



IFSC

**UNIVERSIDADE
DE SÃO PAULO**

Instituto de Física de São Carlos

UNIVERSIDADE DE SÃO PAULO - USP

INTRODUÇÃO À FÍSICA COMPUTACIONAL – 7600017 – 2025/2

PROF. FRANCISCO C. ALCARAZ

RELATÓRIO DO 4º PROJETO

JOÃO VITOR LIMA DE OLIVEIRA - 12694394

São Carlos

2025

Parte I

Introdução Geral

MOTIVAÇÃO

O estudo do movimento oscilatório é um tema fundamental na Física, pois está presente em diversos fenômenos naturais e tecnológicos. Desde o pêndulo simples até sistemas mais complexos, compreender como um corpo oscila e como essas oscilações se comportam com o tempo é essencial para entender vários tipos de movimento.

Neste projeto, o objetivo é analisar o comportamento do pêndulo simples e suas variações usando métodos numéricos, como o de Euler e o de Euler-Cromer. A ideia é observar como diferentes aproximações e parâmetros influenciam o movimento, além de comparar os resultados obtidos numericamente com as soluções analíticas conhecidas.

Além disso, o projeto permite explorar casos mais realistas, incluindo efeitos dissipativos e forças externas, que tornam o sistema mais interessante e imprevisível. Nesses casos, o pêndulo pode apresentar desde movimentos periódicos até comportamentos caóticos, dependendo das condições iniciais e dos parâmetros escolhidos. Estudar esse tipo de sistema ajuda a entender melhor como surgem fenômenos caóticos na natureza e mostra a importância das simulações computacionais para investigar situações em que o cálculo analítico não é suficiente.

Parte II

Desenvolvimento

TAREFA - A

CÓDIGO

Figura 1 – Função principal do código.

```
1 program main
2     implicit real*8 (a-h,o-z)
3     call euler_crommer()
4     call euler()
5 end program main
```

Fonte: Compilado pelo Autor.

Figura 2 – Implementação do método de Euler-Cromer.

```
1 subroutine euler_crommer()
2 implicit real*8 (a-h,o-z)
3 parameter (imax = 1e3)
4 dimension w(0:imax), th(0:imax)
5
6 g = 9.81d0
7 m = 1d0
8 rl = 9.81d0
9 dt = 0.01d0
10 pi = acos(-1d0)
11
12 C      Condições Iniciais
13 w(0) = 0d0
14 th(0) = pi/6
15
16
17
18 do i = 0, imax-1
19 w(i+1) = w(i) - (g/r1)*sin(th(i))*dt
20 th(i+1) = th(i) + w(i+1)*dt
21
22 end do
23
24 open(unit=1,file='saida-1-12694394.txt')
25 do i = 0, imax
26
27 ! Trem do angulo
28 if (th(i+1) .gt. 2*pi) then
29 th(i+1) = th(i+1) - 2*pi
30 else if (th(i+1) .lt. 0d0) then
31 th(i+1) = th(i+1) + 2*pi
32 end if
33
34 write(1,*) i*dt, w(i), th(i)
35 end do
36 close(1)
37
38
39 open(unit=2,file='saida-2-12694394.txt')
40 do i = 0, imax
41 E_kin = 0.5d0*m*((rl*w(i))**2)
42 E_pot = -m*g*rl*cos(th(i))
43
44
45 write(2,*) i*dt, E_kin,E_pot,E_kin+E_pot
46 end do
47 close(2)
48
49 end subroutine euler_crommer
```

Fonte: Compilado pelo Autor.

Figura 3 – Implementação do método de Euler.

```
1 subroutine euler()
2 implicit real*8 (a-h,o-z)
3 parameter (imax = 1e4)
4 dimension w(0:imax), th(0:imax)
5
6 g = 9.81d0
7 m = 1d0
8 rl = 9.81d0
9 dt = 0.01d0
10 pi = acos(-1d0)
11
12 C      Condições Iniciais
13 w(0) = 0d0
14 th(0) = pi/6d0
15
16
17
18 do i = 0, imax-1
19 w(i+1) = w(i) - (g/r1) * sin(th(i)) * dt
20 th(i+1) = th(i) + w(i) * dt
21 end do
22
23 open(unit=3,file='saida-3-12694394.txt')
24 do i = 0, imax
25
26 ! Trem dos angulos
27 if (th(i+1) .gt. 2*pi) then
28 th(i+1) = th(i+1) - 2*pi
29 else if (th(i+1) .lt. 0d0) then
30 th(i+1) = th(i+1) + 2*pi
31 end if
32
33 write(3,*) i*dt, w(i), th(i)
34 end do
35 close(3)
36
37
38 open(unit=4,file='saida-4-12694394.txt')
39 do i = 0, imax
40 E_kin = 0.5d0*m*((rl*w(i))**2)
41 E_pot = -m*g*rl*cos(th(i))
42
43 write(4,*) i*dt, E_kin,E_pot,E_kin+E_pot
44 end do
45 close(4)
46
47 end subroutine euler
```

Fonte: Compilado pelo Autor.

DESCRIÇÃO DO CÓDIGO

O programa `main` tem como objetivo comparar dois métodos numéricos de integração aplicados ao movimento de um pêndulo simples: o método de *Euler* e o método de *Euler-Cromer*. Ambos são implementados em subrotinas independentes e chamados a partir do programa principal.

No início do código, é utilizado o comando:

```
1 implicit real*8 (a-h,o-z)
```

que define todas as variáveis cujos nomes começam com as letras de **a** a **h** e **o** a **z** como números reais de dupla precisão. Em seguida, o programa principal executa as duas subrotinas:

```
1 call euler_crommer()
2 call euler()
```

As duas subrotinas calculam numericamente a evolução temporal da velocidade angular (ω) e do ângulo (θ) de um pêndulo simples, de comprimento r_l e massa m , sob ação da gravidade g .

Subrotina `euler_crommer`

Na subrotina `euler_crommer`, o número máximo de iterações é definido por `imax = 1e3`, e são declarados os vetores `w(0:imax)` e `th(0:imax)`, que armazenam respectivamente a velocidade angular e o ângulo.

Os parâmetros físicos e o passo temporal são definidos como:

```
1 g = 9.81d0
2 m = 1d0
3 rl = 9.81d0
4 dt = 0.01d0
5 pi = acos(-1d0)
```

As condições iniciais são estabelecidas como $\omega(0) = 0$ e $\theta(0) = \pi/6$. O método de Euler-Cromer é então aplicado no laço:

```
1 do i = 0, imax-1
2   w(i+1) = w(i) - (g/r1)*th(i)*dt
3   th(i+1) = th(i) + w(i+1)*dt
4 end do
```

onde o valor atualizado da velocidade angular é utilizado imediatamente no cálculo do ângulo, característica principal do método de Euler-Cromer, que tende a conservar melhor a energia mecânica em sistemas oscilatórios.

Os resultados da evolução temporal de ω e θ são gravados no arquivo saída-1-12694394.txt enquanto as energias cinética, potencial e total são salvas em saída-2-12694394.txt. Antes da escrita, o código realiza uma correção para manter o ângulo θ dentro do intervalo $[0, 2\pi]$.

As expressões utilizadas para as energias são:

$$E_{\text{cin}} = \frac{1}{2}m(r_l \omega)^2,$$
$$E_{\text{pot}} = -mgr_l \cos(\theta).$$

A soma dessas quantidades fornece a energia total $E_{\text{tot}} = E_{\text{cin}} + E_{\text{pot}}$.

Subrotina euler

A subrotina `euler` implementa o método de Euler simples, para fins de comparação com o método anterior. São utilizados os mesmos parâmetros físicos, mas com um número maior de iterações (`imax = 1e4`), a fim de aumentar a resolução temporal.

O esquema numérico empregado é dado por:

```
1      do i = 0, imax-1
2          w(i+1) = w(i) - (g/r1)*th(i)*dt
3          th(i+1) = th(i) + w(i)*dt
4      end do
```

Diferentemente do método de Euler-Cromer, aqui o novo valor de θ é calculado usando a velocidade *anterior* $w(i)$, o que leva a uma maior variação da energia total ao longo do tempo.

Os resultados para ω e θ são gravados no arquivo saída-3-12694394.txt, e as energias correspondentes em saída-4-12694394.txt.

Assim como na subrotina anterior, é aplicado o ajuste de periodicidade para o ângulo e são calculadas as energias cinética, potencial e total com as mesmas expressões.

