



Universidade de São Paulo
Instituto de Física de São Carlos
7600017 - Introdução à Física Computacional
Docente: Francisco Castilho Alcaraz

Projeto 04: Movimento oscilatório

Andressa Colaço
Nº USP: 12610389

São Carlos
2022

Sumário

1	Introdução	2
2	Tarefa A	2
3	Tarefa B	10
3.1	Tarefa B1	10
3.2	Tarefa B2	14
3.3	Tarefa B3	15
3.4	Tarefa B4	17
4	Tarefa C	22
5	Tarefa D	25
6	Tarefa E	32

1 Introdução

O objetivo deste projeto é investigar o movimento oscilatório através do exemplo de um pêndulo em suas diferentes configurações.

As condições iniciais comuns a todos os pêndulos estudados serão: $g = 9.8m/s^2$; comprimento do pêndulo $l = 9.8m$ e massa $m = 1kg$.

2 Tarefa A

Na primeira tarefa, iremos investigar a aproximação das equações do pêndulo simples em um limite de oscilações pequenas ($\sin(\theta) \approx \theta$) através de dois métodos: o de Euler e sua correção, o método de Euler-Cromer.

A equação diferencial que rege o movimento do pêndulo é obtida a partir das leis de Newton e é dada por:

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

Em um limite onde θ é no máximo $\theta = \frac{\pi}{12}$, podemos considerar a simplificação:

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

que, por ser uma equação diferencial linear, se torna mais fácil de resolver. Sua solução é:

$$\theta = \theta_0 \sin(\omega_0 t + \Phi) \quad (3)$$

onde $\omega_0 = \sqrt{g/l}$.

Iremos resolver esta equação numericamente utilizando o método de Euler, que consiste em fazer a aproximação da solução da equação diferencial calculando vários pontos próximos à função que é a solução a partir de uma dada condição inicial, seguindo a seguinte expressão para determinar o ponto de cada iteração:

$$\omega_{i+1} = \omega_i - \frac{g}{l}\theta_i \Delta t \quad (4)$$

e

$$\theta_{i+1} = \theta_i + \omega_i \Delta t \quad (5)$$

sendo $\frac{d\omega}{dt} = -g/l$, $\frac{d\theta}{dt} = \omega$ e Δt uma discretização arbitrária de tempo.

Como veremos após a apresentação do código, esse método possui alguns problemas: para discretizações diferentes, vemos que a solução obtida cresce drasticamente, se afastando da solução com discretização menor. Além disso, a energia mecânica do sistema

também aumenta, algo que não deveria acontecer, já que a energia deve ser conservada neste exemplo. Por isso, podemos utilizar uma pequena correção, o método de Euler-Cromer:

$$\omega_{i+1} = \omega_i - \frac{g}{l}\theta_i\Delta t \quad (6)$$

e

$$\theta_{i+1} = \theta_i + \omega_{i+1}\Delta t \quad (7)$$

Dadas estas considerações, o código escrito foi o dado a seguir:

```

1      PROGRAM pendulo_aprox
2
3      implicit real*8(a-h,o-z)
4
5      parameter(pi = dacos(-1d0))
6      parameter(g = 9.8d0)
7      parameter(p_l = 9.8d0) !l
8      parameter(p_mass = 1d0)
9      parameter(delta_t = 0.1d0)
10
11     11    format(A4, A30, A34)
12     12    format(f6.2, 2e30.11)
13
14     open(10, FILE='saida-a-solucoes-12610389.dat')
15     open(20, FILE='saida-a-energia-12610389.dat')
16
17     t_f = 100
18     N = t_f/delta_t
19     t = 0
20
21     theta_1 = pi/36d0
22     omega_1 = 0d0
23
24     theta_2 = theta_1
25     omega_2 = 0d0
26
27     ratio_gl = (g/p_l)
28
29     write(10, 11) 't', 'theta (Euler)', 'theta (Euler-Cromer)'

```

```

30   write(20, 11) 't', 'Energia (Euler)', 'Energia (Euler-Cromer)'
31
32   do i = 1, N
33     t = delta_t + t
34
35     !Método Euler
36     omega1_np1 = omega_1 - ratio_gl*theta_1*(delta_t)
37     theta1_np1 = theta_1 + omega_1*(delta_t)
38
39     !Método Euler-Cromer
40     omega2_np1 = omega_2 - ratio_gl*theta_2*delta_t
41     theta2_np1 = theta_2 + omega2_np1*delta_t
42
43     !Cálculo da energia
44     e1 = energia(g, p_mass, p_l, omega1_np1, theta1_np1)
45     e2 = energia(g, p_mass, p_l, omega2_np1, theta2_np1)
46
47     write(10, 12) t, theta1_np1, theta2_np1
48     write(20, 12) t, e1, e2
49
50     omega_1 = omega1_np1
51     theta_1 = theta1_np1
52
53     omega_2 = omega2_np1
54     theta_2 = theta2_np1
55   end do
56
57   close(10)
58   close(20)
59
60 end program
61
62 function energia(g, p_mass, p_l, omega, theta)
63 implicit real *8(a-h, o-z)
64
65 energia = 0.5d0*p_mass*omega**2+(1d0/2d0*p_l)*p_mass*g*theta**2
66
67 return
68 end function

```

O código é estruturado de forma com a qual começa-se declarando os parâmetros que serão importantes para a execução e que não serão alterados (com exceção de δt), tais como π , g , p_l (l) e p_{mass} (m). Em sequência, são especificadas as formatações dos arquivos de saída: um deles conterá todas as informações sobre os dois métodos e os dois seguintes tem o objetivo de guardar apenas as variáveis dos dois métodos para facilitar a criação dos gráficos de θ vs t e de energia vs t .

Após, são definidas as variáveis responsáveis pelo tempo: t_f é o tempo total da simulação e N é o número de iterações feitas para percorrer todas as discretizações. Também se inicializa o contador de tempo t em 0. As variáveis θ_1 e ω_1 se referem ao método de Euler e θ_2 e ω_2 ao método de Euler-Cromer. Ambos iniciam em $\theta_0 = \pi/36\ rad e $\omega_0 = 0$. θ_0 é um ângulo arbitrário, mas deve ser pequeno para que a aproximação $\sin(\theta) \approx \theta$ funcione.$

A próxima etapa é um laço que executa os métodos de Euler e de Euler-Cromer seguindo as expressões descritas no início da seção. Após calcular cada ponto para um instante de tempo qualquer, calcula-se também a energia através da expressão:

$$E = \frac{m\omega^2}{2} + \frac{mg\theta^2}{2l} \quad (8)$$

definida em uma função ao final do código.

Por fim, os dados calculados são escritos em seus respectivos arquivos e prossegue-se até o fim das iterações.

Para visualizar as diferenças nos métodos citados, foram simuladas as discretizações $\Delta t = 0.1s$ e $\Delta t = 0.01s$. Para comparar os métodos, foram criados alguns gráficos: primeiramente, das soluções (θ versus t) obtidas e seus comportamentos, sendo comparados em um mesmo gráfico as soluções dos dois métodos para cada discretização; a comparação entre discretizações para um mesmo método; e, por fim, a energia durante o tempo para os dois métodos na discretização $\Delta t = 0.1s$.

Os primeiros gráficos são as comparações entre métodos para cada discretização:

Figura 1: Soluções obtidas utilizando $\Delta t = 0,01s$.

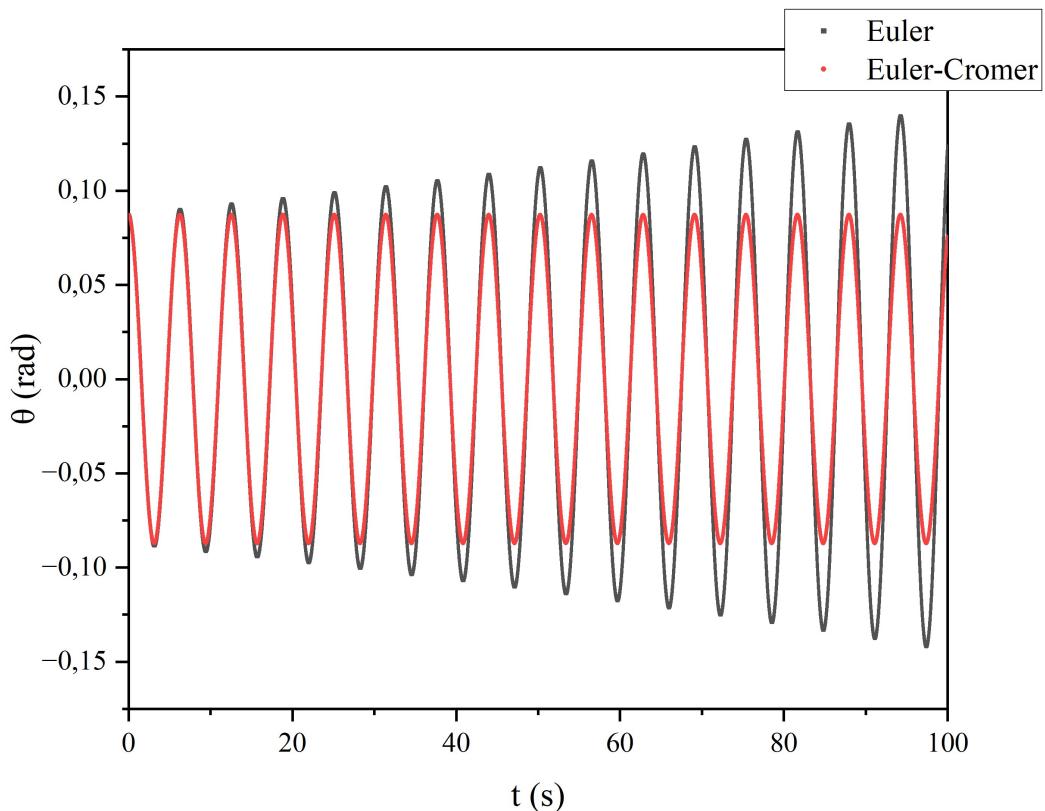
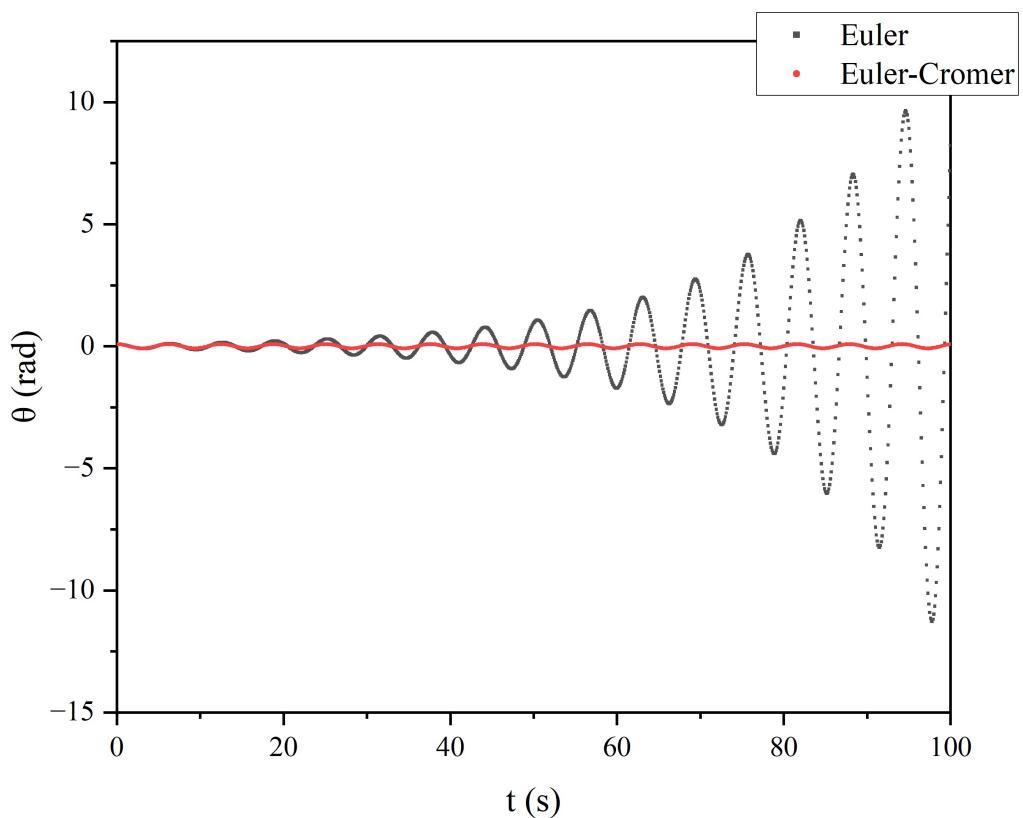


Figura 2: Soluções obtidas utilizando $\Delta t = 0,1s$.



É possível notar que o método de Euler-Cromer produz um resultado oscilatório con-

sistente, enquanto o método de Euler começa a aumentar a amplitude de oscilação enquanto o tempo evolui: este efeito é menor na discretização $\Delta t = 0,01\text{s}$ mas se torna muito significativo com $\Delta t = 0,1\text{s}$, de forma com a qual podemos considerar este método não adequado para simulações com tempos similares ou maiores que utilizem a segunda discretização.

