Processamento Adaptativo de Sinais

Otimização de Funções Objetivo

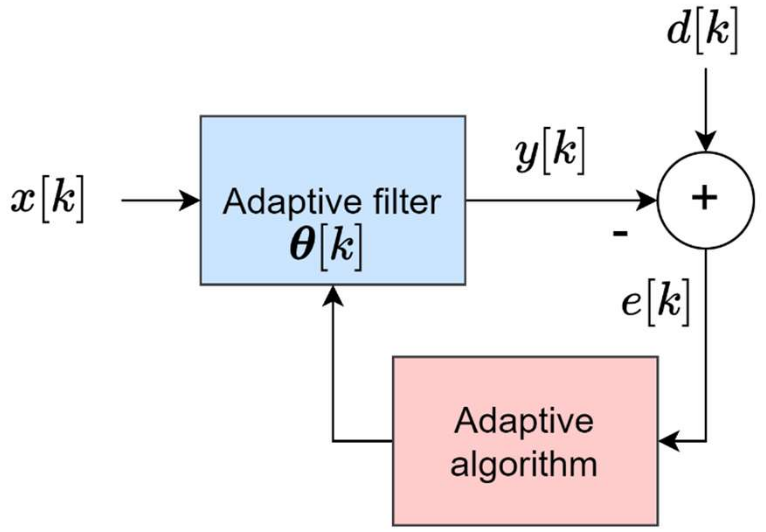
Edson P. da Silva

Programa de Pós-Graduação em Engenharia Elétrica (PPgEE).
Unidade Acadêmica de Engenharia Elétrica (UAEE)
Universidade Federal de Campina Grande (UFCG)

Sumário

Otimização de funções objetivo

- A função de algoritmo de filtragem adaptativa consiste em ajustar o conjunto de parâmetros $\boldsymbol{\theta}[k]$ associado a uma determinada estrutura de filtro de forma a minimizar uma função objetivo $J[k] = J(x[k], y[k], d[k], \boldsymbol{\theta}[k])$ de interesse.



Otimização de funções objetivo

Universidade Federal de Campina Grande

- A função de algoritmo de filtragem adaptativa consiste em ajustar o conjunto de parâmetros $\boldsymbol{\theta}[k]$ associado a uma determinada estrutura de filtro de forma a minimizar uma função objetivo $J[k] = J(x[k], y[k], d[k], \boldsymbol{\theta}[k])$ de interesse.
- Uma função objetivo J consistente deve possuir as seguintes propriedades:

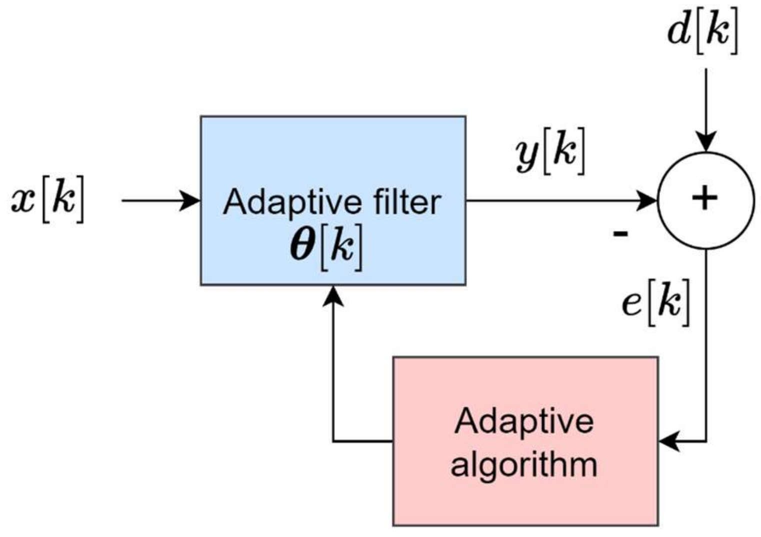
Não-negatividade:

$$J[k] = J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) \geq 0,$$

$$\forall\ y[k], x[k], d[k], \text{ e } \boldsymbol{\theta}[k].$$

Optimalidade:

$$\nabla_{\boldsymbol{\theta}} J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = 0.$$



