Universidade Federal do ABC

Graduação em Engenharia de Informação

Lucas Toshimi Freitas Hossoda

CANCELAMENTO ATIVO DE RUÍDOS UTILIZANDO ALGORITMOS ADAPTATIVOS

Monografia apresentada ao Curso de Engenharia de Informação da Universidade Federal do ABC, como requisito parcial da disciplina de Trabalho de Graduação III

Orientador: Prof. Dr. Kenji Nose Filho

Santo André – SP

2019

Lucas Toshimi Freitas Hossoda

CANCELAMENTO ATIVO DE RUÍDOS UTILIZANDO ALGORITMOS ADAPTATIVOS

Monografia apresentada como requisito parcial à obtenção de título de Graduação em Engenharia de Informação.

Orientador : Profº Dr. Kenji Nose Filho

Santo André – SP

2019

RESUMO

A exposição ao ruído sonoro pode afetar de diversas formas o bem-estar de um indivíduo, podendo acarretar em dores de cabeça, desconforto e até na perda auditiva. Uma das fontes de ruído presentes no cotidiano de muitas pessoas são geradas por dutos de ventilação. Neste trabalho, foram avaliados alguns métodos para a implementação de um sistema de Controle Ativo de Ruído (CAR) em dutos de ventilação, através de algoritmos adaptativos least mean-squares (LMS), Normalized least mean-squares (NLMS) e Sign-error least mean-squares. As simulações numéricas mostraram que o algoritmo LMS apresentou menor média de erro, enquanto o NLMS apresenta a melhor resposta de convergência em termos de velocidade.

Palavras-chave: Algoritmo adaptativo. LMS. Controle Ativo de Ruído.

ABSTRACT

The exposure to acoustic noise can affect a person's well-being in several ways, leading to headaches, discomfort and even hearing loss. One of the sources of noise most present in the daily lives of many people comes from ventilation systems. In this work, we evaluated some methods for the implementation of an Active Noise Control (ANC) system in ventilation ducts, using adaptive algorithms such as the least-mean-squares (LMS), the Normalized least-mean-squares (NLMS) and the Sign-Error least-mean-squares. The LMS algorithm displays the smallest average error, whilst the NLMS algorithm displays the best convergence response in regard to velocity.

Keyword:Adaptive Algorithm.LMS.Active Noise Control.

Sumário

1 INTRODUÇÃO

Em um sinal de áudio, o ruído pode ser definido como um som que produz uma sensação auditiva desagradável. Recentemente, com o avanço da tecnologia, diversas fontes de ruídos foram introduzidas no nosso cotidiano como, por exemplo, os ruídos gerados por carros, máquinas de construção, aparelhos eletrodomésticos etc.

Sabe-se que o bem-estar e a saúde de um indivíduo está diretamente ligada com o ambiente em que ele está inserido e que passa a maior parte de seu tempo. Para grande parte da população, este ambiente é o local de trabalho. No ambiente de trabalho o ruído pode causar estresse por dificultar a comunicação, interferir na concentração de tarefas e pode estar na origem de diversos problemas de saúde crônicos. A perda auditiva laboral é uma das doenças profissionais mais comuns segundo (Bureau Internacional do Trabalho, 2009).

Recentemente, grandes esforços vêm sendo adotados para que o índice de qualidade de vida aumente, principalmente através de novos produtos que possam agregar valor à saúde e a comodidade de um indivíduo.

Com isso em mente, o objetivo deste trabalho é estudar técnicas de controle ativo de ruído (CAR) em dutos de ventilação, pois no local de trabalho, uma das fontes de ruído pode vir especificamente dos dutos de ventilação. O controle ativo de ruído (CAR) é uma técnica relativamente pouco difundida, mas que possui grandes vantagens em relação ao controle passivo de ruído.

O controle de ruído passivo utiliza-se de materiais que servem de barreiras acústicas e silenciadores. São eficientes para uma larga banda de frequência, porém são caros, volumosos e ineficazes em baixas frequências (Kuo e Morgan, 1996).

Por outro lado, técnicas de controle ativo de ruído apresentam certas vantagens com relação às técnicas de controle passivo por possuirem alto desempenho em baixas frequências e volume reduzido (Riyanto,2007).

O conceito de controle ativo de ruído foi proposto pela primeira vez por Lueg (1936), sendo o primeiro a expor a possibilidade de atenuação de ruído pelo teorema da

Superposição de ondas, utilizando uma fonte de sinal com mesma amplitude e fase inversa à do sinal de ruído. No entanto, foram Olson e May (1953) que implementaram de fato o modelo proposto por Lueg (1936), utilizando um amplificador inversor de alto ganho como um controlador para o sinal de ruído, além de dispor alto-falante próximos a um microfone, que atuaria como um sensor.

Este trabalho tem como objetivo estudar e implementar algumas técnicas de Controle Ativo de Ruído, apresentando simulações numéricas com o auxílio do software MATLAB®, com o intuito de avaliar a eficácia de tais sistemas como forma de atenuação de ruídos provenientes de dutos de ventilação.

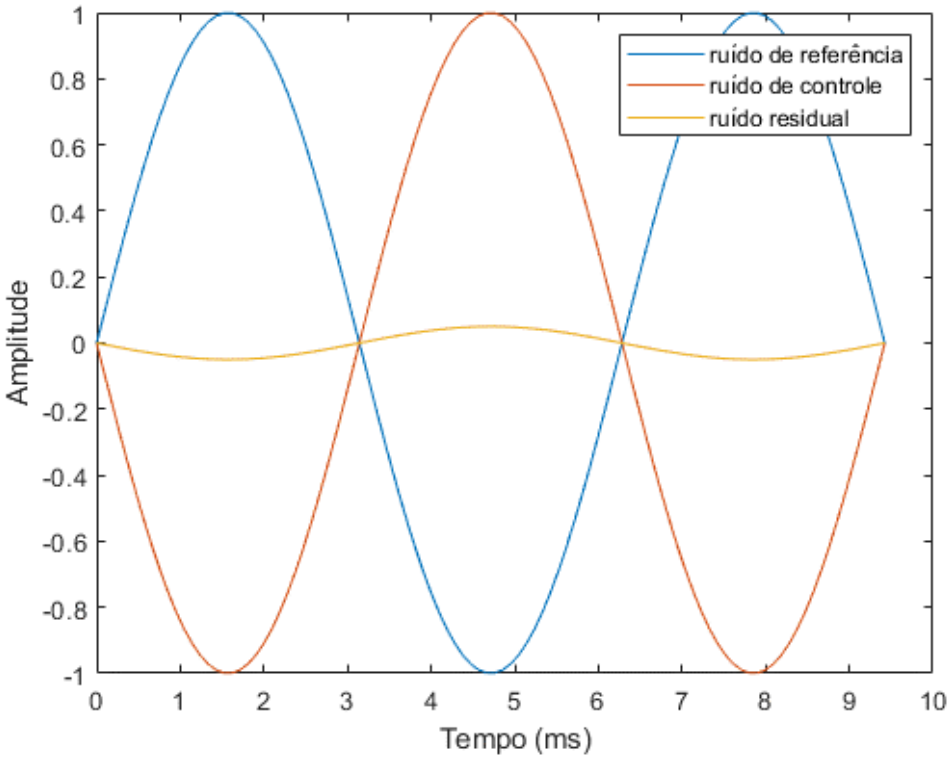
Na primeira etapa do trabalho, foi revisada a bibliografia relacionada aos modelos de sistemas de controle ativo de ruído, de forma a obter conhecimento sobre diferentes tipos de sistemas. Já na segunda etapa do trabalho, buscou-se implementar um destes sistemas, baseado no algoritmo adaptativo do tipo LMS (do inglês *Least-Mean-Squares*) e realizar algumas simulações de modo a verificar o seu funcionamento (SAYED, 2011).

Na etapa final do trabalho, foram realizadas simulações com a inclusão de um caminho secundário, um conceito importante neste tipo de cancelamento ativo de ruído, para diferentes tipos de algoritmos adaptativos, sendo estes o LMS, o Normalized LMS (NLMS) e o Sign-error LMS. O desempenho dos algoritmos é avaliado com relação à taxa de convergência e do erro residual.

