Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia

TRABALHO - IDENTIFICAÇÃO DE SISTEMAS

Raphael Timbó Silva

Professor: Daniel Castello

Rio de Janeiro

Janeiro de 2017

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

FIR	Finite Impulse Response, p. 1
FRF	Função de Resposta em Frequência, p. 2
SNR	Signal to Noise Ratio, p. 4

Capítulo 1

Introdução

O presente trabalho tem por objetivo apresentar os resultados e conclusões referentes ao projeto final da disciplina Identificação de Sistemas.

O trabalho consiste na análise de um sistema através do projeto de um filtro adaptativo FIR (Finite Impulse Response).

1.1 Sistema utilizado

O sistema utilizado é mostrado na fig. 1.1.

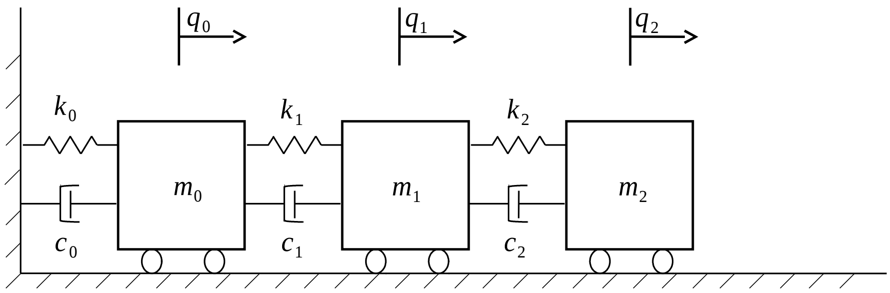

Figura 1.1: Sistema utilizado na análise.

Para este sistema temos que a energia cinética é:

$$T = \frac{1}{2}[m_0\dot{q_0}(t)^2 + m_1\dot{q_1}(t)^2 + m_2\dot{q_2}(t)^2] = \frac{1}{2}\dot{\mathbf{q}}^T(t)M\dot{\mathbf{q}}(t) \tag{1.1}$$

onde

$$\mathbf{q(t)} = [q_0(t)\ q_1(t)\ q_2(t)]^T$$

é o vetor de configuração e

$$M = \begin{bmatrix} m_0 & 0 & 0 \\ 0 & m_1 & 0 \\ 0 & 0 & m_2 \end{bmatrix}$$

é a matriz de massa do sistema.

A energia potencial tem a expressão:

$$\begin{aligned} V &= \frac{1}{2}[k_0 q_0(t)^2 + k_1(q_1(t) - q_0(t))^2 + k_2 q_2(t)^2] \\ &= \frac{1}{2}[(k_0 + k_1)q_0(t)^2 + (k_1 + k_2)q_1(t)^2 + (k_2)q_2(t)^2 - 2k_1 q_0(t)q_1(t) - 2k_2 q_2(t) \\ &= \frac{1}{2}\dot{\mathbf{q}}^T(t)K\dot{\mathbf{q}}(t) \end{aligned} \tag{1.2}$$

onde

$$K = \begin{bmatrix} k_0 + k_1 & -k_1 & 0 \\ -k_1 & k_1 + k_2 & -k_2 \\ 0 & -k_2 & k_2 \end{bmatrix}$$

é a matriz de rigidez do sistema.

Para o sistema utilizado temos que $m_i = 1\ kg$ e $k_i = 1600\ N/m$.

O amortecimento utilizado será o proporcional: $C = \alpha M + \beta K$. Iremos analisar o caso em que $\alpha = 10^{-3}$ e $\beta = 10^{-3}$.

1.2 Resposta do sistema

O sistema em questão possui a resposta FRF (Função de Resposta em Frequência) apresentada na fig. 1.2

Para nossa análise iremos considerar uma força aplicada na massa 2 (m_2) e a medição nesta mesma massa, conforme ilustrado na fig. 1.3. A aplicação da força nessa massa corresponde à FRF que pode ser visualizada no canto inferior direito (input=2 e output=2). A FRF em questão é também mostrada na fig. 1.4

