Implementação e Avaliação de Desempenho de Algoritmos para Cancelamento Adaptativo de Ruído

Rafael Rodrigo Pertum e Eduardo Vinicius Kuhn

***Resumo*—Este artigo trata da implementação e da avaliação de desempenho de um sistema de cancelamento adaptativo de ruído, operando com relevantes algoritmos adaptativos da literatura [isto é, o NLMS (*normalized least-mean-square*) e três diferentes VSS-NLMS (*variable step-size* NLMS)]. Essa implementação é realizada aqui em linguagem C utilizando a placa de desenvolvimento Cortex-FM4 *Starter* da *Cypress Semiconductors*®, a qual possui recursos para aquisição e síntese de sinais de áudio. Comparações de desempenho através de metodologias objetivas para avaliação de qualidade e inteligibilidade de sinais de fala são apresentadas e discutidas, confirmando a eficácia da implementação proposta.**

Palavras-chave*—*Algoritmo NLMS, algoritmos VSS-NLMS, cancelamento adaptativo de ruído, filtragem adaptativa.

***Abstract*—This paper deals with the implementation and performance evaluation of an adaptive noise cancelling system, operating with relevant adaptive algorithms from the literature [i.e., the normalized least-mean-square (NLMS) and three different variable step-size NLMS (VSS-NLMS)]. Such an implementation is carried out here in C language using the Cortex-FM4 Starter kit from Cypress Semiconductors®, which has acquisition and synthesis resources for audio signals. Performance comparisons through objective methodologies for evaluating quality and intelligibility of speech signals are presented and discussed, confirming the effectiveness of the proposed implementation.**

Keywords*—*NLMS algorithm, VSS-NLMS algorithms, adaptive noise canceling, adaptive filtering.

I. Introdução

Atualmente, sistemas de cancelamento adaptativo de ruído (ANC, *adaptive noise cancelling*) vêm sendo utilizados em diversas aplicações práticas, tais como na atenuação de interferência da rede elétrica durante a aquisição de sinais em equipamentos biomédicos, bem como na supressão de ruído em aparelhos auditivos, celulares e/ou de áudio-conferência [1]-[3]. Nessas aplicações, busca-se mitigar o ruído que degrada o sinal de interesse, visando assim facilitar a interpretação da informação desejada e/ou melhorar a qualidade da comunicação. Para tal, visto que fontes de ruído podem variar com o tempo, técnicas de filtragem adaptativa são empregadas para sintetizar (em tempo real) uma estimativa do ruído que corrompe o sinal de interesse para, então, atenuá-lo (por subtração) no domínio elétrico [1].

Dentre os algoritmos adaptativos comumente utilizados, destacam-se o algoritmo NLMS (*normalized least-mean-square*) e os algoritmos VSS-NLMS (*variable step-size* NLMS) [3]-[6]. Em particular, o algoritmo VSS-NLMS desenvolvido por Shin [4] visa prover uma rápida velocidade de convergência e um baixo erro em regime. No entanto, o funcionamento adequado desse algoritmo requer o ajuste de um elevado número de parâmetros, o que dificulta a sua empregabilidade prática. Para contornar esse problema, um algoritmo VSS-NLMS dito não paramétrico é derivado por Benesty *et al.* [5], objetivando aplicações de cancelamento de eco acústico. Todavia, apesar de simulações atestarem seu bom desempenho em termos de velocidade de convergência, capacidade de rastreamento e reduzido erro em regime permanente, o algoritmo depende do conhecimento *a priori* da variância do ruído de medição. Logo, procedimentos precisos para estimar a variância desse ruído se fazem necessários; caso contrário, o desempenho do algoritmo é significativamente comprometido. Ainda, é apresentado por Zipf, Tobias e Seara [6] um algoritmo VSS-NLMS baseado na autocorrelação do erro que não depende do conhecimento da variância do ruído de medição. Tal algoritmo possui baixa complexidade computacional e requer o ajuste de um único parâmetro, o qual mostra ter pouco impacto sobre o seu desempenho.

Em aplicações envolvendo o processamento de sinais de fala, metodologias para avaliações subjetivas [7] e/ou objetivas [8] relacionadas à qualidade e à inteligibilidade são fundamentais para o adequado desenvolvimento e a validação de sistemas. Nessas metodologias, o sinal original e aquele obtido como resultado de algum processamento são comparados e avaliados por um grupo de pessoas ouvintes (em avaliações subjetivas) [7] ou através de métodos computacionais (em avaliações objetivas) [8], [9], resultando em uma pontuação que indica a qualidade ou inteligibilidade do sinal processado. Contudo, metodologias para avaliações subjetivas acabam se tornando excessivamente demoradas e custosas, uma vez que dependem de um grande número de ouvintes. Por isso, as avaliações objetivas vêm sendo preferencialmente utilizadas; sobretudo, por mostrarem elevada correlação com o resultado obtido através de avaliações subjetivas (devido ao aprimoramento dos métodos computacionais) [9].