2 FUNDAMENTAÇÃO TEÓRICA

A técnica do controle ativo de ruído parte do princípio da superposição de ondas, ou seja, aplica-se uma onda de mesma amplitude e fase oposta ao sinal que se deseja cancelar de forma que ocorra interferência destrutiva das ondas sonoras, conforme é ilustrado pela Figura 1.

Figura 1 - Ilustração da interferência destrutiva que resulta em um sinal residual.



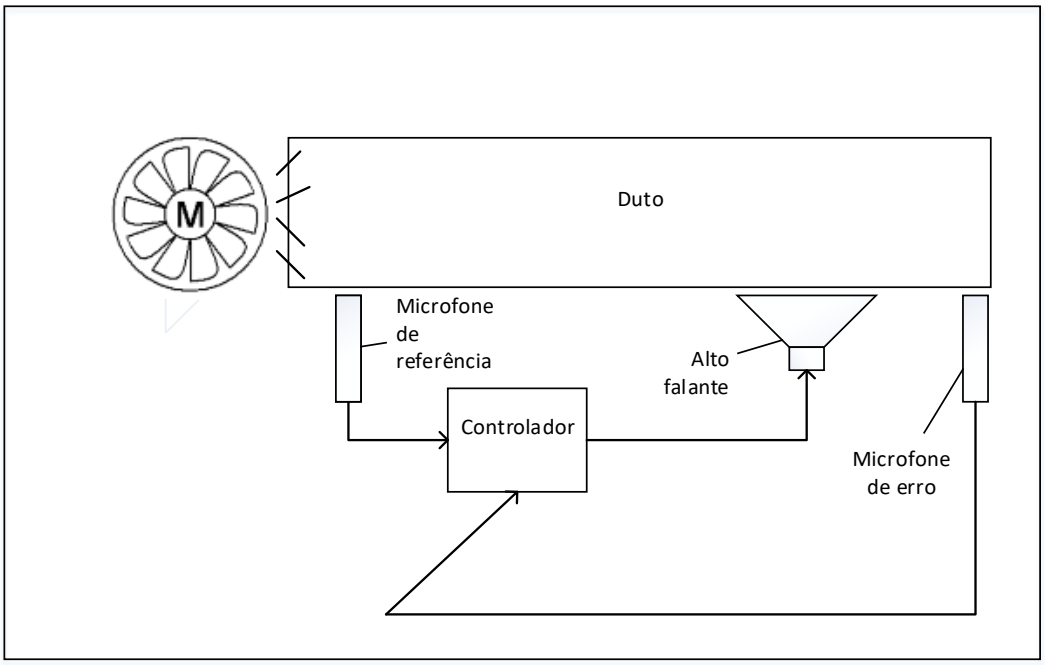
Os sistemas de controle ativo de ruído podem ser analisados de diferentes formas. De forma prática, como no estudo realizado por Lessa (2010) através de ensaios em dutos reais utilizando técnicas de filtragem adaptativa, ou através de um software de simulação numérica como o MATLAB, como realizado no trabalho de Almeida Junior (2014) e que será adotado neste trabalho.

Como ponto de partida, foi avaliado o modelo acústico de um sistema de controle ativo de ruído, o qual tem como principais componentes [Nuñez, 2005]:

- Sensores: geralmente são utilizados microfones para captar o sinal de ruído que é transmitido através do duto, e que será utilizado como parâmetro para a fonte secundária.
- Planta: o ambiente a ser estimado. No caso, o duto de ventilação.
- Atuadores: dispositivos eletroacústicos como, por exemplo, um alto-falante.
- Controlador: controle que implementa o algoritmo computacional.

A Figura 2 apresenta o esquema de um sistema de controle ativo de ruído.

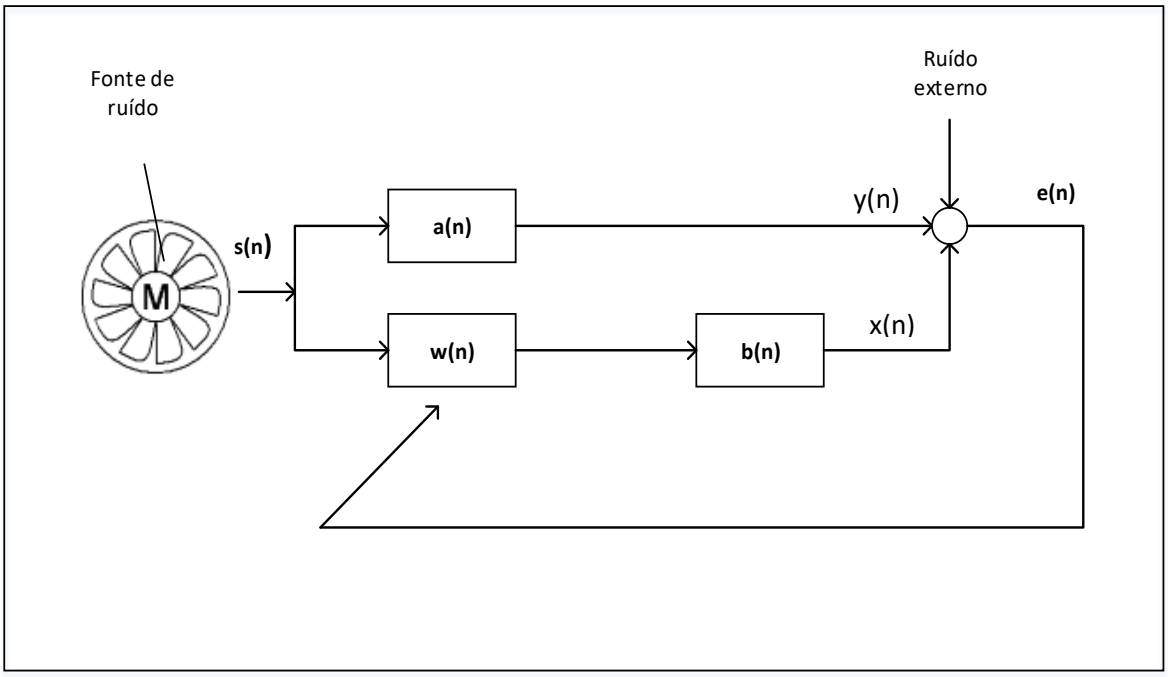
Figura 2. Diagrama de um sistema CAR do tipo *feedforward*.



Neste diagrama, o microfone de referência é responsável pela captação do sinal de ruído $s(n)$ proveniente da fonte sonora, ou seja, do ventilador e que irá alimentar o nosso controlador. Este ruído também percorre o duto e sofre uma variação. A distorção causada pelo duto é modelada através de um sistema linear e invariante no tempo (ZIMMER, 2003). Desta forma, o ruído percebido no final do duto é dado pela convolução de $s(n)$ com a resposta ao impulso $a(n)$ do duto de ventilação, resultando no sinal $x(n)$. O alto falante é responsável pela geração do sinal $y(n)$ que irá se contrapor ao sinal $x(n)$. O sinal resultante $e(n)$, também chamado de sinal de erro, é captado pelo microfone de erro, o qual será utilizado para ajustar o controlador que alimenta o alto-falante. Em geral, como o alto falante nunca é posicionado próximo ao microfone de erro, o som gerado por ele também sofre uma distorção ocasionado pelo

duto. Esta distorção é modelada pela resposta ao impulso $b(n)$ do que chamamos de canal secundário, enquanto que $a(n)$ representa a resposta ao impulso do chamado canal primário. Esta representação pode ser feita através do diagrama de blocos da Figura 3.

Figura 3. Diagrama de blocos de um sistema CAR do tipo feedforward.



O sinal de erro $e(n)$ – na ausência de ruído externo – pode ser expresso na forma:

$$e(n) = x(n) - y(n), \tag{1}$$

onde $x(n) = s(n) * a(n)$ e $y(n) = s(n) * w(n) * b(n)$.Ou ainda, pela propriedade de comutatividade, podemos escrever $y(n) = \hat{s}(n) * w(n)$, onde $\hat{s}(n) = s(n) * b(n)$ e $w(n)$ é dado pela resposta ao impulso de um filtro do tipo FIR (do inglês *Finite Impulse Response*), que será alimentado pelo microfone de referência e cuja saída será utilizada para alimentar o alto falante. Para obter os coeficientes deste filtro será utilizado o algoritmo LMS, que pode ser visto como uma versão estocástica do filtro de Wiener e que busca minimizar de forma adaptativa o critério MSE (do inglês *Mean Squared Error*).

