Universidade de São Paulo
Instituto de Física de São Carlos
Introdução à Física Computacional

Projeto 3: Movimento Realístico

Beatriz de Camargo Castex Ferreira - 10728077

São Carlos - 22/04/2019

Sumário:

1. INTRODUÇÃO

A compreensão de movimento é muito importante para as nossas vidas e dia a dia, pois muito do que fazemos está relacionado a uma forma ou outra de movimento. Como seres humanos temos uma grande necessidade de locomoção para realizarmos nossas tarefas diárias, porém representações clássicas de movimento que podem ser calculadas à mão nunca são representações completas de um movimento real. Existem muitos elementos e fatores envolvidos para ser possível criarmos algo propriamente realístico, mas ainda assim simples o bastante para ser calculado em poucas linhas. Felizmente, temos hoje tecnologia computacional que nos permite trabalhar com soluções mais complexas rapidamente, de forma a conseguir aproximações mais realísticas do fenômeno do movimento de forma mais eficiente. Nesse projeto iremos explorar diferentes métodos computacionais com esse propósito e daremos uma olhada na física por trás do cálculo de movimento realístico.

2. METODOLOGIA

Para compreendermos melhor o movimento realístico iremos explorar quatro problemas distintos: o primeiro envolve efeitos resistivos em movimentos unidimensionais utilizando o exemplo de um ciclista andando em linha reta; o segundo observa o efeito resistivo em movimentos bidimensionais, particularmente no lançamento de projéteis; o terceiro se mantém no tópico de movimentos bidimensionais mas desta vez propõe a exploração do movimento de um pêndulo simples; e o quarto se aprofunda no assunto do exercício anterior observando pêndulos amortecidos influenciados por uma força vertical externa.

2.1. Efeitos resistivos no movimento unidimensional

Neste exercício temos um ciclista pedalando em linha reta em um terreno plano, ele está sobre a influência de múltiplas forças resistivas tais como o atrito do solo e a resistência do ar. Podemos analisar as forças atuantes no sistema para obter uma equação que representa o movimento do ciclista e então utilizar métodos computacionais, tais como o método de Euler, para obter valores que nos ajudem a compreender tal movimento.

Consideremos nesse exercício o coeficiente de arrasto de Stokes e o arraste de Newton para o movimento do ciclista pelo fluído ar (atmosférico terrestre) e em contato com chão, desconsiderando o arraste viscoso e considerando que a área transversal do ciclista permanece constante.

Consideramos também que o ciclista possui massa de $m = 70\,kg$, que a potência P do movimento equivale a 400 W, que ambas são constantes e que o ciclista parte do repouso.

A. Movimento sem resistência do ar

Inicialmente devemos considerar o movimento do ciclista sem haver resistência do ar, para podermos compará-lo com o movimento resistivo depois. Assim procuramos descobrir a velocidade em função do tempo para o ciclista nesta situação.

Para tal podemos utilizar o método de Euler, que para o movimento resistivo se dá por:

$$v_{i+1} = v_i + \frac{P_c}{m v_i}\Delta t - \frac{\rho A v_i^2}{2m}\Delta t \quad (1)$$

Em que a primeira expressão representa a aceleração do movimento calculada em termos da potência, massa e velocidade do movimento; e a segunda expressão representa os fatores de resistência do movimento, no caso temos a densidade do fluído (ar atmosférico) ρ, a área transversal do ciclista A, a velocidade e a massa.

Podemos ver que isso representa um problema porém, pois o primeiro termo é dependente de uma divisão pela velocidade anterior, o que, caso o movimento inicie do repouso onde $v_i = 0\, m/s$ temos um resultado indefinido. Isso pode ser contornado utilizando um valor de velocidade inicial muito pequeno, de forma que seja próximo de zero mas não igual a zero.

Porém, existe outra forma de contornar esse problema que seria considerar a velocidade como $v = \sqrt{\frac{2K}{m}}$, assim podemos utilizar o método de Euler para calcular a energia cinética do movimento e a partir disso calcular a velocidade utilizando a expressão:

$$K_{1+i} = K_i + \Delta t(P_c - \sqrt{\frac{2}{m^3}\rho A K_i^{\frac{3}{2}}}) \quad (2)$$

Nesse caso como não há resistência utilizamos $\rho = 0$.

Finalmente procuramos descobrir a distância percorrida pelo ciclista. Para tal utilizamos uma das rotinas de integração trabalhadas no projeto anterior: o método do Trapézio. Isso é feito utilizando h = Δt.

No código em FORTRAN isso se apresenta como:

```
!Programa que calcula a velocidade de um ciclista pelo método de euler e
compará-la com a velocidade real

program ciclista
IMPLICIT NONE

!Variáveis
real*8 :: v    ! Velocidade que procuramos
!real*8 :: ve  ! velocidade exata
real*8 :: t    ! tempo total decorrido
real*8 :: dt   ! variação do tempo
real*8 :: m    ! massa do ciclista
real*8 :: r    ! densidade
real*8 :: A    ! área
real*8 :: P    ! Potência
real*8 :: K    ! energia cinética
real*8 :: n    ! número de iterações
real*8 :: S    ! Espaço (distância)
real*8 :: vi   ! Velocidade anterior
integer*8 :: i ! contador

! Escolhendo a velocidade inicial:
```

```
      ! A velocidade inicial não pode ser 0 pela natureza do método, portanto
escolhemos números próximos de 0
      v = 1.d0*(10.d0**(-4.d0))
      ! Porém se estivermos fazendo por energia cinética, podemos considerar v =
0
      ! como K = 1/2(mv²) => K = 1/2(m)
      !K = 0.5d0*m


      ! Descobrindo a velocidade:
      Do i=1,int(n,8)

            ! Realizar o cálculo do método de euler

            ! Por velocidade
            v = v + ((P/(m*v))*dt) - (((r*A*(v**2.d0))/2*m)*dt)

            ! Por energia cinética
            !K = K + dt*(P - (sqrt(2.d0/(m**3.d0))*r*A*(K**(1.5d0))))

            !v = sqrt((2.d0*K)/m)

            ! Valor exato
            !ve = sqrt((2.d0*(P*(dt*i)))/m)
```

Quando vamos implementar a integral pelo método de trapézio aproveitamos o loop já existente para o cálculo da velocidade e para ajustar a integral para o método multiplicamos cada passo por meio, fazendo assim a metade do passo anterior com a metade do passo atual vezes a variação do tempo, de forma que o primeiro e último termos sejam meio mas que tenhamos a valores inteiros para os outros passos.

```
            ! A distância percorrida precisa ser calculada por integração
            ! por motivos de simplicidade utilizaremos a integral de trapézio,
utilizando o próprio valor de dt como h
            S = S + 0.5d0*(dt*(v+vi))

            vi = v

            !write(80,*)(dt*i),v
            !write(20,*)(dt*i),ve
            !write(50,*)(dt*i),v


      enddo

            write(99,*)S


      end program ciclista
```

B. Movimento com resistência do ar

No próximo item passamos para um movimento com resistência do ar. Os métodos para o cálculo da velocidade em função do tempo e do espaço percorrido nesse caso não mudam muito, portanto utilizamos o mesmo código apenas trocando o valor de ρpara 1,3.

C. A influência da área

Para entendermos como a área transversal do ciclista afeta o seu desempenho final realizamos novamente o cálculo da velocidade em função do tempo utilizando três valores diferentes para A. Para isto novamente utilizamos o código do ítem A, modificando apenas os calores da área.

2.2. Efeito resistivo do ar em movimentos bidimensionais

Queremos explorar também como o efeito resistivo influência movimentos bidimensionais, tais como o lançamento de projéteis. Para isso utilizando o mesmo modelo de força resistiva anterior e adaptando o método de Euler para funcionar tanto com a velocidade horizontal quanto a velocidade vertical do movimento obtemos as expressões:

$$v_{x_{i+1}} = v_{x_i} + \frac{P_c}{m v_{x_i}} \Delta t - \frac{\rho A v_{x_i}^2}{2m} \Delta t \quad (3)$$

$$v_{y_{i+1}} = v_{y_i} + \frac{P_c}{m v_{y_i}} \Delta t - \frac{\rho A v_{y_i}^2}{2m} \Delta t \quad (4)$$

Consideramos que $v_{x_o} = v_o cos\theta$ e $v_{y_o} = v_o sen\theta$,partindo do repouso com massa $m = 42\, kg$e calibre 149,1 mm, sendo que $v_o = 377\, m/s$, $g = 9{,}8\, m/s^2, \rho = 1{,}3\, Kg/m^3$ $e\ C_d = 0{,}295$.

A. Trajetória do projétil

Precisamos montar um gráfico que represente a trajetória do projétil, para tal precisamos criar um gráfico da posição em x do projétil pelo posição em y do projétil. Utilizando as definições do movimento bidimensional e o método de Euler chegamos nas expressões:

$$x_{i+1} = x_i + v_{xi+1} \Delta t \quad (5)$$
$$y_{i+1} = y_i + v_{yi+1} \Delta t$$

(6)

Para utilizar essas expressões precisamos obter a velocidade em x e y para cada intervalo de tempo, o que podemos fazer utilizando as equações (3) e (4).

Assim, criamos o seguinte código em FORTRAN:

```
!Programa que calcula a trajetória de um projétil

program projetil
IMPLICIT NONE

!Variáveis
real*8 :: vo !velocidade inicial
real*8 :: vx !velocidade em x
real*8 :: vy !velocidade em y
```

```
      real*8 :: y !posição y
      real*8 :: x !posição x
      real*8 :: teta !ângulo da velocidade com o eixo x
      real*8 :: m !massa
      real*8 :: g !gravidade
      real*8 :: r !densidade
      real*8 :: A !área
      real*8 :: CD !drag coefficient
      real*8 :: dt !Variação de tempo
      real*8 :: aux ! auxiliar
      real*8 :: cal !calibre
      real, parameter :: pi = 3.1415927

      !Inicializando variáveis:
      vo = 377.d0 !m/s, dado
      m = 42.d0 !kg, dado
      cal = 149.1d0 !mm, dado
      g = 9.8d0 !m/s², dado
      r = 0.d0 !kg/m³, dado
      CD = 0.295d0 !dado
      dt = 0.01d0 !dado
      teta = pi/3.d0
      vx = vo*(cos(teta))
      vy = vo*(sin(teta))
      x = 0.d0
      y = 0.d0

      !Calculando a área:
      A = pi*(((cal*0.001d0)/2.d0)**2.d0) !pi*r²

      !Abrindo arquivo para impressão de resultados
      open(19,file = "projetil_6.dat")
      write(19,*)"teta: ", teta," densidade: ", r

      !Calculamos o movimento de lançamento assumindo que o objeto começa na mesma
altura que termina, ou seja, zero.

      DO WHILE(y >= 0)

      !Utilizamos o método de Euler para calcular a altura e alcance do objeto a
cada pedaço de tempo, e imprimimos a trajetória.

      aux = vx

      vx = vx - (((r*A*CD)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

      vy = vy - ((g+(((r*A*CD)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

      x = x + vx*dt
      y = y + vy*dt

      write(19,*)x,y
      enddo

      end program projetil
```

Realizamos este cálculo três vezes com diferentes valores de θpara que possamos comparar estes dados.