Neste contexto, o presente trabalho de pesquisa tem os seguintes objetivos:

i) realizar a implementação prática de um sistema de ANC através da placa de desenvolvimento Cortex-FM4 *Starter* da *Cypress Semiconductors*®;
ii) implementar quatro algoritmos adaptativos da literatura, isto é, o algoritmo NLMS bem como os algoritmos VSS-NLMS de Shin [4], Benesty [5] e Zipf [6]; e
iii) avaliar o desempenho dos algoritmos considerados, por meio de metodologias objetivas, em termos de qualidade e inteligibilidade de sinais de fala.

Vale salientar que o presente trabalho estende os resultados apresentados em [10], versando agora sobre outros algoritmos adaptativos como também sobre avaliações de desempenho mais abrangentes.

Este artigo está organizado como segue. A Seção II apresenta a topologia de um sistema de ANC bem como revisita os algoritmos adaptativos aqui considerados. A Seção III descreve aspectos pertinentes à implementação proposta. Na Seção IV, resultados decorrentes de avaliações objetivas de qualidade e inteligibilidade são mostrados e discutidos. Finalmente, a Seção V traz as considerações finais do presente trabalho de pesquisa.

Rafael Rodrigo Pertum e Eduardo Vinicius Kuhn, LAPSE–Laboratório de Processamento de Sinais e Eletrônica, Departamento de Engenharia Eletrônica, Universidade Tecnológica Federal do Paraná (UTFPR), Toledo-PR, Brasil (e-mails: pertum@alunos.utfpr.edu.br e kuhn@utfpr.edu.br).

Este trabalho foi parcialmente financiado pela Universidade Tecnológica Federal do Paraná e pela Fundação Araucária.

II. FORMULAÇÃO DO PROBLEMA

Nesta seção, a topologia de um sistema de ANC bem como as expressões do algoritmo NLMS e daqueles introduzidos em [4]-[6] são brevemente revisitadas. Vale salientar que a notação matemática adotada segue a prática padrão de usar letras minúsculas em negrito para vetores, e letras gregas e romanas em itálico para escalares.

A. Topologia de um sistema de ANC

Em um sistema de ANC (conforme ilustrado na Fig. 1), um algoritmo adaptativo [empregando um sinal de referência $x(n)$ e o sinal de erro $e(n)$] ajusta os coeficientes de uma estrutura de filtragem de forma a produzir uma estimativa $\hat{d}(n)$ do ruído $d(n)$ que corrompe o sinal de interesse $s(n)$. Então, realizando a subtração entre o sinal corrompido por ruído (observado na entrada primária) e a estimativa do ruído, obtém-se o sinal de erro como

$$e(n) = s(n) + d(n) - \hat{d}(n). \tag{1}$$

Note que, conforme o algoritmo adaptativo converge para a solução em regime permanente [isto é, $\hat{d}(n) \to d(n)$ conforme $n \to \infty$], o sinal de erro tende para o sinal de interesse [isto é, $e(n) \to s(n)$ para $n \to \infty$], produzindo assim a atenuação do ruído na saída do sistema (como desejado).

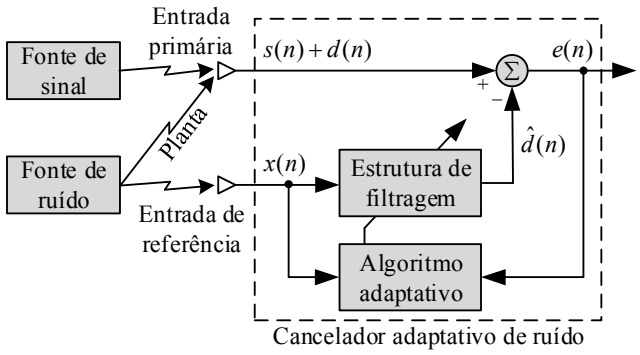

Fig.1. Topologia geral de um sistema de ANC.

B. Algoritmo NLMS

Considerando uma estrutura transversal de filtragem, a equação de adaptação dos coeficientes do algoritmo NLMS pode ser definida como [1]

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \frac{\mathbf{x}(n)e(n)}{\mathbf{x}^{\mathrm{T}}(n)\mathbf{x}(n) + \varepsilon} \tag{2}$$

onde $\mathbf{w}(n) = [w_0(n) \ \cdots \ w_{M-1}(n)]^{\mathrm{T}}$ denota o vetor de coeficientes do filtro adaptativo de ordem M, $(\cdot)^{\mathrm{T}}$, o operador de transposição, $\mathbf{x}(n) = [x(n) \ \cdots \ x(n-M+1)]^{\mathrm{T}}$, um vetor contendo as amostras mais recentes da entrada de referência, $0 < \mu < 2$, o passo de adaptação e $\varepsilon > 0$, um parâmetro de regularização que visa evitar divisão por zero e estabilizar a solução.

