Instituto de Física de São Carlos
Universidade de São Paulo

[7600017] - Introdução à Física Computacional

Projeto 4

Docente:
Francisco Castilho Alcaraz

Aluno:
Breno Henrique Pelegrin da Silva (13687303)

23 de novembro de 2023

Sumário

Lista de Figuras

Códigos

1 Contexto

Nesse projeto, foram elaborados diversos programas em *Fortran 77* para explorar o movimento oscilatório de um pêndulo simples. Inicialmente, considerou-se a aproximação de pequenos ângulos, porém, posteriormente, a equação diferencial do pêndulo foi analisada sem aproximações, tornando possível explorar a evolução temporal do fenômeno em sua plenitude, bem como entender a emergência do caos e a análise deste para diferentes configurações do sistema. Os gráficos e análises gráficas dispostos neste relatório foram

2 Tarefa A

Nessa tarefa, estudou-se o movimento de um pêndulo simples, utilizando a aproximação de ângulos pequenos. Dado um pêndulo de comprimento l e massa m que oscila formando um ângulo θ com o eixo vertical, sua equação de movimento $\theta(t)$ é dada pela Equação Diferencial Ordinária (EDO) exposta na equação 1.

$$ma_\theta = ml\frac{d^2\theta}{dt^2} = -mg\sin(\theta)$$

$$\frac{d^2\theta}{dt^2} = \frac{-g}{l}\sin(\theta) \tag{1}$$

Supondo que o pêndulo exerça pequenas oscilações, ou seja, que $\theta \approx 0$, então pode-se considerar que $sin(\theta) \approx \theta$ e, portanto, é possível reescrever a EDO para o caso de ângulos pequenos, como é exposto na equação 2, onde $\omega(t) = \frac{d\theta}{dt}$.

$$\theta \to 0 \implies \frac{d^2\theta}{dt^2} = \frac{-g}{l}\theta$$

$$\frac{d\theta}{dt} = \omega(t)$$

$$\frac{d\omega}{dt} = \frac{-g}{l}\theta(t) \tag{2}$$

A fim de analisar o comportamento de $\theta(t)$ numericamente, pode-se utilizar o método de Euler. No método de Euler, o eixo temporal é particionado em intervalos de tamanho Δt, de forma que para cada i-ésimo intervalo, o tempo t_{i+1} é dado pela equação 5.

Considerando essa discretização, pode-se aproximar o comportamento da função $f(t)$ no intervalo $i+1$ por $f(t_{i+1}) = f(t_i) + \Delta t f'(t_i)$, ou seja, somando uma pequena variação da função no intervalo (sua derivada multiplicada pelo tamanho Δt) ao valor da função no intervalo anterior. Assim, aplicando esse método à EDO do pêndulo simples com aproximação de pequenos ângulos, obtém-se as equações numéricas 3 e 3, que podem ser computadas em um programa *Fortran*.

$$w_{i+1} = w_i - \frac{g}{l}\theta_i \Delta t \tag{3}$$

$$\theta_{i+1} = \theta_i + \omega_i \Delta t \tag{4}$$

$$t_{i+1} = t_i + \Delta t,\, t_0 = 0 \tag{5}$$

A partir dos resultados apresentados na seção 2.1, observa-se que a amplitude de oscilação e a energia mecânica sofrem um pequeno aumento em função do tempo no método de Euler, o que representa uma descrição física errônea do fenômeno, uma vez que a energia deveria se conservar.

Para contornar esse problema, é realizada uma modificação no método de Euler, substituindo o termo ω_i na equação 4 por w_{i+1}, sendo essa alteração conhecida como método de Euler-Cromer. As equações numéricas para o método de Euler-Cromer são expostas nas equações 6 e 7.

$$w_{i+1} = w_i - \frac{g}{l}\theta_i \Delta t \tag{6}$$

$$\theta_{i+1} = mod(\theta_i + \omega_{i+1}\Delta t, 2\pi),\text{ com } t = i\Delta t \tag{7}$$

$$t_{i+1} = t_i + \Delta t,\text{ com } t_0 = 0$$

A operação de divisão modular é emprega em θ_{i+1} para que θ esteja sempre entre 0 e 2π. A energia mecânica do sistema pode ser calculada utilizando os valores de θ_i para cada passo, de acordo com a equação 8.

$$E_i = T_i + U_i = \frac{1}{2}m(\omega_i l)^2 + mgl[1 - cos(\theta_i)] \tag{8}$$

Para os cálculos dessa tarefa, foi considerado um ângulo inicial $\theta_0 = 0.14$ rad $\approx 8°$, $\Delta t = 10^{-2}$, $g = l = 9.8$, $m = 1.0$ e o número de passos simulados foi 2000 passos. O código do programa desenvolvido para esse projeto é exposto no Código 1, e a explicação para esse código é dada logo em seguida, no verbete 1. Os resultados obtidos e a análise deles estão expostos na seção 2.1.

