

## **Relatório Final**

**Luis Vinicius Costa Silva**

Modelagem Matemática

Prof. Marcos Napoleão Rabelo

Data de Entrega: 18/07/2019

## **Conteúdo**

<b>Modelagem de um Tanque de Nível . . . . .</b>	<b>2</b>
<b>Modelagem de uma viga . . . . .</b>	<b>5</b>
<b>Modelagem de um Flip-Flop RS . . . . .</b>	<b>10</b>
<b>Modelagem de um Filtro Gaussiano . . . . .</b>	<b>13</b>
<b>Modelagem de um circuito RLC . . . . .</b>	<b>17</b>
Circuito 1 . . . . .	17
Circuito 2 . . . . .	18
<b>Modelagem de um motor de corrente contínua . . . . .</b>	<b>21</b>

## Modelagem de um Tanque de Nível

Foi realizada a modelagem do comportamento da altura de fluido em um tanque com uma determinada vazão de entrada e saída:

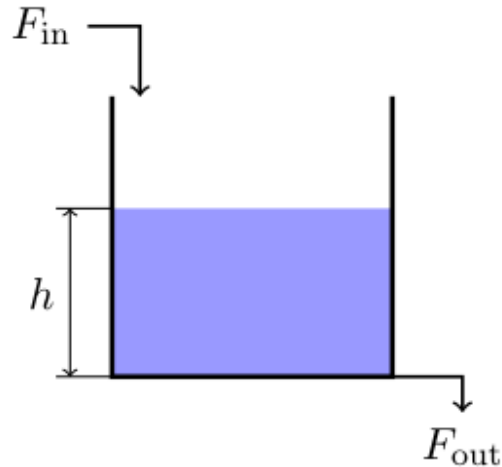


Figura 1: Tanque de nível

Os dados do processo são:

- $Q_e$  – vazão de entrada;
- $Q_s$  – vazão de saída;
- $H$  – nível de fluido do tanque;
- $\rho$  – massa específico do fluido no tanque;
- $A$  – Área da base do tanque;

$$m = \rho AH$$

$$m = \frac{\partial P}{\partial t} AH + \rho \frac{\partial A}{\partial t} H + \rho A \frac{\partial H}{\partial t}$$

Hipóteses do modelo:

Considera-se  $\rho$  constante e  $A$  constante, logo:

$$\frac{\partial m}{\partial t} = \rho A \frac{\partial H}{\partial t}$$

Balanço de massa:

$$\frac{\partial m}{\partial t} = P_e Q_e - P_s Q_s$$

Equação do sistema:

$$\rho A \frac{\partial H}{\partial t} = P_e Q_e - P_s Q_s$$

$$\frac{\partial H}{\partial t} = \frac{P_e Q_e - P_s Q_s}{\rho A}$$

Análise Qualitativa:

- Se  $P_s Q_s > P_e Q_e$ , logo  $\frac{\partial H}{\partial t} < 0 \rightarrow$  tanque esvaziando, portanto  $H$  decresce;
- Se  $P_s Q_s < P_e Q_e$ , logo  $\frac{\partial H}{\partial t} > 0 \rightarrow$  tanque enchendo, portanto  $H$  cresce;
- Se  $P_s Q_s = P_e Q_e$ , logo  $\frac{\partial H}{\partial t} = 0 \rightarrow$  tanque permanece igual, portanto  $H$  conserva-se;

Busca-se computar o nível de fluido no tanque em função da quantidade de massa que sai do tanque. Logo, integra-se a seguinte equação:

$$\frac{\partial H}{\partial t} = \frac{\rho Q_e - \rho Q_s}{\rho A}$$

Para este fim, foi escrito o seguinte programa que resolve o problema:

---

```

from matplotlib.pyplot import *
from numpy import *

A = 10.0          # area do tanque (m**2)
H = [100.0]      # altura inicial de fluido no tanque (m)
a = 0.0          # intervalos de integracao
b = 10.0         #
N = 1000         # discretizacao do intervalo de integracao
passo = (b-a) / N # passo
t = linspace(\
    a,\
    b,\
    N,\
    endpoint=False\
) # vetor de tempo

fQe = lambda t: 10*t # razao da vazao de entrada (m**3)
fQs = lambda t: 50*pi*cos(t) # razao de vazao de saida (m**3)

Qe = fQe(t)

```

```

Qs = fQs(t)

for i in range(N-1):
    # altura em funcao dos parametros de entrada
    H.append(H[i] + passo*(Qe[i] - Qs[i])/A)

plot(t.tolist(),H)
xlabel('Tempo (s)')
ylabel('Altura de fluido no tanque (m)')
grid()
show()

```

---

Tendo como entrada os parâmetros abaixo:

- Área do tanque:  $10 \text{ m}^2$ ;
- Altura inicial de fluido no tanque:  $100.0 \text{ m}$ ;
- Intervalo de integração:  $[0.0, 10.0]$ ;
- Fator de discretização: 1000;
- Vazão de entrada:  $f_{Q_e}(t) = 10t \text{ l}$ ;
- Vazão de saída:  $f_{Q_s}(t) = 50\pi\cos(t) \text{ l}$ ;

obteve-se o seguinte comportamento do sistema:

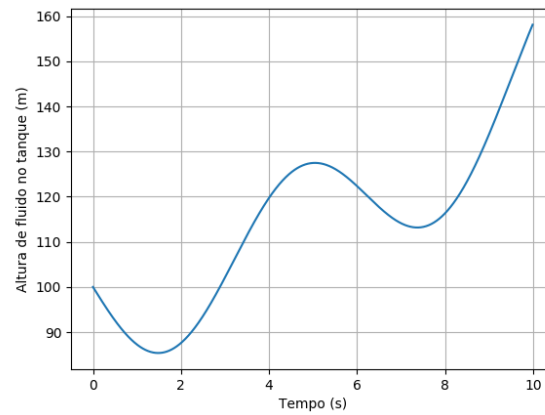


Figura 2: Estado da altura de fluido no tanque a cada passo de tempo

## Modelagem de uma viga

Foi realizada a modelagem da vibração de uma viga Euler-Bernoulli com duas áreas seccionais distintas, esta recebendo uma frequência de vibração  $\omega$ , como segue abaixo:

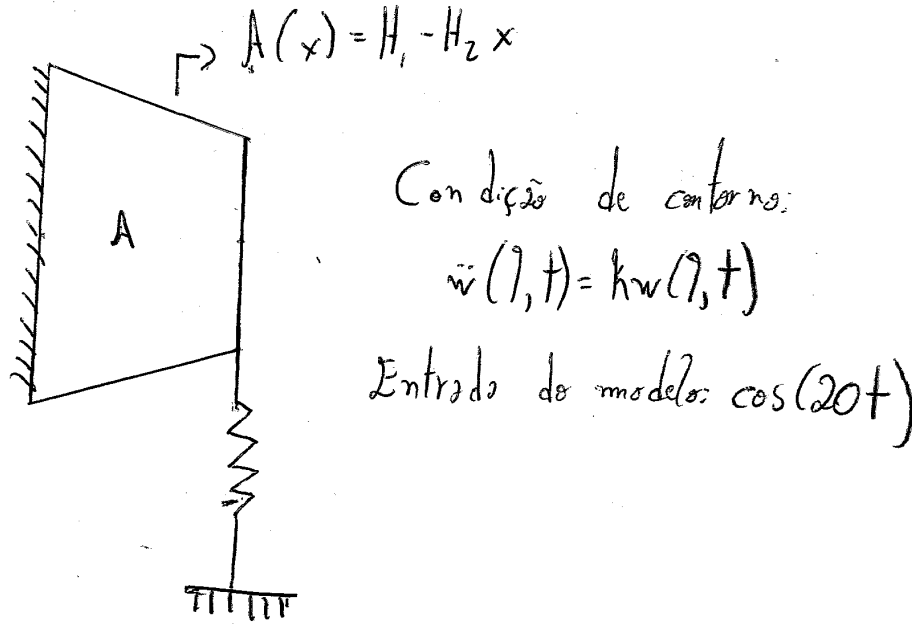


Figura 3: Problema da viga

O sistema foi modelado a partir do método direto da rigidez, para tanto, foram obtida as matrizes de rigidez, massa e inércia, são estas:

$$w_{xx_1} = \begin{bmatrix} 96x^3 - 72x & 192x^4 - 192x^2 + 24 \\ 192x(4x^3 - 3x) & 192x(8x^4 - 8x^2 + 1) \end{bmatrix}$$

$$w_{xx_2} = \begin{bmatrix} 24x(12x^2 - 3) & 24x(32x^3 - 16x) \\ (12x^2 - 3) * (96x^2 - 16) & (96x^2 - 16) * (32x^3 - 16x) \end{bmatrix}$$

$$w_{xx_3} = \begin{bmatrix} 192x^3 & 576x^4 - 192x^2 \\ 576x^4 - 192x^2 & \frac{9216x^5}{5} - 1024x^3 + 256x \end{bmatrix}$$

A análise da dinâmica do sistema se deu pela resolução da seguinte equação através do método

de Euler:

$$R_1 u + R_2 \ddot{u} = \vec{F} \quad \text{tal que:}$$

$$R_1 = P_4 + P_1$$

$$R_2 = P_2 + P_3 \quad \text{onde:}$$

$$P_1 = (w_{xx_1} - w_{xx_3}) \big|_{x=l} - (w_{xx_1} - w_{xx_3}) \big|_{x=0}$$

$$P_2 = \frac{-1}{k} w_{xx_2} \big|_{x=l} - w_{xx_2} \big|_{x=0}$$

$$P_3 = \rho \int_0^l \begin{bmatrix} T_3^2 & T_3 T_4 \\ T_4 T_3 & T_4^2 \end{bmatrix} \delta_1 - \delta_2 x$$

$$P_4 = - \left( \cos(20t) \begin{bmatrix} T_3^2 & T_3 T_4 \\ T_4 T_3 & T_4^2 \end{bmatrix} \big|_{x=l} \right) - \left( \cos(20t) \begin{bmatrix} T_3^2 & T_3 T_4 \\ T_4 T_3 & T_4^2 \end{bmatrix} \big|_{x=0} \right)$$

$$F = \int_0^1 \begin{bmatrix} x \cos(t) T_3 \\ x \cos(t) T_4 \end{bmatrix}$$

onde  $k$  foi definido arbitrariamente como  $10^6$  (constante elástica). Tais termos foram obtidos através da integração por partes do funcional de energia abaixo:

$$\begin{aligned} EI(H(i, x)_{xxx} H(j, x) u(i, t) \big|_0^1 - H(i, x)_{xx} H(j, x)_{xx} H(j, x) u(i, t) \big|_0^1 + \\ \int_0^l A(i, x)_{xx} H(j, x) u(i, t) dx) + \\ \rho A \int_0^1 H(i, x) H(j, x) \ddot{u}(i, t) dx = \int_0^1 f(x, t) H(j, x) dx \end{aligned}$$

onde  $A = H_1 - H_2 x$

Após isso, um experimento foi criado a fim de observar a frequência de vibração da viga dado um sinal de entrada da forma  $\cos(\omega t)$ . O experimento teve os seguintes parâmetros:

- Funções de Forma: Polinômios de Hermite  $T_3$  e  $T_4 \rightarrow 4x^3 - 3x$  e  $8x^4 - 8x^2 + 1$ ;
- Frequência de vibração na barra:  $\omega = \cos(20t)$ ;
- Tempo simulado: 10 s
- Fator de discretização: 10000
- Comprimento da viga: 15 m