B. Alcance

No próximo ítem desejamos encontrar o alcance do movimento em função de θ. Para isso utilizamos os mesmos princípios do ítem anterior, porém modificamos ligeiramente o código incluindo um loop para variar o valor do ângulo ao invés de mantê-lo constante e criamos uma condição em que, se o valor da posição em x em certo momento for maior do que o valor da posição em x anterior este deve ser considerado o valor máximo de x e o ângulo que o produziu seria considerado o $\theta_{máx}$que optimiza o alcance. Assim obtemos o seguinte código:

```
!Programa que calcula o alcance de um projétil

program alcance
IMPLICIT NONE

!Variáveis
real*8 :: vo !velocidade inicial
real*8 :: vx !velocidade em x
real*8 :: vy !velocidade em y
real*8 :: y !posição y
real*8 :: x !posição x
real*8 :: xi !posição x no passo anterior
real*8 :: teta !ângulo da velocidade com o eixo x
real*8 :: teta_max !ângulo que maximiza o alcance
real*8 :: m !massa
real*8 :: g !gravidade
real*8 :: r !densidade
real*8 :: A !área
real*8 :: CD !drag coefficient
real*8 :: dt !Variação de tempo
real*8 :: cal !calibre
real*8 :: aux ! auxiliar
integer*8 :: c !contador
real, parameter :: pi = 3.1415927

!Inicializando variáveis:
vo = 377.d0 !m/s, dado
m = 42.d0 !kg, dado
cal = 149.1d0 !mm, dado
g = 9.8d0 !m/s², dado
r = 1.3d0 !kg/m³, dado
CD = 0.295d0 !dado
dt = 0.01d0 !dado
xi = 0.d0


!Calculando a área:
A = pi*(((cal*0.001d0)/2.d0)**2.d0) !pi*r²

!Abrindo arquivo para impressão de resultados
```

```
      open(20,file = "alcance.dat")

      !Precisamos calcular o alcance para vários tetas diferentes, para fazer um
gráfico.
      Do c = 0, 10000

      !mantemos teta de 0 à pi/2
      teta = real(c,8)*(pi/2.d0)*1d-4

      !precisamos saber das velocidades para encontrarmos as posições, temos o
vetor inicial vo, que separamos em componentes x e y
      vx = vo*(cos(teta))
      vy = vo*(sin(teta))
      x = 0.d0
      y = 0.d0

      !Seguimos o método de Euler para calcular o alcance e a altura para cada
teta

      DO WHILE(y >= 0)

            aux = vx

            vx = vx - (((r*A*CD)/(2.d0*m))*sqrt((aux**2.d0)+(vy**2.d0))*aux*dt)

            vy                    =                    vy                    -
((g+(((r*A*CD)/(2.d0*m))*sqrt((aux**2.d0)+(vy**2.d0))*vy))*dt)

            x = x + vx*dt
            y = y + vy*dt

            !Vendo se cada alcance é maior do que o anterior, nós descobrimos o
ângulo de alcance máximo.

            IF (x .GE. xi) THEN
                  xi = x
                  teta_max = teta
            endif

      enddo

      write(20,*)teta,x
      enddo
      write(20,*)teta_max

      end program alcance
```

C. Variação na resistência

Por observações do mundo a nossa volta podemos perceber que na realidade existem outros fatores influenciando a resistência do movimento de projéteis. Sabemos que de acordo com a altitude a densidade da atmosfera muda e que a velocidade do objeto influência na resistência.

Portanto se quisermos fazer aproximações mais exatas do movimento de lançamento de projéteis precisamos considerar isso, Como dado no exercício podemos considerar que a densidade em relação a altitude é:

$$\rho = \rho_o(1 - \frac{\beta}{T_o}y)^{\alpha}$$

(7)

onde temos a taxa de variação de temperatura de acordo com a altitude $\beta = 6{,}49 * 10^{-3}\ K/m$, a temperatura a nível do mar na Terra $T_o = 300\ K$e o expoente $\alpha = 4{,}256$ para o ar atmosférico terrestre.

Quanto a mudança do coeficiente de arrasto de acordo com a velocidade do projétil iremos utilizar uma simplificação dada no exercício em que o coeficiente de arrasto é $C_d = 0{,}295$ para uma velocidade menor do que a velocidade do som e 0,5 caso contrário.

Queremos então calcular novamente os valores para a trajetória dos projéteis e teta máximo utilizando essa nova aproximação, assim ajustamos os códigos anteriores, obtendo:

Para a trajetória do projétil:

```
      !Programa que calcula a trajetória de um projétil considerando resistência
do ar, altura e velocidade.

      program projétil resistente
      IMPLICIT NONE

      !Variáveis
      real*8 :: vo !velocidade inicial
      real*8 :: vx !velocidade em x
      real*8 :: vy !velocidade em y
      real*8 :: y !posição y
      real*8 :: x !posição x
      real*8 :: teta !ângulo da velocidade com o eixo x
      real*8 :: m !massa
      real*8 :: g !gravidade
      real*8 :: r !densidade
      real*8 :: ro !densidade inicial
      real*8 :: A !área
      real*8 :: CD menor !drag coefficient para velocidades abaixo da velocidade
do som
      real*8 :: CD_ maior !drag coefficient para velocidades acima da velocidade
do som
      real*8 :: dt !Variação de tempo
      real*8 :: aux_x !auxiliar para x
      real*8 :: aux y !auxiliar para y
      real*8 :: aux
      real*8 :: v_ som !velocidade do som
      real*8 :: beta !taxa de variação da temperatura com a altitude
      real*8 :: alpha !expoente para o ar
      real*8 :: To !temperatura a nível do mar
      real*8 :: cal !calibre
      real, parameter :: pi = 3.1415927

      !Inicializando variáveis:
```

```
      vo = 377.d0 !m/s, dado
      m = 42.d0 !kg, dado
      cal = 149.1d0 !mm, dado
      g = 9.8d0 !m/s², dado
      CD_menor = 0.295d0 !dado
      CD_maior = 0.5d0 !dado
      v_som = 343.d0 !dado
      alpha = 4.256d0 !dado
      beta = 6.43d-3 !dado
      To = 300 !K, dado
      ro = 0.d0 !kg/m³, dado
      dt = 0.01d0 !dado
      teta = pi/3.d0
      vx = vo*(cos(teta))
      vy = vo*(sin(teta))
      x = 0.d0
      y = 0.d0

      !Calculando a área:
      A = pi*(((cal*0.001d0)/2.d0)**2.d0) !pi*raio²

      !Abrindo arquivo para impressão de resultados
      open(31,file = "projetil_resistente_1.dat")
      write(31,*)"teta: ", teta," densidade: ", ro

      !Calculamos o movimento de lançamento assumindo que o objeto começa na mesma
altura que termina, ou seja, zero.

      DO WHILE(y >= 0)

      !Utilizamos o método de Euler para calcular a altura e alcance do objeto a
cada pedaço de tempo, e imprimimos a trajetória.

      aux x = x
      aux_y = y

      x = aux_x + vx*dt
      y = aux_y + vy*dt

      !Mudamos a densidade conforme a altitude
      r = ro*((1 - (y*(beta/To)))**alpha)

      aux = vx

      !Verificamos se a velocidade em x ou y está acima da velocidade do som e
usamos o coef. de arrasto adequado

      IF ((vx < v_som) .AND. (vy < v_som)) THEN

            vx = vx - (((r*A*CD_menor)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

            vy = vy - ((g+(((r*A*CD_menor)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)
      ELSE IF ((vx > v_som) .AND. (vy < v_som)) THEN

            vx = vx - (((r*A*CD_maior)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)
```

```
            vy = vy - ((g+(((r*A*CD_menor)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)
      ELSE IF ((vy > v_som) .AND. (vx < v_som)) THEN

            vx = vx - (((r*A*CD_menor)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

            vy = vy - ((g+(((r*A*CD_maior)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

      ELSE

            vx = vx - (((r*A*CD_maior)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

            vy = vy - ((g+(((r*A*CD_maior)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

      endif



      write(31,*)x,y
      enddo

      end program projétil resistente
```

Para o alcance e ângulo máximo:

```
      !Programa que calcula o alcance de um projétil considerando resistência do
ar, altitude e velocidade.

      program alcance_res
      IMPLICIT NONE

      !Variáveis
      real*8 :: vo !velocidade inicial
      real*8 :: vx !velocidade em x
      real*8 :: vy !velocidade em y
      real*8 :: y !posição y
      real*8 :: x !posição x
      real*8 :: xi !posição x no passo anterior
      real*8 :: teta !ângulo da velocidade com o eixo x
      real*8 :: teta_max !ângulo que maximiza o alcance
      real*8 :: m !massa
      real*8 :: g !gravidade
      real*8 :: r !densidade
      real*8 :: ro ! densidade inicial
      real*8 :: A !área
      real*8 :: CD_menor !drag coefficient para velocidades abaixo da velocidade
do som
      real*8 :: CD_maior !drag coefficient para velocidades acima da velocidade
do som
      real*8 :: dt !Variação de tempo
      real*8 :: cal !calibre
      real*8 :: aux_x !auxiliar para x
      real*8 :: aux_y !auxiliar para y
      real*8 :: aux
```

```
      real*8 :: v_som !velocidade do som
      real*8 :: beta !taxa de variação da temperatura com a altitude
      real*8 :: alpha !expoente para o ar
      real*8 :: To !temperatura a nível do mar
      integer*8 :: c !contador
      real, parameter :: pi = 3.1415927


      !Inicializando variáveis:
      vo = 377.d0 !m/s, dado
      m = 42.d0 !kg, dado
      cal = 149.1d0 !mm, dado
      g = 9.8d0 !m/s², dado
      ro = 1.3d0 !kg/m³, dado
      CD_menor = 0.295d0 !dado
      CD_maior = 0.5d0 !dado
      v_som = 343.d0 !dado
      alpha = 4.256d0 !dado
      beta = 6.43d-3 !dado
      To = 300 !K, dado
      dt = 0.01d0 !dado
      xi = 0.d0

      !Calculando a área:
      A = pi*(((cal*0.001d0)/2.d0)**2.d0) !pi*r²

      !Abrindo arquivo para impressão de resultados
      open(22,file = "alcance resistente.dat")

      !Precisamos calcular o alcance para vários tetas diferentes, para fazer um
gráfico.
      Do c = 0, 10000

      !mantemos teta de 0 à pi/2
      teta = real(c,8)*(pi/2.d0)*1d-4

      !precisamos saber das velocidades para encontrarmos as posições, temos o
vetor inicial vo, que separamos em componentes x e y
      vx = vo*(cos(teta))
      vy = vo*(sin(teta))
      x = 0.d0
      y = 0.d0

      ! Calculamos o movimento de lançamento assumindo que o objeto começa na
mesma altura que termina, ou seja, zero.

      DO WHILE(y >= 0)

            !Seguimos o método de Euler para calcular o alcance e a altura para
cada teta
            aux_x = x
            aux_y = y

            x = aux_x + vx*dt
            y = aux_y + vy*dt
```

```
            !Mudamos a densidade conforme a altitude
            r = ro*((1 - (y*(beta/To)))**alpha)

            aux = vx

            !Verificamos se a velocidade em x ou y está acima da velocidade do
som e usamos o coef. de arrasto adequado
            IF ((vx < v_som) .AND. (vy < v_som)) THEN

                  vx = vx - (((r*A*CD_menor)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

                  vy                  =                  vy                  -
((g+(((r*A*CD_menor)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)
            ELSE IF ((vx > v_som) .AND. (vy < v_som)) THEN

                  vx = vx - (((r*A*CD_maior)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

                  vy                  =                  vy                  -
((g+(((r*A*CD_menor)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)
            ELSE IF ((vy > v_som) .AND. (vx < v_som)) THEN

                  vx = vx - (((r*A*CD_menor)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

                  vy                  =                  vy                  -
((g+(((r*A*CD_maior)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

            ELSE

                  vx = vx - (((r*A*CD_maior)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

                  vy                  =                  vy                  -
((g+(((r*A*CD_maior)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

            endif

            !Vendo se cada alcance é maior do que o anterior, nós descobrimos o
angulo de alcance máximo.
            IF (x .GE. xi) THEN
                  xi = x
                  teta_max = teta
            endif

      enddo

      write(22,*)teta,x
      enddo
      write(22,*)teta_max

      end program alcance_res
```

D. Velocidade de lançamento

Neste item devemos utilizar o ângulo máximo e variar a velocidade de lançamento em $\pm 1\%$ e obter o alcance A para cada um. Para isso pegamos o primeiro programa do ítem anterior, mudamos o teta inicial para ser igual a $\theta_{máx}$ e calculamos a variação na velocidade, rodando o programa duas vezes para calcular os novos alcances.