2.1 - O filtro de Wiener e o algoritmo LMS

Seja o sinal

$$y(\mathrm{n}) = \sum_{k=0}^{K-1} w_k \, \hat{s}(n-k) = \boldsymbol{w}^T \hat{\boldsymbol{s}}(n), \tag{2}$$

onde $\boldsymbol{w} = [w_0 \dots w_{K-1}]^T$ é o vetor contendo os coeficientes do filtro, $\hat{\boldsymbol{s}}(n) = [s_0 \dots s_{K-1}]^T$ é o vetor contendo o sinal de entrada amostrado. Podemos definir o critério MSE como:

$$J_{MSE}(\boldsymbol{w}) = E[e^2(n)] = E\left[\left(x(n) - \boldsymbol{w}^T \hat{\boldsymbol{s}}(n)\right)^2\right]. \tag{3}$$

Observe que no sistema CAR da Figura 3, pela propriedade da comutatividade dos sistemas lineares, $\hat{s}(n) = s(n) * b(n)$, ou seja, no caso em que $b(n) = \delta(n)$, temos que $\hat{s}(n) = s(n)$. Uma das formas de se obter o mínimo da função custo acima, em termos do vetor $\boldsymbol{w}$,é igualando o gradiente da função custo a zero, gerando a seguinte condição:

$$E[e(n)\hat{s}(n)] = 0. \tag{4}$$

Desenvolvendo as equações (3) e (4), temos que (ROMANO, 2011):

$$J_{MSE}(\boldsymbol{w}) = \sigma_x^2 - \boldsymbol{p}^T \boldsymbol{w} - \boldsymbol{w}^T \boldsymbol{p} + \boldsymbol{w}^T \boldsymbol{R} \boldsymbol{w}, \tag{5}$$

onde σ_x^2 é a variância do sinal $x(n)$, supondo que o sinal x(n) tem média nula, $\boldsymbol{R}$ é a matriz de autocorrelação de $\hat{s}(n)$ e $\boldsymbol{p}$ é o vetor de correlação cruzada entre $\hat{s}(n)$ e $x(n)$ e o filtro ótimo é dado por:

$$\boldsymbol{w}_{wiener} = \boldsymbol{R}^{-1} \boldsymbol{p}, \tag{6}$$

também conhecido como filtro de Wiener (ROMANO, 2011).

Em um contexto adaptativo, a solução de Wiener pode ser obtida através do algoritmo LMS. O algoritmo LMS busca minimizar o erro quadrático médio instantâneo de forma adaptativa, baseado no algoritmo do gradiente descendente. sua regra de adaptação é dada por (SAYED, 2011):

$$\boldsymbol{w}(n+1) = \boldsymbol{w}(n) - \mu e(n) \hat{\boldsymbol{s}}(n), \tag{7}$$

onde o parâmetro μ é conhecido como passo de adaptação, responsável pela velocidade de convergência do algoritmo. Valores altos podem acelerar a convergência do algoritmo, no entanto levam a valores maiores do erro residual. Já valores baixos deste parâmetro nos leva a uma convergência mais lenta, mas a um erro residual no final da convergência também menor.

2.2 - O algoritmo NLMS

Outro algoritmo que foi avaliado neste trabalho, foi o NLMS, cuja regra de adaptação é dada por (Haykin, 1996):

$$\widehat{\boldsymbol{w}}(n) = \widehat{\boldsymbol{w}}(n-1) + \frac{\mu}{\left||\hat{\boldsymbol{s}}(n)|\right|^2 + \alpha}\hat{\boldsymbol{s}}(n)e(n), \tag{8}$$

onde α é um parâmetro positivo pequeno, de modo a evitar a instabilidade do algoritmo para o caso em que $\left||\hat{\boldsymbol{s}}(n)|\right|^2 = 0$. Este algoritmo também visa a minimização do critério MSE através da minimização do erro quadrático instantâneo. No entanto, através da equação (8) é possível notar o motivo que leva o nome do algoritmo de LMS normalizado, em tradução livre. O produto do vetor $\hat{\boldsymbol{s}}(n)e(n)$ é normalizado em respeito a norma euclidiana quadrática do vetor de entrada $\hat{\boldsymbol{s}}(n)$.

A vantagem deste algoritmo, quando comparado com o LMS é a sua velocidade de convergência. No entanto, apesar de apresentar uma convergência mais rápida que a do LMS, o seu erro residual tende a ser um pouco maior, como demonstrado por Tarrab e Feuer (1988).

2.3 – O algoritmo Sign-error LMS

Além do LMS e do NLMS também foi testado o algoritmo Sign-error LMS. Em comparação com estes algoritmos, o Sign-error LMS busca a minimização do erro médio absoluto, dado por E[e(n)], através da minimização do erro absoluto instantâneo, e a sua regra de adaptação é dada por (Sayed, 2011):

$$w(n+1) = w(n) + \mu\, sign[e(n)]\hat{\boldsymbol{s}}(n) \tag{9}$$

Onde a função *sign* é definida por:

$$sign[x] = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ 0 & x = 0 \end{cases}$$

3 – RESULTADOS

Nesta seção serão apresentados resultados obtidos em simulações para análise de convergência de acordo com o passo de adaptação nas equações (1) e (2) as simulações são feitas para três algoritmos distintos implementando um modelo mais completo, que considera o caminho secundário e ruído externo.

3.1 – Análise de convergência do algoritmo LMS para diferentes passos de adaptação

Resultados iniciais foram obtidos através do algoritmo LMS para um caso bastante simples, onde será desconsiderada a presença de ruídos externos e também será desconsiderado a distorção ocasionada pelo caminho secundário, ou seja, será considerado o caso em que $b(n) = \delta(n)$.

Os coeficientes utilizados para o modelo do duto acústico (caminho primário) utilizado nesta etapa foram extraídos de um exemplo de uso de uma toolbox do Matlab[1].

O modelo do caminho primário é um filtro passa-banda do tipo Chebyshev tipo 2 com os seguintes atributos:

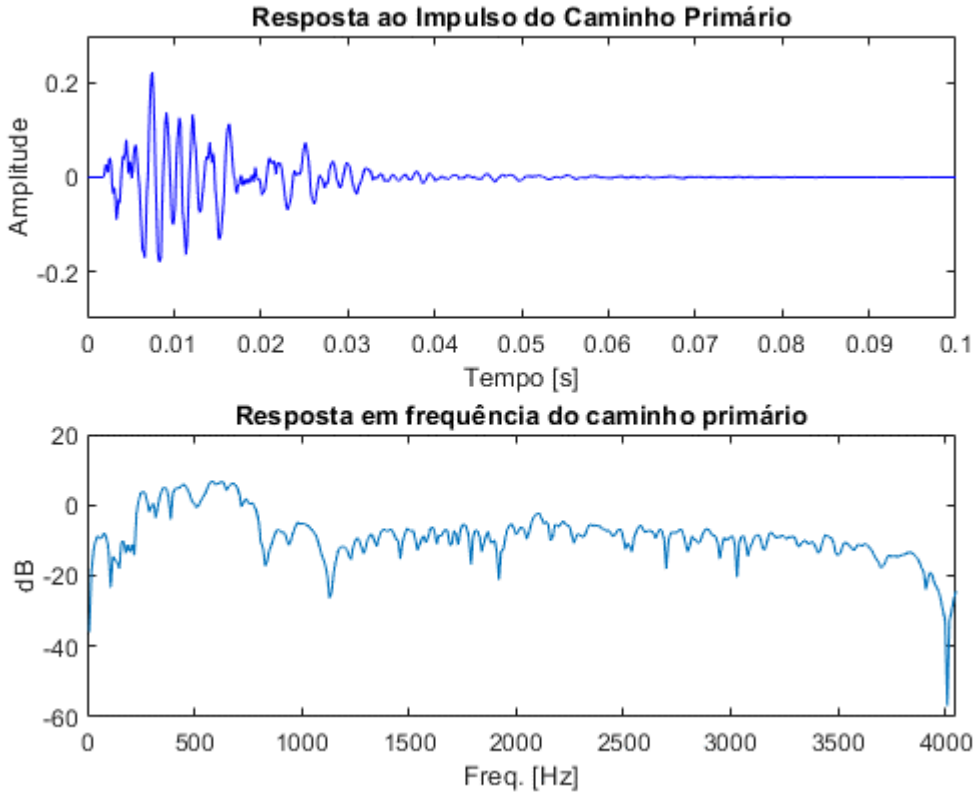
Tabela 1. Atributos do filtro utilizado como modelo do caminho primário.

