شبکه‌های عصبی مصنوعی

درس ۹

بهینه‌سازی کارآیی

Performance Optimization

کاظم فولادی قلعه
دانشکده مهندسی، پردیس فارابی
دانشگاه تهران

http://courses.fouladi.ir/nn

Performance Optimization

بهینه‌سازی کارآیی

PERFORMANCE OPTIMIZATION

هدف: می‌نیمم‌سازی شاخص کارآیی $F(\mathbf{x})$

الگوریتم تکراری: شروع از حدس اولیه‌ی $\mathbf{x}_0$ و اجرای قاعده‌ی تکرار

الگوریتم‌های اصلی	
	○ تندترین شیب (steepest descent)
	○ روش نیوتن (Newton's method)
	○ گرادیان مزدوج (conjugate gradient)

بهینه‌سازی کارآیی

۱

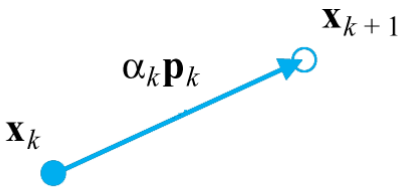
الگوریتم پایه‌ی بهینه‌سازی: تندترین شیب

Basic Optimization Algorithm

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

or

$$\Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$$



$\mathbf{p}_k$ - Search Direction

α_k - Learning Rate

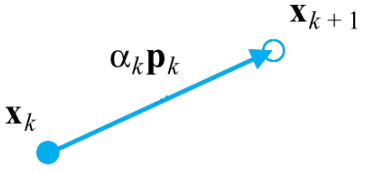
الگوریتم پایه‌ی بهینه‌سازی

BASIC OPTIMIZATION ALGORITHM

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

or

$$\Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$$



$\mathbf{p}_k$ - Search Direction — جهت جستجو

α_k - Learning Rate — نرخ یادگیری (طول گام)

تفاوت الگوریتم‌ها در انتخاب $\mathbf{p}_k$ است.

Steepest Descent

Choose the next step so that the function decreases:

$$F(\mathbf{x}_{k+1}) < F(\mathbf{x}_k)$$

For small changes in $\mathbf{x}$ we can approximate $F(\mathbf{x})$:

$$F(\mathbf{x}_{k+1}) = F(\mathbf{x}_k + \Delta\mathbf{x}_k) \approx F(\mathbf{x}_k) + \mathbf{g}_k^T \Delta\mathbf{x}_k$$

where

$$\mathbf{g}_k \equiv \nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k}$$

If we want the function to decrease:

$$\mathbf{g}_k^T \Delta\mathbf{x}_k = \alpha_k \mathbf{g}_k^T \mathbf{p}_k < 0$$

We can maximize the decrease by choosing:

$$\mathbf{p}_k = -\mathbf{g}_k$$

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k}$$

روش تندترین شیب

STEEPEST DESCENT

Choose the next step so that the function decreases:

$$F(\mathbf{x}_{k+1}) < F(\mathbf{x}_k)$$

For small changes in $\mathbf{x}$ we can approximate $F(\mathbf{x})$:

$$F(\mathbf{x}_{k+1}) = F(\mathbf{x}_k + \Delta\mathbf{x}_k) \approx F(\mathbf{x}_k) + \mathbf{g}_k^T \Delta\mathbf{x}_k$$

where

$$\mathbf{g}_k \equiv \nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k}$$

If we want the function to decrease: $\Delta\mathbf{x}_k = \alpha_k \mathbf{p}_k$

$$\mathbf{g}_k^T \Delta\mathbf{x}_k = \alpha_k \mathbf{g}_k^T \mathbf{p}_k < 0$$

جهت نزول (descent direction):
هر بردار $\mathbf{p}_k$ که $\mathbf{g}_k^T \mathbf{p}_k < 0$ باشد.

We can maximize the decrease by choosing:

$$\mathbf{p}_k = -\mathbf{g}_k$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k$$

$\mathbf{g}_k^T \mathbf{p}_k$: ضرب داخلی بردار جهت و گرادیان:
بیشترین مقدار زمانی است که
زاویه‌ی بین دو بردار صفر باشد.

بیشترین نزول در چه جهتی است؟
جهتی که تابع سریع‌ترین کاهش را دارد:
وقتی که $\mathbf{g}_k^T \mathbf{p}_k$ منفی‌ترین مقدار ممکن باشد.

Example

$$F(\mathbf{x}) = x_1^2 + 2x_1x_2 + 2x_2^2 + x_1$$

$$\mathbf{x}_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \qquad \alpha = 0.1$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} F(\mathbf{x}) \\ \frac{\partial}{\partial x_2} F(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 2x_1 + 2x_2 + 1 \\ 2x_1 + 4x_2 \end{bmatrix} \qquad \mathbf{g}_0 = \nabla F(\mathbf{x})\Big|_{\mathbf{x} = \mathbf{x}_0} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{x}_1 = \mathbf{x}_0 - \alpha \mathbf{g}_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - 0.1 \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.2 \\ 0.2 \end{bmatrix}$$

$$\mathbf{x}_2 = \mathbf{x}_1 - \alpha \mathbf{g}_1 = \begin{bmatrix} 0.2 \\ 0.2 \end{bmatrix} - 0.1 \begin{bmatrix} 1.8 \\ 1.2 \end{bmatrix} = \begin{bmatrix} 0.02 \\ 0.08 \end{bmatrix}$$

روش تندترین شیب

مثال

STEEPEST DESCENT

$$F(\mathbf{x}) = x_1^2 + 2x_1x_2 + 2x_2^2 + x_1$$

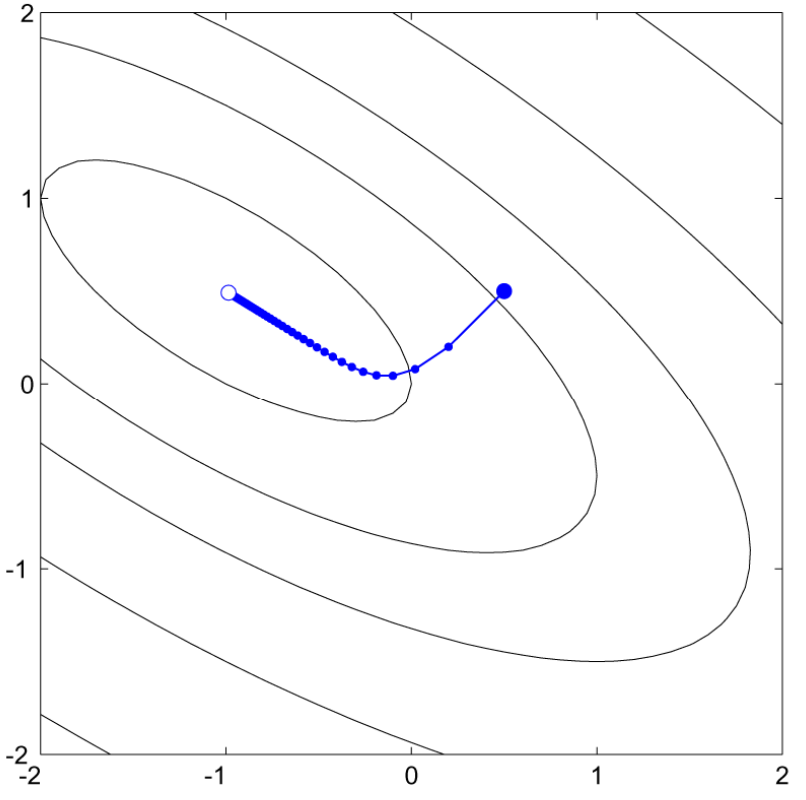
$$\mathbf{x}_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \qquad \alpha = 0.1$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} F(\mathbf{x}) \\ \frac{\partial}{\partial x_2} F(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 2x_1 + 2x_2 + 1 \\ 2x_1 + 4x_2 \end{bmatrix} \qquad \mathbf{g}_0 = \nabla F(\mathbf{x})\Big|_{\mathbf{x} = \mathbf{x}_0} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{x}_1 = \mathbf{x}_0 - \alpha \mathbf{g}_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - 0.1 \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.2 \\ 0.2 \end{bmatrix}$$