E. Velocidade em função do tempo

Para este item devemos calcular a velocidade em função do tempo, para isso utilizamos o código do ítem C. e modificamos levemente, incluindo o seguinte termo:

$$v = \sqrt{v_x^2 + v_y^2} \tag{8}$$

Obtendo o seguinte código (este código é utilizado para ambos itens D. e E.):

```
!Programa que calcula a velocidade de um projétil durante sua trajetória

program projetil_veloz
IMPLICIT NONE

!Variáveis
real*8 :: vo !velocidade inicial
real*8 :: vx !velocidade em x
real*8 :: vy !velocidade em y
real*8 :: y !posição y
real*8 :: x !posição x
real*8 :: teta !angulo da velocidade com o eixo x
real*8 :: m !massa
real*8 :: g !gravidade
real*8 :: r !densidade
real*8 :: ro !densidade inicial
real*8 :: A !área
real*8 :: CD_menor !drag coefficient para velocidades abaixo da velocidade
do som
real*8 :: CD_maior !drag coefficient para velocidades acima da velocidade
do som
real*8 :: dt !Variação de tempo
real*8 :: aux_x !auxiliar para x
real*8 :: aux_y !auxiliar para y
real*8 :: aux
real*8 :: v_som !velocidade do som
real*8 :: beta !taxa de variação da temperatura com a altitude
real*8 :: alpha !expoente para o ar
real*8 :: To !temperatura a nível do mar
real*8 :: cal !calibre
real*8 :: v !velocidade
real*8 :: t !tempo
real, parameter :: pi = 3.1415927

!Inicalizando variáveis:
vo = 373.23d0 !m/s, dado
m = 42.d0 !kg, dado
cal = 149.1d0 !mm, dado
g = 9.8d0 !m/s², dado
CD_menor = 0.295d0 !dado
```

```
      CD_maior = 0.5d0 !dado
      v_som = 343.d0 !dado
      alpha = 4.256d0 !dado
      beta = 6.43d-3 !dado
      To = 300 !K, dado
      ro = 0.d0 !kg/m³, dado
      dt = 0.01d0 !dado
      teta = 0.74267252397537231d0 !obtido com o programa alcance_res
      vx = vo*(cos(teta))
      vy = vo*(sin(teta))
      x = 0.d0
      y = 0.d0
      t = 0.d0
      v = vo

      !Calculando a área:
      A = pi*(((cal*0.001d0)/2.d0)**2.d0) !pi*raio²

      !Abrindo arquivo para impressão de sesultados
      !Aonde escreveremos os alcances
      open(36,file = "projetil_veloz_2.dat")
      write(36,*)"teta: ", teta," velocidade: ", vo
      !aonde escreveremos dados para gráficos
      open(37,file = "projetil_veloz_2_graph.dat")


      !Calculamos o movimento de lançamento assumindo que o objeto começa na mesma
altura que termina, ou seja, zero.

      DO WHILE(y >= 0)

      !Utilizamos o método de Euler para calcular a altura e alcance do objeto a
cada pedaço de tempo, e imprimimos a trajetória.

      aux_x = x
      aux_y = y

      x = aux_x + vx*dt
      y = aux_y + vy*dt

      !Mudamos a densidade conforme a altitude
      r = ro*((1 - (y*(beta/To)))**alpha)

      aux = vx

      !Verificamos se a velocidade em x ou y está acima da velocidade do som e
usamos o coef. de arrasto adequado

      IF ((vx < v_som) .AND. (vy < v_som)) THEN

            vx = vx - (((r*A*CD_menor)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

            vy = vy - ((g+(((r*A*CD_menor)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)
      ELSE IF ((vx > v_som) .AND. (vy < v_som)) THEN
```

```
      vx = vx - (((r*A*CD_maior)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

      vy = vy - ((g+(((r*A*CD_menor)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)
ELSE IF ((vy > v_som) .AND. (vx < v_som)) THEN

      vx = vx - (((r*A*CD_menor)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

      vy = vy - ((g+(((r*A*CD_maior)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

ELSE

      vx = vx - (((r*A*CD_maior)/(2*m))*sqrt((vx**2)+(vy**2))*vx*dt)

      vy = vy - ((g+(((r*A*CD_maior)/(2*m))*sqrt((aux**2)+(vy**2))*vy))*dt)

endif

v = sqrt((vx*vx)+(vy*vy))
t = t + dt

write(37,*)t,v



enddo

write(36,*)x


end program projetil_veloz
```

2.3. Pêndulo Simples

Pêndulos são particularmente úteis e utilizados há muito tempo para, por exemplo, contar o tempo dentro de relógios. Para isso é muito importante saber calcular o período de um pêndulo. Quando o ângulo inicial de um pêndulo é muito pequeno temos que o período $T = 2\pi\sqrt{\frac{l}{g}}$ porém quando o ângulo é maior a solução é mais complicada.

Nesse exercício exploraremos métodos computacionais para avaliar o movimento pendular simples. Consideraremos um pêndulo com $m = 1\,Kg, l = 1\,m, g = 9{,}8\,e\,T_o = 2{,}00709\,s$.

A. Ângulo e energia mecânica

Utilizando o método de Euler e as equações dadas no exercício representando o movimento pendular:

$$w_{i+1} = w_i - \frac{g}{l} sen\theta_i \Delta t \quad (9)$$

$$\theta_{i+1} = \theta_i + w_i \Delta t \quad (10)$$

devemos calcular o ângulo de pêndulo em função do tempo e a energia mecânica em função do tempo.

Para calcularmos a energia mecânica do pêndulo podemos considerar que a energia mecânica é igual a energia potencial mais a energia cinética, ambas podendo ser calculadas pelos parâmetros que conhecemos. Para a energia potencial consideramos a altura da massa no pêndulo como $h = L(1 - cos\theta)$. Assim a energia mecânica é:

$$Em = mgL(1 - cos\theta) - \frac{m(wl)^2}{2} \qquad (11)$$

Ao implementar essas equações obtemos o seguinte código em FORTRAN:

```
!Programa que calcula a o ângulo e energia mecânica de um pêndulo pelo método de Euler

program pendulo_a
IMPLICIT NONE

! Variáveis
real*8 :: m ! massa
real*8 :: l ! comprimento do fio
real*8 :: teta_max ! maior ângulo que o pendulo pode ter
real*8 :: teta ! angulo do pendulo
real*8 :: t ! tempo
real*8 :: dt ! variação do tempo
real*8 :: w ! velocidade angular
real*8 :: Em ! Energia mecânica
real*8 :: Po ! Período inicial
real*8 :: aux
real, parameter :: g = 9.8d0
real, parameter :: pi = 3.1415927


! Inicializando variáveis
m = 1.d0 !kg, dado
l = 1.d0 !m, dado
teta_max = pi/2.d0 ! dado
Po = 2.00709 ! s, dado
dt = 0.005d0 !s, dado
w = 0.d0
t = 0.d0
teta = teta_max

! Abrindo arquivos para imprimir resultados
! Arquivo para energia mecânica
open(42,file = "pendulo_Em_a.dat")
! Arquivo para ângulo
open(46,file = "pendulo_teta_a.dat")




! Devemos calcular o teta e a energia mecânica em um intervalo de tempo 0 <= t <= 40 s
```

```
DO WHILE (t <= 40)


! Usamos o método de Euler para calcular o ângulo e a velocidade angular
aux = teta
teta = teta + (w * dt)
w = w - (sin(aux)*dt*(g/l))


! A energia mecanica é a energia potencial mais a energia cinética
Em = (m*g*(L*(1 - cos(teta)))) + (0.5d0*m*((w*l)**2.d0))

t = t + dt

write(42,*)t,Em
write(46,*)t,teta


enddo




end program pendulo_a
```

B. Espaço de fase

Neste item devemos calcular o espaço de fase do movimento do item anterior e compará-lo com o resultado exato. Para o espaço de fase aproximado utilizamos as equações (9) e (10) pelo método de Euler e imprimimos o ângulo e a velocidade angular para cada tempo.

Já para o resultado exato podemos utilizar o princípio de conservação de energia para obter a velocidade angular para cada teta. Sabemos que em um pêndulo deveria haver conservação da energia mecânica, assim podemos calcular a energia mecânica inicial, a energia potencial para cada ponto do movimento e fazer $Ek = Em - Ep$ para descobrir a velocidade angular de forma que:

$$w = \frac{\sqrt{\frac{2Ek}{m}}}{l} \qquad (12)$$

Assim obtemos o código:

```
! Programa que calcula o angulo e velocidade de um pêndulo pelo método de
euler

program pendulo_graph_a
IMPLICIT NONE

! Variáveis
real*8 :: m ! massa
real*8 :: l ! comprimento do fio
real*8 :: teta_max ! maior ângulo que o pendulo pode ter
real*8 :: teta ! angulo do pendulo
```

```
      real*8 :: t ! tempo
      real*8 :: dt ! variação do tempo
      real*8 :: w ! velocidade angulas
      real*8 :: Po ! Período inicial
      real*8 :: Em !Energia mecanica
      real*8 :: Ep ! Energia potencial
      real*8 :: Ek ! Energia cinética
      real*8 :: w_exato
      real*8 :: aux
      real, parameter :: g = 9.8d0
      real, parameter :: pi = 3.1415927


      ! Inicializando variáveis
      m = 1.d0 !kg, dado
      l = 1.d0 !m, dado
      teta_max = pi/2.d0 ! dado
      Po = 2.00709 ! s, dado
      dt = 0.005d0 !s, dado
      w = 0.d0
      t = 0.d0
      teta = pi/2

      ! Abrindo arquivos para imprimir resultados
      ! Arquivo para valores aproximados
      open(56,file = "pendulo_graph_a.dat")
      ! Arquivo para valores exatos
      open(84,file = "pendulo_graph_a_exato.dat")



      DO WHILE (t <= 40)


      ! Usamos o método de Euler para calcular o ângulo e a velocidade angular
      aux = teta
      teta = teta + (w * dt)
      w = w - (sin(aux)*dt*(g/l))


      ! Valor exato
      ! Para o valor exato da velocidade primeiro calculamos a energia mecânica,
que se conserva:
      Em = m*g*l

      ! Calculamos então a energia potêncial no tempo t
      Ep = m*g*(l*(1-cos(teta)))

      ! Calculamos a energia cinética
      Ek = Em - Ep

      !Pela energia cinética calculamos a velocidade angular
      w_exato = sqrt((2*Ek)/m)/l

      t = t + dt
```

```
write(56,*)teta,abs(w)
write(84,*)teta,w_exato

!teta = teta - dteta



enddo



end program pendulo_graph_a
```

C. Método de Euler-Cromer

Neste item devemos calcular os valores dos itens anteriores novamente porém utilizando o método de Euler-Cromer, definido pelas equações:

$$w_{i+1} = w_i - \frac{g}{l} sen\theta_i \Delta t \quad (13)$$

$$\theta_{i+1} = \theta_i + w_{i+1} \Delta t \quad (14)$$

Assim a implementação foi basicamente a mesma do ítem anterior, apenas com a mudança de algumas linhas trocando o método de Euler pelo método de Euler-Cromer:\\

```
! Usamos o método de Euler-Cromer para calcular o ângulo e a velocidade angular

w = w - (sin(teta)*dt*(g/l))
teta = teta + (w * dt)
```

D. Período do pêndulo

Finalmente procuramos medir o período do pêndulo em relação a teta, o que é particularmente complicado pois envolve o cálculo de uma integral elíptica:

$$T = 4\sqrt{\frac{l}{g}} \int_0^{\frac{\pi}{2}} \frac{du}{\sqrt{1 - k^2 sen^2 u}} \quad (15)$$

em que:

$$k = sen\frac{\theta máx}{2} \quad (16)$$

Uma possível aproximação mais simples foi apresentada por Kidd & Fogg(2001, p.82)[8] em que:

$$T \approx 2\pi\sqrt{l/g cos(\alpha/2)} \text{ para } \alpha \leq \frac{\pi}{2} \quad (17)$$

onde αé o ângulo para o qual o período está sendo calculado.

Assim calculamos o período em função de teta através do seguinte código:

```
      ! Programa que calcula o período de um pendulo em função de teta-max pelo
metodo de euler-cromer

      program periodo
      IMPLICIT NONE

      ! Variáveis
      real*8 :: m ! massa
      real*8 :: l ! comprimento do fio
      real*8 :: teta_max ! maior ângulo que o pendulo pode ter
      real*8 :: teta ! angulo do pendulo
      real*8 :: t ! tempo
      real*8 :: dt ! variação do tempo
      real*8 :: w ! velocidade angulas
      real*8 :: Em ! Energia mecânica
      real*8 :: Po ! Período inicial
      real*8 :: P ! Período
      real, parameter :: g = 9.8d0
      real, parameter :: pi = 3.1415927


      ! Inicializando variáveis
      m = 1.d0 !kg, dado
      l = 1.d0 !m, dado
      teta_max = pi/2.d0 ! dado
      Po = 2.00709 ! s, dado
      dt = 0.005d0 !s, dado
      w = 0.d0
      t = 0.d0
      teta = teta_max

      ! Abrindo arquivos para imprimir resultados
      ! Arquivo para energia mecânica
      open(92,file = "periodo.dat")




      ! Devemos calcular o teta em um intervalo de tempo 0 <= t <= 40 s
      DO WHILE (t <= 40)


      ! Usamos o método de Euler-Cromer para calcular o ângulo e a velocidade
angular

      w = w - (sin(teta)*dt*(g/l))
      teta = teta + (w * dt)


      ! o período pode ser dado através de uma aproximação encontrada em:
      !
http://users.df.uba.ar/sgil/physics_paper_doc/papers_phys/mechan/Pendulo2.pdf
      P = 2.d0*pi*sqrt((l/g)*cos(teta/2.d0))
```

```
t = t + dt

write(92,*)teta,P


enddo

end program periodo
```