Funções objetivo

Universidade Federal
de Campina Grande

Funções objetivo

- Erro Médio Quadrático (*Mean-Square Error* – MSE):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = E\left[|e[k]|^2\right] \approx \frac{1}{m}\sum_{i=1}^{m}(d[i] - y[i])^2$$

Funções objetivo

- Erro Médio Quadrático (*Mean-Square Error* – MSE):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = E\left[|e[k]|^2\right] \approx \frac{1}{m}\sum_{i=1}^{m}(d[i] - y[i])^2$$

- Mínimos Quadrados (*Least Squares* – LS):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = \frac{1}{k+1}\sum_{i=0}^{k}|e(k-i)|^2$$

Funções objetivo

Universidade Federal
de Campina Grande

- Erro Médio Quadrático (*Mean-Square Error* – MSE):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = E\left[|e[k]|^2\right] \approx \frac{1}{m}\sum_{i=1}^{m}(d[i] - y[i])^2$$

- Mínimos Quadrados (*Least Squares* – LS):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = \frac{1}{k+1}\sum_{i=0}^{k}|e(k-i)|^2$$

- Mínimos Quadrados Ponderados (*Weighted Least Squares* – WLS):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = \sum_{i=0}^{k}\lambda^i|e(k-i)|^2, \quad 0 < \lambda < 1$$

Funções objetivo

Universidade Federal de Campina Grande

- Erro Médio Quadrático (*Mean-Square Error* – MSE):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = E\left[|e[k]|^2\right] \approx \frac{1}{m}\sum_{i=1}^{m}\left(d[i] - y[i]\right)^2$$

- Mínimos Quadrados (*Least Squares* – LS):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = \frac{1}{k+1}\sum_{i=0}^{k}|e(k-i)|^2$$

- Mínimos Quadrados Ponderados (*Weighted Least Squares* – WLS):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = \sum_{i=0}^{k}\lambda^i|e(k-i)|^2, \quad 0 < \lambda < 1$$

- Erro Quadrático Instantâneo (*Instantaneous Square Value* – ISV):

$$J(x[k], y[k], d[k], \boldsymbol{\theta}[k]) = |e[k]|^2$$

Funções objetivo

Universidade Federal
de Campina Grande

- Observações:

Funções objetivo

- Observações:

1. MSE, no sentido estrito, tem apenas valor teórico, uma vez que requer uma quantidade infinita de informações para ser medido. Na prática, essa função objetivo ideal pode ser aproximada pelas outras três listadas.

Funções objetivo

Universidade Federal
de Campina Grande

- Observações:

1. MSE, no sentido estrito, tem apenas valor teórico, uma vez que requer uma quantidade infinita de informações para ser medido. Na prática, essa função objetivo ideal pode ser aproximada pelas outras três listadas.
2. As funções LS, WLS e ISV possuem tanto diferentes complexidades de implementação, como características de comportamento de convergência.

Funções objetivo

Universidade Federal
de Campina Grande

- Observações:

1. MSE, no sentido estrito, tem apenas valor teórico, uma vez que requer uma quantidade infinita de informações para ser medido. Na prática, essa função objetivo ideal pode ser aproximada pelas outras três listadas.
2. As funções LS, WLS e ISV possuem tanto diferentes complexidades de implementação, como características de comportamento de convergência.
3. A função ISV geralmente é a de mais fácil implementação, entretando apresenta propriedades de convergência ruidosas. A função LS é conveniente para ser usada em ambientes estacionários, enquanto o WLS é útil em aplicações onde o ambiente varia lentamente.

