Filtragem Adaptativa

Charles Casimiro Cavalcante

charles@gtel.ufc.br

Grupo de Pesquisa em Telecomunicações Sem Fio – GTEL
Programa de Pós-Graduação em Engenharia de Teleinformática
Universidade Federal do Ceará – UFC
http://www.gtel.ufc.br/∼charles

"Filtros adaptativos, os quais têm como meta transformar os sinais portadores de informação em versões 'limpas' ou 'melhoradas', ajustam suas características de acordo com os sinais encontrados. Eles formam os exemplos mais simples de algoritmos no campo de aprendizado de máquinas."

Philip A. Regalia, 2005

IEEE Control Systems Magazine, Agosto de 2005

Conteúdo do curso

1. Introdução
2. Revisão de Processos Estocásticos
3. Filtragem Linear Ótima
4. Algoritmos Recursivos no Tempo
5. Método dos Mínimos Quadrados
6. Estruturas Alternativas de Filtragem Adaptativa
7. Tópicos Avançados

Parte V

Algoritmos Recursivos

Introdução

- Estruturas adaptativas: atualização dos parâmetros para se adequar às características dos sinais de interesse
- Regras de atualização: **algoritmos de recursão temporal**
- Escolha
  1. **Critério:** o que maximizar/minimizar?
  2. **Método de busca:** como minimizar?
  3. **Complexidade:** quanto se pode "pagar" pelo desempenho?
  4. **Velocidade de convergência:** qual o tempo desejado/disponível?

Introdução - cont.

- Estruturas de filtragem: FIR $\times$ IIR
- Escolha afeta complexidade e número de interações para atingir desempenho desejado
- **FIR**
  - Estabilidade garantida
  - Critério unimodal
  - Condições de estabilidade para algoritmo de atualização mais fácil
  - Problemas reais: maior complexidade de modelagem
- **IIR**
  - Requer verificação de estabilidade
  - Critério multimodal
  - Possibilidade de garantir estabilidade do algoritmo de adaptação
  - Menor complexidade para modelar problemas reais

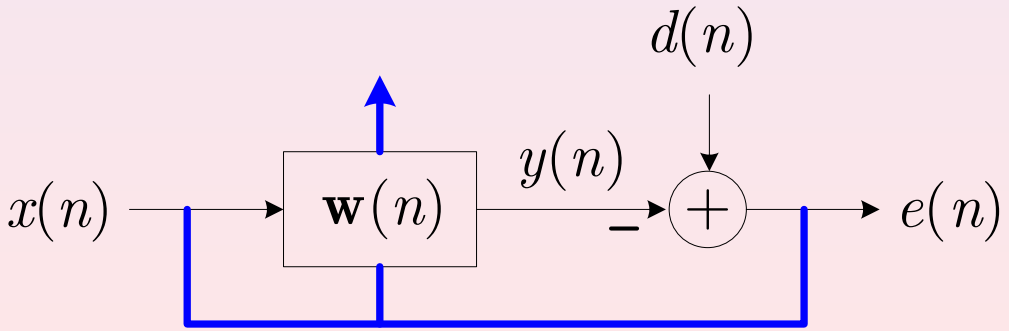
Filtragem ótima (Wiener)

1. Aquisição dos dados: obtenção de $\mathbf{R_x}$ e $\mathbf{p}_{xd}$
2. Determinação dos filtro ótimo: $\mathbf{w}_{\text{opt}} = \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd}$
3. Complexidade computacional $\sim M^3$

Filtragem adaptativa

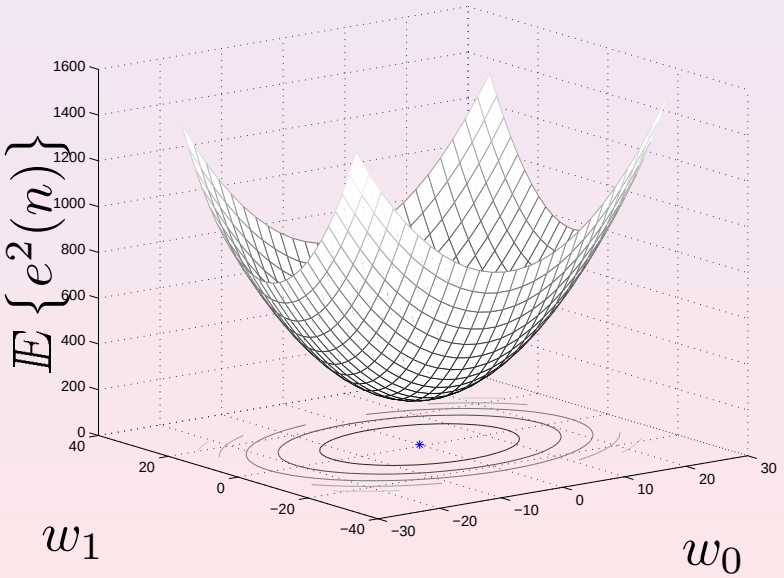
- Adquirir os dados e otimizar o sistema adaptativo ao mesmo tempo (complexidade computacional menor!)