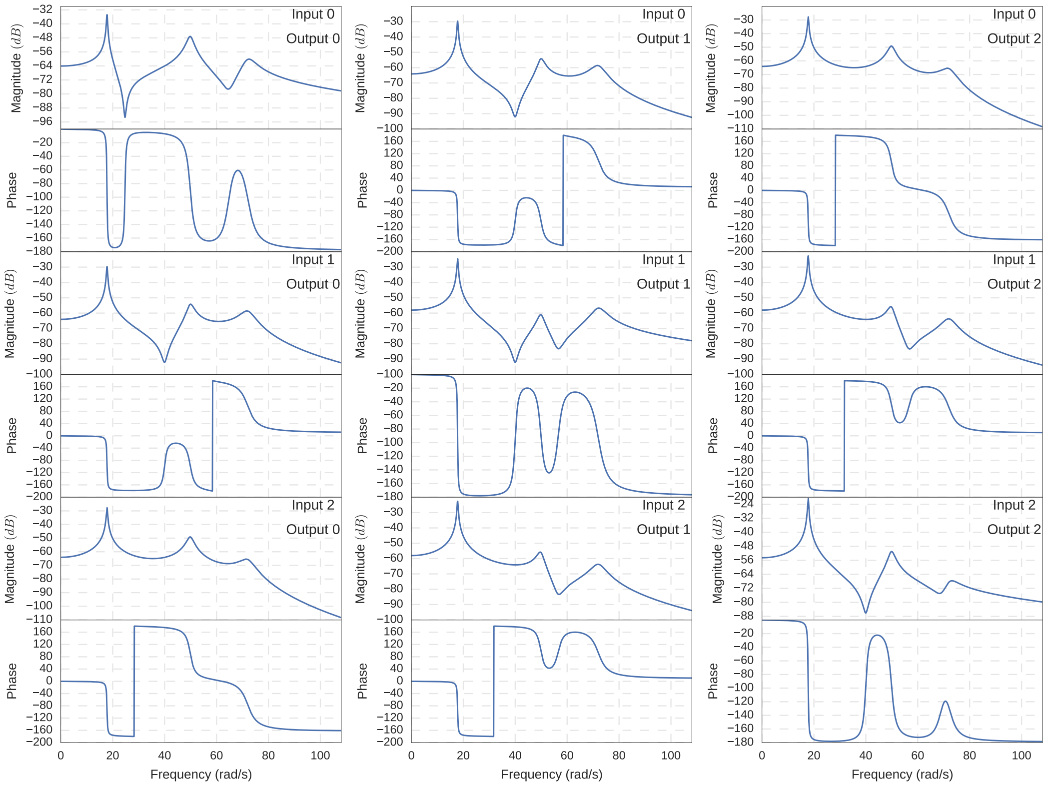

Figura 1.2: FRF para o sistema em análise.

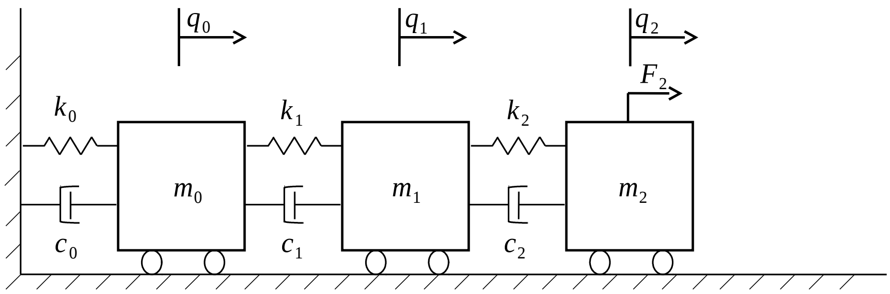

Figura 1.3: Aplicação de força e medição na massa m_2.

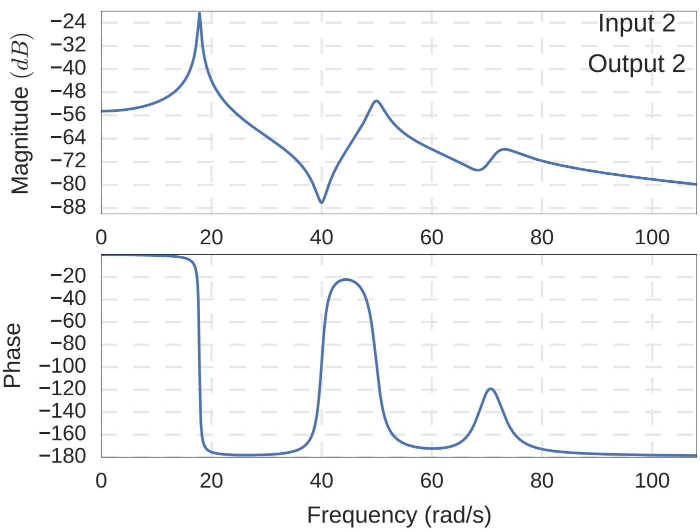

Figura 1.4: FRF para input em m_2 e medição em m_2.