Outra forma de visualizar este problema é comparando em um mesmo gráfico as duas discretizações para cada um dos métodos:

Figura 3: Comparação entre soluções obtidas para o método de Euler em diferentes discretizações.

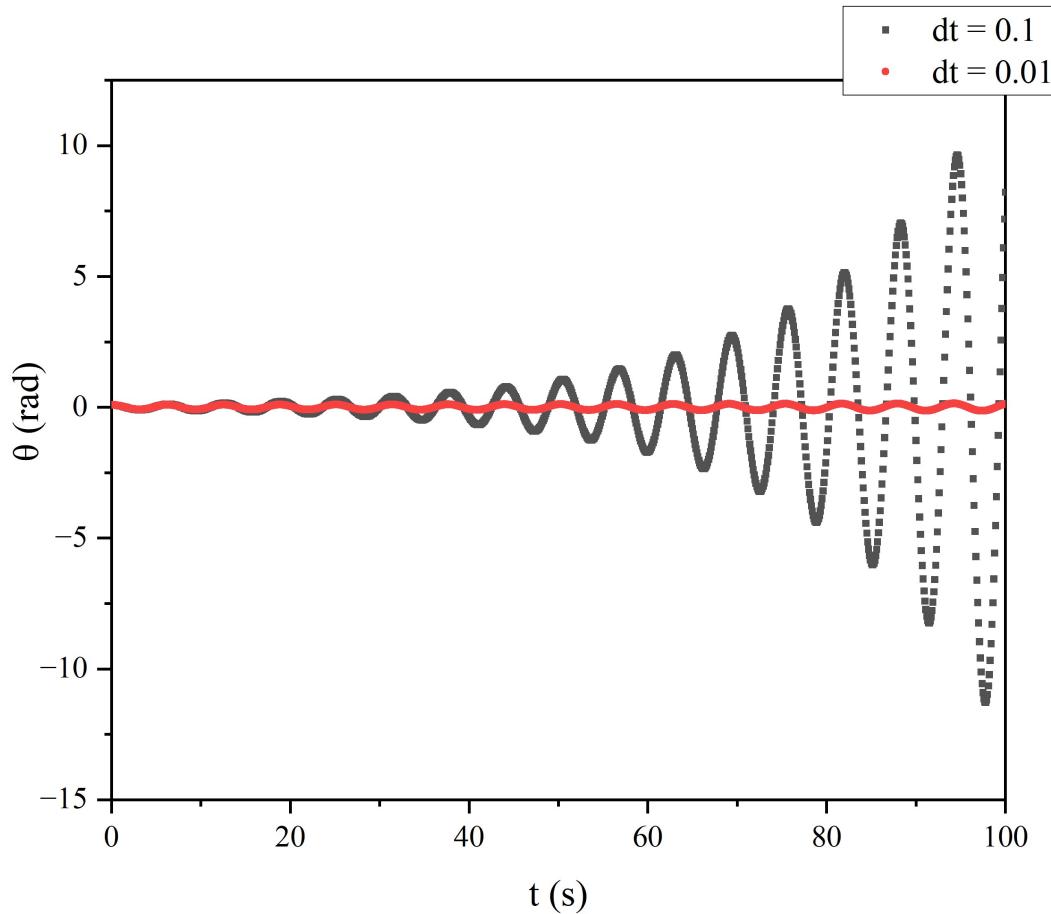
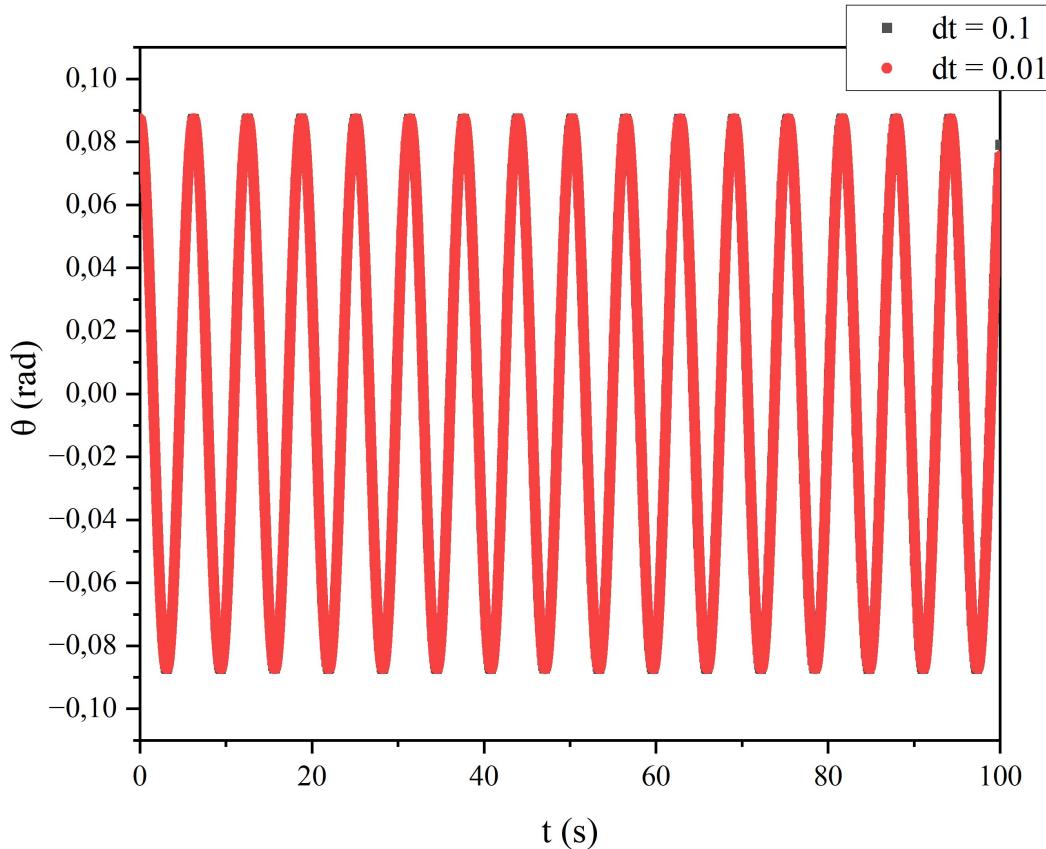


Figura 4: Comparação entre soluções obtidas para o método de Euler-Cromer em diferentes discretizações.



Aqui é ainda mais notável o problema com o método de Euler: enquanto mudar a discretização para este ocasiona divergências enormes nas soluções obtidas, o método de Euler-Cromer se mantém estável em seu resultado nas duas discretizações utilizadas, conforme o gráfico que mostra suas duas soluções perfeitamente sobrepostas.

Por fim, temos a análise da energia mecânica do sistema: como todas as forças envolvidas são conservativas, devemos observar a conservação da energia. Para isto, temos os seguintes gráficos:

Figura 5: Gráfico de energia vs. t para os dois métodos, discretização $\Delta t = 0,01s$.

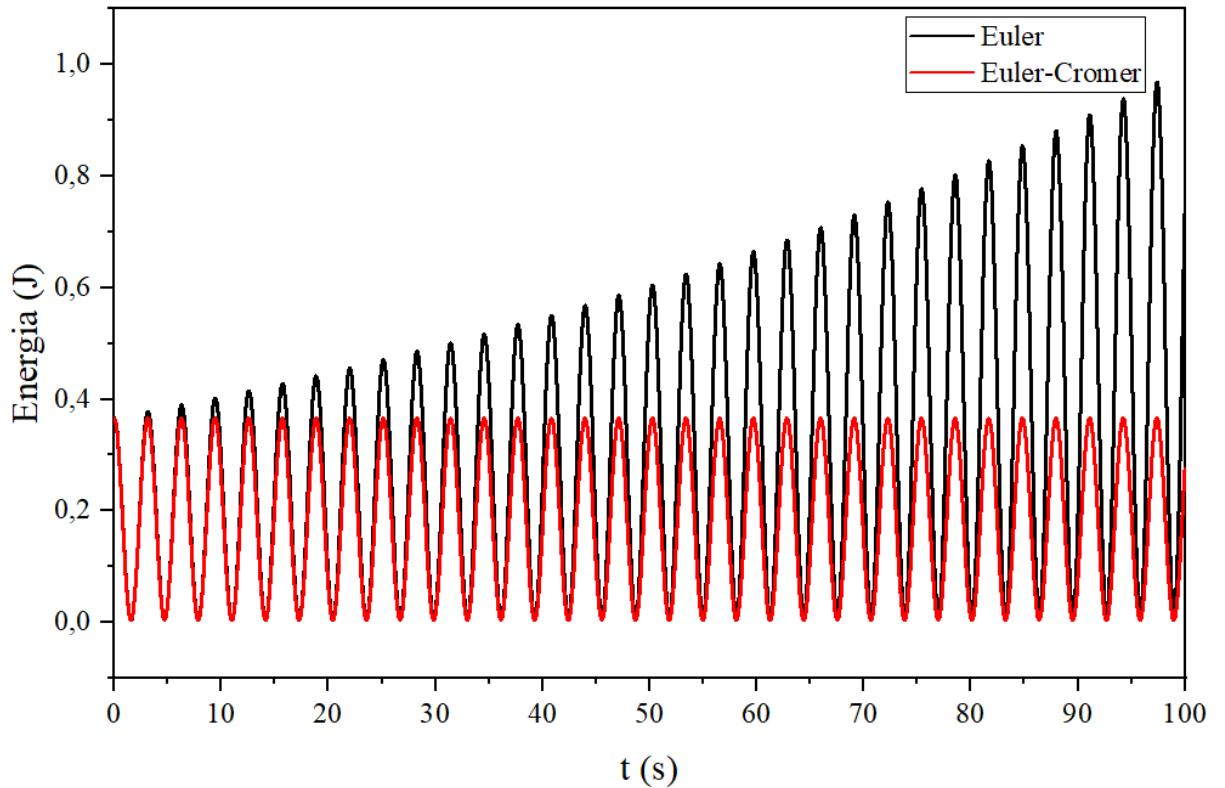
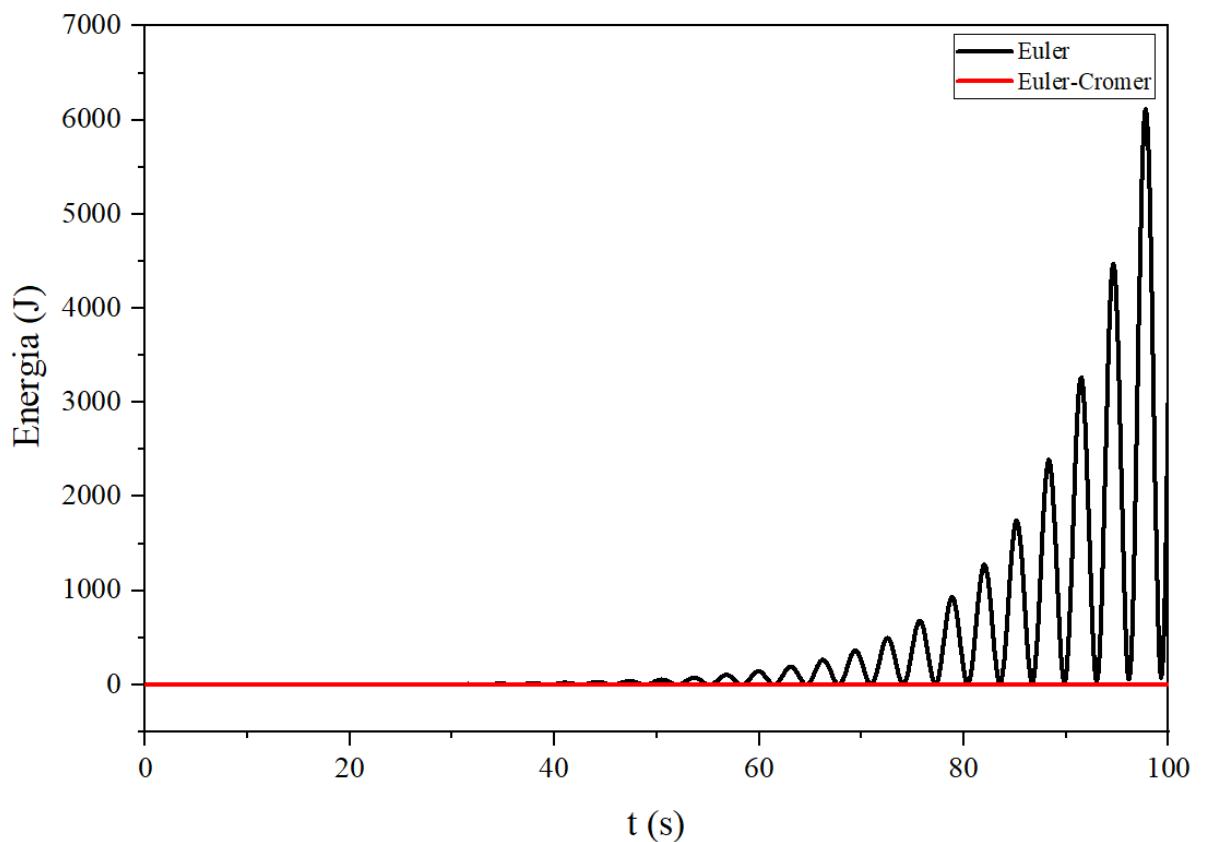


Figura 6: Gráfico de energia vs. t para os dois métodos, discretização $\Delta t = 0,1s$.



