Uma comparação entre técnicas adaptativas baseadas em grafos

Daniel Gilio Tiglea, Renato Candido e Magno T. M. Silva

***Resumo*— Processamento de sinais em grafos tem despertado interesse na comunidade científica, pois consiste em uma ferramenta eficiente para representar grandes quantidades de dados inter-relacionados. Encontra aplicações em áreas como redes de comunicação, análise de imagens, estimação de temperatura, etc. Recentemente, foram propostas duas soluções adaptativas para o processamento de sinais em grafos baseadas no algoritmo LMS (*least-mean-squares*). Neste artigo, elas são revisitadas e comparadas por meio de simulações. Verifica-se que uma delas apresenta maior dificuldade em acompanhar variações do sinal de entrada, o que dificulta a escolha do passo de adaptação.**

***Palavras-Chave*— Grafos, Processamento de Sinais, Filtragem Adaptativa, Algoritmo LMS.**

***Abstract*— Graph signal processing has been gaining widespread attention within the scientific community, due to the fact that it is an effective tool to deal with great quantities of interrelated data. Applications include communication networks, image analysis, temperature estimation, etc. Recently, two adaptive solutions based on the LMS (*least-mean-squares*) algorithm have been proposed. In this article, they are revisited and compared through computer simulations. It is shown that one of them has more difficulty keeping track of variations in the signal, which makes the selection of its adaptive step more difficult.**

***Keywords*— Graphs, Signal Processing, Adaptive Filtering, LMS Algorithm.**

I. Introdução

Grafos são estruturas comumente utilizadas para representar interações entre objetos de interesse, consistindo em um conjunto de **nós** e de **arestas**, que por sua vez conectam determinados pares de nós [1]. Essas estruturas são capazes de representar uma variedade de situações do mundo real, e, por esse motivo, encontram aplicações nas mais diversas áreas, tais como proteômica [2], análise de imagens [3], [4], engenharia de *software* [5], processamento de linguagem natural [6], entre outras. Nos últimos anos, com a popularização das redes sociais e de técnicas de mineração de dados, bem como o avanço de redes inteligentes em áreas como distribuição de energia e transporte de pessoas e cargas, tem havido um interesse ainda maior por parte da comunidade científica em grafos, já que eles são uma forma conveniente de representar grandes quantidades de dados relacionados entre si [7]–[9].

Em muitas situações, a cada nó do grafo está associado um valor de interesse que se deseja estimar. Por exemplo, no caso de um grafo representando uma rede elétrica, pode-se associar a cada um de seus nós um certo valor referente à geração ou ao consumo de energia naquele ponto. Diante disso, introduziu-se na literatura o conceito de **sinais definidos sobre grafos** [10], [11]. Desde então, vêm sendo propostas maneiras de se estender conceitos da teoria clássica de processamento de sinais à área de grafos, tais como filtragem e transformada de Fourier.

Recentemente, foram propostas duas soluções adaptativas baseadas no algoritmo LMS (*least-mean-squares*) para estimar sinais definidos sobre grafos [12]–[14]. Diferentes abordagens dessas soluções levaram a duas propostas distintas de extensão do algoritmo LMS clássico[1] ao processamento de sinais definidos sobre grafos. Em [12], foi proposta uma versão do algoritmo voltada a aplicações de predição. Em [13], em contrapartida, foi proposta uma outra versão, voltada principalmente a situações envolvendo identificação de sistemas.

O objetivo deste trabalho consiste em revisitar essas duas versões do algoritmo LMS voltadas ao processamento de sinais definidos sobre grafos e comparar as diferentes ideias que embasam seus desenvolvimentos. O artigo está organizado da seguinte forma. Na Seção II, são apresentadas algumas preliminares matemáticas sobre grafos e processamento de sinais definidos sobre essas estruturas. Em seguida, na Seção III, são apresentadas as versões do algoritmo LMS estendido a grafos de [12] e [13]. Na Seção IV, são mostrados os resultados de simulações computacionais aplicando-se ambas as versões do algoritmo. Por fim, na Seção V, são apresentadas as principais conclusões obtidas com base nos resultados experimentais.

II. Definições

Seja $G = (\mathcal{V}, \mathcal{E})$ um grafo dotado de N nós, em que $\mathcal{V} = \{v_1, \cdots, v_N\}$ denota o seu conjunto de nós e $\mathcal{E}$ é o seu conjunto de arestas. Representa-se um sinal definido sobre G no instante n por um vetor coluna $\mathbf{x}(n) \in \mathbb{C}^N$ da forma $\mathbf{x}(n) = [x_1(n) \cdots x_N(n)]^{\mathrm{T}}$, em que cada elemento $x_i(n)$ representa o valor do sinal no nó v_i do grafo e $(\cdot)^{\mathrm{T}}$ representa transposição [10], [11]. Na Figura 1 é mostrado um exemplo de sinal definido sobre um grafo.