Código 1: Código-fonte da tarefa A

```fortran
      program TarefaA
         implicit real*8 (a-h,o-z)
         parameter(dt = 1e-2)
         parameter(ipassos = 2000)
         parameter(theta0 = 0.14d0)
         parameter(g = 9.8d0)
         parameter(al = 9.8d0)
         parameter(m = 1.0d0)
         dimension t(ipassos), w(ipassos), e(ipassos), theta(ipassos)
         dimension w_ec(ipassos), e_ec(ipassos), theta_ec(ipassos)
         pi = 4.0d0*atan(1.0d0)
         theta(1) = theta0
         theta_ec(1) = theta0
         w(1) = 0.0d0
         w_ec(1) = 0.0d0
         e(1) = m*g*al*(1-cos(theta0))
         e_ec(1) = m*g*al*(1-cos(theta0))
         t(1) = 0.0d0
         open(50, file="saida-a-13687303-theta_t.csv")
         open(51, file="saida-a-13687303-theta_ec_t.csv")
         open(60, file="saida-a-13687303-e_t.csv")
         open(61, file="saida-a-13687303-e_ec_t.csv")
         write(50,100)
 100     format("t,theta")
         write(51, 101)
 101     format("t,theta_ec")
         write(60,102)
 102     format("t,e")
         write(61, 103)
 103     format("t,e_ec")
         write(50,*) t(1), ",", theta(1)
         write(51,*) t(1), ",", theta_ec(1)
         write(60,*) t(1), ",", e(1)
         write(61,*) t(1), ",", e_ec(1)
         do i=1, ipassos-1, 1
            t(i+1) = i*dt
            w(i+1) = w(i) - (g/al)*theta(i)*dt
            w_ec(i+1) = w_ec(i) - (g/al)*theta_ec(i)*dt
            theta(i+1) = mod(theta(i) + w(i)*dt, 2.0d0*pi)
            theta_ec(i+1) = mod(theta_ec(i) + w_ec(i+1)*dt, 2.0d0*pi)
c           E = U + K = mgl * (1 - cos(theta)) + 0.5 * m * (w*l)^2
            e(i+1) = m*g*al*(1-cos(theta(i+1)))+ 0.5d0*m*(w(i+1)*al)**2
            e_ec(i+1) = m*g*al*(1-cos(theta_ec(i+1)))+0.5d0*m*(w_ec(i+1
     &)*al)**2
            write(50, *) t(i+1), ",", theta(i+1)
            write(51, *) t(i+1), ",", theta_ec(i+1)
            write(60, *) t(i+1), ",", e(i+1)
            write(61, *) t(i+1), ",", e_ec(i+1)
         end do
         close(50)
         close(51)
         close(60)
         close(61)
      end program
```

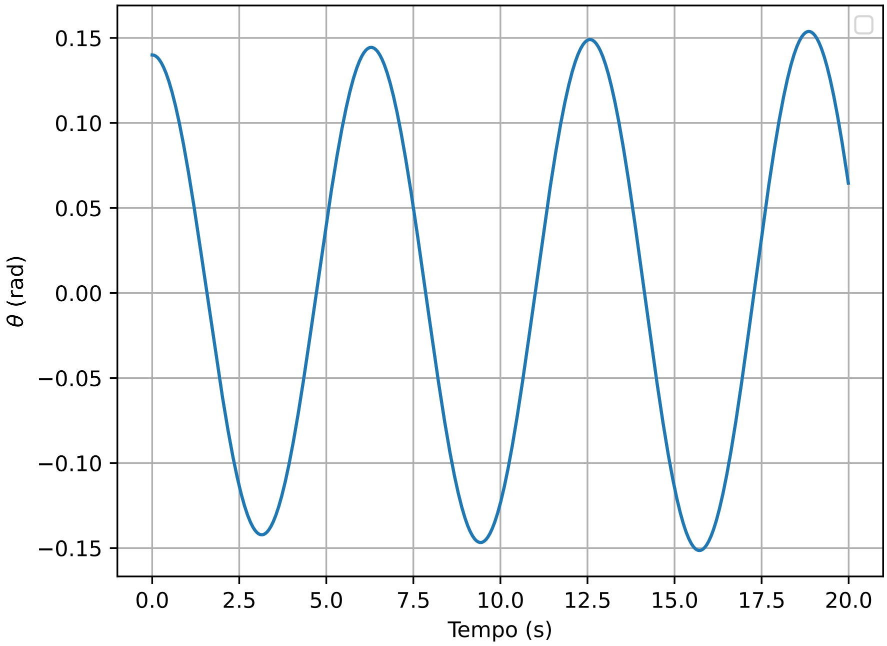
Verbete 1: Explicação do código da tarefa A

Inicialmente, define-se os parâmetros e os vetores a serem utilizados durante a simulação do pêndulo. Os vetores são inicializados com o tamanho de acordo com a quantidade N de passos. Como o pêndulo parte do repouso, tem-se que $\omega_0 = 0$.
Em seguida, após as definições e abertura do arquivo de saída, realiza-se um laço i de 1 a $N-1$, de 1 em 1. A cada passo, são calculadas as equações numéricas para o método de Euler e o método de Euler-Cromer, assim como a energia mecânica total do sistema calculada utilizando os valores dos dois métodos.
Os valores calculados para cada $theta_i$ e E_i em função de t_i são escritos em um arquivo para cad a método numérico avaliado, para fins de comparação.

2.1 Resultados e análise

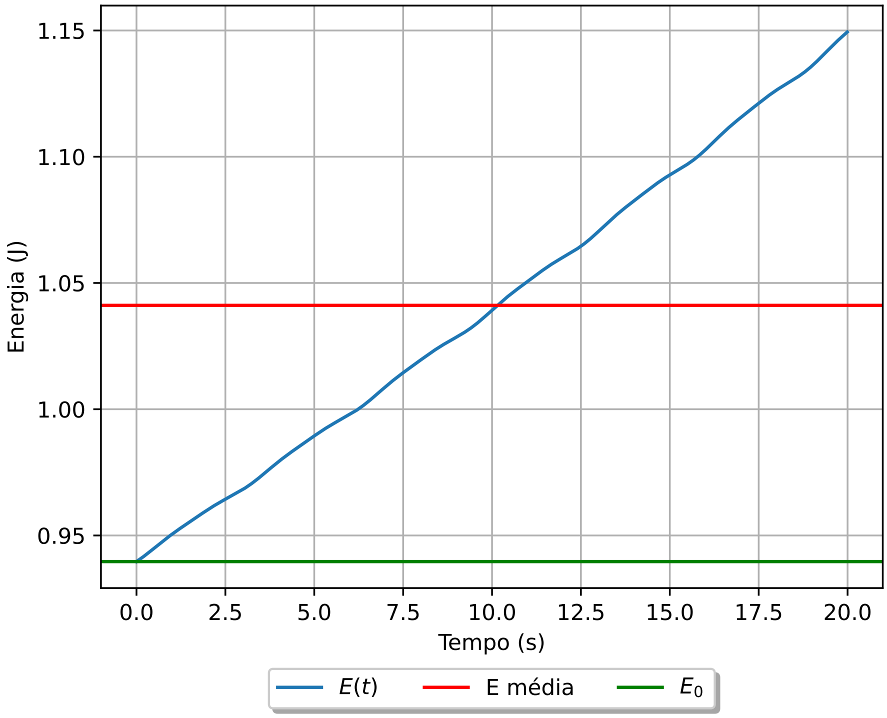
A partir dos resultados obtidos nessa tarefa, foram criados os gráficos de $\theta(t)$ e $E(t)$ para os dois métodos numéricos utilizados, de Euler e de Euler-Cromer. Os gráfico de $\theta(t)$ e $E(t)$ para o método de Euler são exibidos nas figuras 1 e 2, respectivamente, e os gráficos de $\theta(t)$ e $E(t)$ para o método de Euler-Cromer são exibidos nas figuras 3 e 4, respectivamente.