Podemos observar que na primeira discretização o método de Euler já apresenta um comportamento de aumento na amplitude da grandeza, sendo que o comportamento oscilatório se deve à aproximação considerada. Mesmo assim, vemos que o método de Euler-Cromer apresenta amplitude constante. No segundo caso, vemos um aumento muito grande na energia dada pelo primeiro método, o qual indica claramente que este não é adequado para o caso estudado. Por conta do aumento tão grande, vemos a energia do segundo como constante e muito próxima de 0, já que é um valor pequeno, como vemos no primeiro gráfico.

Desta maneira, conclui-se que o método de Euler-Cromer é mais adequado para os casos estudados e será o método empregado durante as próximas tarefas, que investigarão o comportamento do pêndulo para casos além do pêndulo simples em pequenos ângulos desta seção.

3 Tarefa B

Nesta seção, iremos lidar com o pêndulo com oscilações em ângulos arbitrários e sujeito a efeitos dissipativos e forças externas oscilatórias. A equação a ser considerada agora é:

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

e os códigos serão ajustados de forma a considerar estes parâmetros.

3.1 Tarefa B1

Nas tarefas B1 e B2 iremos investigar o período do pêndulo e onde suas aproximações são válidas. Teremos um pêndulo amortecido e as condições $\gamma = 0$ e $F_0 = 0$. Por simplicidade das saídas, há apenas um código para as tarefas B1 e B2, que está descrito nesta seção.

Antes de apresentar o código, temos os métodos que serão utilizados para determinar o período do pêndulo simples (agora livre para assumir quaisquer valores de θ_0). Em primeiro lugar, podemos utilizar o método de Euler-Cromer, que retorna valores da função solução $\theta(t)$ e adaptar esta parte do código para gravar a quantidade de tempo necessária para que o pêndulo passe por uma mesma posição, tal qual faríamos em um experimento real, com um cronômetro. Aqui, vamos gravar a duração de tempo de meio período, que equivale a uma mudança de sinal na função velocidade ω e depois, fazer uma média dos valores encontrados para determinar o período do pêndulo.

Em segundo lugar, podemos calcular o período através da integral elíptica:

$$T = \sqrt{\frac{2l}{g}} \int_{-\theta_0}^{\theta_0} \frac{d\theta}{\sqrt{\cos \theta - \cos \theta_0}} \quad (10)$$

Para resolvê-la, vamos empregar o método de Boole estudado no projeto passado, porém, desta vez tem-se que tomar cuidado com os extremos da função, pois ambos tendem ao infinito (porém a integral converge). Para lidar com isso, o método de Boole na realidade calculará o valor da seguinte integral:

$$A_{boole} = \sqrt{\frac{2l}{g}} \int_{-\theta_0+\epsilon}^{\theta_0-\epsilon} \frac{d\theta}{\sqrt{\cos \theta - \cos \theta_0}} \quad (11)$$

Nos intervalos $[-\theta, -\theta + \epsilon]$ e $[\theta, \theta - \epsilon]$ podemos obter a área, que será a mesma para os dois, através da expressão:

$$A_{int} = 2\sqrt{\frac{2l}{g}} \frac{\sqrt{\epsilon}}{\sin \theta_0} \quad (12)$$

Para obter o valor da integral original, basta somar os resultados:

$$A_{total} = A_{boole} + 2A_{int} \quad (13)$$

Por último, sabemos que existem aproximações válidas para valores de θ pequenos. Podemos utilizar a expressão:

$$T \approx 2\pi \sqrt{\frac{l}{g}} \left(1 + \frac{\theta_0^2}{16}\right) \quad (14)$$

Para observar os limites para o qual esta aproximação é boa e se assemelha aos períodos obtidos com os outros dois métodos. Para isso, vamos investigar tanto ângulos iniciais pequenos como outros maiores.

O código desenvolvido foi o seguinte:

```

1      PROGRAM pendulo_aprox
2
3      implicit real*8(a-h,o-z)
4
5      parameter(pi = dacos(-1d0))
6      parameter(theta_0 = (10*pi/180d0))
7      parameter(g = 9.8d0)
8      parameter(p_l = 9.8d0) !l
9      parameter(p_mass = 1d0)
10     parameter(delta_t = 0.01d0)
11
12     t_f = 100d0
13     N = t_f/(delta_t)
14     t = 0d0
15
16     theta = theta_0

```

```

17     omega = 0d0
18     ratio_gl = (g/p_1)
19
20     sum = 0d0
21     icount = 0
22     t_ant = 0d0
23
24     write(*,*) ' inicial = ', theta_0
25     write(*,*) 'Tempo simulado:', t_f
26
27     do i = 1, N
28         t = t + delta_t
29
30         omega_np1 = omega - ratio_gl*dsin(theta)*delta_t
31         theta_np1 = theta + omega_np1*delta_t
32
33         if (omega_np1*omega .lt. 0) then
34             sum = sum + (t-t_ant) !Medindo o meio período
35             t_ant = t
36             icount = icount + 1
37         end if
38
39         omega = omega_np1
40         theta = theta_np1
41     end do
42
43     write(*, *) 'Tn médio =', 2d0*(sum/icount*1d0)
44
45     !Integral elíptica
46     epsilon = 0.001d0
47     a = -abs(theta_0) + epsilon
48     b = abs(theta_0) - epsilon
49     h = (b-a)/N
50     area = 0d0
51
52     do i=0, N-4, 4
53         x0 = a+i*h
54
55         f0 = f(x0, 0, h, theta_0)
56         f1 = f(x0, 1, h, theta_0)

```

```

57         f2 = f(x0, 2, h, theta_0)
58         f3 = f(x0, 3, h, theta_0)
59         f4 = f(x0, 4, h, theta_0)

60
61         area = area +((2*h)/45d0)*(7d0*f0+32d0*f1+
62             $               12d0*f2+32d0*f3+7d0*f4)
63     end do

64
65         area = sqrt(2*p_1/g)*(area+4d0*sqrt(epsilon/dsin(theta_0)))
66         write(*,*) 'Tn (integral) =', area

67
68         !Aprox. para ângulos pequenos
69         per_aprox = 2d0*pi*sqrt(p_1/g)*(1+((theta_0**2)/16d0))
70         write(*,*) 'Tn (aproximação) =', per_aprox

71
72         close(10)

73
74     end program

75
76     function f(x, n, h, theta_0)
77         implicit real*8(a-h,o-z)

78
79         x_n = x+n*h
80         f = 1/sqrt(dcos(x_n)-dcos(theta_0))

81
82         return
83     end function

```

Nele os parâmetros são definidos em primeiro momento, como anteriormente, e as variáveis que cuidam dos contadores de tempo e de θ e ω são ajustadas. Após isso, temos a determinação do período com o auxílio do método de Euler-Cromer, que funciona da seguinte maneira: o método calcula $\theta(t)$ e $\omega(t)$, que indicam como o pêndulo se comporta no tempo e sabemos que, sempre que o pêndulo completa meio período, o sinal de sua velocidade muda (ao atingir a altura ou ângulo máximo). Após de calcular o valor de cada variável correspondente a um instante de tempo, verificamos se ω mudou de sinal. Se tiver mudado, incrementamos uma variável de soma inicializada em 0 antes de iniciar a contagem de tempo com o intervalo de tempo encontrado para o meio período, que será o t atual menos o t onde ocorreu a última mudança de sinal, ou seja, o início deste meio período. Também mantemos uma variável que armazena a quantidade de meios períodos guardados, que servirá para calcular a média através da soma final. Quando acabam as

iterações, esta é obtida e multiplicamos o valor por 2, já que calculamos apenas meio período.

A seguir no código, temos a implementação do cálculo da integral elíptica. Definimos o valor de $\epsilon = 0.001$ e calculamos os limites superior b e inferior a da integral. Em sequência, definimos o valor do incremento h dos intervalos e a integral é calculada através de um loop que percorre estes intervalos em um passo $N = 4$. É definido o valor de x_0 do intervalo e calculados os valores da função nos pontos necessários para o método de Boole. Após, a área é incrementada conforme a fórmula do método. Ao finalizar as iterações, utilizamos a equação (13) que engloba os intervalos desconsiderados anteriormente para saber o valor do período obtido, o qual é impresso também na tela.

Por fim, a aproximação dada na equação (14) é calculada e impressa na tela.

A saída é dada conforme o exemplo onde $\theta = \pi/4$:

Figura 7: Exemplo de saída da tarefa B1.

```

θ inicial = 0.78539816339744828
Tempo simulado: 100.000000000000000
Tn médio = 6.534666666675493
Tn (integral) = 6.5343275646678602
Tn (aproximação) = 6.5254218437444287

```

Testando o código para $\theta = 45^\circ$, $\theta = 60^\circ$ e $\theta = 75^\circ$, obtém-se os seguintes resultados:

θ_0	T - Numérico	T - Integral elíptica	T - Aproximação
45°	6.534666666675493	6.5343275646678602	6.5254218437444287
60°	6.7434482758629732	6.7429924603371267	6.7138280388504166
75°	7.0307142857152467	7.0306237936886360	6.9560645754152590

Podemos ver que o método numérico e o método da integral elíptica concordam até a terceira casa decimal, possivelmente causado pelos erros contidos na determinação do meio período através do método de Euler-Cromer. A aproximação, no entanto, começa a divergir na segunda casa para os dois primeiros ângulos e diverge ainda mais no caso do maior ângulo.

3.2 Tarefa B2

Utilizando o mesmo código da seção B1, nas mesmas condições ($\gamma = 0$, $F_0 = 0$) mas para ângulos pequenos (menores que 15°), podemos notar que de fato, a aproximação se torna muito mais precisa em comparação com os outros dois métodos. Na tabela a seguir, se encontram os períodos calculados através dos três métodos para os valores $\theta_0 = 2^\circ$, $\theta_0 = 5^\circ$ e $\theta = 10^\circ$:

θ_0	T - Numérico	T - Integral elíptica	T - Aproximação
2°	6.2838709677427689	6.2813638921156336	6.2836637991036639
5°	6.2864516129040613	6.2855985306422868	6.2861758817050779
10°	6.2954838709685816	6.2949653769418816	6.2951476052815529

Podemos ver claramente que este comportamento é descrito de acordo com a expressão (14), sendo que a aproximação concorda até a terceira casa decimal com os outros métodos, que no caso anterior também concordavam neste limite.

3.3 Tarefa B3

Na seção 03 da tarefa B, é inserida uma força resistiva ao movimento do pêndulo, proporcional a $\gamma = 0,5$. No entanto, mantemos $F_0 = 0$. O objetivo será obter um gráfico θ versus t , onde poderemos observar como a força de atrito influencia no movimento e identificar o regime do amortecimento causado por ela com base no formato do gráfico encontrado. Iremos utilizar $\theta_0 = 60^\circ$ e um tempo total de simulação de $t = 100s$.

O código correspondente se encontra a seguir. Sua lógica é a mesma descrita para os outros problemas, apenas foi adicionado o termo $-\gamma\omega$ ao método numérico e foram retiradas as partes que não são relevantes para a construção do gráfico:

```

1      PROGRAM pendulo_amortecido
2
3      implicit real*8(a-h,o-z)
4
5      parameter(pi = dacos(-1d0))
6      parameter(theta_0 = (60*pi/180d0))
7      parameter(g = 9.8d0)
8      parameter(p_l = 9.8d0) !l
9      parameter(p_mass = 1d0)
10     parameter(delta_t = 0.01d0)
11
12    11    format(A4, A16)
13    12    format(f6.2, e20.11)
14
15    open(10, FILE='saida-b3-12610389.dat')
16
17    gamma = 0.5d0
18
19    t_f = 100d0
20    N = t_f/(delta_t)
```

