7600017 - Introdução à Física Computacional
Projeto 3: Movimento realístico

Prof. Dr. Eric de Castro e Andrade

Luis Rodrigo Torres Neves
#10260563

São Carlos, 2018

1 Introdução

Neste trabalho, abordaremos alguns problemas envolvendo movimentos de corpos em situações físicas comuns e relativamente simples, mas que, em muitos casos, exigem uma abordagem computacional para a resolução de suas equações. Estas situações serão: o movimento de um ciclista; movimentos de projéteis com resistência do ar; e as oscilações de um pêndulo simples abandonado a partir de um ângulo qualquer. Para a resolução das EDOs, aplicaremos os métodos de Euler e Euler-Cromer, conforme o caso. As rotinas numéricas serão implementadas em `FORTRAN 90`, e os gráficos dos resultados serão feitos em `Python` (biblioteca Matplotlib). Ao longo das explicações sobre os métodos, serão exibidos os trechos de código considerados relevantes; certos comandos, como declarações de variáveis, serão omitidos.

2 Métodos

2.1 Método de Euler

O método de Euler é uma das abordagens mais elementares para a resolução numérica de equações diferenciais de primeira ordem. A ideia fundamnetal por trás do mesmo consiste em particionar o domínio (discretizá-lo), de modo que, em cada um dos subintervalos obtidos, possa-se aproximar o incremento da variável dependente pelo produto da sua derivada pela variação da variável livre.

Consideremos a EDO de primeira ordem em x:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(x,t) \tag{1}$$

Seja $x(t)$ uma solução para essa EDO no intervalo (a,b), e $a = t_0 < t_1 < ... < t_i < ... < t_n = b$ uma partição do domínio de x. Por hipótese, teremos, no ponto t_i,

$$x'(t_i) = f(x_i, t_i) \tag{2}$$

onde $x_i \equiv x(t_i)$. Conforme descrito anteriormente, o método de Euler consiste em tomar a aproximação

$$x(t_{i+1}) \approx x(t_i) + x'(t_i) \cdot \Delta t_i$$

ou

$$x_{i+1} \approx x_i + x'(t_i) \cdot \Delta t_i \tag{3}$$

onde $\Delta t_i \equiv t_{i+1} - t_i$. Usando (2) em (3), teremos

$$x_{i+1} = x_i + f(x_i, t_i) \cdot \Delta t_i \tag{4}$$

onde passamos a usar o sinal de igualdade, no sentido de que os x_i representam a solução aproximada obtida pelas iterações.

A eq. (4) define como processo iterativo o método de Euler. Conhecida a EDO (1), parte-se de uma condição inicial (t_0, x_0) e itera-se sobre todo o domínio de interesse, o que permite obter uma aproximação para o valor da função x próximo a qualquer ponto do domínio - desde que os passos Δt_i sejam suficientemente pequenos. Tipicamente, adota-se um valor único $\Delta t_i = h,\ i = 0, ..., n-1$, e a eq. (4) torna-se

$$x_{i+1} = x_i + h \cdot f(x_i, t_i) \tag{5}$$

Ao menos em teoria (limitações computacionais à parte), a aproximação é tanto melhor quanto menor o passo h.

O método de Euler também pode ser empregado, por exemplo, na resolução de uma EDO de segunda ordem da forma

$$\frac{\mathrm{d}^2 x}{\mathrm{d}t^2} = f(x, t) \tag{6}$$

Para isso, basta definirmos uma função para a primeira derivada de x, $v = \mathrm{d}x/\mathrm{d}t$, de modo que a eq. (6) se torna o sistema

$$\begin{cases} \mathrm{d}x/\mathrm{d}t = v \\ \mathrm{d}v/\mathrm{d}t = f(x, t) \end{cases} \tag{7}$$

o qual pode ser resolvido iterativamente pelo método de Euler:

$$\begin{cases} x_{i+1} = x_i + h \cdot v_i \\ v_{i+1} = v_i + h \cdot f(x_i, t_i) \end{cases} \tag{8}$$

Método de Euler-Cromer

Uma variante do método de Euler, para o caso de uma EDO da forma (6), consiste em atualizar a variável x numa dada iteração com o valor já atualizado da variável v, ou seja:

$$\begin{cases} v_{i+1} = v_i + h \cdot f(x_i, t_i) \\ x_{i+1} = x_i + h \cdot v_{i+1} \end{cases} \tag{9}$$

Assim se define o chamado método de Euler-Cromer, o qual, conforme será visto mais adiante, funciona melhor que o método de Euler para certos sistemas (notoriamente, movimentos periódicos).

A seguir, discutiremos a implementação destes métodos na solução computacional de alguns problemas de Física.

2.2 Movimento unidimensional com resistência do ar

Discussão teórica

Consideremos um corpo de massa m que se move em uma dimensão sob a ação de uma força que lhe transfere uma potência constante P. É imediato que

$$P = \frac{\mathrm{d}K}{\mathrm{d}t} \tag{10}$$

onde K é a energia cinética do corpo no instante t. Sendo v a velocidade nesse mesmo instante, teremos

$$P = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{1}{2}mv^2\right) = \frac{m}{2}\frac{\mathrm{d}}{\mathrm{d}t}(v^2) \tag{11}$$

Neste caso, como P é uma constante, temos uma EDO trivial em v^2. Esta grandeza, cuja derivada é constante, deve variar linearmente no tempo; teremos então

$$v^2 = v_0^2 + \frac{2P}{m}t \tag{12}$$

onde usamos a condição inicial $v(0) = v_0$. Assim, o problema tem solução analítica dada por:

$$v(t) = \sqrt{v_0^2 + \frac{2P}{m}t} \tag{13}$$

Por outro lado, pode-se enxergar o problema como uma EDO em v. Basta aplicar a regra da cadeia em (11):

$$P = \frac{m}{2} \cdot 2v \cdot \frac{\mathrm{d}v}{\mathrm{d}t}$$

$$\therefore \frac{\mathrm{d}v}{\mathrm{d}t} = \frac{P}{mv} \tag{14}$$

Para resolver numericamente a EDO acima, pode-se aplicar o método de Euler. Isso será feito mais adiante.

O problema se torna mais interessante quando, além da força motriz que transfere potência constante P, atua sobre o corpo uma força resistiva (do ar, por exemplo). Denotaremos esta força por F_r. Uma forma de modelar essa força é atribuir uma dependência quadrática com a velocidade; sabe-se experimentalmente que tal modelo se aplica bem para certas faixas de rapidez. Assim, teremos

$$F_r = -sgn(v) \cdot kv^2$$

onde $sgn(v) = v/|v|$ é a função sinal. Se a velocidade inicial do corpo for positiva, ela permanecerá sempre positiva - intuitivamente, é fácil se convencer disso: a força motriz empurra o corpo no sentido positivo; para que v se tornasse negativa, seria necessário que inicialmente se anulasse e depois F_r acelerasse o corpo para trás, o que não faz sentido nenhum. Assim, podemos escrever simplesmente

$$F_r = -kv^2 \tag{15}$$

Para estimar a constante k, pode-se atribuir a resistência do ar ao fato de que o objeto "empurra" o ar consigo enquanto se desloca. É possível mostrar que a força resistiva que surge nessa análise depende quadraticamente de v. Mais ainda, mostra-se que a constante de proporcionalidade é

$$k = \frac{1}{2}\rho A \tag{16}$$

onde ρ e A são, respectivamente, a densidade do ar e a área efetiva da seção transversal do corpo (área de contato frontal com o ar). Assim, juntando (15) e (16), tem-se

$$F_r = -\frac{1}{2}\rho A v^2 \tag{17}$$

A contribuição desta força no balanço de energia cinética do objeto é $\mathrm{d}W_r/\mathrm{d}t = (F_r \mathrm{d}x)/\mathrm{d}t = F_r v$. Assim,

$$\frac{\mathrm{d}W_r}{\mathrm{d}t} = -\frac{1}{2}\rho A v^3 \tag{18}$$

Portanto, para estudar a variação da energia cinética K neste caso, acrescentamos à eq. (10) a parcela dada pelo segundo membro de (18):

$$\frac{\mathrm{d}K}{\mathrm{d}t} = P - \frac{1}{2}\rho A v^3$$

o que, por meio de uma manipulação algébrica semelhante ao que se fez anteriormente, equivale a

$$mv\frac{\mathrm{d}v}{\mathrm{d}t} = P - \frac{1}{2}\rho A v^3$$

Assim, chegamos à EDO de primeira ordem em v:

$$\frac{\mathrm{d}v}{\mathrm{d}t} = \frac{P}{mv} - \frac{\rho A}{2m}v^2 \tag{19}$$

Cabe notar que, como seria de se esperar, a equação (14) pode ser vista como o caso especial de (19) em que $\rho = 0$.

Ocorre ainda que, neste caso, há um limite para a velocidade: existe um valor v^* de v em que a aceleração se anularia, e portanto v nunca ultrapassa v^*. Esse valor se calcula fazendo $\mathrm{d}v/\mathrm{d}t = 0$ em (19):

$$\frac{P}{mv^*} - \frac{\rho A}{2m}(v^*)^2 = 0 \Rightarrow v^* = \sqrt[3]{\frac{2P}{\rho A}} \tag{20}$$

É interessante observar que v^* independe da massa m do corpo.