Figura 1: $\theta(t)$ para o Método de Euler



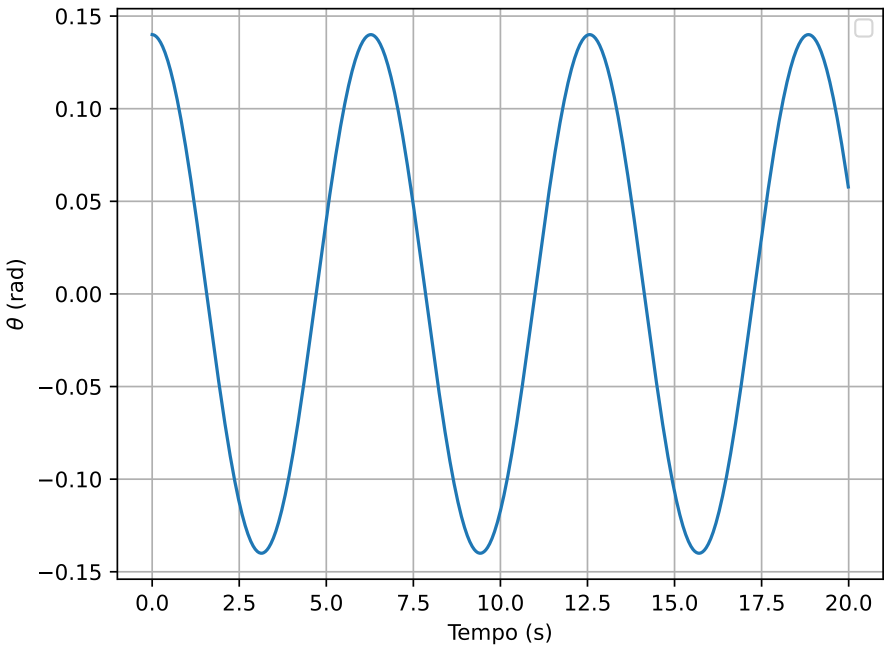
Fonte: Autoria própria

Figura 2: $E(t)$ para o Método de Euler



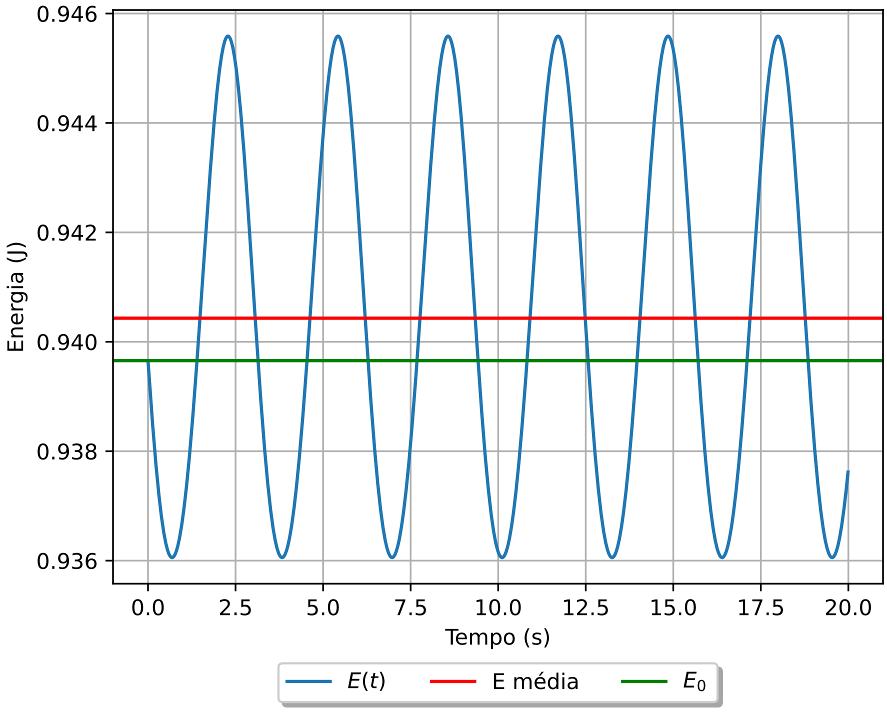
Fonte: Autoria própria

Figura 3: $\theta(t)$ para o Método de Euler-Cromer



Fonte: Autoria própria

Figura 4: $E(t)$ para o Método de Euler-Cromer



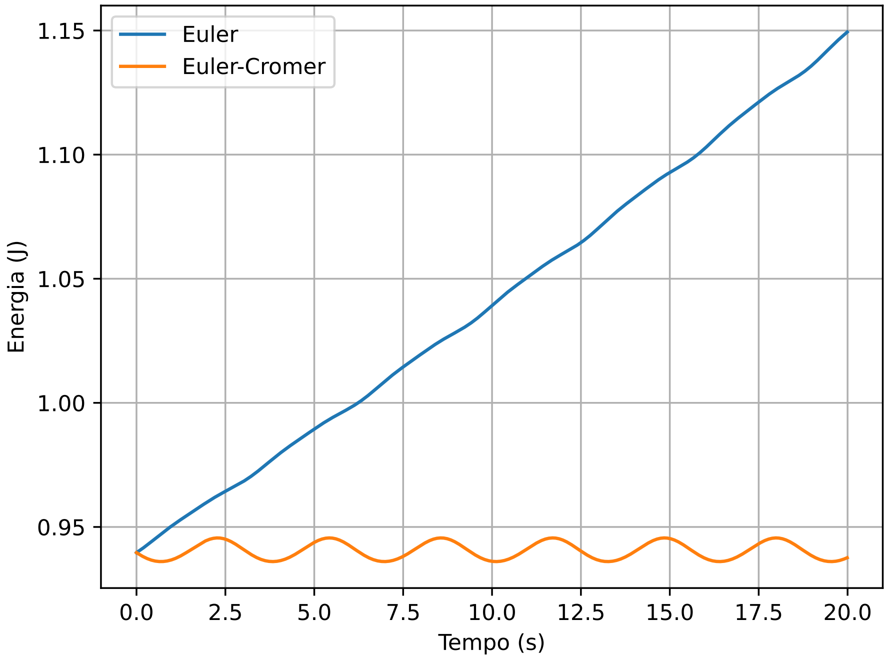
Fonte: Autoria própria

Ao observar os gráficos de $\theta(t)$ e $E(t)$ para os dois métodos, observa-se que, no método de Euler, a amplitude de $\theta(t)$ e o valor de $E(t)$ tendem a crescer com o tempo, o que representa uma inconsistência física, já que a amplitude de $\theta(t)$ e o valor de $E(t)$ deveriam ser aproximadamente constantes no tempo, exceto por variações de ponto flutuante.

Já nos gráficos para o método de Euler-Cromer, observa-se que a amplitude de $\theta(t)$ permanece constante no tempo e a energia $E(t)$ varia na escala de Δt, ou seja, é aproximadamente constante no tempo, tendo apenas variações de ponto flutuante. Assim, o método de Euler-Cromer descreve o fenômeno físico com maior precisão que o método de Euler.

A fim de comparar os dois métodos, foi criado um gráfico conjunto de $E(t)$ para os dois métodos numéricos, exibido na figura 5. Ao analisar o gráfico, observa-se que, no método de Euler, a energia cresce linearmente com o tempo, enquanto no método de Euler-Cromer, a energia oscila regularmente em torno de um ponto, permanecendo aproximadamente constante no tempo.

Figura 5: $E(t)$ para o Método de Euler e Método de Euler-Cromer



Fonte: Autoria própria

3 Tarefa B

Nessa tarefa, utilizando o Método de Euler-Cromer, explorou-se o fenômeno do pêndulo simples sem aproximação de ângulos pequenos, e acrescentando um termo resistivo e uma força externa na EDO, tornando a simulação mais geral e próxima da realidade. A EDO que descreve $\theta(t)$ com os novos termos acrescentados é exibida na equação 9, onde γ é o fator de resistência do meio, F_0 é a amplitude da força externa e Ω é a frequência da força externa.

$$\frac{d\omega}{dt} = -\frac{g}{l}sin(\theta) - \gamma\frac{d\theta}{dt} + F_0 sin(\Omega t) \tag{9}$$