Capítulo 2

Dados Pseudo-Experimentais

2.1 Resposta do sistema no tempo

Para a construção dos dados pseudo-experimentais foram observados os seguintes casos:

Forçamento:

- $F_0(t) = A_0 sin(2\pi f_0 t)$ (Considere $\frac{\omega_1}{2\pi} \leq f_0 \leq \frac{\omega_2}{2\pi}$)
- $F_1(t) = A_1 sin(2\pi f_1 t) + A_2 sin(2\pi f_2 t)$ (Escolha $\frac{0.8\omega_1}{2\pi} \leq f_j \leq \frac{1.2\omega_2}{2\pi}$ e $A_2 = 2A_1$; $j = 1, 2$)
- $F_2(t) =$ ruído branco

Número de amostras N:

- $N = 1000$
- $N = 5000$

Valores para a relação entre sinal e ruído - SNR (Signal to Noise Ratio):

- $SNR = 90$
- $SNR = 50$
- $SNR = 10$

A fig. 2.1 mostra a posição da frequência de excitação para a aplicação da força F_0, em que uma amplitude $A_0 = 1$ foi utilizada.

A fig. 2.2 mostra a resposta no tempo do sistema ao aplicarmos a força F_0 na frequência mostrada na fig. 2.1 para uma amostragem $N = 1000$. Podemos observar que, para $N = 1000$, temos uma excitação de aproximadamente 16 segundos e ainda

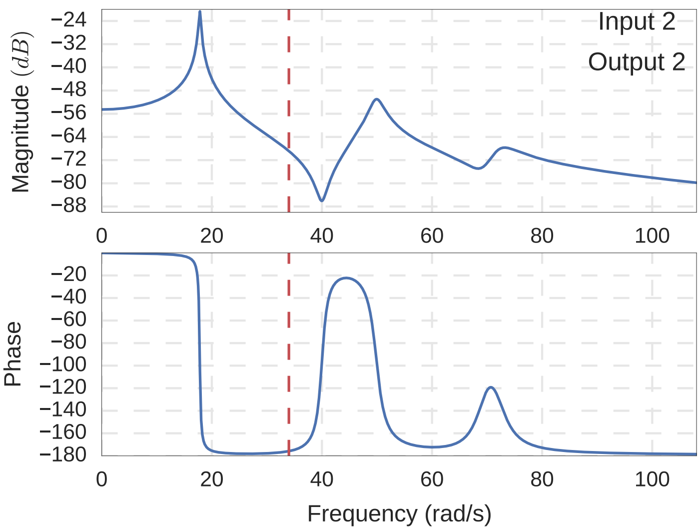

Figura 2.1: Frequência de excitação para a força F_0.