Método de Newton para otimização

Universidade Federal de Campina Grande

- Aproximação quadrática de uma função unidimensional via série de Taylor

$$f\left(x_k+\Delta x\right) \approx f\left(x_k\right)+f^{\prime}\left(x_k\right) \Delta x+\frac{1}{2} f^{\prime \prime}\left(x_k\right)(\Delta x)^2$$

- Determinação do ponto extremo (máximo ou mínimo) da aproximação de $f(x)$:

$$\frac{d}{d \Delta x} f\left(x_k+\Delta x\right)=0 \rightarrow 0=f^{\prime}\left(x_k\right)+f^{\prime \prime}\left(x_k\right) \Delta x$$

$$\Delta x=-\frac{f^{\prime}\left(x_k\right)}{f^{\prime \prime}\left(x_k\right)} \qquad x_{k+1}=x_k+\Delta x=x_k-\frac{f^{\prime}\left(x_k\right)}{f^{\prime \prime}\left(x_k\right)}$$

Método de Newton para otimização

Universidade Federal de Campina Grande

- A interpretação geométrica do método de Newton:
  1. A cada iteração, o gráfico de $f(x)$ é aproximado por uma parábola em x_k, tendo esta a mesma inclinação e curvatura que $f(x)$ nesse ponto;

Método de Newton para otimização

Universidade Federal de Campina Grande

- A interpretação geométrica do método de Newton:
  1. A cada iteração, o gráfico de $f(x)$ é aproximado por uma parábola em x_k, tendo esta a mesma inclinação e curvatura que $f(x)$ nesse ponto;
  2. O valor de x_{k+1} é calculado de tal forma que este seja a coordenada (ou esteja próximo) associada ao máximo ou mínimo dessa parábola.

$$x_{k+1} = x_k - \mu \frac{f'(x_k)}{f''(x_k)}$$

Obs: se $f(x)$ for uma função quadrática e $\mu = 1$, então o extremo exato é encontrado em um único passo.

Método de Newton para otimização

Universidade Federal de Campina Grande

- Exemplo: $\mu = 1$, $tol = 10^{-4}$

$$f(x) = 0.8x^4 - x^3 - 4x^2 + 10x + 25$$

$$\frac{df(x)}{dx} = 3.2x^3 - 3x^2 - 8x + 10$$

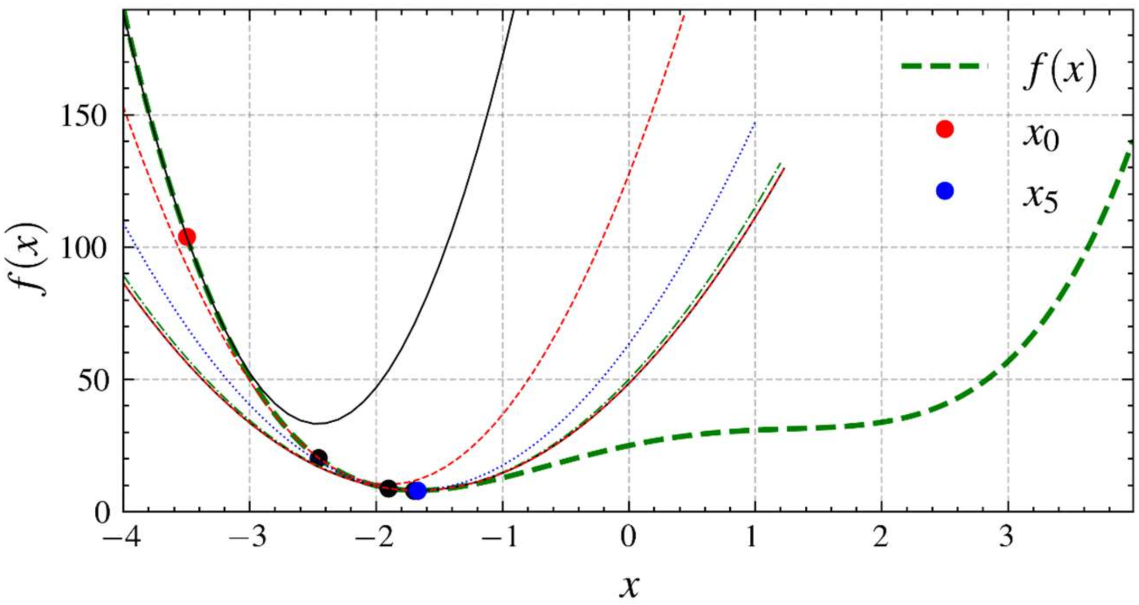
$$\frac{d^2 f(x)}{dx^2} = 9.6x^2 - 6x - 8$$