C. Algoritmo VSS-NLMS de Shin

Em algoritmos de passo variável, o passo de adaptação em (2) é feito variante no tempo, isto é, μ é substituído por $\mu(n)$. Dessa forma, (2) pode ser revisada para

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu(n) \frac{\mathbf{x}(n)e(n)}{\mathbf{x}^{\mathrm{T}}(n)\mathbf{x}(n) + \varepsilon} \tag{3}$$

onde $\mu(n)$ caracteriza o valor do passo de adaptação em um dado instante n, o qual é ajustado iterativamente de acordo com alguma regra específica. Note que diferentes estratégias são utilizadas na literatura para derivar regras de ajuste para o passo de adaptação $\mu(n)$, dando assim origem a um grande número de algoritmos adaptativos de passo variável.

Especificamente no algoritmo VSS-NLMS de Shin [4], o ajuste do passo de adaptação é realizado de acordo com

$$\mu(n) = \mu_{\text{máx}} \frac{\mathbf{p}^{\mathrm{T}}(n)\mathbf{p}(n)}{\mathbf{p}^{\mathrm{T}}(n)\mathbf{p}(n) + C} \tag{4}$$

onde

$$\mathbf{p}(n) = \beta \mathbf{p}(n-1) + (1-\beta) \frac{\mathbf{x}(n)e(n)}{\mathbf{x}^{\mathrm{T}}(n)\mathbf{x}(n)} \tag{5}$$

com $0 < C \ll 1$ representando uma constante positiva inversamente proporcional a SNR (*signal-to-noise ratio*), $\mu_{\text{máx}}$, o valor máximo que o passo de adaptação pode assumir e $0 \ll \beta < 1$, uma constante positiva de suavização.

D. Algoritmo VSS-NLMS de Benesty

No algoritmo VSS-NLMS de Benesty (dito não paramétrico) [5], o passo de adaptação é computado através de

$$\mu(n) = \begin{cases} \gamma(n), & \text{se } \hat{\sigma}_e(n) \geq \hat{\sigma}_v \\ 0, & \text{caso contrário} \end{cases} \tag{6}$$

com

$$\gamma(n) = 1 - \frac{\hat{\sigma}_v}{\delta + \hat{\sigma}_e(n)} \tag{7}$$

onde δ representa uma constante de regularização, $\hat{\sigma}_e^2(n)$, uma estimativa da variância do sinal de erro obtida de

$$\hat{\sigma}_e^2(n) = \lambda \hat{\sigma}_e^2(n-1) + (1-\lambda)e^2(n) \tag{8}$$

para

$$\lambda = 1 - \frac{1}{KM}, \ K \geq 2 \tag{9}$$

e $\hat{\sigma}_v^2$, uma estimativa da variância do ruído de medição. Vale salientar que Ciochina, Paleologu e Benesty [11] sugerem que essa estimativa pode ser obtida de acordo com [12], isto é,

$$\hat{\sigma}_v^2 = \hat{\sigma}_e^2(n) - \frac{1}{\hat{\sigma}_x^2(n)} \hat{\mathbf{r}}_{ex}^{\mathrm{T}}(n)\hat{\mathbf{r}}_{ex}(n) \tag{10}$$

com

$$\hat{\sigma}_x^2(n) = \lambda \hat{\sigma}_x^2(n-1) + (1-\lambda)x^2(n) \tag{11}$$

e

$$\hat{\mathbf{r}}_{ex}(n) = \lambda \hat{\mathbf{r}}_{ex}(n-1) + (1-\lambda)\mathbf{x}(n)e(n). \tag{12}$$

E. Algoritmo VSS-NLMS de Zipf

Para o algoritmo VSS-NLMS de Zipf [6], a regra de ajuste para o passo de adaptação em (3) é dada por

$$\mu(n) = \frac{p^2(n)}{q^2(n)} \tag{13}$$

com

$$p(n) = \beta p(n-1) + (1-\beta)e(n)e(n-1) \tag{14}$$

e

$$q(n) = \beta q(n-1) + (1-\beta)e^2(n) \tag{15}$$

sendo $0 \ll \beta < 1$ um parâmetro de suavização (detalhes sobre o funcionamento do algoritmo são discutidos em [6]).