Em [10], considera-se que a extensão da transformada de Fourier a sinais em grafos (GFT - *graph Fourier transform*) é dada pela projeção do sinal sobre os autovetores da matriz

$$\boldsymbol{\mathcal{L}} \triangleq \boldsymbol{\Theta} - \mathbf{A}, \tag{1}$$

em que $\boldsymbol{\Theta}$ é uma matriz diagonal, cujo elemento $\theta_{i,i}$ é igual ao número de vizinhos do nó v_i, e $\mathbf{A}$ é a matriz de adjacência do grafo em questão, cujos elementos a_{ij} são iguais a um

Daniel Gilio Tiglea, Renato Candido e Magno T. M. Silva. Departamento de Engenharia de Sistemas Eletrônicos da Escola Politécnica da Universidade de São Paulo. E-mails: daniel.tiglea@usp.br, renatocan@lps.usp.br e magno.silva@usp.br. Este trabalho foi financiado pelo CNPq (132586/2018-5 e 304715/2017-4) e pela FAPESP (2017/20378-9).

[1] Neste trabalho, para evitar confusão, o termo "LMS clássico" será utilizado para se referir ao algoritmo LMS que não é baseado em grafos [15], [16].

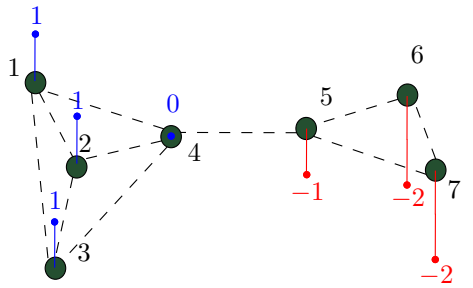

Fig. 1: Exemplo de sinal definido sobre um grafo. Os nós estão numerados de 1 a 7 e as arestas são representadas pelas linhas tracejadas, ao passo que o valor do sinal $\mathbf{x}(n)$ em cada nó i é representado pela linha contínua que se projeta daquele nó.

se $(i,j) \in \mathcal{E}$ e iguais a zero, caso contrário[2]. A matriz $\boldsymbol{\mathcal{L}}$ também é chamada de operador laplaciano para grafos [10]. É possível mostrar que, se a matriz de adjacência do grafo for simétrica, a matriz $\boldsymbol{\mathcal{L}}$ também será, e nesse caso ela admite uma decomposição em autovalores e autovetores da forma $\boldsymbol{\mathcal{L}} = \boldsymbol{\Omega}\boldsymbol{\Lambda}\boldsymbol{\Omega}^{\mathrm{H}}$, em que $(\cdot)^{\mathrm{H}}$ indica o hermitiano. Nesse caso, a GFT $\mathbf{s}(n)$ do sinal $\mathbf{x}(n)$ é dada por [10]

$$\mathbf{s}(n) = \boldsymbol{\Omega}^{\mathrm{H}}\mathbf{x}(n), \tag{2}$$

e sua anti-transformada por

$$\mathbf{x}(n) = \boldsymbol{\Omega}\mathbf{s}(n). \tag{3}$$

Já em [11], considera-se que a GFT de um sinal $\mathbf{x}(n)$ é dada pela projeção do sinal sobre os autovetores da própria matriz de adjacência $\mathbf{A}$. Nessa mesma referência, define-se um filtro linear ("*graph filter*") $\mathbf{H}$ de M coeficientes no grafo como sendo

$$\mathbf{H} = h_o\mathbf{I} + h_1\mathbf{A} + \cdots + h_{M-1}\mathbf{A}^{M-1}, \tag{4}$$

em que $h_0,\ h_1, \cdots,\ h_{M-1}$ são coeficientes possivelmente complexos. Esse resultado é obtido fazendo-se uma analogia entre o operador atraso unitário z^{-1} e a matriz de adjacência $\mathbf{A}$. Isso ocorre porque ao se multiplicar um sinal $\mathbf{x}(n)$ definido sobre um grafo pela sua matriz de adjacência, o valor do sinal resultante em cada nó é dado por uma combinação linear dos valores de $\mathbf{x}(n)$ em seus vizinhos.

III. Versões do Algoritmo LMS Baseado em Grafos

Nesta seção, revisitam-se as duas extensões do algoritmo LMS a sinais definidos sobre grafos que são comparadas neste trabalho.