(a) Implementação: caso sem atrito

Partiremos agora para a implementação do método de Euler, em `FORTRAN 90`, inicialmente para o caso sem resistência. Os parâmetros (constantes) do programa são P, m, A e v_0, as constantes físicas envolvidas, além do passo de discretização e o tempo de duração da simulação. Esses valores foram lidos de um arquivo texto:

```
open(1, file = 'problema1a.in')
        read(1,*)P, m, v0, dt, tmax
close(1)

jmax = int(tmax/dt, 8)
```

A variável `jmax` corresponde ao número total de iterações no método de Euler. Define-se a função que dá a aceleração do corpo para um certo valor de velocidade:

```
real*8 function acel(P, m, v, A)
        implicit none
        real*8 P, m, v, A

        acel = P/(m*v)
        return
end
```

Note que essa função faz o papel da f em (5). Definida a função, basta inicializar as variáveis e aplicar o método. Escreveu-se, também, um comando para calcular o deslocamento x a cada iteração, considerando que $v = \mathrm{d}x/\mathrm{d}t$.[1] Os resultados foram gravados a cada iteração em arquivos textos separados, para facilitar sua posterior leitura.

```
open(2, file = 'problema1a_tempo.dat')
open(3, file = 'problema1a_vel.dat')
open(4, file = 'problema1a_pos.dat')

v = v0
t = 0.0d0
x = 0.0d0

write(2,*) t
write(3,*) v
write(4,*) x

do j = 1, jmax
        v = v + acel(P, m, v,A)*dt
        t = t + dt
        x = x + v*dt
        write(2,*) t
        write(3,*) v
        write(4,*) x
end do

close(2)
close(3)
close(4)
```

Para calcular o deslocamento `x` de maneira mais precisa, fez-se também a integração numérica em cima dos resultados de `v` via método do trapézio. Para isso, criou-se um programa separado, onde se definiu a função abaixo, que aplica o método do trapézio entre a 1^a e a n-ésima entradas de um vetor de tamanho 3000 (pois utilizaremos `tmax = 300` e `dt = 0.1`):

```
real*8 function int_f_trap(n, vetor, passo)
        implicit none
        real*8 soma, vetor(3000), passo
        integer i, n
```

[1] Essencialmente, o que se fez foi uma aplicação do método de Euler-Cromer, pois a variável `x` é incrementada com o valor já atualizado de `v` a cada iteração. Para este sistema, não há, em princípio, nenhum motivo particular para se optar entre essa implementação ou simplesmente o método de Euler (atualizar `x` antes de `v`). Na seção Resultados, serão analisadas as consequências da escolha que foi feita.

```
        soma = 0.5d0*( vetor(1) + vetor(n) )

        do i = 2, n - 1
                soma = soma + vetor(i)
        end do

        int_f_trap = passo*soma

        return
end
```

No programa principal, leu-se o arquivo com os resultados das velocidades, e aplicou-se a função acima a todos os elementos para obter o deslocamento `x` a cada instante. Esses resultados também foram gravados num arquivo texto:

```
open(1, file = 'problema1a_vel.dat')
        do i = 1, 3000
                read(1,*) vel(i)
        end do
close(1)

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

open(2, file = 'problema1a_pos_trapezio.dat')

        write(2,*)0.0d0
        do i = 2, 3000
                write(2,*) int_f_trap(i, vel, 1.0d-1)
        end do

close(2)
```

Finalmente, fez-se um programa em `Python` para ler e graficar os resultados.

(b) Implementação: caso com atrito

A principal diferença entre este programa e o programa principal descrito no caso **(a)** é a definição da função que dá a aceleração, agora de acordo com a eq. (19):

```
real*8 function acel(P, m, v, rho, A)
        implicit none
        real*8 P, m, v, rho, A

        acel = P/(m*v) - (rho*A*v**(2.0d0))/(2.0d0*m)

        return
end
```

Agora também será necessário ler a densidade `rho` do ar e a área `A` no início do programa:

```
open(1, file = 'problema1b.in')
        read(1,*)P, m, rho, A, v0, dt, tmax
close(1)
```

De resto, a aplicação iterativa do método de Euler é idêntica ao que já se fez anteriormente. Também foi calculado o deslocamento `x` a cada instante, tanto pelo método de Euler quanto por integração via método do trapézio. Os resultados foram gravados em arquivos textos que foram posteriormente importados e graficados em Python. Ainda, calculou-se a velocidade terminal v^* pela fórmula (20):

```
vterm = ((2.0d0*P)/(rho*A))**(1.0d0/3.0d0)

write(*,*) 'v_terminal = ', vterm
```

Para decidir quando a velocidade $v(t)$ "atingiu" a velocidade terminal, é necessário, na verdade, estabelecer um critério de "igualdade", o qual em princípio é arbitrário (pois tecnicamente a aproximação entre $v(t)$ e v^* é assintótica). No caso, optou-se por escolher como critério a velocidade atingir 99% do seu valor terminal. Escreveu-se um trecho de código para que o valor do tempo `t` fosse impresso na tela assim que isso ocorresse. Para isso, declarou-se uma variável booleana, `atingiu`, à qual foi atribuído inicialmente o valor `.false.`. No laço do método de Euler, inseriu-se o trecho de código abaixo:

```
if  ( (v > 0.99d0*vterm) .and.  .not. (atingiu) ) then
        write(*,*) t
        atingiu = .true.
end if
```

(c) Implementação: caso com atrito, variando A

Uma análise interessante a se fazer é repetir a simulação do item anterior, variando algum dos parâmetros da equação do movimento. Seguindo a sugestão do texto norteador, faremos variar o parâmetro A, que representa a área da seção de contato entre o móvel e o meio resistivo (no caso, o ar). Pela eq. (19), é esperado que um aumento de A provoque uma diminuição no valor assumido por v a cada instante t.

Para isso, pode-se escrever um programa muito semelhante ao do item anterior, apenas com o adendo de que o parâmetro `A` deverá assumir diferentes valores. Optou-se por fazer a área igual a $0.25\,\mathrm{m}^2, 0.5\,\mathrm{m}^2, 0.75\,\mathrm{m}^2$ e $1.0\,\mathrm{m}^2$. Esses valores foram armazenados num vetor. Para facilitar a criação dos arquivos de saída, criou-se uma variável do tipo `character`:

```
        real*8 areas(4)
        character (len=2) nome
!
        areas = (/ .25d0, .5d0, .75d0, 1.0d0 /)
```

Agora, todo o laço em que consiste a simulação de cada movimento (muito semelhante ao que se mostrou no caso **(b)**) fica contido num laço mais externo, que corresponde a iteração sobre o vetor das áreas:

```
do i = 1, 4

        write(nome, '(i2)') i

        A = areas(i)

        vterm = ((2.0d0*P)/(rho*A))**(1.0d0/3.0d0)

        open(2, file = 'problema1c_' // nome // '_tempo.dat')
```

```
        open(3, file = 'problema1c_' // nome // '_vel.dat')


        v = v0
        t = 0.0d0

        write(2,*) t
        write(3,*) v

        atingiu = .false.

        do j = 1, jmax

                        [...]

        end do
end do
```

Neste caso, fez-se a simulação apenas para a velocidade. Os resultados foram graficados em `Python`.

2.3 Lançamento de projéteis

Discussão teórica

Consideremos um projétil que, no instante $t = 0$, é atirado com velocidade inicial $\vec{v}_0$, numa região próxima à superfície da Terra, onde o campo gravitacional pode ser considerado uniforme e igual a $\vec{g}$. Seja $\vec{r}(t) = x(t)\hat{\imath} + y(t)\hat{\jmath}$ a posição[2] do projétil num instante $t > 0$. Se levarmos em conta exclusivamente a ação do campo gravitacional, segue imediatamente da Segunda Lei de Newton que as equações de movimento do projétil são

$$\begin{cases} \mathrm{d}^2x/\mathrm{d}t^2 = 0 \\ \mathrm{d}^2y/\mathrm{d}t^2 = -g \end{cases} \tag{21}$$

Trata-se de duas equações desacopladas ("independentes"), cujas soluções gerais são triviais e bem-conhecidas:

$$\begin{cases} x(t) = v_0 \cos\theta\, t \\ y(t) = v_0 \sin\theta\, t - \frac{1}{2}gt^2 \end{cases} \tag{22}$$

onde θ é o ângulo entre $\vec{v}_0$ e $\hat{\imath}$.

A partir das equações acima, podemos facilmente encontrar uma equação para a trajetória, $y(x)$. Em particular, é possível achar uma relação entre o alcance horizontal A do movimento (valor positivo de x que dá $y = 0$) e o ângulo de lançamento θ:

$$A = \frac{v_0^2}{g} \sin(2\theta) \tag{23}$$

Vemos que o problema do lançamento de um projétil sujeito exclusivamente à ação de um campo gravitacional uniforme é de fácil solução analítica. Isso, no entanto, não nos impede de discretizar o tempo e aplicar o método de Euler para obter uma simulação computacional

[2] Adotamos um sistema cartesiano Oxy que contém $\vec{v}_0$, com Oy orientado para cima e O no ponto de partida.

deste movimento. Fazendo isso, como as eqs. (21) são de segunda ordem, teremos, como visto anteriormente, duas EDOs (de primeira ordem) para x e outras duas para y. Mais adiante, serão expostos os aspectos computacionais referentes a este caso.

Uma abordagem bem mais realista leva em conta uma força de resistência do ar, $\vec{F}_r$, tal qual a considerada em 2.2 (i. e., variando com o quadrado da velocidade):

$$\vec{F}_r = -\gamma|\vec{v}|^2\hat{v} = -\gamma|\vec{v}|\vec{v} \tag{24}$$

onde $\vec{v} = \mathrm{d}\vec{r}/\mathrm{d}t = v_x\hat{\imath} + v_y\hat{\jmath}$. Como vimos anteriormente, o coeficiente γ está diretamente relacionado à densidade do ar; portanto, neste tratamento, assume-se implicitamente que tal densidade é constante (atmosfera homogênea).

Neste caso, segue da Segunda Lei de Newton que as equações de movimento assumem a forma

$$\begin{cases} \mathrm{d}^2x/\mathrm{d}t^2 = -\beta\sqrt{v_x^2 + v_y^2}\, v_x \\ \mathrm{d}^2y/\mathrm{d}t^2 = -g - \beta\sqrt{v_x^2 + v_y^2}\, v_y \end{cases} \tag{25}$$

onde se introduziu $\beta \equiv \gamma/m$, sendo m a massa do projétil.

Agora, temos um sistema de duas EDOs de segunda ordem acopladas, onde figuram, inclusive, as derivadas de primeira ordem (v_x e v_y). Uma solução analítica para tal sistema parece ser uma pretensão inalcançável.