III. IMPLEMENTAÇÃO PROPOSTA

Na implementação do sistema de ANC, foi utilizada uma placa de desenvolvimento Cortex-FM4 *Starter* da *Cypress Semiconductors*, a qual possui um microcontrolador ARM® Cortex-M4 S6E2CC operando a 200 MHz [13]. Essa placa de desenvolvimento conta com um *codec* (*coder-decoder*) Wolfson® WM8731 responsável por realizar a interface entre entrada/saída de áudio (disponíveis através de conectores do

tipo P2 estéreo) e o microcontrolador por meio do protocolo de comunicação I2S (*inter-IC sound*). Salienta-se que os sinais de áudio (fala) envolvidos foram digitalizados com uma frequência de amostragem de 8 kHz e cada amostra quantizada em 16 bits. Além disso, a placa de desenvolvimento possui um dispositivo específico para depuração via USB (*universal serial bus*), permitindo assim acompanhar a execução do código bem como acessar as variáveis de interesse armazenadas na memória.

Para o desenvolvimento dos códigos, foi considerada a IDE (*integrated development environment*) Keil µVision® MDK (*microcontroller development kit*) versão 5 fornecida pela ARM® [14]. Essa IDE permite a programação em *assembly*, linguagem C e C++. Adicionalmente, a IDE oferece suporte à biblioteca CMSIS-DSP (*Cortex microcontroller software interface standard – digital signal processing*) versão 1.5.1, contendo funções já otimizadas para realizar operações de processamento de sinais em microcontroladores Cortex-M. A IDE possibilita ainda compilar, gravar e depurar o código de forma integrada com a placa de desenvolvimento utilizada sem a necessidade de qualquer outra ferramenta externa, o que auxilia sobremaneira durante a implementação (concepção, teste e otimização).

Considerando a linguagem de programação C, trechos de código relacionados à implementação de cada um dos quatro algoritmos adaptativos considerados aqui são mostrados nas Figs. 2-5. Especificamente, a Fig. 2 traz a implementação do algoritmo NLMS (descrito na Seção II-B), a Fig. 3 trata da implementação do algoritmo VSS-NLMS de Shin (revisitado na Seção II-C), a Fig. 4 apresenta a implementação do algoritmo VSS-NLMS de Benesty[1] (discutido na Seção II-D), enquanto a Fig. 5 ilustra a implementação do algoritmo VSS-NLMS de Zipf (dado na Seção II-E). Observa-se, a partir dessas figuras, o uso das funções de processamento de sinais fornecidas na biblioteca CMSIS-DSP. Vale destacar que a estrutura de filtragem tem ordem $M = 128$ e que os valores dos parâmetros dos diferentes algoritmos foram ajustados conforme mostrado na Tabela 1.

Tabela 1. Valores utilizados para os parâmetros dos algoritmos considerados.

NLMS	VSS-NLMS de Shin	VSS-NLMS de Benesty	VSS-NLMS de Zipf
$\mu = 0,05$	$C = 0,001$	$K = 6,0$	$\beta = 0,99$
$\varepsilon = 1,0$	$\beta = 0,9961$	$\lambda = 0,9987$	$\varepsilon = 1,0$
	$\mu_{máx} = 1,5$	$\delta = 1,0$	
	$\varepsilon = 1,0$	$\varepsilon = 20\hat{\sigma}_x^2$	

IV. Resultados Alcançados

No intuito de avaliar o desempenho das implementações propostas, comparações com respeito à qualidade e inteligibilidade dos sinais de fala originais e processados foram realizadas. Para tal, sinais de áudio foram sintetizados com o auxílio de *scripts*[2] em linguagem Python a fim de emular aqueles observados na entrada primária e de referência do sistema de ANC (veja Fig. 1). Especificamente, sinais de fala (dezesseis sentenças em português do Brasil) fornecidos em [16] são considerados. Também, um arquivo de áudio com ruído branco gaussiano é gerado (entrada de referência), cuja

[1] Por simplicidade de implementação, (8) e (11) são obtidas através de $\hat{\sigma}_e^2 = (1/M)\mathbf{e}^{\mathrm{T}}(n)\mathbf{e}(n)$ e $\hat{\sigma}_x^2 = (1/M)\mathbf{x}^{\mathrm{T}}(n)\mathbf{x}(n)$, respectivamente.

[2] Os *scripts* criados, áudios utilizados e resultados obtidos estão disponíveis em [15].

```
//cálculo da energia de x[]
arm_power_f32(x,M,&energia);
//produto interno entre w[] e x[]
arm_dot_prod_f32(w,x,M,&d_chapeu);
//determinação do erro
e=d-d_chapeu;
//cálculo do fator de adaptação
arm_scale_f32(x,alfa*e/(energia+ep),fator,M);
//atualiza os coeficientes do filtro
arm_add_f32(w,fator,w,M);
```