A. *O Algoritmo LMS Baseado em Grafos Segundo [12]*

Em [12], desenvolve-se uma extensão do algoritmo LMS a sinais definidos sobre grafos tendo-se em mente a seguinte aplicação: dado um conjunto de amostras ruidosas coletadas sobre os nós de um grafo ao longo do tempo, como estimar o valor do sinal $\mathbf{x}(n)$ utilizando apenas um número limitado de nós, de modo a mitigar a influência do ruído?

[2]Essa definição da matriz $\mathbf{A}$ é utilizada para grafos não ponderados [1], como os considerados neste artigo.

Para obter um algoritmo capaz de atender a esses requisitos, utiliza-se o fato de que, em muitos casos, a GFT de $\mathbf{x}(n)$ pode ser considerada uma representação **esparsa** do sinal [12]. Assim, a ideia consiste em estimar em tempo real a GFT de $\mathbf{x}(n)$ seguindo-se a definição da Equação (2), de modo a permitir a reconstrução do sinal com o menor número de amostras possível a cada iteração.

O sinal amostrado nos nós do grafo é dado pelo vetor [12]

$$\mathbf{d}(n) = \mathbf{T}(n)(\mathbf{x}(n) + \mathbf{r}(n)), \tag{5}$$

em que

$$\mathbf{r}(n) \triangleq [r_1(n),\ \cdots,\ r_N(n)]^{\mathrm{T}} \tag{6}$$

é um vetor de ruído branco independente e identicamente distribuído (i.i.d.) não correlacionado com os demais sinais, e a matriz $\mathbf{T}(n)$ consiste **em um operador de amostragem**. Trata-se de uma matriz diagonal, cujo elemento $t_{i,i}(n)$ é igual a 1 se o nó v_i pertence ao conjunto $\mathcal{S}(n)$ dos nós amostrados no instante n, e igual a 0 caso contrário. É possível mostrar que uma condição necessária para que o algoritmo consiga estimar o sinal $\mathbf{x}(n)$ a partir dos nós contidos em $\mathcal{S}(n)$ é que o número de elementos desse conjunto seja maior ou igual ao número de elementos não nulos de $\mathbf{s}(n)$ [12]. Cabe observar que tanto a matriz $\mathbf{T}$ como o conjunto $\mathcal{S}$ podem variar ao longo do tempo.

Utilizando-se a Equação (3), é possível reescrever a Equação (5) como [12]

$$\mathbf{d}(n) = \mathbf{T}(n)\boldsymbol{\Omega}\mathbf{s}(n) + \mathbf{T}(n)\mathbf{r}(n). \tag{7}$$

A partir da Equação (7), é possível formular uma expressão que permita estimar a GFT $\mathbf{s}(n)$ diretamente a partir do sinal medido $\mathbf{d}(n)$. Denotando-se por $\hat{\mathbf{s}}(n)$ a estimativa de $\mathbf{s}(n)$, a estimativa $\hat{\mathbf{x}}(n)$ do sinal pode então ser obtida aplicando-se a anti-transformada de Fourier estendida a grafos a $\hat{\mathbf{s}}(n)$, por meio da expressão $\hat{\mathbf{x}}(n) = \boldsymbol{\Omega}\hat{\mathbf{s}}(n)$ [12]. Essa abordagem é vantajosa, pois permite selecionar a matriz $\mathbf{T}(n)$ em cada iteração com base no número de elementos não nulos de $\hat{\mathbf{s}}(n)$ segundo alguma estratégia de amostragem pré-determinada [12].

A partir da Equação (7), pode-se escrever o problema como [12]

$$\min_{\hat{\mathbf{s}}(n),\ \mathbf{T}(n)\in\mathcal{T}(n)} \mathbb{E}\{||\mathbf{d}(n) - \mathbf{T}(n)\boldsymbol{\Omega}\hat{\mathbf{s}}(n)||^2\} + \lambda f(\hat{\mathbf{s}}(n)), \tag{8}$$

em que $\mathcal{T}(n)$ é o conjunto que restringe a escolha da matriz $\mathbf{T}(n)$, $f(\cdot)$ é uma função introduzida a fim de forçar a esparsidade de $\hat{\mathbf{s}}(n)$, λ é um parâmetro utilizado para regular o quão esparso se deseja que $\hat{\mathbf{s}}(n)$ seja, e $\mathbb{E}\{\cdot\}$ é o operador esperança matemática.