Resumo

Em síntese, o código compara numericamente os métodos de Euler e Euler-Cromer na simulação de um pêndulo simples, permitindo observar diferenças na

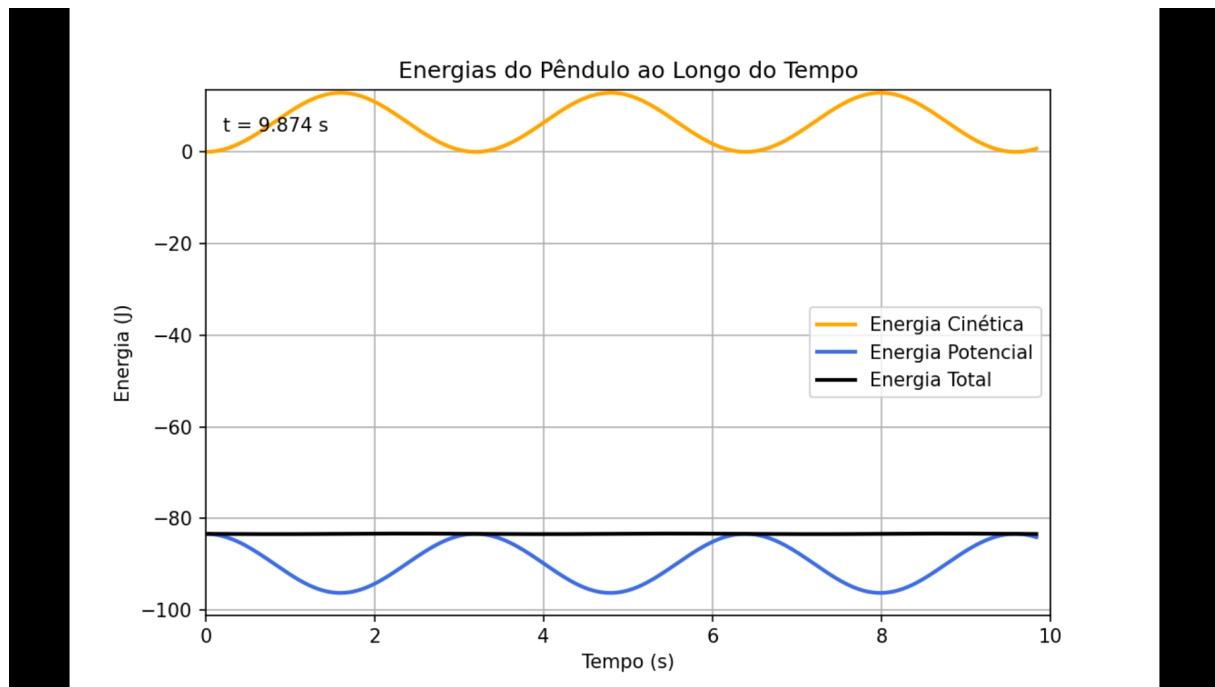
conservação da energia ao longo do tempo. Cada método gera dois arquivos de saída: um contendo os parâmetros dinâmicos (ω e θ) e outro contendo as energias calculadas em cada instante.

RESULTADOS

Figura 4 – Video simulação do pendulo.

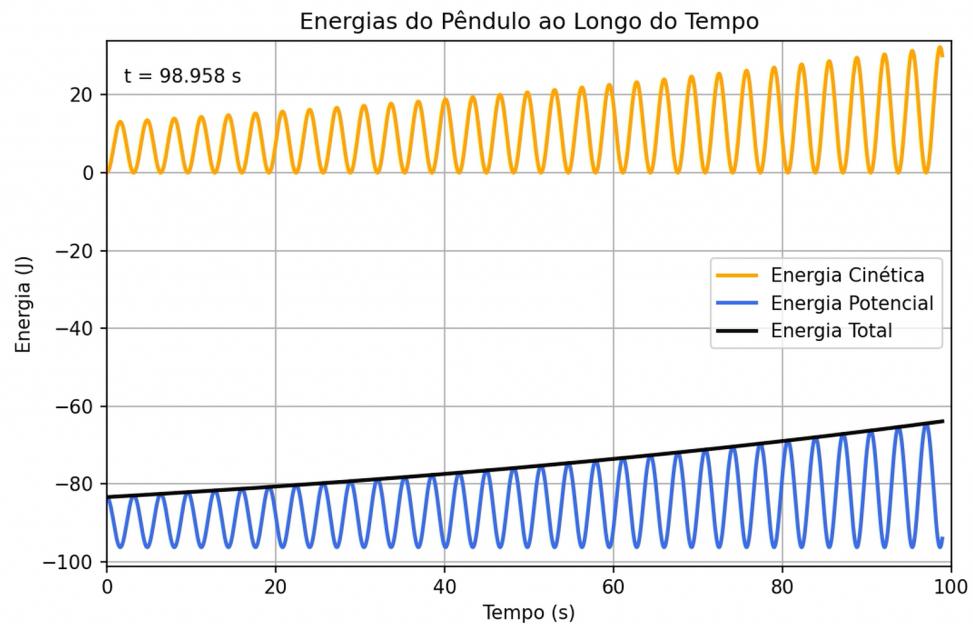
Fonte: Compilado pelo Autor.

Figura 5 – Energia método Euler-Cromer



Fonte: Compilado pelo Autor.

Figura 6 – Energia método Euler



Fonte: Compilado pelo Autor.

TAREFA - B

CÓDIGO

Figura 7 – Função principal do código.

```
1 program main
2 call euler_crommer()
3 end program main
```

Fonte: Compilado pelo Autor.

Figura 8 – Implementação do método de Euler-Cromer.

```
1      subroutine euler_crommer()
2      parameter (imax=1e4)
3      implicit real*8(a-h,o-z)
4      dimension th(0:imax),w(0:imax)
5
6      C      Constantes
7      rl = 9.81d0
8      g = 9.81d0
9      m = 1d0
10     y = 0.050d0 !Gamma
11     F0 = 0.50d0 !Força externa
12     ome = 0.75d0 ! Frequênciada força externa
13     pi = acos(-1d0)
14     dt = 0.04d0
15
16      C      Valores iniciais
17      w(0) = 0
18      th(0) = pi/3d0
19
20      C      Cálculo
21
22      do i = 0,imax-1
23          F_ex = -y*w(i) + F0*sin(ome*i*dt)
24          w(i+1) = w(i)-(g/r1)*sin(th(i))*dt + F_ex*dt
25          th(i+1) = th(i) + w(i+1)*dt
26      end do
27
28      C      Salva a posição
29      open(unit=1,file='saída-1-12694394.txt')
30      do i =0,imax
31          ! Trem das posições
32          if (th(i+1) .GT. 2*pi) then
33              th(i+1) = th(i+1) - 2*pi
34          else if (th(i+1) .LT. 0 ) then
35              th(i+1) = th(i+1) + 2*pi
36          end if
37
38          write(1,7) i*dt, w(i), th(i)
39      end do
40      format(F12.6,F12.6,F12.6)
41      close(1)
42
43      end subroutine euler_crommer
```

Fonte: Compilado pelo Autor.

DESCRIÇÃO DO CÓDIGO

O programa `main` tem como objetivo simular numericamente o movimento de um pêndulo simples sujeito a atrito viscoso e a uma força externa periódica, utilizando o método de integração de *Euler-Crommer*. O cálculo é implementado na subrotina `euler_crommer`, chamada a partir do programa principal.

A diretiva

```
1 implicit real*8 (a-h,o-z)
```

define todas as variáveis cujos nomes iniciam com as letras de **a** a **h** e **o** a **z** como números reais de dupla precisão, garantindo maior precisão nos cálculos numéricos.

Subrotina `euler_crommer`

A subrotina `euler_crommer` executa o cálculo principal da simulação. Primeiramente, são definidos o número máximo de iterações (`imax = 1e4`) e os vetores `w(0:imax)` e `th(0:imax)`, que armazenam respectivamente a velocidade angular (ω) e o ângulo (θ) do pêndulo em cada passo de tempo.

As constantes físicas e parâmetros do sistema são definidos como:

```
1 r1 = 9.81d0      ! Comprimento do pêndulo
2 g  = 9.81d0      ! Aceleração da gravidade
3 m  = 1d0         ! Massa
4 y  = 0.050d0     ! Coeficiente de amortecimento (Gamma)
5 F0 = 0.50d0      ! Amplitude da força externa
6 ome = 0.75d0     ! Frequência angular da força externa
7 dt = 0.04d0      ! Passo temporal
8 pi = acos(-1d0)  ! Valor de pi
```

As condições iniciais são fixadas como $\omega(0) = 0$ e $\theta(0) = \pi/3$.

O método de Euler-Cromer é então aplicado dentro de um laço de iterações que atualiza a velocidade angular e o ângulo a cada passo de tempo. A força externa e o termo de amortecimento são incluídos na equação de movimento, resultando no seguinte esquema numérico:

```
1 do i = 0, imax-1
2   F_ex = -y*w(i) + F0*sin(ome*i*dt)
3   w(i+1) = w(i) - (g/r1)*sin(th(i))*dt + F_ex*dt
4   th(i+1) = th(i) + w(i+1)*dt
5 end do
```

O termo F_{ex} representa a soma da força de amortecimento ($-y\omega$) e da força externa periódica ($F_0 \sin(\omega_{\text{ext}}t)$). O uso do método de Euler-Cromer garante maior estabilidade e melhor conservação de energia do sistema em comparação com o método de Euler tradicional, especialmente em sistemas oscilatórios.

Saída dos resultados

Após o cálculo, os resultados da simulação são gravados no arquivo `saída-1-12694394.txt`. Cada linha do arquivo contém o tempo ($t = i \cdot \Delta t$), a velocidade angular ω e o ângulo θ , conforme o formato:

```
1 7 format(F12.6,F12.6,F12.6)
```

Antes da escrita, o programa corrige o ângulo θ para mantê-lo dentro do intervalo $[0, 2\pi]$, utilizando a verificação:

```
1 if (th(i+1) .GT. 2*pi) then
2   th(i+1) = th(i+1) - 2*pi
3 else if (th(i+1) .LT. 0) then
4   th(i+1) = th(i+1) + 2*pi
5 end if
```

Por fim, o arquivo é fechado com o comando `close(1)`.

Resumo

Em resumo, o código realiza a integração numérica das equações de movimento de um pêndulo forçado e amortecido pelo método de Euler-Cromer. O programa permite estudar o comportamento dinâmico do sistema sob diferentes condições de força e amortecimento, sendo útil para análises de regimes oscilatórios, ressonância e comportamento caótico em pêndulos não-lineares.

RESULTADOS

Figura 9 – Enunciado da Tarefa 1.

Fonte: Compilado pelo Autor.

TAREFA - C

ENUNCIADO

Figura 10 – Enunciado da Tarefa C.