Nº de coeficientes	N = 800;
Frequência limite da banda passa-baixas (Hz)	Flow = 200;
Frequência limite da banda passa-alta (Hz)	Fhigh = 800;
Ordem do filtro	Ordem do filtro = 10
Atraso de fase (amostras)	delayW = 15;
Taxa de amostragem	8 kHz

A resposta ao impulso do caminho primário e a sua resposta em frequência são apresentados na Figura 4.

[1] Disponível em https://www.mathworks.com/help/audio/examples/active-noise-control-using-a-filtered-x-lms-fir-adaptive-filter.html

Figura 4. Resposta ao Impulso e Resposta em frequência do caminho primário que modela o duto acústico utilizado neste experimento.



Para a modelagem do sinal de entrada $s(n)$, foi utilizado um sinal senoidal na forma:

$$s(n) = \sum_{i=1}^{12} A(i).\cos(2.\pi.i.F_0 n + \phi(i)) ,$$

onde $A(i) = [0{,}01\ 0{,}01\ 0{,}02\ 0{,}2\ 0{,}3\ 0{,}4\ 0{,}3\ 0{,}2\ 0{,}1\ 0{,}07\ 0{,}02\ 0{,}01]$ e $\phi(i) = N(0{,}1)$ é uma variável aleatória Gaussiana de média nula e variância unitária. Foi utilizado uma frequência inicial $F_0 = 60Hz$, um total de 160000 amostras (20s) e uma taxa de amostragem igual a 8kHz.

Na primeira simulação, escolheu-se arbitrariamente um passo de adaptação $\mu = 0{,}01$ e $K = 800$, onde K é o número de coeficientes do filtro adaptativo. O vetor $\boldsymbol{w}(0)$ é inicializado com valores nulos.

Para a análise dos sinais no domínio do tempo, utilizou-se um quadro de 120 ms iniciais e 120 ms finais da janela de 20s, obtendo-se as Figuras 5 e 6.

Figura 5. Sinal $s(n)$ na entrada do duto acústico, sinal $x(n)$, sinal $y(n)$ gerado pelo alto falante e sinal $e(n) = x(n) - y(n)$ na saída do duto nos 120ms iniciais para $\mu = 0{,}01$.

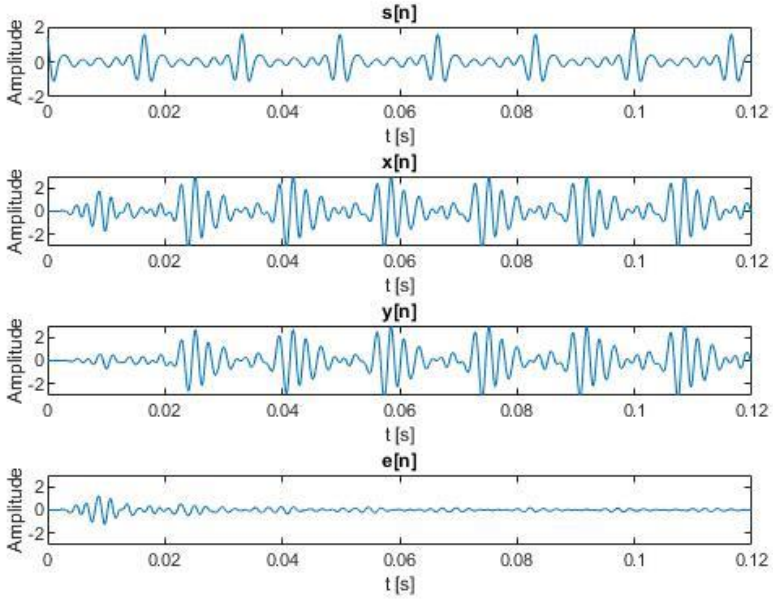
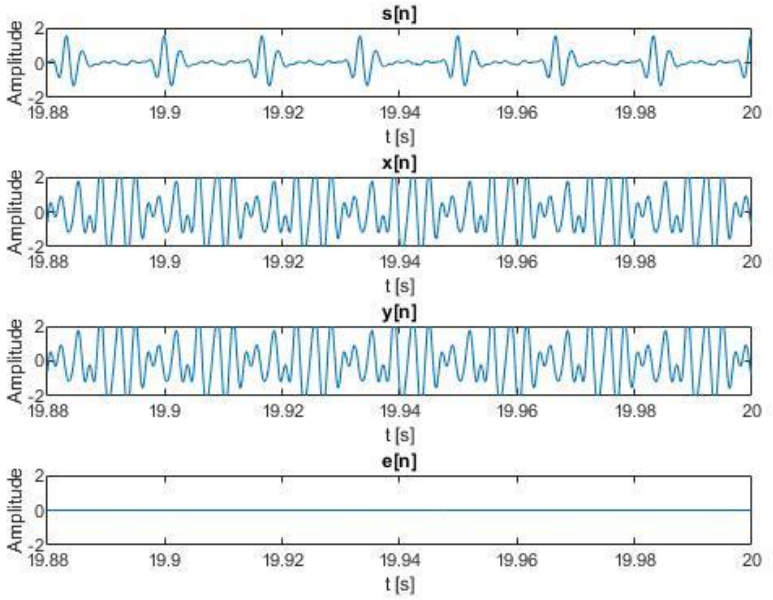


Figura 6. Sinal *s(n)* na entrada do duto acústico, sinal $x(n)$, sinal $y(n)$ gerado pelo alto falante e sinal $e(n) = x(n) - y(n)$ na saída do duto nos 120ms finais para $\mu = 0{,}01$.



A função custo, dado pela equação (3) foi calculada para os 120 primeiros ms e para os 120 ms finais, obtendo-se os valores de 0,0396 e 2,1942x10^{-5}, respectivamente.

Na segunda simulação, foi alterado o passo de adaptação para μ=0,001, mantendo-se o tamanho do filtro em 800 amostras, obtendo-se as Figuras 7 e 8.

Figura 7. Sinal $s(n)$ na entrada do duto acústico, sinal $x(n)$, sinal $y(n)$ gerado pelo alto falante e sinal $e(n) = x(n) - y(n)$ na saída do duto nos 120ms iniciais para $\mu = 0{,}001$.