Reescrevendo o sistema com apenas equações de ordem 1, teremos

$$\begin{cases} \mathrm{d}x/\mathrm{d}t = v_x \\ \mathrm{d}v_x/\mathrm{d}t = -\beta\sqrt{v_x^2 + v_y^2}\, v_x \\ \mathrm{d}y/\mathrm{d}t = v_y \\ \mathrm{d}v_y/\mathrm{d}t = -g - \beta\sqrt{v_x^2 + v_y^2}\, v_y \end{cases} \tag{26}$$

Como a única saída parece ser uma solução numérica, implementaremos o método de Euler para resolver iterativamente o sistema (26).

O modelo se torna ainda mais realista se considerarmos a variação da densidade do ar com a altitude. Uma das abordagens para modelar tal variação considera a existência de correntes de convecção no ar, onde massas de ar atingem altitudes maiores e se expandem, adiabaticamente, tornando-se então menos densas. Uma análise feita em cima dessas hipóteses mostra que, nesse caso, a dependência da densidade ρ do ar com a a altitude y é dada por

$$\rho(y) = \rho_0\left(1 - b\frac{y}{T_0}\right)^\alpha$$

onde α é um parâmetro característico do gás (no caso, o ar atmosférico) e b depende do gás e do campo gravitacional. Portanto, como γ é proporcional a ρ, teremos

$$\gamma(y) = \gamma_0\left(1 - b\frac{y}{T_0}\right)^\alpha \tag{27}$$

e incluiremos esta correção nas eqs. (26).

(a) Implementação: caso sem atrito

Inicialmente, fez-se um programa para calcular a trajetória do projétil instante a instante, via método de Euler aplicado às eqs. (21), guardando os dados de tempo, abscissa e ordenada em arquivos separados. Os parâmetros foram atribuídos de acordo com a sugestão do texto norteador: $v_0 = 700\,\mathrm{m/s}$, $g = 9.8\,\mathrm{m/s^2}$, $\Delta t = 0.01\,\mathrm{s}$. A simulação foi programada para cinco valores diferentes do ângulo inicial θ, a serem lidos de um arquivo.

```
v_0 = 700.0d0
g = 9.8d0
dt = 0.01d0
pi = 4.0d0*atan(1.0d0)

open(1, file = 'problema2_theta.in')
        read(1, *) thetas
close(1)

do i = 1, n
        thetas(i) = thetas(i)*(pi/180.0d0)        !converter para radianos
end do
```

Assim, o laço da simulação ficou dentro do laço que itera sobre os valores de θ. A simulação foi condicionada a $y \geqslant 0$, pois, naturalmente, estamos interessados apenas no que ocorre até o projétil atingir o solo.

```
do j = 1, n
        theta = thetas(j)

        write(nome, '(i2)') j

        open(2, file = 'problema2a_' // nome // '_tempo.dat')
        open(3, file = 'problema2a_' // nome // '_x.dat')
        open(4, file = 'problema2a_' // nome // '_y.dat')


                t = 0.0d0
                x = 0.0d0
                y = 0.0d0

                v_x = v_0*dcos(theta)
                v_y = v_0*dsin(theta)

                write(2,*) 0.0d0
                write(3,*) 0.0d0
                write(4,*) 0.0d0

                do while (y >= 0.0d0)

                        x = x + v_x*dt
                        y = y + v_y*dt

                        v_y = v_y - g*dt

                        t = t + dt

                        write(2,*) t
                        write(3,*) x
                        write(4,*) y

                end do
```

```
            close(2)
            close(3)
            close(4)

        end do
```

Para cada valor de θ, fez-se um gráfico da trajetória, $y \times x$.

Para analisar os efeitos do ângulo θ sobre o alcance do lançamento, fez-se um outro programa, que realiza a simulação para uma quantidade ajustável de valores de θ igualmente espaçados entre 0 e 90º. Neste caso, não foram gravados os valores de x e y a cada iteração; guardaram-se apenas, para cada lançamento, o ângulo inicial θ e o alcance A (definido como o valor de x ao final da simulação).

```
        do i = 1, n
            thetas(i) = 90.0d0*real(i,8)/real(n,8)
        end do

!        ########################

        open(2, file = 'problema2a_angulo.dat')
        open(3, file = 'problema2a_alcance.dat')


        do j = 1, n
            theta = thetas(j)

                t = 0.0d0
                x = 0.0d0
                y = 0.0d0

                v_x = v_0*dcos(theta*pi/180.0d0)
                v_y = v_0*dsin(theta*pi/180.0d0)

                do while (y >= 0.0d0)

                    x = x + v_x*dt
                    y = y + v_y*dt

                    v_y = v_y - g*dt

                    t = t + dt

                end do

                write(2,*)theta
                write(3,*)x

        end do

        close(2)
        close(3)
```

Fez-se `n = 50` e, com os dados obtidos, graficou-se $A \times \theta$.

(b) Implementação: caso com atrito

Exatamente como no caso **(a)**, fez-se um programa para simular e gravar as trajetórias para cinco valores diferentes de θ, e outro para fazê-lo para 50 valores diferentes desse ângulo, gravando apenas os ângulos e os alcances correspondentes. A única diferença estava na atualização das variáveis, que seguiu as EDOs (25). Após o fim do programa principal, definiram-se as funções abaixo:

```
        real*8 function a_x(k, vx, vy)      ! k = gamma/m
                implicit none
                real*8 k, vx, vy

                a_x = -k*sqrt(vx**2 + vy**2)*vx

                return
        end
!
        real*8 function a_y(k, vx, vy)      ! k = gamma/m
                implicit none
                real*8 k, vx, vy

                a_y = -k*sqrt(vx**2 + vy**2)*vy

                return
        end
```

A simulação, então, foi escrita como segue:

```
                do while (y >= 0.0d0)

                        x = x + v_x*dt
                        y = y + v_y*dt

                        temp(1) = v_x
                        temp(2) = v_y

                        v_x = v_x + a_x(4.0d-5, temp(1), temp(2) )*dt
                        v_y = v_y + (a_y(4.0d-5, temp(1), temp(2) ) - g)*dt

                        t = t + dt

                        write(2,*) t
                        write(3,*) x
                        write(4,*) y

                end do
```

Note que é necessário guardar `v_x` e `v_y` em variáveis "temporárias" a cada iteração, pois a atualização da primeira influencia na segunda. Adotou-se `k` $= \gamma/m = 4 \cdot 10^{-5}\,\mathrm{m}^{-1}$, seguindo a sugestão do enunciado.

(c) Implementação: caso com atrito e atmosfera heterogênea

Repetiu-se todo o processo de simulação e coleta de dados dos itens **(a)** e **(b)**, mas agora com as funções de aceleração definidas de modo a incluir os efeitos da heterogeneidade da atmosfera, eq. (27). Conforme a sugestão, adotou-se $\alpha = 2.5$ e $b = 6.5 \cdot 10^{-3}\,\mathrm{K/m}$:

```
real*8 function a_x(k, vx, vy, y)      ! k = gamma/m
        implicit none
        real*8 k, vx, vy, fator, y
        fator = (1.0d0 - 6.5d-3* (y/300.0d0) )**2.5
        a_x = -k*fator*sqrt(vx**2 + vy**2)*vx

        return
end

real*8 function a_y(k, vx, vy, y)      ! k = gamma/m
        implicit none
        real*8 k, vx, vy, fator, y
        fator = (1.0d0 - 6.5d-3* (y/300.0d0) )**2.5

        a_y = -k*fator*sqrt(vx**2 + vy**2)*vy

        return
end
```

Outras análises foram feitas, a partir do ângulo que dá o alcance máximo em cada caso estudado. Como essas análises dependem dos dados das simulações, elas serão comentadas apenas na seção Resultados.

2.4 Pêndulo simples

Discussão teórica

Consideremos um pêndulo simples constituído de um corpo puntiforme de massa m pendurado a uma corda inextensível, de massa desprezível e comprimento L. Em coordenadas polares, com origem no ponto de fixação da corda, a coordenada r da massa é constante e igual a L; segue que o movimento é essencialmente unidimensional, pois pode ser estudado em termos apenas do ângulo polar θ (veja a Fig. 1).

Na direção $\hat{\theta}$, atua apenas a componente tangencial do peso $m\vec{g}$, de modo que a 2ª Lei de Newton dá

$$-mg\sin\theta = ma_\theta = m\frac{\mathrm{d}^2}{\mathrm{d}t^2}(L\theta)$$

e a equação de movimento do pêndulo é

$$\frac{\mathrm{d}^2\theta}{\mathrm{d}t^2} = -\frac{g}{L}\sin\theta \tag{28}$$

Se adotarmos a energia potencial (gravitacional) como zero em $\theta = 0$, mostra-se facilmente que a energia mecânica do sistema é dada por

$$E = K + U = \frac{1}{2}mL^2\omega^2 + mgL(1-\cos\theta) \tag{29}$$

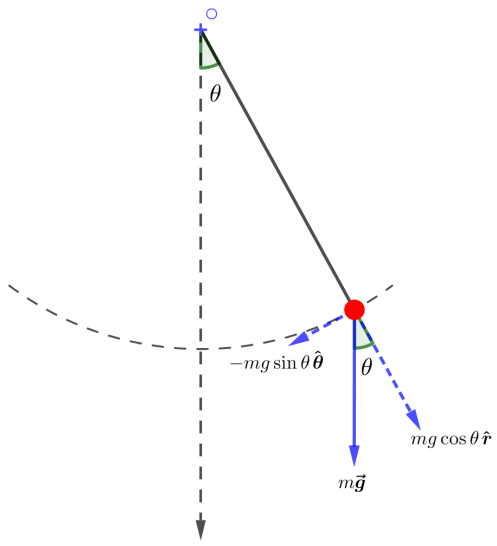

Figura 1: Esquema de um pêndulo simples, mostrando o peso e suas componentes polares.

onde $\omega = \mathrm{d}\theta/\mathrm{d}t$ é a velocidade angular da massa m. Como atuam em m somente duas forças – a saber, a tração no fio (que é sempre ortogonal ao movimento e portanto não realiza trabalho) e o peso (que é conservativo), a grandeza escalar dada em (29) deve se conservar ao longo de todo o movimento.