TAREFA C: Para verificarmos a existência ou não do regime caótico vamos considerar, como antes, $\gamma = 0.05$, $\Omega = \frac{2}{3}$, $\Delta t = 0.04$, $F_0 = \frac{1}{2}$ e $F_0 = 1.2$. Consideramos agora o movimento de dois pêndulos soltos com velocidade nula em ângulos iniciais que difiram de $\theta_0^{(2)} - \theta_0^{(1)} = \Delta\theta_0 = 0.001$ radianos. Faça um gráfico de $\Delta\theta(t) = \theta^{(2)}(t) - \theta^{(1)}(t)$ para os casos em que $F_0 = 0.5$ e $F_0 = 1.2$. Repare que no primeiro caso as trajetórias se aproximam exponencialmente (não caótico), enquanto que no segundo caso as mesmas se afastam exponencialmente (caótico), ou seja

$$\Delta\theta(t) \approx \exp \lambda t \quad (0.10)$$

onde

$$\lambda < 0 \rightarrow \text{não caótico}, \quad \lambda > 0 \rightarrow \text{caótico}. \quad (0.11)$$

Faça os gráficos $\Delta(\theta) \times t$ com escala semi-logarítmica e estime o parâmetro λ , chamado de **expoente de Liapunov**.

3

Na realidade o movimento caótico não é tão imprevisível quanto nos parece à primeira vista. Ele possui certa estrutura que podemos visualizar traçando o gráfico $\omega(\theta)$.

Fonte: Compilado pelo Autor.

CÓDIGO

Figura 11 – Função principal do código.

```
1 program main
2 call euler_crommer()
3 end program main
```

Fonte: Compilado pelo Autor.

Figura 12 – Implementação do método de Euler-Cromer.

```
1      subroutine euler_crommer()
2          parameter(imax=(3*1e3))
3          dimension w1(0:imax), th1(0:imax)
4          dimension w2(0:imax), th2(0:imax)
5
6
7      C      Parametros
8          g = 9.81e0
9          rl = 9.81e0
10         rm = 1e0
11         pi = acos(-1e0)
12         dt = 0.04e0
13         ! Parametros da força
14         F0_1 = 0.5e0
15         F0_2 = 0.5e0
16         ome_1 = 0.75e0
17         ome_2 = 0.75e0
18         y_1 = 0.05e0
19         y_2 = 0.05e0
20
21     C      Condições iniciais
22         ! Velocidades angulares
23         w1(0) = 0
24         w2(0) = 0
25         ! Angulo inicial
26         th1(0) = 1e0
27         th2(0) = 1e0 + 0.001e0
28
29     C      Cálculo
30         do i = 0,imax-1
31             F1 = - y_1*w1(i) + F0_1*sin(ome_1*i*dt)
32             F2 = - y_2*w2(i) + F0_2*sin(ome_2*i*dt)
33
34             w1(i+1) = w1(i) - (g/r1)*sin(th1(i))*dt + F1*dt
35             w2(i+1) = w2(i) - (g/r2)*sin(th2(i))*dt + F2*dt
36
37             th1(i+1) = th1(i) + w1(i+1)*dt
38             th2(i+1) = th2(i) + w2(i+1)*dt
39         end do
40
41     C      Salva os resultados
42         open(unit=1,file='saída-2-12694394.txt')
43         write(1,3)
44         do i = 0,imax
45             if (th1(i+1) .GT. 2e0*pi) then
46                 th1(i+1) = th1(i+1) - 2e0*pi
47             else if (th1(i+1) .LT. 0e0) then
48                 th1(i+1) = th1(i+1) + 2e0*pi
49             end if
50
51             if (th2(i+1) .GT. 2e0*pi) then
52                 th2(i+1) = th2(i+1) - 2e0*pi
53             else if (th2(i+1) .LT. 0e0) then
54                 th2(i+1) = th2(i+1) + 2e0*pi
55             end if
56
57             r1 = th1(i)
58             r2 = th2(i)
```

DESCRIÇÃO DO CÓDIGO

O programa `main` tem como objetivo comparar a evolução temporal de dois pêndulos simples sujeitos a atrito viscoso e a uma força externa periódica, utilizando o método numérico de *Euler-Crommer*. A simulação busca observar como pequenas variações nas condições iniciais podem gerar diferenças significativas no comportamento do sistema, caracterizando um regime caótico.

O cálculo principal é realizado na subrotina `euler_crommer`, chamada diretamente pelo programa principal.

Parâmetros e variáveis

Na subrotina `euler_crommer`, é definido o número máximo de iterações `imax` = 3×10^3 e são criados quatro vetores: `w1`, `th1`, `w2` e `th2`, que representam respectivamente as velocidades angulares (ω) e ângulos (θ) dos dois pêndulos.

As constantes físicas e os parâmetros da força externa são definidos como:

```
1 g   = 9.81e0          ! Aceleração da gravidade
2 rl  = 9.81e0          ! Comprimento do pêndulo
3 rm  = 1e0              ! Massa
4 pi  = acos(-1e0)       ! Valor de pi
5 dt  = 0.04e0           ! Passo temporal

6

7 ! Parâmetros da força externa
8 F0_1 = 0.5e0           ! Amplitude da força (pêndulo 1)
9 F0_2 = 0.5e0           ! Amplitude da força (pêndulo 2)
10 ome_1 = 0.75e0         ! Frequência da força (pêndulo 1)
11 ome_2 = 0.75e0         ! Frequência da força (pêndulo 2)
12 y_1  = 0.05e0          ! Amortecimento (pêndulo 1)
13 y_2  = 0.05e0          ! Amortecimento (pêndulo 2)
```

Os dois pêndulos são idênticos em todos os parâmetros físicos, diferindo apenas nas condições iniciais dos ângulos. Ambos começam com velocidade angular nula, mas o segundo pêndulo possui um pequeno desvio inicial de 0,001 radiano:

```
1 w1(0) = 0
2 w2(0) = 0
3 th1(0) = 1.0
4 th2(0) = 1.0 + 0.001
```

Integração pelo método de Euler-Cromer

O laço principal da simulação executa `imax` iterações, aplicando o método de Euler-Cromer para calcular a evolução temporal dos ângulos e velocidades angulares. A cada passo, são calculadas as forças efetivas sobre os dois pêndulos, compostas pelo amortecimento e pela força externa periódica:

```
1 F1 = - y_1*w1(i) + F0_1*sin(ome_1*i*dt)
2 F2 = - y_2*w2(i) + F0_2*sin(ome_2*i*dt)
```

Em seguida, atualizam-se as velocidades e ângulos segundo:

```
1 w1(i+1) = w1(i) - (g/r1)*sin(th1(i))*dt + F1*dt
2 w2(i+1) = w2(i) - (g/r1)*sin(th2(i))*dt + F2*dt
3
4 th1(i+1) = th1(i) + w1(i+1)*dt
5 th2(i+1) = th2(i) + w2(i+1)*dt
```

O método de Euler-Cromer utiliza a nova velocidade angular (ω_{i+1}) no cálculo da posição angular (θ_{i+1}), o que resulta em maior estabilidade numérica para sistemas oscilatórios.

Armazenamento dos resultados

Após o cálculo, os resultados são gravados no arquivo `saída-2-12694394.txt`. Antes da gravação, é feita a correção do ângulo θ para garantir que permaneça dentro do intervalo $[0, 2\pi]$, evitando o acúmulo de voltas excedentes. Cada linha do arquivo contém:

- o tempo ($t = i \cdot \Delta t$),
- a velocidade angular e o ângulo do primeiro pêndulo,
- a velocidade angular e o ângulo do segundo pêndulo,
- e a diferença angular $\Delta\theta = \theta_2 - \theta_1$.

O formato de escrita utilizado é:

```
1 7 format(5(F12.6, ', '), F12.6)
```

e o arquivo é encerrado com o comando `close(1)`.