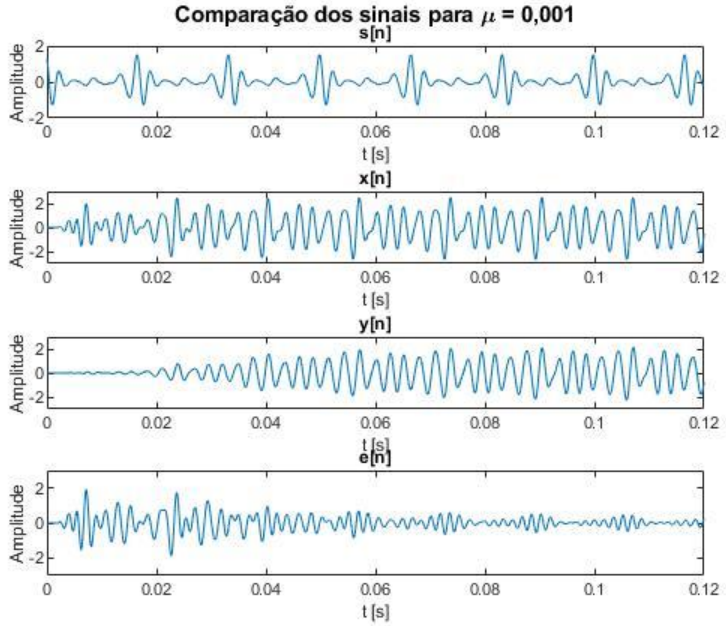
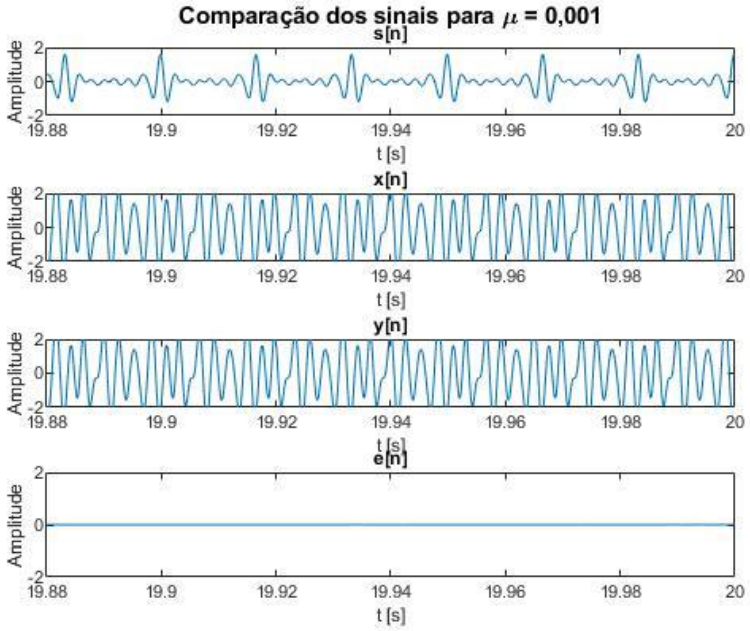


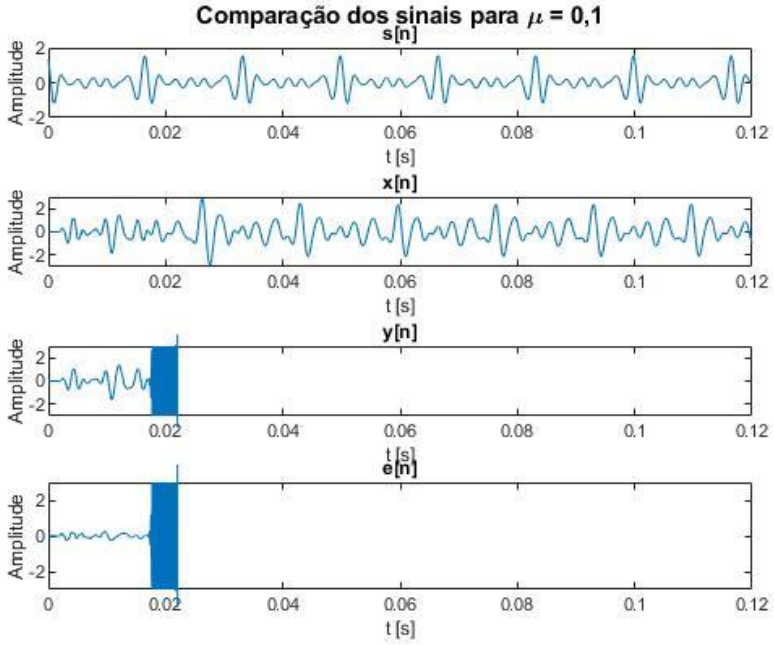
Figura 8. Sinal s(n) na entrada do duto acústico, sinal $x(n)$, sinal $y(n)$ gerado pelo alto falante e sinal $e(n) = x(n) - y(n)$ na saída do duto nos 120ms finais para $\mu = 0{,}001$.



Para $\mu = 0{,}001$ obteve-se o valor de 0,2889 para a função custo nos 120 ms iniciais e 7,2402x10^{-7} para os 120 ms finais.

Em uma terceira e última simulação, foi testado um passo de adaptação $\mu = 0{,}1$, obtendo-se os resultados ilustrados na Figura 9.

Figura 9. Sinal $s(n)$ na entrada do duto acústico, sinal $x(n)$, sinal $y(n)$ gerado pelo alto falante e sinal $e(n) = x(n) - y(n)$ na saída do duto nos 120ms iniciais para $\mu = 0{,}1$.



Para este último caso, foi observado que o passo de adaptação $\mu = 0{,}1$ levou o algoritmo a divergir.

Para fins de comparação, a tabela abaixo mostra os erros mínimos quadráticos médios para as três simulações realizadas:

Tabela 2. Comparação dos Erros Quadráticos médios obtidos nos 120 ms iniciais e nos 120 ms finais por passo de adaptação μ.

μ (Passo de adaptação)	MSE (120 ms iniciais)	MSE (120 ms finais)
0,1	-	-
0,01	0,0396	2,1942x10^{-5}
0,001	0,2889	7,2402x10^{-7}

A partir da análise das figuras obtidas para os sinais $x(n)$, $y(n)$, e $e(n)$, nota-se que para os casos em que o passo de adaptação é pequeno o suficiente para convergir, o passo de adaptação $\mu = 0{,}01$ foi o que obteve um erro quadrático médio menor nos 120 ms iniciais. Nesta situação, o vetor w(n) converge mais rapidamente para a solução conforme a equação (7), dado que o passo de adaptação é maior.

Entretanto, o passo de adaptação $\mu = 0{,}001$ obteve um erro quadrático médio menor nos 120 ms finais do que com o passo de adaptação $\mu = 0{,}01$. Isto indica que, um passo de adaptação pequeno é capaz de obter melhores resultados quando próximo da solução mas um passo de adaptação maior é capaz de convergir mais rapidamente para a solução.

Nota-se também que há um limite para o passo de adaptação para que o algoritmo convirja à solução, o que foi possível observar na Figura 9, onde o passo de adaptação foi igual $\mu = 0{,}1$. Neste último caso, o sinal gerado pelo algoritmo $y(n)$ tornou-se instável. Estes resultados demostram a importância de escolher um passo de adaptação adequado para que se possa obter uma rápida convergência, mas também que atinja bons resultados ao final do processo.

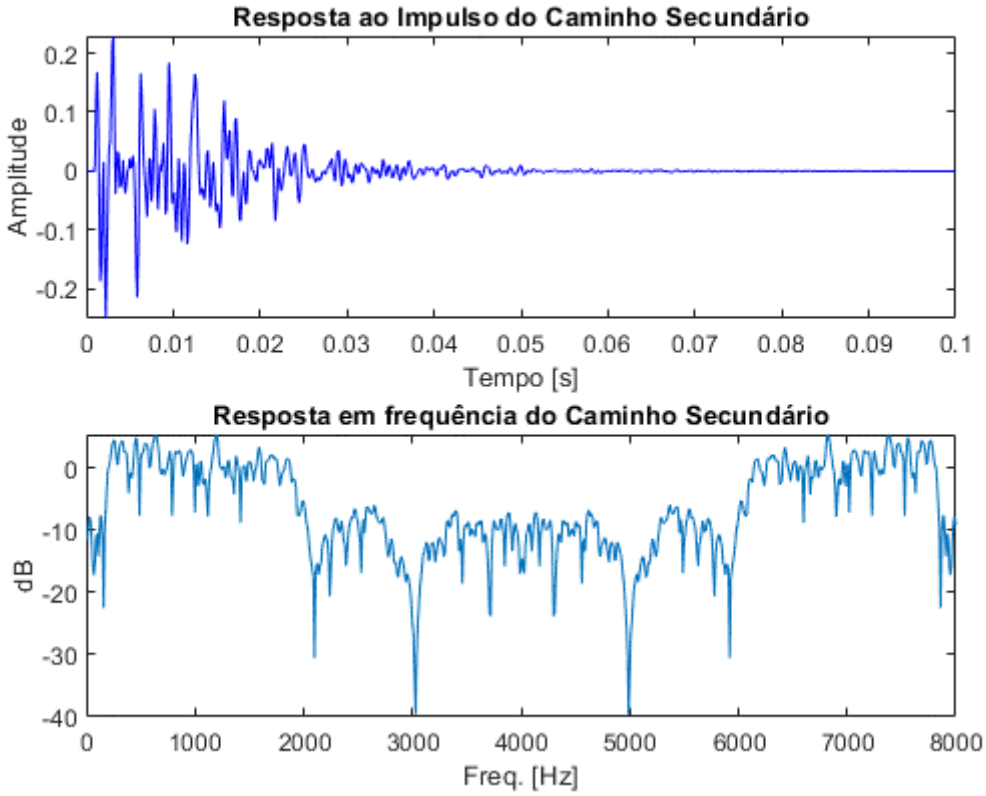
3.2 – Comparação de desempenho dos algoritmos LMS, NLMS e Sign-error LMS na presença de ruído aditivo Gaussiano

Como modelo do caminho secundário, foi utilizado o modelo adotado por Douglas (1999), cujo atributos são mostrados na Tabela 3 e resposta ao impulso e resposta em frequeência do caminho secundário na Figura 10.