Para θ pequeno, uma aproximação de primeira ordem dá $\sin\theta \approx \theta$, de modo que a eq. (28) se reduz à bem conhecida equação do oscilador harmônico, de simples solução analítica. No caso geral, porém, uma tal solução se mostra inalcançável, pelo menos em termos de funções elementares. Assim, buscaremos uma solução numérica.

Escrevendo a (28) em termos de equações de primeira ordem, vem

$$\begin{cases} \mathrm{d}\theta/\mathrm{d}t = \omega \\ \mathrm{d}\omega/\mathrm{d}t = -(g/L)\sin\theta \end{cases} \tag{30}$$

Inicialmente, aplicaremos o método de Euler (8) para resolver iterativamente o sistema acima. No entanto, como se sabe (e será verificado na simulação), este método não funciona bem para sistemas periódicos, de modo que a energia mecânica deixa de ser conservada. Assim, recorreremos ao método de Euler-Cromer (9), e observaremos as diferenças entre os resultados.

A partir da invariância da energia mecânica (29), pode-se mostrar que o período de oscilação do pêndulo solto a partir do repouso de um ângulo θ_0 é dado por

$$T(\theta_0) = 4\sqrt{\frac{L}{g}}\int_0^{\pi/2} \frac{\mathrm{d}u}{\sqrt{1-k^2\sin^2 u}} \tag{31}$$

onde $k \equiv \sin(\theta_0/2)$. Trata-se de uma integral elíptica, que não pode ser resolvida em termos de funções elementares. Para θ pequeno, no entanto, uma aproximação de Taylor no integrando mostra que o período é dado aproximadamente por

$$T(\theta_0) \approx 2\pi\sqrt{\frac{L}{g}}\left(1 + \frac{1}{16}\theta_0^2\right) \tag{32}$$

Se desprezarmos ainda o termo quadrático em θ_0 acima, recuperamos a famosa aproximação harmônica para o período:

$$T(\theta_0) \approx 2\pi\sqrt{\frac{L}{g}} \tag{33}$$

(a) Implementação: método de Euler

A implementação é muito semelhante ao que já se fez nos problemas anteriores.

Inicialmente, leem-se os parâmetros do problema: o ângulo inicial, a aceleração da gravidade, o comprimento do pêndulo, o passo temporal, o tempo de simulação e a massa m:

```
open(10, file = 'problema3a.in')
        read(10,*) theta0, g, L, dt, tmax, m
close(10)

jmax = int(tmax/dt, 8)

theta0 = theta0*(pi/180.0d0)
```

Consideraremos sempre um pêndulo partindo do repouso.

Definiram-se as funções abaixo para calcular a aceleração e a energia mecânica em cada passo:

```
real*8 function acel(g, L, theta)
        implicit none
        real*8 g, L, theta

        acel = -(g/L)*dsin(theta)
        return
end


real*8 function E(m, L, omega, g, theta)
        implicit none
        real*8 m, L, omega, g, theta

        E = 0.5d0*m*(L*omega)**2 + m*g*L*(1.0d0 - dcos(theta))

        return
end
```

O método de Euler foi então implementado conforme abaixo:

```
open(20, file = 'problema3a_tempo.dat')
open(30, file = 'problema3a_teta.dat')
open(40, file = 'problema3a_energia.dat')

t = 0.0d0
theta = theta0
omega = 0.0d0

write(20,*) t
```

```
write(30,*) theta
write(40,*)E(m, L, omega, g, theta)

do j = 1, jmax
        temp = theta
        theta = theta + omega*dt
        omega = omega + acel(g, L, temp)*dt
        t = t + dt
        write(20,*) t
        write(30,*) theta
        write(40,*) E(m, L, omega, g, theta)
end do


close(20)
close(30)
close(40)
```

Note que foi necessário armazenar a variável `theta` numa variável temporária, `temp`, para que pudéssemos aplicar efetivamente o método de Euler – ω_{i+1} sendo calculado a partir de θ_i, e *não* de θ_{i+1}. Com os resultados, graficaram-se $\theta \times t$ e $E \times t$.

(b) Implementação: método de Euler-Cromer

A única diferença entre este método e o anterior está na atualização das variáveis, conforme se vê na comparação entre as eqs. (8) e (9). Portanto, tomou-se o mesmo código da seção acima, apenas com o laço principal modificado conforme abaixo:

```
do j = 1, jmax
        omega = omega + acel(g, L, theta)*dt
        theta = theta + omega*dt
        t = t + dt
        write(20,*) t
        write(30,*) theta
        write(40,*) E(m, L, omega, g, theta)
end do
```

(c) Estudo do período de oscilação

Podemos, também, analisar a dependência do período de oscilação do pêndulo com a amplitude angular da oscilação (que coincide com o ângulo inicial, θ_0, pois estamos partindo sempre do repouso). Para tal fim, fez-se um código repetindo a simulação anterior, mas para uma gama ajustável de valores de θ_0, e gravando apenas o período para cada simulação.

Para determinar o valor do período para um certo valor de θ_0, usou-se o fato de que a velocidade angular ω sempre começa nula, vai aumentando em valor absoluto mas no sentido negativo, começa a diminuir em valor absoluto, e anula-se novamente após meio período. Ou seja, cada simulação foi feita até ω se tornar positivo; o valor correspondente do período é o dobro do valor final de `t`.

```
open(10, file = 'problema3c.in')
        read(10,*) g, L, dt, m
close(10)
```

```
do j = 1, n
        vetor_theta0(j) = 90.0d0*real(j,8)/(real(n,8))
end do


!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!


open(20, file = 'problema3c_theta.dat')
open(30, file = 'problema3c_periodo.dat')
open(40, file = 'problema3c_periodo_simpson.dat')


do j = 1, n

        theta0 = (vetor_theta0(j))*pi/180.0d0

        t = 0.0d0
        theta = theta0
        omega = 0.0d0


        do while (omega <= 0.0d0)
                omega = omega + acel(g, L, theta)*dt
                theta = theta + omega*dt
                t = t + dt
        end do

        write(20,*) theta0*180.0d0/pi
        write(30,*)2.0d0*t
        write(40,*)T_simp(theta0, 1.0d-4)

end do

close(20)
close(30)
close(40)
```

A função `T_simp` foi utilizada para calcular, para cada valor de θ_0, o correspondente período $T(\theta_0)$ a partir da integral elíptica (31). Para isso, definiram-se, após o programa principal, as três funções abaixo; a primeira corresponde a uma implementação do método de Simpson, estudado no Projeto 2 deste curso.

```
real*8 function int_f_simp(h, theta)
        implicit none
        real*8 a, b, h, soma, f, c, theta
        integer i, n

        a = 0.0d0
        b = asin(1.0d0)
```

```
        n = int((b-a)/h)

        soma = f(a, theta) + f(b, theta)

        do i = 1, N - 1
                if (mod(i, 2) == 1) then
                        c = 4.0d0
                else
                        c = 2.0d0
                end if
                soma = soma + c*f(a + int(i,8)*h, theta)
        end do

        int_f_simp = (h/3.0d0)*soma

        return
end

!        ##############################################################

real*8 function f(u, theta)
        implicit none
        real*8 theta, u

        f = 1.0d0/dsqrt(1.0d0 - ( (dsin(theta/2.0d0)) **2.0d0)*(dsin(u))**2.0d0 )
        return
end

!        ##############################################################

real*8 function T_simp(theta, h)
        implicit none
        real*8 theta, h, int_f_simp

        T_simp = 4.0d0*dsqrt(1.0d0/1.0d1)*int_f_simp(h, theta)

        return
end
```

Note que a integral de Simpson foi calculada para $h = 10^{-4}$, o que corresponde a particionar o domínio de integração em 10^4 intervalos. No caso analisado no projeto anterior, havia-se concluído que o valor de h que otimizava a precisão era da mesma ordem de grandeza (1/4096, precisamente); veremos que, para o presente problema, o valor adotado funciona bem, embora não disponhamos de um método preciso para avaliar o erro cometido.

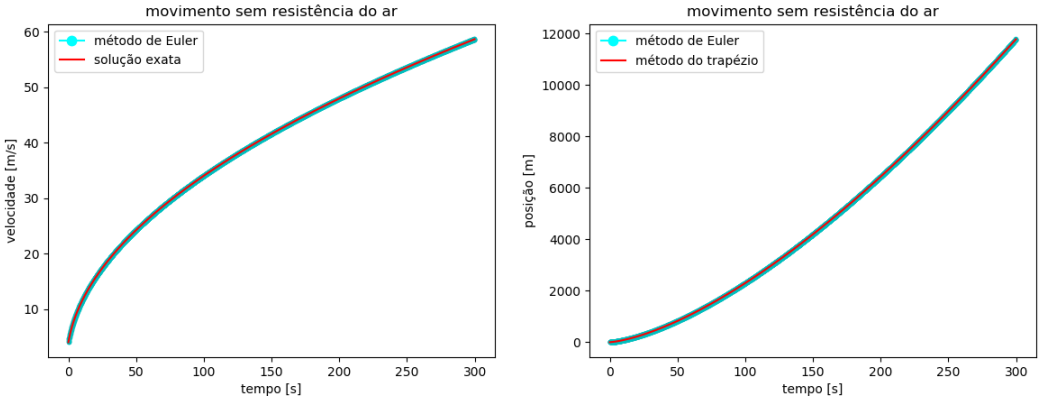

Figura 2: Gráficos de $v(t)$ e $x(t)$ para o movimento sem atrituo.