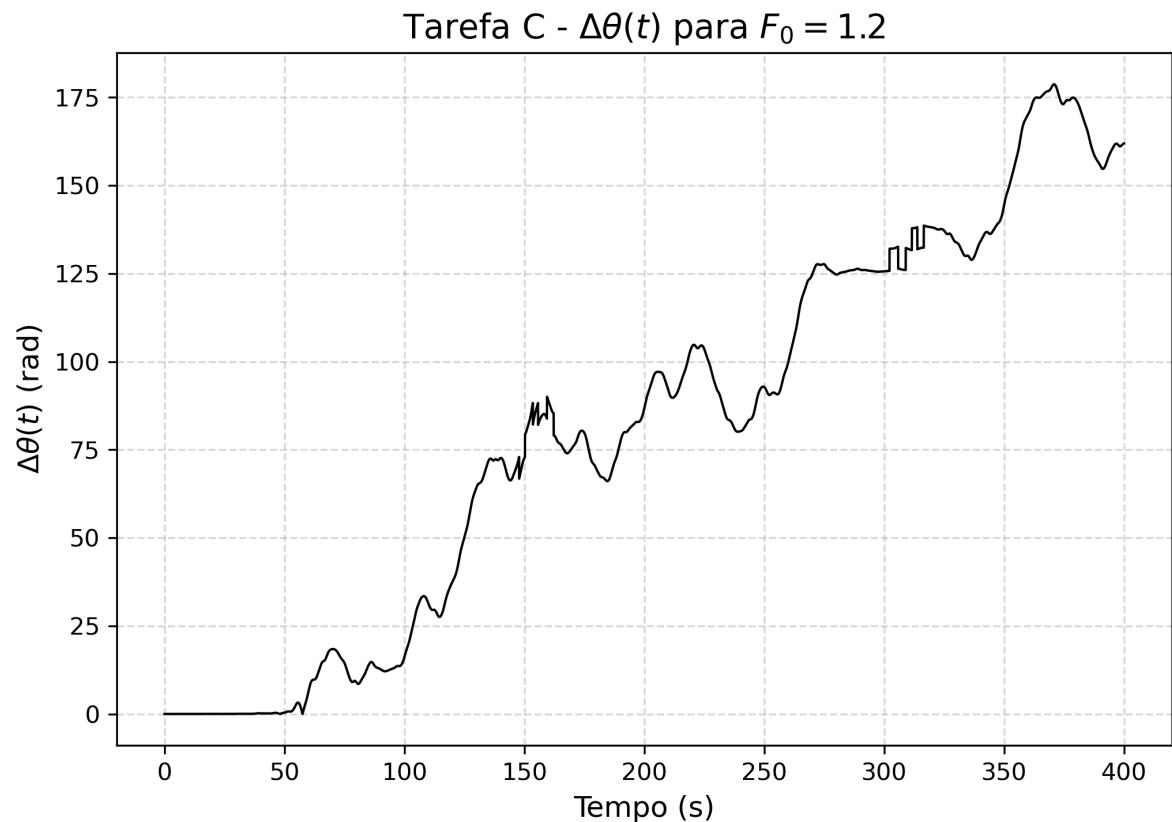
Resumo

Em síntese, o código realiza a simulação de dois pêndulos forçados e amortecidos com condições iniciais ligeiramente diferentes, permitindo analisar a divergência de trajetórias ao longo do tempo. Essa abordagem é típica em estudos de sistemas dinâmicos caóticos, onde pequenas perturbações nas condições iniciais levam a comportamentos macroscopicamente distintos, mesmo sob as mesmas leis de movimento.

RESULTADOS

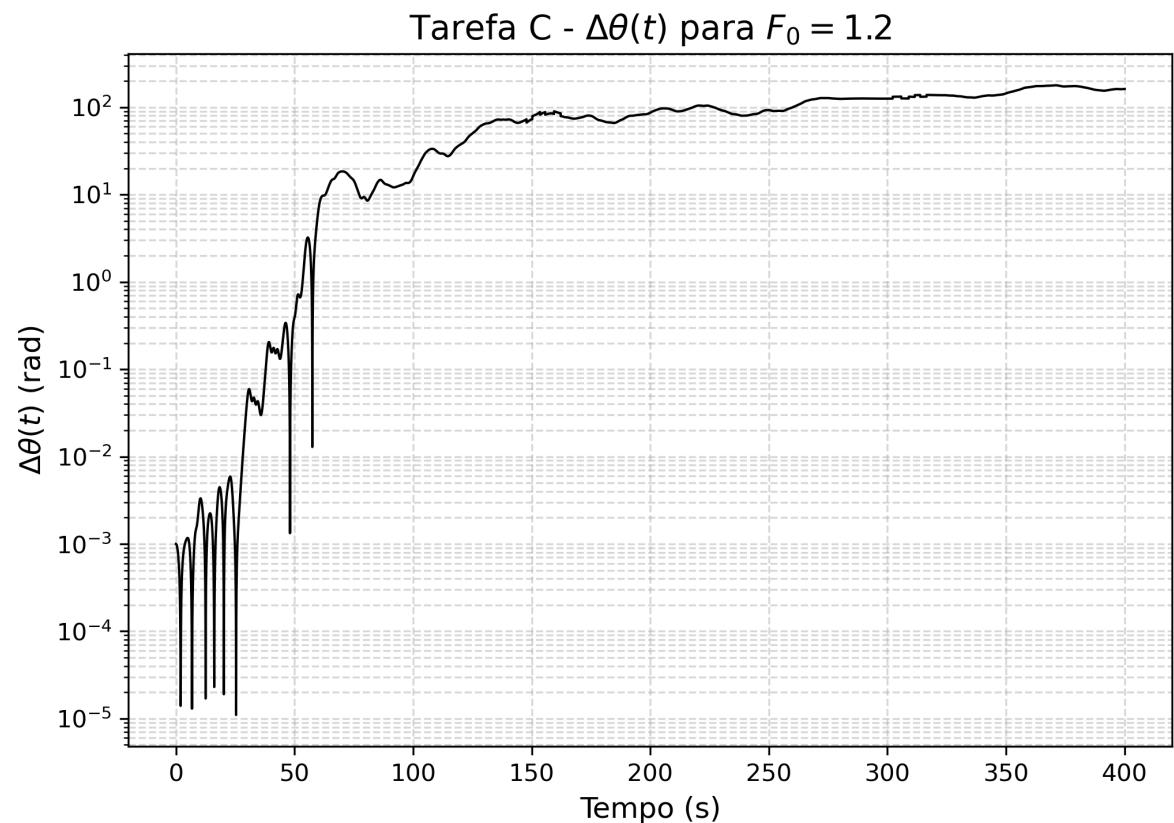
Estimativa do coeficiente de Liapunov: 9.2039e-02

Figura 13 – Resultado obtidos.



Fonte: Compilado pelo Autor.

Figura 14 – Resultado obtidos.

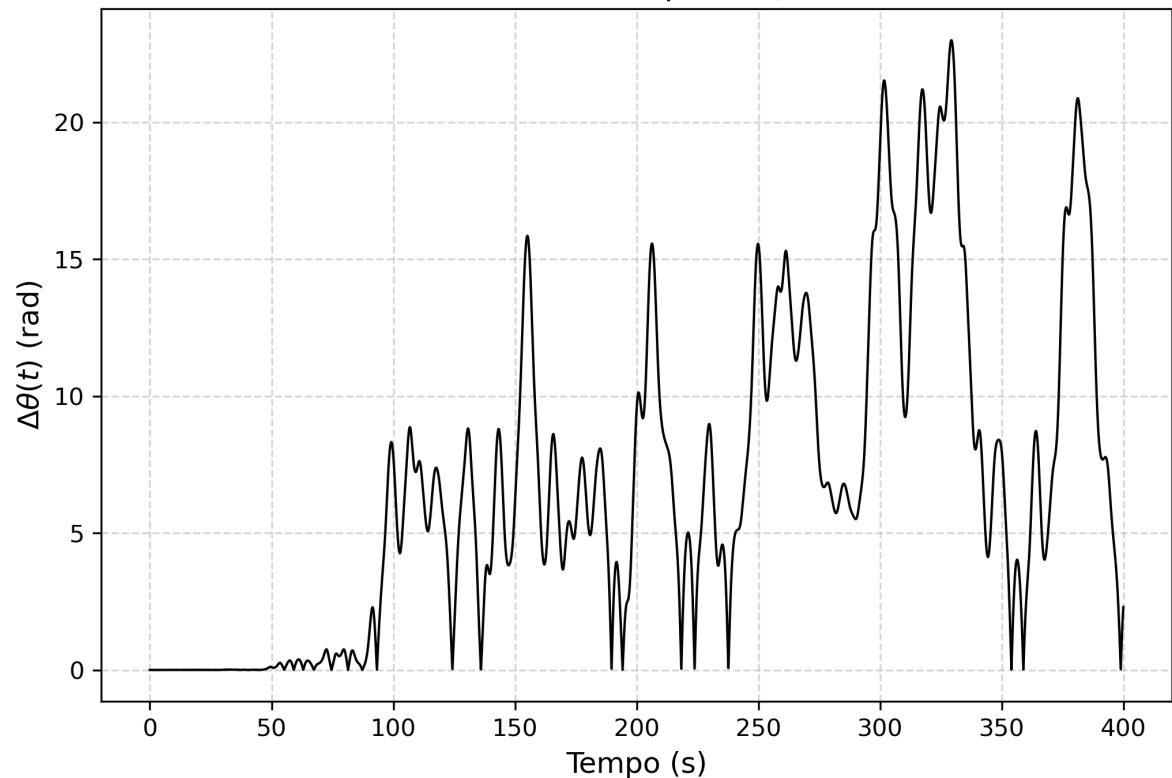


Fonte: Compilado pelo Autor.

Estimativa do coeficiente de Liapunov: 1.0632e-01

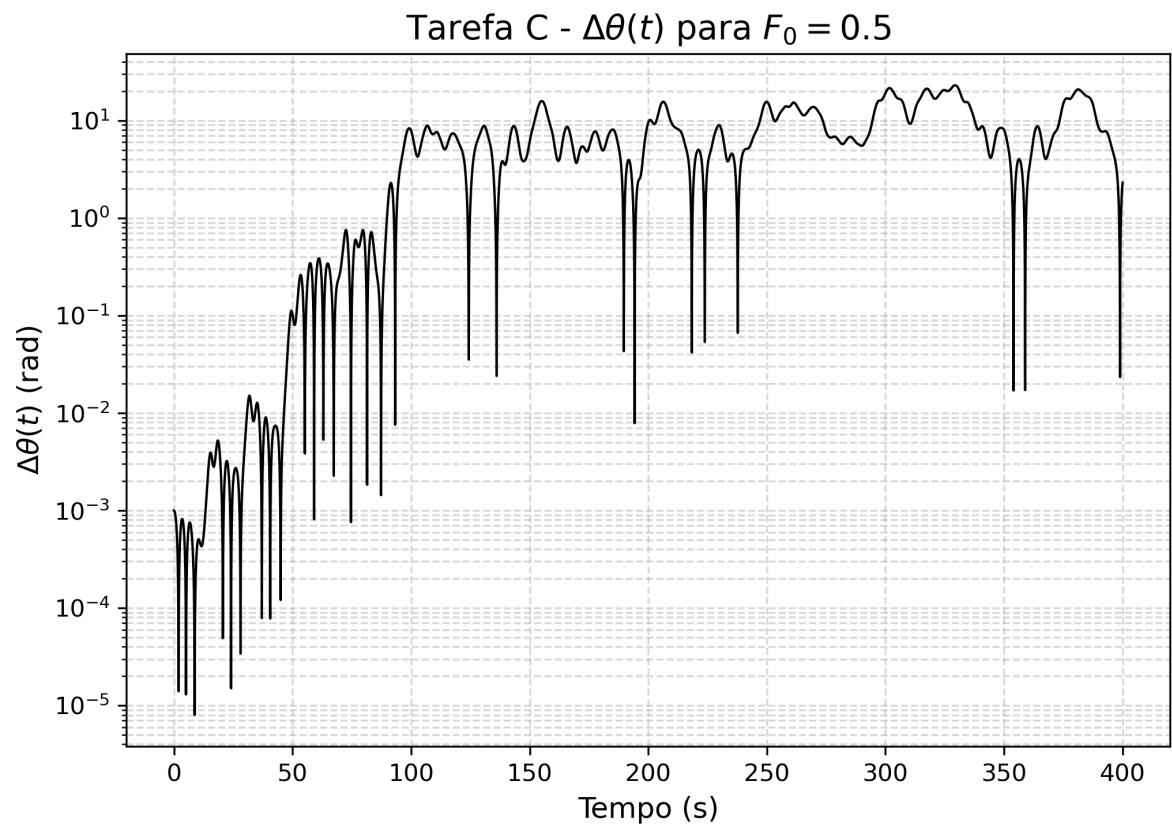
Figura 15 – Resultado obtidos.