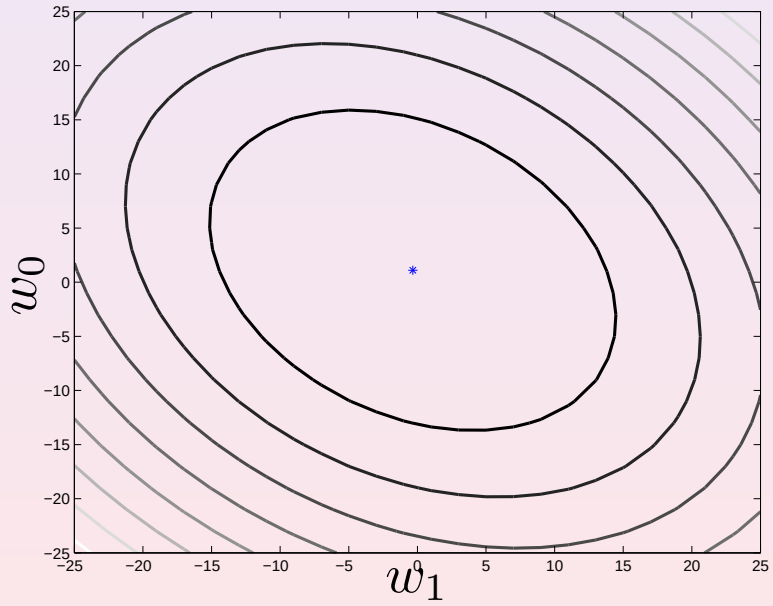
Idéia geral



Critério: atualizar $\mathbf{w}(n)$ de forma a minimizar $\mathbb{E}\left\{e^2(n)\right\}$.

Para o caso de $\mathbf{w}(n)$ com dois coeficientes $\mathbf{w}(n) = [\ w_0\ w_1\ ]^T$, temos





Métodos de busca - cont.

Otimização interativa: a partir de uma condição inicial $\mathbf{w}(0)$ chegar a $\mathbf{w}_{\text{opt}}$ para $0 < n \leq N$ iterações

Métodos de busca: baseados nos métodos clássicos de otimização de 1a (gradiente) e 2a (Newton) ordens.

Dada a função $J(\mathbf{w}) = \mathbb{E}\left\{e^2(n)\right\}$ deseja-se que

$$\mathbf{w}(n) \to \mathbf{w}(n+1) \Rightarrow J_{n+1} < J_n$$

Considerando a função $J(\mathbf{w})$ expandida em série de Taylor em torno do ponto $\mathbf{w}(n)$ tem-se

$$J(\mathbf{w})|_{\mathbf{w}(n+1)} = J(\mathbf{w})|_{\mathbf{w}(n)} + \left.\frac{\partial J}{\partial \mathbf{w}^T}\right|_{\mathbf{w}(n)} \Delta\mathbf{w}(n+1) + \frac{1}{2}\Delta\mathbf{w}^T(n+1)\left.\frac{\partial^2 J}{\partial \mathbf{w}\partial\mathbf{w}^T}\right|_{\mathbf{w}(n)} \Delta\mathbf{w}(n+1) \tag{107}$$

em que $\Delta\mathbf{w}(n+1) = \mathbf{w}(n+1) - \mathbf{w}(n)$

Métodos de busca - cont.

Dois algoritmos importantes

1. baseado na expansão de 1a ordem
2. baseado na expansão de 2a ordem

Meta: Gerar $\Delta \mathbf{w}(n+1)$ tal que $J(\mathbf{w})|_{\mathbf{w}(n+1)} < J(\mathbf{w})|_{\mathbf{w}(n)}$

1a ordem: $\frac{\partial J}{\partial \mathbf{w}^T}\Big|_{\mathbf{w}(n)} \Delta\mathbf{w}(n+1)$

Algoritmo *steepest descent* ("descida mais íngreme")

$$\mathbb{E}\left\{e^2(n)\right\} = \sigma_d^2 - 2\mathbf{w}^T\mathbf{p}_{xd} + \mathbf{w}^T\mathbf{R}_\mathbf{x}\mathbf{w}$$

$$\frac{\partial \mathbb{E}\left\{e^2(n)\right\}}{\partial \mathbf{w}} = -2\mathbf{p}_{xd} + 2\mathbf{R}_\mathbf{x}\mathbf{w}$$

Algoritmo *steepest descent* (gradiente determinístico)

$$\mathbf{w}(n+1) = \mathbf{w}(n) - 2\mu\left[\mathbf{R}_\mathbf{x}\mathbf{w}(n) - \mathbf{p}_{xd}\right] \tag{108}$$

Notar que ainda se faz necessário conhecer $\mathbf{R}_\mathbf{x}$ e $\mathbf{p}_{xd}$!

2a ordem: Método de Newton

Temos que

$$\begin{aligned}\mathbf{H}(\mathbf{w}) &= \frac{\partial^2 J}{\partial \mathbf{w} \partial \mathbf{w}^T} \\ &= 2\mathbf{R}_{\mathbf{x}}\end{aligned} \tag{109}$$

é a matriz Hessiana de $J(\mathbf{w})$.

$\mathbf{H}(\mathbf{w})$ é uma matriz definida positiva (autocorrelação), logo a aproximação quadrática tem um único e bem definido ponto de mínimo

Meta: obter $\Delta\mathbf{w}(n+1)$ tal que $\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = 0$.

Assim, temos na regra de Newton que

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) + \mu\mathbf{H}^{-1}(\mathbf{w})\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} \\ &= \mathbf{w}(n) - 2\mu\mathbf{H}^{-1}(\mathbf{w})\left[\mathbf{R}_{\mathbf{x}}\mathbf{w}(n) - \mathbf{p}_{xd}\right] \\ \mathbf{w}(n+1) &= \mathbf{w}(n) - \mu\left[\mathbf{w}(n) - \underbrace{\mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd}}_{\mathbf{w}_{\text{opt}}}\right]\end{aligned} \tag{110}$$

Algoritmo de Newton

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\left[\mathbf{w}(n) - \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd}\right] \tag{111}$$

Métodos de busca - cont.