$$\frac{d\theta}{dt} = \omega$$

3.1 Parte B1 e B2

Nas partes B1 e B2, o período do movimento oscilatório foi calculado para diversos ângulos iniciais θ_0 utilizando três métodos distintos:

1. O primeiro consiste em realizar a simulação numérica de vários pêndulos com ângulos iniciais θ_0 e determinar o período a partir da simulação.

2. O segundo consiste em resolver a EDO para encontrar o período, descrito pela integral da equação 12, que deve ser calculada numericamente considerando as singularidades da função.

3. O terceiro consiste em calcular o período através da equação aproximada 13, obtida a partir de uma expansão em série do período.

Ao discretizar a EDO da equação 9 utilizando o Método de Euler-Cromer, obtém-se as equações 10 e 11.

$$\omega_{i+1} = \omega_i + \Delta t \left[\frac{-g}{l} sin(\theta_i) - \gamma\omega_i + F_0 sin(\Omega t_i) \right] \tag{10}$$

$$\theta_{i+1} = mod(\theta_i + w_{i+1}\Delta t, 2\pi) \tag{11}$$

$$t_{i+1} = t_i + \Delta t, \text{ com } t_0 = 0$$

A integral da equação 12 pode ser obtida ao multiplicar a EDO 1 por $\dot{\theta}$, aplicar as condições iniciais $\theta(t = 0) = \theta_0$ e $\dot{\theta}(t = 0) = 0$, e separar as variáveis θ e t, como exposto abaixo.

$$\dot{\theta}\ddot{\theta} + \omega^2 sin(\theta)\dot{\theta} = 0 \implies (\frac{1}{2}\dot{\theta}^2)' - (cos(\theta))'\omega^2 = 0$$

$$\int (\frac{1}{2}\dot{\theta}^2) - cos(\theta)\omega^2 dt \implies \frac{1}{2}\dot{\theta}^2 - \omega^2 cos(\theta) = C$$

$$\theta(0) = \theta_0 \text{ e } \dot{\theta}(0) = 0 \implies C = -w^2 cos(\theta_0)$$

$$\frac{d\theta}{dt} = \pm\omega\sqrt{2}\sqrt{cos(\theta) - cos(\theta_0)}$$

$$\int_0^{\frac{T}{2}} dt = \frac{T}{2} = \pm\frac{1}{\omega\sqrt{2}} \int_{-\theta_0}^{\theta_0} \frac{d\theta}{\sqrt{cos(\theta) - cos(\theta_0)}}$$

$$T = \sqrt{\frac{2l}{g}} \int_{-\theta_0}^{\theta_0} \frac{d\theta}{\sqrt{cos(\theta) - cos(\theta_0)}} \tag{12}$$

Aproximando o período em série, obtém-se a equação 13.

$$T \approx 2\pi\sqrt{\frac{l}{g}} \left[1 + \frac{\theta_0^2}{16} \right] \tag{13}$$

O código elaborado para essa tarefa é exposto no Código 2, e sua explicação é dada logo em seguida, no verbete 2.