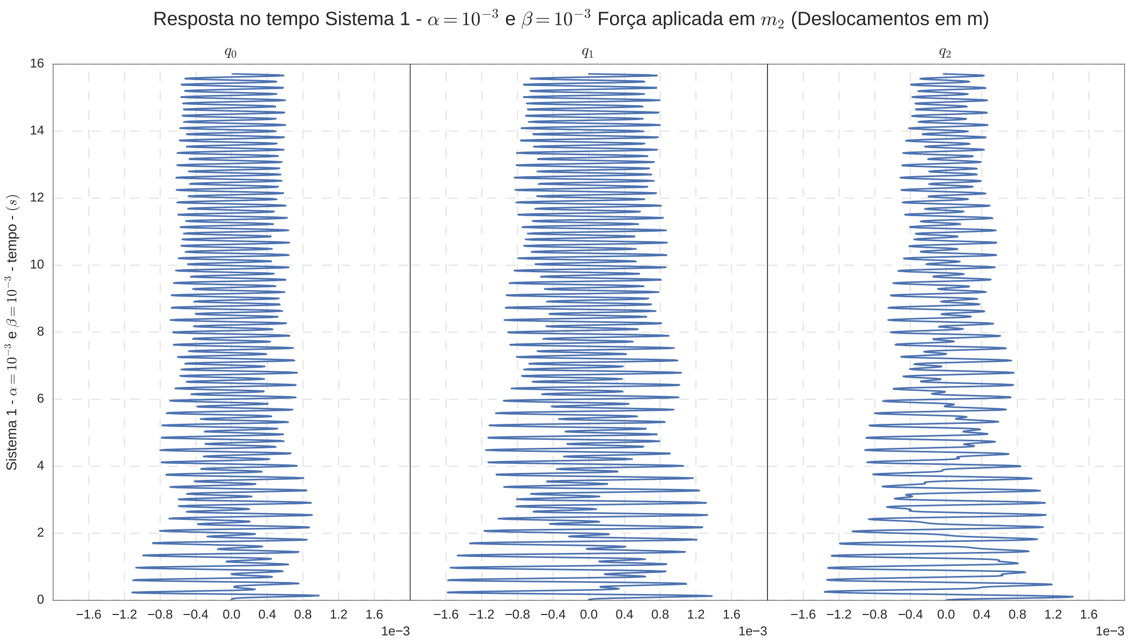

Figura 2.2: Resposta no tempo para a força F_0 com $N = 1000$.

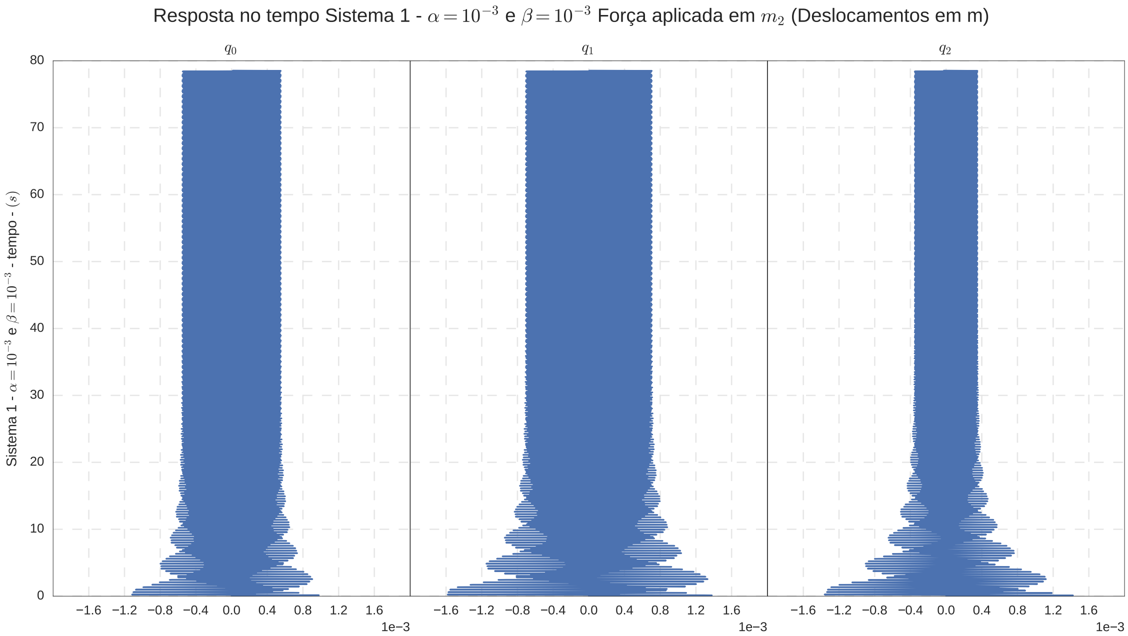

Figura 2.3: Resposta no tempo para a força F_0 com $N = 5000$.

temos algum transiente na resposta no tempo. Também é possível observar que essa parcela apresenta mais que uma frequência de oscilação.

A fig. 2.3 mostra a resposta no tempo para $N = 5000$. Neste caso, o tempo vai até aproximadamente 80 segundos e podemos observar que a parcela transiente é praticamente inexistente após os 20 segundos de excitação. Após esse tempo, é esperado que o sistema oscile apenas na frequência de excitação.

Para a força F_1 a fig. 2.4 mostra as frequências de excitação que foram aplicadas na massa m_2. Podemos notar que nesse caso as forças aplicadas estão próximas as frequências naturais do sistema.

A fig. 2.5 mostra a resposta no tempo para F_1 com $A_1 = 1$, $A_2 = 2$ e $N = 5000$. Como esperado, notamos um aumento na amplitude de 1×10^{-3} m para 1×10^{-2} m quando comparado à força F_0.

O último caso de forçamento é mostrado na fig. 2.6 onde um ruído branco com variância 1 é aplicado ao sistema.