Ainda no algoritmo de Newton, suponha

- $\mu = 1$
- $\mathbf{w}(0) = \mathbf{0}$

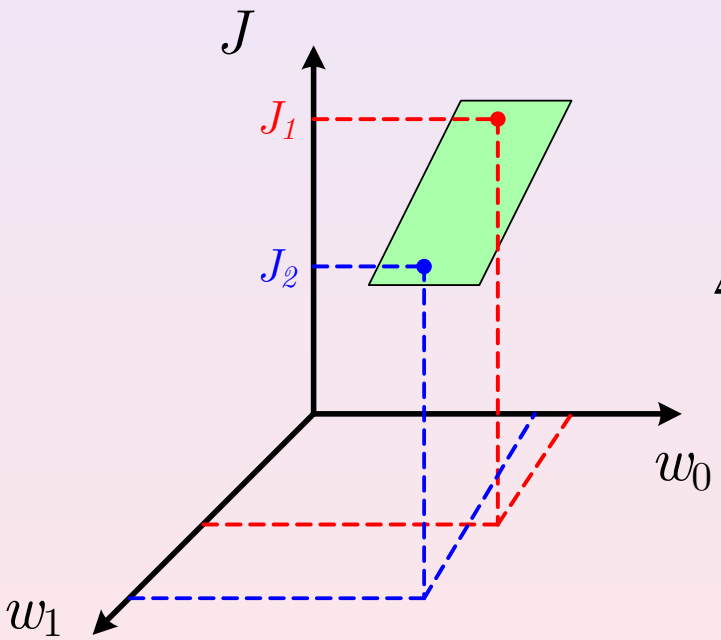
Na primeira iteração temos

$$\mathbf{w}(1) = \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd}$$

⚠ **Solução ótima em uma iteração!**

Características do *Steepest descent*

- O método de 1a ordem aproxima $J(\mathbf{w})$ como uma função linear e "caminha" nessa função com o maior "passo" (maior declividade) possível



$$\Delta \mathbf{w}_{i+1} = -\mu \nabla_{\mathbf{w}} J(\mathbf{w})$$

Características do *Steepest descent* - cont.

Método linear $\longrightarrow$ algoritmo do gradiente determinístico

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu \nabla_{\mathbf{w}} J(\mathbf{w})$$
$$\mathbf{w}(n+1) = \mathbf{w}(n) - 2\mu \left[\mathbf{R}_{\mathbf{x}}\mathbf{w}(n) - \mathbf{p}_{xd}\right]$$

Características do algoritmo de Newton

- O método de 2a ordem "aproxima" $J(\mathbf{w})$ como uma função quadrática e procura o mínimo desta função
- Encontrar um $\Delta\mathbf{w}$ que nos leve ao mínimo, ou seja, a uma condição de gradiente nulo, no passo $i+1$.
- Obter $\Delta\mathbf{w}_{i+1}$ tal que $\left.\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}}\right|_{\mathbf{w}_{i+1}} = 0$

$$\nabla_{\mathbf{w}} J(\mathbf{w}) + \mathbf{H}(\mathbf{w})\Delta\mathbf{w} = 0$$
$$\Delta\mathbf{w} = -\mathbf{H}^{-1}(\mathbf{w}) \cdot \nabla_{\mathbf{w}} J(\mathbf{w})$$

Como $\mathbf{H}(\mathbf{w}) = 2\mathbf{R}_{\mathbf{x}}$ tem-se

Características do algoritmo de Newton - cont.

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \frac{1}{2}\mathbf{R}_{\mathbf{x}}^{-1} \cdot \nabla_{\mathbf{w}} J(\mathbf{w})$$

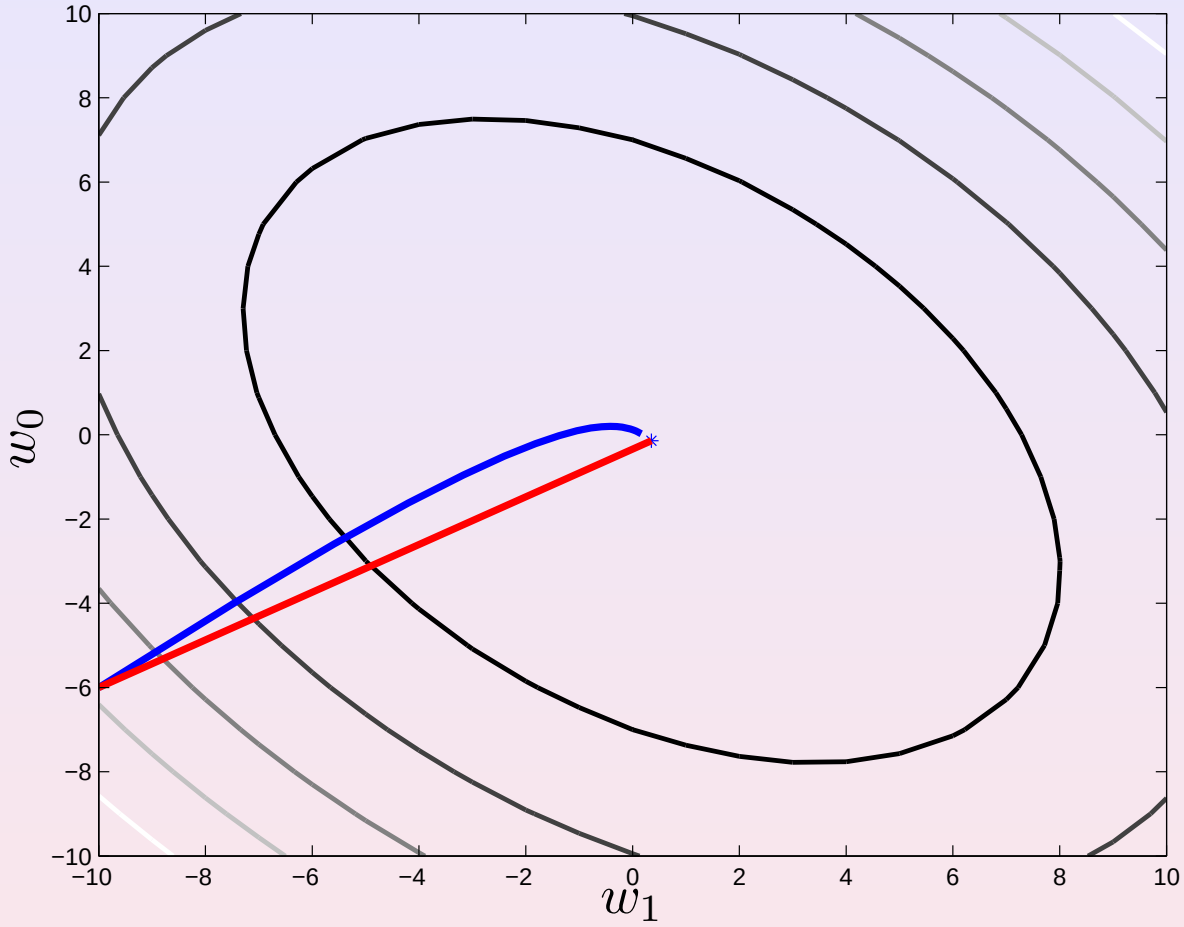
$$\mathbf{w}_{i+1} = \mathbf{w}_i + \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd} - \mathbf{w}_i$$