Código 2: Código-fonte da tarefa B, parte B1-B2

```fortran
      program TarefaB1eB2
         implicit real*8 (a-h,o-z)
         parameter (iangulos = 100)
         parameter (idivisoes = 5000)
         dimension theta(iangulos)

         pi = 4.0d0*atan(1.0d0)
         theta(1) = pi/2.0d0

         open(50, file='saida-b1-b2-13687303-periodos.csv')
         write(50,100)
 100     format("theta,periodo_sim,periodo_int,periodo_apx")
         do i=1, iangulos-1, 1
            theta(i+1) = theta(i) - pi/(2*iangulos)
            periodo = 0.0d0
            eps = 1e-3
            do j=1, 10, 1
               periodo = periodo + simulate(theta(i), 0d0, 0d0, 0d0)
            end do
            periodo = periodo/10.0d0
            periodo_int = spint(-theta(i)+eps, theta(i)-eps, idivisoes,
     &theta(i))*sqrt(2.0d0)+ 4*sqrt(2.0d0)*sqrt(eps/sin(theta(i)))
            periodo_apx = 2.0d0*pi*(1 + (theta(i)**2)/16.0d0)
            write(50,*) theta(i), ",", periodo, ",", periodo_int, ",",
     &periodo_apx
         end do
         close(50)
      end program

      function simulate(theta0, gamma, f0, omega)
         implicit real*8 (a-h,o-z)
         parameter(ipassos = 3000)
         parameter(dt = 1e-2)
         parameter(g = 9.8d0)
         parameter(al = 9.8d0)
         parameter(m = 1.0d0)

         dimension t(ipassos), w(ipassos), e(ipassos), theta(ipassos)

         simulate = 0.0d0
         pi = 4.0d0*atan(1.0d0)

         theta(1) = theta0
         w(1) = 0.0d0
         e(1) = m*g*al*(1-cos(theta0))
         t(1) = 0.0d0
         iosc = 0

         do i=1, ipassos-1, 1
            t(i+1) = i*dt
```

```
              w(i+1) = w(i) + (- (g/al)*sin(theta(i)) - gamma*w(i) + f0*s
     &in(omega*t(i)))*dt
              if(w(i+1) * w(i) .lt. 0) iosc = iosc + 1
              if(iosc .eq. 2) then
                 simulate = t(i)
                 return
              end if
              theta(i+1) = mod(theta(i) + w(i+1)*dt, 2.0d0*pi)
c             E = U + K = mgl * (1 - cos(theta)) + 0.5 * m * (w*l)^2
              e(i+1) = m*g*al*(1-cos(theta(i+1)))+ 0.5d0*m*(w(i+1)*al)**2
           end do
      end function

      function func(x, x0)
         implicit real*8 (a-h,o-z)
         func = 1.0d0/sqrt(cos(x) - cos(x0))
         return
      end function

      function fn(x0, n, h, b0)
         implicit real*8 (a-h,o-z)
         fn = func(x0 + n*h, b0)
         return
      end function fn

      function spint(a,b,n,b0)
         implicit real*8 (a-h,o-z)
         spint=0.0d0
         h=(b-a)/real(n)
         do i=1, n-1, 2
            x0=a+i*h
            spint=spint+(h/3.0d0)*(fn(x0, 1, h, b0) + 4.0d0*fn(x0, 0, h
     &,b0) + fn(x0,-1, h, b0))
         end do
         return
      end function

      function trint(a,b,n,b0)
         implicit real*8 (a-h,o-z)
         trint=0.0d0
         h=(b-a)/n
         do i=1, n-1, 2
            x0=a + i*h
            trint=trint+(h/2.0d0)*(fn(x0, -1, h, b0) + 2.0d0*fn(x0, 0,
     &h, b0)+fn(x0,1,h,b0))
         end do
         return
      end function
```

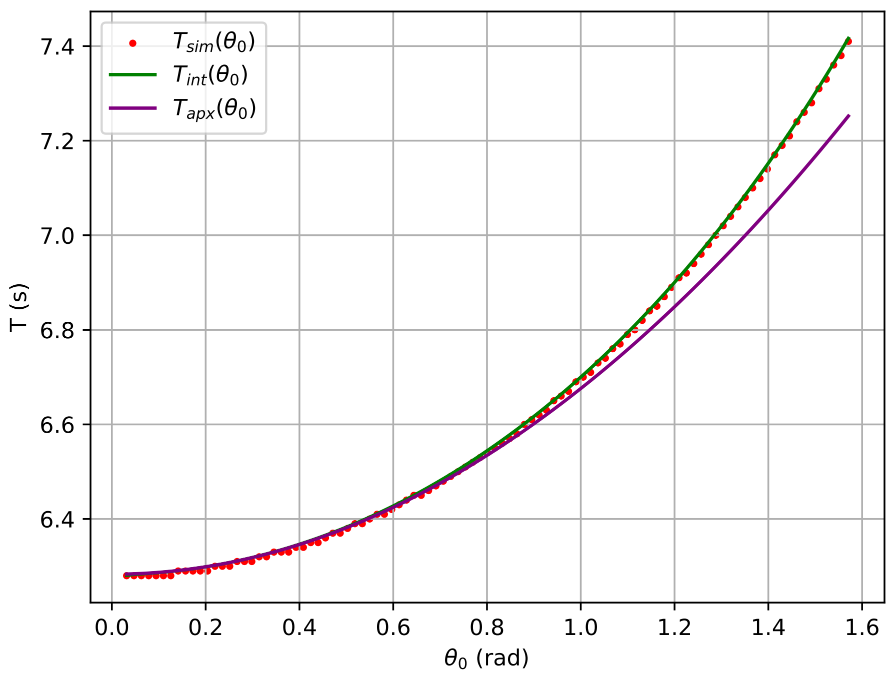
Verbete 2: Explicação do código da tarefa B

