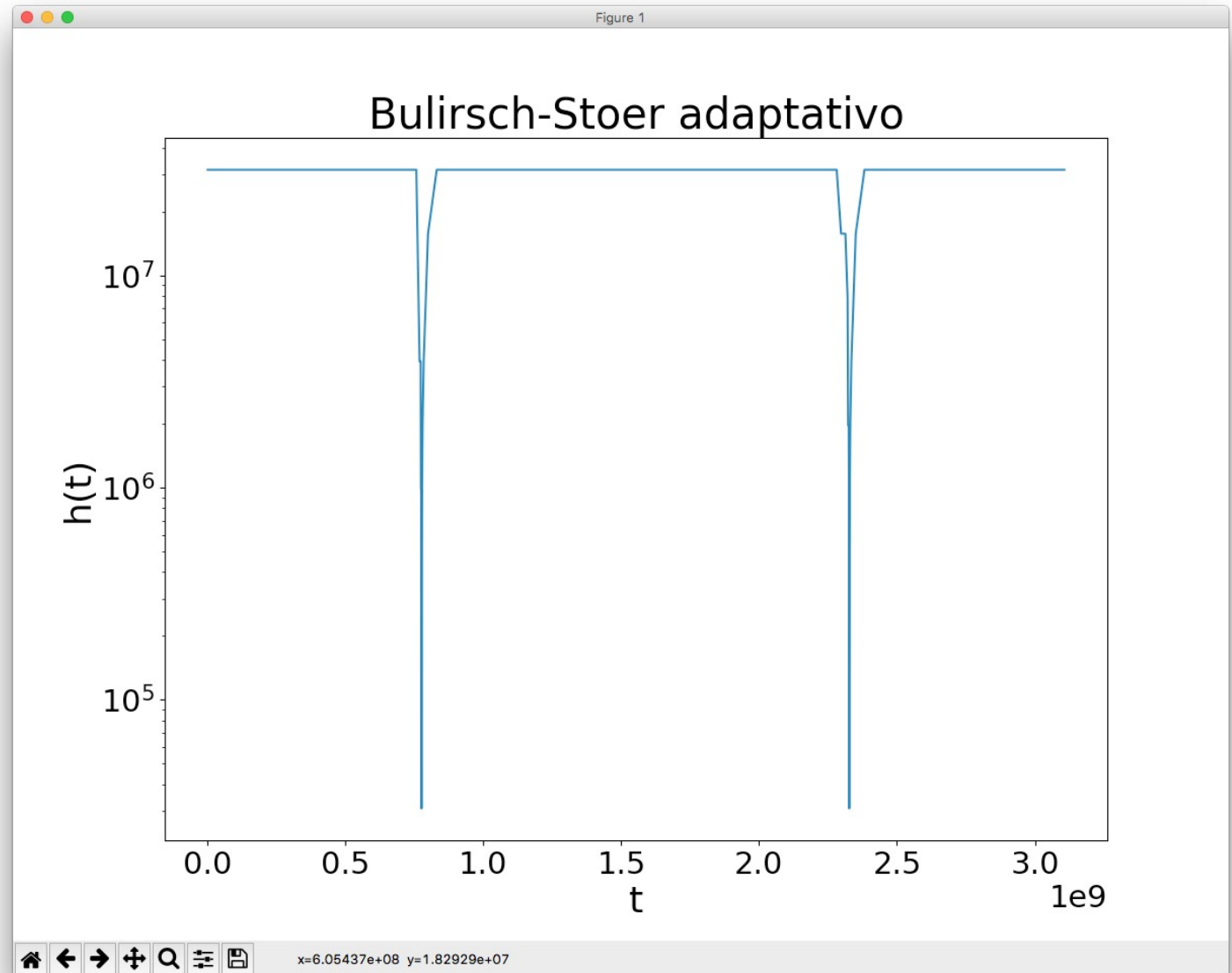


A simulação percorre **dois** períodos, com tempo de execução de **0,5** segundo.
Por que um passo menor diminui o tempo de execução?

Exemplo 3: órbita de um cometa

$$\begin{aligned}\frac{dx}{dt} &= v_x \\ \frac{dv_x}{dt} &= -GM \frac{x}{r^3} \\ \frac{dy}{dt} &= v_y \\ \frac{dv_y}{dt} &= -GM \frac{y}{r^3} \\ r &= \sqrt{x^2 + y^2}\end{aligned}$$

$$\begin{aligned}\delta &= 1 \text{ km/ano} \\ H(0) &= 1 \text{ ano}\end{aligned}$$



A simulação percorre **dois** períodos, com tempo de execução de **0,5** segundo.
Por que um passo menor diminui o tempo de execução?

Exercícios no moodle

- São 2 exercícios, explorando o conteúdo da aula de hoje, que podem ser feitos com base nos programas dos exemplos, que estão disponíveis no moodle. Os exercícios incluem figuras que mostram os resultados esperados, para que você possa verificar se seus códigos estão corretos.

A data para entrega é o dia **29 de abril**.