3 Resultados

3.1 Movimento unidimensional com resistência do ar

(a) Caso sem resistência

Seguindo a sugestão do enunciado, fez-se $P = 400\,\mathrm{W}, m = 70\,\mathrm{kg}, v_0 = 4\,\mathrm{m/s}, \Delta t = 0.1\,\mathrm{s}, t_{max} = 300\,\mathrm{s}$. (O ciclista não pode partir do repouso pois as EDOs (14) e (19) ficam indefinidas para $v = 0$.) Esses parâmetros foram lidos do arquivo `problema1a.in`:

```
400.0d0, 70.0d0, 4.0d0, 0.1d0, 300.0d0
```

Os gráficos mostrando os resultados desta simulação estão mostrados na Fig. 2. No caso da velocidade, fez-se ainda o gráfico da solução exata dada pela eq. (13).

Os gráficos de $v \times t$ sugerem (e a eq. (13) garante) que, no caso sem atrito, v cresce indefinidamente com t, embora a taxa de crescimento vá se reduzindo ($P = Fv = mav \Rightarrow a$ diminui quando v aumenta).

A distância total percorrida após os 5 min foi `11791.173953950152` m, segundo o método de Euler-Cromer, e `11788.439261203323` m, segundo o método do trapézio. De fato, era de se esperar que o primeiro desse um resultado ligeiramente maior que o segundo. Essencialmente, fazer tal integração via método de Euler seria o mesmo que aproximar a área sob a curva $v \times t$ por retângulos (assim como na definição usual da soma de Riemann). Por outro lado, da forma como esse cálculo foi implementado (`x` atualizado *depois* de `v` a cada iteração - método de Euler-Cromer), tomou-se como altura de cada retângulo o valor de v no limite *superior* de cada intervalo da partição. Para uma função estritamente crescente (sabemos que é o caso de $v(t)$), isso só podia dar uma aproximação por excesso (veja a Fig. 3).

Já o método do trapézio, por definição, leva em conta uma média entre os valores de v nos dois limites de cada intervalo, dando um ajuste mais fino à curva (Fig. 4). Para uma curva de concavidade negativa, isso deveria dar uma aproximação sempre por falta. Assim, se fizéssemos um cálculo exato (analítico) de $x(t)$ em $t = 5$ min, esperaríamos obter um número compreendido entre os dois valores calculados numericamente. De fato, podemos fazer esse teste; basta resolver a integral abaixo:

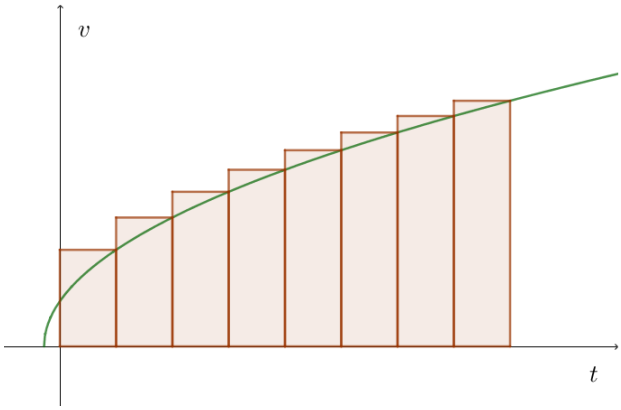

Figura 3: Integral aproximada pelo método de Euler-Cromer, avaliado no extremo superior de cada intervalo.

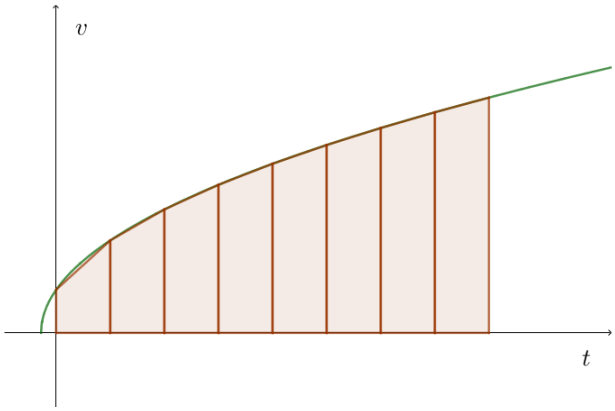

Figura 4: Integral aproximada pelo método do trapézio.

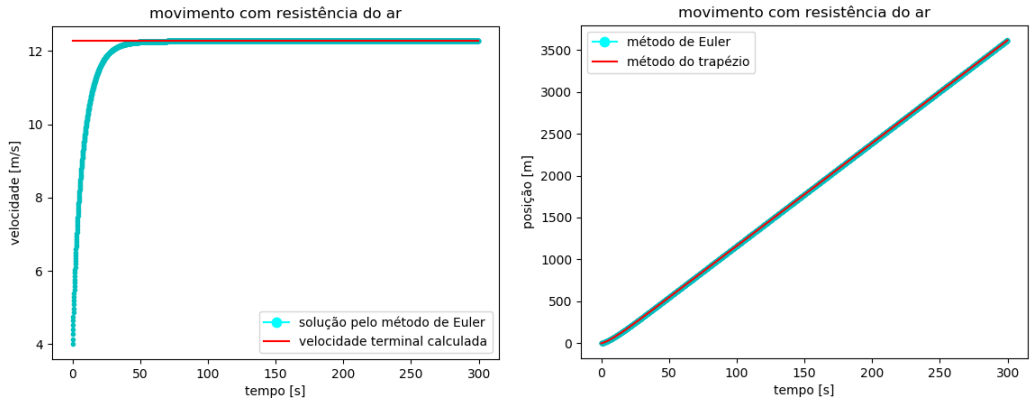

Figura 5: Gráficos de $v(t)$ e $x(t)$ para o movimento com atrituo.

$$x(t) = \int_0^t v(t')\mathrm{d}t' = \int_0^t \left(\sqrt{v_0^2 + \frac{2P}{m}t'}\right)\mathrm{d}t'$$

o que dá

$$x(t) = \frac{m}{3P}\left(\left(v_0^2 + \frac{2P}{m}t\right)^{3/2} - v_0^3\right)$$

que pode ser reescrito como

$$x(t) = \frac{m}{3P}\left(v(t)^3 - v_0^3\right) \tag{34}$$

Avaliando em `FORTRAN` essa expressão com os parâmetros utilizados, em t = 300 s, obteve-se o seguinte resultado: `11789.138712128490` m. Isso confirma a hipótese de que os métodos de Euler-Cromer e do trapézio forneceram aproximações por excesso e por falta, respectivamente; também é fácil perceber que o resultado fornecido pelo método do trapézio é mais preciso.

(b) Caso com resistência

Foi dado como entrada o arquivo `problema1b.in`, reproduzido abaixo:

```
400.0d0, 70.0d0, 1.3d0, .333d0, 4.0d0, 0.1d0, 300.0d0
```

A diferença entre este arquivo e a entrada do programa anterior é que agora figuram também a área $A = 0.333\,\mathrm{m}^2$ e a densidade do ar $\rho = 1.3\,\mathrm{kg/m}^3$. A velocidade terminal calculada pela eq. (20) foi de `12.271588969854935` m/s. Os gráficos de $v(t)$ e $x(t)$ para esta simulação estão mostrados na Fig. 5.

O valor de `v` ao fim dos 5 min de movimento era de `12.271588969854861` m/s. O tempo decorrido até que a velocidade atingisse 99% de v^* foi igual a `35.700000000000237` s. A distância total percorrida foi de `3614.9382657857041`, pelo método de Euler-Cromer, e `3614.5246863370921`, pelo do trapézio. Novamente, o resultado deste método foi menor que o daquele, conforme esperado. É interessante observar que, nesta simulação, a diferença entre os resultados foi menor que na anterior - o que faz sentido, pois, na maior parte do tempo, o gráfico de $v(t)$ é muito próximo a uma reta horizontal, de modo que intuitivamente pode-se esperar que não haja muita

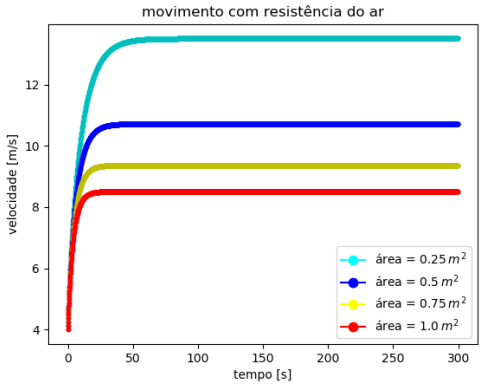

Figura 6: Gráficos de $v(t)$ para diferentes valores de A.

diferença entre aproximar a área por trapézios ou retângulos. Dessa vez, não há método analítico com que comparar os resultados, mas seria de se esperar, por argumento análogo ao aplicado anteriormente, que o valor exato ficasse compreendido entre as duas aproximações obtidas (e mais próximo da segunda).

(c) Caso com resistência - área variável

Conforme explicado em Métodos, fez-se a área assumir os valores de $0.25\,\mathrm{m}^2, 0.5\,\mathrm{m}^2, 0.75\,\mathrm{m}^2$ e $1.0\,\mathrm{m}^2$. Os resultados da simulação estão mostrados na Fig. 6. As velocidades terminais calculadas pela eq. (20) estão mostradas na tabela 1, juntamente com o tempo t^* para que se atingisse 99% da velocidade (resultado da simulação).

A (m^2)	v^* (m/s)	t^* (s)
0.25	13.502127623583272	43.600000000000350
0.5	10.716645796742485	26.900000000000112
0.75	9.3618624155326046	20.100000000000016
1.0	8.5058074056598016	16.299999999999962

Tabela 1: Velocidade terminal e tempo gasto para atingi-la (numa margem de 1%), para diferentes valores de A.

Conforme esperado, uma área de contato maior faz com que se atinja uma velocidade máxima menor.