Inicialmente, define-se o número de passos e um vetor com vários valores de θ_0 para os quais o período será calculado, e depois é aberto o arquivo de saída de dados. A fim de tornar o código mais simples, a parte de simulação foi transformada em uma função *simulate*, que recebe os parâmetros da simulação e retorna o período do movimento. O período obtido a partir da simulação é calculado da seguinte maneira: conta-se quantas vezes $\omega(t)$ trocou de sinal e, quando a contagem for 2, tem-se um período e a função retorna o valor do tempo t_i atual. Além disso, para cada θ_0, a integral do período é calculada numericamente utilizando o método de Boole, e a expressão aproximada do período também é calculada. Após calcular cada período, os resultados de cada método em função do ângulo θ_i são escritos no arquivo de saída.

3.1.1 Resultados e análise

A partir dos períodos obtidos pela integral (T_{int}), pela simulação (T_{sim}) e pela equação aproximada (T_{apx}), foi criado um gráfico conjunto de $T(\theta_0)$ para os três métodos utilizados, a fim de compará-los, que é exposto na figura 6.

Figura 6: $T(\theta_0)$ para cada método calculado.

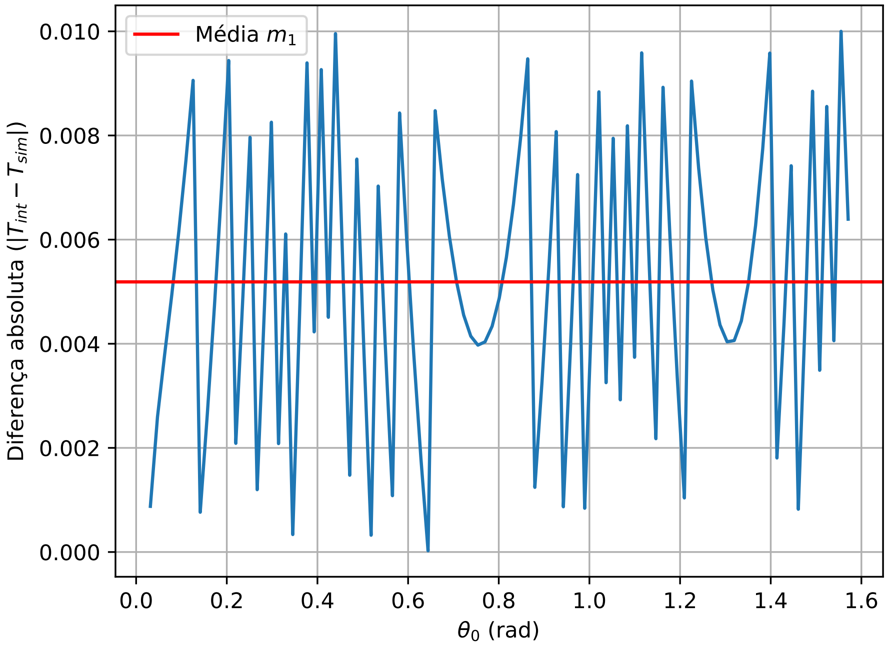


Fonte: Autoria própria

Ao observar a figura 6, pode-se concluir que os métodos da integral e da simulação fornecem resultados aparentemente próximos, sendo difícil discernir graficamente o erro entre esses dois métodos.

Dessa forma, para comparar o método da integral e da simulação de maneira mais precisa, foi criado um gráfico da diferença absoluta $|T_{int} - T_{sim}|$ entre os métodos em função de θ_0, exibido na figura 7. A partir da análise do gráfico, observa-se que os métodos fornecem resultados muito semelhantes, uma vez que a diferença absoluta oscila na ordem de Δt em torno da média, que está na escala de 10^{-3}, ou seja, o erro entre os dois métodos é desprezível.

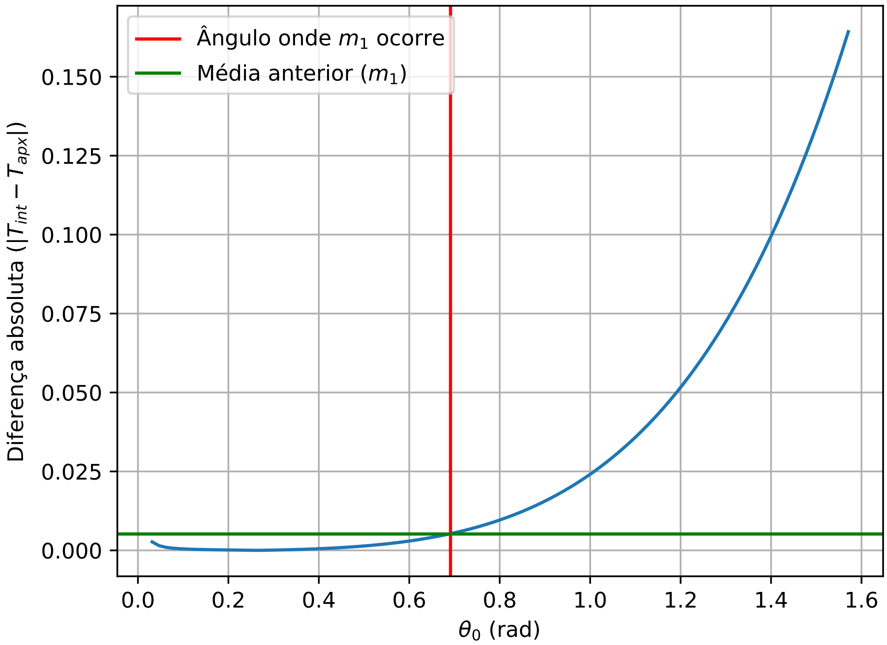
Figura 7: Diferença absoluta entre T_{sim} e T_{int}



Fonte: Autoria própria

Analogamente, a fim de comparar os resultados da equação aproximada em relação aos métodos da integral e da simulação, foi criado o gráfico da diferença absoluta $|T_{apx} - T_{int}|$ entre o método aproximado e o método da integral, exposto na figura 8. No gráfico, a média da diferença absoluta anterior, entre T_{int} e T_{sim}, está evidenciada pela linha verde, e o ângulo θ_0 onde ela ocorre está marcado pela linha vermelha. Assim, é possível encontrar a partir de qual ângulo o método aproximado fornece resultados com erro maior que o método da integral e da simulação.