$$\mathbf{x}_2 = \mathbf{x}_1 - \alpha \mathbf{g}_1 = \begin{bmatrix} 0.2 \\ 0.2 \end{bmatrix} - 0.1 \begin{bmatrix} 1.8 \\ 1.2 \end{bmatrix} = \begin{bmatrix} 0.02 \\ 0.08 \end{bmatrix}$$

Plot



روش تندترین شیب

مثال: رسم نمودار

STEEPEST DESCENT

نرخ یادگیری کوچک

⇓

تراجکتوری بیشترین نزول شیب،
برای نرخ یادگیری کوچک، مسیری را می‌پیماید که همیشه بر خطوط کانتور عمود است.
(زیرا بردار گرادیان بر خطوط کانتور عمود است)

نرخ یادگیری بزرگ

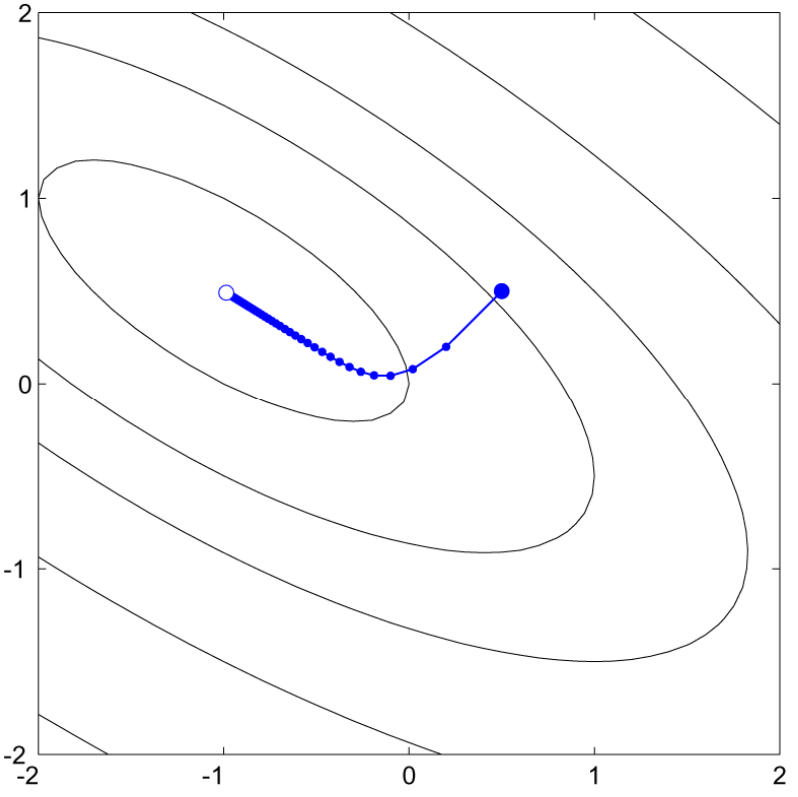
⇓

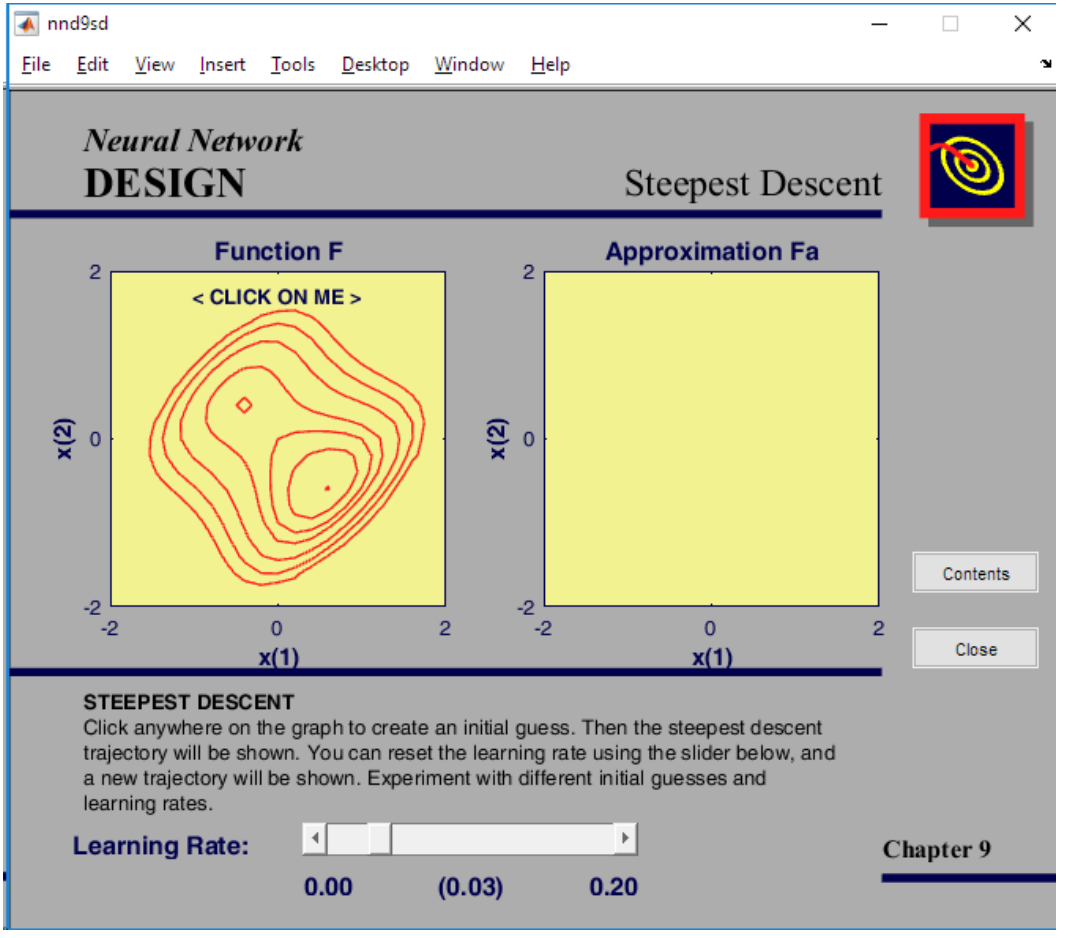
نوسان‌های میرا

نرخ یادگیری خیلی بزرگ

⇓

نوسان‌های فزاینده ⇐ ناپایداری





MATLAB

```
>> nnd9sd
```

Stable Learning Rates (Quadratic)