O algoritmo LMS estendido a grafos é então obtido buscando-se uma solução recursiva para o Problema (8). Pode-se mostrar que a solução de (8) é dada por [12]

$$\boxed{\hat{\mathbf{s}}(n+1) = \mathbf{g}_\gamma\Big(\hat{\mathbf{s}}(n) + \mu\boldsymbol{\Omega}^{\mathrm{H}}\mathbf{T}(n)\big[\mathbf{d}(n) - \boldsymbol{\Omega}\hat{\mathbf{s}}(n)\big]\Big),} \tag{9}$$

em que a função $\mathbf{g}_\gamma(\cdot)$ depende da função $f(\cdot)$ escolhida para compor a função custo, e μ é um passo de adaptação como o utilizado no LMS clássico.

Quanto à função $\mathbf{g}_\gamma(\cdot)$, uma opção simples mas eficiente consiste em aplicar, a cada elemento de $\hat{\mathbf{s}}(n)$, a seguinte função [12]:

$$\mathbf{g}_\gamma(s_i) = \begin{cases} s_i, \text{ se } |s_i| > \gamma \\ 0, \text{ caso contrário} \end{cases}, \tag{10}$$

em que γ é um limiar. Em [12], γ foi feito igual a $\lambda\mu$. Considerando essa função $\mathbf{g}_\gamma(\cdot)$, o algoritmo necessita de $3N^2+N$ multiplicações, $3N^2-2N$ somas e $2N$ comparações a cada iteração para o cálculo da estimativa $\hat{\mathbf{x}}(n+1)$.

Nota-se que há uma série de diferenças entre a Equação (9) e a equação de adaptação do algoritmo LMS clássico [15], [16]. Em primeiro lugar, em vez de atualizar um vetor de coeficientes, atualiza-se em (9) a estimativa da transformada de Fourier do próprio sinal $\mathbf{x}(n)$. Além disso, uma segunda diferença consiste na utilização de uma função que força a estimativa da GFT de $\mathbf{x}(n)$ a ser esparsa, o que permite que nem todos os nós sejam utilizados na estimativa do sinal definido sobre o grafo.

B. O Algoritmo LMS Baseado em Grafos Segundo [13]

Em [13], considera-se que há um sinal desejado $\mathbf{d}(n)$ a ser estimado que está relacionado ao sinal sobre o grafo $\mathbf{x}(n)$ por meio de

$$\mathbf{d}(n) = \sum_{m=0}^{L-1} h_m^o \mathbf{A}^m \mathbf{x}(n) + \mathbf{r}(n), \tag{11}$$

em que $\mathbf{r}(n)$ é um vetor de ruído definido pela Equação (6) e $\mathbf{h}^o \triangleq [h_0^o, \cdots, h_{L-1}^o]^{\mathrm{T}}$ é o vetor com os coeficientes de um filtro no grafo que deve ser estimado [13]. Como o ruído é i.i.d., sua matriz de covariância é diagonal dada por $\mathbf{R}_r = \mathbb{E}\{\mathbf{r}(n)\mathbf{r}(n)^{\mathrm{H}}\} = \mathrm{diag}\{\sigma_{r,i}^2\}_{k=1}^N$, em que $\sigma_{r,i}^2$ é a variância do ruído no nó i. Definindo-se uma matriz $\mathbf{Z}(n)$ de dimensão $N \times M$ como

$$\mathbf{Z}(n) \triangleq [\mathbf{x}(n), \mathbf{A}\mathbf{x}(n), \cdots, \mathbf{A}^{M-1}\mathbf{x}(n)], \tag{12}$$

os coeficientes ótimos $\mathbf{h}^o$ podem ser encontrados para $M \geq L$ resolvendo-se o problema de otimização [13]

$$\mathbf{h}^o = \arg\min_{\mathbf{h}} \{\mathbb{E}||\mathbf{d}(n) - \mathbf{Z}(n)\mathbf{h}||^2\}. \tag{13}$$

Embora seja possível calcular analiticamente o valor de $\mathbf{h}^o$ por meio da Equação (13), este depende do conhecimento da matriz de autocorrelação de $\mathbf{Z}(n)$ e da correlação cruzada entre $\mathbf{Z}(n)$ e $\mathbf{d}(n)$ [13], que podem ser desconhecidas ou variar no tempo. Uma alternativa consiste em utilizar o método do gradiente estocástico [15], [16]. Fazendo-se isso, obtém-se [13]

$$\boxed{\mathbf{h}(n+1) = \mathbf{h}(n) + \mu\mathbf{Z}^{\mathrm{T}}(n)\big[\mathbf{d}(n) - \mathbf{Z}(n)\mathbf{h}(n)\big],} \tag{14}$$

em que μ é um parâmetro denominado **passo de adaptação** análogo ao utilizado no algoritmo LMS clássico.