Este sistema pode ser pensado como um modelo para descrever o movimento de um ciclista. Para isso, basta assumir que este seja capaz de transferir uma potência constante para o movimento da bicicleta, via esforço muscular. A relação observada (tanto teórica quanto computacionalmente) entre a área de contato A e a velocidade terminal v^* pode ser relacionado tanto com o *design* de bicicletas esportivas, que busca reduzir a área de contato, quanto com certas técnicas adotadas pelos profissionais. Dentre elas, destaca-se a postura: ciclistas geralmente se posicionam com o tronco inclinado, quase na horizontal, para minimizar a área frontal de contato do próprio corpo com o ar, consequentemente reduzindo a força de atrito.

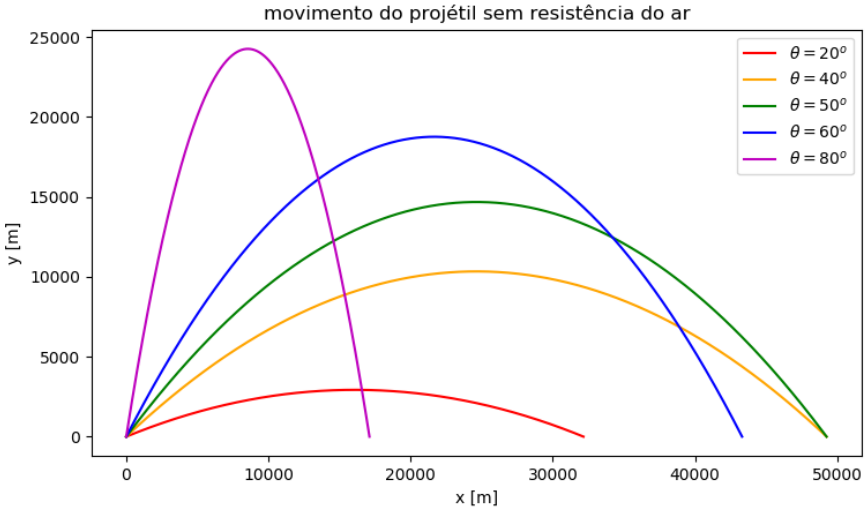

Figura 7: Trajetórias do projétil para diferentes ângulos de lançamento.

3.2 Lançamento de projéteis

(a) Caso sem atrito

Os cinco valores de θ (em graus) para os quais simularam-se as trajetórias foram lidos do arquivo `problema2_theta.in`:

```
20.0d0, 40.0d0, 50.0d0, 60.0d0, 80.0d0
```

As trajetórias obtidas a partir da simulação pelo método de Euler, para cinco valores de θ, estão mostradas na Fig. 7. É interessante notar que, visualmente, os alcances horizontais para $\theta = 40^o$ e $\theta = 50^o$ parecem coincidir. De fato, segue diretamente da eq. (23) que ângulos de tiro complementares resultam em alcances iguais.

Já o resultado da dependência de A com θ está mostrado na Fig. 8. Plotou-se ainda a curva dada pela eq. (23), observando-se boa concordância entre os dados, como seria de se esperar.

Outra consequência imediata da eq. (23) é que, fixado v_0, o alcance é máximo para $\theta = 45^o$. As simulações confirmaram este resultado: entre os ângulos testados, $\theta = 45^o$ deu maior alcance, o que permite assegurar ser esse o valor de máximo dentro de uma margem de 1.8^o (pois foi tal o espaçamento entre os ângulos utilizados). Para tal ângulo de tiro, o alcance foi de `50002.348924832790` m. Seguindo o que se pediu no enunciado, repetiu-se a simulação, para $\theta = 45^o$, com uma linha de código adicional para guardar o valor final de `t`, ou seja, o tempo de voo t^*_{max}. Obteve-se `t_max = 101.02000000001478`. Sendo de 0.01 s o passo temporal, o resultado da simulação pode ser traduzido, mais rigorosamente, como

$$t^*_{\text{max}} = (101.02 \pm 0.01)\,\text{s}$$

Segue facilmente das eqs. (22) que o tempo de voo, para $\theta = 45^o$, é dado por $\sqrt{2}v_0/g$. Fazendo este cálculo para $v_0 = 700\,\text{m/s}$ e $g = 9.8\,\text{m/s}^2$, obtemos `101.01525445522107` s como resposta. Notadamente, a diferença entre este resultado e o anterior é inferior à margem de erro dada pelo passo temporal adotado, $\Delta t = 0.01\,\text{s}$.

Calculando o alcance a partir da eq. (23), obteve-se o resultado de `49999.999999999993` m. Este número concorda bem com o resultado da simulação.

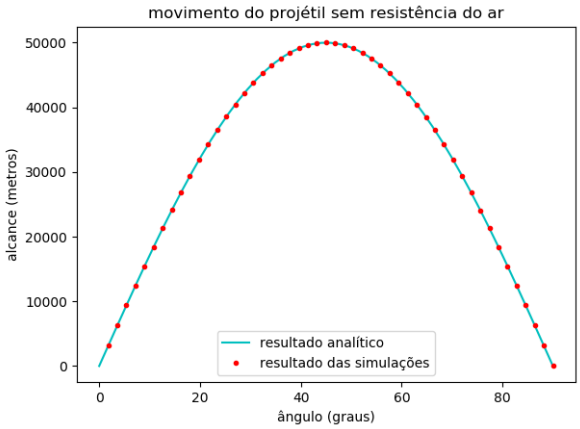

Figura 8: Alcance em função do ângulo de tiro.

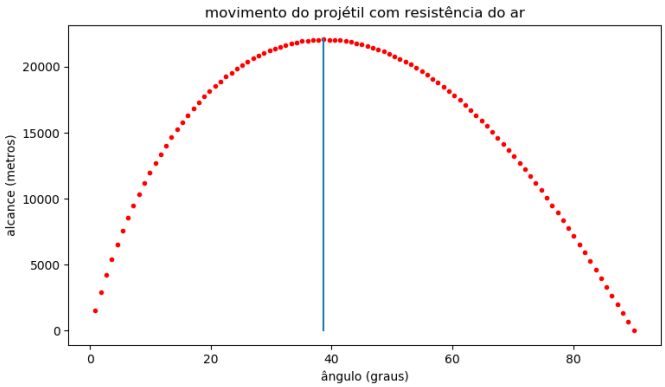

Figura 9: Alcance em função do ângulo de tiro, para o caso com resistência do ar.

(b) Caso com atrito, atmosfera homogênea

O gráfico de $A \times \theta$ obtido a partir da simulação está exibido na Fig. 9. Dessa vez, para poder inferir o ângulo $\theta_{\max}$ com maior precisão, simulou-se o alcance para 100 ângulos de tiro diferentes. (As trajetórias simuladas serão exibidas mais adiante, para fins comparativos.)

A inspeção dos dados mostrou que, dentre os ângulos testados, aquele que deu maior alcance foi `38.700000000000003` graus, sendo de `22070.195683770402` metros o alcance correspondente. Como o intervalo de amostragem dos ângulos foi de $(90^{\circ})/100 = 0.9^{\circ}$, o resultado permite afirmar que, sob as condições simuladas, o alcance é máximo para um ângulo de tiro de $(38.7 \pm 0.9)^{\circ}$. Assim como anteriormente, repetiu-se a simulação para esse ângulo específico, a fim de obter $t^*_{\max}$. O resultado foi `t_max = 69.559999999998681`. Levando em conta o erro dado pelo passo Δt, tem-se

$$t^*_{\max} = (69.56 \pm 0.01)\,\mathrm{s}$$

Podemos ver que a presença do atrito reduziu em mais de 50% o alcance máximo permitido para uma velocidade inicial fixa (e igual a 700 m/s). Naturalmente, o tempo de vôo nesta condição é também menor.

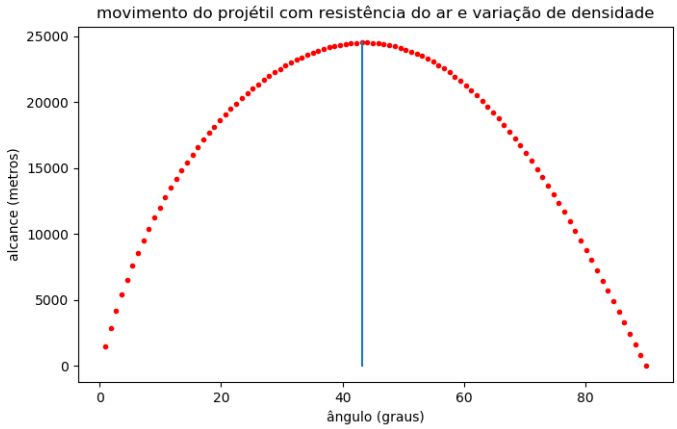

Figura 10: Alcance em função do ângulo de tiro, para o caso com resistência do ar e atmosfera heterogênea (adiabática).

(c) Caso com atrito, atmosfera heterogênea

O gráfico de $A \times \theta$ está mostrado na Fig. 10. Novamente, foram simulados 100 ângulos distintos.

Pela leitura dos dados numéricos, verificou-se que, para um ângulo de `43.200000000000003` graus, o alcance foi máximo e igual a `24522.603815402574` metros. De maneira análoga ao que se fez anteriormente, pode-se inferir que o ângulo de tiro máximo nestas condições é de $(43.2 \pm 0.9)^{\circ}$. Repetiu-se a simulação para este ângulo, a fim de encontrar $t^*_{\max}$, e o resultado numérico foi `t_max = 77.470000000002727`; segue então

$$t^*_{\max} = (77.47 \pm 0.01)\,\mathrm{s}$$

A tabela 2 retoma, para fins comparativos, os resultados de $\theta_{\max}$, $A_{\max}$ e $t^*_{\max}$ para os três diferentes casos simulados: (a), (b) e (c).

	$\theta_{\max}$ ($^{\circ}$)	$A_{\max}$ (m)	$t^*_{\max}$ (s)
(a)	45.0 ± 1.8	`50002.348924832790`	101.02 ± 0.01
(b)	38.7 ± 0.9	`22070.195683770402`	69.56 ± 0.01
(c)	43.2 ± 0.9	`24522.603815402574`	77.47 ± 0.01

Tabela 2: Comparação entre as três condições: sem resistência do ar, com resistência do ar numa atmosfera uniforme, e com resistência do ar numa atmosfera heterogênea.