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{x} + c$$

$$\nabla F(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{d}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha\mathbf{g}_k = \mathbf{x}_k - \alpha(\mathbf{A}\mathbf{x}_k + \mathbf{d}) \Rightarrow \mathbf{x}_{k+1} = [\mathbf{I} - \alpha\mathbf{A}]\mathbf{x}_k - \alpha\mathbf{d}$$

Stability is determined by the eigenvalues of this matrix.

$$[\mathbf{I} - \alpha\mathbf{A}]\mathbf{z}_i = \mathbf{z}_i - \alpha\mathbf{A}\mathbf{z}_i = \mathbf{z}_i - \alpha\lambda_i\mathbf{z}_i = (1 - \alpha\lambda_i)\mathbf{z}_i$$

(λ_i - eigenvalue of $\mathbf{A}$)

Eigenvalues of $[\mathbf{I} - \alpha\mathbf{A}]$.

Stability Requirement:

$$\left|(1 - \alpha\lambda_i)\right| < 1 \qquad \alpha < \frac{2}{\lambda_i}$$

$$\alpha < \frac{2}{\lambda_{max}}$$

کران بالا برای نرخ یادگیری پایدار

در حالت درجه دوم

STABLE LEARNING RATES (QUADRATIC)

$\mathbf{A}$: ماتریس هسی

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{x} + c$$

$$\nabla F(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{d}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha\mathbf{g}_k = \mathbf{x}_k - \alpha(\mathbf{A}\mathbf{x}_k + \mathbf{d}) \Rightarrow \mathbf{x}_{k+1} = \underbrace{[\mathbf{I} - \alpha\mathbf{A}]}\mathbf{x}_k - \alpha\mathbf{d}$$

Stability is determined by the eigenvalues of this matrix.

به‌عنوان یک سیستم دینامیکی خطی، شرط پایداری این است که قدر مطلق مقادیر ویژه‌ی ماتریس $[\mathbf{I} - \alpha\mathbf{A}]$ کوچکتر از 1 باشد.

$$[\mathbf{I} - \alpha\mathbf{A}]\mathbf{z}_i = \mathbf{z}_i - \alpha\mathbf{A}\mathbf{z}_i = \mathbf{z}_i - \alpha\lambda_i\mathbf{z}_i = \underbrace{(1 - \alpha\lambda_i)}\mathbf{z}_i \qquad \boxed{\Delta\mathbf{x}_k = \alpha_k\mathbf{p}_k}$$

(λ_i - eigenvalue of $\mathbf{A}$)

Eigenvalues of $[\mathbf{I} - \alpha\mathbf{A}]$.

* بردارهای ویژه‌ی ماتریس $\mathbf{A}$ و ماتریس $[\mathbf{I} - \alpha\mathbf{A}]$ یکسان هستند.
* مقادیر ویژه‌ی ماتریس $[\mathbf{I} - \alpha\mathbf{A}]$ به‌صورت $(1 - \alpha\lambda_i)$ است.

Stability Requirement:

شرط پایداری: $|(1 - \alpha\lambda_i)| < 1 \Rightarrow \alpha < \frac{2}{\lambda_i} \Rightarrow \boxed{\alpha < \frac{2}{\lambda_{max}}}$

ماکزیمم نرخ یادگیری پایدار با ماکزیمم انحنای تابع درجه دوم نسبت معکوس دارد.

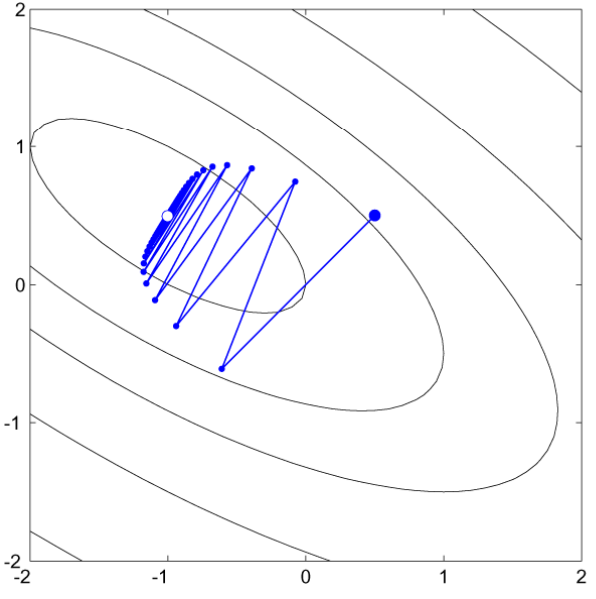
(انحنا: معیاری از سرعت تغییر گرادیان)

Example

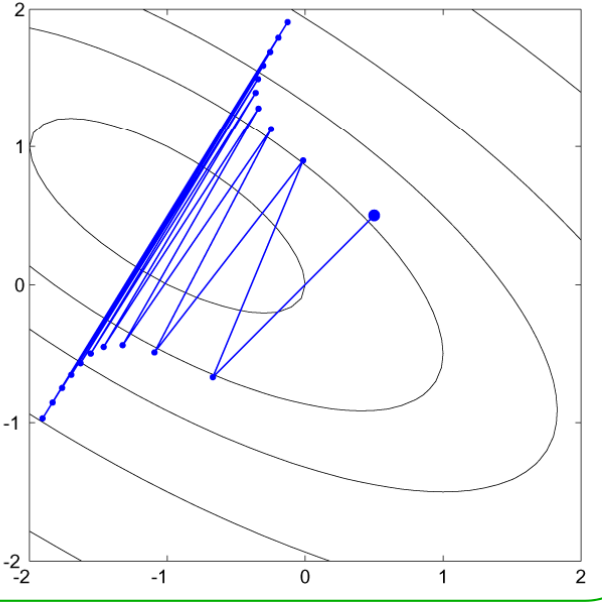
$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix} \qquad \left\{(\lambda_1 = 0.764), \left(\mathbf{z}_1 = \begin{bmatrix} 0.851 \\ -0.526 \end{bmatrix}\right)\right\}, \left\{\lambda_2 = 5.24, \left(\mathbf{z}_2 = \begin{bmatrix} 0.526 \\ 0.851 \end{bmatrix}\right)\right\}$$