Fig 2. Trecho de código da implementação do algoritmo NLMS.

```
//cálculo da energia de x[]
arm_power_f32(x,M,&energia);
//produto interno entre w[] e x[]
arm_dot_prod_f32(w,x,M,&d_chapeu);
//determinação do erro
e=d-d_chapeu;
//cálculo do vetor p (considerando p = pA + pB)
arm_scale_f32(p,beta,pA,M);
arm_scale_f32(x,(1-beta)*e/energia,pB,M);
arm_add_f32(pA,pB,p,M);
//cálculo da energia do vetor p
arm_power_f32(p,M,&energia_p);
//cálculo do passo variável
mu=mu_max*energia_p/(energia_p+C);
//cálculo do fator de adaptação
arm_scale_f32(x,(mu*e)/(energia+ep),fator,M);
//atualiza os coeficientes do filtro
arm_add_f32(w,fator,w,M);
```

Fig. 3. Trecho de código da implementação do algoritmo VSS-NLMS de Shin.

```
//produto interno entre w[] e x[]
arm_dot_prod_f32(x,w,M,&d_chapeu);
//determinação do erro
e=d-d_chapeu;
//estimativa da variância do sinal de erro e de x[]
memmove(&erro[1],&erro[0],(M-1)*sizeof(float32_t));
erro[0]=e;
arm_power_f32(erro,M,&energia_erro);
var_e=(1.0/M)*energia_erro;
arm_power_f32(x,M,&energia_x);
var_x=(1.0/M)*energia_x;
//define o parâmetro delta
ep=20*var_x;
//correlação entre x[] e o erro (r_ex = A + B)
arm_scale_f32(r_ex,lambda,A,M);
arm_scale_f32(x,(1-lambda)*e,B,M);
arm_add_f32(A,B,r_ex,M);
arm_power_f32(r_ex,M,&energia_r_ex);
//variância do sinal de fala (ruído de medição)
var_v=var_e-(energia_r_ex/var_x);
//cálculo do desvio padrão (erro e ruído de medição)
arm_sqrt_f32(var_e,&desv_e);
arm_sqrt_f32(var_v,&desv_v);
//fator gama
gama=(1-desv_v/(delta+desv_e)));
//escolha do fator de controle de passo
if(desv_e>=desv_v){mu=gama;}
else{mu=0;}
//cálculo do fator de adaptação
arm_scale_f32(x,mu/(ep+energia_x)*e,fator,M);
//atualização dos coeficientes do filtro adaptativo
arm_add_f32(w,fator,w,M);
```

Fig. 4. Trecho de código da implementação do algoritmo VSS-NLMS de Benesty.

```
//cálculo da energia de x[]
arm_power_f32(x,M,&energia);
//produto interno entre w[] e x[]
arm_dot_prod_f32(w,x,M,&d_chapeu);
//determinação do erro
e=d-d_chapeu;
//estimativa da correlação do sinal de erro
p=beta*p_anterior+(1-beta)*e*e_anterior;
//estimativa da variância do sinal de erro
q=beta*q_anterior+(1-beta)*(e*e);
//determinação de mu
mu=(p/q)*(p/q);
//cálculo do fator de adaptação
arm_scale_f32(x,(mu/(ep+energia)*e,fator,M);
//atualização dos coeficientes do filtro adaptativo
arm_add_f32(w,fator,w,M);
```

Fig. 5. Trecho de código da implementação do algoritmo VSS-NLMS de Zipf.

variância é ajustada de forma a produzir valores de SNR[3] de -20 dB a 20 dB (com incremento de 2 dB). Cada arquivo de áudio contendo ruído é filtrado por uma planta obtida de [17, Modelo 4] e adicionado aos sinais de fala, gerando 336 arquivos (entrada primária). Esses sinais servem como entrada para o sistema implementado na placa de desenvolvimento (o canal esquerdo da entrada *line-in* é usado como entrada de referência e o direito como entrada primária). Então, é realizada a aquisição do sinal de erro observado na saída do sistema para cada um dos sinais de entrada, produzindo assim os resultados experimentais. Por fim, três metodologias são utilizadas para avaliar o desempenho do sistema, a saber: i) *perceptual evaluation of speech quality* (PESQ) [18] para avaliar a qualidade, bem como ii) *short-time objective intelligibility measure* (STOI) [8] e iii) *application programming interface* (API) *speech-to-text* da Google Inc. [19] para avaliar a inteligibilidade.

A. Avaliação de qualidade

A Recomendação P.862 da *International Telecomunication Union-Telecommunication Standardization Sector* (ITU-T) estabelece a PESQ como um método de avaliação objetiva de qualidade de voz [18]. A PESQ consiste em comparar um áudio de referência (sinal original) com um áudio resultante de algum tipo de processamento em meio digital, o qual pode ter sofrido degradação. Essa avaliação é realizada via processamento computacional e tem como resultado uma pontuação entre $-0,5$ e $4,5$ (quanto menor a pontuação, mais baixa a qualidade do áudio avaliado). Então, para representar a pontuação na escala MOS (*mean opinion score*) [20] comumente empregada em avaliações subjetivas, recorre-se usualmente à Recomendação P.862.1 da ITU-T. Tal recomendação traz uma função de mapeamento entre a pontuação obtida através da PESQ e aquela da escala MOS, resultando assim na escala denominada PESQ (MOS-LQO) (MOS-*listening quality objetive*). Dessa forma, a faixa de valores de $-0,5$ e $4,5$ é convertida para a faixa de 1 a 4,5, sendo 1 o valor atribuído aos áudios de qualidade percebida mais baixa e 4,5 aos de qualidade percebida mais alta. Portanto, dispondo da implementação da PESQ fornecida em [18, ITU-T P.862.1], as pontuações obtidas para cada áudio foram computadas. A partir dessas pontuações, foi contabilizada a média para cada valor de SNR e os resultados obtidos são apresentados na Fig. 6.