Tabela 3. Atributos do filtro utilizado como modelo do caminho secundário.

Frequência de Amostragem (Hz)	Fs = 8 KHz;
Nº de Coeficientes	N = 800;
Frequência limite da banda passa-baixas (Hz)	Flow = 160;
Frequência limite da banda passa-alta (Hz)	Fhigh = 2000;
Ordem do filtro	Ordem do filtro = 8
Atraso de fase	delayW = 15;
Atenuação da banda de rejeição	Ast = 20 dB

Figura 10. Resposta ao Impulso e Resposta em frequência do caminho secundário utilizado neste experimento



O ruído externo adicionado ao caminho secundário, conforme ilustrado pelo diagrama da Figura 3, foi modelado como um ruído aditivo branco e gaussiano. Diferentes valores de SNR foram adicionados ao sinal $\hat{s}(n)$, iniciando em 9 dB com incrementos de 3 dB.

Como forma de avaliar o MSE dos algoritmos utilizados, o programa foi executado 100 vezes, para diferentes níveis de ruído, para os três algoritmos testados, LMS, NLMS e Sign-error LMS.

Obteve-se então a média do MSE nos últimos 120 ms para cada nível de SNR do ruído gaussiano para os três algoritmos.

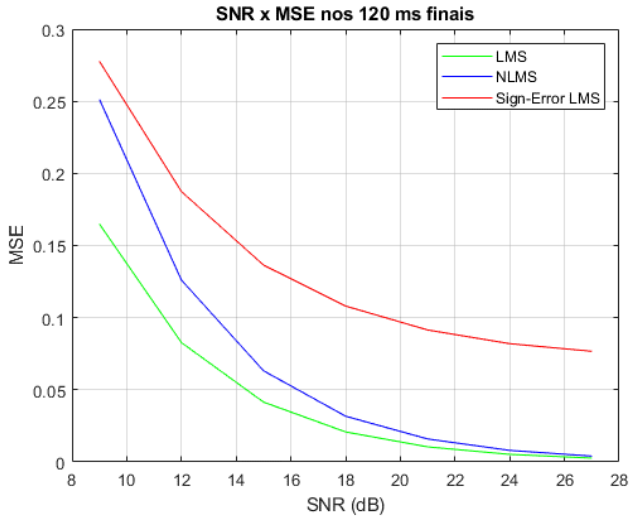
Na Tabela 4, são mostrados os valores das médias do MSE para os diferentes níveis de ruído. O valor do passo de adaptação μ escolhido arbitrariamente, com um valor suficientemente pequeno para que o algoritmo convergisse antes dos 18 segundos. Para estas simulações, o valor escolhido foi de $\mu = 0{,}001$ para os algoritmos LMS e Sign-Error LMS. O passo de adaptação para o algoritmo NLMS escolhido foi $\mu = 1$.

Tabela 4. *Mean-Squared Error* para diferentes níveis de SNR

SNR (dB)	MSE_{LMS}	MSE_{NLMS}	$MSE_{Sign\text{-}error\ LMS}$
9	0,165	0,251	0,278
12	0,083	0,126	0,187
15	0,041	0,063	0,136
18	0,021	0,032	0,108
21	0,010	0,016	0,091
24	0,005	0,008	0,082
27	0,003	0,004	0,077

Obteve-se a curva da MSE para os três algoritmos na Figura 13.

Figura 11 – Simulação de três algoritmos tipo LMS por SNR



Observa-se na figura 11 que a curva do algoritmo LMS apresenta o menor erro médio (MSE) entre os três algoritmos.

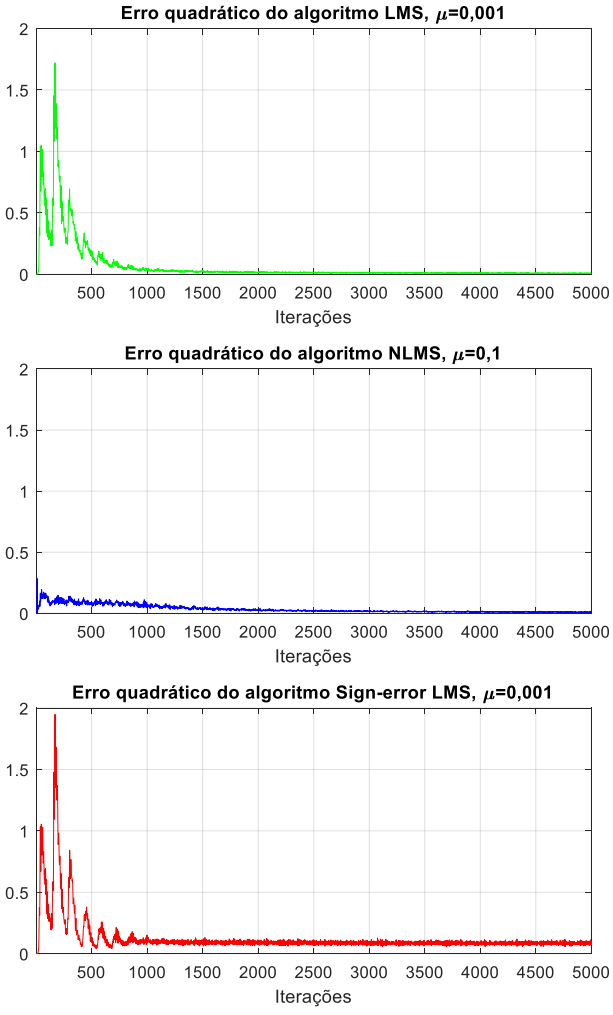
O algoritmo sign-error LMS apresenta um custo computacional menor (Sayed, 2011) com relação aos demais, o que pode ser vantajoso para aplicações nas quais são exigidas respostas mais rápidas ou o equipamento controlador não for tão robusto. Porém, este algoritmo apresenta a desvantagem de resultar em um erro maior comparado ao LMS.

Neste caso, a velocidade de convergência aumenta e o erro no estado estacionário é maior.

O algoritmo NLMS tem como característica uma convergência mais rápida que o LMS, apesar de ter apresentado um erro maior. A complexidade do NLMS também é maior que a do algoritmo LMS.

A Figura 12 ilustra a média do erro quadrático para 100 simulações com uma SNR igual a 27dB, para os três algoritmos nas primeiras 5000 iterações. Observa-se, nesta figura, que a amplitude do erro, após a convergência, é levemente maior para a simulação utilizando o algoritmo Sign-error LMS. No caso do NLMS, o algoritmo convergiu mais rapidamente em termos de iterações quando comparado aos demais.

Figura 12. Média do erro quadrático para 100 simulações, por algoritmo.



4 – CONCLUSÃO

Neste trabalho foram apresentados os conceitos básicos que foram utilizados para o desenvolvimento da simulação de um modelo de controle ativo de ruído através da aplicação do algoritmo adaptativo LMS e seus derivados.

A utilização da ferramenta Matlab mostrou-se essencial para a compreensão do comportamento dos algoritmos adaptativos avaliados neste estudo[2]. Através desta ferramenta, foi possível uma aproximação do modelo real, no qual variáveis tais como o ruído externo adicionado ao caminho secundário foram modelados e aproximam-se de um modelo real.