Já para calcular o valor exato do período utilizamos a integral de trapézio para solucionar a equação (15):

```
! Código que calcula o período exato de um pêndulo para qualquer teta

program periodo_exato
IMPLICIT NONE

! Variáveis
real*8 :: m ! massa
real*8 :: l ! comprimento do fio
real*8 :: teta_max ! maior ângulo que o pendulo pode ter
real*8 :: teta ! angulo do pendulo
real*8 :: t ! tempo
real*8 :: dt ! variação do tempo
real*8 :: w ! velocidade angulas
real*8 :: Em ! Energia mecânica
real*8 :: Po ! Período inicial
real*8 :: P ! Período
real*8 :: I ! integral
real*8 :: tetai
real, parameter :: g = 9.8d0
real, parameter :: pi = 3.1415927
real*8 :: h


! Inicializando variáveis
m = 1.d0 !kg, dado
l = 1.d0 !m, dado
teta_max = pi/2.d0 ! dado
Po = 2.00709 ! s, dado
dt = 0.005d0 !s, dado
w = 0.d0
t = 0.d0
teta = teta_max
I = Po
tetai = 0.d0

! Abrindo arquivos para imprimir resultados
! Arquivo para energia mecânica
open(95,file = "periodo_exato.dat")




! Devemos calcular o teta em um intervalo de tempo 0 <= t <= 40 s
```

```
      DO WHILE (t <= 40)


      ! Usamos o método de Euler-Cromer para calcular o ângulo e a velocidade
angular

      w = w - (sin(teta)*dt*(g/l))
      teta = teta + (w * dt)


      ! Usamos a integral do trapézio para calcular o período exato
      I = I + 0.5*(f(teta+tetai)*dt)
      P = 4sqrt(l/g)*I
      tetai = teta
      t = t + dt

      write(95,*)teta,P


      enddo

      end program periodo_exato

      real*8 function f(p)
      IMPLICIT NONE

      !Variáveis:
      real*8, intent(in)  :: p !Variável de entrada

      f = 1/sqrt(1 - ((sin((pi/2)/2)**2)*((sen(p))**2)))

      RETURN
      end
```

E. Método de Verlet

Um outro método que podemos utilizar para encontrar teta é o método de Verlet que se dá por:

$$\theta_{i+1} = 2\theta_{i} - \theta_{i-1} - \frac{g}{l}(sen\theta_{i})\Delta t^{2} \qquad (18)$$

Assim reimplementamos os códigos nos itens C. e D. trocando o método de Euler-Cromer pelo método de Verlet

```
      ! Usamos o método de verlet para calcular o ângulo
      tetai_1 = tetai
      tetai = teta
      teta = ((2.d0*tetai)-tetai_1)-((g/l)*(sin(tetai)*(dt*dt)))
```

2.4. Estabilização Dinâmica

Na vida real pêndulos também estão geralmente sob influência de alguma força resistiva. Neste problema iremos tentar considerar um pêndulo mais realístico. Também iremos incluir uma força na vertical, deslocando o pêndulo.

A. Movimento

Devemos graficar o valor do ângulo de acordo com o tempo, para isso devemos utilizar todos os três métodos apresentados. A equação do movimento pendular na situação citada é:

$$\frac{d^2\theta}{dt} = -bw - (1 - \alpha v sen(v\tau))sen\theta \qquad (19)$$

Criamos então o código:

Para Euler:

```
! Programa que calcula o ângulo em função do tempo de um pêndulo amortecido e influênciado por uma forç externa na vertival pelo método de Euler

program pendulo_amortecido_euler
IMPLICIT NONE

! Variáveis:
real*8 :: alpha ! amplitude do movimento
real*8 :: beta  ! coef. de Stokes
real*8 :: t ! tempo adimensional
real*8 :: dt ! variação do tempo
real*8 :: freq ! frequencia do movimento vertical
real*8 :: w ! velocidade angular
real*8 :: teta ! angulo
real*8 :: a ! aceleração
real*8 :: aux
real, parameter :: g = 9.8d0
real, parameter :: pi = 3.1415927

! Inicializando variáveis:
freq = 0.d0
alpha = 0.1d0
beta = 0.05d0
dt = 0.01d0
teta = 0.9999d0*pi
w = 0.d0

! Abrindo arquivos para imprimir resultados
! Arquivo para quando dt = 0.01
open(22,file = "pendulo_amortecido_euler_01.dat")
!Arquivo para quando dt = 0.0001
!open(23,file = "pendulo_amortecido_euler_0001.dat")

! Devemos calcular o teta em um certo intervalo de tempo
DO WHILE (t <= 30.d0*2.d0*pi)

! Vamos utilizar o método de Euler para calcular o movimento
! Começamos com uma aproximação da aceleração
```

```
      a = - (beta * w) - ((1.d0 - alpha * (freq * freq) * sin(freq * t)) *
sin(teta))
      teta = teta + (w * dt)
      w = w + (dt * a)

      write(22,*)t,teta

      t = t+dt

      enddo

      end program pendulo_amortecido_euler
```

Para Euler-Cromer:

```
      ! Devemos calcular o teta em um certo intervalo de tempo
      DO WHILE (t <= 30.d0*2.d0*pi)

      ! Vamos utilizar o método de Euler para calcular o movimento
      ! Começamos com uma aproximação da aceleração
      a = - (beta * w) - ((1.d0 - alpha * (freq * freq) * sin(freq * t)) *
sin(teta))
      w = w + (dt * a)
      teta = teta + (w * dt)

      write(32,*)t,teta

      t = t+dt

      enddo
```

Para Verlet:

```
      ! Devemos calcular o teta em um certo intervalo de tempo
      DO WHILE (t <= 30.d0*2.d0*pi)

      ! Vamos utilizar o método de Verlet para calcular o movimento
      ! Começamos com uma aproximação da aceleração
      tetai_1 = tetai
      tetai = teta
      a = -(beta * w) - ((1.d0 - (alpha * (freq * freq) * sin(freq * t))) *
sin(tetai))
      teta = ((2.d0*tetai)-tetai_1)+(a * (dt * dt))
      w = (tetai - tetai_1)/dt

      write(52,*)t,teta

      t = t+dt

      enddo
```

B. Frequência

Variando a frequência da força vertical atuante no pêndulo devemos encontrar o menor teta para cada frequência, fazemos isso a partir de:

e calcula o menor teta em função da frequência de um pêndulo amortecido e influênciado por uma força externa na vertical pelo método de Verlet

```
      program pendulo_amortecido_frequencia
      IMPLICIT NONE

      ! Variáveis:
      real*8 :: alpha ! amplitude do movimento
      real*8 :: beta  ! coef. de Stokes
      real*8 :: t ! tempo adimensional
      real*8 :: dt ! variação do tempo
      integer*8 :: freq ! frequencia do movimento vertical
      real*8 :: w ! velocidade angular
      real*8 :: teta ! angulo
      real*8 :: a ! aceleração
      real*8 :: teta_min
      real*8 :: tetai_1
      real*8 :: tetai
      real*8 :: aux
      real, parameter :: g = 9.8d0
      real, parameter :: pi = 3.1415927

      ! Inicializando variáveis:
      alpha = 0.1d0
      beta = 0.05d0
      dt = 0.0001d0

      ! Abrindo arquivos para imprimir resultados
      open(87,file = "pendulo_amortecido_frequencia.dat")


      !Para o ângulo e velocidade angular teta1 e w1 precisamos usar o método de
Euler:
      tetai = teta

      DO freq = 0,20

      teta = 0.9999d0*pi
      w = 0.d0


      !Para o ângulo e velocidade angular teta1 e w1 precisamos usar o método de
Euler:
      tetai = teta

      ! Devemos calcular o teta em um certo intervalo de tempo
      DO WHILE (t <= 30.d0*2.d0*pi)

            ! Vamos utilizar o método de Verlet para calcular o movimento
            ! Começamos com uma aproximação da aceleração
            tetai_1 = tetai
```

```
            tetai = teta
            a = -(beta * w) - ((1.d0 - (alpha * (freq * freq) * sin(freq * t))) *
sin(tetai))
            teta = ((2.d0*tetai)-tetai_1)+(a * (dt * dt))
            w = (tetai - tetai_1)/dt

            IF (tetai > teta) THEN
                  teta_min = teta
            endif

            t = t+dt

      enddo

      write(87,*)freq,teta_min

      enddo

      end program pendulo_amortecido_frequencia
```

C. Frequência de estabilização dinâmica.

Devemos achar a frequência que causa estabilização dinaminca para cada ângulo inicial. Assim temos:

! Programa que calcula o menor teta em função da frequência para diferentes tetas iniciais de um pêndulo amortecido e influenciado por uma força externa na vertical pelo método de Verlet

```
      program pendulo_amortecido_frequencia_2
      IMPLICIT NONE

      ! Variáveis:
      real*8 :: alpha ! amplitude do movimento
      real*8 :: beta  ! coef. de Stokes
      real*8 :: t ! tempo adimensional
      real*8 :: dt ! variação do tempo
      integer*8 :: freq ! frequencia do movimento vertical
      real*8 :: w ! velocidade angular
      real*8 :: teta ! angulo
      real*8 :: teta0 ! angulo inicial
      real*8 :: dteta ! variação do ângulo inicial
      real*8 :: a ! aceleração
      real*8 :: teta_min
      real*8 :: tetai_1
      real*8 :: tetai
      real*8 :: aux
      real, parameter :: g = 9.8d0
      real, parameter :: pi = 3.1415927

      ! Inicializando variáveis:
      alpha = 0.1d0
      beta = 0.d0
      dt = 0.0001d0
```

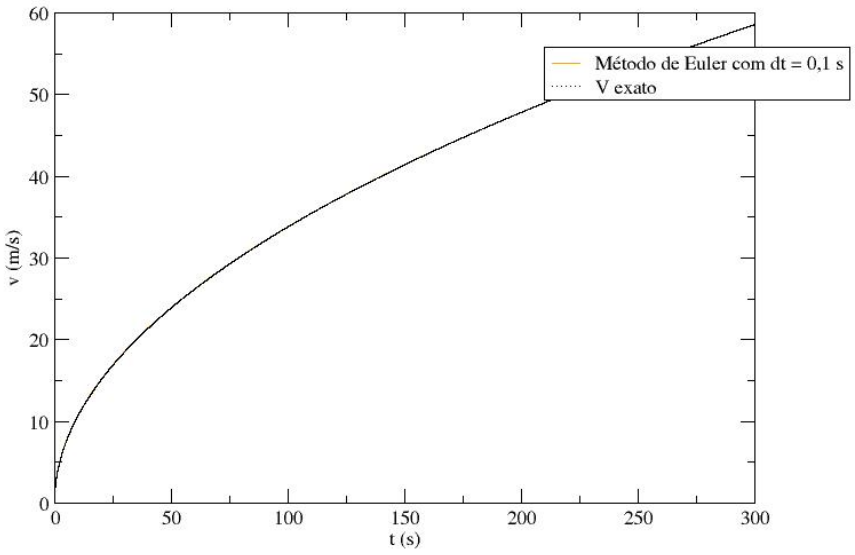
```
      dteta = 0.1d0
      teta0 = 0.9999d0*pi
      w = 0.d0

      ! Abrindo arquivos para imprimir resultados
      open(89,file = "pendulo_amortecido_frequencia_2.dat")


      DO WHILE (teta0 > 0.1)

      DO freq = 0,20

            !Para o ângulo e velocidade angular teta1 e w1 precisamos usar o
método de Euler:
            tetai = teta0

            ! Devemos calcular o teta em um certo intervalo de tempo
            DO WHILE (t <= 30.d0*2.d0*pi)

                  ! Vamos utilizar o método de Euler para calcular o movimento
                  ! Começamos com uma aproximação da aceleração
                  tetai_1 = tetai
                  a = -(beta * w) - ((1.d0 - (alpha * (freq * freq) * sin(freq *
t))) * sin(tetai))
                  teta = ((2.d0*tetai)-tetai_1)+(a * (dt * dt))
                  tetai = teta
                  w = (tetai - tetai_1)/dt

                  IF (tetai > teta) THEN
                        teta_min = teta
                  endif

                  t = t+dt

            enddo

            IF ( teta_min != 0.d0 ) THEN
                  write(89,*)teta0,freq
            endif

      enddo

      teta0 = teta0+dteta

      enddo

      end program pendulo_amortecido_frequencia_2
```

3. RESULTADOS

3.1. Efeitos resistivos no movimento unidimensional

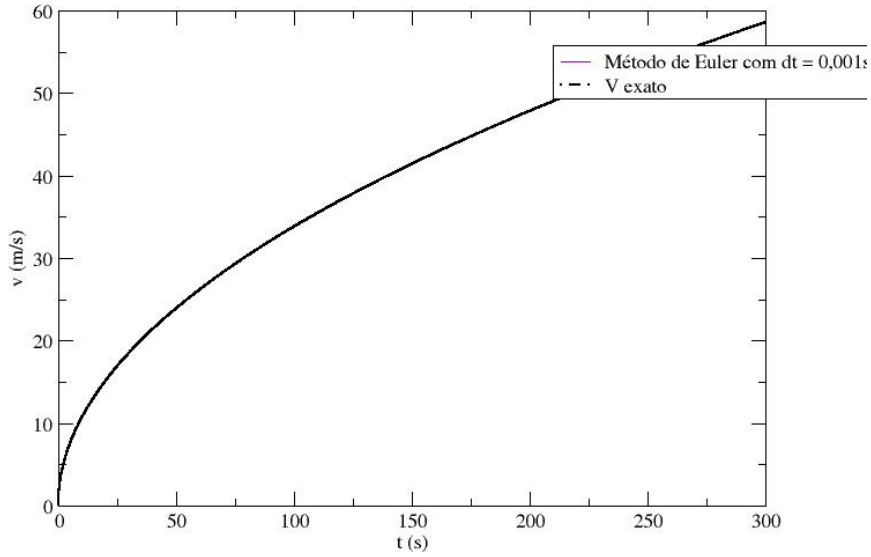
A. Movimento sem resistência do ar
Ao rodar o programa obtemos os seguintes gráficos:

Imagem 1 - Gráfico da velocidade em função do tempo sem resistência para $\Delta t = 0{,}1\ s$



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Imagem 2 - Gráfico da velocidade em função do tempo sem resistência para $\Delta t = 0{,}001\ s$



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Como podemos ver ambos resultados são consideravelmente similares e se encontram próximos o bastante do valor exato da velocidade para serem considerados formas acuradas de calculá-la.