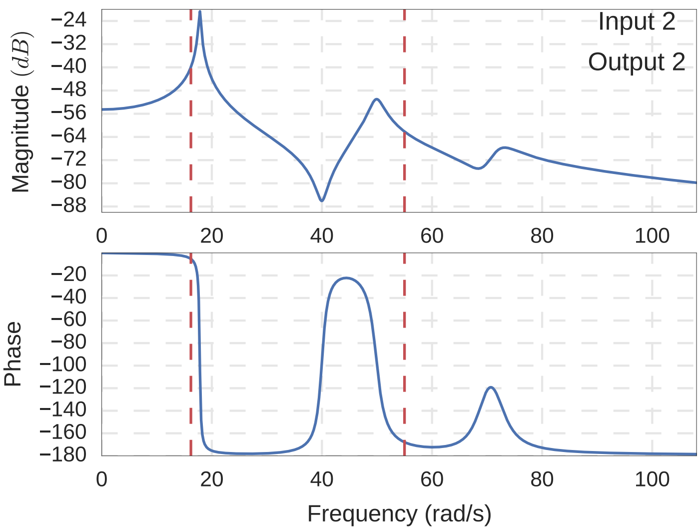

Figura 2.4: Frequência de excitação para a força F_1.

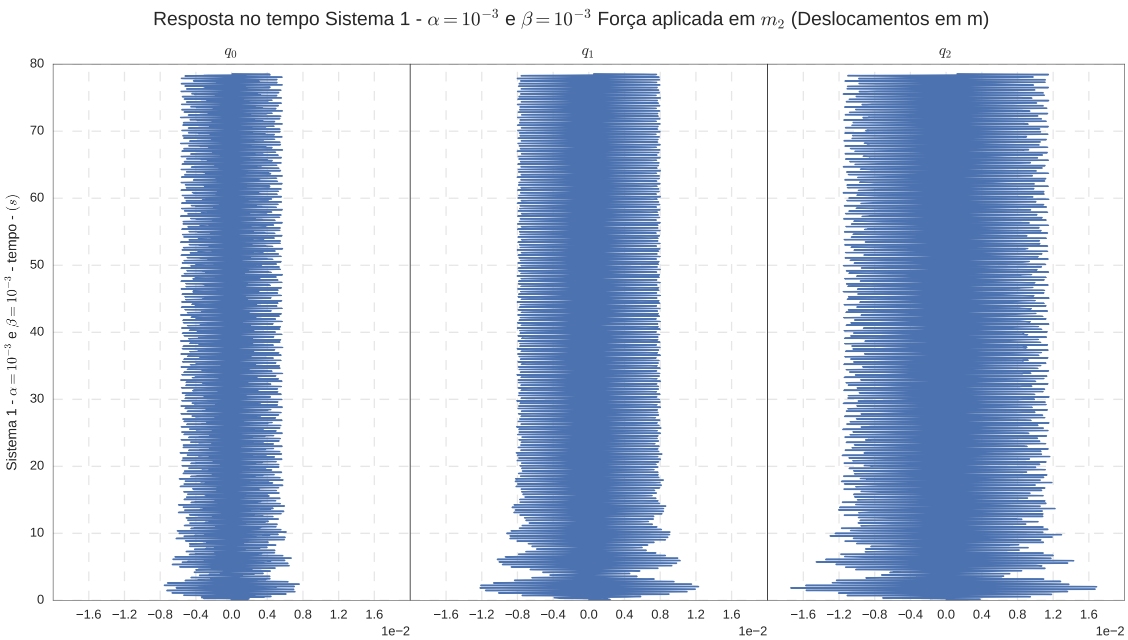

Figura 2.5: Resposta no tempo para a força F_1 com $N = 5000$.