$$\boxed{\mathbf{w}_{i+1} = \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd}}$$

que é a própria solução ótima.

Steepest descent $\times$ Algoritmo de Newton

- O algoritmo *steepest decent* busca encontrar, à cada iteração, em qual direção a função decresce mais rapidamente (gradiente descendente)
- O método de Newton, calcula para a função, qual a direção, a partir do ponto inicial, que chega mais rapidamente ao ponto ótimo.
- Método de Newton é mais complexo (inversão de matriz) e mais rápido. Para $J(\mathbf{w})$ quadrático, uma iteração é suficiente se $\mu = 1$.
- *Steepest descent* é mais simples, mas tem uma latência maior para convergir ao ponto ótimo.



Convergência dos algoritmos *steepest descent* (azul) e de Newton (vermelho). Passos $\mu_{sd} = 0.1$ e $\mu_n = 1$.

Problemas

- Equações necessitam conhecimento ou estimativa das estatísticas $\mathbf{R_x}$ e $\mathbf{p}_{xd}$
- Processamento caro e não garante uma convergência ao valor desejado
- **Alternativa**: aproximações estocásticas

Origem

H. Robbins and S. Monro, "A Stochastic Approximation Method", *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400-407, 1951

Idéia: estimação recursiva de um determinado número de parâmetros θ, de forma:

$$\theta(n) = \theta(n-1) - \mu(n) \cdot f\left[\theta(n-1), x(n)\right] \tag{112}$$

em que
$x(n) =$ dados observados no tempo
$\mu(n) =$ seqüência decrescente
$f(\cdot) =$ função dos dados e parâmetros

Exemplo

Sejam

$$\theta(0) = 0$$

$$\mu = \frac{1}{n}$$

$$f[\theta(n-1), x(n)] = \theta(n-1) - x(n)$$

daí decorre que

$$\theta(n) = \frac{x(1) + x(2) + \ldots + x(n)}{n}$$

1a observação: O algoritmo de Robbins-Monro converge para $f[\theta(n-1), x(n)] = 0$.

Supondo várias realizações do algoritmo
θ_{opt} é tal que $\mathbb{E}\{f[\theta(n-1), x(n)]\} = 0$

No nosso caso (filtragem): quem é $\mathbb{E}\{f[\theta(n-1), x(n)]\}$?

Sabemos que $\nabla_{\mathbf{w}} \mathbb{E}\{f[\theta(n-1), x(n)]\} = 0$ para $\mathbf{w} = \mathbf{w}_{\text{opt}}$ então

$$f[\theta(n-1), x(n)] = \frac{\partial e^2(n)}{\partial \mathbf{w}} \tag{113}$$

Partindo da aproximação da Eq. (113), a recursão temporal para aproximar o critério de minimizar o erro quadrático médio seria do tipo:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu(n+1)\mathbf{x}(n)e(n) \tag{114}$$

em que $e(n) = d(n) - \mathbf{w}^T(n)\mathbf{x}(n)$.

Se considerarmos $\mu(n+1) = \mu$ teremos então o **algoritmo do gradiente estocástico** dado por

$$\boxed{\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{x}(n)e(n)}$$

Gradiente determinístico: Busca na direção negativa do gradiente

$$\mathbf{w}(n+1) = \mathbf{w}(n) - 2\mu\left[\mathbf{R}_{\mathbf{x}}\mathbf{w}(n) - \mathbf{p}_{xd}\right]$$

Algoritmo de Newton: Mais rápido e mais complexo

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu\left[\mathbf{w}(n) - \mathbf{R}_{\mathbf{x}}^{-1}\mathbf{p}_{xd}\right]$$

Gradiente estocástico: Mais simples, menos requisitos, desempenho pior

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{x}(n)e(n)$$

Algoritmos estocásticos

- Diferentes aproximações podem ser realizadas
- Meta é reduzir a complexidade dos algoritmos provendo uma convergência para o ponto ótimo
- Algumas técnicas são discutidas a seguir

O algoritmo LMS (*Least Mean Square*) é um algoritmo de busca que utiliza uma simplificação do vetor gradiente por meio de uma modificação na função custo (objetivo)

Propriedades

- Simplicidade computacional
- Prova de convergência em ambiente estacionário
- Convergência não-polarizada, em média, para a solução ótima (Wiener)

Se tomarmos o gradiente estocástico temos então

$$\mathbf{w}(n+1) = \mathbf{w}(n) - 2\mu \left[\mathbf{R}_{\mathbf{x}}\mathbf{w}(n) - \mathbf{p}_{xd}\right]$$

mas, deseja-se trabalhar com estimativas das estatísticas no instante n, uma vez que as mesmas podem não estar disponíveis completamente, então teremos algo como

$$\mathbf{w}(n+1) = \mathbf{w}(n) - 2\mu \left[\widehat{\mathbf{R}}_{\mathbf{x}}(n)\mathbf{w}(n) - \widehat{\mathbf{p}}_{xd}(n)\right] \tag{115}$$