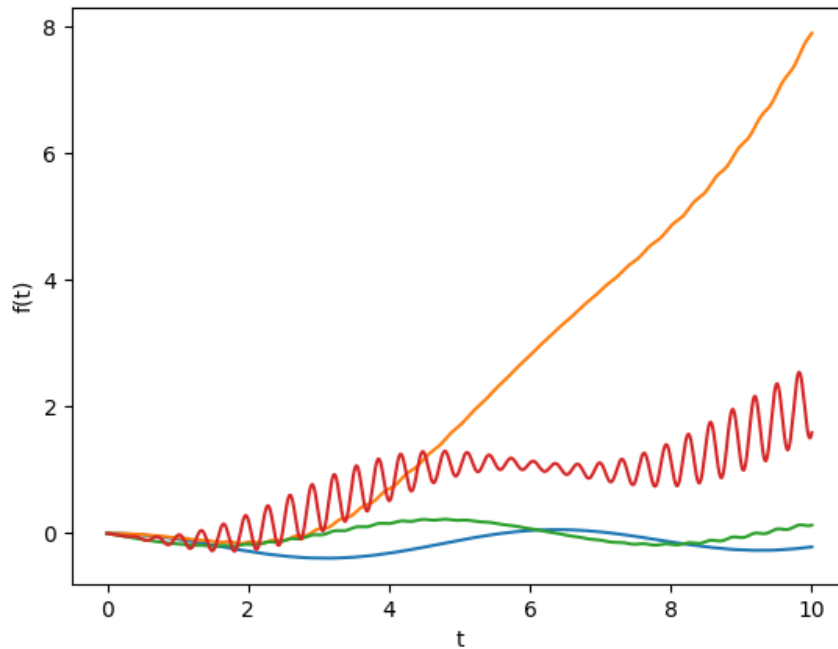


Figura 4: Simulação da dinâmica da viga em função do tempo

A implementação abaixo reproduz o experimento descrito:

---

```

from sympy import symbols, sympify, diff, integrate, Matrix, cos, sin
from numpy import linspace
from matplotlib.pyplot import plot, show, xlabel, ylabel

x, t, E, I, k, u_1, u_2, delta_1, delta_2, rho = \
symbols('x t E I k u_1 u_2 delta_1 delta_2 rho')

# polinomios de hermite de grau 3 e 4 - alternativamente chebysev
t3 = sympify("4*x**3-3*x")
t4 = sympify("8*x**4-8*x**2+1")

#matrizes de rigidez, inercia e massa
wxx_1 = Matrix([\
    [diff(diff(diff(t3)))*t3, diff(diff(diff(t3)))*t4],\
    [diff(diff(diff(t4)))*t3, diff(diff(diff(t4)))*t4]\
])

wxx_2 = Matrix([\
    [(diff(diff(t3)))*diff(t3), (diff(diff(t3)))*diff(t4)],\
    [(diff(diff(t4)))*diff(t3), (diff(diff(t4)))*diff(t4)]\
])

```

```

])

wxx_3 = Matrix([\
    [integrate(diff(diff(t3))*diff(diff(t3)),x),\
    integrate(diff(diff(t4))*diff(diff(t3)),x)],\
    [integrate(diff(diff(t3))*diff(diff(t4)),x),\
    integrate(diff(diff(t4))*diff(diff(t4)),x)]\
])

intervalo_integracao = 15
P1 = wxx_1-wxx_3.subs(x,intervalo_integracao)-wxx_1-wxx_3.subs(x,0)

P2 = (-1/k)*wxx_2.subs(x,intervalo_integracao)-wxx_2.subs(x,0)

P3 = rho*integrate(\
    Matrix(\
        [[t3*t3, t3*t4],\
        [t4*t3, t4*t4]]\
    ) *\
    (delta_1-delta_2*x),(x,0,intervalo_integracao))

P4 = -1*(cos(20*t) * Matrix(\
    [[t3*t3,t3*t4],\
    [t4*t3,t4*t4]])\
).subs(x,intervalo_integracao)-\
(cos(20*t)*Matrix([[t3*t3,t3*t4],\
[t4*t3,t4*t4]]))\
).subs(x,0)

F = integrate(Matrix(\
    [[x*cos(t)*t3],[x*cos(t)*t4]]),\
    (x,0,1))

R1 = P4+P1

R2 = (P2+P3).subs((k, rho, delta_1, delta_2),(10**6, 3, 0.5, 0.2))
R2 = R2.subs(k, 10**6)
R2 = R2.subs(rho, 3)
R2 = R2.subs(delta_1, 0.5)
R2 = R2.subs(delta_2, 0.2)

# Analise da dinamica de  $R1*u + R2*u'' = F$ 
barra_momento = R1*R2.inv()
barra_forca = R2.inv()*F

```



```

a = 0
b = 10
N = 10000
h = (b-a)/N

t = linspace(a,b,N,endpoint=True)

v1 = [0.0]
v2 = [0.0]
v3 = [0.0]
v4 = [0.0]

for i in range(len(t)-1):
    v1.append(v1[i]+h*v2[i])
    v2.append(eval(str(v2[i]+h*((barra_momento[0,0].subs(t,t[i])*v1[i]+\
        barra_momento[0,1].subs(t,t[i])*v3[i]))+\
        (F[0].subs(t,t[i]))).subs(t,t[i]))\
        ).replace("t",str(t[i]))))\
    )
    v3.append(v3[i]+h*v4[i])
    v4.append(eval(str(v4[i]+h*((barra_momento[1,0].subs(t,t[i])*v1[i]+\
        barra_momento[1,1].subs(t,t[i])*v3[i]))+\
        (F[1].subs(t,t[i]))).subs(t,t[i]))\
        ).replace("t",str(t[i]))))\
    )

plot(t,v1)
plot(t,v3)
plot(t,v2)
plot(t,v4)
xlabel("t")
ylabel("f(t)")
show()

```

---

## Modelagem de um Flip-Flop RS

Foi realizada a modelagem do Flip-Flop RS abaixo:

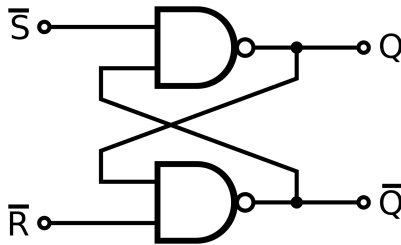


Figura 5: Flip Flop RS Síncrono

A saída do Flip-Flop RS no instante  $k + 1$  é dada pelas equações abaixo:

$$f(x_1, x_2, x_3) = \overline{x_3^{(k+1)}} = \overline{\left( x_1^{(k)} + \left( x_2^{(k)} + \overline{x_3^{(k)}} \right) \right)}$$

$$f(x_1, x_2, x_3) = x_3^{(k+1)} = \overline{\left( x_2^{(k)} + \left( x_1^{(k)} + x_3^{(k)} \right) \right)}$$

Logo, têm-se a seguinte tabela verdade:

$S$	$R$	$Q_{ant}$	Saída
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	(Estado não permitido)
1	1	1	(Estado não permitido)

Tabela 1: Tabela Verdade do Flip Flop RS

O gráfico abaixo demonstra o comportamento do Flip-Flop RS sob a seguinte entrada:

- Comprimento do sinal: 1024 bits;
- Sinal da entrada de  $R$ :  $5\pi t$ ;
- Sinal da entrada de  $S$ :  $10\pi t$ ;
- Sinal da entrada de  $Q_{ant}$ :  $15\pi t$ ;

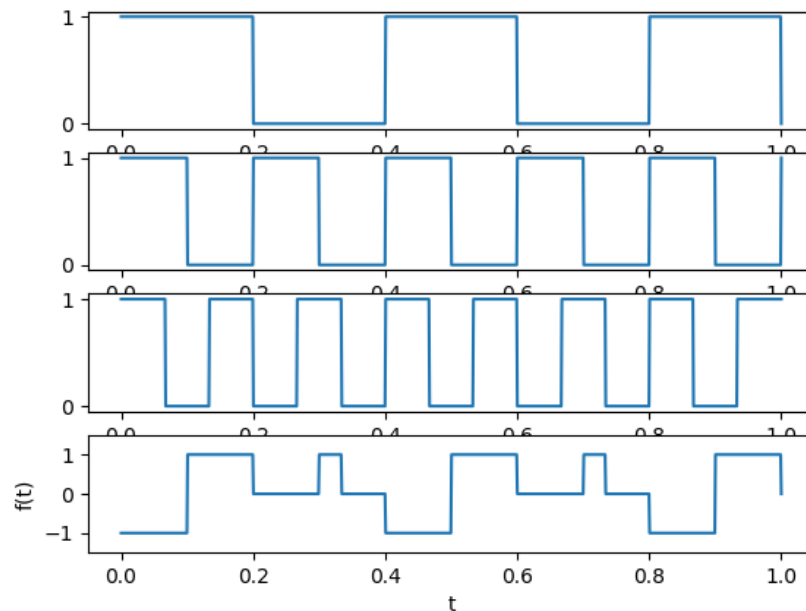


Figura 6: Simulação do Flip-Flop RS - Considere -1 como estado proibido

O experimento pode ser replicado através do código abaixo:

---

```

from __future__ import print_function
from itertools import product as produto_cartesiano
from matplotlib.pyplot import *
from scipy import signal
from random import randint as rand
from numpy import *

def FFRS(r, s,q):#considera-se -1 sinal invalido
    return 0 if (r,s,q) in [(0,0,0),(0,1,0),(0,1,1)] else (1 if (r,s,q) in

def tabela_verdade():
    print('r\ts\tQ(k)\tQ(k+1)')
    for r, s,q in produto_cartesiano((0, 1), repeat=3):
        print('{!s:5}\t{!s:5}\t{!s:5}\t{!s:5}'.format(r, s,q, FFRS(r,s,q)))

print("Tabela Verdade do Flip Flop RS\n"+\
      "Considere \"-1\" como estado proibido\n"\
      )
tabela_verdade()

```

```

tamanho_palavra = 1024

t = linspace(0, 1, tamanho_palavra, endpoint=True)

r = [0 if x<0 else 1 for x in signal.square(5 * pi * t)]
subplot(4, 1, 1)
plot(t, r)

s = [0 if x<0 else 1 for x in signal.square(10 * pi * t)]
subplot(4, 1, 2)
plot(t, s)

q = [0 if x<0 else 1 for x in signal.square(15 * pi * t)]
subplot(4, 1, 3)
plot(t, q)

fx = [FFRS(_r,_s,_q) for (_r,_s,_q) in zip(r,s,q)]
subplot(4, 1, 4)
plot(t, fx)

xlabel("t")
ylabel("f(t)")

ylim(-1.5, 1.5)
plot()
show()

```

---

## Modelagem de um Filtro Gaussiano

O filtro gaussiano será modelado a partir do seguinte filtro passa-baixo:

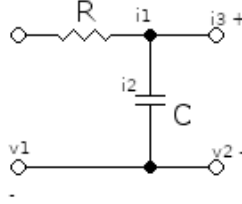


Figura 7: Filtro passa-baixo genérico

Temos que:

$$i_1 = i_2 + i_3$$

$$i_3 = i_1 - i_2$$

$$\frac{v_3}{R} = \frac{v_1}{R} - C\dot{v}_2$$

$$v_3 = v_1 - RC\dot{v}_2 \text{ visto que } v_2 = v_3$$

$$\frac{\dot{v}_2}{R} = \frac{v_1}{R} - C\dot{v}_2$$

$$v_2 = v_1 - RC\dot{v}_2$$

Aplica-se Transformada de Laplace para ir do domínio do tempo para domínio da frequência

$$v_2 = v_1 - RCSv_2 (S \in \mathbb{C} \text{ é o domínio da função})$$

$$v_1 = v_2 \cdot (1 + RCS) \cdot v_2$$

Relação de entrada/saída:

$$T = \frac{\text{Saída}}{\text{Entrada}}$$

$$T = \frac{v_2}{v_1} = \frac{1}{1 + RCS} \quad \left( S = S_1 + j_1 \right) \quad \text{Função de Transferência}$$

$$T = cv_2v_1 = \frac{1}{(1 + RCS_1) + jRC\omega} \cdot \frac{(1 + RCS_1) - jRC\omega}{(1 + RCS_1) - jRC\omega} = \frac{1 + RCS_1 - jRC\omega}{(1 + RCS_1)^2 + (RC\omega)^2}$$