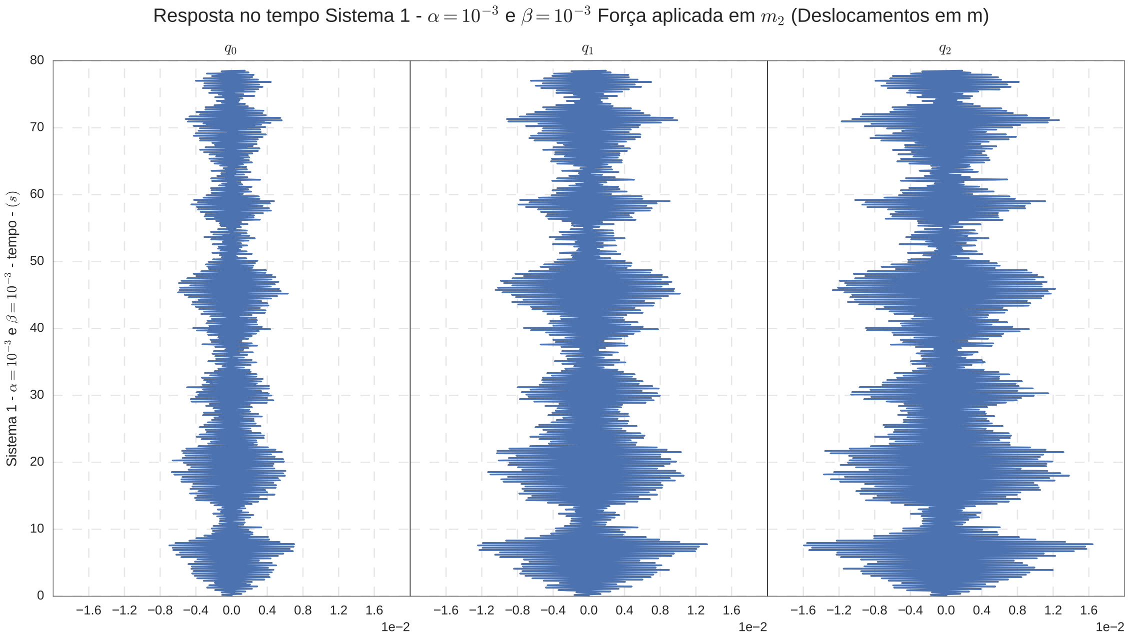

Figura 2.6: Resposta no tempo para a força F_2 com $N = 5000$.

2.2 Adição do ruído

Conforme mostrado no item 2.1, a análise será feita para três diferentes níveis de ruído ($SNR = 90$, 50, 10).

Temos então que o sinal utilizado para o projeto do filtro será:

$$y = y^{ideal} + n \tag{2.1}$$

onde n representa um ruído inserido no sinal.

Para calcularmos a amplitude do ruído inserido 'n' utilizaremos a eq. (2.2).

$$SNR = 20log_{10}\left(\frac{A_s}{A_n}\right) \rightarrow A_n = \frac{A_s}{10^{SNR/20}} \tag{2.2}$$

Abaixo (fig. 2.7, fig. 2.8 e fig. 2.9), são mostrados alguns resultados comparando o sinal puro e o sinal corrompido para um determinado nível de ruído.

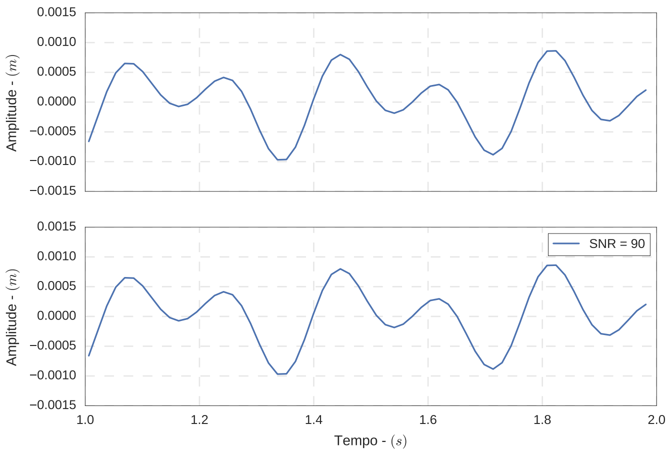

Figura 2.7: Sinal puro e sinal corrompido para F_0 e $SNR = 90$.

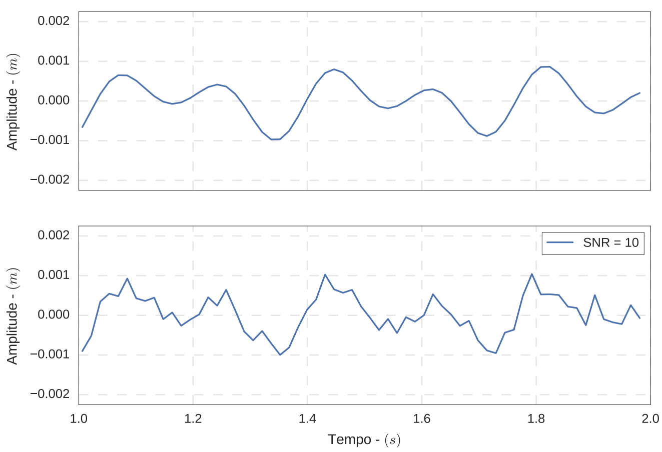

Figura 2.8: Sinal puro e sinal corrompido para F_0 e $SNR = 10$.