B. Avaliação de inteligibilidade

A STOI e a API *speech-to-text* da Google Inc. são utilizadas para conduzir avaliações objetivas de inteligibilidade. Empregando um áudio de referência (assim como na PESQ), a STOI é utilizada para avaliar quanto a inteligibilidade de um áudio obtido como resultado de algum processamento foi degradada, resultando em uma pontuação de 0 a 1 (0 indica pior inteligibilidade e 1, a melhor). A partir da implementação da metodologia STOI disponível em [21], resultados experimentais foram computados e o desempenho médio obtido para cada valor de SNR é fornecido na Fig.7. Adicionalmente, a API *speech-to-text* da Google Inc. (disponível em [22]) é utilizada[4] aqui para contabilizar a taxa de acerto do número de palavras corretamente reconhecidas pelo conversor fala-texto. Para isso, áudios degradados resultantes do processamento são submetidos à API *speech-to-text* a fim de se obter uma transcrição do sinal, comparada então com a transcrição de [16]. Em seguida, a taxa de acerto é computada como a razão entre o número total de palavras reconhecidas corretamente e o número total de palavras presentes nas transcrições das sentenças de referência. Vale destacar que as palavras corretamente reconhecidas são consideradas aquelas que possuem mais de três letras e pertencem à sentença de referência analisada. Dessa forma, os resultados experimentais foram computados para cada valor de SNR e o desempenho em termos de taxa de acerto é apresentado na Fig. 8.

C. Discussão

As Figs. 6-8 ilustram o desempenho do sistema de ANC operando com os algoritmos NLMS [1] VSS-NLMS de Shin [4], VSS-NLMS de Benesty [5] e VSS-NLMS de Zipf [6] frente a sinais de fala contaminados por ruído branco. A partir da Fig. 6, observa-se que o algoritmo VSS-NLMS de Benesty exibe o melhor desempenho em termos de qualidade quando comparado aos demais, seguido do algoritmo NLMS. Por sua vez, constata-se que o algoritmo VSS-NLMS de Shin apresenta um comportamento anômalo para valores de SNR acima de 8 dB, o que pode ser justificado pela sensibilidade do algoritmo quanto ao ajuste do parâmetro C (assumido fixo para toda faixa de valores de SNR). Ainda, verifica-se que o algoritmo VSS-NLMS de Zipf exibe o pior resultado em termos de melhoria da qualidade do sinal. Além do mais, destaca-se que os algoritmos de Shin e de Zipf apresentam resultados piores para valores de SNR acima de 12 dB quando comparado aos obtidos com o sistema de ANC desligado; consequentemente, esses dois algoritmos degradam o sinal frente a tal condição de operação. Vale salientar que os resultados da PESQ (MOS-LQO) obtidos para valores de SNR inferiores a -16 dB não correspondem à qualidade perceptível do áudio e, por isso, devem ser desconsiderados na análise. Portanto, para a faixa de valores de SNR de -16 dB até 12 dB, todos os algoritmos resultaram em melhoria da qualidade dos sinais de fala [aumento da pontuação da PESQ (MOS-LQO)] em comparação com a pontuação obtida desprezando o uso do sistema de ANC.

Das Figs. 7 e 8, observa-se que os resultados obtidos através da STOI e da API *speech-to-text* da Google Inc. levam a conclusões semelhantes em termos de inteligibilidade dos sinais de fala. Especificamente, percebe-se que os algoritmos NLMS e VSS-NLMS de Benesty propiciam o melhor desempenho. Também, dado o comportamento aproximadamente linear da STOI para a condição em que o sistema de ANC está desligado, infere-se que os algoritmos VSS-NLMS de Shin e de Zipf degradam a inteligibilidade do sinal de fala para valores de SNR acima de 8 dB. Por outro lado, através da API *speech-to-text* da Google Inc., a diferença no desempenho dos algoritmos não parece tão significativa quando comparada à STOI. Na verdade, para valores de SNR acima de -12 dB, obtém-se taxas de acerto no reconhecimento de palavras acima de 80% para todos os algoritmos considerados. (Em valores de SNR acima de 0 dB, taxas de reconhecimento de palavras acima de 80% são obtidas mesmo com o sistema de ANC desligado, devido à presença de técnicas para mitigar o ruído já incorporadas na API.) Para encerrar, é possível inferir que os algoritmos considerados resultam em melhoria da inteligibilidade do

[3] A SNR é calculada através de [9] usando os sinais de fala e o ruído branco da entrada de referência, uma vez que a planta apresenta norma unitária.