Vemos também que a curva no gráfico segue um formato comum a funções com raízes e portanto não vemos um limite superior para a velocidade nesse caso.

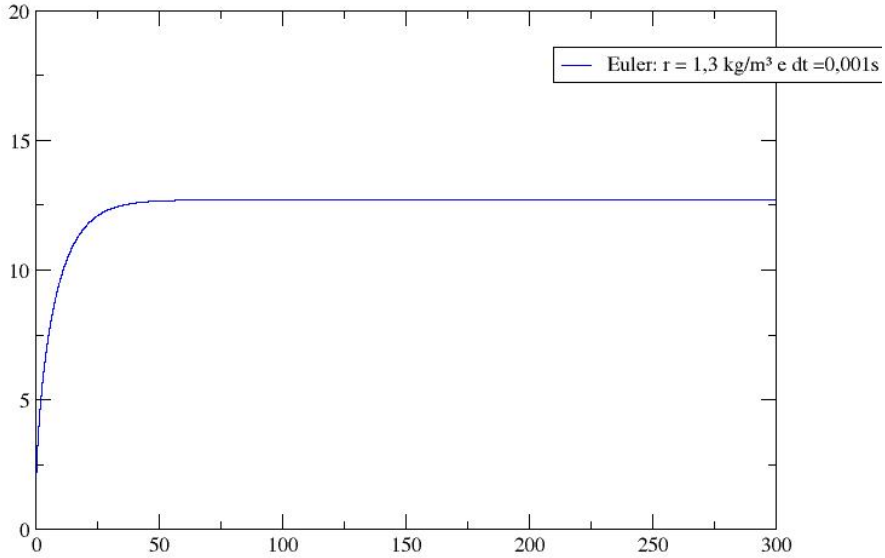
Ao rodar o programa também obtivemos a distância percorrida pelo ciclista nesse caso:

$$\Delta x = 171874{,}9299\ m \tag{20}$$

B. Movimento com resistência do ar

Ao executar o programa descrito obtemos:

Imagem 3 - Gráfico da velocidade em função do tempo com resistência para $\Delta t = 0{,}001\ s$



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Analisando o gráfico podemos perceber que existe um momento que a velocidade se estabiliza, essa seria a velocidade terminal, que nesse gráfico se aproxima de 12,5 m/s. Ela é alcançada próximo do tempo t = 50 s.

Podemos verificar se isso é compatível com resultados reais utilizando a equação:

$$v* = \left(\frac{2P_c}{\rho A}\right)^{\frac{1}{3}} \tag{21}$$

$$v* = 12{,}7059885670598856\ m/s \tag{22}$$

O que é compatível com o gráfico.

Já calculando a distância percorrida pelo ciclista nessa situação:

$$\Delta x = 3724.2045\ m/s$$

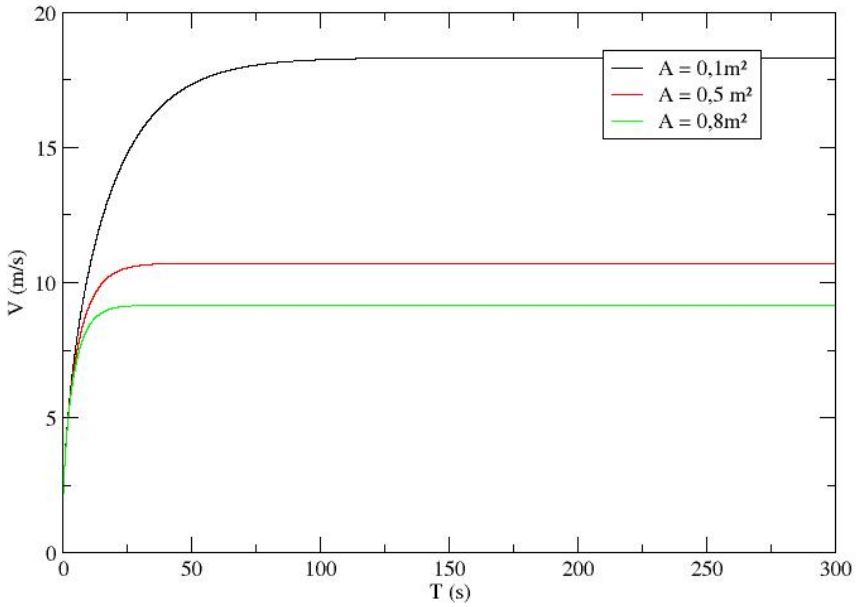
(23)

Que é claramente bem diferente da distância percorrida no item anterior. Podemos então ver (assumindo que os resultados estejam corretos) aplicar resistência pode afetar drasticamente a performance do ciclista.

C. A influência da área.

Ao executar o programa obtemos:

Imagem 4 - Gráfico da velocidade em função do tempo para diferentes áreas transversais.



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

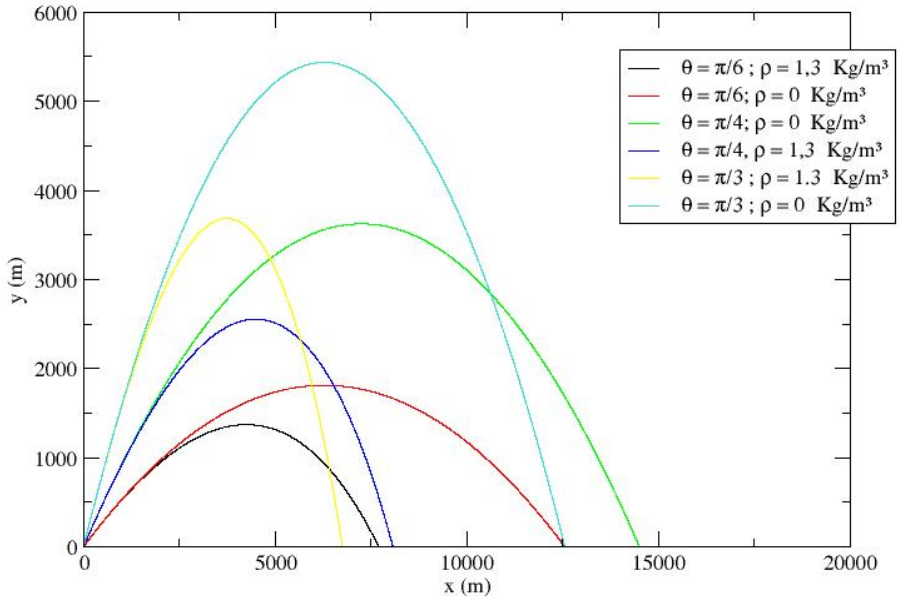
Através desse gráfico vemos que quanto menor a área transversal do ciclista maior a velocidade terminal que ele pode alcançar, por isso ciclistas profissionais se abaixam quando andando de bicicleta, para que possam diminuir sua área transversal.

3.2. Efeito resistivo do ar em movimentos bidimensionais

A. Trajetória do projétil

Ao rodar o programa obtemos:

Imagem 5 - Gráfico do percurso do projétil para diferentes resistências e ângulos iniciais.

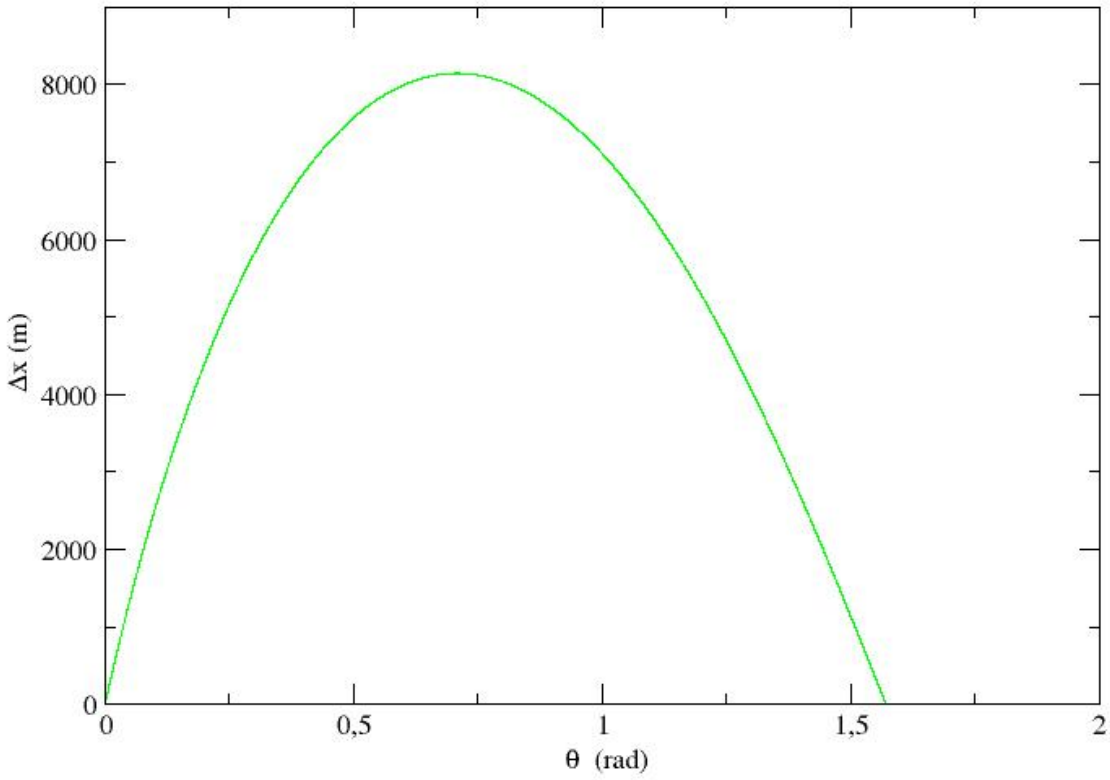


Fonte: Elaborado pelo compilador utilizando xmgrace[2]

B. Alcance

Ao executar o programa descrito obtivemos:

Imagem 6 - Gráfico do alcance em função do ângulo de lançamento



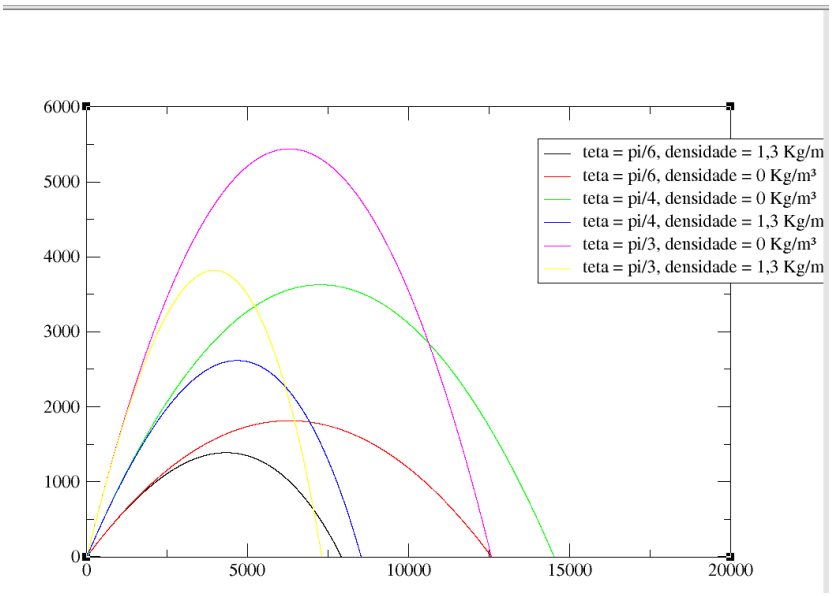
Fonte: Elaborada pelo compilador utilizando xmgrace[2]

A partir disso o ângulo $\theta máx = 0.7201\, rad$.