Note que, para as três colunas, os valores da terceira linha estão compreendidos entre os das duas primeiras. Em outras palavras, os resultados da simulação numa atmosfera com variação de densidade foram intermediários entre aquelas com atmosfera homogênea e sem atmosfera.

Fisicamente, isso faz muito sentido. Introduzir o fator heterogeneidade do ar significa que em altitudes maiores a densidade (e, consequenetmente, a resistência) do ar será menor (eq. (27)). Na simulação com atmosfera heterogênea, o valor do coeficiente $\beta = \gamma/m$ (eqs. (25)) ao "nível do mar" ($y = 0$) foi adotado como igual ao valor constante deste coeficiente no caso da atmosfera uniforme. Ou seja: da forma como as simulações foram feitas, o "valor médio" da densidade do ar no segundo caso foi certamente menor que no primeiro, de modo que a "ação líquida" da resistência do ar deveria ser, como foi, menor no segundo caso. Isso fica nítido, como observado anteriormente, nos dados da Tabela 2.

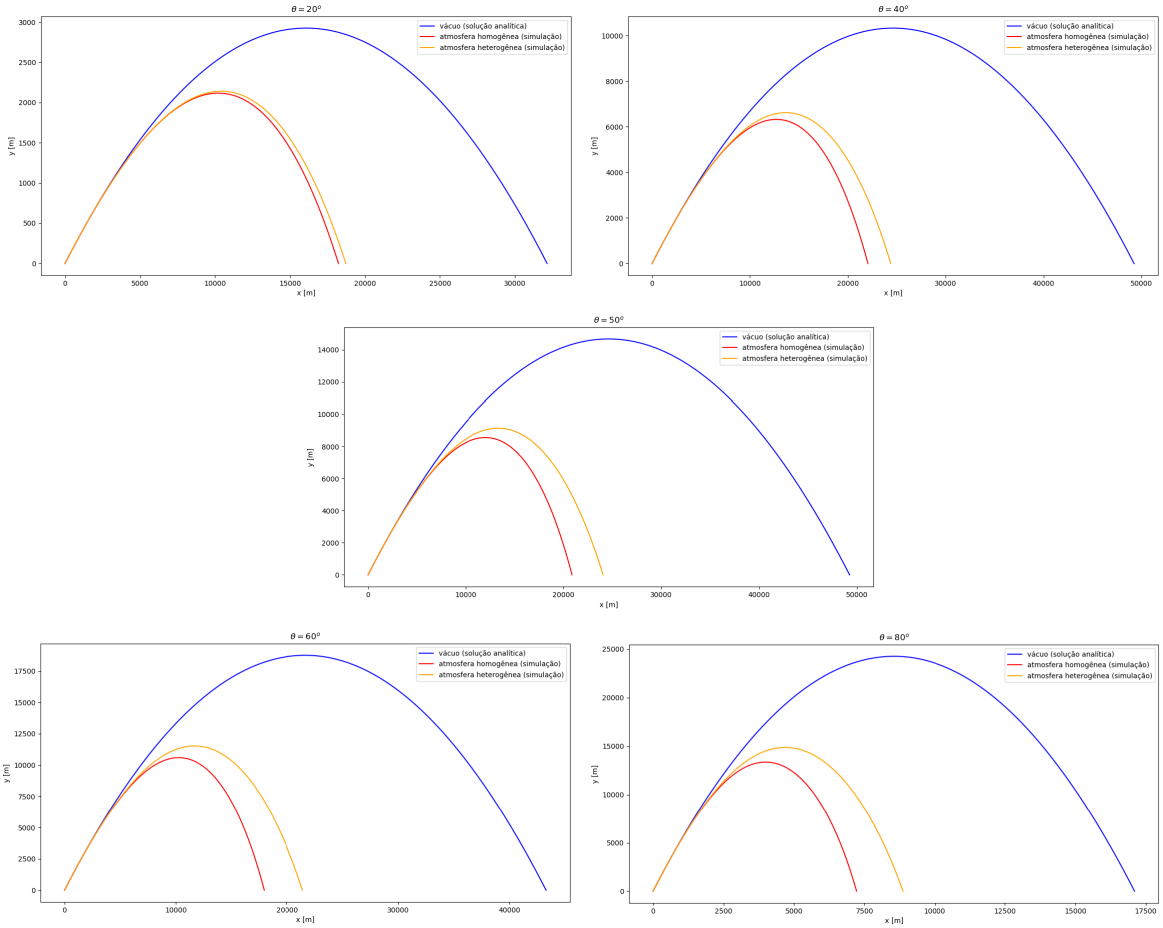

Figura 11: Trajetórias do projétil sob diferentes condições de simulação e para diferentes ângulos de tiro θ.

Finalmente, graficaram-se as trajetórias do projétil, para cinco ângulos de lançamento diferentes, tanto no caso da atmosfera homogênea quanto no da atmosfera adiabática. Para cada ângulo, graficaram-se ambas as situações na mesma figura, para fins comparativos. Foi feito, também, o gráfico da trajetória esperada sem resistência do ar, eqs. (22). Veja a Fig. 11.

Vemos que, em todos os casos, a trajetória simulada na atmosfera adiabática é intermediária entre as duas outras (atmosfera homogênea e "vácuo"), o que corrobora a discussão anterior. Ainda, é interessante notar que, para valores suficientemente pequenos de x, as três trajetórias são visualmente indistinguíveis, o que é uma consequência imediata do fato de as três terem o mesmo ângulo de lançamento.

(d) Variando a velocidade inicial

Seguindo o enunciado, repetiu-se a simulação com resistência do ar variável, para o ângulo de tiro que otimizou o alcance ($\theta = 43.2^{\circ}$), variando de $\pm 1\%$ a velocidade de lançamento. As trajetórias obtidas estão mostradas na Fig. 12, e os correspondentes alcances horizontais foram agrupados na Tabela 3.

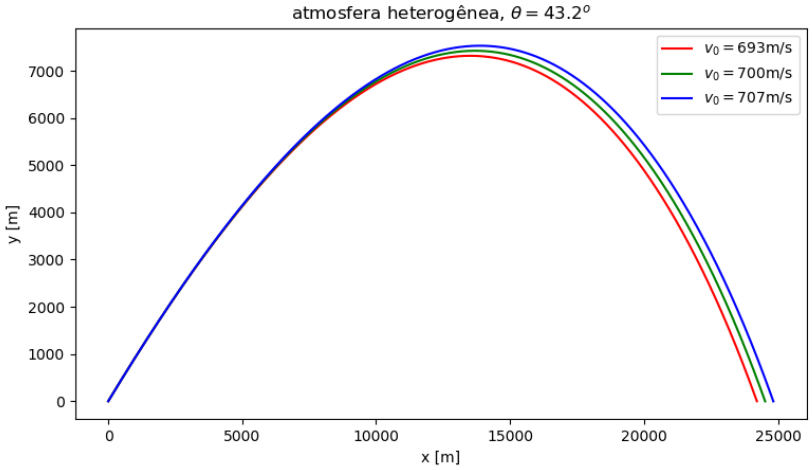

Figura 12: Trajetórias mostrando o efeito de uma pequena variação na velocidade de lançamento.

v_0 (m/s)	A (m)
693	`24214.876566101088`
700	`24522.603815402574`
707	`24828.992723347350`

Tabela 3: Alcance máximo para diferentes velocidades iniciais.

A conclusão mais imediata é que uma variação de "apenas" 1% na velocidade provocou uma variação de cerca de 300 m no alcance. No entanto, esse erro, relativamente ao alcance correspondente a $v_0 = 700\,\mathrm{m/s}$, é muito próximo de 1% (1.25%, em ambos os casos, truncando na segunda casa decimal). Essencialmente, o que esses cálculos parecem sugerir é apenas o fato óbvio de que, caso se queira atingir um alvo com precisão, é preciso ter um controle igualmente preciso sobre as condições em que o tiro é realizado (em particular, a velocidade de disparo).

3.3 Pêndulo simples

(a) Método de Euler

Conforme a sugestão, fez-se $m = 1\,\mathrm{kg}, L = 1\,\mathrm{m}$, $g = 10\,\mathrm{m/s^2}$. Para os itens (a) e (b), tomou-se ainda $\theta_0 = 30^o$, $\Delta t = 0.005\,\mathrm{s}$ e $t_{\max} = 20\,\mathrm{s}$. Essas entradas foram lidas no arquivo `problema3a.in`, conforme explicado em Métodos:

```
30.0d0, 10.0d0, 1.0d0, 0.005, 20.0d0, 1.0d0
```

Os gráficos mostrando os resultados obtidos para $\theta \times t$ e $E \times t$ estão mostrados na Fig. 13.

Vemos que a energia mecânica do sistema sempre aumenta, o mesmo ocorrendo com a amplitude da oscilação. De fato, o método de Euler não funciona bem para sistemas periódicos.

(b) Método de Euler-Cromer

Os parâmetros do programa foram exatamente os mesmos do item anterior. A Fig. 14 mostra o resultado da simulação pelo método de Euler-Cromer, em contraste com os resultados do método de Euler.

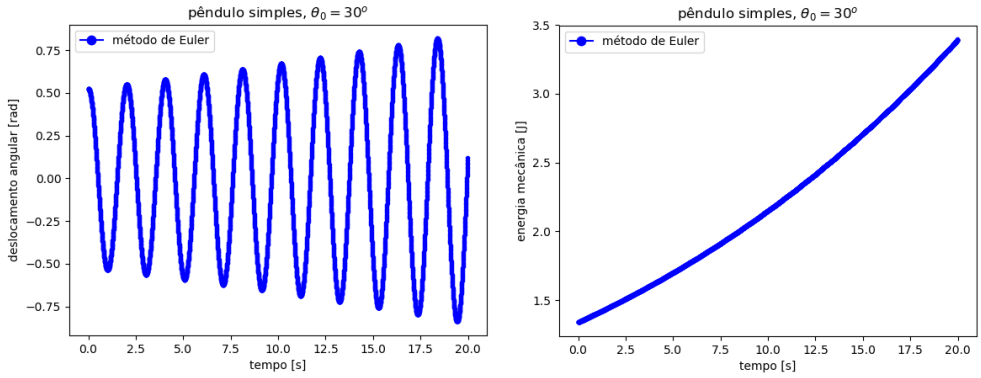

Figura 13: Método de Euler aplicado ao pêndulo simples.