Método de Newton para otimização

Universidade Federal de Campina Grande

- Exemplo: $\mu = 1$, $tol = 10^{-4}$

$$f(x) = 0.8x^4 - x^3 - 4x^2 + 10x + 25$$
$$\frac{df(x)}{dx} = 3.2x^3 - 3x^2 - 8x + 10$$
$$\frac{d^2f(x)}{dx^2} = 9.6x^2 - 6x - 8$$



iteration 0 : $x_k = -3.500$

$f(-3.500 + u) = 65.3u^2 - 135.95u + 103.925$

iteration 1 : $x_k = -2.459$

$f(-2.459 + u) = 32.402u^2 - 36.0504u + 20.3432$

iteration 2 : $x_k = -1.903$

$f(-1.903 + u) = 19.0861u^2 - 7.6831u + 8.8656$

iteration 3 : $x_k = -1.701$

$f(-1.701 + u) = 15.0003u^2 - 0.8354u + 8.0359$

iteration 4 : $x_k = -1.674$

$f(-1.674 + u) = 14.4656u^2 - 0.0149u + 8.0241$

iteration 5 : $x_k = -1.673$

$f(-1.673 + u) = 14.4558u^2 + 8.0241$

Convergence achieved after 6 iterations.
Minimum value of $f(x) = f(-1.673) = 8.024$

Método de Newton para otimização

- **Trabalhando em múltiplas dimensões**: vetor gradiente e matriz Hessiana

$$\nabla_{\boldsymbol{\theta}} J = \begin{bmatrix} \frac{\partial J}{\partial \theta_1} \\ \frac{\partial J}{\partial \theta_2} \\ \vdots \\ \frac{\partial J}{\partial \theta_L} \end{bmatrix} \qquad \mathbf{H}_{\boldsymbol{\theta}} J = \begin{bmatrix} \frac{\partial^2 J}{\partial \theta_1^2} & \frac{\partial^2 J}{\partial \theta_1 \partial \theta_2} & \cdots & \frac{\partial^2 J}{\partial \theta_1 \partial \theta_n} \\ \frac{\partial^2 J}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 J}{\partial \theta_2^2} & \cdots & \frac{\partial^2 J}{\partial \theta_2 \partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 J}{\partial \theta_n \partial \theta_1} & \frac{\partial^2 J}{\partial \theta_n \partial \theta_2} & \cdots & \frac{\partial^2 J}{\partial \theta_n^2} \end{bmatrix}$$

Método de Newton para otimização

Universidade Federal
de Campina Grande

- Seja a função objetivo J uma função contínua em $\boldsymbol{\theta}$, esta pode ser aproximada na vizinhança de um determinado ponto $\boldsymbol{\theta}[k]$, por meio de uma série de Taylor truncada na forma

$$J(\boldsymbol{\theta}[k] + \Delta\boldsymbol{\theta}) \approx J(\boldsymbol{\theta}[k]) + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])^T \Delta\boldsymbol{\theta} + \frac{1}{2}\Delta\boldsymbol{\theta}^T \mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

Método de Newton para otimização

Universidade Federal de Campina Grande

- Algumas operações do cálculo de matrizes:

$$\frac{\partial y}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix}$$

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$$
$$\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$$

Método de Newton para otimização

Universidade Federal de Campina Grande

- Algumas operações do cálculo de matrizes:

$$\frac{\partial y}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix} \quad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix}$$

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$$

$$\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$$

Método de Newton para otimização

- Algumas operações do cálculo de matrizes:

$$\frac{\partial y}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix} \quad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \quad \frac{d\mathbf{y}}{dx} \triangleq \begin{bmatrix} \frac{dy_1}{dx} \\ \frac{dy_2}{dx} \\ \vdots \\ \frac{dy_n}{dx} \end{bmatrix}^T$$