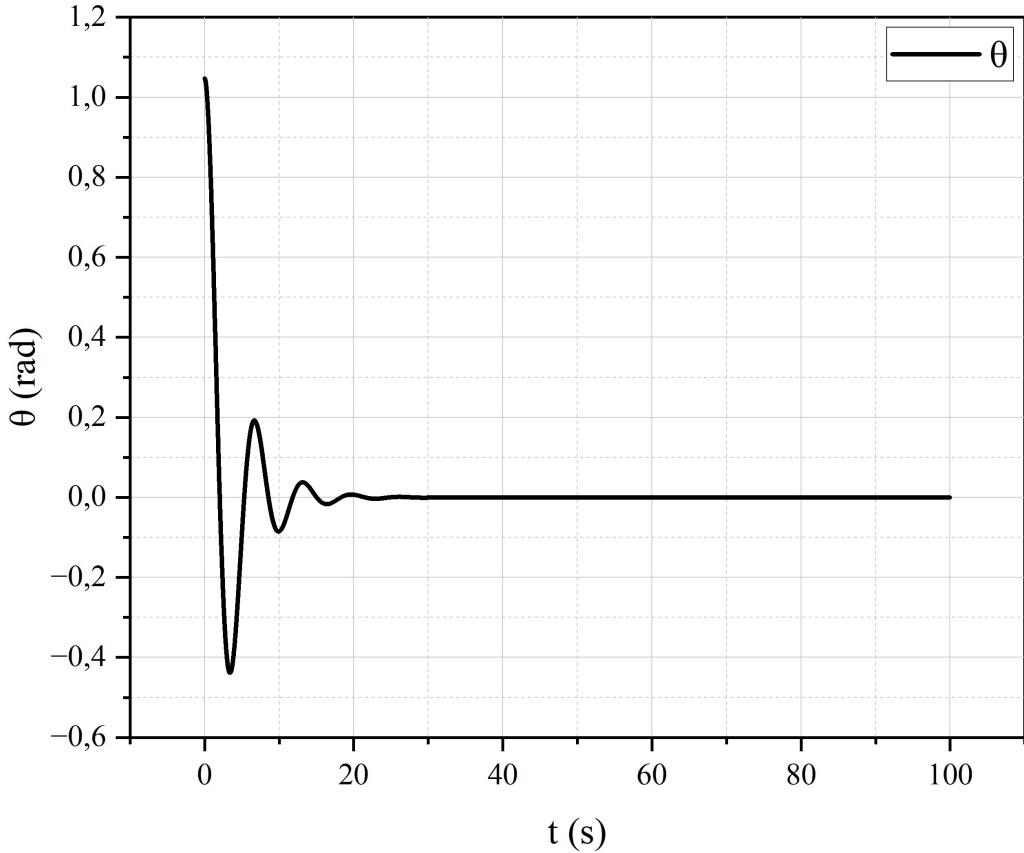
```

21      t = 0d0
22
23      theta = theta_0
24      omega = 0d0
25      ratio_gl = (g/p_l)
26
27      write(10, 11) 't', 'theta'
28      write(10, 12) t, theta
29
30      do i = 1, N
31          t = t + delta_t
32
33          omega_np1 = omega - ratio_gl*dsin(theta)*delta_t
34          $ - gamma*omega*delta_t
35
36          theta_np1 = theta + omega_np1*delta_t
37
38          write(10, 12) t, theta_np1
39
40          omega = omega_np1
41          theta = theta_np1
42      end do
43
44      close(10)
45      end program

```

A partir do arquivo de saída, temos o seguinte gráfico:

Figura 8: Comportamento do pêndulo amortecido por um fator $\gamma = 0.5$.



Podemos notar aqui que o comportamento deste pêndulo corresponde ao amortecimento subcrítico.

3.4 Tarefa B4

Para completar a investigação sobre os diferentes fatores que afetam o movimento de um pêndulo, temos a adição de uma força externa, tornando o movimento deste oscilador amortecido forçado. Utilizaremos $\gamma = 0,5$, $\Omega = \frac{2}{3}s^{-1}$ e $\Delta t = 0,03s$ durante toda a seção e investigaremos o que ocorre com o movimento quando impomos as forças de módulo $F_0 = 0$, $F_0 = 0,5$ e $F_0 = 1,2$. Para tal, será criado o gráfico conjunto de ω versus t e de θ versus t para cada um dos casos citados.

O código utilizado é muito similar aos anteriores, apenas adicionou-se o parâmetro $F_0 \sin(\Omega t)$ ao método numérico e alterou-se a saída para se adequar ao gráfico que resultará do programa. Para facilitar a criação dos gráficos, cada arquivo conterá o instante de tempo e os valores de θ ou ω para cada uma das forças. Isso será feito com a criação de vetores para as variáveis que nos outros códigos eram θ_n , θ_{n+1} , ω_n e ω_{n+1} onde o índice (1) é $F_0 = 0$, o (2) é $F_0 = 0,5$ e (3) é $F_0 = 1,2$.

```

3      implicit real*8(a-h,o-z)
4
5      parameter(pi = dacos(-1d0))
6      parameter(theta_0 = (60*pi/180d0))
7      parameter(g = 9.8d0)
8      parameter(p_l = 9.8d0) !l
9      parameter(p_mass = 1d0)
10     parameter(delta_t = 0.03d0)
11
12     dimension f_0(1:3)
13     dimension omega(1:3), theta(1:3)
14     dimension omega_np1(1:3), theta_np1(1:3)
15     parameter(f_0 = (/0d0, 0.5d0, 1.2d0/))
16
17   11    format(A4, 3A20)
18   12    format(f6.2, 3e20.11)
19
20   open(10, FILE='saida-b4-theta-12610389.dat')
21   open(20, FILE='saida-b4-omega-12610389.dat')
22
23   gamma = 0.5d0
24   freq = 2d0/3d0
25   ratio_gl = (g/p_l)
26
27   t_f = 102d0
28   N = t_f/(delta_t)
29   t = 0d0
30
31   do i = 1, 3
32       theta(i) = theta_0
33       omega(i) = 0d0
34   end do
35
36   write(10, 11) 't', 'Theta F_0 0', 'Theta F_0 0.5', 'Theta F_0 1.2'
37   write(20, 11) 't', 'Omega F_0 0', 'Omega F_0 0.5', 'Omega F_0 1.2'
38
39   write(10, 12) t, theta(1), theta(2), theta(3)
40   write(20, 12) t, omega(1), omega(2), omega(3)
41
42   do i = 1, N

```

```

43      t = t + delta_t
44
45      do j = 1, 3
46          omega_np1(j) = omega(j)
47          $ - ratio_gl*dsin(theta(j))*delta_t
48          $ - gamma*omega(j)*delta_t
49          $ + f_0(j)*dsin(freq*t)*delta_t
50
51      theta_np1(j) = theta(j) + omega_np1(j)*delta_t
52
53      omega(j) = omega_np1(j)
54      theta(j) = theta_np1(j)
55  end do
56
57      write(10, 12) t, mod(theta(1)+100*pi, -2*pi),
58      $ mod(theta(2)+100*pi, -2*pi),
59      $ mod(theta(3)+100*pi, -2*pi)
60      write(20, 12) t, omega(1), omega(2), omega(3)
61  end do
62
63  close(10)
64  end program

```

Se apenas obtivermos os resultados e os escrevermos no arquivo sem o tratamento $mod(\theta + 100\pi, -2\pi)$, temos os seguintes gráficos:

Figura 9: Comportamento de θ para o pêndulo amortecido por um fator $\gamma = 0.5$.

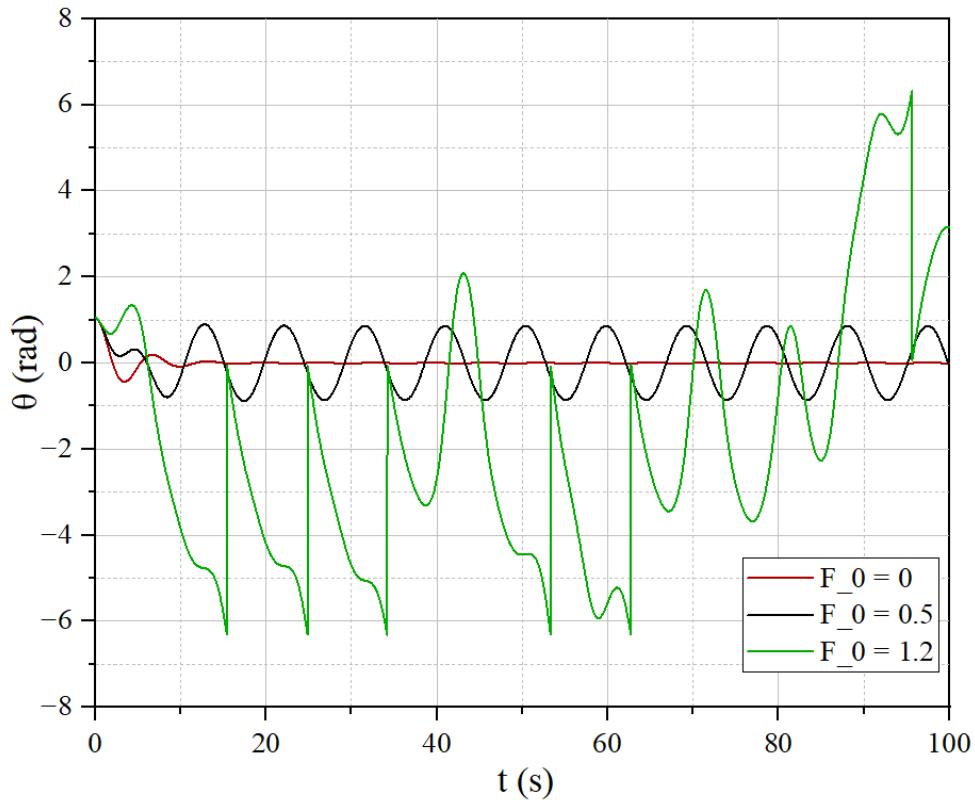
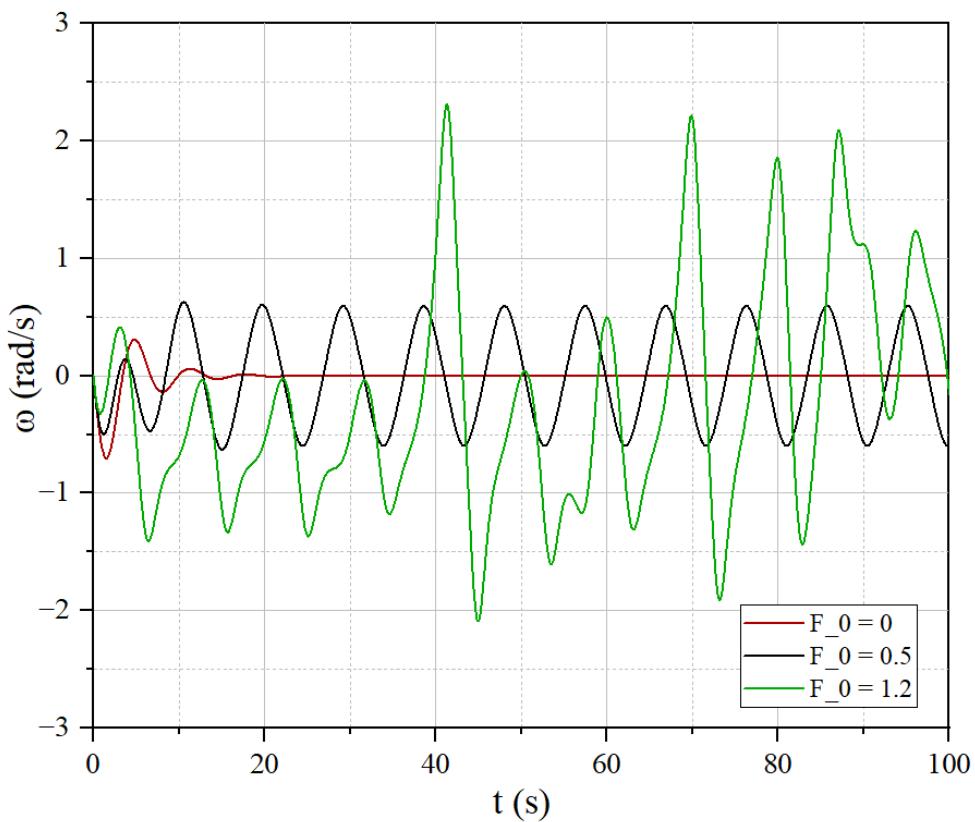


Figura 10: Comportamento de ω para o pêndulo amortecido por um fator $\gamma = 0.5$.



Porém, podemos ver que, para $F_0 = 1.2$ os resultados começam a sair do intervalo $[-\pi, \pi]$, logo é útil utilizar a seguinte forma de saída dos resultados:

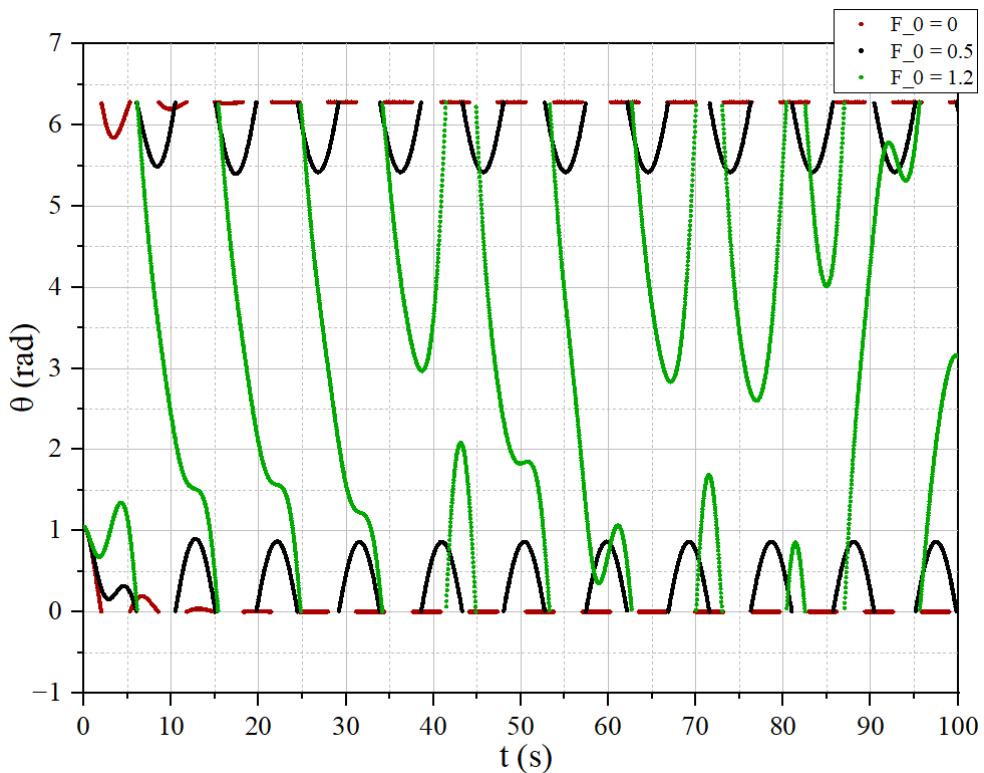
```

1      write(10, 12) t, mod(theta(1)+100*pi, -2*pi),
2      $ mod(theta(2)+100*pi, -2*pi),
3      $ mod(theta(3)+100*pi, -2*pi)
4      write(20, 12) t, omega(1), omega(2), omega(3)
5      end do

```

Que produz os seguintes resultados:

Figura 11: Comportamento de θ para o pêndulo amortecido por um fator $\gamma = 0.5$.