De maneira geral, as simulações mostram que o controle ativo de ruído utilizando os algoritmos adaptativos LMS,NLMS e Sign-Error LMS são uma alternativa viável para a atenuação de ruído em um canal com fontes sonoras de baixa frequência.

De acordo com as simulações, o algoritmo LMS apresenta a menor média de erro, enquanto o NLMS é o algoritmo com a melhor resposta de convergência. Em termos de velocidade, o sign-error LMS mostrou um erro final maior entre os três algoritmos testados.

[2] Códigos disponíveis no Apêndice A.

5 – BIBLIOGRAFIA

ALMEIDA JUNIOR, A.B. *Simulação e implementação de um sistema de controle ativo de ruído mono-canal em dutos acústicos com propagação de ondas planas*. Trabalho de Conclusão de Curso, Engenharia Eletrônica, Universidade de Brasília, 2014.

BUREAU INTERNACIONAL DO TRABALHO, Genebra. *Introdução à Saúde e Segurança no Trabalho*. 1996.

DOUGLAS, S. C. *Fast implementations of the filtered-X LMS and LMS algorithms for multichannel active noise control*. IEEE Transactions on speech and audio processing, v. 7, n. 4, p. 454-465, 1999.

HAYKIN, S. *Adaptive Filter Theory.* 3rd Ed., Prentice-Hall, Inc., Upper Saddle River, NJ, USA. 1996.

KUO, S. M.; MORGAN, D. R. *Active Noise Control Systems.* John Wiley & Sons, Inc 1996, 1996.

LESSA, E. M. *Controle Ativo de Ruído em Dutos de Ventilação em Navios e Plataformas Offshore.* Trabalho de Conclusão de Curso. Engenharia de Controle e Automação, Universidade do Rio de Janeiro, RJ, Brasil, 2010.

LUEG, Paul*. Process of Silenting Sound Oscillations*. US Patent 2043416. June09, 1936.

NUÑEZ, I. J. C. *O controle ativo de ruído em dutos: um estudo teórico - experimental.* Tese de Doutorado, Universidade Federal de Uberlândia, 2005.

OLSON, H. F.; MAY, E. G. Electronic Sound Absorber. *The Journal of the Acoustical Society of America*, 25, 1130–1136, 1953.

RIYANTO, B. Real-time DSP Implementation of Active Noise Control for Broadband Noise Using Adaptive LMS Filter Algorithm. *International Conference on Electrical Engineering and Informatics*, 718–722, 2007.

TARRAB, M.; FEUER, A. . *Convergence and performance analysis of the normalized LMS algorithm with uncorrelated Gaussian data.* IEEE Transactions on Information Theory, v. 34, n. 4, p. 680-691, 1988.

ROMANO, J. M. T.; CAVALCANTE, C.C.; ATTUX, R.R. de F.; SUYAMA, R.. *Unsupervised signal processing: channel equalization and source separation.* CRC Press, 2011.

SAYED, Ali H. *Adaptive filters.* John Wiley & Sons, 2011.

ZIMMER, Benjamin J. et al. An improved acoustic model for active noise control in a duct. *Journal of dynamic systems, measurement, and control,* v. 125, n. 3, p. 382-395, 2003.

APÊNDICE A - Códigos MATLAB

Código para Resultados para parte 3.1 -Análise de convergência do algoritmo LMS para diferentes passos de adaptação:

```
clc
clear all
close all

%%experimento, primeira parte
%semente
defaultStream = RandStream.getGlobalStream;
%Para executar a parte sem ruído externo, usar parâmetro abaixo:
MC=1;
    for jj=1:MC
    stream_exp = RandStream('mt19937ar','seed',jj);
    defaultStream.State = stream_exp.State;
Fs     = 8e3;  % 8 kHz
N      = 800;  % 800 samples@8 kHz = 0.1 seconds
Flow   = 160;  % Lower band-edge: 160 Hz
Fhigh  = 2000; % Upper band-edge: 2000 Hz
delayS = 7;
Ast    = 20;   % 20 dB stopband attenuation
Nfilt  = 8;    % Filter order

%% Secondary path

% Design bandpass filter to generate bandlimited impulse response
filtSpecs = fdesign.bandpass('N,Fst1,Fst2,Ast',Nfilt,Flow,Fhigh,Ast,Fs);
bandpass = design(filtSpecs,'cheby2','FilterStructure','df2tsos', ...
    'SystemObject',true);

% Filter noise to generate impulse response

tcoeffs = (1:N)/Fs;
fcoeffs = (1:N)*Fs/N;

%% Primary path
delayW = 15;
Flow   = 200; % Lower band-edge: 200 Hz
Fhigh  = 800; % Upper band-edge: 800 Hz
Ast    = 20;  % 20 dB stopband attenuation
Nfilt  = 10;  % Filter order

% Design bandpass filter to generate bandlimited impulse response
filtSpecs2 = fdesign.bandpass('N,Fst1,Fst2,Ast',Nfilt,Flow,Fhigh,Ast,Fs);
bandpass2 = design(filtSpecs2,'cheby2','FilterStructure','df2tsos', ...
    'SystemObject',true);

% Filter noise to generate impulse response
primaryPathCoeffs = bandpass2([zeros(delayW,1); log(0.99*rand(N-
delayW,1)+0.01).* ...
    sign(randn(N-delayW,1)).*exp(-0.01*(1:N-delayW)')]);
primaryPathCoeffs = primaryPathCoeffs/norm(primaryPathCoeffs);
```

```
% Sine wave generator to synthetically create the noise
A = [.01 .01 .02 .2 .3 .4 .3 .2 .1 .07 .02 .01];
La = length(A);
F0 = 60;
k = 1:La;
F = F0*k;
phase = rand(1,La); % Random initial phase

t = 0:1/Fs:20-1/Fs;
s = zeros(size(t));
for ii=1:La
    s = s + A(ii)*cos(2*pi*F(ii)*t+phase(ii));
end

a = primaryPathCoeffs;
x = filter(a,1,s);

%Passo de adaptação, alterar para 0,01 e 0,1
mu   = 0.001;
taps = 800;

[W,y,e,J1,~] = mse(s,x,mu,taps);  %LMS, sem ruido externo

w = W(end,:)';
%200ms=20s e 960 amostras = 20 ms

% mse_ini(jj,kk)=mean(e(1:960).^2)
% mse_end(jj,kk)=mean(e(end-960:end).^2)

mse_ini(jj)=mean(e(1:960).^2);
mse_end(jj)=mean(e(end-960:end).^2);

    end
% end
%%
figure(1),
subplot(211), plot(tcoeffs,primaryPathCoeffs,'b');
xlabel('Tempo [s]');
ylabel('Amplitude');
% title('Primary Path Impulse Response');
title('Resposta ao Impulso do Caminho Primário');
axis ([0 0.1 -0.3 0.3])
subplot(212), plot(fcoeffs,10*log10(abs(fft(primaryPathCoeffs))))
 axis([0 4050 -60 20])
xlabel('Freq. [Hz]');
ylabel('dB');
title('Resposta em frequência do caminho primário');

%plot do primeiro experimento
figure(2),
subplot(411), plot(t,s), title('s[n]') %'Barulho do ventilador'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([0 0.12 -2 2])
subplot(412), plot(t,x), title('x[n]')%'Barulho na saída do duto sem CAR'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([0 0.12 -3 3])
subplot(413), plot(t,y), title('y[n]')%'Barulho na saída do alto
```

```
%falante produzido pelo CAR'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([0 0.12 -3 3])
subplot(414), plot(t,e), title('e[n]')%'Barulho na saída do duto com CAR'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([0 0.12 -3 3])

p=mtit('Comparação dos sinais para \mu = 0,001');

%plot primeiro exp, 120 MS finais
figure(3),
subplot(411), plot(t,s), title('s[n]') %'Barulho do ventilador'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([20-0.12 20 -2 2])
subplot(412), plot(t,x), title('x[n]')
%'Barulho na saída do duto sem CAR'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([20-0.12 20 -2 2])
subplot(413), plot(t,y), title('y[n]')
%'Barulho na saída do alto falante produzido pelo CAR'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([20-0.12 20 -2 2])
subplot(414), plot(t,e), title('e[n]')%'Barulho na saída do duto com CAR'
xlabel ('t [s]');
ylabel ('Amplitude');
axis ([20-0.12 20 -2 2])
p=mtit('Comparação dos sinais para \mu = 0,001');
%%FIM
```