Estabilidade no domínio da frequência ( $Re(T)$  denota a parte real de  $T$ ):

$$Re(T) \leq 0 \quad 1 + RCS_1 < 0 = S_1L - \frac{1}{RC}$$

$$\frac{1}{RC} = \text{constante de tempo}$$

Se  $Re(T) > 0$ , então sistema é instável:  
 Supondo que  $S_1 = 0$ , temos que:

$$T = \frac{1}{1 + jRC\omega} \cdot \frac{1 - jRC\omega}{1 - jRC\omega} = \frac{1 - jRC\omega}{1 - (RC\omega)^2}$$

$$T = \frac{1}{1 + (RC\omega)^2} - j \frac{RC\omega}{1 + (RC\omega)^2}$$

$$|T|^2 = \frac{1}{(1 + (RC\omega)^2)^2} + \frac{(RC\omega)^2}{(1 + (RC\omega)^2)^2} = \frac{1 + (RC\omega)^2}{(1 + (RC\omega)^2)^2} = \frac{1}{1 + (RC\omega)^2}$$

$$|T| = \frac{1}{(1 + (RC\omega)^2)^{0.5}}$$

$$arg(T) = \frac{\frac{RC\omega}{1 + (RC\omega)^2}}{\frac{1}{1 + (RC\omega)^2}} = \frac{-RC\omega}{1 + (RC\omega)^2} \cdot (1 + (RC\omega)^2) =$$

$$arg(T) = \tan^{-1} v_1 = \tan^{-1}(RC\omega) = \theta$$

$$l = \left( \frac{1}{(1 + R^2 C^2 \omega^2)^{0.5}} \right)$$

$$T = l \cdot (\cos(-\tan^{-1}(RC\omega)) + j \sin(-\tan^{-1}(RC\omega)))$$

$$T = e^{(1 + R^2 C^2 \omega^2)^{-0.5}} \cdot (\cos \theta - j \sin \theta)$$

$$\frac{1}{(1 + R^2 C^2 \omega^2)^{0.5}} \rightarrow \text{Constante de Tempo}$$

A implementação em Python abaixo gera um pulso de sinal modelado através das equações desenvolvidas anteriormente:

---

```

from matplotlib.pyplot import *
from numpy import *

def FiltroGaussiano (t, freq_central=5, larg_banda=0.5, \
                      ref_larg_banda=-6, limiar_truncamento=-60):
    #frequencia central deve ser maior ou igual a zero
    #largura de banda fracionaria deve ser maior que zero
    #limiar de referencia para largura de banda deve ser menor que 0
    #ja que a funcao gaussiana eh para x \in (-\infty, \infty), o limiar de
    #truncamento trunca os valores do filtro apos um ponto

    # exp(-a t^2) <-> sqrt(pi/a) exp(-pi^2/a * f^2) = g(f)
    ref = pow(10.0, ref_larg_banda / 20.0)
    
```

```

# fdel = freq_central*larg_banda/2:  g(fdel) = ref
# colocando em funcao de a temos que...
#  $\pi^2/a * freq\_central^2 * larg\_banda^2 /4 = -\log(ref)$ 
a = -(pi * freq_central * larg_banda) ** 2 / (4.0 * log(ref))

sinal_envoltorio = exp(-a * t * t)
sinal_real = sinal_envoltorio * cos(2 * pi * freq_central * t)
sinal_imaginario = sinal_envoltorio * sin(2 * pi * freq_central * t)
return sinal_real, sinal_imaginario, sinal_envoltorio

intervalo = (-1.0,1.0)
h = 0.001

t = linspace(intervalo[0], intervalo[1], 1/h)
freq_central      = 4.0
larg_banda        = 0.5
ref_larg_banda    = -6.0
limiar_truncamento = -60.0

sinal_real, sinal_imaginario, envoltorio = FiltroGaussiano(
    t, \
    freq_central, \
    larg_banda, \
    ref_larg_banda, \
    limiar_truncamento \
)
plot(t, sinal_real, t, sinal_imaginario, t, envoltorio, '--')

legend(["Parte Real","Parte Imaginaria","Envelope"])
title("Constante de Tempo RC de Filtro Gaussiano\n" + \
      "$e^{-at^2}e^{1j\ 2\pi f_c t}$")
show()

```

---

O gráfico abaixo foi gerado através do código com os seguintes parâmetros de entrada:

- Largura de banda no domínio da frequência: 0.5 Hz;
- Nível de referência no qual a largura de banda é calculada: -6.0 dB;
- Limiar de truncamento: -60.0 dB

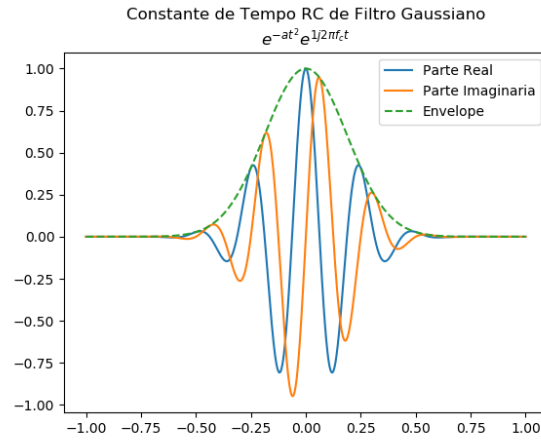


Figura 8: Pulso Gaussiano gerado pela execução do código

No gráfico seguinte, é apresentada a variação da saída da parte real do circuito sob diferentes frequências do filtro:

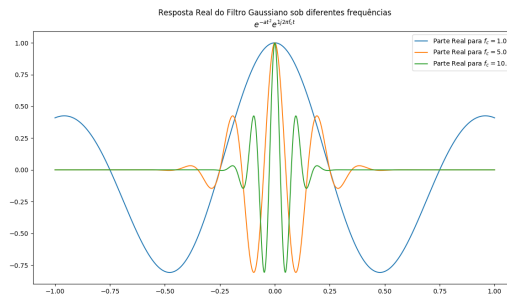


Figura 9: Saída Real do Filtro Gaussiano para diferentes frequências



# Modelagem de um circuito RLC

## Circuito 1

Considerando o circuito abaixo:

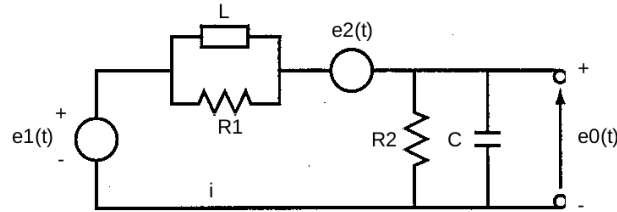


Figura 10: Circuito RLC

Foi criado um modelo do tipo  $f(e_1(t), e_2(t)) = e_0(t)$  em função das correntes:  
Tem-se que:

$$i_C + i_{R2} + i_2 = 0$$

$$i_2 = i_{R1} + i_L \text{ logo:}$$

$$i_C + i_{R2} + i_{R1} + i_L = 0 \text{ (Lei de Kirchoff)}$$

Para cada elemento do circuito, temos que:

$$i_{R1} = \frac{1}{R_1}(e_0 + e_2 - e_1)$$

$$i_{R2} = \frac{e_0}{R_2}$$

$$i_C = C \frac{de_0}{dt}$$

$$i_L = i_L(0) + \frac{1}{L} \int_0^1 (e_0 + e_2 - e_1) dt$$

Logo, a dinâmica do sistema é dada por:

$$e_0(t) = \frac{1}{R_1}(e_1(t) - e_2(t)) + \frac{1}{L} \int_0^t (e_1(t) - e_2(t)) dt - i_L(0)$$

Foi modelado o circuito abaixo:


$$\left\{ \begin{array}{lll} \text{nó 1} \rightarrow & i_{L_1} + i_{L_2} + i_{R_1} & = 0 \\ \text{nó 2} \rightarrow & i_{L_1} + i_{R_2} + i_5 & = 0 \\ \text{nó 3} \rightarrow & i_5 + i + i_{L_2} & = 0 \\ \text{nó 4} \rightarrow & i_{L_2} + i_4 + i_3 & = 0 \\ \text{nó 5} \rightarrow & i_{R_1} + i_{R_2} + i_{L_2} + i_1 & = 0 \\ \text{nó 6} \rightarrow & i_{L_2} + i_3 + i_2 & = 0 \\ \text{nó 7} \rightarrow & i_1 + i_{R_4} + i_{R_3} & = 0 \\ \text{nó 8} \rightarrow & i_{R_4} + i_{R_3} + i_2 & = 0 \\ \text{nó 9} \rightarrow & i_{L_1} + i + i_1 & = 0 \end{array} \right.$$

Também pela Lei de Kirchoff, as tensões são expressas da seguinte forma:

$$\begin{cases} V_{C_1} - V_i + V_{L_1} &= 0 \\ V_i + V_{L_2} &= 0 \\ V_{L_2} + V_{R_2} + V_{C_2} &= 0 \\ V_{L_2} + V_{R_3} &= 0 \\ V_{R_3} + V_{R_4} &= 0 \\ V_{L_1} + V_{R_2} + V_{R_1} &= 0 \end{cases}$$

Uma vez que o modelo foi implementado, foram utilizados os seguintes parâmetros no experimento:

- $R_1 : 1 \, \Omega$ ;
- $R_2 : 2 \, \Omega$ ;
- $R_3 : 3 \, \Omega$ ;
- $R_4 : 4 \, \Omega$ ;
- $C_1 : 1 \times 10^{-9} F$ ;
- $C_2 : 2 \times 10^{-9} F$ ;
- $L_1 : 1 \times 10^{-9} H$ ;
- $L_2 : 2 \times 10^{-9} H$ ;
- Tensão da fonte:  $1 \, A$ ;
- Tempo simulado:  $7 \times 10^{-5} \, s$  (devido ao rápido equilíbrio do sistema)

Foram obtidos os seguintes resultados:

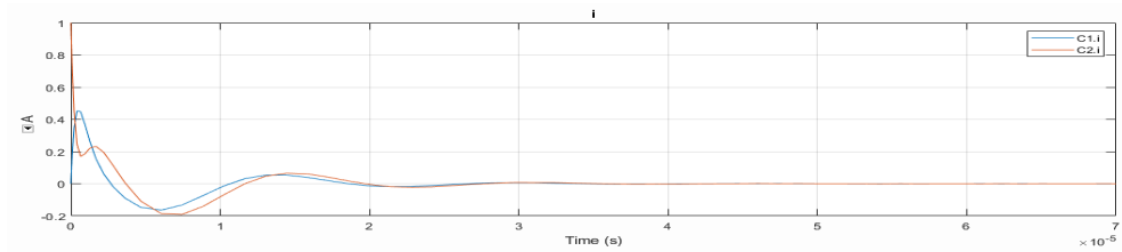


Figura 12: Corrente nos capacitores  $C_1$  e  $C_2$

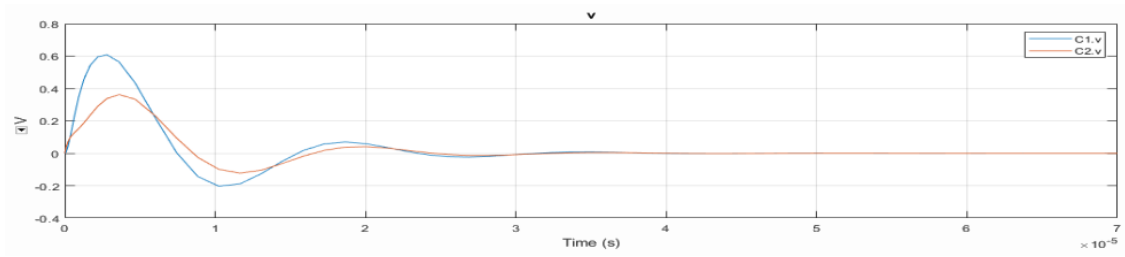


Figura 13: Tensão nos capacitores  $V_1$  e  $V_2$

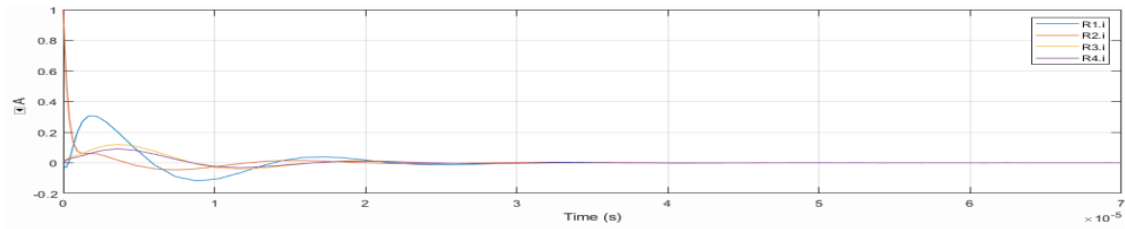


Figura 14: Corrente nas resistências  $R_1$ ,  $R_2$ ,  $R_3$  e  $R_4$

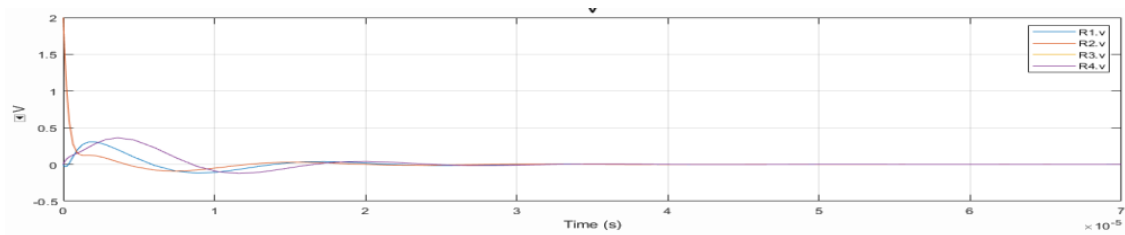


Figura 15: Tensão nas resistências  $R_1$ ,  $R_2$ ,  $R_3$  e  $R_4$

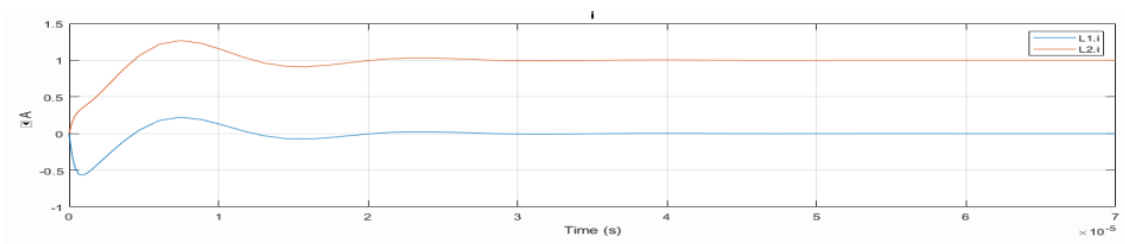


Figura 16: Corrente nos indutores  $L_1$  e  $L_2$

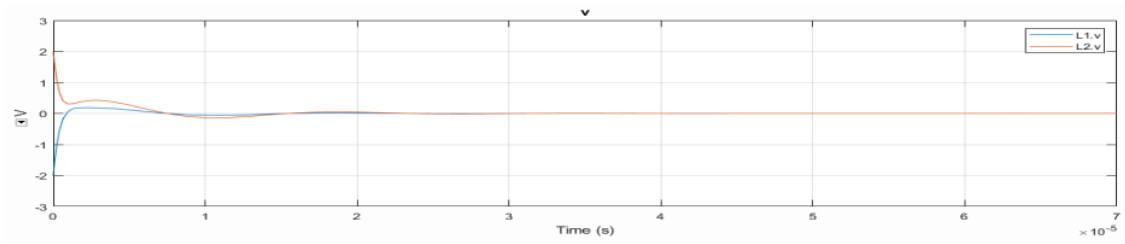


Figura 17: Tensão nos indutores  $L_1$  e  $L_2$

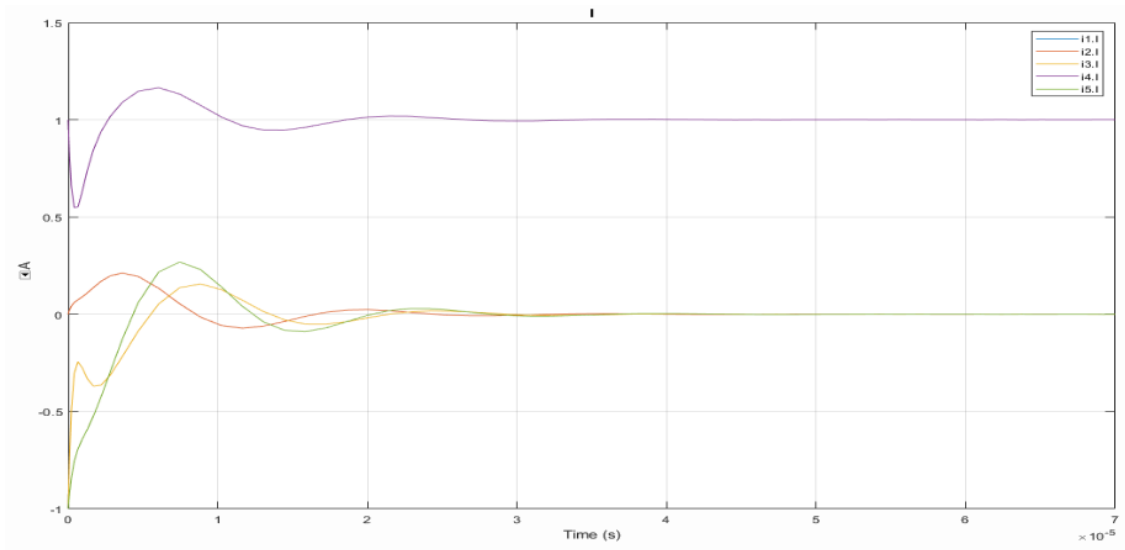


Figura 18: Correntes  $i_1$ ,  $i_2$ ,  $i_3$ ,  $i_4$  e  $i_5$

# Modelagem de um motor de corrente contínua

Foi realizada a modelagem do motor DC abaixo, a fim de simular a corrente elétrica e velocidade no motor:

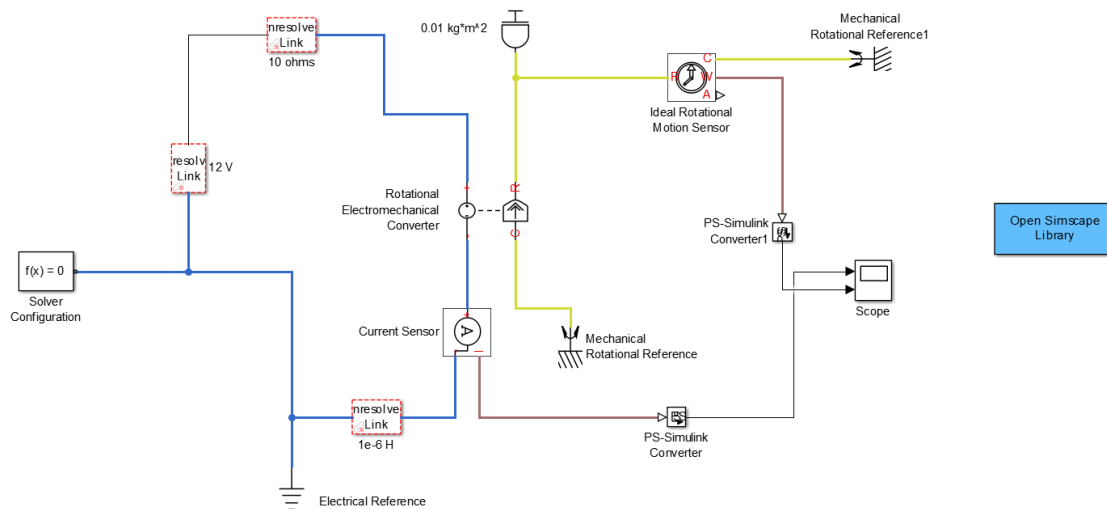


Figura 19: Circuito RLC

Para tanto, foi integrado o seguinte sistema de equações diferenciais ordinárias:

$$\begin{cases} L \frac{di}{dt} + Ri + K &= V \\ I \frac{d\omega}{dt} &= Ki \end{cases}$$

Foram utilizados os seguintes parâmetros no experimento:

- Resistência:  $10\Omega$ ;
- Tensão da fonte:  $12V$ ;
- Indutância:  $1mH$  (milihenrie);
- Inércia do sistema:  $10^{-2} \text{ kgm}^2$
- Razão Tensão/Frequência do motor:  $0.1 \text{ V/rad/s}$

Obteve-se a seguinte saída:

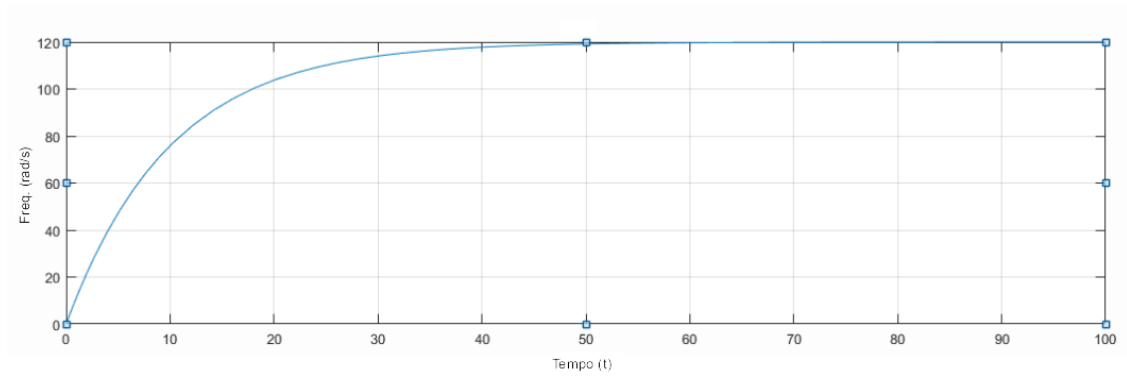


Figura 20: Frequência do motor DC no experimento

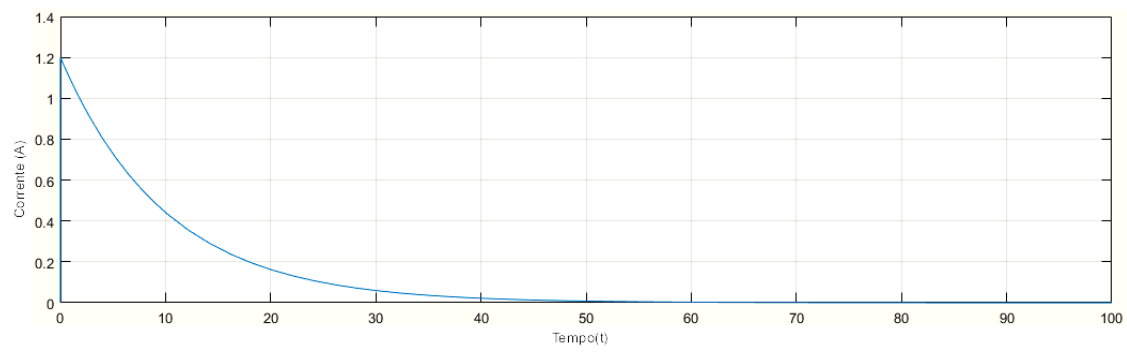


Figura 21: Corrente elétrica através motor DC no experimento