Projeto de Graduação

DEE DEPARTAMENTO DE ENGENHARIA ELÉTRICA

10 de Dezembro de 2018

DESENVOLVIMENTO DE ALGORITMOS ADAPTATIVOS PARA REDES DE SENSORES

Jayme Elias de Oliveira Neto

PUC RIO

Projeto de Graduação

DEE DEPARTAMENTO DE ENGENHARIA ELÉTRICA

DESENVOLVIMENTO DE ALGORITMOS ADAPTATIVOS PARA REDES DE SENSORES

Aluno: Jayme Elias de Oliveira Neto

Orientador: Rodrigo De Lamare

Trabalho apresentado com requisito parcial à conclusão do curso de Engenharia Elétrica na Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brasil.

Agradecimentos

Agradeço a meus pais Beatriz e Jackson, por todo o carinho e dedicação, ao meu irmão Márcio, que me inspira e orgulha como pessoa e futuro engenheiro.

Agradeço também aos colegas e professores da PUC-Rio, com quem pude aprender tanto nos últimos anos. Em especial ao Marcelo Balisteri, pelo ensinamento e amizade.

Por fim agradeço ao professor Rodrigo de Lamare pela oportunidade de produzir esse trabalho que foi fruto da pesquisa em Iniciação Cientifica.

Resumo

Desenvolvimento de algoritmos adaptativos pra redes de sensores

A conexão sem fio vem se tornando mais comum nos últimos anos isso aumenta a probabilidade de interferência entre sinais e com o ruido ambiente, fazendo com que o sinal na recepção apresente uma diferença considerável em relação ao sinal transmitido.

Neste trabalho são desenvolvidos algoritmos que aplicam simultaneamente técnicas de filtros adaptativos e adaptação de redes. Os algoritmos desenvolvidos têm como objetivo serem aplicados em um sistema de comunicação para dispositivos IoT e redes de sensores sem fio. O objetivo é reconfigurar os filtros e a ligação entre os nós para que erro do sinal na recepção seja o menor possível.

Em um algoritmo de filtros adaptativos temos conhecimento dos sinais de entrada e saída, começamos o processo aplicando um vetor de valor inicial 0 que em cada interação é variado em função da magnitude do erro. Com base nessa técnica podemos desenvolver variações ainda mais eficientes. Usamos variações que localizam e excluem frequências com menor probabilidade de estarem presentes no sinal alem de ajustarem a configuração da rede.

As técnicas utilizadas são variações do algoritmo Least Mean Square (LMS). As técnicas usadas para adaptação de redes são Busca Exaustiva (BE) e Explorador de Esparsidade (EE) para adaptar o filtro aplicamos a Adaptação Continuo-Discreta Alternada (ACDA). Alem disso, propõem-se combinar os algoritmos de adaptação de rede e de parâmetros em um único que gera um resultado mais exato.

A modulação por espalhamento espectral usada no LoRa tem potencial para ser a principal rede aplicada na maioria dos projetos e aplicações IoT. Nas simulações apresentaremos sinais e configurações compatíveis com o sistema LoRa.

Palavras-chave: Filtros adaptativos, Redes de Sensores sem fio, Internet das coisas

Development of algorithms for adaptive sensor networks

Abstract

The wireless connection has become more common in recent years this increases the probability of interference between signals and the environment noise, causing the signal at reception to present a considerable difference from the transmitted signal.

In this work we develop algorithms that simultaneously apply adaptive filter techniques and network adaptation. The algorithms developed are intended to be applied in a communication system for IoT devices and wireless sensor networks. The purpose is to reconfigure the filters and the connection between the nodes so that the signal error at reception is as small as possible.

In an adaptive filter algorithm we have knowledge of the input and output signals, we start the process by applying an initial value vector 0 which in each interaction is varied as a function of the magnitude of the error. Based on this technique we can develop even more efficient variations. We use variations that locate and exclude frequencies that are less likely to be present in the signal besides adjusting the network configuration.

The techniques used are variations of the Least Mean Square algorithm (LMS). The techniques used for adapting the networks are Exhaustive Search (ES) and Sparsity-Inspired (SI) to adapt the filter we apply to ACDA. In addition, it is proposed to combine the algorithms of network adaptation and parameters in a single one that generates a more accurate result.

The spread spectrum modulation used in LoRa has the potential to be the main network applied in most IoT projects and applications. In the simulations we will present signals and configurations compatible with the LoRa system.

Keywords: Adaptive Filters, Wireless Sensor Networks, Internet of Things

Sumário

Lista de Figuras

1 Introdução

A previsão para os próximos anos é haver uma expansão na quantidade de dispositivos adeptos à Internet das coisas (IoT), com isso teremos uma quantidade cada vez maior de dispositivos interligados. As aplicações estendem-se de controle remoto dos eletrodomésticos, monitoramento de plantações na agroindústria e de pacientes na medicina.

A IoT pode ser implementada por diversos sistemas que apresentam desempenho variável em alcance, taxa de dados e consumo de energia. Alguns dos mais conhecidos são LoRaWAN , Neul , Sigfox , Wi-Fi, Thread, 6LowPAN, Z-Wave, Zigbee e Bluetooth.

Nesse contexto algoritmos de filtros adaptativos baseados em uma rede distribuída são uma opção conveniente para estimar a recepção do sinal.

Os nós abordados matematicamente neste trabalho podem representar dispositivos intermediários de comunicação como gateways (LoraWan) ou estações de radio (Sigfox).

2 Modelo de sinais

Desejamos estimar o sinal transmitido s usando uma rede de sensores com N nós [1]. A densidade espectral de potência (DEP) em cada sinal s em cada frequência é representada por $\Phi_s(f)$

$$\Phi_s(f) = \sum_{m=1}^{M} b_m(f)^T \omega_0 = b_0^T(f)\omega_0 \tag{1}$$

onde $b_0(f) = [b_1(f), ..., b_M(f)]^T$ é o vetor das funções de base para cada frequência f, $\omega_0 = [\omega_{01}, ..., \omega_{0M}]$ é o vetor dos coeficientes de peso que representam a potência do sinal transmitido s em cada base e M é o numero de funções de base. Para um valor suficientemente grande de M em (1) pode-se ter uma representação aproximada do espectro. Possíveis escolhas para funções de base $b_m(f)_{m=1}^M$ são função retangular, cosseno levantado e função gaussiana. Definimos a função de transferência do canal entre o nó de transmissão do sinal s e o nó receptor k no instante de tempo i por $H(f, i)$, a DEP do sinal recebido pode ser expressa por:

$$\begin{aligned} \Phi_k(f) &= |H_k(f,i)|^2 \Phi_s(f) + v_{n,k}^2 \\ &= \sum_{m=1}^{M} |H_k(f,i)|^2 b_m(f) + v_{n,k}^2 \\ &= b_{k,i}^T(f)\omega_{0m} + v_{n,k}^2, \end{aligned} \tag{2}$$

em que $b_{k,i}^T(f) = [|H_k(f,i)|^2 b_m(f)]_{m=1}^M$ e $v_{n,k}^2$ é a potência do ruido recebido no nó k.