Código do algoritmo LMS:

```
function [W,y,e,J,gradJ] = mse(x,d,mu,taps)

w = zeros(1,taps);

%empilhamento do sinal e seus atrasos
X = zeros(taps,length(x));
for it3=1:taps
    X(it3,it3:end) = x(1:end-it3+1);
end

N = length(X);
y = zeros(1,N);
e = zeros(1,N);
J = zeros(1,N);
W = zeros(N,taps);
gradJ = zeros(taps,N);

for it=1:N

    y(it) = w*X(:,it); %filtragem / combinação linear
    e(it) = d(it) - y(it); %sinal de erro
```

```
    gradJ(:,it) = e(it)*X(:,it);
    J(it) = e(it).^2;
    w = w + mu*gradJ(:,it)'; %ajuste do novo vetor de pesos -> gradient
ascent

    W(it,:) = w;

end

end
```

Código do algoritmo NLMS:

```
function [W,y,e,J,gradJ] = nlms2(x,d,mu,taps)

w = zeros(1,taps);

%empilhamento do sinal e seus atrasos
X = zeros(taps,length(x));
for it3=1:taps
    X(it3,it3:end) = x(1:end-it3+1);
end

N = length(X);
y = zeros(1,N);
e = zeros(1,N);
J = zeros(1,N);
W = zeros(N,taps);
gradJ = zeros(taps,N);

for it=1:N

    y(it) = w*X(:,it); %filtragem / combinação linear
    e(it) = d(it) - y(it); %sinal de erro

    gradJ(:,it) = e(it)*X(:,it)./(X(:,it)'*X(:,it)+eps);
    J(it) = e(it).^2;
    w = w + mu*gradJ(:,it)'; %ajuste do novo vetor de pesos -> gradient
ascent
%     mu=mu*0.99;%adaptive step-size
    W(it,:) = w;

end

end
```

Código do algoritmo Sign-error LMS:

```
function [W,y,e,J,gradJ] = slms(x,d,mu,taps)

w = zeros(1,taps);

%empilhamento do sinal e seus atrasos
X = zeros(taps,length(x));
for it3=1:taps
    X(it3,it3:end) = x(1:end-it3+1);
end

N = length(X);
y = zeros(1,N);
e = zeros(1,N);
J = zeros(1,N);
W = zeros(N,taps);
gradJ = zeros(taps,N);

for it=1:N

    y(it) = w*X(:,it); %filtragem / combinação linear
    e(it) = d(it) - y(it); %sinal de erro

    gradJ(:,it) = sign(e(it))*X(:,it);
    J(it) = abs(e(it));
    w = w + mu*gradJ(:,it)'; %ajuste do novo vetor de pesos -> gradient
ascent
%     mu=mu*0.99;%adaptive step-size
    W(it,:) = w;

end

end
```

Código para a parte de Resultado 3.2 -Comparação de desempenho dos algoritmos LMS, NLMS e Sign-error LMS na presença de ruído aditivo Gaussiano

```
snr_db = [9 12 15 18 21 24 27];
% snr_db = [27];

defaultStream = RandStream.getGlobalStream;
%Numero de experimentos;
MC = 100;
%prealocação para melhor desempenho;
mse_ini1=zeros(100,7);
mse_end1=zeros(100,7);
mse_ini2=zeros(100,7);
mse_end2=zeros(100,7);
mse_ini3=zeros(100,7);
mse_end3=zeros(100,7);

for kk=1:length(snr_db)
% rodar 100 vezes exp de monte carlo (jj 1:100)
```

```
    for jj=1:MC
    stream_exp = RandStream('mt19937ar','seed',jj);
    defaultStream.State = stream_exp.State;
Fs     = 8e3;  % 8 kHz
N      = 800;  % 800 samples@8 kHz = 0.1 seconds
Flow   = 160;  % Lower band-edge: 160 Hz
Fhigh  = 2000; % Upper band-edge: 2000 Hz
delayS = 7;
Ast    = 20;   % 20 dB stopband attenuation
Nfilt  = 8;    % Filter order

%% Secondary path

% Design bandpass filter to generate bandlimited impulse response
filtSpecs = fdesign.bandpass('N,Fst1,Fst2,Ast',Nfilt,Flow,Fhigh,Ast,Fs);
bandpass = design(filtSpecs,'cheby2','FilterStructure','df2tsos', ...
    'SystemObject',true);

% Filter noise to generate impulse response
secondaryPathCoeffsActual = bandpass([zeros(delayS,1); ...
                      log(0.99*rand(N-delayS,1)+0.01).* ...
                      sign(randn(N-delayS,1)).*exp(-0.01*(1:N-
delayS)')]);
secondaryPathCoeffsActual = ...
    secondaryPathCoeffsActual/norm(secondaryPathCoeffsActual);

tcoeffs = (1:N)/Fs;
fcoeffs = (1:N)*Fs/N;

%% Primary path
delayW = 15;
Flow   = 200; % Lower band-edge: 200 Hz
Fhigh  = 800; % Upper band-edge: 800 Hz
Ast    = 20;  % 20 dB stopband attenuation
Nfilt  = 10;  % Filter order

% Design bandpass filter to generate bandlimited impulse response
filtSpecs2 = fdesign.bandpass('N,Fst1,Fst2,Ast',Nfilt,Flow,Fhigh,Ast,Fs);
bandpass2 = design(filtSpecs2,'cheby2','FilterStructure','df2tsos', ...
    'SystemObject',true);

% Filter noise to generate impulse response
primaryPathCoeffs = bandpass2([zeros(delayW,1); log(0.99*rand(N-
delayW,1)+0.01).* ...
    sign(randn(N-delayW,1)).*exp(-0.01*(1:N-delayW)')]);
primaryPathCoeffs = primaryPathCoeffs/norm(primaryPathCoeffs);
%

% Sine wave generator to synthetically create the noise
A = [.01 .01 .02 .2 .3 .4 .3 .2 .1 .07 .02 .01];
La = length(A);
F0 = 60;
k = 1:La;
F = F0*k;
phase = rand(1,La); % Random initial phase


t = 0:1/Fs:20-1/Fs;
s = zeros(size(t));
```

```
for ii=1:La
    s = s + A(ii)*cos(2*pi*F(ii)*t+phase(ii));
end


a = primaryPathCoeffs;
b = secondaryPathCoeffsActual;

x = filter(a,1,s);
% calcular pot de x

%pararametro: passo de adaptacao
mu   = 0.001;

taps = 800;

s_hat = filter (b,1,s);
% snr_fazer loop para 9 à 27 db
x_awgn = awgn(x,snr_db(kk));

[W1,y,e1,j1,~] = mse(s_hat,x_awgn,mu,taps);
[W2,y,e2,j2,~] = nlms2(s_hat,x_awgn,1,taps);
[W3,y,e3,j3,~] = slms(s_hat,x_awgn,mu,taps);
% w = W1(end,:)';

%200ms=20s e 960 amostras = 20 ms
mse_ini1(jj,kk)=mean(e1(1:960).^2)
mse_ini2(jj,kk)=mean(e2(1:960).^2)
mse_ini3(jj,kk)=mean(e3(1:960).^2)

mse_end1(jj,kk)=mean(e1(end-960:end).^2)
mse_end2(jj,kk)=mean(e2(end-960:end).^2)
mse_end3(jj,kk)=mean(e3(end-960:end).^2)

    end
end
    mean_ini1=mean(mse_ini1);
    mean_ini2=mean(mse_ini2);
    mean_ini3=mean(mse_ini3);

    mean_end1=mean(mse_end1);
    mean_end2=mean(mse_end2);
    mean_end3=mean(mse_end3);

    figure,

    plot (snr_db,mean_end1,'g');hold on;
    plot (snr_db,mean_end2,'b');hold on;
    plot (snr_db,mean_end3,'r');
    title ('SNR x MSE nos 120 ms finais');
    xlabel ('SNR (dB)');
    ylabel ('MSE');
    grid on;
    legend ('LMS','NLMS','Sign-Error LMS');
```