Em termos de custo computacional, esse algoritmo necessita de $2MN + N + N^2(M-1)$ multiplicações e $N^2(M-1) + MN - M + 2N$ somas a cada iteração para calcular sua estimativa do sinal desejado, dada por $\mathbf{Z}(n)\mathbf{h}(n)$.

Nota-se que há uma analogia entre o vetor $\mathbf{h}(n)$ e a matriz $\mathbf{Z}(n)$ de (14) e os vetores de coeficientes e de entrada do algoritmo LMS clássico, respectivamente [15], [16]. Essa analogia se torna ainda mais clara quando se leva em conta o paralelo estabelecido anteriormente entre a matriz de adjacência $\mathbf{A}$ do grafo e o operador atraso unitário.

IV. Simulações Computacionais

Nesta seção, são mostrados resultados de simulação em diferentes situações. Em todas elas, considerou-se um grafo gerado aleatoriamente em cada realização segundo o modelo de Erdös-Renyi [13]. Na Figura 2, é mostrado um exemplo desse grafo com $N = 20$ nós. Além disso, em todas as simulações considerou-se que a variância do ruído é diferente em cada nó, como mostrado na Figura 3.

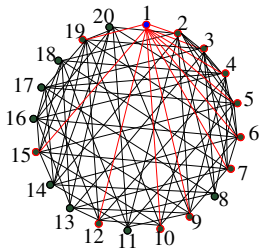

Fig. 2: Grafo gerado com $N = 20$ nós em uma realização segundo o modelo de Erdös-Renyi [13]. Em destaque, o nó 1 e suas conexões.

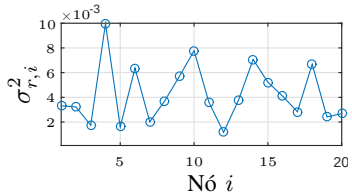

Fig. 3: Variância do ruído em cada nó do grafo.

Na subseção IV-A, o sinal desejado foi gerado segundo (11) considerando-se diferentes valores de L e de $\mathbf{h}_o$. Por outro lado, na Subseção IV-B, o sinal desejado foi gerado segundo o modelo de (5). Nas comparações de desempenho dos algoritmos, utilizou-se o MSE (*mean-square-error*), definido como

$$\mathrm{MSE}(n) \triangleq \frac{1}{N}\mathbb{E}\{||\mathbf{d}(n) - \widehat{\mathbf{d}}(n)||^2\}, \tag{15}$$

com $\widehat{\mathbf{d}}(n) = \mathbf{\Omega}\hat{\mathbf{s}}(n)$ para o LMS de [12] ou $\widehat{\mathbf{d}}(n) = \mathbf{Z}(n)\mathbf{h}(n)$ para o LMS de [13]. O MSE foi estimado com uma média de 200 realizações. Além disso, para o LMS de [12], considerou-se $\mathbf{T}(n) = \mathbf{I}$.

A. Identificação de Sistemas

Inicialmente, o sinal desejado foi gerado segundo (11) com $\mathbf{h}^o = [1\ 5]^{\mathrm{T}}$. A fim de testar a capacidade dos algoritmos de estimar sinais variantes no tempo, foram considerados sinais da forma

$$x_i(n) = b_i + c_i \sin(\omega_0 n + \phi_i), \tag{16}$$

com $i = 1, 2, \cdots, 20$. Em cada realização os escalares b_i foram gerados segundo uma distribuição gaussiana de média nula e variância unitária, ao passo que os escalares c_i e

ϕ_i foram geradas seguindo-se distribuições uniformes nos intervalos $[0, 1]$ e $[0, 2\pi]$, respectivamente. As curvas de MSE são mostradas na Figura 4, considerando-se diferentes valores de ω_0 e μ. Para o LMS de [12] considerou-se $\gamma = \lambda\mu$, com $\lambda = 0{,}5$, e para o algoritmo de [13] utilizou-se $M = 2$.

Analisando-se as Figuras 4a e 4b, é possível observar que, para $\omega_0 = 0$, ambos os algoritmos atingiram patamares de MSE semelhantes. Em ambos os casos, o erro em regime foi inferior a -20 dB, o que indica que o LMS de [12] é capaz de estimar o sinal desejado gerado segundo (11), embora não estime explicitamente o sistema $\mathbf{h}^o$. Verifica-se também que o LMS de [13] convergiu mais rapidamente do que o de [12]. Entretanto, nessas mesmas figuras se observa que o desempenho em regime do LMS de [12] piora com o aumento da frequência ω_0, o que não é observado para o LMS de [13]. Possivelmente, este resultado está associado ao fato de que o LMS de [13] atualiza diretamente as estimativas dos coeficientes do sistema $\mathbf{h}^o$, que foi considerado constante nesta simulação, ao passo que o LMS de [12] estima o sinal $\mathbf{x}(n)$, que neste caso variou no tempo para $\omega_0 \neq 0$.