Tarefa C - $\Delta\theta(t)$ para $F_0 = 0.5$



Fonte: Compilado pelo Autor.

Figura 16 – Resultado obtidos.



Fonte: Compilado pelo Autor.

TAREFA - D

ENUNCIADO

Figura 17 – Enunciado da Tarefa D.

TAREFA D: Faça os gráficos $\omega(\theta)$ para os casos $F_0 = 0.5$ e $F_0 = 1.2$, para algumas condições iniciais próximas. Compare os resultados obtidos.

Conforme você deve ter observado você não obteve no caso caótico algo tão desordenado. Existe regiões do diagrama que nunca foram visitadas.

Uma maneira mais efetiva de se visualizar a "estrutura" existente no movimento caótico é a realização de uma **secção de Poincaré**. Isto é. só graficamos $\omega(\theta)$ quando $\Omega t = n\pi$ (n inteiro).

Fonte: Compilado pelo Autor.

CÓDIGO

Figura 18 – Função principal do código.

```
1 program main
2     implicit real*8 (a-h,o-z)
3     call poicare()
4 end program main
```

Fonte: Compilado pelo Autor.

Figura 19 – Implementação do método de Euler-Cromer.

```
1 subroutine poincare()
2     parameter(imax=1e6)
3     implicit real*8 (a-h,o-z)
4     dimension w(0:imax), th(0:imax)
5     dimension ival(0:imax)
6
7 C Constantes
8     pi = acos(-1d0)
9     g = 9.81d0
10    rl = 9.81d0
11    dt = 0.04d0
12    c1 = 0.05d0 ! Fator de amortecimento (gamma)
13    c2 = 1.2d0 ! F_0
14    c3 = 0.666d0 ! Frequencia angular da força extena
15
16 C Valores iniciais
17     w(0) = 0
18     th(0) = pi/60d0
19     ival(0) = 0
20 C Realiza a simulação
21     open(unit=7,file='saida-2-12694394.txt')
22     write(7,9)
23
24     do i = 0,imax-1
25         rr = c3*i*dt
26         F = -c1*w(i) + c2*sin(rr)
27         w(i+1) = w(i) -(g/r)*sin(th(i))*dt +F*dt
28         th(i+1) = th(i) + w(i+1)*dt
29
30         if (abs(mod(rr, pi)) .LE. 1d-3) then
31             ival(i+1) = 1
32             write(7,8) w(i+1),th(i+1)
33         else
34             ival(i+1) = 0
35         end if
36
37     end do
38
39     close(7)
40     format(F16.8,',',F16.8)
41     format('omega,theta')
```

Fonte: Compilado pelo Autor.

Figura 20 – Implementação do método de Euler-Cromer.

```
1      C           Salva os dados
2      open(unit=1,file='saida-1-12694394.txt')
3          write(1,3)
4          do i = 0,imax-1
5              ! Aqui eu tenho que eu tenho que fazer o trem do angulo
6              if (th(i+1).GT. 2*pi) then
7                  th(i+1) = th(i+1) -2*pi
8              else if (th(i+1) .LT. 0d0) then
9                  th(i+1) = th(i+1) + 2*pi
10             end if
11
12
13
14             write(1,2) ival(i),dt*i, w(i),th(i)
15         end do
16         format(I2,3( , ,F16.8))
17         format('poincare,temp,omega,theta')
18         close(1)
19
20
21
22     end subroutine poincare
```

Fonte: Compilado pelo Autor.

DESCRIÇÃO DO CÓDIGO

O programa `main` tem como função principal chamar a subrotina `poincare`, responsável por simular o movimento de um pêndulo forçado e amortecido por meio do método numérico de *Euler-Cromer*. O objetivo da simulação é gerar o mapa de Poincaré do sistema, permitindo a análise do comportamento dinâmico e a identificação de regimes periódicos ou caóticos.

Toda a parte computacional é implementada dentro da subrotina `poincare`, que realiza tanto a integração temporal das equações de movimento quanto o registro dos dados em arquivos de saída.

Parâmetros e variáveis

Na subrotina `poincare`, é definido o número máximo de iterações `imax = 1×106`, e são criados três vetores principais: `w` (velocidades angulares), `th` (ângulos) e `ival` (indicador lógico que identifica os pontos de Poincaré).

As constantes e parâmetros físicos do sistema são definidos como:

```
1 pi = acos(-1d0)    ! Valor de pi
```

```

2 g = 9.81d0      ! Aceleração da gravidade
3 rl = 9.81d0     ! Comprimento do pêndulo
4 dt = 0.04d0     ! Passo temporal
5 c1 = 0.05d0     ! Coeficiente de amortecimento (gamma)
6 c2 = 1.2d0      ! Amplitude da força externa (F0)
7 c3 = 0.666d0    ! Frequência angular da força externa

```

As condições iniciais são:

```

1 w(0) = 0
2 th(0) = pi/60d0
3 ival(0) = 0

```

Esses valores definem um pêndulo inicialmente quase vertical e sem velocidade angular, sujeito a uma força externa periódica e a um amortecimento viscoso.

Integração pelo método de Euler-Cromer

A simulação é realizada dentro de um laço que percorre todas as iterações de $i = 0$ até $imax - 1$. Em cada passo, calcula-se a força total atuante sobre o pêndulo, composta pelo termo de amortecimento e pela força externa periódica:

```

1 rr = c3*i*dt
2 F = -c1*w(i) + c2*sin(rr)

```

Em seguida, as equações diferenciais do sistema são integradas utilizando o método de Euler-Cromer:

```

1 w(i+1) = w(i) - (g/r1)*sin(th(i))*dt + F*dt
2 th(i+1) = th(i) + w(i+1)*dt

```

Esse método utiliza o valor atualizado da velocidade angular para calcular a posição, conferindo maior estabilidade numérica, especialmente para sistemas oscilatórios.

Construção do mapa de Poincaré

Durante a simulação, o programa verifica se o argumento da força $c_3 i dt$ é aproximadamente um múltiplo de π . Essa condição identifica os instantes em que o sistema completa um ciclo da força externa. Quando isso ocorre, o valor da velocidade angular ω e do ângulo θ são registrados no arquivo `saída-2-12694394.txt`, formando o conjunto de pontos do mapa de Poincaré:

```

1 if (abs(mod(rr, pi)) .LE. 1d-3) then
2   ival(i+1) = 1
3   write(7,8) w(i+1), th(i+1)

```

```
4 | else
5 |     ival(i+1) = 0
6 | end if
```

Esses pontos permitem analisar visualmente a evolução do sistema no espaço de fases sob o regime forçado, revelando comportamentos periódicos, quase-periódicos ou caóticos.

Armazenamento dos resultados completos

Além do mapa de Poincaré, o código também grava os dados completos da simulação no arquivo `saída-1-12694394.txt`. Antes de salvar, o ângulo é ajustado para permanecer dentro do intervalo $[0, 2\pi]$, garantindo a consistência do gráfico de fase. O arquivo contém, para cada iteração:

- o identificador de ponto de Poincaré (`ival(i)`),
- o tempo ($t = i \cdot \Delta t$),
- a velocidade angular (ω),
- e o ângulo (θ).