[4] Um exemplo de aplicação da API *speech-to-text* da Google Inc. envolvendo a avaliação de inteligibilidade no monitoramento da evolução da síndrome de Parkinson é discutida em [23] e [24].

áudio em comparação com a condição do sistema de ANC desligado.

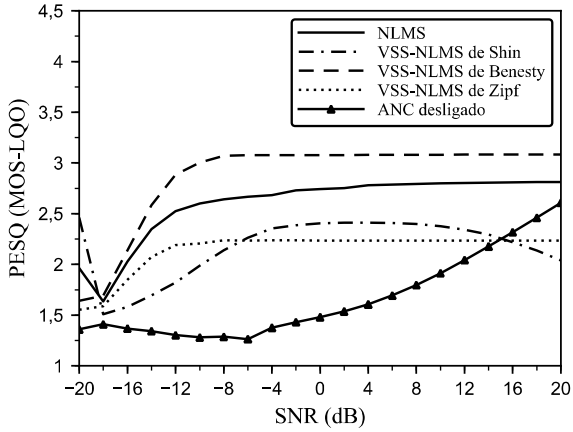

Fig 6. Curvas de qualidade obtidas por meio da metodologia PESQ.

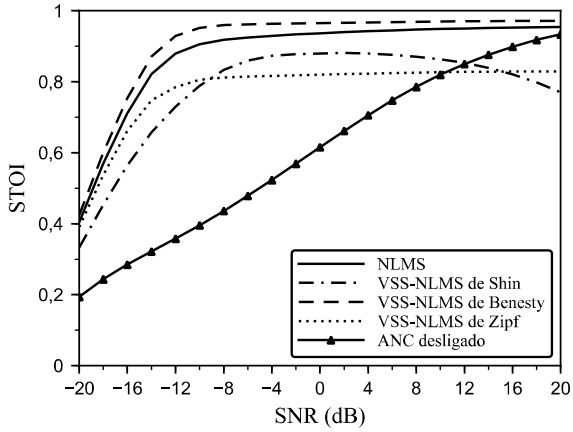

Fig 7. Curvas de inteligibilidade obtidas através da metodologia STOI.

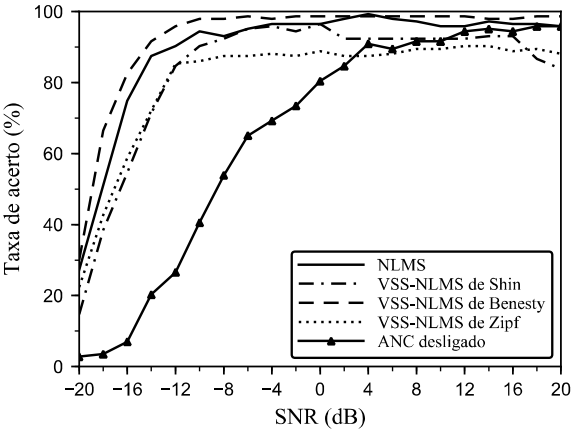

Fig 8. Curvas de inteligibilidade obtidas a partir da taxa de acerto no reconhecimento de palavras via API *speech-to-text* do Google Inc., considerando um total de 143 palavras corretamente reconhecidas dentre 135 palavras diferentes (avaliação realizada em 21 de julho de 2019).

V. Considerações Finais

Neste artigo, a implementação de um sistema de cancelamento adaptativo de ruído (ANC) foi apresentada. Para tal, a placa de desenvolvimento Cortex-FM4 *Starter* da *Cypress Semiconductors* foi utilizada já que possui recursos importantes para a aquisição e síntese de sinais de áudio. Nessa implementação, o algoritmo NLMS (*normalized least-mean-square*) e três outros algoritmos VSS-NLMS (*variable step-size* NLMS) foram avaliados em termos de atenuação de ruído envolvendo sinais de fala. Comparações de desempenho através da PESQ (*perceptual evaluation of speech quality*), STOI (*short-time objective intelligibility measure*) e API (*application programming interface*) *speech-to-text* da Google Inc. foram apresentadas e discutidas, confirmando a validade da implementação proposta. Visando dar continuidade ao presente trabalho de pesquisa, pretende-se agora avaliar o desempenho do sistema de ANC proposto operando com sinais de fala contaminados por diferentes tipos de ruído (por exemplo, ruído colorido, balbúrdia e obras em construção), construir um arranjo de microfones para integrar ao sistema, bem como analisar a complexidade computacional e o tempo de convergência dos algoritmos.