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$$

$$\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$$

Método de Newton para otimização

Universidade Federal de Campina Grande

- Algumas operações do cálculo de matrizes:

$$\frac{\partial y}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y}{\partial x_1} \\ \frac{\partial y}{\partial x_2} \\ \vdots \\ \frac{\partial y}{\partial x_n} \end{bmatrix} \qquad \frac{\partial \mathbf{y}}{\partial \mathbf{x}} \triangleq \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_2}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_1} \\ \frac{\partial y_1}{\partial x_2} & \frac{\partial y_2}{\partial x_2} & \cdots & \frac{\partial y_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_n} & \frac{\partial y_2}{\partial x_n} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \qquad \frac{d\mathbf{y}}{dx} \triangleq \begin{bmatrix} \frac{dy_1}{dx} \\ \frac{dy_2}{dx} \\ \vdots \\ \frac{dy_n}{dx} \end{bmatrix}^T$$

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$$

$$\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$$

Derivada escalar			Derivada vetorial		
$f(x)$	$\rightarrow$	$\frac{\mathrm{d}f}{\mathrm{d}x}$	$f(\mathbf{x})$	$\rightarrow$	$\frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}$
ax	$\rightarrow$	a	$\mathbf{Bx}$	$\rightarrow$	$\mathbf{B}^T$
bx	$\rightarrow$	b	$\mathbf{x}^T\mathbf{B}$	$\rightarrow$	$\mathbf{B}$
bx	$\rightarrow$	b	$\mathbf{x}^T\mathbf{b}$	$\rightarrow$	$\mathbf{b}$
x^2	$\rightarrow$	$2x$	$\mathbf{x}^T\mathbf{x}$	$\rightarrow$	$2\mathbf{x}$
bx^2	$\rightarrow$	$2bx$	$\mathbf{x}^T\mathbf{Bx}$	$\rightarrow$	$\mathbf{Bx} + \mathbf{B}^T\mathbf{x}$
			$\mathbf{x}^T\mathbf{Bx}$	$\rightarrow$	$2\mathbf{Bx}$, se $\mathbf{B}$ for simétrica.

Método de Newton para otimização

Universidade Federal
de Campina Grande

- Seja a função objetivo J uma função contínua em $\boldsymbol{\theta}$, esta pode ser aproximada na vizinhança de um determinado ponto $\boldsymbol{\theta}[k]$, por meio de uma série de Taylor truncada na forma

$$J(\boldsymbol{\theta}[k] + \Delta\boldsymbol{\theta}) \approx J(\boldsymbol{\theta}[k]) + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])^T \Delta\boldsymbol{\theta} + \frac{1}{2}\Delta\boldsymbol{\theta}^T[k]\mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

Método de Newton para otimização

Universidade Federal de Campina Grande

- Seja a função objetivo J uma função contínua em $\boldsymbol{\theta}$, esta pode ser aproximada na vizinhança de um determinado ponto $\boldsymbol{\theta}[k]$, por meio de uma série de Taylor truncada na forma

$$J(\boldsymbol{\theta}[k]+\Delta\boldsymbol{\theta}) \approx J(\boldsymbol{\theta}[k]) + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])^T \Delta\boldsymbol{\theta} + \frac{1}{2}\Delta\boldsymbol{\theta}^T[k]\mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

$$\frac{d}{d\Delta\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]+\Delta\boldsymbol{\theta}[k]) = 0 \rightarrow 0 = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) + \mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

Método de Newton para otimização

Universidade Federal de Campina Grande

• Seja a função objetivo J uma função contínua em $\boldsymbol{\theta}$, esta pode ser aproximada na vizinhança de um determinado ponto $\boldsymbol{\theta}[k]$, por meio de uma série de Taylor truncada na forma

$$J(\boldsymbol{\theta}[k] + \Delta\boldsymbol{\theta}) \approx J(\boldsymbol{\theta}[k]) + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])^T \Delta\boldsymbol{\theta} + \frac{1}{2}\Delta\boldsymbol{\theta}^T[k]\mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

$$\frac{d}{d\Delta\boldsymbol{\theta}} J(\boldsymbol{\theta}[k] + \Delta\boldsymbol{\theta}[k]) = 0 \to 0 = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) + \mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) = -\mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