Figura 8: Diferença absoluta entre T_{apx} e T_{int}



Fonte: Autoria própria

Ao analisar o gráfico da figura 8, observa-se que o método aproximado funciona bem para ângulos menores que 0.7 rad, todavia, para ângulos maiores que 0.7 rad, o erro começa a crescer exponencialmente, fornecendo resultados distantes da realidade. Assim, podemos concluir que os métodos da integral e da simulação fornecem resultados aproximadamente iguais, e o método aproximado fornece bons resultados para ângulos de até 0.7 rad.

3.2 Parte B3 e B4

Código 3: Código-fonte da tarefa B, parte B3

```
      program TarefaB3
         implicit real*8 (a-h,o-z)
         parameter(ipassos = 9000)
         parameter(g = 9.8d0)
         parameter(al = 9.8d0)
         parameter(m = 1.0d0)

         dimension t(ipassos), w(ipassos), e(ipassos), theta(ipassos)

         pi = 4.0d0*atan(1.0d0)

         theta0 = 0.14d0
         omega = 0.0d0
         gamma = 0.05d0
         f0 = 0.0d0
         dt=1e-2
```

```

      open(50, file="saida-b3-13687303-theta_t.csv")
      write(50,100)
100   format("t,theta")

      theta(1) = theta0
      w(1) = 0.0d0
      e(1) = m*g*al*(1-cos(theta0))
      t(1) = 0.0d0

      do i=1, ipassos-1, 1
         t(i+1) = i*dt
         w(i+1) = w(i) + (- (g/al)*sin(theta(i)) - gamma*w(i) + f0*s
     &in(omega*t(i)))*dt
         theta(i+1) = mod(theta(i) + w(i+1)*dt, 2.0d0*pi)
c        E = U + K = mgl * (1 - cos(theta)) + 0.5 * m * (w*l)^2
         e(i+1) = m*g*al*(1-cos(theta(i+1)))+ 0.5d0*m*(w(i+1)*al)**2
         write(50,*) t(i), ",", theta(i)
      end do
      close(50)
   end program
```

Verbete 3: Explicação do código da tarefa B

Escrever...

Código 4: Código-fonte da tarefa B, parte B4

```
   program TarefaB4
      implicit real*8 (a-h,o-z)
      parameter(ipassos = 2500)
      parameter(g = 9.8d0)
      parameter(al = 9.8d0)
      parameter(m = 1.0d0)

      dimension t(ipassos), w(ipassos), e(ipassos), theta(ipassos),
     &f0(3)
      character fname*50
      pi = 4.0d0*atan(1.0d0)

      theta0 = 0.14d0
      omega = (2.0d0/3.0d0)
      gamma = 0.5d0
      f0 = [0.0d0, 0.5d0, 1.2d0]
      dt=0.04d0

c     Loop k_gamma para escolher qual gamma usar
c     k_gamma = 1 => gamma = 0.05 e k_gamma = 2 => gamma = 0.5
      do k_gamma = 1, 2, 1
      if(k_gamma .eq. 1) gamma = 0.05d0
      if(k_gamma .eq. 2) gamma = 0.5d0

c     Loop j_f0 para escolher qual f0 usar
```

```
c         j_f0 = 1 => f0 = 0, j_f0 = 2 => f0 = 0.5 e j_f0 = 3 => f0=1.2
          do j_f0=1, 3, 1
          write(fname,100) j_f0, k_gamma
 100      format('saida-b4-13687303-dados_f0_', i1, '_g_', i1 ,'.csv')
          open(50, file=fname)
          write(50,101)
 101      format("t,theta,w")

          theta(1) = theta0
          w(1) = 0.0d0
          e(1) = m*g*al*(1-cos(theta0))
          t(1) = 0.0d0

          do i=1, ipassos-1, 1
             t(i+1) = i*dt
             w(i+1) = w(i) + (- (g/al)*sin(theta(i)) - gamma*w(i) + f0(j
     & f0)*sin(omega*t(i)))*dt
             theta(i+1) = mod(theta(i) + w(i+1)*dt, 2.0d0*pi)
c            E = U + K = mgl * (1 - cos(theta)) + 0.5 * m * (w*l)^2
             e(i+1) = m*g*al*(1-cos(theta(i+1)))+ 0.5d0*m*(w(i+1)*al)**2
             write(50,*) t(i), ",", theta(i), ",", w(i)
          end do
          close(50)

          end do
          end do
       end program
```

Verbete 4: Explicação do código da tarefa B

Escrever..

3.2.1 Resultados e análise

Escrever...