Então, uma solução possível é fazer uma aproximação das estatísticas por seus valores instantâneos, ou seja

$$\begin{aligned} \mathbf{R}_{\mathbf{x}} = \mathbb{E}\left\{\mathbf{x}(n)\mathbf{x}^T(n)\right\} &\approx \widehat{\mathbf{R}}_{\mathbf{x}}(n) = \mathbf{x}(n)\mathbf{x}^T(n) \\ \mathbf{p}_{xd} = \mathbb{E}\left\{\mathbf{x}(n)d(n)\right\} &\approx \widehat{\mathbf{p}}_{xd}(n) = \mathbf{x}(n)d(n) \end{aligned} \tag{116}$$

Desta forma, teremos

$$\begin{aligned}\mathbf{w}(n+1) &= \mathbf{w}(n) - 2\mu\left[\widehat{\mathbf{R}}_{\mathbf{x}}(n)\mathbf{w}(n) - \widehat{\mathbf{p}}_{xd}(n)\right] \\ &= \mathbf{w}(n) - 2\mu\left[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - \mathbf{x}(n)d(n)\right] \\ &= \mathbf{w}(n) - 2\mu\mathbf{x}(n)\left[\mathbf{x}^T(n)\mathbf{w}(n) - d(n)\right] \\ &= \mathbf{w}(n) - 2\mu\mathbf{x}(n)\left[y(n) - d(n)\right]\end{aligned} \tag{117}$$

Então, a equação de recursão do LMS é dada por:

Algoritmo LMS

$$\boxed{\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu\mathbf{x}(n)e(n)} \tag{118}$$

Note que o algoritmo LMS possui a mesma regra de recursão que a aproximação do gradiente estocástico, por isto é comum usar a mesma notação para ambos.

Uma questão importante reside na garantia da convergência do algoritmo para os parâmetros ótimos. Bem como observar se esta convergência é não-polarizada.

Gradiente: o gradiente do algoritmo converge para algum valor?

Tomando as expressões do gradiente para o algoritmo determinístico e do LMS

$$\begin{aligned} \nabla_{\text{det}} &= 2\left[\mathbf{R}_{\mathbf{x}}\mathbf{w}(n) - \mathbf{p}_{xd}\right] \\ \nabla_{\text{LMS}} &= 2\left[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - \mathbf{x}(n)d(n)\right] \end{aligned} \tag{119}$$

podemos ver que as direções determinadas por ambos os algoritmos são diferentes (como esperado). Entretanto, se tomarmos o valor médio no caso do LMS temos

$$\begin{aligned} \mathbb{E}\left\{\nabla_{\text{LMS}}\right\} &= \mathbb{E}\left\{2\left[\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - \mathbf{x}(n)d(n)\right]\right\} \\ &= 2\left[\mathbb{E}\left\{\mathbf{x}(n)\mathbf{x}^T(n)\right\}\mathbf{w}(n) - \mathbb{E}\left\{\mathbf{x}(n)d(n)\right\}\right] \\ &= 2\left[\mathbf{R}_{\mathbf{x}}\mathbf{w}(n) - \mathbf{p}_{xd}\right] = \nabla_{\text{det}} \end{aligned} \tag{120}$$

Estabilidade: quais os valores de μ para os quais o algoritmo converge?

Vamos considerar uma perturbação do vetor de coeficientes em torno do filtro ótimo, assim temos

$$\Delta\mathbf{w}(n) = \mathbf{w}(n) - \mathbf{w}_{\text{opt}} \tag{121}$$

Utilizando esta definição, podemos escrever o LMS como

$$\begin{aligned}
\Delta\mathbf{w}(n+1) &= \Delta\mathbf{w}(n) + 2\mu e(n)\mathbf{x}(n) \\
&= \Delta\mathbf{w}(n) + 2\mu\mathbf{x}(n)\left[\mathbf{x}(n)^T\mathbf{w}_{\text{opt}} + b(n) - \mathbf{x}(n)^T\mathbf{w}(n)\right] \\
&= \Delta\mathbf{w}(n) + 2\mu\mathbf{x}(n)\left[e_{\text{opt}}(n) - \mathbf{x}(n)^T\Delta\mathbf{w}(n)\right] \\
&= \left[\mathbf{I} - 2\mu\mathbf{x}(n)\mathbf{x}(n)^T\right]\Delta\mathbf{w}(n) + 2\mu e_{\text{opt}}(n)\mathbf{x}(n)
\end{aligned} \tag{122}$$

Sabendo que $e_{\text{opt}}(n) = d(n) - \mathbf{x}(n)^T \mathbf{w}_{\text{opt}} = b(n)$ temos então, o valor esperado de

$$\begin{aligned}\mathbb{E}\left\{\Delta\mathbf{w}(n+1)\right\} = \mathbb{E}\left\{\left[\mathbf{I} - 2\mu\mathbf{x}(n)\mathbf{x}(n)^T\right]\Delta\mathbf{w}(n)\right\} \\ + 2\mu\mathbb{E}\left\{e_{\text{opt}}(n)\mathbf{x}(n)\right\}\end{aligned} \tag{123}$$

Assumindo independência entre $\mathbf{x}(n)$, $\Delta\mathbf{w}(n)$ e $e_{\text{opt}}(n)$, temos então

$$\begin{aligned}\mathbb{E}\left\{\Delta\mathbf{w}(n+1)\right\} &= \left[\mathbf{I} - 2\mu\mathbb{E}\left\{\mathbf{x}(n)\mathbf{x}(n)^T\right\}\right]\mathbb{E}\left\{\Delta\mathbf{w}(n)\right\} \\ &= (\mathbf{I} - 2\mu\mathbf{R}_{\mathbf{x}})\mathbb{E}\left\{\Delta\mathbf{w}(n)\right\}\end{aligned} \tag{124}$$