E o gráfico de $\omega(t)$ não é alterado.

Como $\text{mod}(\theta + 100\pi, -2\pi)$ ajusta os valores de θ para o intervalo $[0, 2\pi]$, os primeiros dois casos ficam um pouco menos fáceis de visualizar, porém o comportamento do terceiro fica mais fácil de visualizar. Esta forma de ajustar os intervalos será também utilizada nos próximos códigos.

É notável que os primeiros dois pêndulos possuem comportamentos muito bem definidos, o primeiro totalmente amortecido e o segundo com o padrão de oscilações bem definido. Porém, o terceiro já começa a apresentar um movimento complicado, que iremos estudar melhor nas próximas seções.

4 Tarefa C

Para investigar a existência do regime caótico para cada uma das situações citadas na tarefa anterior, vamos observar o movimento de dois pêndulos que são soltos de ângulos muito próximos ($\Delta\theta_0 = 0.001$).

Para tal, podemos adicionar um pêndulo novo no código, o qual terá $\theta_0 = \theta_{01} + 0.001$ como condição inicial. Plotaremos a diferença entre suas trajetórias no tempo ($\Delta\theta$ vs t), de forma com que podemos identificar o movimento caótico através da relação

$$\Delta\theta \approx \exp \lambda t \quad (15)$$

sendo que λ é o expoente de Liapunov, o qual, se positivo, identifica o regime caótico. Para facilitar, podemos aplicar \ln aos dois lados da equação, obtendo

$$\ln(\Delta\theta) = \lambda t \quad (16)$$

uma função linear, à qual podemos tomar o gráfico $\Delta\theta(t)$ e fazer seu fitting a $f(x) = a + bx$. O parâmetro b corresponderá ao expoente de Lyapunov.

O código desenvolvido foi o seguinte:

```
1 PROGRAM pendulo_comparacao
2
3     implicit real*8(a-h,o-z)
4
5     parameter(pi = dacos(-1d0))
6     parameter(theta_0 = (6*pi/18d0))
7     parameter(g = 9.8d0)
8     parameter(p_l = 9.8d0) !l
9     parameter(p_mass = 1d0)
10    parameter(delta_t = 0.03d0)
11
12    11    format(f6.2, e30.11)
13    12    format(A4, A30)
14
15    open(10, FILE='saida-c-12610389.dat')
16
17    gamma = 0.5d0
18    f_0 = 1.2d0
19    freq = 2d0/3d0
20
21    t_f = 102d0
22    N = t_f/(delta_t)
```

```

23      t = 0d0
24
25      theta1 = theta_0
26      theta2 = theta1 + 0.001d0
27      omega1 = 0d0
28      omega2 = omega1
29      ratio_gl = (g/p_1)
30
31      write(10, *) 'F_0 = ', f_0
32      write(10, 12) 't', 'log(delta_t)'
33
34      do i = 1, N
35          t = t + delta_t
36
37          omega1_np1 = omega1 - ratio_gl*dsin(theta1)*delta_t
38          $ - gamma*omega1*delta_t +f_0*dsin(freq*t)*delta_t
39
40          theta1_np1 = theta1 + omega1_np1*delta_t
41
42          omega2_np1 = omega2 - ratio_gl*dsin(theta2)*delta_t
43          $ - gamma*omega2*delta_t +f_0*dsin(freq*t)*delta_t
44
45          theta2_np1 = theta2 + omega2_np1*delta_t
46
47          write(10, 11) t, dlog(abs(theta2_np1 - theta1_np1))
48
49          omega1 = omega1_np1
50          theta1 = theta1_np1
51
52          omega2 = omega2_np1
53          theta2 = theta2_np1
54      end do
55
56      close(10)
57      end program

```

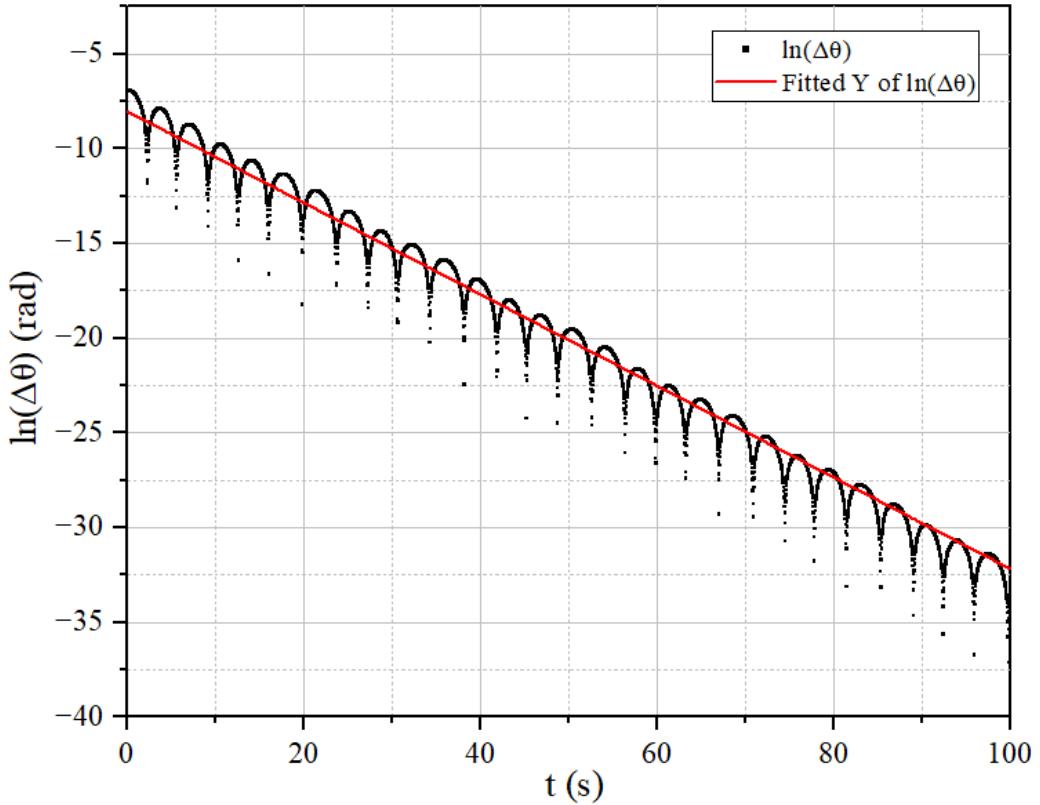
O qual se assemelha bastante com os anteriores, logo sua explicação será poupada. O código é rodado duas vezes, alterando-se a variável f_0 para os dois valores investigados: $F_0 = 0,5N$ e $F_0 = 1,2N$.

A partir do arquivo de saída, o qual vemos no código que contém uma tabela de valores

de $\ln(\Delta\theta)$ contra t , podemos criar os dois gráficos e fazer o fitting à função linear.

Para $F_0 = 0,5N$, temos:

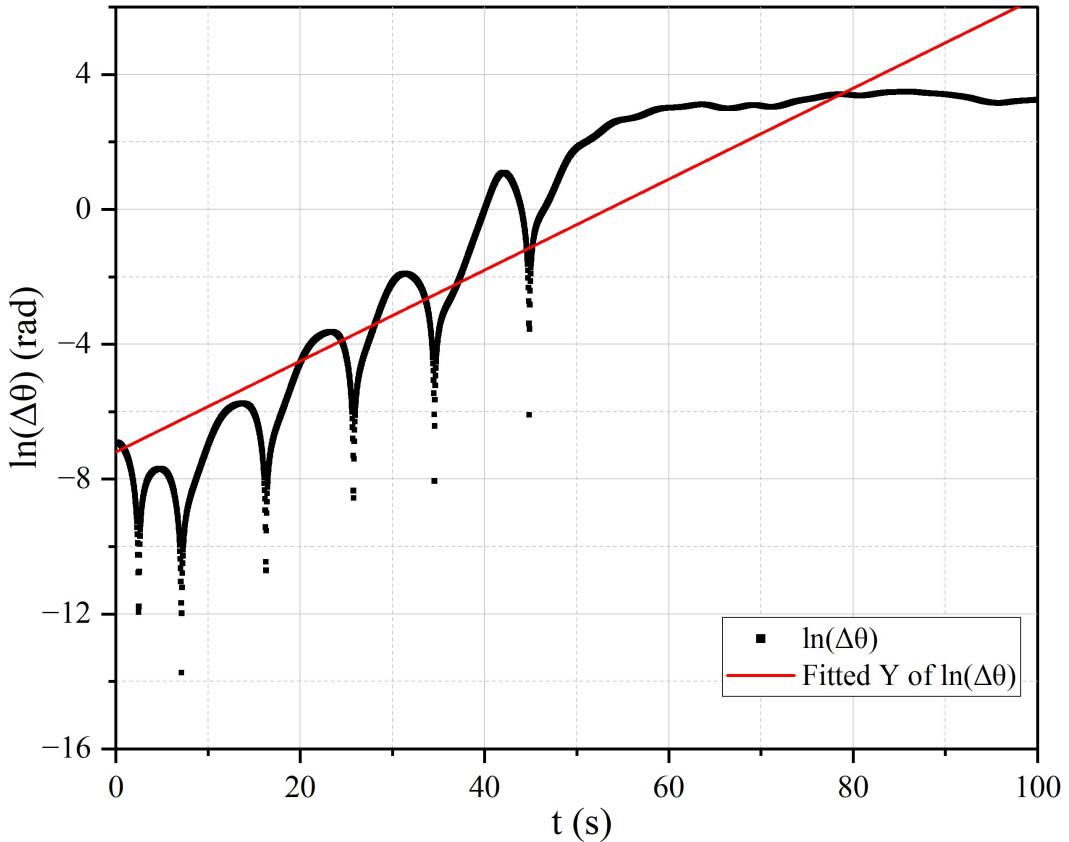
Figura 12: $F_0 = 0.5N$



Ao qual encontramos os coeficientes linear e angular como $a = (-8,02 \pm 0,03)$ e $b = \lambda = (0,2415 \pm 0,0001)$. Uma vez que temos λ negativo, é possível afirmar que este pêndulo não exibe movimento caótico, já que as trajetórias não se afastam de maneira exponencial.

Para $F_0 = 1,2N$, temos:

Figura 13: $F_0 = 1.2N$



Neste caso, obtemos $a = (-7, 19 \pm 0, 06)$ e $b = \lambda = (0, 135 \pm 0, 001)$, mostrando que este movimento é, conforme pudemos já observar na tarefa anterior, caótico.