Método de Newton para otimização

- Seja a função objetivo J uma função contínua em $\boldsymbol{\theta}$, esta pode ser aproximada na vizinhança de um determinado ponto $\boldsymbol{\theta}[k]$, por meio de uma série de Taylor truncada na forma

$$J(\boldsymbol{\theta}[k] + \Delta\boldsymbol{\theta}) \approx J(\boldsymbol{\theta}[k]) + \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])^T \Delta\boldsymbol{\theta} + \frac{1}{2}\Delta\boldsymbol{\theta}^T[k]\mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

$$\frac{d}{d\Delta\boldsymbol{\theta}} J(\boldsymbol{\theta}[k] + \Delta\boldsymbol{\theta}[k]) = 0 \rightarrow 0 = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) + \mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta}$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) = -\mathbf{H}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])\Delta\boldsymbol{\theta} \qquad \Delta\boldsymbol{\theta} = -\mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k])\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$$

Método de Newton para otimização

- Atualização do vetor de parâmetros $\boldsymbol{\theta}[k]$, segundo o método de Newton:

$$\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k]) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$$

- Em que μ é um parâmetro que controla a tamanho do passo de adaptação do algoritmo (*step size*), ou seja, quão rápido os valores do vetor de parâmetros serão alterados entre iterações.

Método de Newton para otimização

Universidade Federal de Campina Grande

- O método de Newton, em sua versão original, apresenta várias ressalvas:

Método de Newton para otimização

Universidade Federal de Campina Grande

- O método de Newton, em sua versão original, apresenta várias ressalvas:

1. **Gradiente e Hessiana:** o método depende do cálculo do vetor gradiente e da inversa da matriz Hessiana a cada iteração, o que geralmente implica em um alto esforço computacional.

Método de Newton para otimização

Universidade Federal de Campina Grande

- O método de Newton, em sua versão original, apresenta várias ressalvas:

1. **Gradiente e Hessiana:** o método depende do cálculo do vetor gradiente e da inversa da matriz Hessiana a cada iteração, o que geralmente implica em um alto esforço computacional.
2. **Matriz Hessiana**: o método não funciona se a Hessiana não for invertível. Isso é evidente pela própria definição do método de Newton, que exige a inversão da Hessiana.

Método de Newton para otimização

- O método de Newton, em sua versão original, apresenta várias ressalvas:

1. **Gradiente e Hessiana:** o método depende do cálculo do vetor gradiente e da inversa da matriz Hessiana a cada iteração, o que geralmente implica em um alto esforço computacional.
2. **Matriz Hessiana**: o método não funciona se a Hessiana não for invertível. Isso é evidente pela própria definição do método de Newton, que exige a inversão da Hessiana.
3. **Convergência**: o método pode não convergir, entrando num comportamento cíclico envolvendo mais de um ponto extremo. O método pode convergir para um ponto de sela em vez de para um mínimo local.

Método iterativos para otimização

Método iterativos para otimização

- **Método de Newton**: $\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k]) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$

Método iterativos para otimização

Universidade Federal
de Campina Grande

- **Método de Newton**: $\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k]) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$
- **Métodos quase-Newton**: objetivam minimizar a função objetivo utilizando uma estimativa calculada recursivamente do inverso da matriz Hessiana.

$$\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \mathbf{S}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$$

$$\lim_{k \to \infty} \mathbf{S}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) = \mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k])]$$

Método iterativos para otimização

Universidade Federal
de Campina Grande

- **Método de Newton**: $\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k]) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$
- **Métodos quase-Newton**: objetivam minimizar a função objetivo utilizando uma estimativa calculada recursivamente do inverso da matriz Hessiana.

$$\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \mathbf{S}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$$

$$\lim_{k \to \infty} \mathbf{S}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k]) = \mathbf{H}_{\boldsymbol{\theta}}^{-1} J(\boldsymbol{\theta}[k])]$$

- **Método do gradiente descendente**: busca o ponto de mínimo da função objetivo seguindo a direção oposta ao vetor gradiente dessa função.

$$\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \mu \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}[k])$$