Um fator que nos ajuda é saber que podemos decompor a matriz $\mathbf{R}_{\mathbf{x}}$ como

$$\mathbf{R}_{\mathbf{x}} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T \tag{125}$$

em que $\mathbf{Q}$ é a matriz (ortogonal) dos autovetores de $\mathbf{R}_{\mathbf{x}}$

Pré-multiplicando então a Eq. (124) por $\mathbf{Q}^T$ temos

$$\mathbb{E}\left\{\mathbf{Q}^T\Delta\mathbf{w}(n+1)\right\} = (\mathbf{I} - 2\mu\mathbf{Q}^T\mathbf{R_x}\mathbf{Q})\mathbb{E}\left\{\mathbf{Q}^T\Delta\mathbf{w}(n)\right\} \quad (126)$$

Mas sabe-se ainda que

$$\begin{aligned}\mathbf{R_x} &= \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T \\ \mathbf{Q}^T\mathbf{R_x} &= \mathbf{Q}^T\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T \\ &= \mathbf{\Lambda}\mathbf{Q}^T\end{aligned}$$

E podemos então definir

$$\mathbf{v}(n+1) = \mathbb{E}\left\{\mathbf{Q}^T\Delta\mathbf{w}(n+1)\right\} \quad (127)$$

que são versões rotacionadas dos erros dos coeficientes.

Daí, temos então

$$\begin{aligned}\mathbf{v}(n+1) &= \mathbf{v}(n) - 2\mu\mathbf{\Lambda}\mathbf{v}(n) \\ &= (I - 2\mu\mathbf{\Lambda})\mathbf{V}(n)\end{aligned} \tag{128}$$

Ou seja, para cada elemento $v_k(n+1)$ do vetor $\mathbf{v}(n+1)$ temos

$$v_k(n+1) = (1 - 2\mu\lambda_k)v_k(n) \tag{129}$$

Condição de estabilidade:

$$\begin{aligned}|1 - 2\mu\lambda_k| < 1 \quad &\rightarrow \quad -1 < 1 - 2\mu\lambda_k < 1 \\ 0 < 2\mu\lambda_k < 2 \quad &\rightarrow \quad 0 < \mu < \frac{1}{\lambda_k}\end{aligned} \tag{130}$$

$$\boxed{\text{Estabilidade: } 0 < \mu < \frac{1}{\lambda_{\text{max}}}}$$

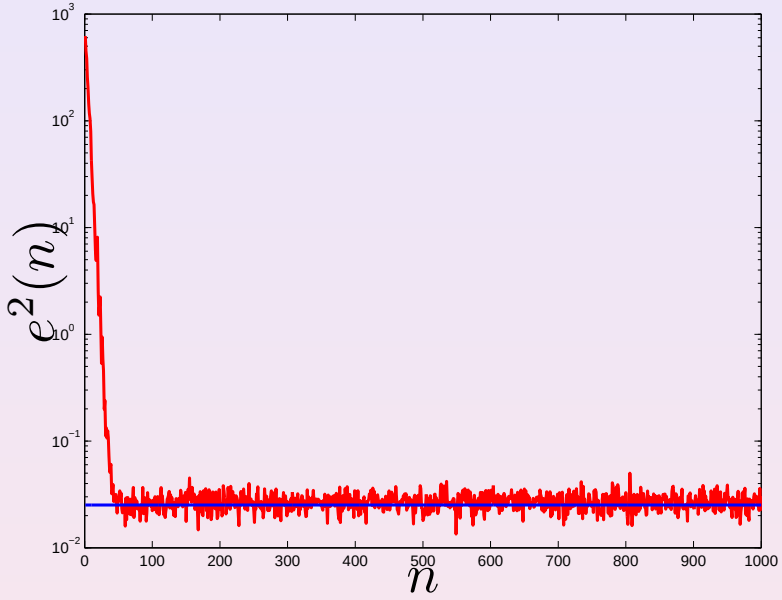
Misadjustment (desajuste): quanto a solução do LMS difere da solução ótima?

Tomando $\mathbf{w}(n) = \mathbf{w}_{\text{opt}}$, teremos

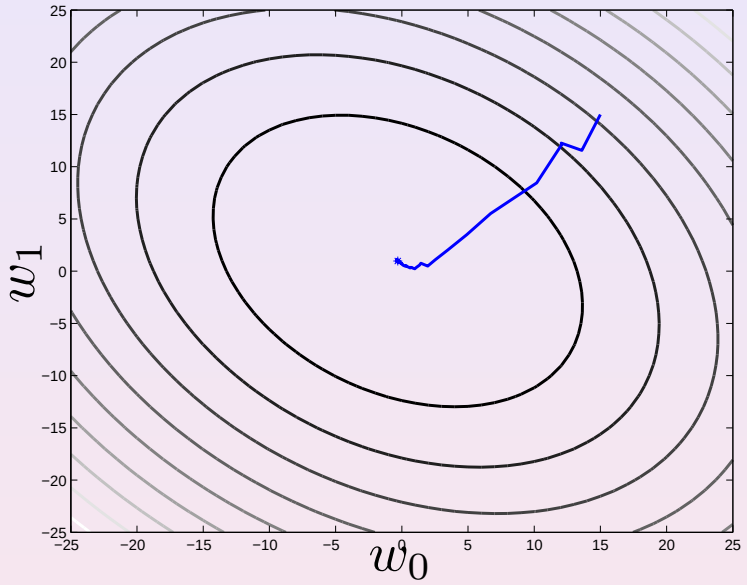
$$\begin{aligned}
\mathbf{w}(n+1) &= \mathbf{w}_{\text{opt}} + 2\mu\mathbf{x}(n)e(n) \\
&= \mathbf{w}_{\text{opt}} + 2\mu\mathbf{x}(n)[d(n) - \mathbf{x}^T(n)\mathbf{w}_{\text{opt}}] \\
&= \mathbf{w}_{\text{opt}} + 2\mu\mathbf{x}(n)[\mathbf{x}^T(n)\mathbf{w}_{\text{opt}} + b(n) - \mathbf{x}^T(n)\mathbf{w}_{\text{opt}}] \\
\mathbf{w}(n+1) - \mathbf{w}_{\text{opt}} &= 2\mu\mathbf{x}(n)b(n)
\end{aligned} \tag{131}$$