$$\alpha < \frac{2}{\lambda_{max}} = \frac{2}{5.24} = 0.38$$

$\alpha = 0.37$



$\alpha = 0.39$

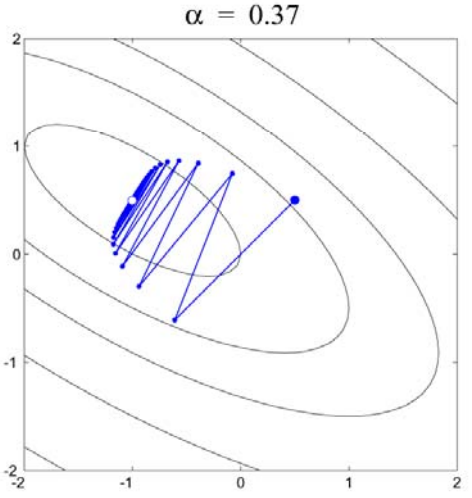


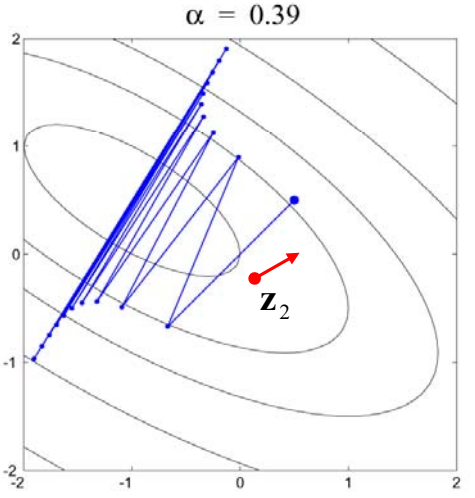
کران بالا برای نرخ یادگیری پایدار

در حالت درجه دوم: مثال

$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix} \qquad \left\{(\lambda_1 = 0.764), \left(\mathbf{z}_1 = \begin{bmatrix} 0.851 \\ -0.526 \end{bmatrix}\right)\right\}, \left\{\lambda_2 = 5.24, \left(\mathbf{z}_2 = \begin{bmatrix} 0.526 \\ 0.851 \end{bmatrix}\right)\right\}$$

$$\alpha < \frac{2}{\lambda_{max}} = \frac{2}{5.24} = 0.38$$





الگوریتم در جهت بردار ویژه‌ی متناظر با بزرگ‌ترین مقدار ویژه، سریع‌تر همگرا می‌شود.

الگوریتم در جهت بردار ویژه‌ی متناظر با کوچک‌ترین مقدار ویژه، آهسته‌تر همگرا می‌شود.

* کوچک‌ترین مقدار ویژه در ترکیب با نرخ یادگیری تعیین می‌کند که الگوریتم چه‌قدر سریع همگرا خواهد شد.

Minimizing Along a Line

Choose α_k to minimize $F(\mathbf{x}_k + \alpha_k \mathbf{p}_k)$

$$\frac{d}{d\alpha_k}(F(\mathbf{x}_k + \alpha_k \mathbf{p}_k)) = \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k + \alpha_k \mathbf{p}_k^T \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k$$

$$\alpha_k = -\frac{\nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k}{\mathbf{p}_k^T \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k} = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A}_k \mathbf{p}_k}$$

where

$$\mathbf{A}_k \equiv \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k}$$

تعیین نرخ یادگیری در هر گام

می‌نیمم‌سازی در راستای یک خط

MINIMIZING ALONG A LINE

در حالت کلی نیاز به جستجوی خط (line search) داریم:

$$\text{Choose } \alpha_k \text{ to minimize } \quad F(\mathbf{x}_k + \alpha_k \mathbf{p}_k)$$

$$F(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \approx F(\mathbf{x}_k) + \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k} \cdot \alpha_k \mathbf{p}_k + \frac{1}{2}(\alpha_k \mathbf{p}_k)^T \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k} \cdot (\alpha_k \mathbf{p}_k)$$

$$\frac{d}{d\alpha_k}(F(\mathbf{x}_k + \alpha_k \mathbf{p}_k)) = \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k + \alpha_k \mathbf{p}_k^T \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k$$

$$= 0 \Rightarrow$$

$$\alpha_k = -\frac{\nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k}{\mathbf{p}_k^T \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k} \mathbf{p}_k} = -\frac{\mathbf{g}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{A}_k \mathbf{p}_k}$$

where

$$\mathbf{A}_k \equiv \nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k} \qquad \mathbf{g}_k = \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k}$$

* در تابع درجه دوم، ماتریس هسی تابع k (شماره‌ی گام) نیست.

Example

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\mathbf{x} + \begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x} \qquad \mathbf{x}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix}$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix}\frac{\partial}{\partial x_1}F(\mathbf{x})\\ \frac{\partial}{\partial x_2}F(\mathbf{x})\end{bmatrix} = \begin{bmatrix}2x_1 + 2x_2 + 1\\ 2x_1 + 4x_2\end{bmatrix} \qquad \mathbf{p}_0 = -\mathbf{g}_0 = -\nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix}-3\\ -3\end{bmatrix}$$

$$\alpha_0 = -\frac{\begin{bmatrix}3 & 3\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}}{\begin{bmatrix}-3 & -3\end{bmatrix}\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}} = 0.2 \qquad \mathbf{x}_1 = \mathbf{x}_0 - \alpha_0\mathbf{g}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix} - 0.2\begin{bmatrix}3\\ 3\end{bmatrix} = \begin{bmatrix}-0.1\\ -0.1\end{bmatrix}$$

تعیین نرخ یادگیری در هر گام

می‌نیمم‌سازی در راستای یک خط: مثال

MINIMIZING ALONG A LINE

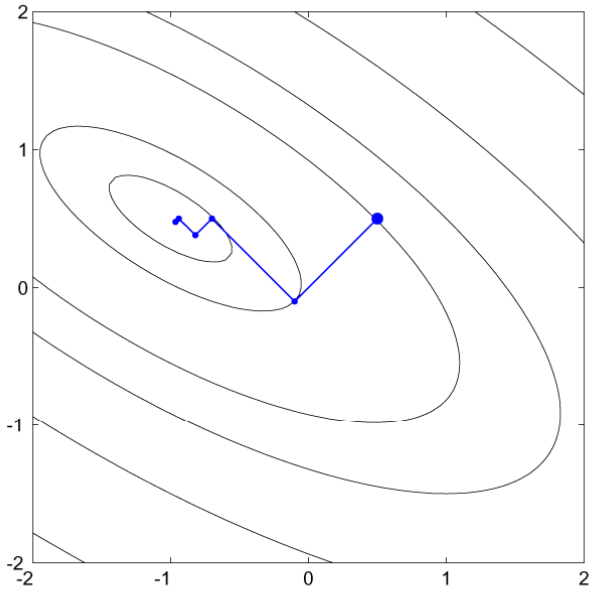
$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\mathbf{x} + \begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x} \qquad \mathbf{x}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix}$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix}\frac{\partial}{\partial x_1}F(\mathbf{x})\\ \frac{\partial}{\partial x_2}F(\mathbf{x})\end{bmatrix} = \begin{bmatrix}2x_1 + 2x_2 + 1\\ 2x_1 + 4x_2\end{bmatrix} \qquad \mathbf{p}_0 = -\mathbf{g}_0 = -\nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix}-3\\ -3\end{bmatrix}$$

جهت جستجو برای تندترین شیب نزولی، منفی گرادیان است.

$$\alpha_0 = -\frac{\begin{bmatrix}3 & 3\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}}{\begin{bmatrix}-3 & -3\end{bmatrix}\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}} = 0.2 \qquad \mathbf{x}_1 = \mathbf{x}_0 - \alpha_0\mathbf{g}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix} - 0.2\begin{bmatrix}3\\ 3\end{bmatrix} = \begin{bmatrix}-0.1\\ -0.1\end{bmatrix}$$

Plot



Successive steps are orthogonal.