4 Tarefa C

Código 5: Código-fonte da tarefa C

```
       program TarefaC
          implicit real*8 (a-h,o-z)
          parameter(ipassos = 2000)
          parameter(dt = 0.04d0)
          parameter(g = 9.8d0)
          parameter(al = 9.8d0)
          parameter(m = 1.0d0)

          dimension t1(ipassos), w1(ipassos), theta1(ipassos)
          dimension t2(ipassos), w2(ipassos), theta2(ipassos)
          dimension dtheta(ipassos)
```

```
         dimension f0_arr(2), theta0_arr(2)
         character fname*50

         gamma = 0.5d0
         omega = 2.0d0/3.0d0
         f0_arr = [0.5d0, 1.2d0]
         theta0_arr = [0.14d0, 0.14d0 - 0.001d0]

         do i=1, 2, 1
            f0 = f0_arr(i)

            write(fname,100) i
 100        format("saida-c-13687303-dados_f0_", I0, ".csv")
            open(50, file=fname)
            write(50,101)
 101        format("t,dtheta")

            theta0 = theta0_arr(1)
            call simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t1,w1
     &,theta1)
            theta0 = theta0_arr(2)
            call simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t2,w2
     &,theta2)

            do j=1, ipassos, 1
               dtheta(j) = theta2(j) - theta1(j)
               write(50,*) t1(j), ",", dtheta(j)
            end do

            close(50)
         end do
      end program

      subroutine simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t,w,t
     &heta)
         implicit real*8 (a-h,o-z)
         dimension t(*), w(*), theta(*)

         pi = 4.0d0*atan(1.0d0)

         theta(1) = theta0
         w(1) = 0.0d0
         t(1) = 0.0d0
         iosc = 0

         do i=1, ipassos-1, 1
            t(i+1) = i*dt
            w(i+1) = w(i) + (- (g/al)*sin(theta(i)) - gamma*w(i) + f0*s
     &in(omega*t(i)))*dt
            theta(i+1) = mod(theta(i) + w(i+1)*dt, 2.0d0*pi)
         end do
      end subroutine
```

Verbete 5: Explicação do código da tarefa C

Escrever...

4.1 Resultados e análise

Escrever...

5 Tarefa D

Código 6: Código-fonte da tarefa D

```
      program TarefaD
         implicit real*8 (a-h,o-z)
         parameter(ipassos = 3000)
         parameter(dt = 0.04d0)
         parameter(g = 9.8d0)
         parameter(al = 9.8d0)
         parameter(m = 1.0d0)
         parameter(iangulos = 4)

         dimension t(ipassos), w(ipassos), theta(ipassos)
         dimension f0_arr(2), theta0_arr(iangulos)
         character fname*50

         gamma = 0.5d0
         omega = 2.0d0/3.0d0
         f0_arr = [0.5d0, 1.2d0]
         do j=1, iangulos, 1
            theta0_arr(j) = 0.14d0 - 1e-3*j
         end do

         do i=1, 2, 1
            f0 = f0_arr(i)

            do j=1, iangulos, 1

            write(fname,100) i, j
100         format("saida-d-13687303-dados_f0_", I0,"_theta0_", I0, ".csv")
            open(50, file=fname)
            write(50,101)
101         format("t,theta,w")
            theta0 = theta0_arr(j)
            call simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t,w
     &,theta)
            do k=1, ipassos, 1
               write(50,*) t(k), ",", theta(k), ",", w(k)
            end do
            close(50)

            end do
         end do
      end program
```

```

      subroutine simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t,w,t
     &heta)
         implicit real*8 (a-h,o-z)
         dimension t(*), w(*), theta(*)

         pi = 4.0d0*atan(1.0d0)

         theta(1) = theta0
         w(1) = 0.0d0
         t(1) = 0.0d0
         iosc = 0

         do i=1, ipassos-1, 1
            t(i+1) = i*dt
            w(i+1) = w(i) + (- (g/al)*sin(theta(i)) - gamma*w(i) + f0*s
     &in(omega*t(i)))*dt
            theta(i+1) = mod(theta(i) + w(i+1)*dt, 2.0d0*pi)
         end do
      end subroutine
```

Verbete 6: Explicação do código da tarefa D

Escrever...

5.1 Resultados e análise

Escrever...

6 Tarefa E

Código 7: Código-fonte da tarefa E

```
      program TarefaE
         implicit real*8 (a-h,o-z)
         parameter(ipassos = 100000)
         parameter(dt = 0.04d0)
         parameter(g = 9.8d0)
         parameter(al = 9.8d0)
         parameter(m = 1.0d0)
         parameter(iangulos = 5)

         dimension t(ipassos), w(ipassos), theta(ipassos)
         dimension f0_arr(2), theta0_arr(iangulos)
         character fname*60

         pi = 4.0d0*atan(1.0d0)
         gamma = 0.5d0
         omega = 2.0d0/3.0d0
         f0_arr = [0.5d0, 1.2d0]
         do j=1, iangulos, 1
```

```fortran
              theta0_arr(j) = pi/6.0d0 - 1e-3*j
           end do

           do i=1, 2, 1
              f0 = f0_arr(i)

              do j=1, iangulos, 1

              write(fname,100) i, j
 100          format("saida-e-13687303-dados_f0_", I0,"_theta0_", I0,
     &".csv")
              open(50, file=fname)
              write(50,101)
 101          format("t,theta,w")
              theta0 = theta0_arr(j)
              call simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t,w
     &,theta)
              do k=1, ipassos, 1
                 if(mod(omega*t(k), pi) .lt. dt/2.0d0) then
                    write(50,*) t(k), ",", theta(k), ",", w(k)
                 end if
              end do
              close(50)

              end do
           end do
        end program

        subroutine simulate(ipassos,dt,g,al,m,theta0,gamma,f0,omega,t,w,t
     &heta)
           implicit real*8 (a-h,o-z)
           dimension t(*), w(*), theta(*)

           pi = 4.0d0*atan(1.0d0)

           theta(1) = theta0
           w(1) = 0.0d0
           t(1) = 0.0d0
           iosc = 0

           do i=1, ipassos-1, 1
              t(i+1) = i*dt
              w(i+1) = w(i) + (- (g/al)*sin(theta(i)) - gamma*w(i) + f0*s
     &in(omega*t(i)))*dt
              theta(i+1) = mod(theta(i) + w(i+1)*dt, 2.0d0*pi)
           end do
        end subroutine
```

Verbete 7: Explicação do código da tarefa E

Escrever...

6.1 Resultados e análise