O modelo distribuído apresentado em (4) representa o valor da DEP em cada interação i e cada nó k, acrescentamos então as N_c amostras de frequência $f_j = f_{min} : \frac{f_{max} - f_{min}}{N_c} : f_{max}$, para $j = 1, ..., N_c$ o sinal desejado é definido como:

$$d_{k,i} = b_{k,i}^T(f_j)\omega_0 + v_{n,k}^2 + n_{k,i}(j) \tag{3}$$

em que o último termo define o ruído observado com média zero e variância $\sigma_{n,j}^2$. O ruido $v_{n,k}^2$ no receptor do nó k pode ser estimado com alta precisão e podemos subtrai-lo da expressão (5). O modelo linear desenvolvido pela medição dos N_c canais contínuos é dado por

$$d_{k,i} = B_{k,i}\omega + n_{n,i} \tag{4}$$

em que $B_{k,i} = [b_{k,i}^T(f_j)]_{j=1}^{N_c} \in \Re^{N_c x M}$ e $n_{k,i}$ é um vetor aleatório de média zero e dimensão N_c.

3 Desenvolvimento do LMS

a Introdução

O algoritmo least-mean square (LMS) foi desenvolvido por Bernard Widrow em 1960 [2]. Nessa técnica a correção do vetor de parâmetros é feita baseada apenas no erro, diferente de métodos desenvolvidos anteriormente o LMS não depende de médias temporais ou operadores matemáticos como o valor esperado sendo assim mais simples. Fazendo as modificação necessárias podemos aplicar o algoritmo em um ambiente de rede [3].

b Desenvolvimento matemático

O objetivo do algoritmo é minimizar a função de custo definida por

$$C(\omega_{k,i}) \triangleq \mathbf{E}[|\mathbf{d}_{k,i} - \mathbf{B}_{k,i}\boldsymbol{\omega}_{k,i}|^2], \qquad k = 1, ..., N \tag{5}$$

em quê $\mathbf{E}$ é o operador valor esperado, $\boldsymbol{\omega}_{k,i}$ é o vetor de coeficientes estimado pelo nó k até a interação i. Desejamos encontrar o ponto de minimo global ou seja onde:

$$\nabla C(\boldsymbol{\omega}_{k,i}) = 0 \tag{6}$$

Para isso aplicamos o gradiente na função de custo:

$$\begin{aligned}
\frac{\partial C(\boldsymbol{\omega}_{k,i})}{\partial \boldsymbol{\omega}_{k,i}} &= \frac{\partial E[|\mathbf{d}_{k,i} - \mathbf{B}_{k,i}\boldsymbol{\omega}_{k,i}|^2]}{\partial \boldsymbol{\omega}_{k,i}} && (7)\\
&= \mathbf{E}[\frac{\partial (d_{k,i} - B_{k,i}\omega_{k,i})^*(\mathbf{d}_{k,i} - \mathbf{B}_{k,i}\boldsymbol{\omega}_{k,i})}{\partial \boldsymbol{\omega}_{k,i}}] && (8)\\
&= \mathbf{E}[\mathbf{B}^*_{k,i}\mathbf{d}_{k,i} - \mathbf{B}^*_{k,i}\mathbf{B}_{k,i}\boldsymbol{\omega}_{k,i}] && (9)\\
&= b_{B,d} - \boldsymbol{R}_B\boldsymbol{\omega}_{k,i} && (10)
\end{aligned}$$

Aplicando (9) em (5) tem-se

$$\begin{aligned}
b_{B,d} - R_B\omega_{k,i} &= 0\\
R_B\omega_{k,i} &= b_{B,d}\\
\omega_{k,i} &= R_B^{-1}b_{B,d} && (11)
\end{aligned}$$

em que R_B é a função autocorrelação da matriz B, $b_{B,d}$ é a função correlação cruzada de B e d. Como esses valores são aleatórios podemos considerar:

$$\begin{aligned}
R_B &= B^*_{k,i}B_{k,i} && (12)\\
b_{B,d} &= B^*_{k,i}d_{k,i} && (13)
\end{aligned}$$

substituindo-se (12) e (13) em (11) temos então:

$$\frac{\partial C(\omega_{k,i})}{\partial \omega_{k,i}} = B^*_{k,i}d_{k,i} - B^*_{k,i}B_{k,i}\omega_{k,i} \tag{14}$$

A expressão para modificar o vetor ω é dada por:

$$\omega_{k,i+1} = \omega_{k,i} + \mu\frac{\partial C(\omega_{k,i})}{\partial \omega_{k,i}} \tag{15}$$

Aplicando (14) em (15) temos:

$$\omega_{k,i+1} = \omega_{k,i} + \mu(B^*_{k,i} d_{k,i} - B^*_{k,i} B_{k,i} \omega_{k,i}) \tag{16}$$

em que μ é uma variável arbitrária que determina o tamanho do passo.

A expressão definida acima é a de adaptação. No algoritmo usado cada nó usa sua própria informação na adaptação e usa uma combinação das adaptações em $i-1$ de seus vizinhos (define-se como vizinhos os nós com ligação entre si), cuja a expressão é mostrada abaixo

$$\omega_{k,i} = \sum_{l \epsilon \mathcal{N}_k} c_{k,l} \psi_{l,i} \tag{17}$$

em que $c_{k,l}$ é o coeficiente de combinação entre o nó k e l e $\mathcal{N}_k$ é o conjunto de nós ligados ao nó k. Esse coeficiente define o peso de cada nó na adaptação e pode ser calculado pela regra da Metropolis, Laplaciana ou vizinho mais próximo [4].

A Regra da Metropolis é implementada pela seguinte expressão:

$$c_{k,l} = \begin{cases} \frac{1}{max|\mathcal{N}_k|,|\mathcal{N}_l|}, & \text{quando } k \neq l \\ 1 - \sum\limits_{l \epsilon \mathcal{N}_k} c_{k,l}, & \text{quando} k = l \end{cases} \tag{18}$$

em que $|\mathcal{N}_k|$ representa a cardinalidade de $\mathcal{N}_k$.

Por fim o pseudo código é apresentado abaixo:

Inicializando $\omega_k = 0$
for $i = 1,...I$ **do**
  for $k = 1,...N$ **do**
    $\psi_{k,i+1} = \omega_{k,i} + \mu(B^*_{k,i} d_{k,i} - B^*_{k,i} B_{k,i} \omega_{k,i})$
  end for
  for $k = 1,...N$ **do**
    $\omega_{k,i} = \sum_{l \epsilon \mathcal{N}_k} c_{k,l} \psi_{l,i}$
  end for
end for

---

4 Algoritmos de adaptação de redes

Os algoritmos apresentados a seguir utilizam técnicas de seleção de enlaces para encontrar a melhor configuração da rede [5].

a Busca Exaustiva

Primeiro nós definimos o conjunto Ω_k de todas as combinações de enlaces para o nó k:

$$\Omega_k \triangleq C_{M,j}, \qquad j = 1, 2, ..., M \tag{19}$$

Para cada configuração nós calculamos o vetor de combinação dos parâmetros:

$$\Psi_{k,i} \triangleq \sum_{l \epsilon \Omega_k} c_{kl} \psi_{k,i} \tag{20}$$

$$e_{\Omega_k,i} \triangleq D_{k,i} - B_{k,i} \Psi_{k,i} \tag{21}$$

Nós selecionamos então o vetor associado à configuração com menor erro:

$$\widehat{\Omega_k} = |\arg\min e_{\Omega_k}| \tag{22}$$

O algoritmo é apresentado abaixo:

Inicializando $\omega_k = 0$
for $I = 1, 2, ..., i$ **do**
  for $k = 1,...N$ **do**
    $\psi_{k,i+1} = \omega_{k,i} + \mu(B^*_{k,i} d_{k,i} - B^*_{k,i} B_{k,i} \omega_{k,i})$
  end for
  for $k = 1,...N$ **do**
    Encontre todos os conjuntos de3 ω_k
    $e_{\Omega_k,i} \triangleq D_{k,i} - B_{k,i} \Psi_{k,i}$
    $\widehat{\Omega_k} = \arg\min |e_{\Omega_k}|$
    $\omega_{i,k} = \sum_{l \epsilon \Omega_k} c_{kl} \psi_{l,i}$
  end for
end for

---

b Explorador de Esparsidade

O algoritmo de LMS-BE precisa examinar todas as possíveis configurações de enlace o que demanda muito processamento. Para resolver esse problema foi desenvolvido o LMS-EE.

Primeiro consideramos a função de regularização:

$$f_1(e_{l,i}) = \sum_{l \epsilon \mathcal{N}_k} \log(1 + \epsilon |e_{l,i}|) \tag{23}$$

em que o erro $e_{l,i}$ é definido como

$$e_{l,i} \triangleq D_{k,i} - B \psi_{l,i}, \tag{24}$$

e ϵ é a magnitude de redução. Então aplicamos (22), à equação de adaptação:

$$\omega_{k,i} = \sum_{l \epsilon \mathcal{N}_k} [c_{kl} - \rho \frac{\partial f_1(e_{l,i})}{\partial e_l}] \psi_{l,i} \tag{25}$$

em que ρ é uma variável arbitraria que controla a intensidade da redução. A redução é atualizada por

$$\frac{\partial f_1(e_{l,i})}{\partial e_l} = \varepsilon \frac{sign(e_{l,i})}{1+\varepsilon|\xi_{min}|} \tag{26}$$

Em (25) o parâmetro ξ_{min} representa o valor minimo de $e_{l,i}$ em cada grupo de nós incluindo incluindo cada nó k e seus vizinhos. A função $sign(x)$ é definida como:

$$sign(x) = \begin{cases} \frac{x}{|x|} & x \neq 0 \\ 0 & x = 0 \end{cases} \tag{27}$$

Para simplificar as expressões apresentadas, representaremos as quantidades e variáveis em forma vetorial e matricial. Primeiro definimos o vetor **c** que contem os coeficientes de combinação do nó k e seus vizinhos conforme descrito por

$$\mathbf{c} \triangleq [c_{kl}]_{l \epsilon \mathcal{N}_k} \tag{28}$$

Depois introduzimos a matriz Ψ que inclui todos os vetores estimados gerados apos a equação de adaptação conforme dado por

$$\Psi \triangleq [\psi_l]_{l \epsilon \mathcal{N}_k} \tag{29}$$

O vetor **e** inclui os módulos de todos os erros calculados em (24)

$$\mathbf{e} \triangleq [|e_l|]_{l \epsilon \mathcal{N}_k} \tag{30}$$

Para implementar a exploração de esparsidade devemos modificar o vetor **e** da seguinte forma. O valor máximo será mantido, o valor minimo será modificado para $-e_{max}$ e as demais entradas serão modificadas para zero. Substituindo as representações (28)-(30) em (25), a equação de difusão é representada por

$$\begin{aligned} \omega_{k,i+1} &= \sum_{j=1}^{\mathcal{N}_k} [\mathbf{c}_j - \rho \frac{\partial f_1}{\partial \mathbf{e}_j}(\mathbf{e}_j)] \Psi_j \\ &= \sum_{j=1}^{\mathcal{N}_k} [\mathbf{c}_j - \rho\varepsilon \frac{sign(\mathbf{e}_j)}{1+\varepsilon|\xi_{min}|}] \Psi_j \end{aligned} \tag{31}$$

Por fim o algoritmo é apresentado abaixo:

Inicializando $\omega_k = 0$
for $I = 1, 2, ..., i$ **do**
  for $k = 1,...N$ **do**
    $\psi_{k,i+1} = \omega_{k,i} + \mu(B^*_{k,i} d_{k,i} - B^*_{k,i} B_{k,i} \omega_{k,i})$
  end for
  for $k = 1,...N$ **do**
    $e_{l,i} = D_{k,i} - B\psi_{l,i}$
    $\mathbf{c} = [c_{kl}]_{l \epsilon \mathcal{N}_k}$
    $\Psi = [\psi_l]_{l \epsilon \mathcal{N}_k}$
    $\mathbf{e} = [|e_l|]_{l \epsilon \mathcal{N}_k}$
    Encontre o maior e menor termo do vetor **e**
    Modifique **e** para $[0, ..., 0, e_{max}, 0, ..., 0, -e_{max}, 0, ..., 0]$
    $\xi = min(e_l)$
    $\omega_{k,i} = \sum_{j=1}^{\mathcal{N}_k} [\mathbf{c}_j - \rho\varepsilon \frac{sign(\mathbf{e}_j)}{1+\varepsilon|\xi_{min}|}] \Psi_j$
  end for
end for

---

c Adaptação Continuo-Discreta Alternada

Na Adaptação Continuo-Discreta Alternada (ACDA) acrescentamos um vetor de elementos discreto e o aplicamos ao vetor de parâmetros para que possamos zerar os pontos onde não há sinal e assim gerar uma estimação bem mais próxima do real [1].

Nesse algoritmo temos o sinal estimado em detalhes abaixo:

$$\hat{D}_{k,i} \quad = \quad B_k P_k \omega_{k,i} = B_k W_k p_k \tag{32}$$

em que P_k é uma matriz diagonal com dimensão J, cujos elementos são 1's e 0's, que é modificada conforme as iterações para melhorar a estimação. O vetor p_k contém os elementos da diagonal principal de P_k. Por fim a matriz $W_k = diag(\omega_k)$.

Para modificar o vetor $p_{k,i}$ durante cada iteração usaremos a equação abaixo:

$$p_{k,i+1} = p_{k,i} + \eta \nabla_{p_{k,i}} C(\omega_{k,i}, p_{k,i}) \tag{33}$$

Nós então aplicamos a nova definição da estimação à função custo $C(\omega_k)$.

$$C(\omega_{k,i}, p_{k,i}) \triangleq E[|D_{k,i} - B_k W_k p_k|^2], \qquad k = 1, ..., N \tag{34}$$

Como anteriormente aplicamos o gradiente em relação a $p_{k,i}$:

$$\begin{aligned} \nabla_{p_{k,i}} C(p_{k,i}, \omega_{k,i}) \quad &= \quad \frac{\partial}{\partial p_{k,i}} E[|D_{k,i} - B_k W_k p_k|^2] && (35) \\ &= \quad \frac{\partial}{\partial p_{k,i}} (E[D^2] - E[D^H BWp] - E[(BWp)^H D] - E[(BWp)^H BWp]) && (36) \end{aligned}$$

Como anteriormente podemos substituir os valores esperados pelos instantâneos:

$$\begin{aligned} \nabla_{p_{k,i}} C(p_{k,i}, \omega_{k,i}) \quad &= \quad \frac{\partial}{\partial p_{k,i}} (D^2 - D^H BWp - (BWp)^H D - (BWp)^H BWp) && (37) \\ &= \quad -D^H BWp + (BWp)^H BWp && (38) \\ &= \quad -(D - BWp) BWp && (39) \\ &= \quad -e^H BWp && (40) \end{aligned}$$

Substituindo (39) em (32) temos a equação de adaptação de p_k no ACDA:

$$p_{k,i+1} = p_{k,i} + \eta e^H B_k W_{k,i} p_{k,i} \tag{41}$$

Depois da adaptação de p_k seguimos o seguinte procedimento para que cada um dos elementos do vetor e impomos uma condição para que o valor mantenha-se igual a 1.