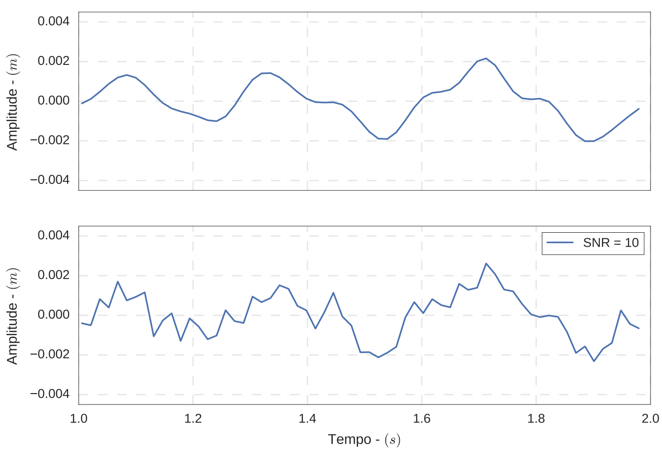

Figura 2.9: Sinal puro e sinal corrompido para F_2 e $SNR = 10$.

Capítulo 3

Projeto do Filtro Adaptativo

Para o algoritmo do filtro adaptativo CASTELLO e ROCHINHA [2] apresentam algumas opções que podem ser utilizadas. No caso do presente trabalho o algoritmo LMS (Least Mean Squares) foi escolhido.

3.1 Algoritmo LMS

Como mostrado por DINIZ [1], a configuração normalmente aplicada para identificação de sistemas com filtros adaptativos é mostrada na fig. 3.1. Nesta configuração temos que $x(k)$ é o sinal de entrada, o sinal de saída do sistema que desejamos identificar é $d(k)$ e a saída do filtro é $y(k)$. Estes sinais são comparados e o erro $e(k)$ é calculado.

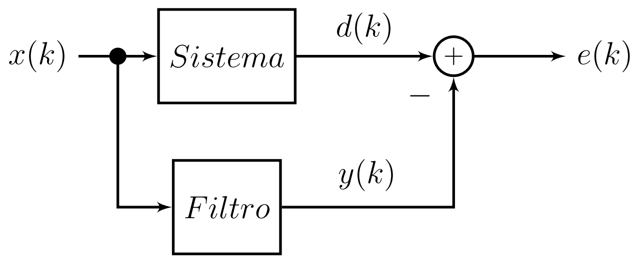

Figura 3.1: Configuração utilizada no algoritmo para filtros adaptativos.

Os valores de saída do filtro são calculados a partir de uma combinação linear dos seus coeficientes e do sinal de entrada, isto é:

$$y(k) = \sum_{i=0}^{N} w_i(k)x_i(k) = \mathbf{w}^T(k)\mathbf{x}(k) \tag{3.1}$$

onde $\mathbf{w}(k)$ representa os coeficientes do filtro.

Para aplicações onde o vetor do sinal de entrada é uma versão atrasada do mesmo sinal, isto é: $x_0(k) = x(k), x_1(k) = x(k-1), ..., x_N(k) = x(k-N)$, $y(k)$ é o resultado da aplicação de um filtro FIR ao sinal de entrada $x(k)$. Neste caso temos:

$$y(k) = \sum_{i=0}^{N} w_i(k)x(k-i) = \mathbf{w}^T(k)\mathbf{x}(k) \tag{3.2}$$

onde $\mathbf{x}(k) = [x(k)\ x(k-1)\ ...\ x(k-N)]^T$.

O MSE (Mean Square Error) pode ser calculado como:

$$\xi(k) = E[e^2(k)] = E[d^2(k) - 2d(k)y(k) + y^2(k)] \tag{3.3}$$

Podemos reescrever a eq. (3.3):

$$\xi = E[d^2(k)] - 2\mathbf{w}^T\mathbf{p} + \mathbf{w}^T\mathbf{R}\mathbf{w} \tag{3.4}$$

onde $p = E[d(k)\mathbf{x}(k)]$ é o vetor de correlação cruzada entre o sinal desejado e o sinal de entrada e $\mathbf{R} = E[\mathbf{x}(k)\mathbf{x}^T(k)]$ é a matriz de correlação do sinal de entrada.