C. Variação de resistência

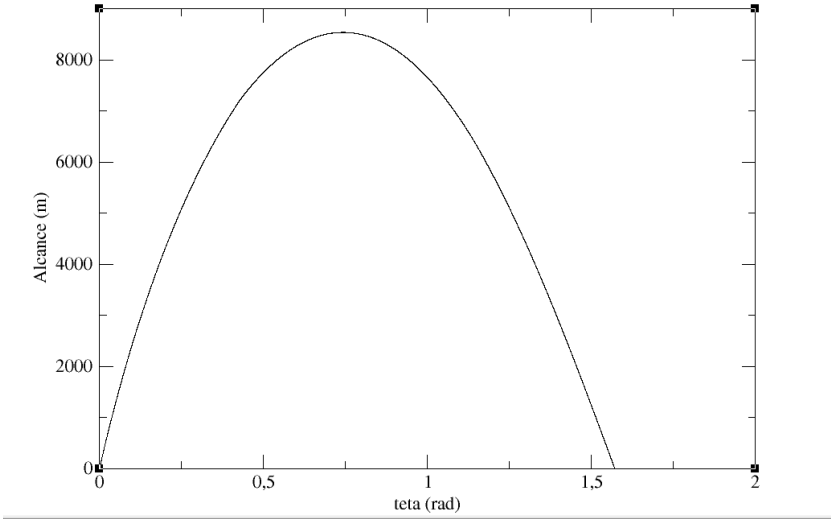
Refazendo os itens anteriores com uma resistência variável obtemos:

Imagem 7 - Gráfico da trajetória do projétil para resistências variáveis



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Imagem 8 - Gráfico do alcance em função do ângulo de lançamento para resistência variável



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Nesse caso $\theta máx = 0{,}7427\ rad$.

D. Velocidade de lançamento
Os novos alcances são:

Para $v = 373{,}23\ m/s$:

$$\Delta x = 14167{,}9188\ m$$

Para $v = 380{,}77\ m/s$

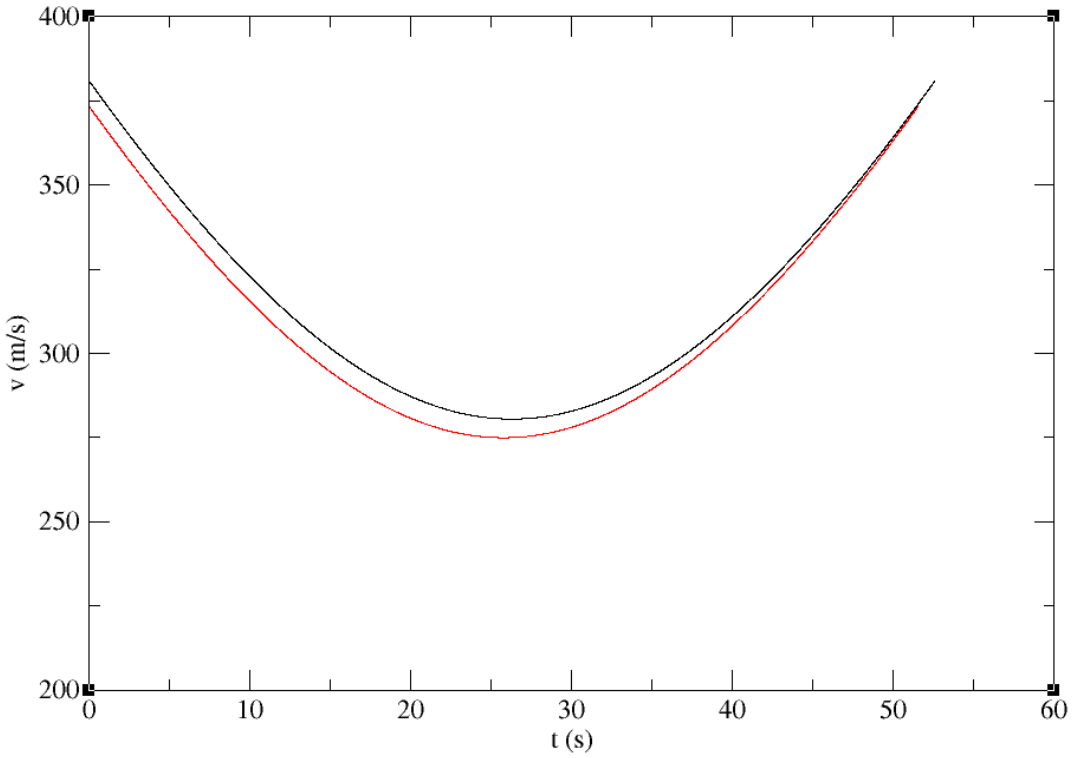
$$\Delta x = 14745{,}86\ m$$

Mesmo com a mudança de velocidade a área geral de pouso do projétil ficou próxima. A facilidade do acerto de um alvo dependerá do tamanho do alvo. Nesse caso se ele for menor do 600 m de diâmetro talvez uma dessas velocidades acerte e a outra não.

E. Velocidade em função do tempo

Executando o programa obtemos:

Imagem 9 - Gráfico da velocidade em função do tempo

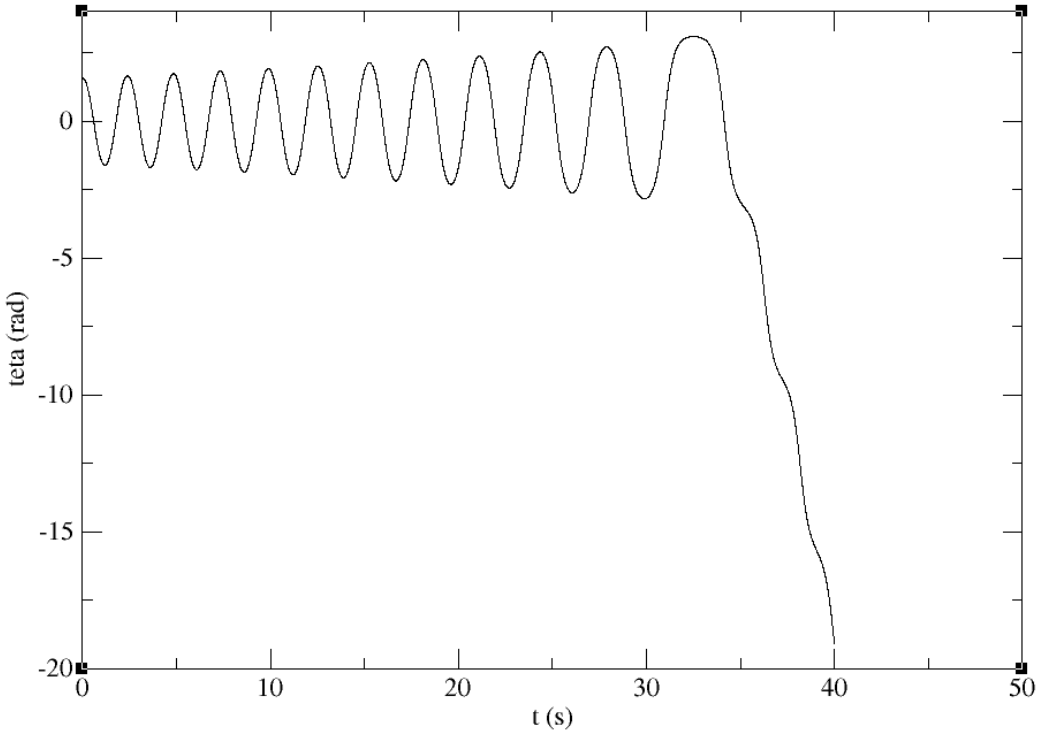


Fonte: Elaborado pelo compilador utilizando xmgrace[2]

3.3. Pêndulo SImples

A. Ângulo e Energia Mecânica
Ao rodar o programa obtemos:

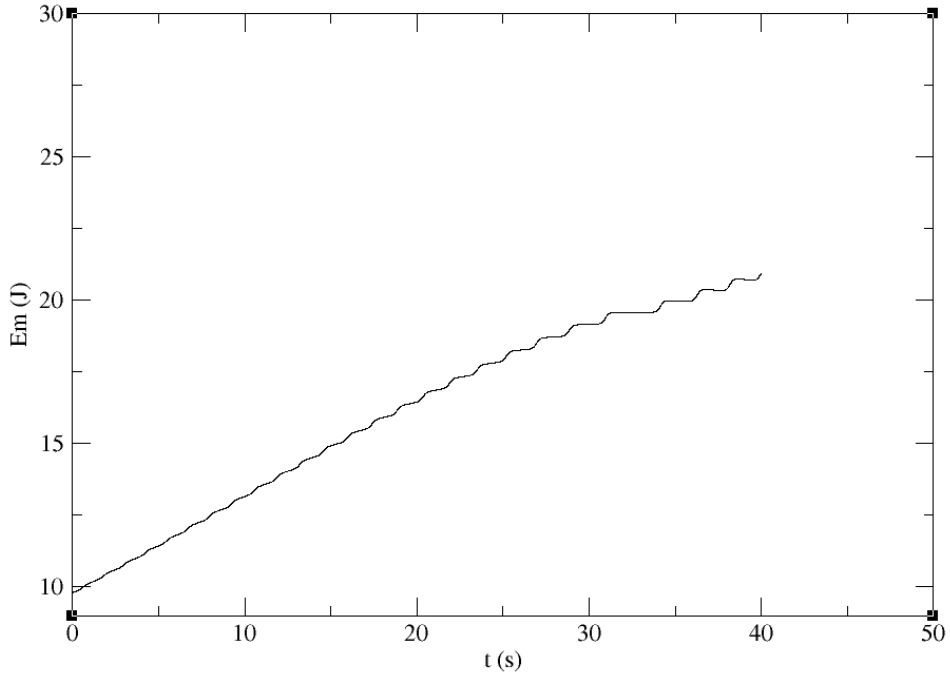
Imagem 10 - Gráfico do ângulo teta em função do tempo para o método de Euler



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Podemos ver que existe uma espécie de quebra próximo dos 30s desse gráfico. Talvez olhando o gráfico da Energia mecânica em função do tempo possamos ver uma explicação para isto:

Imagem 11 - Gráfico da energia mecânica em função do tempo para o método de Euler

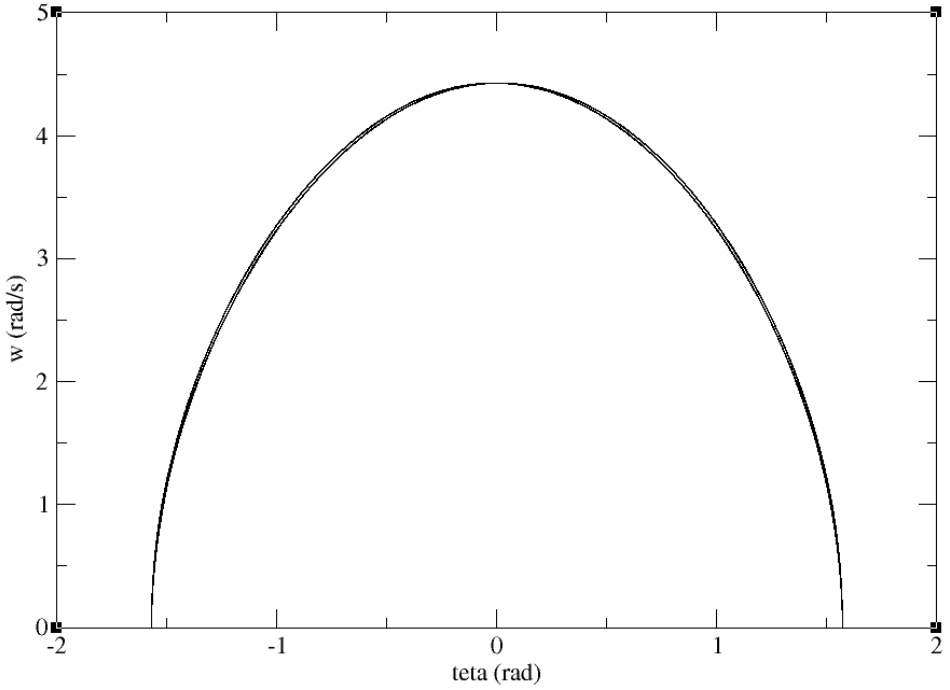


Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Vemos aqui que a energia mecânica não se conserva pelo método de Euler, ou seja ele não é eficiente para o cálculo de pêndulos simples.