Desta forma, podemos ver que $\mathbb{E}\{\mathbf{w}(n+1) - \mathbf{w}_{\text{opt}}\} = 0$ mas que a variância não é zero devido ao termo $b(n)$.

- μ impacta na variância do erro de ajuste
- $\mathbf{w}(n+1)$ ao final da convergência fica "em torno" de $\mathbf{w}_{\text{opt}}$



Convergência do LMS (vermelho) para J_{min} (azul) usando $\mu = 0.1$



Trajetória do LMS (azul) para o ponto ótimo (solução de Wiener) usando $\mu = 0.1$

Resumo:

- Algoritmo com baixa complexidade
- Converge, em média, para o filtro ótimo
- Fator de passo influencia na velocidade de convergência
- Compromisso com o erro de desajuste

Motivação

- O algoritmo LMS apresenta o fator de passo dependente das características da correlação
- Para aumentar a velocidade de convergência, aumenta-se o fator de passo, mas o mesmo fornece um erro residual maior
- Idéia: colocar os dados para servirem de regulação ao desajuste

Sabendo que

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\mu e(n)\mathbf{x}(n) = \mathbf{w}(n) + \Delta\widetilde{\mathbf{w}}(n) \tag{132}$$

temos então que

$$e^2(n) = d^2(n) + \mathbf{w}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) - 2d(n)\mathbf{w}^T(n)\mathbf{x}(n) \tag{133}$$

Se usarmos uma troca de $\widetilde{\mathbf{w}}(n) = \mathbf{w}(n) + \Delta\widetilde{\mathbf{w}}(n)$, teremos então:

$$\begin{aligned}\widetilde{e}^2(n) = e^2(n) &+ 2\Delta\widetilde{\mathbf{w}}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{w}(n) \\ &+ \Delta\widetilde{\mathbf{w}}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\Delta\widetilde{\mathbf{w}}(n) - 2d(n)\Delta\widetilde{\mathbf{w}}^T(n)\mathbf{x}(n)\end{aligned} \tag{134}$$

Então, definindo

$$\begin{aligned}\Delta e^2(n) &= \widetilde{e}^2(n) - e^2(n) \\ &= -2\Delta\widetilde{\mathbf{w}}^T(n)\mathbf{x}(n)e(n) + \Delta\widetilde{\mathbf{w}}^T(n)\mathbf{x}(n)\mathbf{x}^T(n)\Delta\widetilde{\mathbf{w}}(n)\end{aligned} \tag{135}$$

Meta: tornar $\Delta e^2(n)$ negativo e mínimo pela escolha apropriada de μ

Substituindo $\Delta\widetilde{\mathbf{w}}(n) = 2\mu e(n)\mathbf{x}(n)$ na Eq. (135) tem-se

$$\Delta e^2(n) = -4\mu e^2(n)\mathbf{x}^T(n)\mathbf{x}(n) + 4\mu^2 e^2(n)[\mathbf{x}^T(n)\mathbf{x}(n)]^2 \tag{136}$$

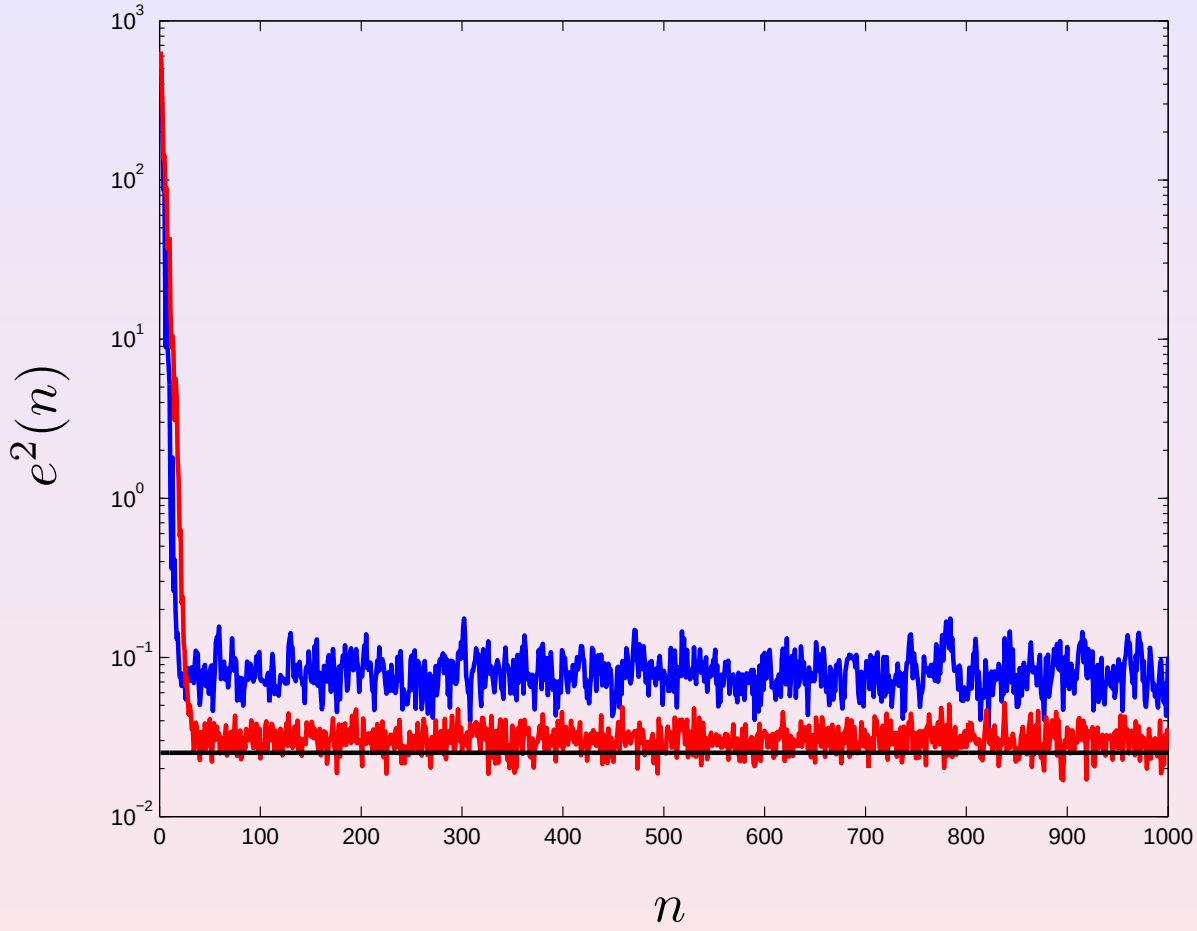
Valor de μ é dado por $\frac{\partial \Delta e^2(n)}{\partial \mu} = 0$, de onde tem-se