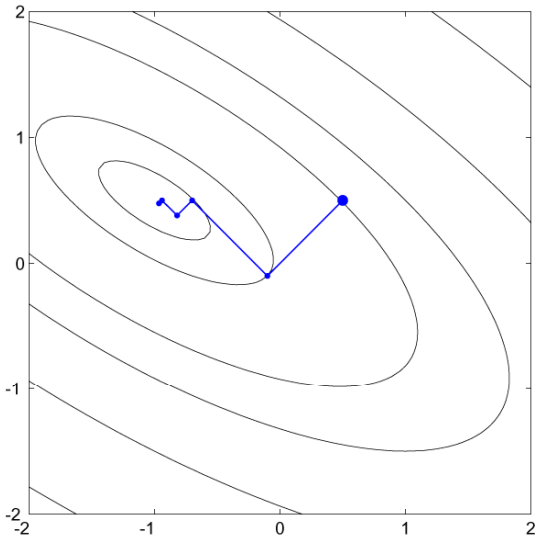
$$\frac{d}{d\alpha_k}F(\mathbf{x}_k + \alpha_k \mathbf{p}_k) = \frac{d}{d\alpha_k}F(\mathbf{x}_{k+1}) = \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_{k+1}} \frac{d}{d\alpha_k}[\mathbf{x}_k + \alpha_k \mathbf{p}_k]$$

$$= \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_{k+1}} \mathbf{p}_k = \mathbf{g}_{k+1}^T \mathbf{p}_k$$

تعیین نرخ یادگیری در هر گام

می‌نیمم‌سازی در راستای یک خط: مثال: رسم نمودار

MINIMIZING ALONG A LINE



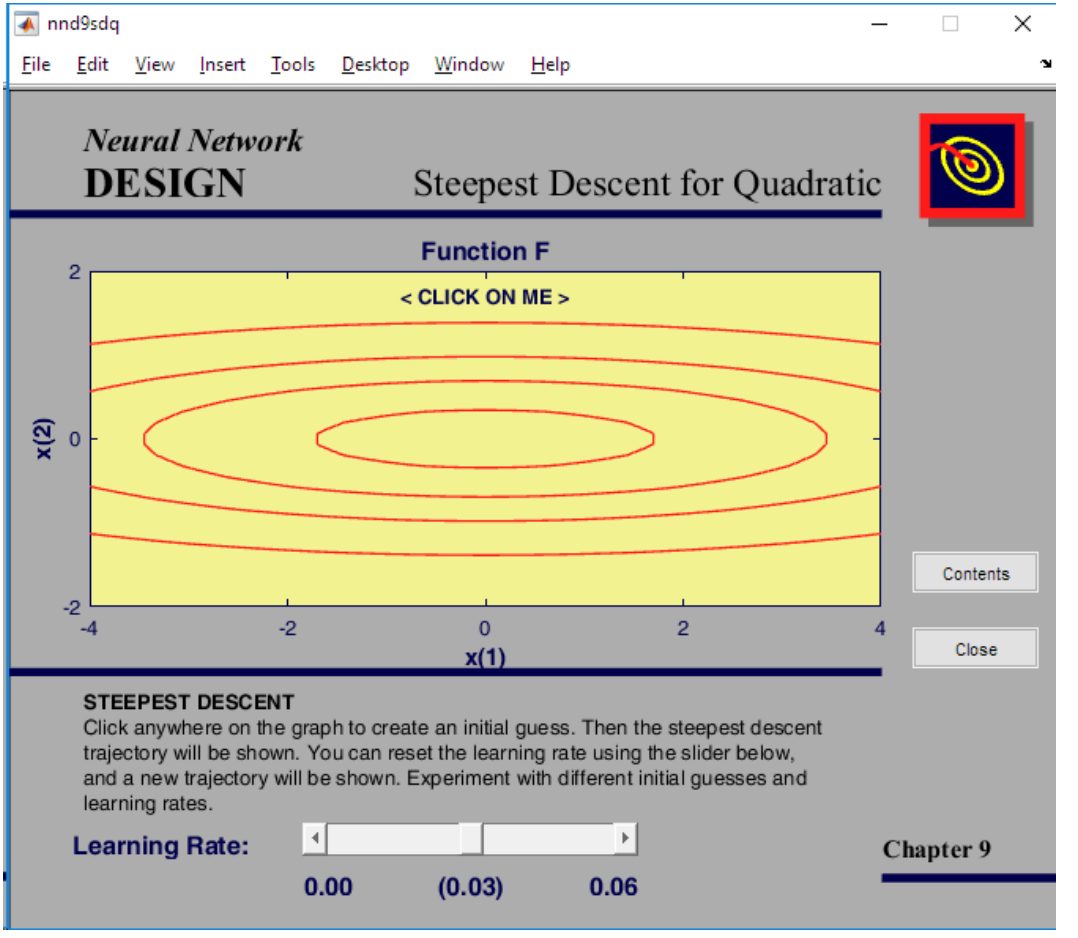
گام‌های متوالی بر هم عمود هستند.

Successive steps are orthogonal.