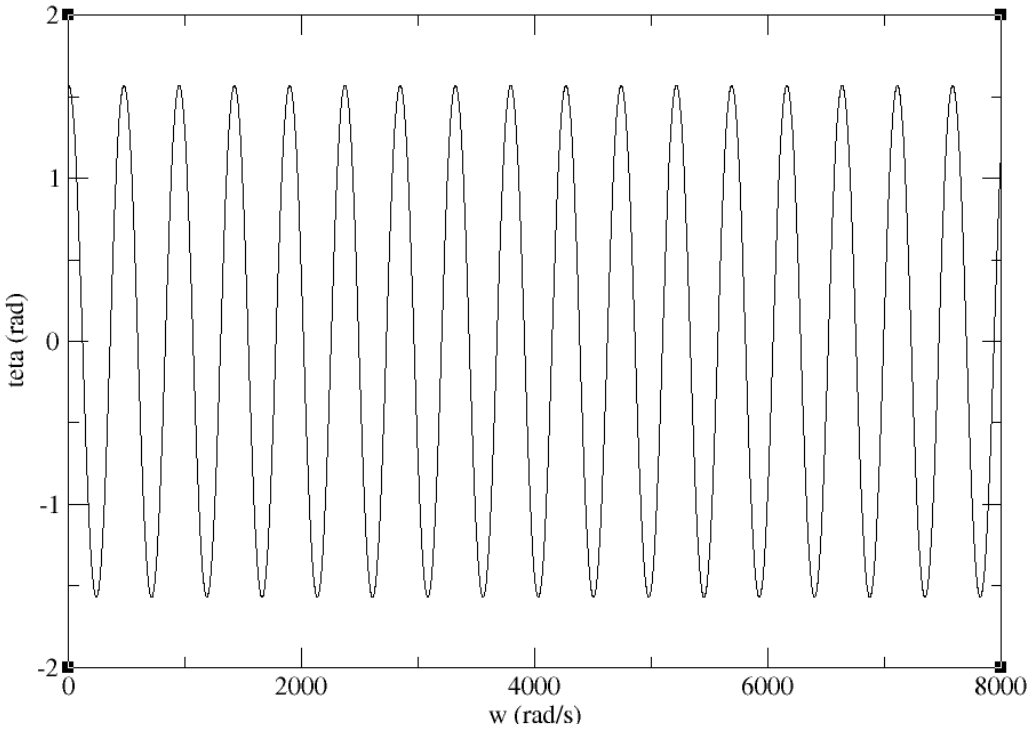
B. Espaço de fase

Imagem 12 - Espaço de fase para o método de Euler



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Imagem 13 - Espaço de fase exato



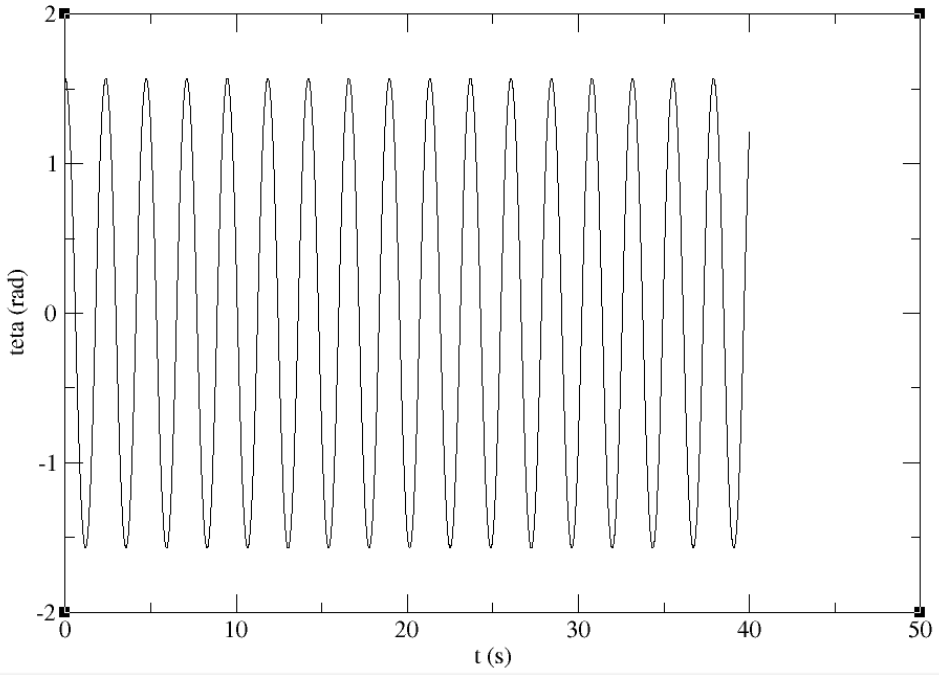
Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Vemos uma clara diferença o que suporta a ideia de que o método de Euler é inadequado para lidar com movimento pendular.

C. Método de Euler-Cromer

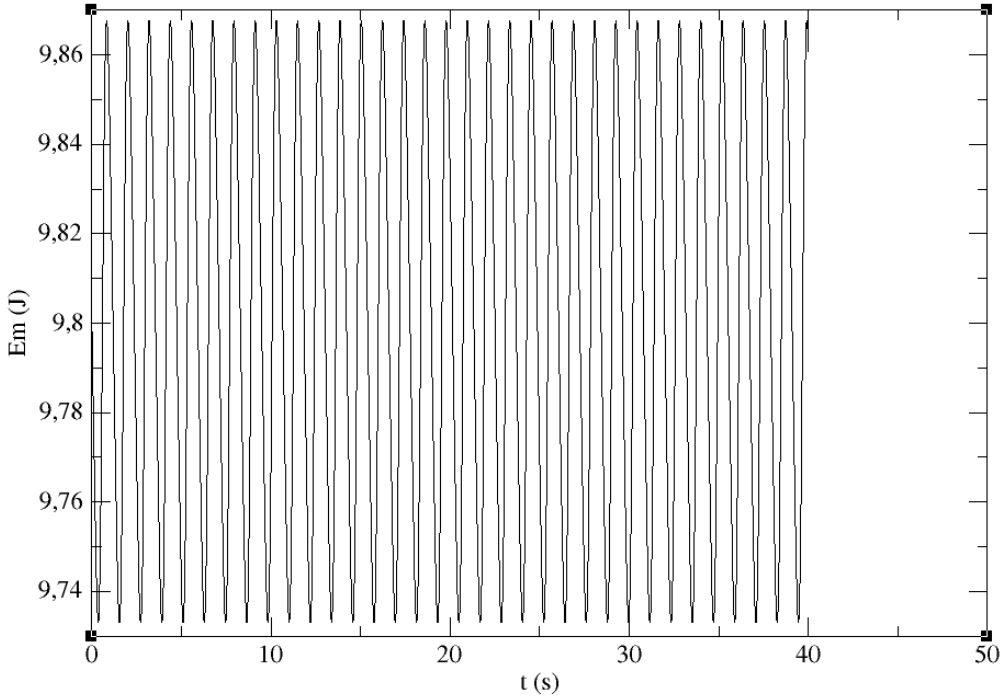
Refazendo os itens anteriores utilizando o método de Euler-Cromer, obtemos:

Imagem 14 - Gráfico do ângulo em relação ao tempo para método de Euler-Cromer



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Imagem 15 - Gráfico da Energia mecânica em função do tempo



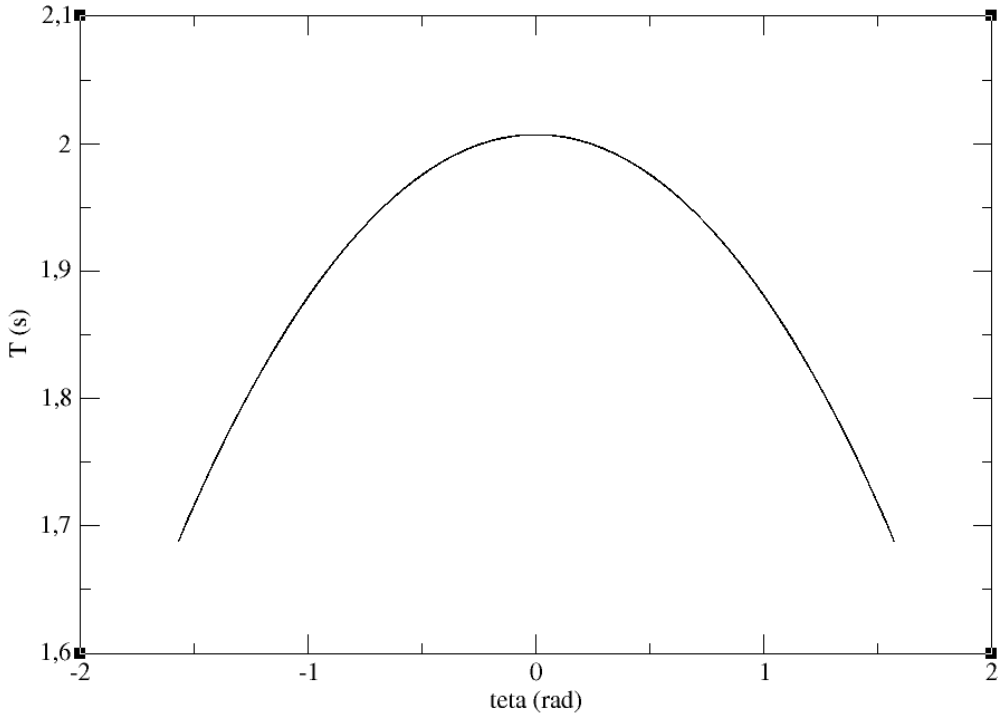
Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Podemos perceber que para esse método temos a conservação da energia mecânica, fazendo com que este fique mais preciso e seja mais adequado que o método de Euler.

D. Período do pêndulo

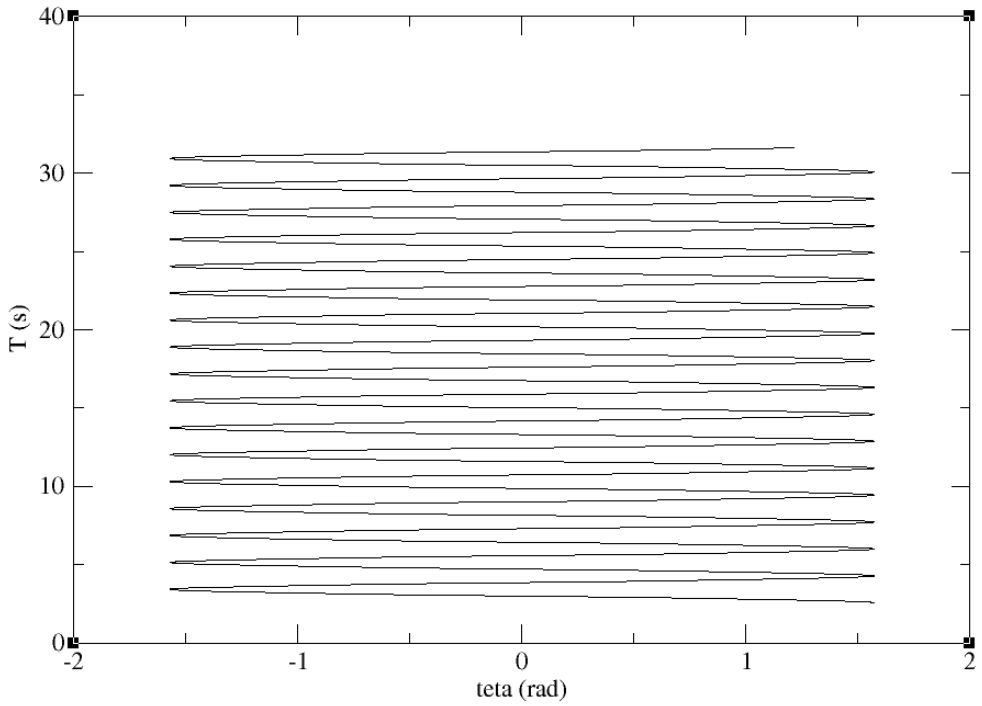
Executando os programas obtemos:

Imagem 16 - Gráfico do período por teta utilizando o método de Euler-Cromer



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Imagem 17 - Gráfico do período pelo teta por integral de trapézio



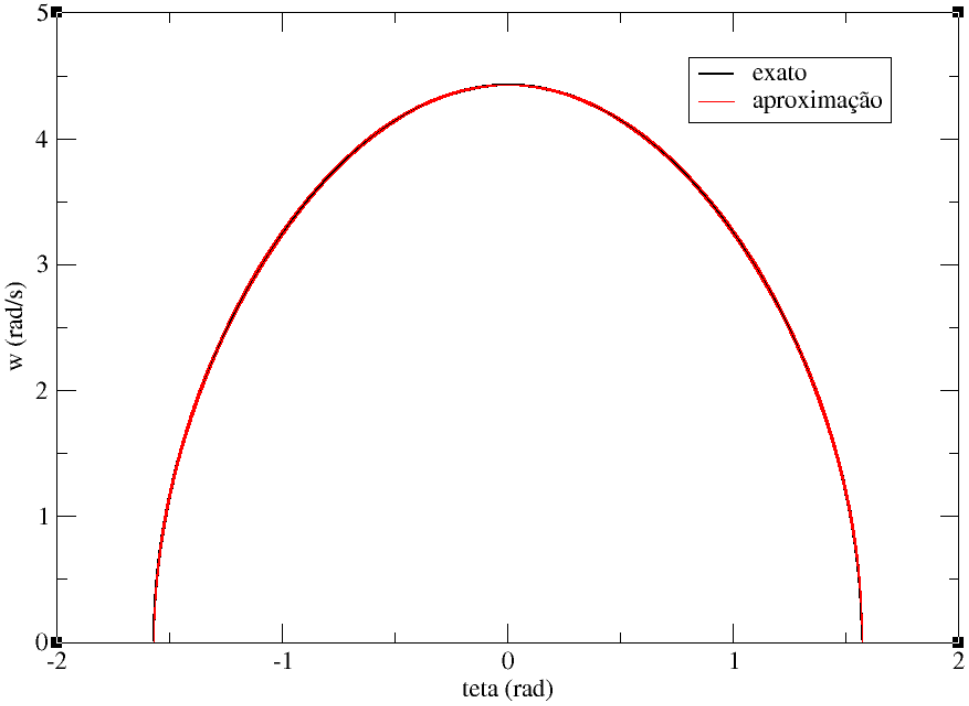
Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Este é um resultado peculiar. Seria plausível dizer que houve algum problema na implementação da integral do trapézio para uma integral elíptica.