Aumentando-se os passos de adaptação dos dois algoritmos para valores em que a divergência não foi observada em nenhuma realização, foram obtidas as curvas de MSE mostradas nas Figuras 4c e 4d. Com $\mu = 0{,}5$ e frequências $\omega_0 = \pi/2500$, $\omega_0 = \pi/10^4$ e $\omega_0 = 0$, o LMS de [12] alcança um patamar de MSE em regime ligeiramente superior com uma velocidade de convergência maior quando comparado ao LMS de [13] com $\mu = 0{,}001$. No entanto, ele ainda não consegue acompanhar adequadamente as variações do sinal para $\omega_0 = \pi$. Em contrapartida, o LMS de [13] apresenta uma velocidade de convergência maior com $\mu = 0{,}001$, mas atinge o mesmo patamar de MSE em regime quando adaptado com $\mu = 0{,}0005$ independentemente da frequência do sinal.

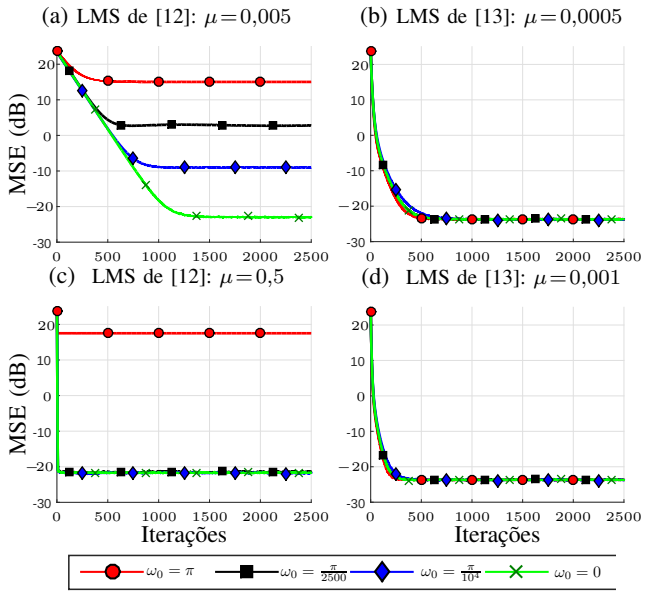

Fig. 4: MSE apresentado pelos algoritmos de [12] e [13] considerando-se $\mathbf{d}(n)$ dado por (11) com $\mathbf{h}^o = [1\ 5]$ e $\mathbf{x}(n)$ variando de forma senoidal no tempo.

A fim de testar as capacidades dos algoritmos de acompanhar variações no sistema $\mathbf{h}^o$, também foi realizada uma simulação em que o sistema $\mathbf{h}^o = [1\ \ 5\ \ 1\ \ 1]^{\mathrm{T}}$ variou abruptamente para $\mathbf{h}^o = [1\ \ 5\ \ 1\ \ 1\ \ 0{,}1]^{\mathrm{T}}$ na iteração $n = 1{,}2{\cdot}10^5$. O sinal $\mathbf{x}(n)$ foi considerado constante no tempo, mas gerado de forma aleatória em cada realização seguindo-se uma distribuição gaussiana com média nula e variância unitária.

Considerando $M = 4$ e $\mu = 2{,}5 \cdot 10^{-5}$ para o LMS de [13] e $\lambda = 0{,}5$ e $\mu = 0{,}005$ para o LMS de [12], foram obtidas as curvas de MSE da Figura 5. Nota-se que, ao contrário do que ocorreu nos gráficos da Figura 4, o algoritmo de [12] convergiu mais rapidamente do que o de [13]. Além disso, na iteração em que ocorre a variação do sistema, há um pico nas curvas de MSE dos dois algoritmos. Contudo, o LMS de [12] é capaz de voltar ao patamar de erro atingido antes da variação, enquanto que o desempenho do LMS de [13] piora, uma vez que o número de coeficientes usados para identificar o sistema passa a ser insuficiente. Esse comportamento indica que o LMS de [12] é vantajoso em situações em que o número de coeficientes do sistema é desconhecido ou varia no tempo.