با استفاده از قاعده‌ی زنجیری (برای تابع درجه دوم):

$$\frac{d}{d\alpha_k}F(\mathbf{x}_k+\alpha_k\mathbf{p}_k) = \frac{d}{d\alpha_k}F(\mathbf{x}_{k+1}) = \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_{k+1}}\frac{d}{d\alpha_k}[\mathbf{x}_k+\alpha_k\mathbf{p}_k]$$

$$= \nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_{k+1}}\mathbf{p}_k = \mathbf{g}_{k+1}^T\mathbf{p}_k \qquad = 0 \Rightarrow \mathbf{g}_{k+1}\perp\mathbf{p}_k$$


nnd9sdq
File Edit View Insert Tools Desktop Window Help
Neural Network
DESIGN
Steepest Descent for Quadratic
Function F
< CLICK ON ME >
x(2)
x(1)
2
0
-2
-4
-2
0
2
4
Contents
Close
STEEPEST DESCENT
Click anywhere on the graph to create an initial guess. Then the steepest descent trajectory will be shown. You can reset the learning rate using the slider below, and a new trajectory will be shown. Experiment with different initial guesses and learning rates.
Learning Rate:
0.00
(0.03)
0.06
Chapter 9

```
>> nnd9sdq
```

شبکه‌های عصبی مصنوعی

بهینه‌سازی کارآیی

۲

روش نیوتون

Newton's Method

$$F(\mathbf{x}_{k+1}) = F(\mathbf{x}_k + \Delta\mathbf{x}_k) \approx F(\mathbf{x}_k) + \mathbf{g}_k^T \Delta\mathbf{x}_k + \frac{1}{2}\Delta\mathbf{x}_k^T \mathbf{A}_k \Delta\mathbf{x}_k$$

Take the gradient of this second-order approximation and set it equal to zero to find the stationary point:

$$\mathbf{g}_k + \mathbf{A}_k \Delta\mathbf{x}_k = \mathbf{0}$$

$$\Delta\mathbf{x}_k = -\mathbf{A}_k^{-1}\mathbf{g}_k$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{A}_k^{-1}\mathbf{g}_k$$

روش نیوتن

NEWTON'S METHOD

روش تندترین شیب نزولی از تقریب مرتبه اول استفاده می‌کرد. در روش نیوتن از تقریب مرتبه دوم استفاده می‌کنیم:

$$F(\mathbf{x}_{k+1}) = F(\mathbf{x}_k + \Delta\mathbf{x}_k) \approx F(\mathbf{x}_k) + \mathbf{g}_k^T \Delta\mathbf{x}_k + \frac{1}{2}\Delta\mathbf{x}_k^T \mathbf{A}_k \Delta\mathbf{x}_k$$

سری تیلور مرتبه دوم

Take the gradient of this second-order approximation and set it equal to zero to find the stationary point:

$$\mathbf{g}_k + \mathbf{A}_k \Delta\mathbf{x}_k = \mathbf{0}$$

نقاط ایستان:

این روش، می‌نیمم تابع درجه دوم را همیشه در **یک** گام می‌یابد.

ولی برای توابع غیر درجه دوم تضمین نمی‌کند در یک مرحله همگرا شود؛ زیرا در واقع نمی‌توانیم مطمئن باشیم که اصلاً همگرا می‌شود! (چون وابسته به تابع و حدس اولیه است)

$$\Delta\mathbf{x}_k = -\mathbf{A}_k^{-1}\mathbf{g}_k$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{A}_k^{-1}\mathbf{g}_k$$

Example

$$F(\mathbf{x}) = x_1^2 + 2x_1x_2 + 2x_2^2 + x_1$$

$$\mathbf{x}_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} F(\mathbf{x}) \\ \frac{\partial}{\partial x_2} F(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 2x_1 + 2x_2 + 1 \\ 2x_1 + 4x_2 \end{bmatrix}$$

$$\mathbf{g}_0 = \nabla F(\mathbf{x})\Big|_{\mathbf{x} = \mathbf{x}_0} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}$$

$$\mathbf{x}_1 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$$

روش نیوتن

مثال

NEWTON'S METHOD

$$F(\mathbf{x}) = x_1^2 + 2x_1x_2 + 2x_2^2 + x_1$$

$$\mathbf{x}_0 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

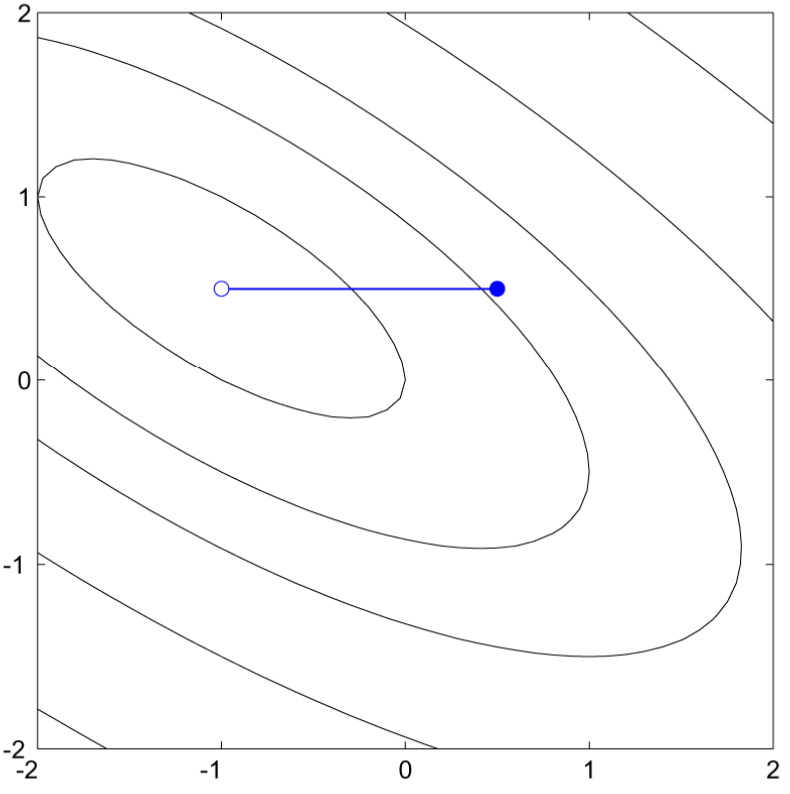
$$\nabla F(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} F(\mathbf{x}) \\ \frac{\partial}{\partial x_2} F(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 2x_1 + 2x_2 + 1 \\ 2x_1 + 4x_2 \end{bmatrix}$$