Os dados são escritos no formato:

```
1 | 2 format(I2,3(‘,’,F16.8))
```

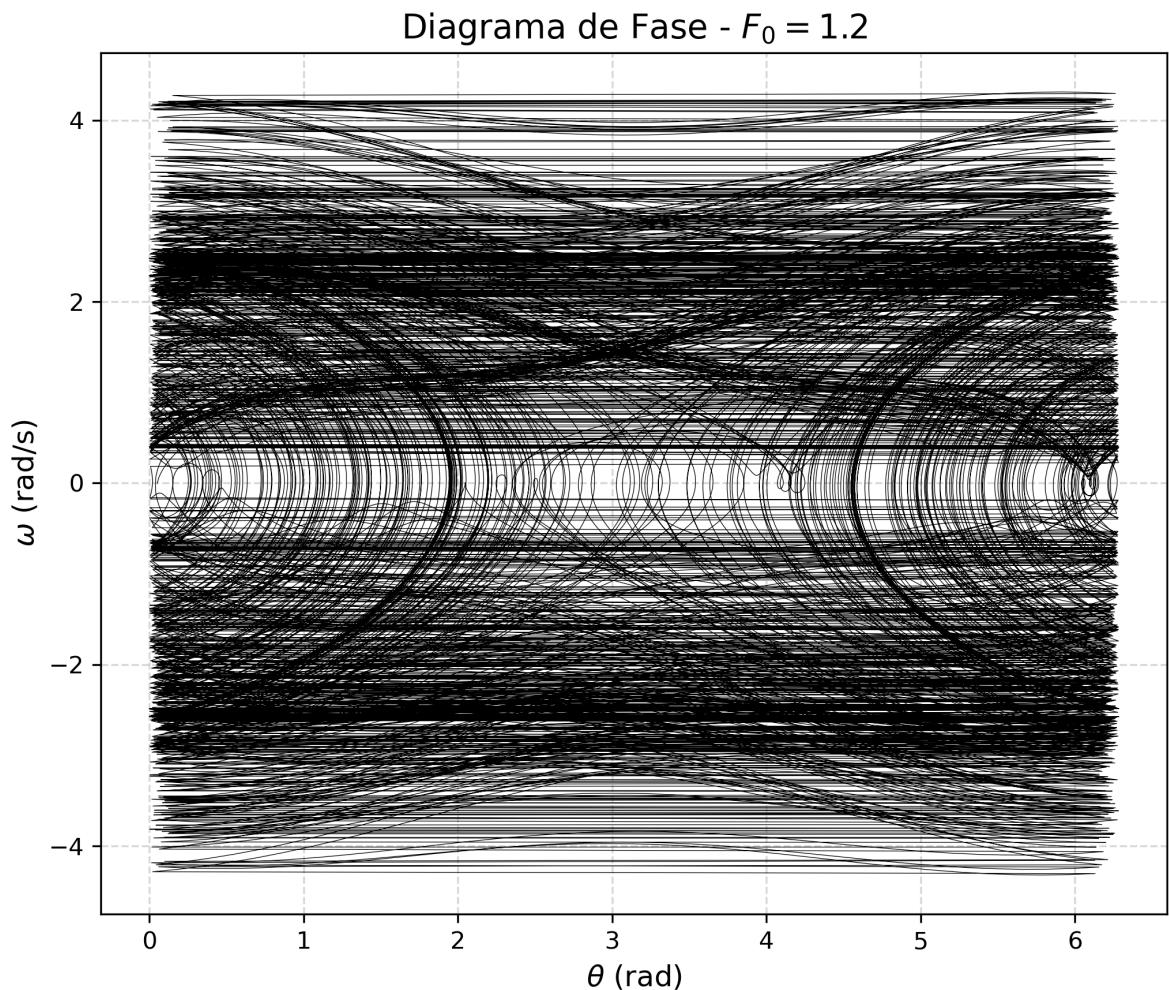
e o arquivo é finalizado com o comando `close(1)`.

Resumo

Em resumo, o código implementa a simulação de um pêndulo amortecido e forçado, aplicando o método de Euler-Cromer para integração numérica e gerando o mapa de Poincaré do sistema. A análise dos pontos registrados permite identificar regimes de periodicidade, transições para o caos e comportamento determinístico complexo, característico de sistemas não lineares forçados.

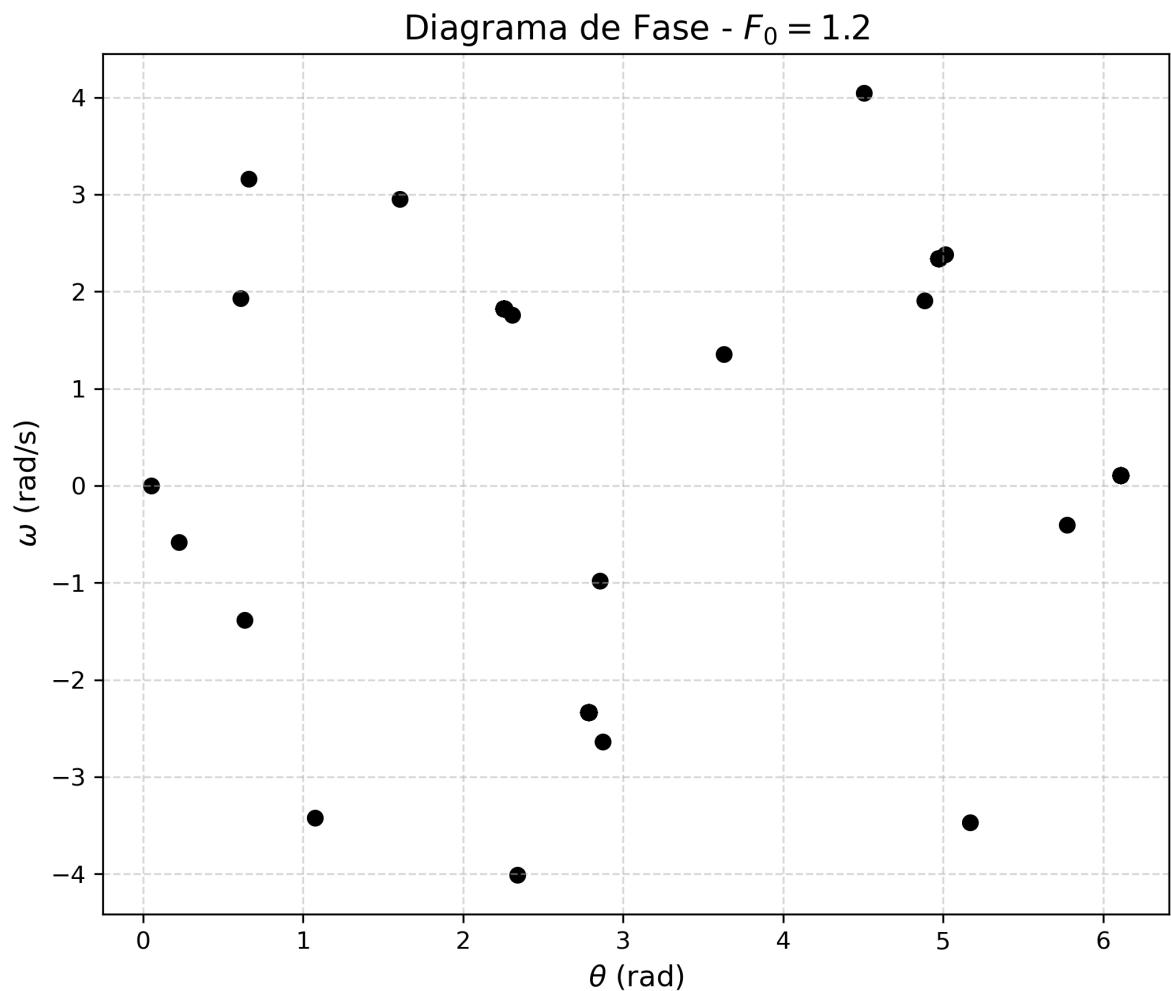
RESULTADOS

Figura 21 – Resultado obtidos para $F_0 = 1.2$



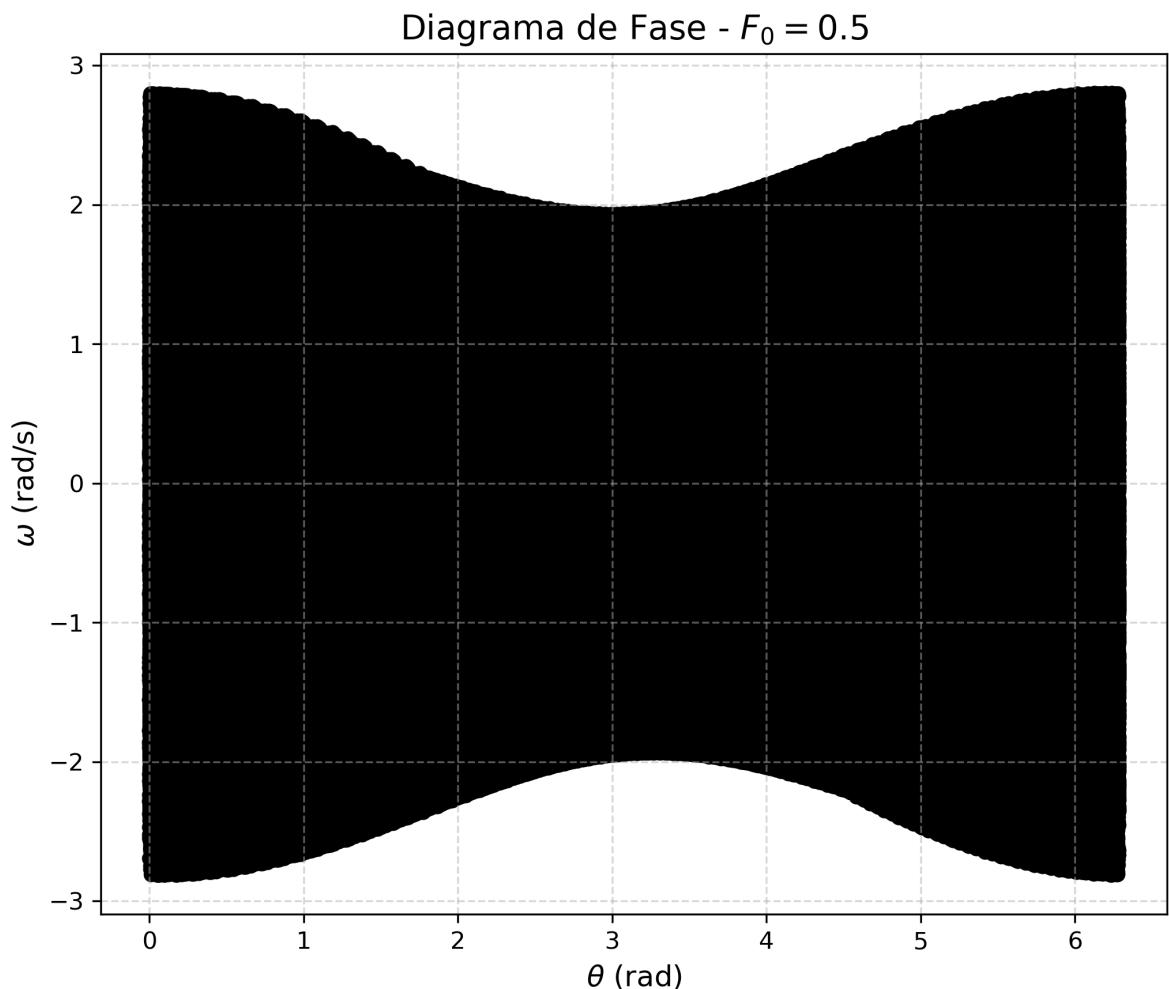
Fonte: Compilado pelo Autor.