5 Tarefa D

Nesta seção, nosso objetivo é observar o comportamento de ω versus θ para os pêndulos discutidos na seção anterior para condições iniciais próximas. Para isto, adicionamos um pêndulo ao código anterior e definimos que, dado um θ_0 arbitrário, soltaremos os outros dois de $\theta = \theta_0 - 0.001$ e $\theta = \theta_0 + 0.001$:

```

1      PROGRAM pendulo_trajetoria
2
3      implicit real*8(a-h,o-z)
4
5      parameter(pi = dacos(-1d0))
6      parameter(theta_0 = (6*pi/18d0))
7      parameter(g = 9.8d0)
8      parameter(p_l = 9.8d0)
9      parameter(p_mass = 1d0)
10     parameter(delta_t = 0.03d0)

```

```

11
12      dimension omega(1:3), theta(1:3)
13      dimension omega_np1(1:3), theta_np1(1:3)
14
15 11      format(2e30.11)
16
17      open(10, FILE='saida-d-p1.dat')
18      open(20, FILE='saida-d-p2.dat')
19      open(30, FILE='saida-d-p3.dat')
20
21      gamma = 0.5d0
22      f_0 = 1.2d0
23      freq = 2d0/3d0
24
25      t_f = 300d0
26      N = t_f/(delta_t)
27      t = 0d0
28
29      theta(1) = theta_0 - 0.001d0
30      theta(2) = theta_0
31      theta(3) = theta_0 + 0.001d0
32
33      do i = 1, 3
34          omega(i) = 0d0
35          index = i*10
36          write(index, 11) theta(i), omega(i)
37      end do
38
39      ratio_gl = (g/p_1)
40
41      do i = 1, N
42          t = t + delta_t
43
44          do j = 1, 3
45
46              omega_np1(j) = omega(j) - ratio_gl*dsin(theta(j))*delta_t
47              $ - gamma*omega(j)*delta_t + f_0*dsin(freq*t)*delta_t
48
49          theta_np1(j) = theta(j) + omega_np1(j)*delta_t
50

```

```

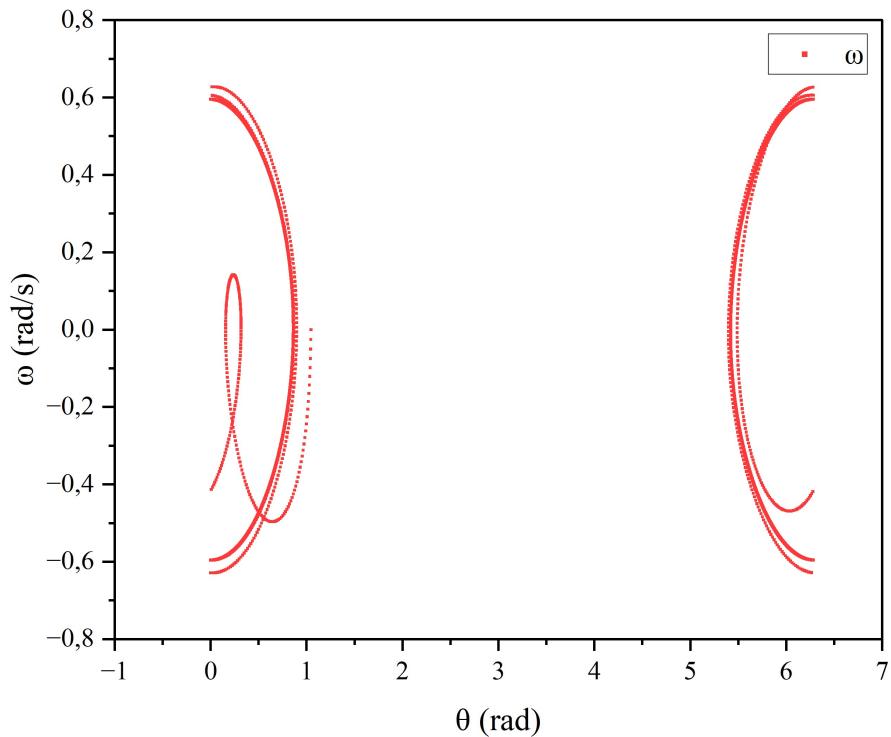
51      index = j*10
52      write(index, 11) mod(theta_np1(j)+100*pi, -2*pi),
53      $ omega_np1(j)
54
55      omega(j) = omega_np1(j)
56      theta(j) = theta_np1(j)
57
58      end do
59  end do
60
61  close(10)
62  close(20)
63  close(30)
64
65  end program

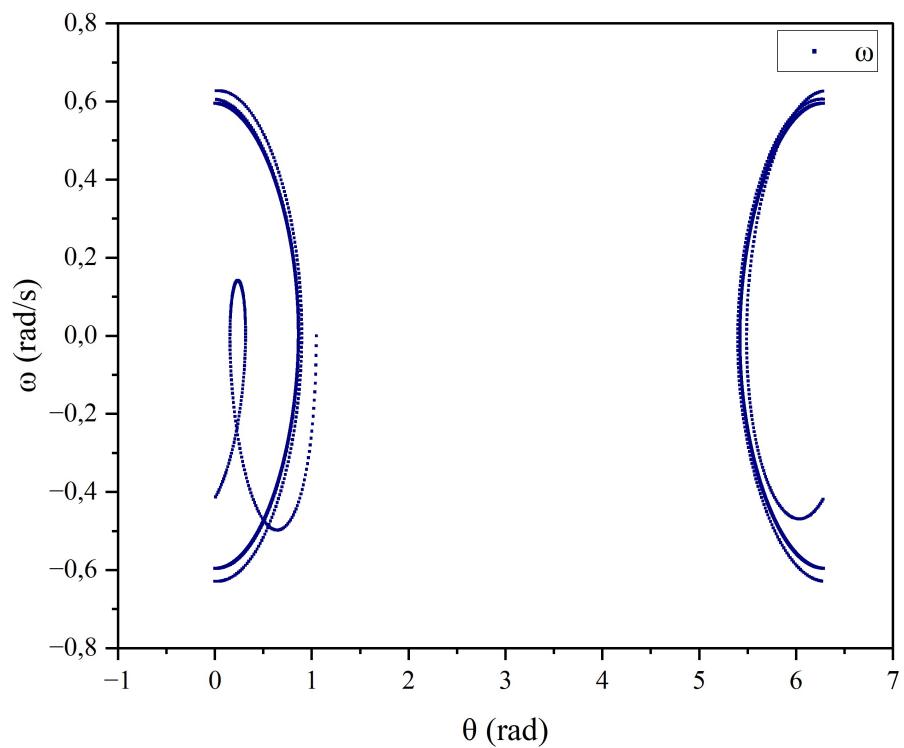
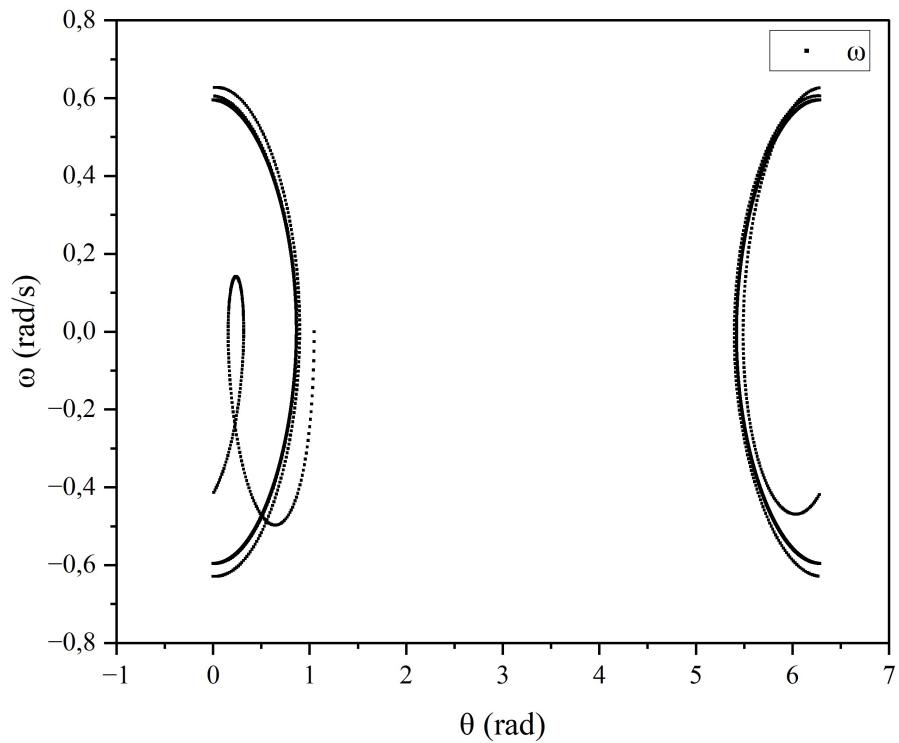
```

O funcionamento deste código é similar aos anteriores. Os vetores das variáveis θ_n , θ_{n+1} , ω_n e ω_{n+1} agora se referem aos três pêndulos e o programa é rodado duas vezes, uma para $F_0 = 0,5$ e uma segunda para $F_0 = 1,2$. As figuras em vermelho se referem ao pêndulo solto de $\theta = \theta_0 - 0.001$, as em preto se referem ao solto de $\theta = \theta_0$ e as em azul ao de $\theta = \theta_0 + 0.001$.

A seguir, temos os plots de $\omega(\theta)$ para $\theta_0 = \pi/3$ dos três pêndulos conforme o esquema de cores mencionado para $F_0 = 0.5$:

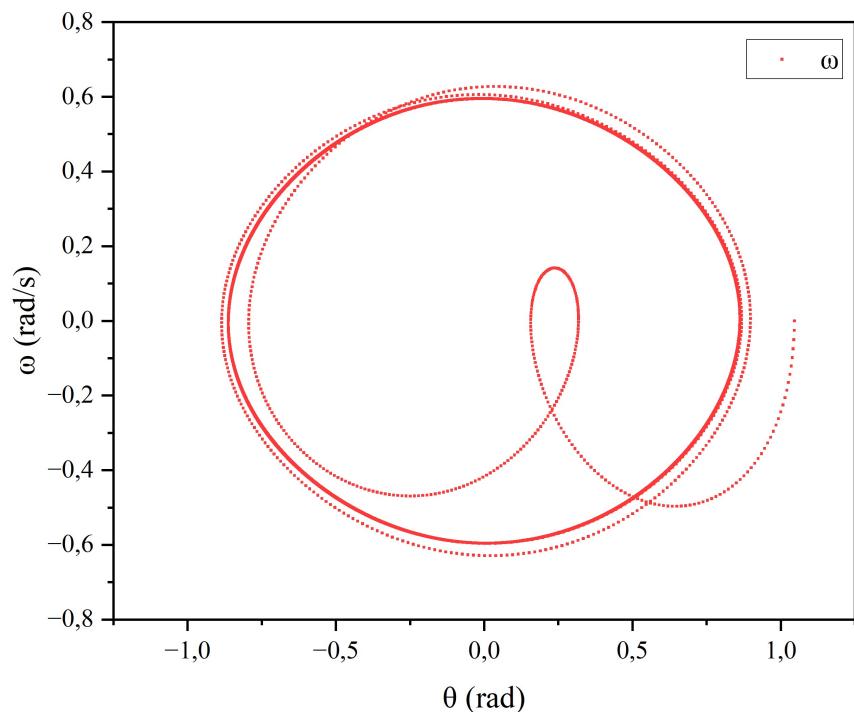
Figura 14: ω versus θ onde $F_0 = 0,5$





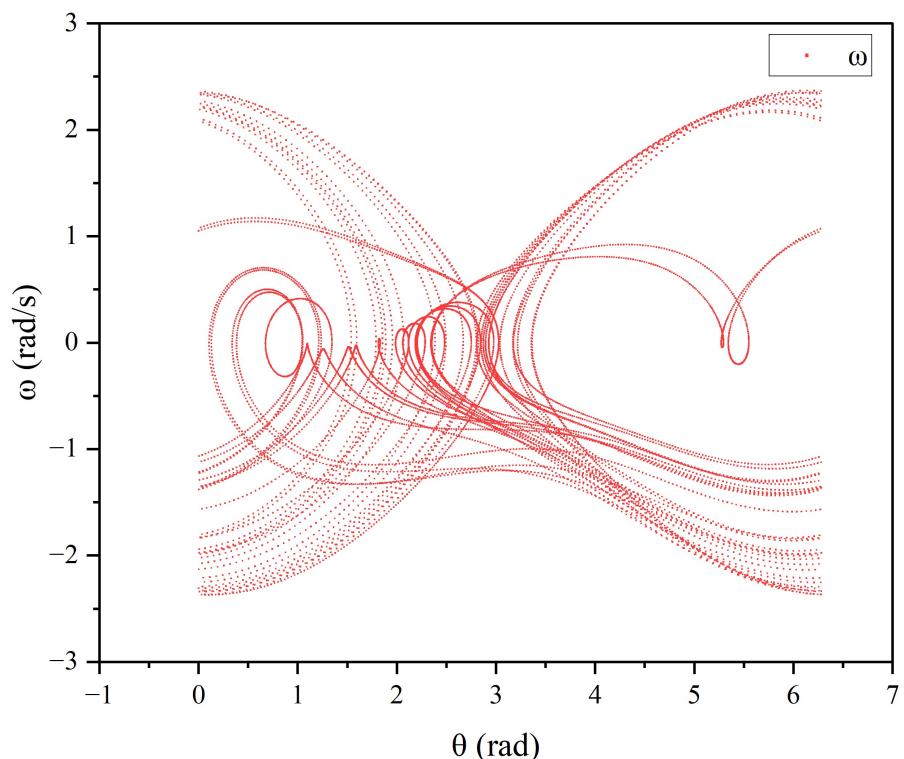
É importante notar que restringimos o intervalo de θ para $[0, 2\pi]$, porém as figuras são contínuas. Se escolhêssemos $[-\pi, \pi]$, a figura ainda seria a mesma, mas se pareceria com:

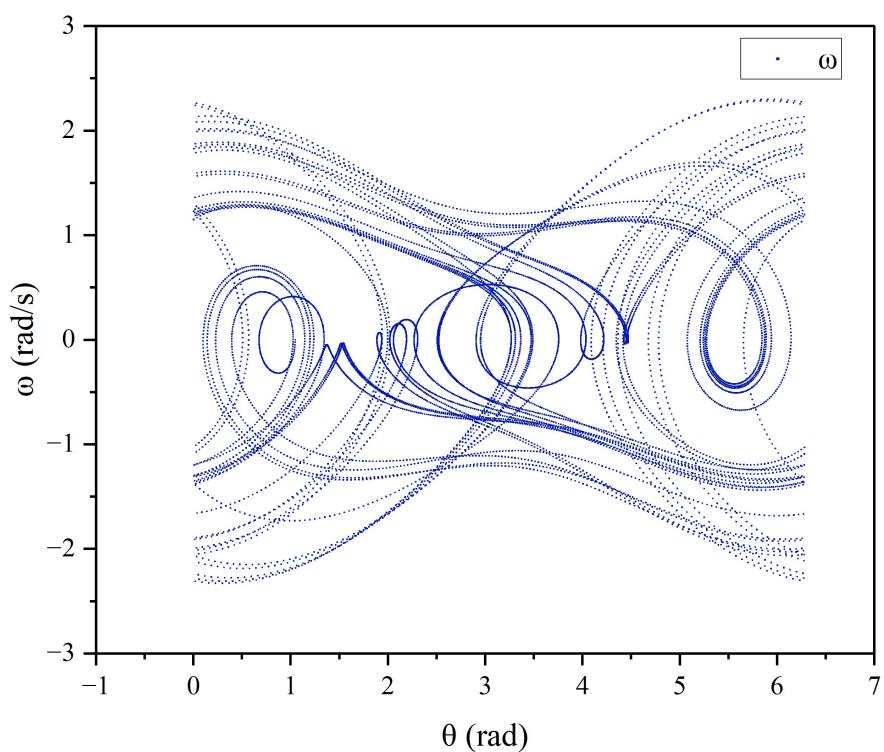
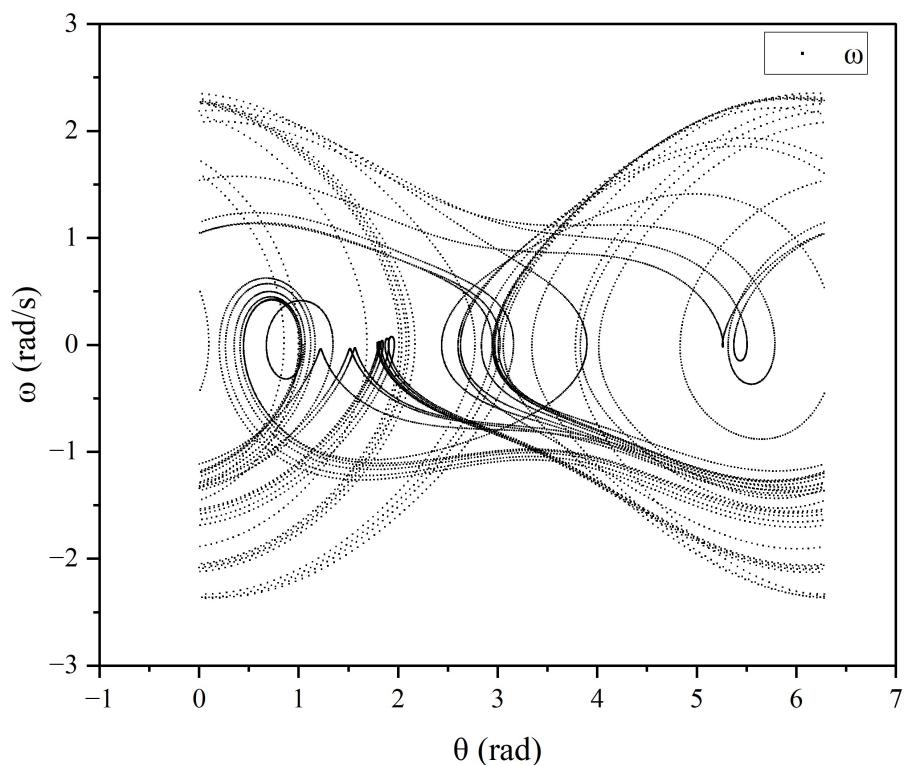
Figura 15: ω versus θ onde $F_0 = 0,5$, solto de $\theta = \pi/3 - 0.001$



Agora temos os plots de $\omega(\theta)$ para $\theta_0 = \pi/3$ dos três pêndulos para $F_0 = 1.2$:

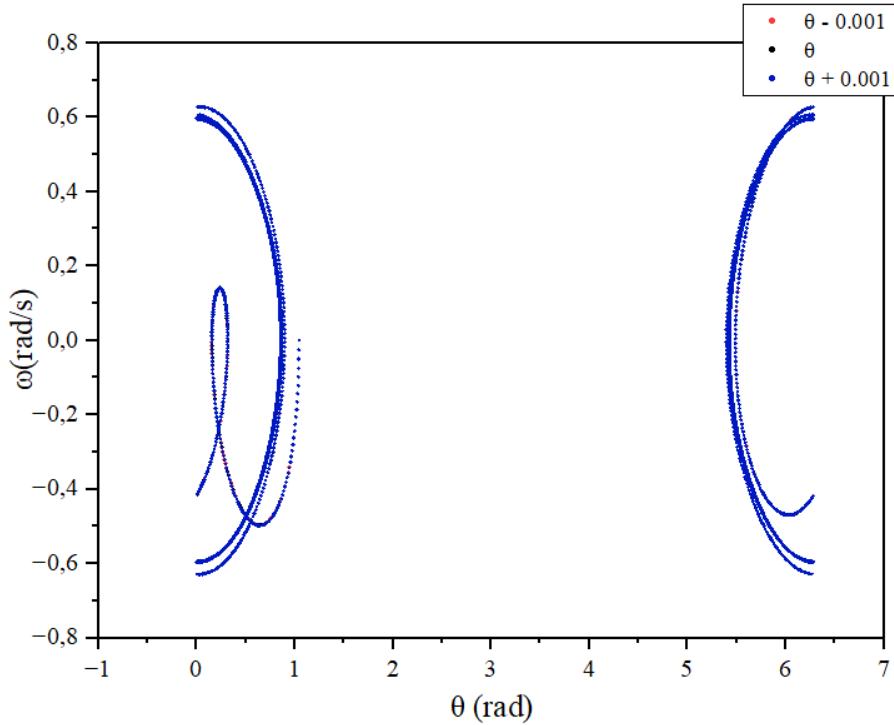
Figura 16: ω versus θ onde $F_0 = 1,2$





Analisando as imagens, vemos que as figuras para os pêndulos do caso $F_0 = 0,5$ são muito similares, de forma com que não vemos diferenças visuais significativas - plotando os três juntos vemos as imagens totalmente sobrepostas:

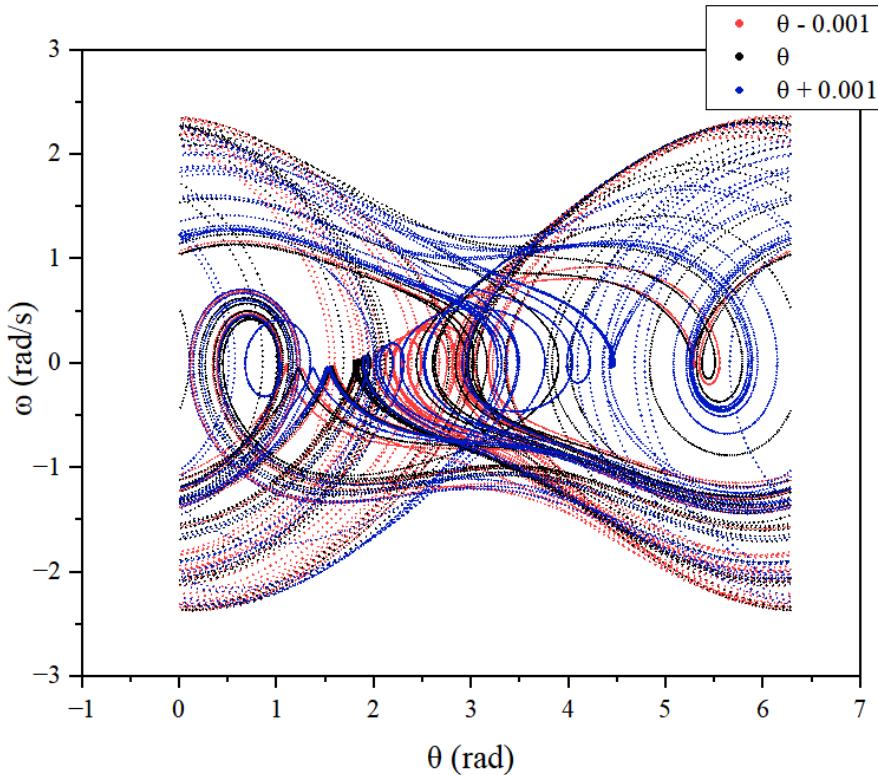
Figura 17: ω versus θ onde $F_0 = 0,5$



Portanto, como já esperávamos de um movimento não-caótico, temos um padrão bem definido ao qual condições iniciais muito próximas produzem resultados exatamente similares.

Já no caso de $F_0 = 1, 2$, apesar da estrutura do movimento ser muito similar, podemos notar lugares da figura que diferem para os três casos. Plotando todos no mesmo gráfico, temos:

Figura 18: ω versus θ onde $F_0 = 1, 2$



O que nos mostra que mesmo soltos de condições muito similares, realizarão movimentos bem diferentes, ainda que este movimento tenha uma estrutura similar.

6 Tarefa E

Na última tarefa, vamos fazer o gráfico anterior $\omega(\theta)$ na secção de Poincaré $\Omega t = n\pi$. Isso significa que observaremos outro tipo de estrutura que emerge do movimento caótico. Testaremos as duas condições iniciais de F_0 como na seção anterior e os mesmos três pêndulos anteriores.

O código é muito similar, porém adicionamos uma condição para que o ponto apareça no gráfico: que Ωt seja múltiplo de π . Para contemplar possíveis erros associados ao método numérico, aceitaremos uma tolerância ao dividir Ωt por π : que seu valor absoluto seja menor do que $\Delta_t/2$. Também o tempo da simulação é aumentado para que possamos observar com mais clareza a Seção de Poincaré (teremos mais pontos que atendem ao critério):

```

1   PROGRAM pendulo_poincare
2
3   implicit real*8(a-h,o-z)
4
5   parameter(pi = dacos(-1d0))

```

```

6   parameter(theta_0 = (6*pi/18d0))
7   parameter(g = 9.8d0)
8   parameter(p_l = 9.8d0)
9   parameter(p_mass = 1d0)
10  parameter(delta_t = 0.03d0)

11
12  dimension omega(1:3), theta(1:3)
13  dimension omega_np1(1:3), theta_np1(1:3)

14
15  11  format(2e30.11)

16
17  open(10, FILE='saida-e-p1.dat')
18  open(20, FILE='saida-e-p2.dat')
19  open(30, FILE='saida-e-p3.dat')

20
21  gamma = 0.5d0
22  f_0 = 1.2d0      ! variar parâmetro
23  freq = 2d0/3d0

24
25  t_f = 30000d0
26  N = t_f/(delta_t)
27  t = 0d0

28
29  theta(1) = theta_0 - 0.001d0
30  theta(2) = theta_0
31  theta(3) = theta_0 + 0.001d0

32
33  do i = 1, 3
34      omega(i) = 0d0

35
36  index = i*10
37  write(index, 11) theta(i), omega(i)
38 end do

39
40  ratio_gl = (g/p_l)

41
42  do i = 1, N
43      t = t + delta_t

44
45  do j = 1, 3