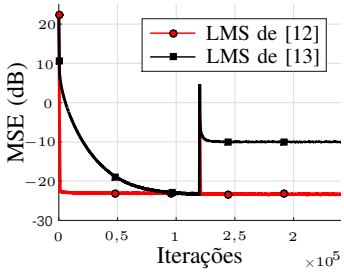

Fig. 5: MSE considerando o LMS de [12] ($\mu = 0{,}005$) e o LMS de [13] ($M = 4$, $\mu = 2{,}5 \cdot 10^{-5}$), sinal no grafo constante e sistema com variação abrupta em $n = 1{,}2 \cdot 10^5$.

Em termos de custo computacional, é interessante mencionar que nas simulações da Figura 4, o algoritmo de [12] efetuou 1220 multiplicações, 1160 somas e 40 comparações por iteração. Em contrapartida, para o LMS de [13], com $M = 2$, foi necessário calcular 500 multiplicações e 478 somas. Já nas simulações da Figura 5, a quantidade de operações efetuadas por iteração pelo algoritmo de [12] se manteve, ao passo que, com $M = 4$, o LMS de [13] efetuou 1380 multiplicações e 1316 somas.

B. Predição

Nesta seção, são mostradas simulações em que o sinal desejado foi gerado segundo (5). A exemplo do que foi feito na Subseção IV-A, foram considerados sinais na forma da Equação (16) considerando-se diferentes valores de ω_0. Na Figura 4 são mostradas as curvas de MSE obtidas com os algoritmos de [12] e [13] com diferentes passos de adaptação. Para o LMS de [12] considerou-se novamemte $\gamma = \lambda\mu$, mas agora com $\lambda = 0{,}25$. Já para o LMS de [13] considerou-se $M = 1$.

Analisando-se a Figura 6a, é possível verificar que novamente o desempenho em regime do LMS de [12] piora com o aumento da frequência ω_0. Além disso, comparando-se com os resultados da Figura 6b, constata-se que ambos os algoritmos atingiram patamares de MSE semelhantes para $\omega_0 = 0$, com o LMS de [13] convergindo mais rapidamente.

Ainda de modo semelhante ao realizado na Subseção IV-A, aumentaram-se os passos de adaptação dos dois algoritmos até os valores máximos em que a divergência não foi observada em nenhuma realização. Com isso, foram obtidas as curvas de MSE mostradas nas Figuras 6c e 6d. Nota-se que com $\mu = 0{,}5$ o LMS de [12] alcança um patamar de MSE em regime ligeiramente superior com uma velocidade de convergência maior quando comparado ao LMS de [13] com $\mu = 0{,}001$ para as frequências $\omega_0 = \pi/2500$ e $\omega_0 = \pi/10^4$. No entanto, para $\omega_0 = \pi$, o algoritmo de [12] foi novamente incapaz de acompanhar de maneira satisfatória as variações no sinal.

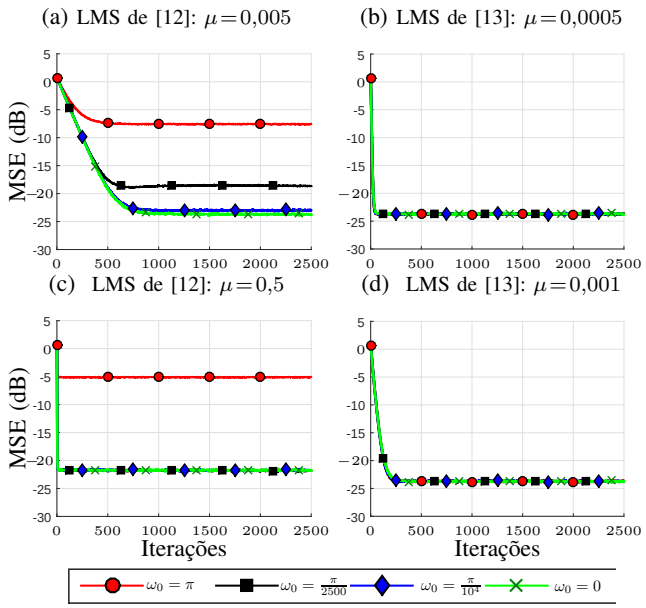

Fig. 6: MSE apresentado pelos algoritmos de [12] e [13] considerando-se $\mathbf{d}(n)$ dado por (5) com $\mathbf{x}(n)$ variando de forma senoidal no tempo.

Por fim, cabe mencionar que nas simulações apresentadas nesta subseção o algoritmo de [12] efetuou, a cada iteração, 1220 multiplicações, 1160 somas e 40 comparações, ao passo que o LMS de [13] efetuou 60 multiplicações e 59 somas.