Figura 22 – Resultado obtidos para $F_0 = 1.2$ com filtro de Poincaré



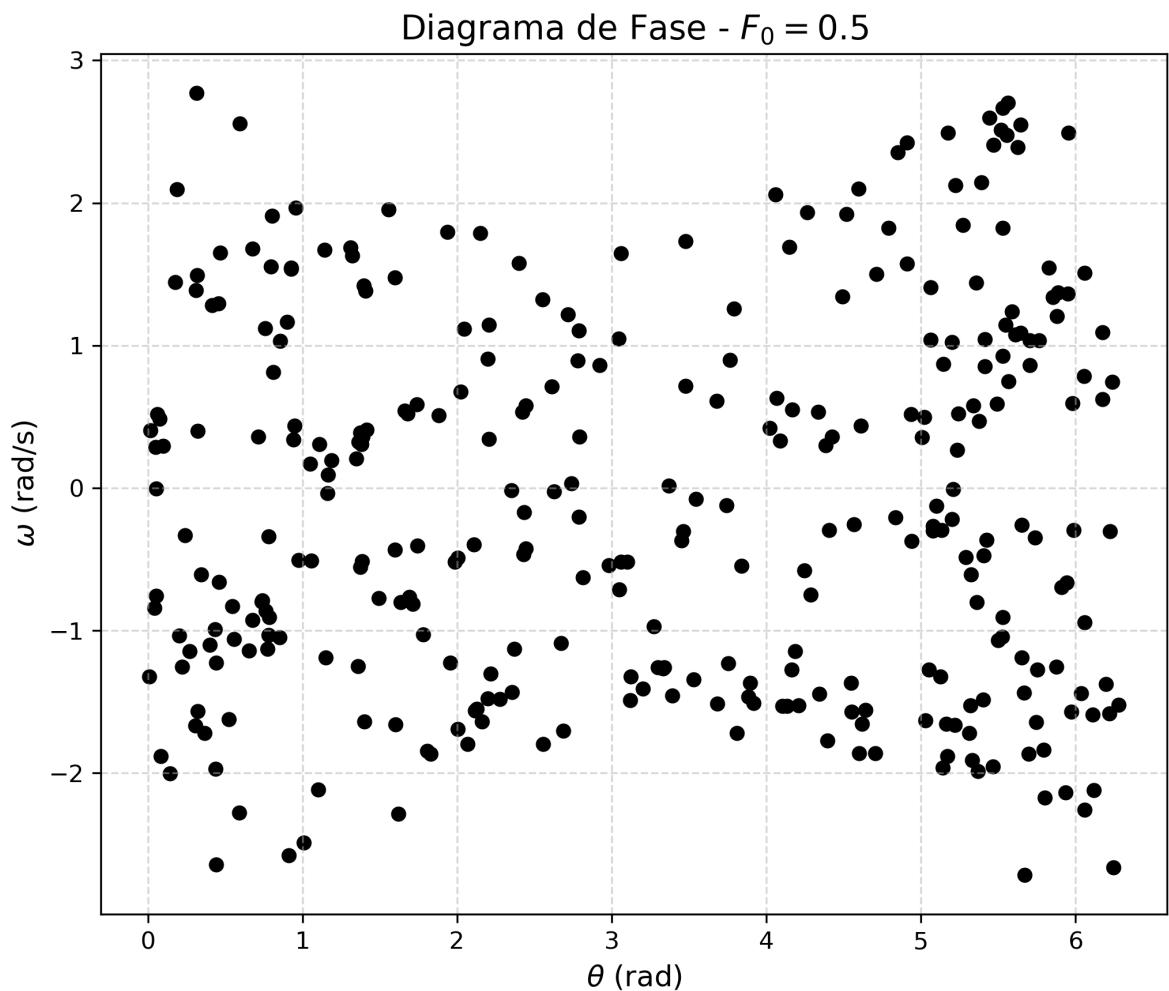
Fonte: Compilado pelo Autor.

Figura 23 – Resultado obtidos para $F_0 = 0.5$



Fonte: Compilado pelo Autor.

Figura 24 – Resultado obtidos para $F_0 = 0.5$ com filtro de Poincaré



Fonte: Compilado pelo Autor.

TAREFA - E

ENUNCIADO

Figura 25 – Enunciado da Tarefa E.

TAREFA E: Faça o gráfico de $\omega(\theta)$ na secção de Poincaré $\Omega t = n\pi$, que no presente caso deve ser traduzido numericamente por $|t - n\pi/\Omega| < \Delta t/2$. O gráfico deve ser feito pra o caso $F_0 = 0.5$ e $F_1 = 1.2$. A Figura que você obtem é o "R.G." do movimento caótico em questão. Varie ligeiramente as condições iniciais e verifique que a figura fica inalterada, o que mostra a "universalidade" do seu caos. Na realidade a figura que você obteve não é contínua e define um fractal. O estudo de fractais e caos estará então intimamente ligado. A figura que dá o "R.G." do caos e que você obteve é chamada de "atrator estranho". Repare que no caso determinístico o atrator estranho é um ponto. ‘

Fonte: Compilado pelo Autor.

CÓDIGO

Figura 26 – Função principal do código.

```
1 program main
2     implicit real*8 (a-h,o-z)
3     call poincare_secao()
4 end program main
```

Fonte: Compilado pelo Autor.

Figura 27 – Implementação do método de Euler-Cromer.

```
1 subroutine poincare_secao()
2     parameter(imax=1e6)
3     implicit real*8 (a-h,o-z)
4     dimension w(0:imax), th(0:imax)
5
6     pi = acos(-1d0)
7     g = 9.81d0
8     rl = 9.81d0
9     dt = 0.04d0
10    c1 = 0.05d0
11    c2 = 1.2d0
12    c3 = 0.666d0
13
14    w(0) = 0.0d0
15    th(0) = pi/60d0
16
17
18    open(unit=10,file='saida-1-12694394.txt')
19    write(10,9)
20
21    do i = 0, imax-1
22        t = i*dt
23        rr = c3*t
24        F = -c1*w(i) + c2*sin(rr)
25        w(i+1) = w(i) - (g/r1)*sin(th(i))*dt + F*dt
26        th(i+1) = th(i) + w(i+1)*dt
27
28        if (th(i+1) .GT. 2*pi) then
29            th(i+1) = th(i+1) - 2*pi
30        else if (th(i+1) .LT. 0d0) then
31            th(i+1) = th(i+1) + 2*pi
32        end if
33
34        if (abs(mod(c3*t, pi)) .LT. (c3*dt/2d0)) then
35            write(10,8) w(i+1), th(i+1)
36        end if
37    end do
38
39    close(10)
40    format(F16.8, ', ', F16.8)
41    format('omega,theta')
42    return
43    end
```

Fonte: Compilado pelo Autor.

DESCRÍÇÃO DO CÓDIGO

O programa `main` tem como objetivo chamar a subrotina `poincare_secao`, responsável por calcular a seção de Poincaré de um pêndulo amortecido e forçado,

utilizando o método numérico de *Euler-Cromer*. A simulação permite analisar o comportamento dinâmico do sistema em regimes periódicos e caóticos, registrando apenas os pontos específicos que compõem a seção de Poincaré.

Toda a integração numérica e o armazenamento dos resultados são realizados dentro da subrotina `poincare_secao`.

Parâmetros e variáveis

Na subrotina, define-se o número máximo de iterações `imax = 1×106` e são criados dois vetores principais: `w` e `th`, que armazenam, respectivamente, a velocidade angular (ω) e o ângulo (θ) do pêndulo ao longo do tempo.

As constantes físicas e parâmetros da força externa são definidos como:

```
1 pi = acos(-1d0)      ! Valor de pi
2 g   = 9.81d0          ! Aceleração da gravidade
3 rl  = 9.81d0          ! Comprimento do pêndulo
4 dt  = 0.04d0          ! Passo temporal
5 c1  = 0.05d0          ! Coeficiente de amortecimento (gamma)
6 c2  = 1.2d0            ! Amplitude da força externa (F_0)
7 c3  = 0.666d0          ! Frequência angular da força externa
```

As condições iniciais do sistema são:

```
1 w(0)  = 0.0d0
2 th(0) = pi/60d0
```

Esses valores correspondem a um pêndulo inicialmente em repouso, com um pequeno deslocamento angular inicial, sujeito à ação de uma força externa periódica e a um termo de amortecimento viscoso.

Integração pelo método de Euler-Cromer

A integração numérica é realizada dentro de um laço que percorre de $i = 0$ até $imax - 1$. Em cada iteração, calcula-se a força resultante sobre o pêndulo, composta pelos termos de amortecimento e força externa:

```
1 t    = i*dt
2 rr  = c3*t
3 F   = -c1*w(i) + c2*sin(rr)
```

As equações diferenciais são integradas pelo método de *Euler-Cromer*, que atualiza primeiro a velocidade angular e, em seguida, o ângulo:

```

1 w(i+1) = w(i) - (g/r1)*sin(th(i))*dt + F*dt
2 th(i+1) = th(i) + w(i+1)*dt

```

Esse método é mais estável para sistemas oscilatórios, pois utiliza o valor atualizado da velocidade para calcular a posição.