- Se o valor de $p^m_{k,i} > 0$ então $p^m_{k,i+1} = 1$
- Caso contrario $p^m_{k,i+1} = 0$ e fazemos uma nova análise para assegurar a confiança dessa modificação.
- Calculamos o erro com $p^m_{k,i+1}$ e se for menor que o erro com $p^m_{k,i}$ mantemos a modificação
- Caso contrario $p^m_{k,i+1} = 1$.

$$p^m_{k,i+1} = \begin{cases} 0, & Caso \quad p^m_{k,i+1} < \tau \quad e \quad |e_{k,i+1}| < |e_{k,i}| \\ 1, & Caso \quad contrario \end{cases} \tag{42}$$

Inicializando
$\omega_k = 0$
$p_k = 0$
for $i = 1, 2, ..., I$ **do**
  for $k = 1, 2, ...N$ **do**
    $e_p = d_{k,i} - B_k P_k \omega_k$
    $p_{k,i+1} = p_{k,i} + \eta e_p^H B_k W_{k,i} p_{k,i}$
  end for
  for $k = 1, ...N$ **do**
    for $j = 1, 2, ..., J$ **do**
      if $p_{k,i+1} > \tau$ **then**
        $p_{k,i} = 1$
      else
        $p_{k,i} = 0$
        $en = d_{k,i} - B_k P_k \omega_k$
        if $|en| > |e_p|$ **then**
          $p_{k,i} = 1$
        end if
      end if
    end for
    $\psi_{k,i+1} = \omega_{k,i} + \mu(B_{k,i}^* d_{k,i} - B_{k,i}^* B_{k,i} \omega_{k,i})$
  end for
  for $k = 1, ...N$ **do**
    $\omega_{i,k} = \sum_{l \epsilon \Omega_k} c_{kl} \psi_{l,i}$
  end for
end for

d Gráficos

Os gráficos apresentados a partir de agora apresentam os resultados de acordo com o erro médio quadrático(EMQ), desvio médio quadrático (DMQ), densidade espectral de potencia (DSP). A primeira análise é feita pela formula:

$$MSE(i) = \sum_{k=1}^{N} |d - B_{k,i} \omega_{k,i}|^2 \tag{43}$$

ou seja analisamos o erro entre o sinal real e o sinal estimado. Na segunda análise usamos a fórmula:

$$MSD(i) = \sum_{k=1}^{N} |\omega_0 - \omega_{k,i}|^2 \tag{44}$$

ou seja analisamos o erro entre o vetor de parâmetros real e o estimado. Na última analisamos o vetor:

$$DSP = B \omega_{k,I} \tag{45}$$

diferente das anteriores a DSP não apresenta variância no tempo, mas a distribuição de potência do sinal em cada frequência.

Para comparar a eficiência de cada algoritmo nos variamos a quantidade de nós do ambiente. Usamos um sinal de 4 funções de base de 0.7 mW acrescida de um ruído gaussiano branco de variância 0.001.

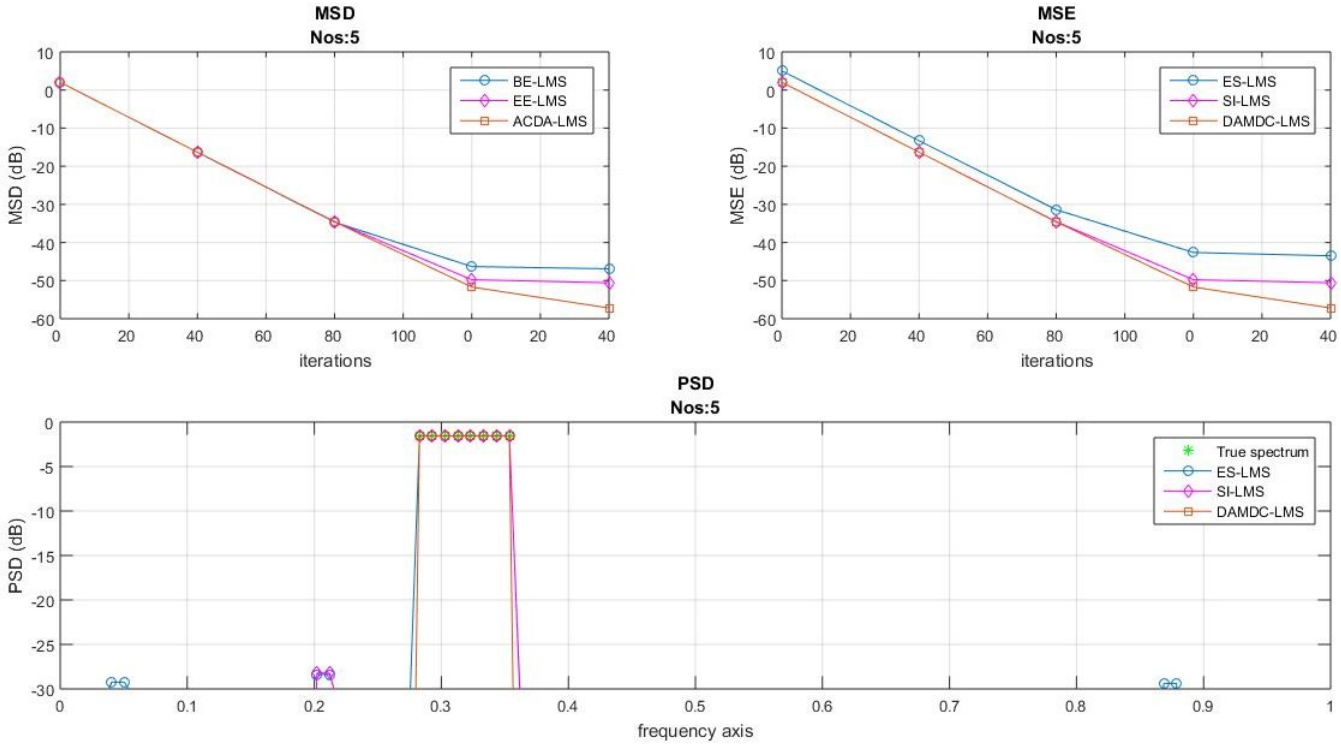

Figura 1: Graficos gerados com ruido de 0.001

O DMQ e EMQ de 5 nós apresenta o algoritmo de LMS-DAMDC como o melhor desempenho, o de LMS-EE como mediano e o de LMS-BE como pior. Pela DSP vemos que ambos apresentam um sinal muito próximo do real com uma potência de ruido desprezível.