$$\mathbf{g}_0 = \nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}$$

$$\mathbf{x}_1 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$$

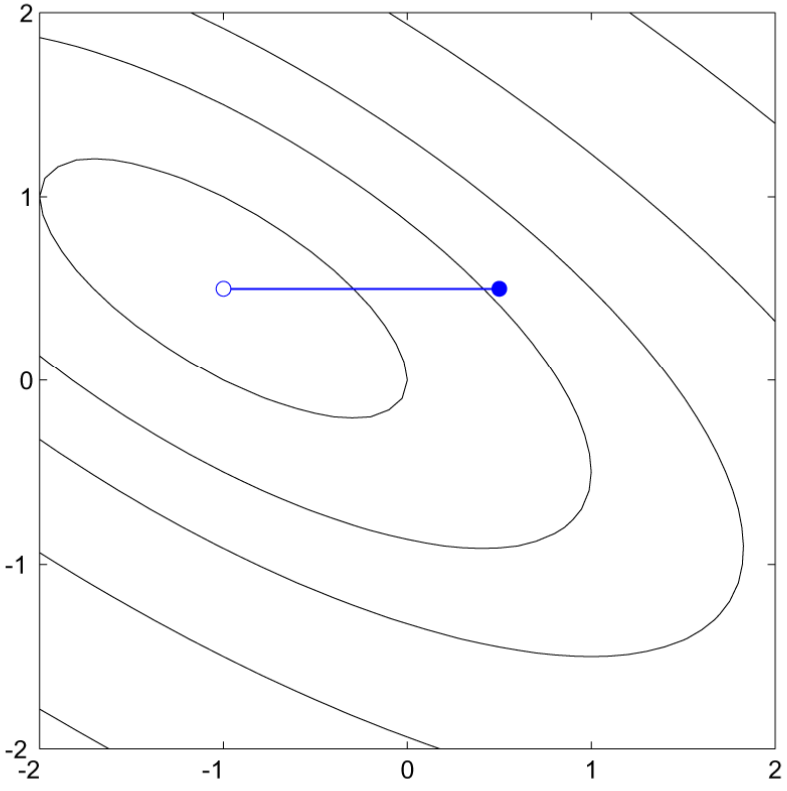
Plot



روش نیوتن

مثال: رسم نمودار

NEWTON'S METHOD

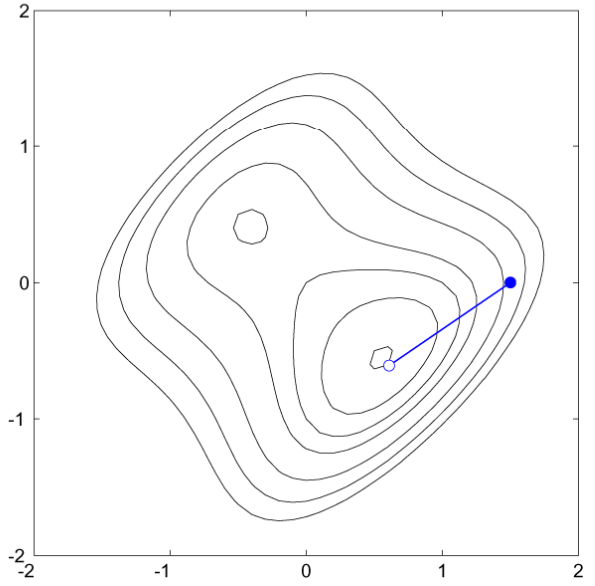


Non-Quadratic Example

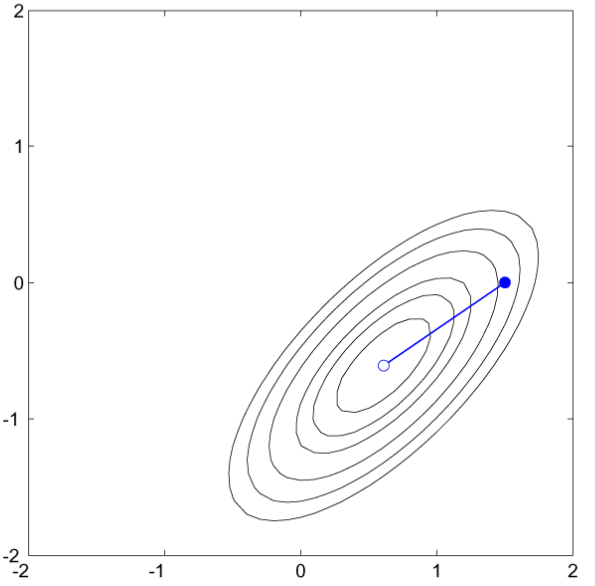
$$F(\mathbf{x}) = (x_2 - x_1)^4 + 8x_1x_2 - x_1 + x_2 + 3$$

Stationary Points: $\mathbf{x}^1 = \begin{bmatrix} -0.42 \\ 0.42 \end{bmatrix}$ $\mathbf{x}^2 = \begin{bmatrix} -0.13 \\ 0.13 \end{bmatrix}$ $\mathbf{x}^3 = \begin{bmatrix} 0.55 \\ -0.55 \end{bmatrix}$

$F(\mathbf{x})$



$F_2(\mathbf{x})$



روش نیوتن

مثال از تابع غیر درجه دوم

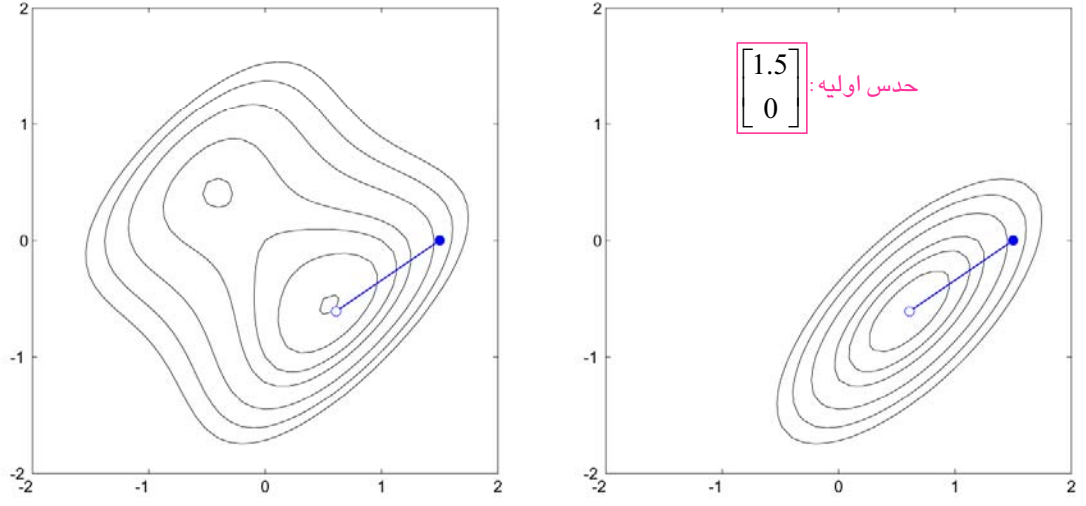
NON-QUADRATIC EXAMPLE

$$F(\mathbf{x}) = (x_2 - x_1)^4 + 8x_1x_2 - x_1 + x_2 + 3$$

Stationary Points: $\mathbf{x}^1 = \begin{bmatrix} -0.42 \\ 0.42 \end{bmatrix}$ $\mathbf{x}^2 = \begin{bmatrix} -0.13 \\ 0.13 \end{bmatrix}$ $\mathbf{x}^3 = \begin{bmatrix} 0.55 \\ -0.55 \end{bmatrix}$

می‌نیمم محلی قوی — نقطه‌ی زینی — می‌نیمم سراسری قوی

تابع اصلی $F(\mathbf{x})$ — تقریب درجه دوم $F_2(\mathbf{x})$



می‌نیمم این تابع، در یک گام پیدا نمی‌شود (عجیب نیست، زیرا تابع درجه دوم نیست)، اما به نقطه‌ی می‌نیمم نزدیک شدیم.