$$\mu = \frac{1}{2\mathbf{x}^T(n)\mathbf{x}(n)} \tag{137}$$

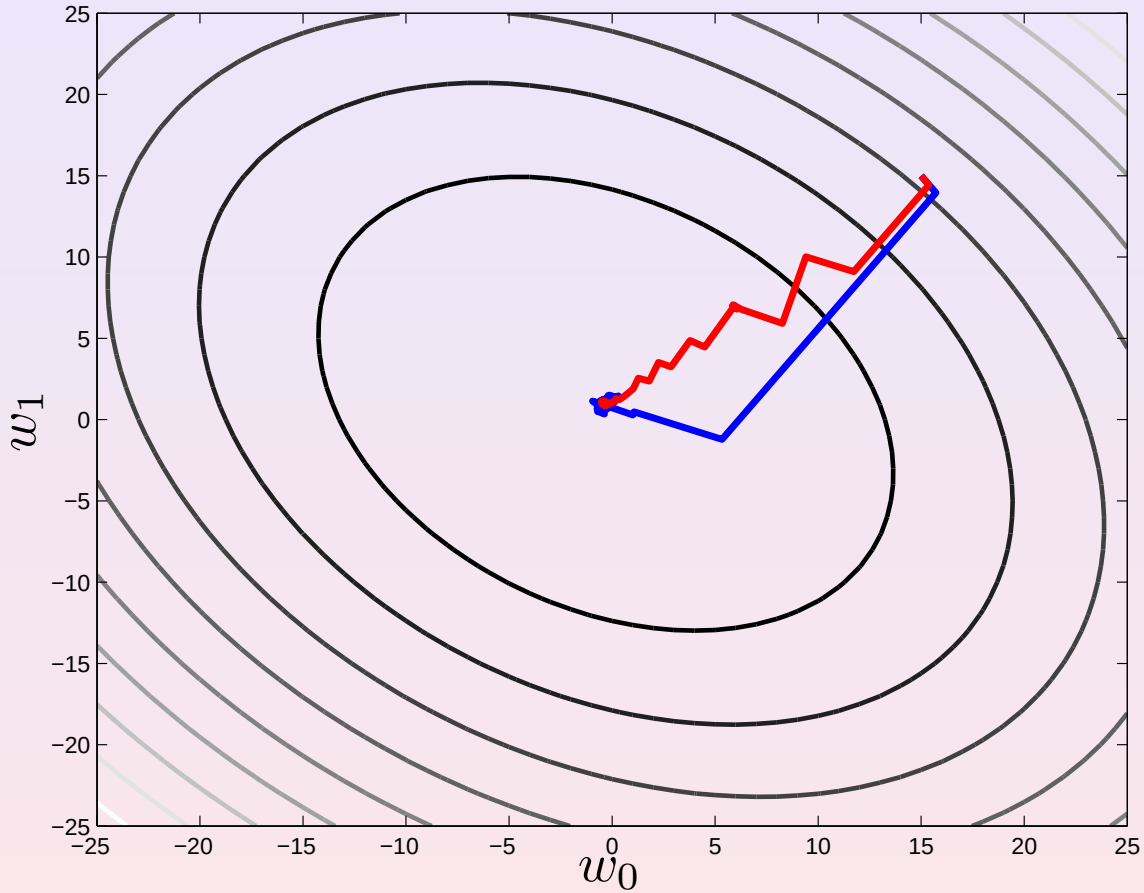
Com isso, o algoritmo do LMS normalizado é então dado por

Algoritmo LMS Normalizado

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \frac{\mu}{\gamma + \mathbf{x}^T(n)\mathbf{x}(n)}\mathbf{x}(n)e(n) \tag{138}$$



Algoritmos LMS (azul) e LMS-Normalizado (vermelho) com mesmo fator de passo $\mu_{\text{LMS}} = \mu_{\text{LMS-Norm}} = 0.5$, comparados com J_{min} (preto)



Trajetórias dos algoritmos LMS (azul) e LMS-Normalizado (vermelho) para o ponto ótimo