E. Método de Verlet

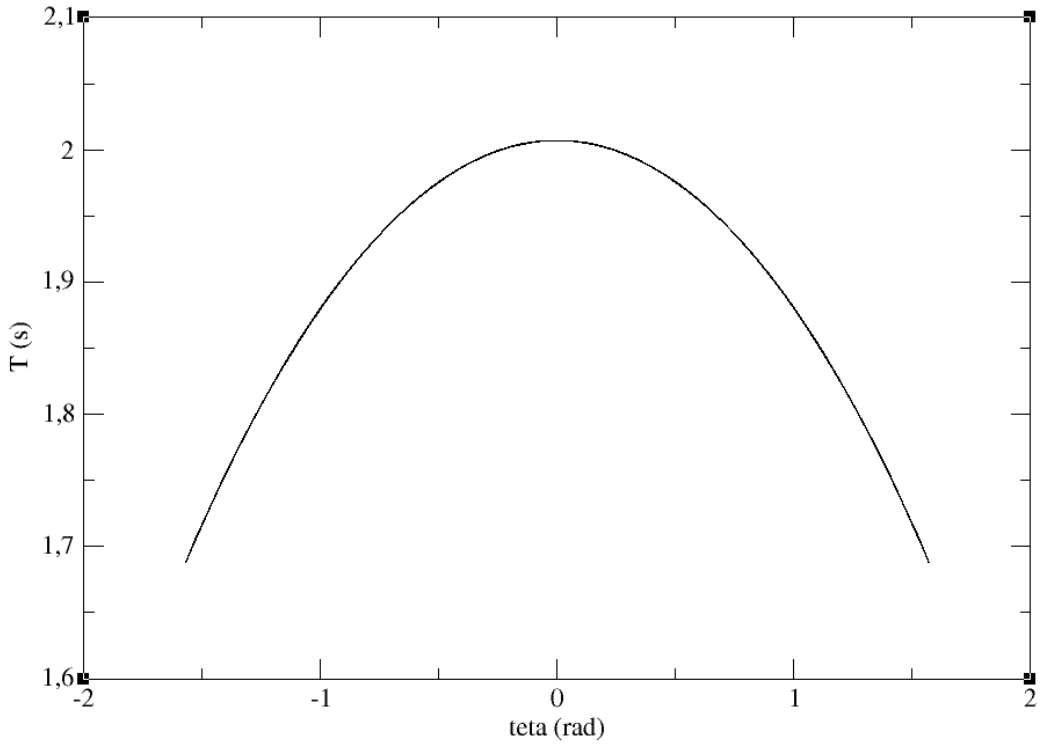
Realizando os cálculos pelo método de Verlet obtemos:

Imagem 18 - Espaço de fase pelo método de Verlet



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

Imagem 19 - Período em função de teta calculado pelo método de Verlet



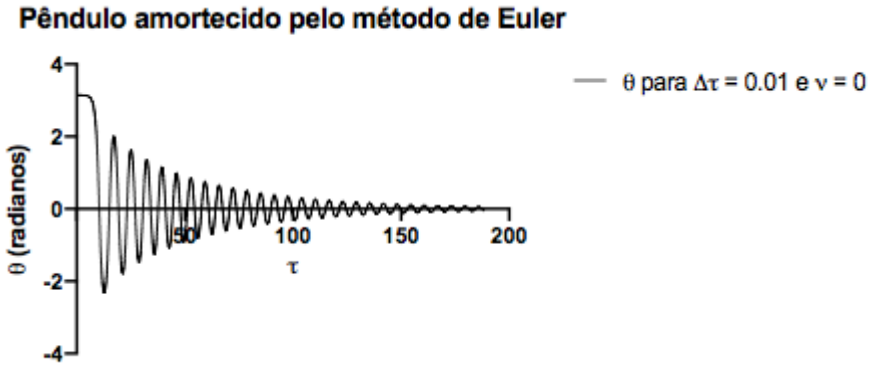
Fonte: Elaborado pelo compilador utilizando xmgrace[2]

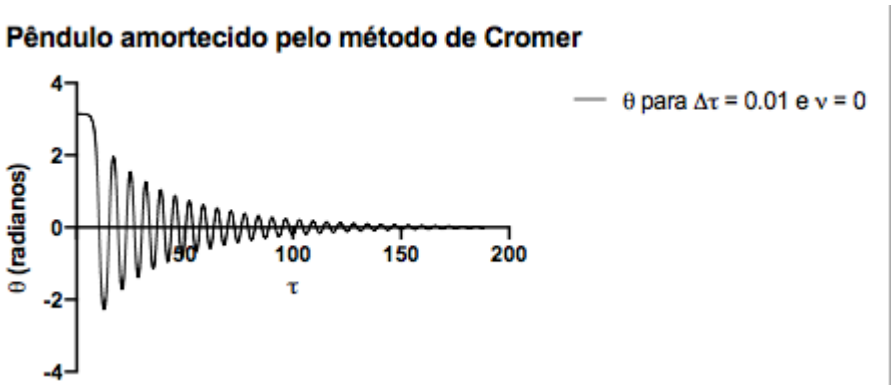
Considerando esses resultados temos evidências de que o método de Verlet é possivelmente o mais apurado.

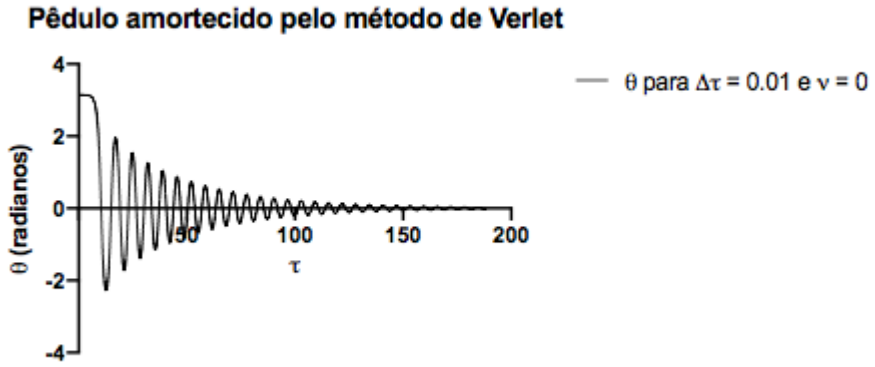
3.4. Estabilização dinâmica

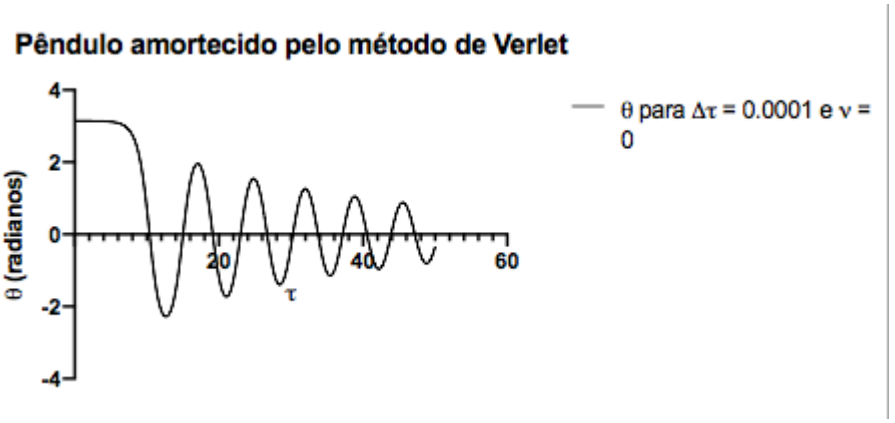
A. Movimento

Imagem 20, 21, 22 e 23- Teta em função de t para $\nu = 0$pelo método de Cromer, Euler e Verlet







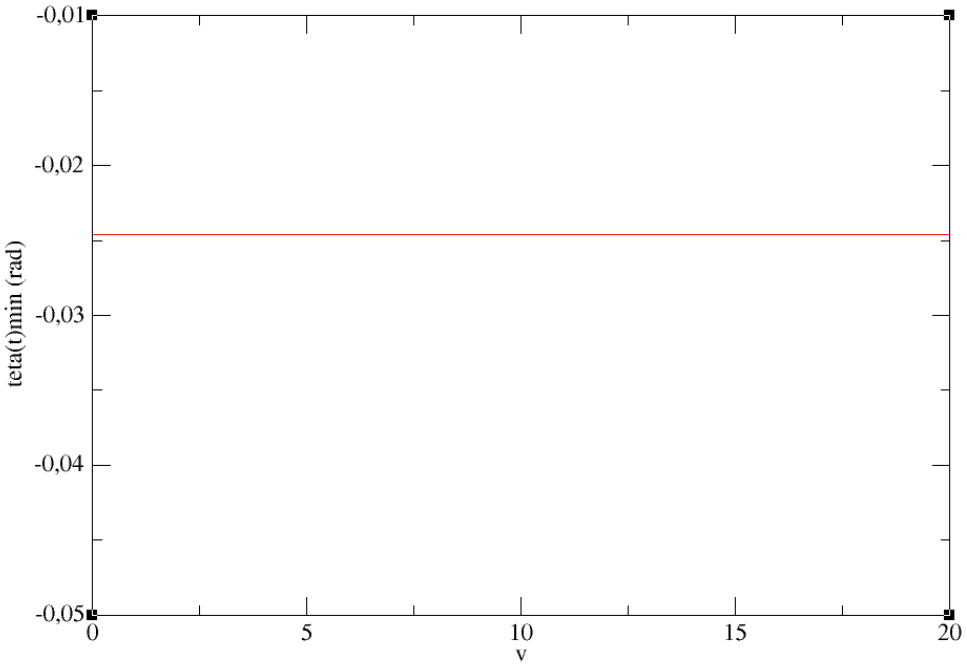


Fonte: Elaborado pelo compilador utilizando Graphpad Prism[4]

O movimento é sub-amortecido, pois ocila antes de decair e decai lentamente.

B e C. Frequência

Imagem 24 - Gráfico de θmin por $\nu *$



Fonte: Elaborado pelo compilador utilizando xmgrace[2]

4. REFERÊNCIAS

[1]**Gary Garber's Blog**. Gary Garbers Blog Energy in a Pendulum Comments. Disponível em: <https://blogs.bu.edu/ggarber/interlace/pendulum/energy-in-a-pendulum/>.

[2]Grace Home. WIS Plasma Lab. Disponível em: <http://plasma-gate.weizmann.ac.il/Grace/>.

[3]LGUNSEOR. **Pendulum velocity at any given angle**. Physics Forums | Science Articles, Homework Help, Discussion. Disponível em: <https://www.physicsforums.com/threads/pendulum-velocity-at-any-given-angle.737577/>.

[4]Prism. GraphPad. Disponível em: <https://www.graphpad.com/scientific-software/prism/>.

[5]Simple Pendulum. Pendulum. Disponível em: <http://hyperphysics.phy-

astr.gsu.edu/hbase/pend.html#c3>.

[6]Hoyos; José A. **Introdução a programação - Notas de aula** Disponível em:

<http://www.ifsc.usp.br/~hoyos/courses/2019/7600017/intro.pdf>

[7] Hoyos; José A. **Projeto 3 - Movimento realístico** Disponível em:

<http://www.ifsc.usp.br/~hoyos/courses/2019/7600017/p2.pdf>

[8] Kidd, Richard B.; Fogg, Stuart L. **A Simple Formula for the Large-Angle Pendulum Period.** Disponível em:

<http://users.df.uba.ar/sgil/physics_paper_doc/papers_phys/mechan/Pendulo2.pdf

,>