```

```

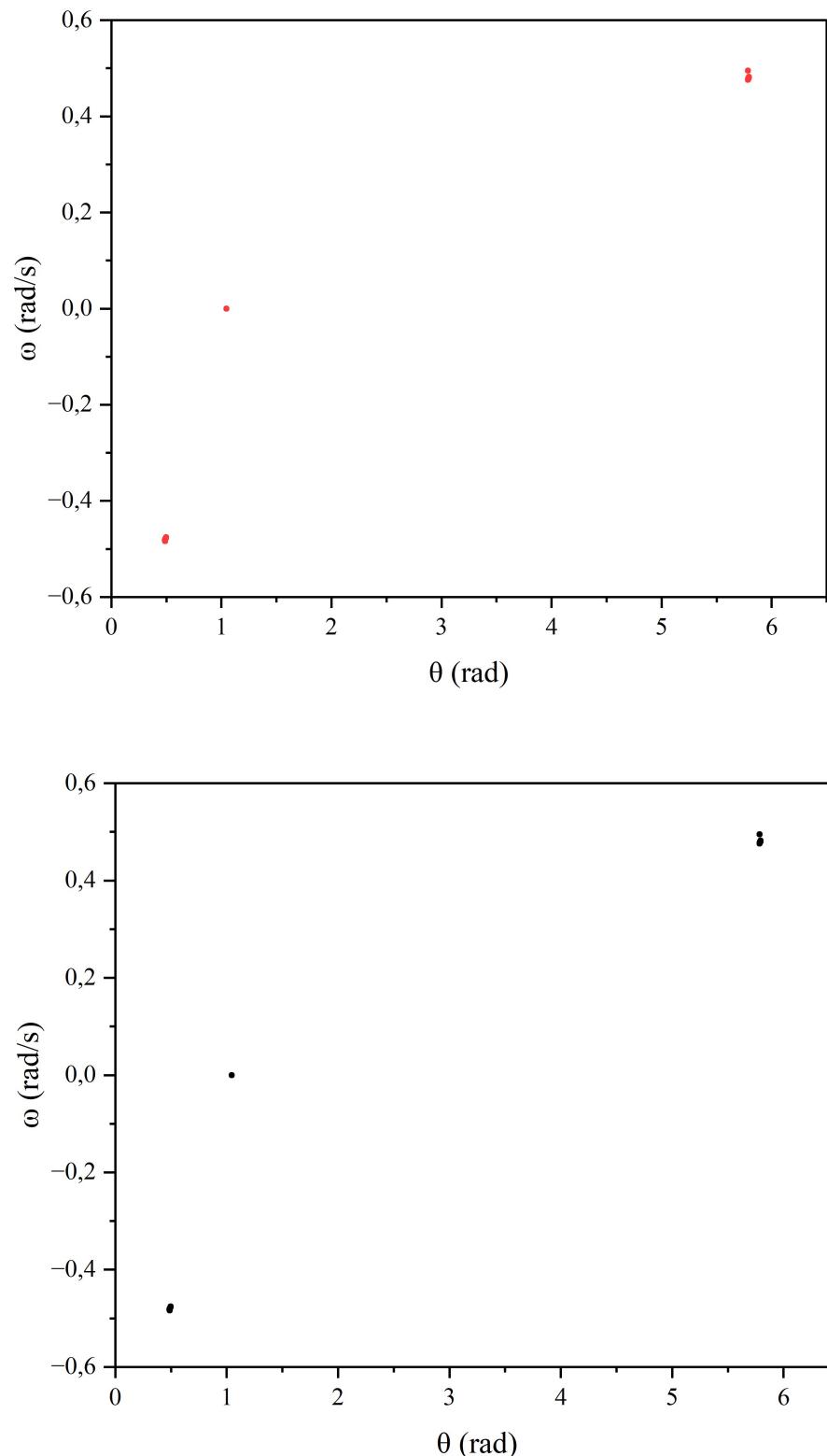
46
47          omega_np1(j) = omega(j) - ratio_gl*dsin(theta(j))*delta_t
48      $ - gamma*omega(j)*delta_t + f_0*dsin(freq*t)*delta_t
49
50      theta_np1(j) = theta(j) + omega_np1(j)*delta_t
51
52      omega(j) = omega_np1(j)
53      theta(j) = theta_np1(j)
54
55      end do
56
57      if(abs(mod(freq*t, pi)) .lt. delta_t/2) then
58          write(10, 11) mod(theta(1)+100d0*pi, -2d0*pi), omega(1)
59          write(20, 11) mod(theta(2)+100d0*pi, -2d0*pi), omega(2)
60          write(30, 11) mod(theta(3)+100d0*pi, -2d0*pi), omega(3)
61      end if
62      end do
63
64      close(10)
65      close(20)
66      close(30)
67
68      end program

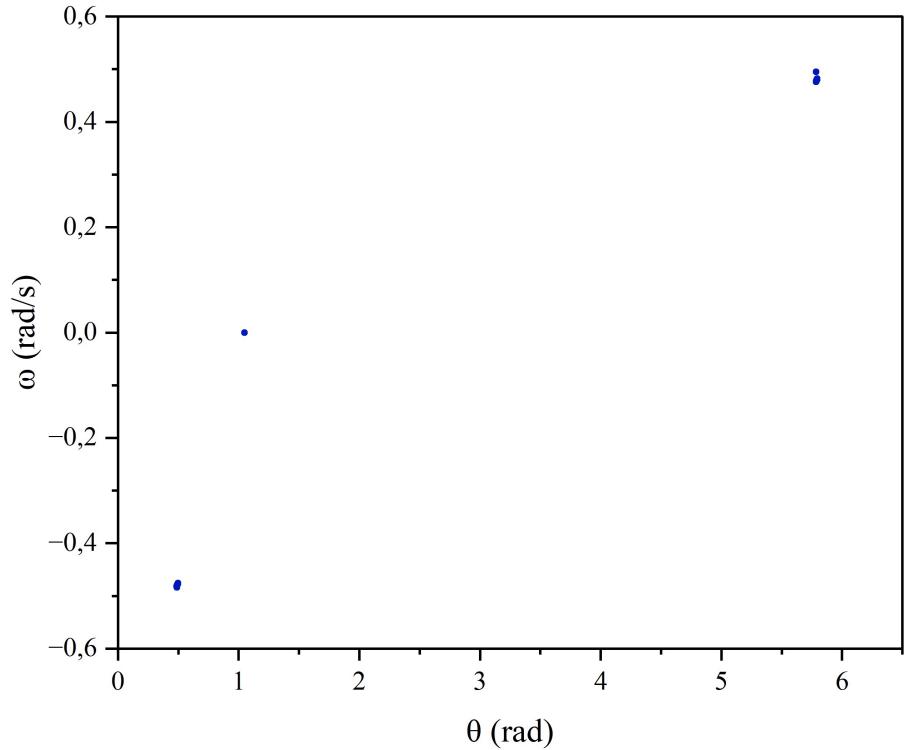
```

As saídas deste programa são graficadas conforme o esquema de cores convencionado na seção D para $\theta = \theta_0 - 0.001$, $\theta = \theta_0$, $\theta = \theta_0 + 0.001$, sendo $\theta_0 = \pi/3$.

Para $F_0 = 0,5$, temos:

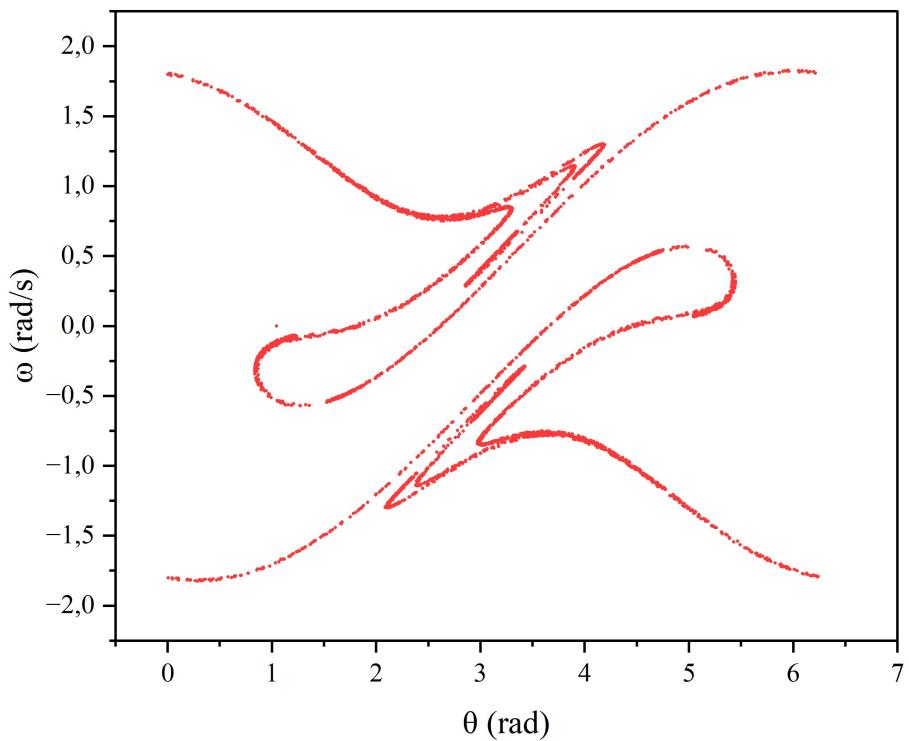
Figura 19: Secção de Poincaré $\Omega t = n\pi$ para $F_0 = 0,5$

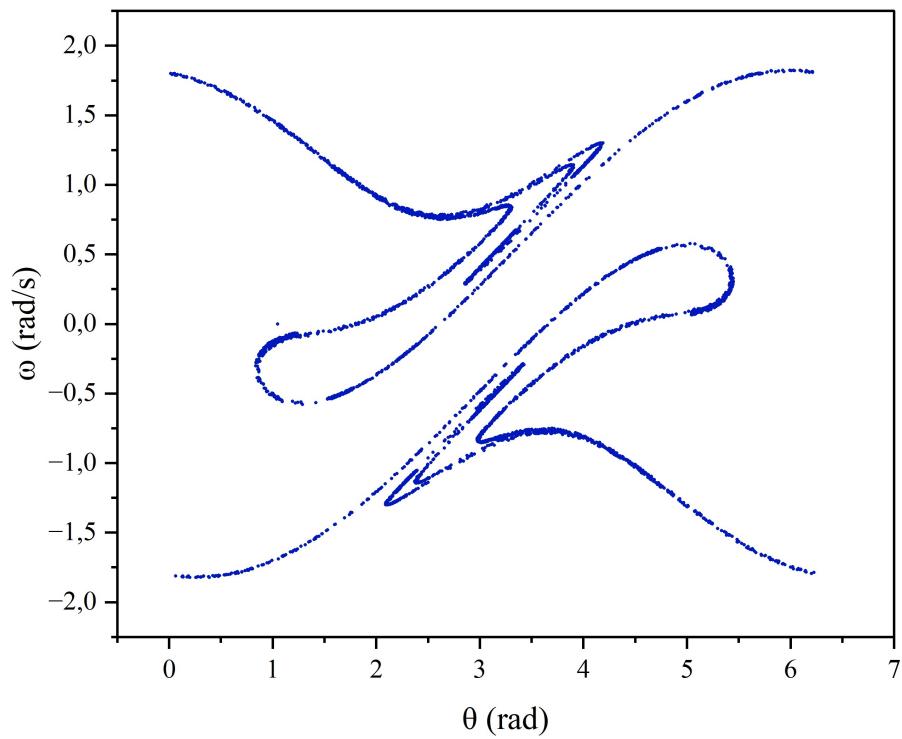
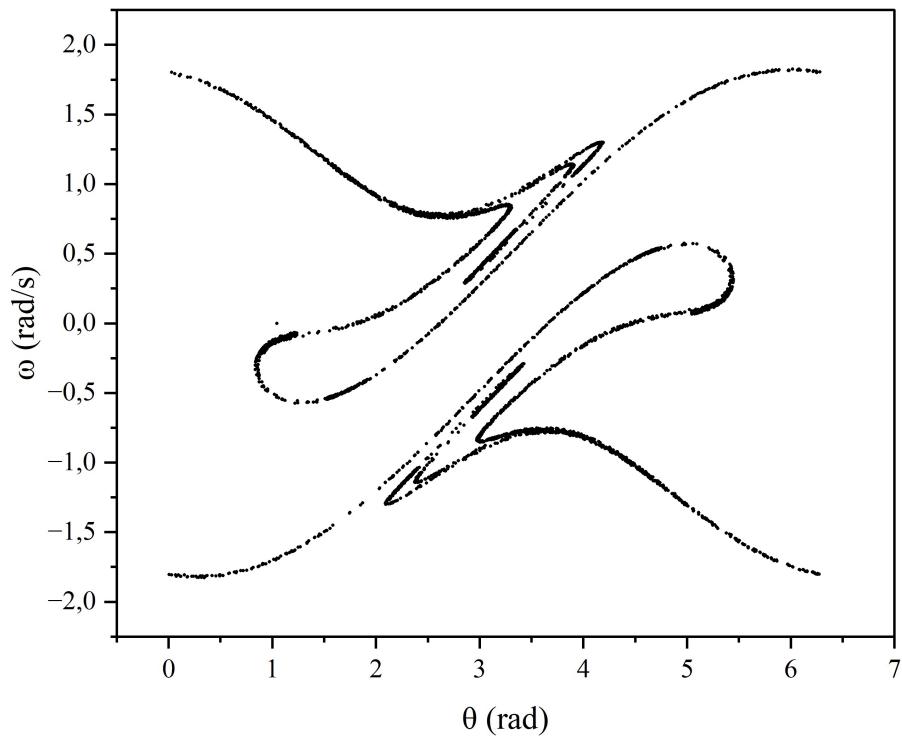




É notável que para este caso, que é determinístico, obtemos apenas dois pontos, conforme já esperado. Como delimitamos que $\Omega t = n\pi$ e o espaço de fase é uma órbita, conforme mostrado na seção D, teremos que os pontos que obedecem $\Omega t = n\pi$ tem sempre os mesmos parâmetros ω e θ . Já para o caso caótico, notamos uma estrutura bem diferente:

Figura 20: Secção de Poincaré $\Omega t = n\pi$ para $F_0 = 1, 2$





Desta vez, obtemos a mesma figura para os três casos com condições próximas do pêndulo caótico. Esta superfície é chamada de atrator estranho e é uma das formas de visualizar "ordem" em um movimento, à primeira vista, sem ordem alguma. O caso determinístico também tem um atrator, mas este tem uma estrutura muito simples (o ponto), enquanto os atratores estranhos dos casos caóticos possuem estruturas fractais. Assim, podemos concluir que, através dos espaços de fase é possível observar mais características importantes do movimento de um pêndulo que a princípio não ficam à vista.