Correção angular e critério da seção de Poincaré

Após a atualização, o ângulo é ajustado para permanecer dentro do intervalo $[0, 2\pi]$, evitando que valores excedam esse limite devido à integração acumulada:

```

1 if (th(i+1) .GT. 2*pi) then
2   th(i+1) = th(i+1) - 2*pi
3 else if (th(i+1) .LT. 0d0) then
4   th(i+1) = th(i+1) + 2*pi
5 end if

```

A cada passo, o código verifica se o instante atual t corresponde aproximadamente a um múltiplo de π/ω_{fora} , ou seja, se o sistema se encontra na mesma fase da força externa. Quando essa condição é satisfeita, o par (ω, θ) é registrado no arquivo de saída, compondo a *seção de Poincaré*:

```

1 if (abs(mod(c3*t, pi)) .LT. (c3*dt/2d0)) then
2   write(10,8) w(i+1), th(i+1)
3 end if

```

Armazenamento dos resultados

Os pontos da seção de Poincaré são salvos no arquivo `saida-1-12694394.txt`, em formato CSV, com as colunas `omega` e `theta`. O formato de escrita utilizado é:

```

1 8 format(F16.8, ', ', F16.8)

```

O arquivo é finalizado com o comando `close(10)` após o término das iterações.

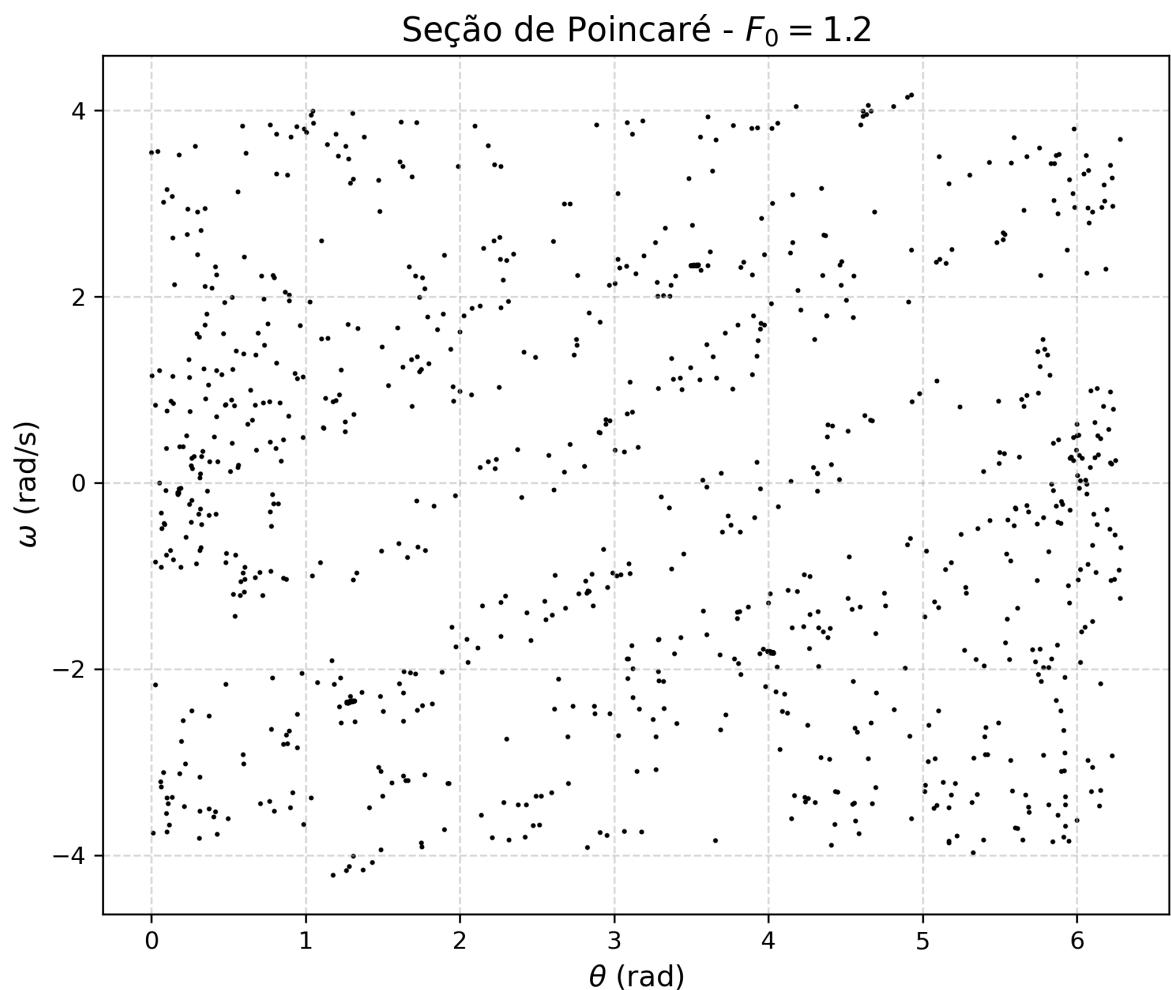
Resumo

Em síntese, o código implementa a simulação de um pêndulo amortecido e forçado, aplicando o método de Euler-Cromer e registrando apenas os pontos pertencentes à seção de Poincaré. Esses pontos representam o estado do sistema em instantes sincronizados com a força externa, permitindo a visualização de órbitas periódicas, quase-periódicas e caóticas no espaço de fases. Esse tipo de análise é

fundamental no estudo de sistemas não lineares e fenômenos caóticos em mecânica clássica.

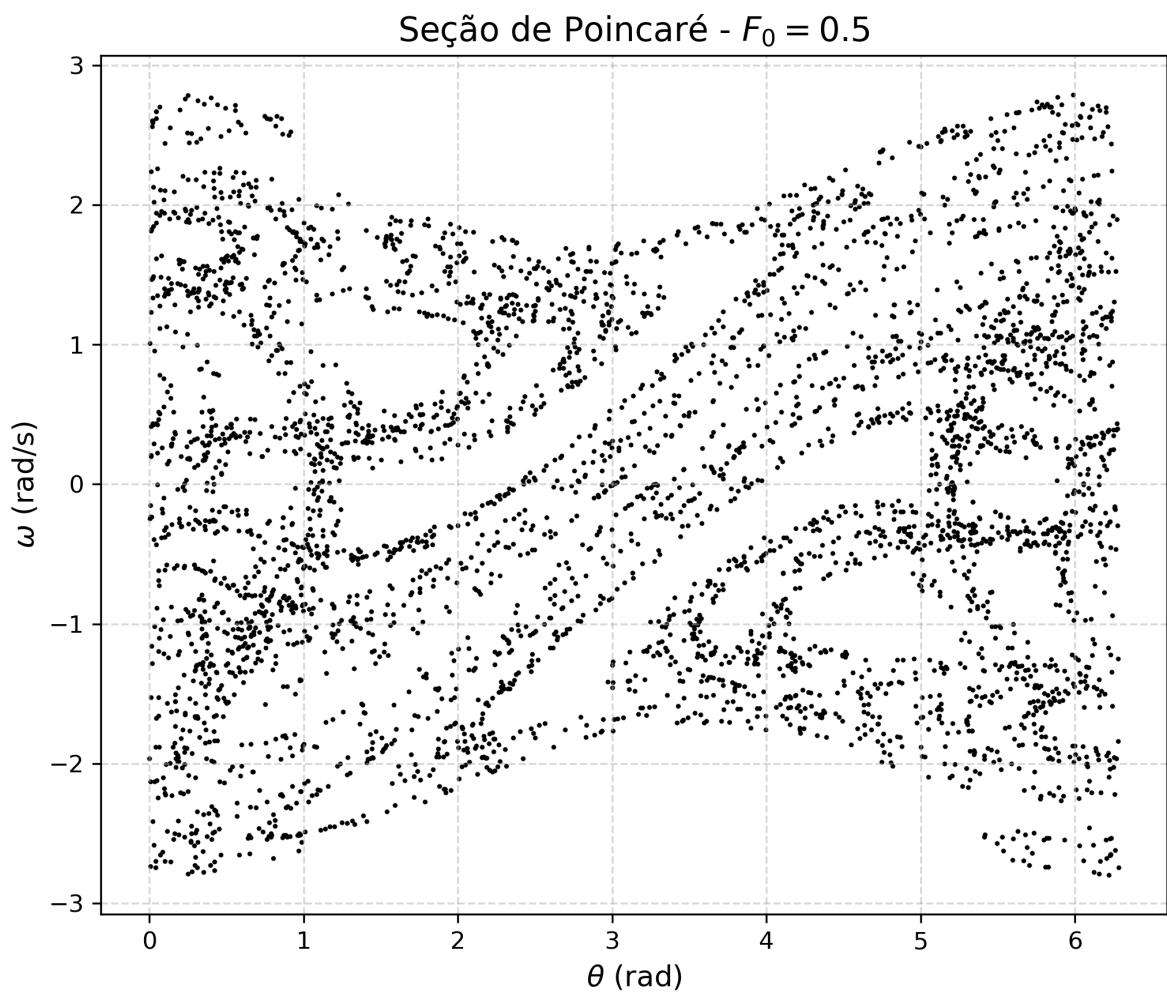
RESULTADOS

Figura 28 – Resultado obtidos para $F_0 = 1.2$



Fonte: Compilado pelo Autor.

Figura 29 – Resultado obtidos para $F_0 = 0.5$



Fonte: Compilado pelo Autor.