Referências

[1] S. Haykin, *Adaptive Filter Theory*, 5th ed. Upper Saddle River, NJ: Prentice-Hall, 2014.

[2] I. Panahi, N. Kehtarnavaz, and Thibodeau, "Smartphone-based noise adaptive speech enhancement for hearing aid applications", in *Ann. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Orlando, FL, USA, Aug. 2016, pp. 85-88.

[3] A. Sugiyama, R. Miyahara, and K. Oosugi, "A noise robust hearable device with an adaptive noise canceller and its DSP implementation" in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Brighton, UK, May 2019, pp. 2722-2726.

[4] H.-C. Shin, A. H. Sayed, and W.-J. Song, "Variable step-size NLMS and affine projection algorithms", in *IEEE Signal Process. Letters*, vol. 11, pp. 132-135, Feb. 2004.

[5] J. Benesty, H. Rey, L. R. Veja, and S. Tressens, "A nonparametric VSS NLMS algorithm", in *IEEE Signal Process. Letters*, vol. 13, pp. 581-584, Oct. 2006.

[6] J. G. F. Zipf, O. J. Tobias, and R. Seara, "Non-parametric VSS-NLMS algorithm with control parameter based on the error correlation," in *Proc. IEEE Int. Telecommun. Symp.*, Manaus, AM, Brazil, Sep. 2010, pp. 1-5.

[7] ITU-T Recommendation P.800 - *Methods for subjective determination of transmission quality*, Geneva, Switzerland: Int. Telecomm. Union, 1996.

[8] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time-frequency weighted noisy speech", *IEEE Trans. Audio Speech Lang. Process.*, vol. 19, pp. 2125-2136, Sep. 2011.

[9] P. C. Loizou, *Speech Enhancement: Theory and Practice*, 2nd ed. Boca Raton, FL: CRC Press, 2013.

[10] R. R. Pertum e E. V. Kuhn, "Implementação de sistema de cancelamento adaptativo de ruído utilizando um algoritmo VSS-NLMS robusto," in *Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT)*, Campina Grande, PB, Brasil, Set. 2018, pp. 45-46.

[11] S. Ciochina, C. Paleologu, and J. Benesty, "An optimized NLMS algorithm for system identification," *Signal Process.*, vol. 118, pp. 115-121, Jan. 2016.

[12] M. A. Iqbal and S. L. Grant, "Novel variable step size NLMS algorithms for echo cancellation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP),* Las Vegas, NV, 31 Mar.-4 Apr. 2008, pp. 241-244.

[13] Cypress Semiconductors, *ARM Cortex-FM4 Starter kit (S6E2CC-ETH)*, Disponível em: http://www.cypress.com/documentation/development-kitsboards/sk-fm4-176l-s6e2cc-fm4-family-quick-start-guide

[14] ARM, *Keil µVision® MDK*, Disponível em: https://www.keil.com

[15] R. R. Pertum e E. V. Kuhn. (25 de maio 2019). *Scripts criados, áudios utilizados e resultados obtidos* [Online]. Disponível em: http://lapse.td.utfpr.edu.br/downloads/artigo_sbrt2019.zip

[16] ITU-T Recommendation P.50 - *Appendix I: Real speech recordings (Brazilian Portuguese)*, Geneva, Switzerland: Int. Telecomm. Union, 2000.

[17] ITU-T Recommendation G.168 - *Digital Network Echo Cancellers*. Geneva, Switzerland: Int. Telecomm. Union, Apr. 2015.

[18] ITU-T Recommendation P.862 - *Perceptual evaluation of speech quality (PESQ)*, Geneva, Switzerland: Int. Telecomm. Union, 2005.

[19] Google Inc. (25 de fevereiro 2019). *Cloud speech-to-text* [Online]. Disponível em: https://cloud.google.com/speech-to-text/

[20] ITU-T Recommendation P.800 - *Methods for subjective determination of transmission quality*, Geneva, Switzerland: Int. Telecomm. Union, 1996.

[21] M. Pariente, FR-J: Laboratoire des Systèmes Perceptifs. (14 de maio 2019). *pystoi*. [Online]. Disponível em: https://github.com/mpariente/pystoi

[22] A. Zhang, Hypotenuse Labs. (14 de maio 2019). *SpeechRecognition*. [Online]. Disponível em: https://github.com/Uberi/speech_recognition

[23] G. Dimauro, V. Di Nicola, V. Bevilacqua, D. Caivano, and F. Girardi, "Assessment of speech intelligibility in Parkinson's disease using a speech-to-text system," *IEEE Access*, vol. 5, pp. 22199-22208, Oct. 2017.

[24] J. R. Orozco-Arroyave, "NeuroSpeech: an open-source software for Parkinson's speech analysis", *Dig. Signal Process.*, vol. 77, pp. 207-221, Jun. 2018.