Se $\mathbf{p}$ e $\mathbf{R}$ são conhecidos, podemos encontrar a solução para $\mathbf{w}$ que minimiza ξ.

O gradiente do MSE relativo aos coeficientes é:

$$\mathbf{g_w} = \frac{\partial \xi}{\partial \mathbf{w}} = \left[\frac{\partial \xi}{\partial w_0}\frac{\partial \xi}{\partial w_1}...\frac{\partial \xi}{\partial w_N}\right] = -2\mathbf{p} + 2\mathbf{R}\mathbf{w} \tag{3.5}$$

Igualando o gradiente a zero encontramos o vetor de coeficientes que minimiza ξ:

$$\mathbf{w}_0 = \mathbf{R}^{-1}\mathbf{p} \tag{3.6}$$

Se boas estimativas $\hat{\mathbf{p}}$ e $\hat{\mathbf{R}}$ estão disponíveis podemos buscar uma solução:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu\hat{\mathbf{g}}_{\mathbf{w}}(k) \tag{3.7}$$

$$= \mathbf{w}(k) + 2\mu(\hat{\mathbf{p}}(k) - \hat{\mathbf{R}}(k)\mathbf{w}(k)) \tag{3.8}$$

para $k = 0, 1, 2, ...$, onde $\hat{\mathbf{g}}_{\mathbf{w}}(k)$ representa uma estimativa do gradiente da função objetivo com respeito aos coeficientes do filtro.

Uma possível solução é estimar o gradiente com estimativas instantâneas de $\mathbf{R}$ e de $\mathbf{p}$.

$$\hat{\mathbf{R}}(k) = \mathbf{x}(k)\mathbf{x}^T(k) \tag{3.9}$$

$$\hat{\mathbf{p}}(k) = d(k)\mathbf{x}(k) \tag{3.10}$$

Temos então que a estimativa para o gradiente é:

$$\hat{\mathbf{g}}_{\mathbf{w}}(k) = -2e(k)\mathbf{x}(k) \tag{3.11}$$

Chegamos então ao algoritmo LMS em que a equação de atualização é:

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu e(k)\mathbf{x}(k) \tag{3.12}$$

onde o fator de convergência μ deve ser escolhido em um range que garanta a convergência.

3.2 Implementação do filtro

O filtro será implementado em Python como explicado abaixo.

Uma classe `class LMSFilter()` será criada para o filtro. Essa classe contém uma função de inicialização que possui como argumentos de entrada o número de coeficientes do filtro e o fator de convergência μ. Na criação do filtro os coeficientes são igualados a zero como mostrado no código abaixo.

```
class LMSFilter(object):
  def __init__(self, Nc, mu):
    """
    Iniciar filtro com Nc coeficientes.
    """
    self.Nc = Nc
    self.mu = mu
    # valores iniciais para o filtro w = [0, 0, ..., 0]
    self.w = np.zeros(Nc)
```

O filtro possui uma função `predict(self, x)` que calcula a saída prevista para o filtro baseado no sinal de entrada `x` ($x(k)$). O cálculo é feito conforme a eq. (3.1):

$$y(k) = \sum_{i=0}^{N} w_i(k)x_i(k) = \mathbf{w}^T(k)\mathbf{x}(k) \tag{3.1 revisitada}$$

```
def predict(self, x):
  y = self.w @ x
  return y
```

A atualização do filtro é feita pela função `update(self, d, x)` considerando a eq. (3.12):

$$\mathbf{w}(k+1) = \mathbf{w}(k) + 2\mu e(k)\mathbf{x}(k) \tag{3.12 revisitada}$$

```python
def update(self, d, x):
    """
    Atualizar filtro baseado no sinal de entrada x
    e no valor desejado d.
    """
    y = self.w @ x
    e = d - y
    self.w += 2*self.mu * e * x
```

Capítulo 4

Resultados e Discussões

Assim como mostrado por CASTELLO e ROCHINHA [2], no caso analisado o fenômeno de anti-ressonância também não foi capturado pelo LMS.

Capítulo 5

Conclusões

Referências Bibliográficas

[1] DINIZ, P. S. *Adaptive filtering.* Springer, 1997.

[2] CASTELLO, D. A., ROCHINHA, F. A. "An experimental assessment of transverse adaptive fir filters as applied to vibrating structures identification", *Shock and Vibration*, v. 12, n. 3, pp. 197–216, 2005.

Apêndice A

Algumas Demonstrações