V. Conclusões

Neste trabalho, as versões do algoritmo LMS aplicadas a sinais definidos sobre grafos propostas recentemente em [12] e em [13] foram revisitadas e comparadas. Para sinais variantes no tempo, os resultados indicam que o desempenho do algoritmo de [12] é significativamente mais sensível a variações temporais do sinal, o que não ocorre com o algoritmo de [13]. Foi mostrado que a adoção de passos de adaptação mais elevados pode reduzir essa sensibilidade para algumas taxas de variação do sinal. A escolha do passo de adaptação do LMS de [12] nesses cenários é uma questão em aberto na literatura. Verificou-se ainda que, pelo fato de estimar diretamente o sistema $\mathbf{h}^o$, o algoritmo de [13] pode sofrer mais com variações nesse sistema que o algoritmo de [12]. Do ponto de vista do custo computacional, verificou-se que quanto maior o número de coeficientes utilizados no LMS de [13], maior é o custo desse algoritmo em comparação com o LMS de [12]. Para $M \leq 3$ e $N = 20$, o algoritmo de [13] tende a ser menos custoso do ponto de vista computacional. Essa situação se inverte para $M > 3$, como exemplificado na Seção IV. Outra questão de interesse é se é possível modificar o algoritmo de [13] de modo a permitir a não utilização de todos os nós no processamento, assim como em [12]. Por fim, existem propostas de versões distribuídas dos algoritmos analisados neste trabalho, cujo estudo também é um tópico de interesse [13], [14], [17].

Agradecimentos

Os autores gostariam de agradecer aos revisores pelas sugestões enriquecedoras.

Referências

[1] P. Latouche e F. Rossi, "Graphs in machine learning: an introduction," in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2015, pp. 207-218.

[2] P. Baldi e G. Pollastri, "The principled design of large-scale recursive neural network architectures-dag-rnns and the protein structure prediction problem," *Journal of Machine Learning Research*, vol. 4, pp. 575-602, 2003.

[3] A. E. W. Mason e E. H. Blake, "A graphical representation of the state spaces of hierarchical level-of-detail scene descriptions," *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 1, pp. 70-75, 2001.

[4] E. Krahmer, S. Van Erk, e A. Verleg, "Graph-based generation of referring expressions," *Computational Linguistics*, vol. 29, no. 1, pp. 53-72, 2003.

[5] C. Collberg, S. Kobourov, J. Nagra, J. Pitts, e K. Wampler, "A system for graph-based visualization of the evolution of software," in *Proceedings of the 2003 ACM Symposium on Software Visualization*. ACM, 2003, pp. 77-86.

[6] A. Bua, M. Gori, e F. Santini, "Recursive neural networks applied to discourse representation theory," *Artificial Neural Networks-ICANN* 2002, pp. 138-138, 2002.

[7] G. A. Pagani e M. Aiello, "The power grid as a complex network: a survey," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 11, pp. 2688-2700, 2013.

[8] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, e S. Suri, "Feedback effects between similarity and social influence in online communities," in *Proceedings of the 14th International Conference on Knowledge Discovery and Data Mining*. ACM, 2008, pp. 160-168.

[9] A. Inokuchi, T. Washio, e H. Motoda, "An apriori-based algorithm for mining frequent substructures from graph data," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2000, pp. 13-23.

[10] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, e P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83-98, 2013.

[11] A. Sandryhaila e J. M.F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644-1656, 2013.

[12] P. Di Lorenzo, S. Barbarossa, P. Banelli, e S. Sardellitti, "Adaptive least mean squares estimation of graph signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 4, pp. 555-568, 2016.

[13] R. Nassif, C. Richard, J. Chen, e A. H. Sayed, "A graph diffusion LMS strategy for adaptive graph signal processing," in *Asilomar Conference on Signals, Systems, and Computers*, 2017.

[14] R. Nassif, C. Richard, J. Chen, e A. H. Sayed, "Distributed diffusion adaptation over graphs signals" in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 4129-4133, 2017.

[15] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Upper Saddle River, 4th edition, 2002.

[16] V. H. Nascimento e M. T. M. Silva, "Adaptive filters," in *Academic Press Library in Signal Processing: Signal Processing Theory and Machine Learning*, R. Chellapa e S. Theodoridis, Eds., vol. 1, chapter 12, pp. 619-761. Academic Press, Chennai, 2014.

[17] P. Di Lorenzo, P. Banelli, S. Barbarossa, e S. Sardellitti, "Distributed adaptive learning of graph signals," *IEEE Transactions on Signal Processing*, vol. 65, no. 16, pp. 4193-4208, 2017.