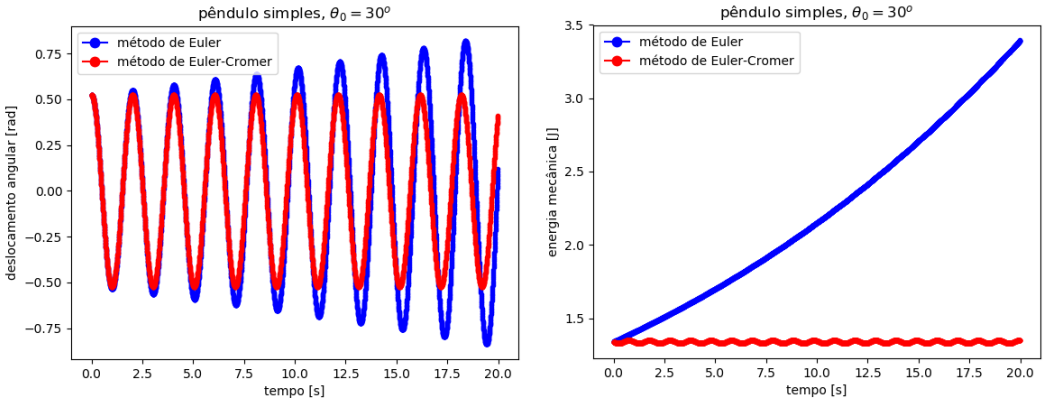

Figura 14: Pêndulo simpmles: comparação entre os métodos de Euler e Euler-Cromer.

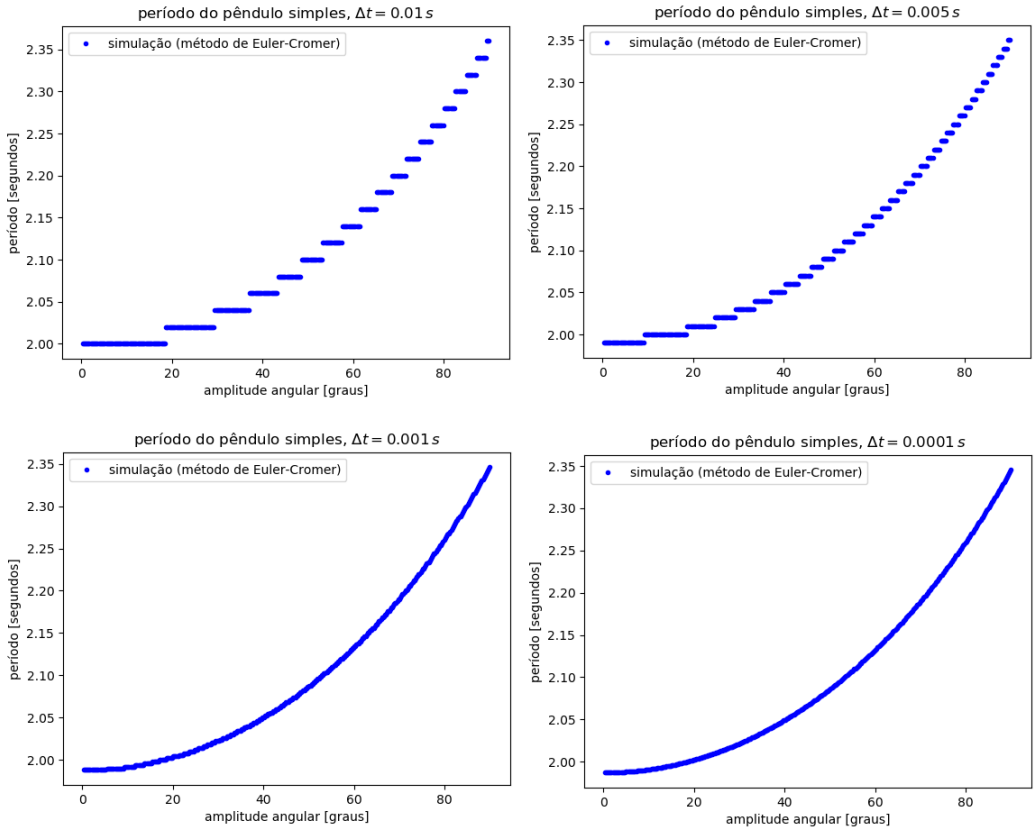

Figura 15: Período do pêndulo simples.

Vemos que, de fato, o método de Euler-Cromer funciona muito melhor para este sistema. Embora não permaneça inalterada a cada passo, a energia mecânica oscila com pequena amplitude em torno de um valor médio; no gráfico de $\theta(t)$, observa-se que a oscilação do pêndulo é de fato periódica, e a amplitude máxima por oscilação permanece inalterada.

(c) Período de oscilação

O programa mostrado em Métodos foi executado para diferentes valores do passso temporal Δt, a fim de que se pudessem observar os efeitos desta escolha nos resultados. Em todos os casos, os ângulos assumiram 400 valores igualmente espaçados entre 0 e 90° (excluído o limite inferior), ou seja, fez-se `n = 400` no código. Os gráficos de $T \times \theta_0$ assim obtidos estão mostrados na Fig. 15.

O aspecto de escada dos primeiros dois gráficos é uma consequência óbvia do fato de que o resultado do período não pode ser mais preciso que a resolução temporal da simulação. Como nesses casos a amostragem de valores de θ_0 foi muito mais particionada que os possíveis resultados de T, tivemos ângulos θ_0 diferentes dando períodos T iguais. Para o último caso, $\Delta t = 10^{-4}$ s, o refinamento do passo temporal já foi suficiente para que tal anomalia do tempo discreto não fosse observada (Fig. 16).

Como vimos, a dependência entre T com θ_0 também foi calculada por meio da integral elíptica (31). Os resultados destes cálculos estão mostrados na Fig 16, juntamente com o resultado obtido

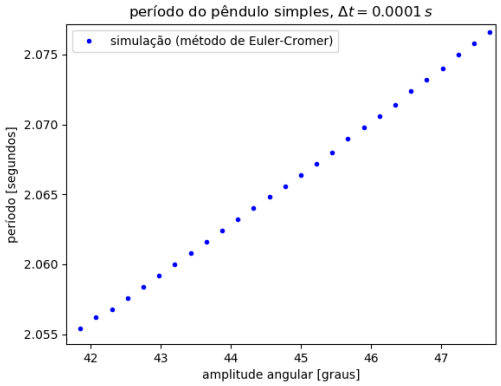

Figura 16: *Zoom* do gráfico de $T \times \theta_0$ para $\Delta t = 10^{-4}$ s.

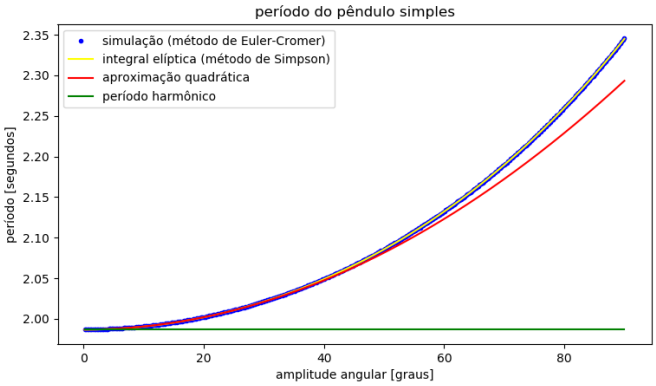

Figura 17: Período do pêndulo, em função da amplitude angular, por diferentes métodos.

na simulação (para $\Delta t = 10^{-4}$ s), a aproximação quadrática (32) e a aproximação harmônica (33). A concordância entre os resultados das simulações e a integração por Simpson é tamanha que os gráficos se encontram tão perfeitamente superpostos quanto é possível distinguir a olho nu.

A aproximação quadrática também parece funcionar bem para θ_0 não muito grande. Uma pergunta curiosa é: qual o erro relativo cometido nessa aproximação, bem como na aproximação harmônica? O gráfico da Fig. 18 mostra esses erros, tomando sempre por base os resultados provenientes da simulação numérica (método de Euler-Cromer). Em livros didáticos de Física, é comum se encontrar a afirmação de que a aproximação harmônica ($\sin\theta \approx \theta$) funciona bem para $\theta < 10^{\circ}$. De fato, verificamos no gráfico que, para $\theta = 9.9^{\circ}$, o erro cometido no período por tal aproximação é de apenas cerca de 0.2%.

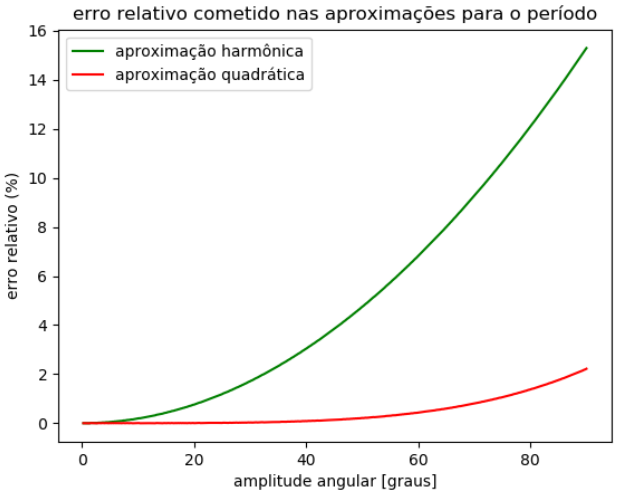

Figura 18: Erro cometido nas aproximações para o período do pêndulo simples.