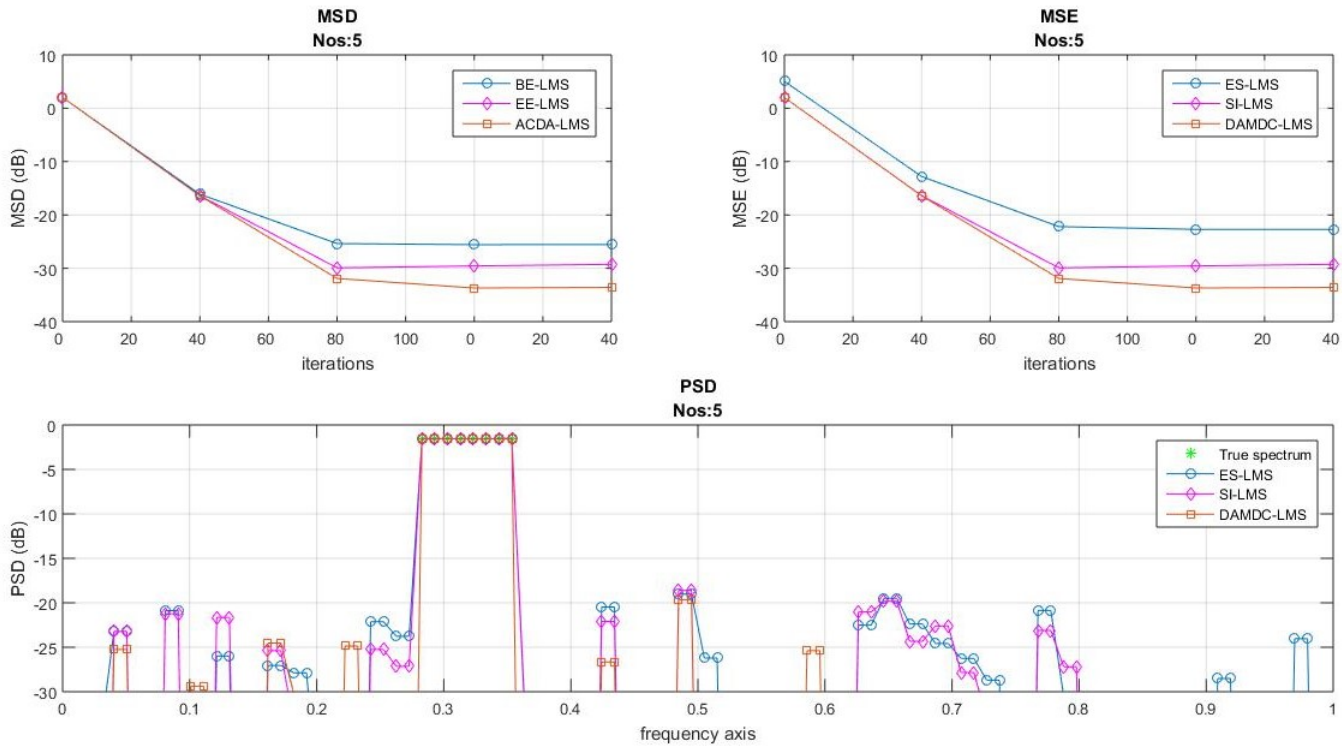

Figura 2: Graficos gerados com ruido de 0.01

No gráfico acima aumentamos a variância do ruido e com isso podemos ver que os algoritmos apresentam a mesma ordem de desempenho, porem a ACDA apresenta uma filtragem melhor nas bordas do sinal real.

e Aplicações

Os algoritmos apresentados podem ser aplicados em um sistema de comunicação que apresenta um numero considerável de receptores que estejam interligados, como consequência obteria-se um aumento da robustez e do alcance de comunicação do sistema.

Outra aplicação é um ambiente com redes de sensores que medem determinados parâmetros e utilizam o processamento distribuído para terem uma medição mais apurada.

5 Algoritmos de adaptação de rede e parâmetros

Os próximos algoritmos são feitos pela combinação da técnica de adaptação discreta de parâmetros com a adaptação de redes.

a Adaptação simultânea de parâmetros e rede usando Busca exaustiva ASPR-BE

Nesse algoritmo o vetor de parâmetros $\omega_k = 0$ é adaptado usando a técnica descrita na ACDA e produz o vetor ψ, na combinação é aplicado a esse vetor o mesmo processo descrito no algoritmo de BE. O vetor ω_k que obtemos após esse processo apresenta uma adaptação de parametros feita pelo ACDA e de rede feita pela BE.

Inicializando
$\omega_k = 0$
$p_k = 0$
for $i = 1, 2, ..., I$ **do**
  for $k = 1, 2, ...N$ **do**
    $e_p = d_{k,i} - B_k P_k \omega_k$
    $p_{k,i+1} = p_{k,i} + \eta e_p^H B_k W_{k,i} p_{k,i}$
  end for
  for $k = 1, ...N$ **do**
    for $j = 1, 2, ..., J$ **do**
      if $p_{k,i+1} > \tau$ **then**
        $p_{k,i} = 1$
      else
        $p_{k,i} = 0$
        $en = d_{k,i} - B_k P_k \omega_k$
        if $|en| > |e_p|$ **then**
          $p_{k,i} = 1$
        end if
      end if
    end for
    $\psi_{k,i+1} = \omega_{k,i} + \mu(B_{k,i}^* d_{k,i} - B_{k,i}^* B_{k,i} \omega_{k,i})$
  end for
  for $k = 1, ...N$ **do**
    Encontre todos os conjuntos de3 ω_k
    $e_{\Omega_k,i} \triangleq D_{k,i} - \Psi_{k,i}$
    $\widehat{\Omega_k} = \arg\min |e_{\Omega_k}|$
    $\omega_{i,k} = \sum_{l \epsilon \Omega_k} c_{kl} \psi_{l,i}$
  end for
end for

---

b Adaptação simultânea de parâmetros e rede usando exploração de esparsidade ASPR-EE

Nesse algoritmo o vetor de parâmetros $\omega_k = 0$ é adaptado usando a técnica descrita na ACDA e produz o vetor ψ, na combinação é aplicado a esse vetor o mesmo processo descrito no algoritmo de EE. O vetor ω_k que obtemos após esse processo apresenta uma adaptação de parâmetros feita pelo ACDA e de rede feita pela EE.

c Gráficos

Apos simular os algoritmos no MATLAB podemos gerar gráficos como anteriormente e comparar o desempenho de cada um.

Inicializando
$\omega_k = 0$
$p_k = 0$
for $i = 1, 2, ..., I$ **do**
  for $k = 1, 2, ...N$ **do**
    $e_p = d_{k,i} - B_k P_k \omega_k$
    $p_{k,i+1} = p_{k,i} + \eta e_p^H B_k W_{k,i} p_{k,i}$
  end for
  for $k = 1, ...N$ **do**
    for $j = 1, 2, ..., J$ **do**
      if $p_{k,i+1} > \tau$ **then**
        $p_{k,i} = 1$
      else
        $p_{k,i} = 0$
        $en = d_{k,i} - B_k P_k \omega_k$
        if $|en| > |e_p|$ **then**
          $p_{k,i} = 1$
        end if
      end if
    end for
    $\psi_{k,i+1} = \omega_{k,i} + \mu(B_{k,i}^* d_{k,i} - B_{k,i}^* B_{k,i} \omega_{k,i})$
  end for
  for $k = 1, ...N$ **do**
    $e_{l,i} = D_{k,i} - B\psi_{l,i}$
    $\mathbf{c} = [c_{kl}]_{l \epsilon \mathcal{N}_k}$
    $\Psi = [\psi_l]_{l \epsilon \mathcal{N}_k}$
    $\mathbf{e} = [|e_l|]_{l \epsilon \mathcal{N}_k}$
    Encontre o maior e menor termo do vetor $\mathbf{e}$
    Modifique $\mathbf{e}$ para $[0, ..., 0, e_{max}, 0, ..., 0, -e_{max}, 0, ..., 0]$
    $\xi = min(e_l)$
    $\omega_{k,i} = \sum_{j=1}^{\mathcal{N}_k} [\mathbf{c}_j - \rho\varepsilon \frac{sign(\mathbf{e}_j)}{1+\varepsilon|\xi_{min}|}] \Psi_j$
  end for
end for

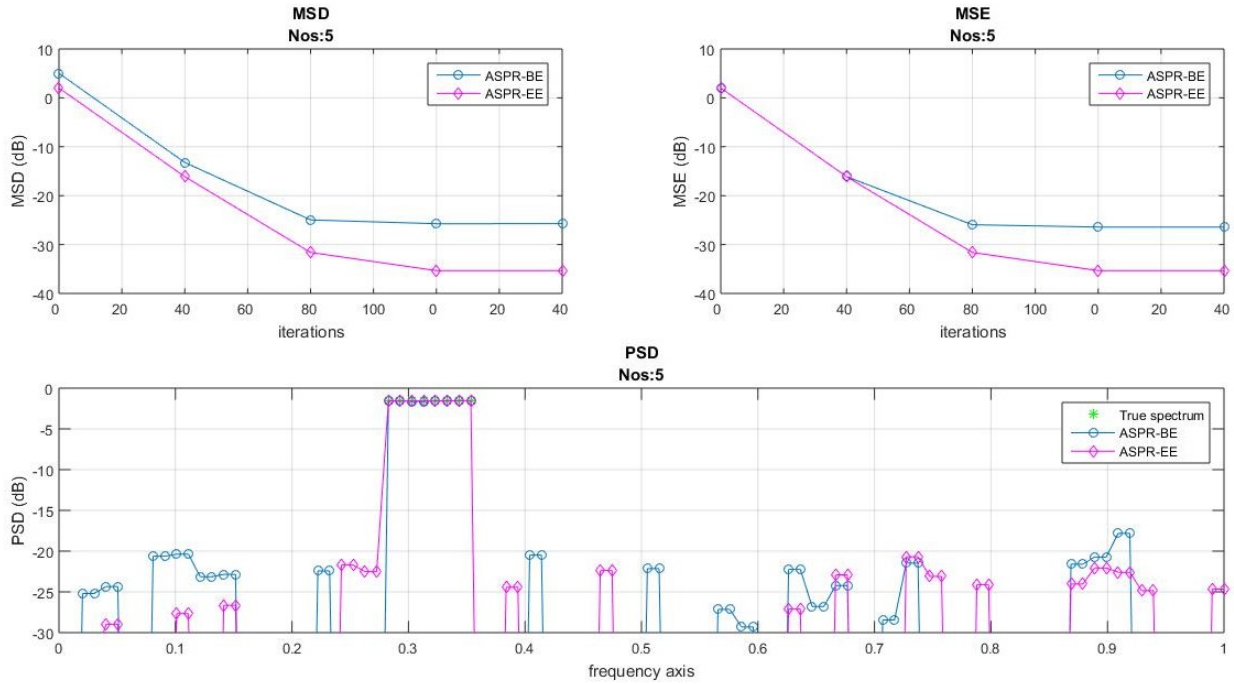

Figura 3: Gráficos gerados com ruido de 0.01

6 Aplicação em redes IoT

Neste capítulo aplicamos os algoritmos desenvolvidos a uma rede IoT simulada com o auxilio de uma USRP e um PC. O projeto LoRa apresenta grande potencial para ser o principal protocolo das aplicações IoT pelo seu alcance, economia de energia e eficiência espectral.

O projeto é dividido em camada lógica (LoRaWAN) que é mantido pela the things network seguindo o modelo de código aberto [6]. A camada física (LoRa) é patenteada pela Sentech.

a LoRaWAN

A LoRaWAN gerencia os parâmetros da camada física e aplica a codificação à mensagem.

No protocolo LoRaWAN temos como componentes [7]:

- Módulo: dispositivo acoplado a um ou mais sensores que transmite os dados para o Gateway pela modulação de espalhamento espectral usada no LoRa
- Gateway: dispositivo que recebe o sinal do módulo e traduz para a rede TCP/IP
- Servidor de Rede LoRa: responsável por gerenciar a rede. Esse componente é responsável por acrescentar novos módulos à rede, quando requeridos e para evita que um dado seja enviado de forma duplicada para a rede TCP/IP quando for recebido por mais de um gateway.
- Servidor de aplicação: provem uma interface-web e API's para a administração do usuário.

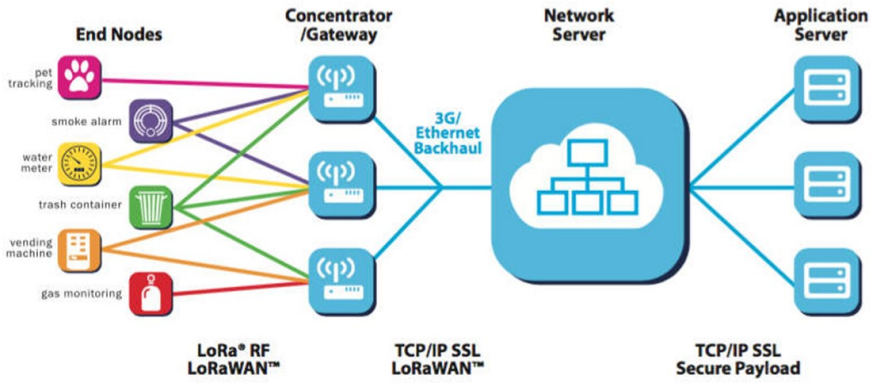

Figura 4: Representação da arquitetura LoRaWAN

O módulo LoRa faz sua comunicação por um das três classes:

- Classe A: É utilizada em aplicações que necessitam de baixo consumo de energia. A cada envio de informação para o servidor (uplink) duas janelas de transmissão são abertas, em instantes aleatórios, por tempo suficiente para a detecção do preâmbulo de transmissão. Caso a primeira janela faça a detecção a segunda não será aberta.
- Classe B: Semelhante à classe A porém apresenta agendamento nas janelas de recepção.
- Classe C: O dispositivo é configurado para que a recepção esteja continuamente habilitada. Sendo assim, consome mais energia do que um dispositivo classe A. No entanto, apresenta menor latência uma vez que o gateway não deve esperar a próxima janela agendada.

b LoRa

A modulação LoRa é baseada em um sinal de frequência variante denominado chirp e pode ser de uma frequencia menor para maior (up-chirp) ou maior para menor (down-chirp). Podemos ver esse sinal representado na Figura 3 [8].

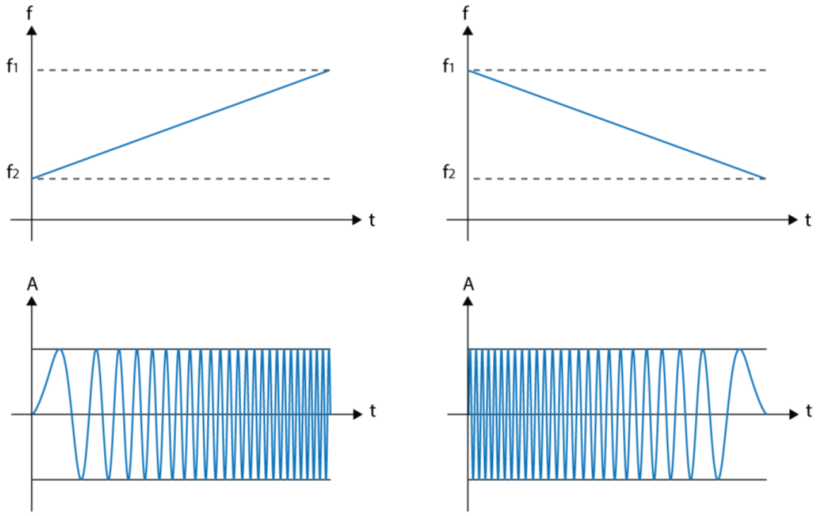

Figura 5: Representação de sinal LoRa em frequência na parte superior e em tempo na inferior, sendo up-chirp à esquerda e down-chirp à direita.

Os tempo para ir de uma determinada frequência até uma segunda é determinado pelo Spreading Factor (SF) que apresenta valor de 7 a 12. Outro parâmetro determinante é a largura de banda determina a distancia entre a frequência inicial e final. Na imagem a seguir podemos ver exemplos da modulação, sendo que da esquerda para a direta temos modulações de SF igual a 7,8 e 7 e largura de banda de 125kHz, 125kHz e 250kHz. Podemos ver que aumentar em uma unidade o SF dobra o tempo do sinal e aumentar a largura de banda reduz pela metade.

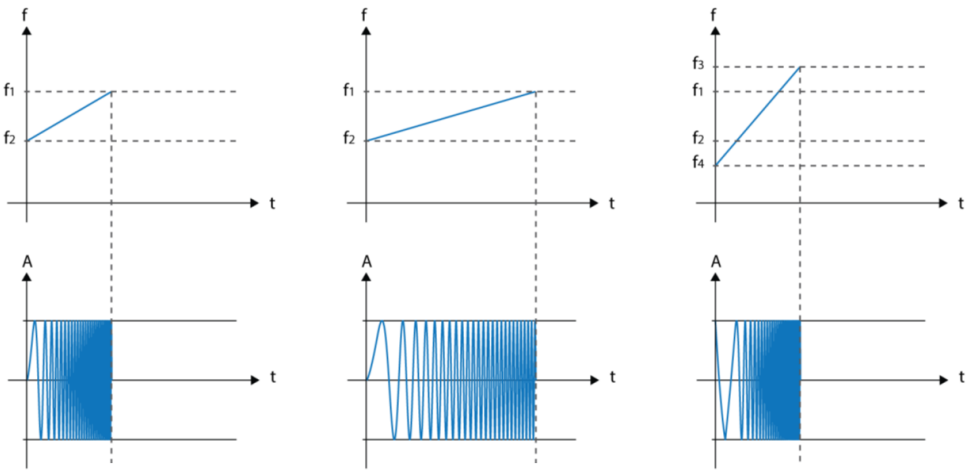

Figura 6: Grafico com representação do Spread Factor

c Simulação com LoRa

A seguir é apresentada a simulação dos algoritmos apresentados até agora sobre um sinal com modulação LoRa. Para essa simulação usamos um programa que gera o sinal.

Esse código recebe como parâmetros fator de Espalhamento (SF), largura de Banda (B), frequência de amostragem (F_s), numero de amostras (N), simbolo (s) e uma variável binaria que indica se o chirp sera up-chirp ou down-chirp.

O sinal produzido configura-se como um vetor do conteúdo espectral do sinal modulado:

$$sinal = [\cos p_1 + i \sin p_1, \cos p_2 + i \sin p_1, ..., \cos p_N + i \sin p_N] \tag{46}$$

Como podemos ver em (46) p é a variável com o valor da frequência. A inicialização de p é feita com o valor 0 e os seguintes são incrementados de acordo com (47)

$$p_{n+1} = p_n + \frac{2\pi f}{F_s} \tag{47}$$

em que f é uma variável com valor inicial:

$$f_1 = \frac{Bs}{2^{SF}} \tag{48}$$

A incrementação é feita pela frequência de offset:

$$FO = \frac{F_s}{2} - \frac{B}{2} \tag{49}$$

Como as variáveis passadas não são utilizadas apos calculados seus valores seguintes o algoritmo apresentado abaixo não utiliza equivalentes temporários.

Inicializando $p = 0$, $FO = \frac{F_s}{2} - \frac{B}{2}$ e $s = simbolo$
for k=1,..,N **do**
  $saida_k = cos(p) + i\dot{sen}(p)$
  $f = \frac{Bs}{2^{SF}}$
  if inverso $= -1$ **then**
    $f = B - f$
  end if
  $f = f + FO$
  $p = p + \frac{2\pi f}{F_s}$
  $s = s + \frac{B}{F_s}$
  if $s \geq 2^{SF}$ **then**
    $s = s - 2^{SF}$
  end if
end for

Apos gerar o sinal modulado apresentamos o gráfico de potencia pela frequência abaixo:

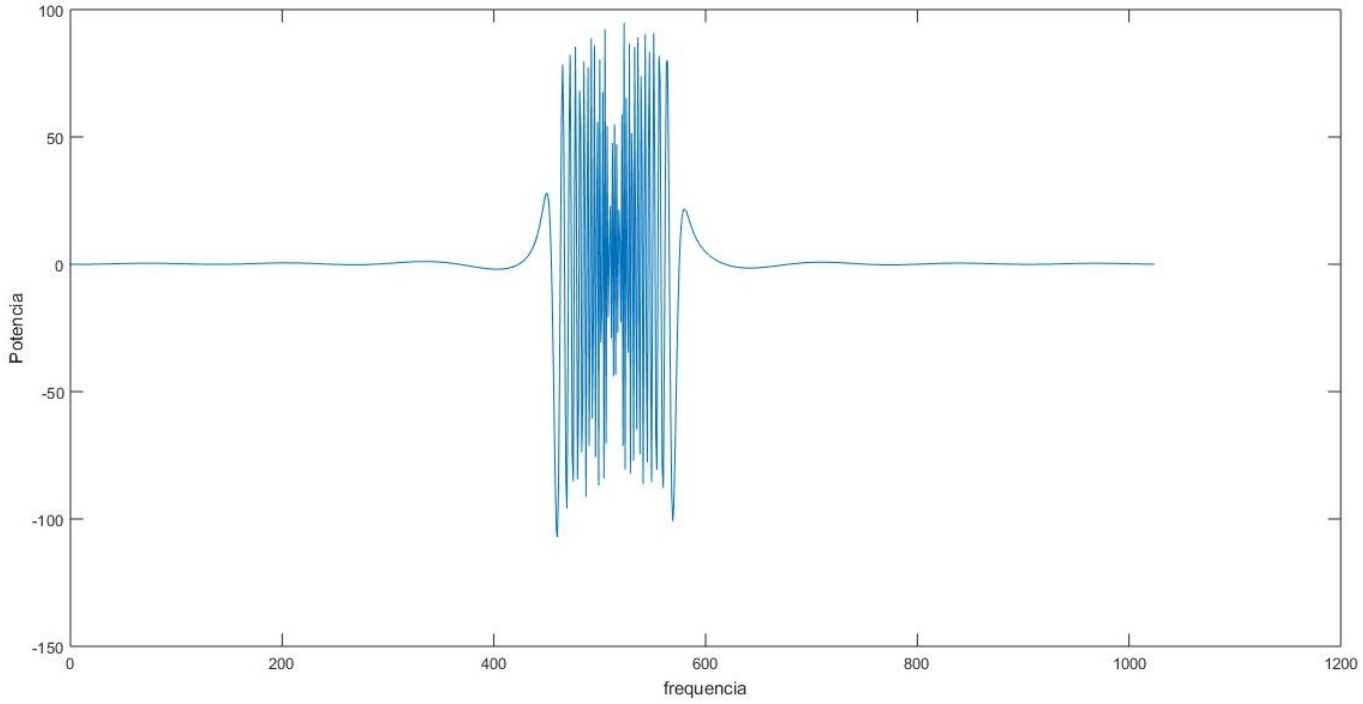

Figura 7: Gráfico da DSP do sinal com modulação LoRa

Aplicando a modulação LoRa aos algoritmos desenvolvidos temos o seguinte resultado na figura 8:

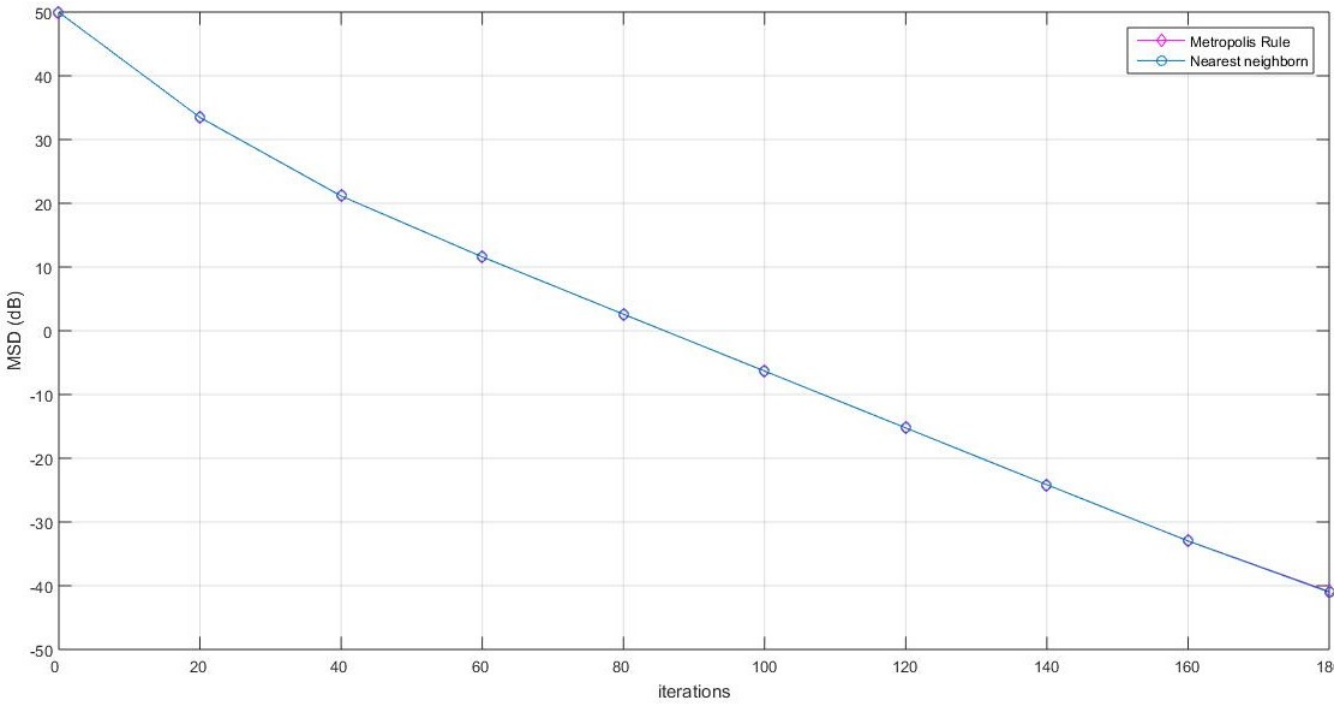

Figura 8: Gráfico do DMQ aplicado ao sinal LoRa

Vemos que o sinal responde bem a cada iteração do algoritmo que melhora rapidamente sua estimação.

7 Conclusão

Neste trabalho foram apresentados algoritmos criados a partir da técnica LMS. O uso do processamento distribuído acrescentou robustez ao algoritmo. Entre os algoritmos de adaptação de rede o EE apresentou melhor desempenho, geralmente o algoritmo de BE apresenta é melhor, porém a busca implementada vária as ligações de forma discreta apenas, o EE vária também os pesos de cada ligação e assim pode encontrar uma configuração melhor. Combinando os algoritmos de adaptação de redes com o algoritmo de adaptação de parâmetros obteve-se um resultado ainda mais eficiente. Também foi realizada a apresentação e simulação do sinal LoRa, a aplicação do algoritmo de LMS mostrou uma estimação espectral de alta eficiencia.

Como os algoritmos desenvolvidos apresentam rápida convergência e cálculos simples têm potencial de serem aplicados em dispositivos IoT que são geralmente equipados com baixo poder de processamento e pouca memoria.

Referências

[1] T. G. Miller, S. Xu, R. C. de Lamare, and H. V. Poor, "Distributed spectrum estimation based on alternating mixed discrete-continuous adaptation," *IEEE Signal Processing Letters*, vol. 23, no. 4, pp. 551–555, 2016.

[2] B. Widrow, "Adaptive filters 1: Fundamentals,‖ stanford electronics lab," 1966.

[3] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," 2007.

[4] A. H. Sayed and C. G. Lopes, "Adaptive processing over distributed networks," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 90, no. 8, pp. 1504–1510, 2007.

[5] S. Xu, R. C. de Lamare, and H. V. Poor, "Adaptive link selection algorithms for distributed estimation," *EURASIP Journal on Advances in Signal Processing*, vol. 2015, no. 1, p. 86, 2015.

[6] *Site oficial da The Things Network*. [Online]. Available: https://www.thethingsnetwork.org/

[7] *Site oficial do projeto LoRa Server*. [Online]. Available: https://www.loraserver.io/overview/architecture/

[8] E. Ruano Lin, "Lora protocol. evaluations, limitations and practical test," 2016.