روش نیوتن

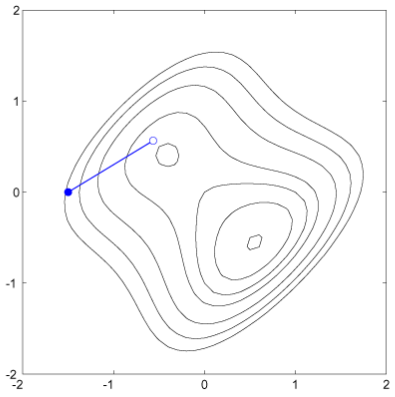
NEWTON'S METHOD

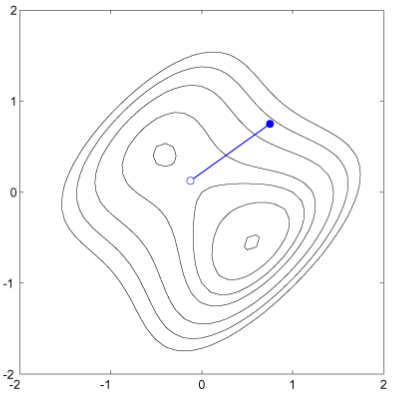
روش نیوتن در بسیاری از کاربردها به‌سرعت همگرا می‌شود،
زیرا توابع تحلیلی می‌توانند در یک همسایگی کوچک از یک می‌نیمم قوی
با دقت کافی توسط یک تابع درجه دوم تقریب زده شوند.

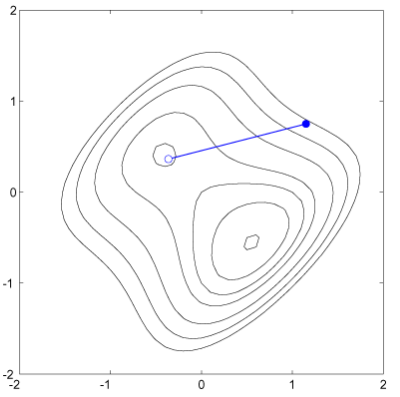
هر چه به نقطه‌ی می‌نیمم نزدیک‌تر شویم،
روش نیوتن با دقت بیشتری محل آن را پیش‌بینی می‌کند.

Different Initial Conditions

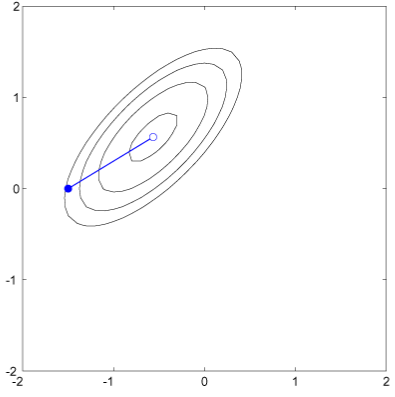
$F(\mathbf{x})$

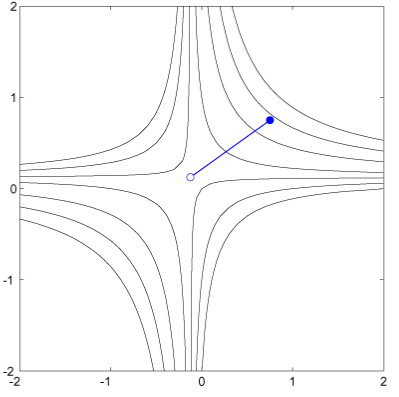


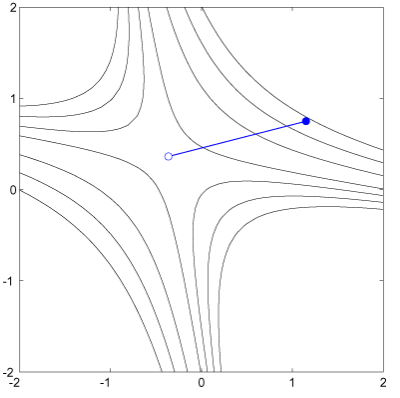




$F_2(\mathbf{x})$



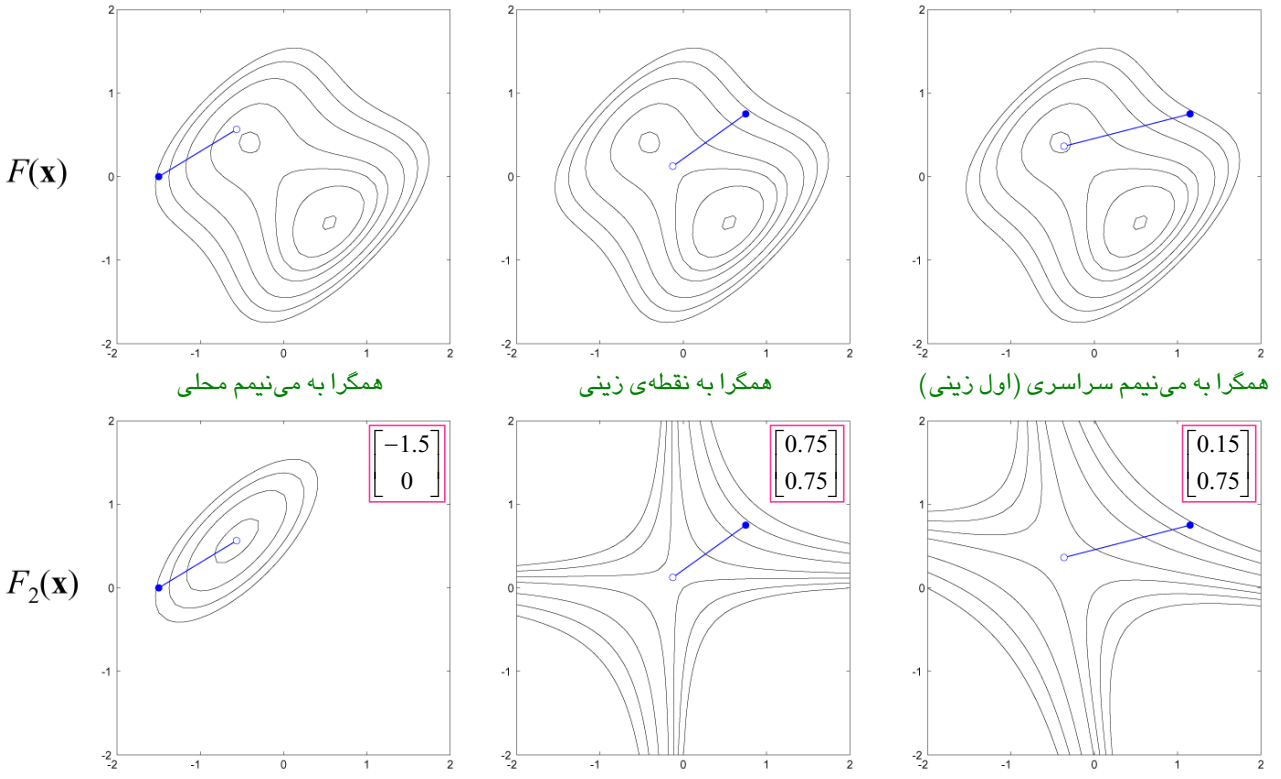




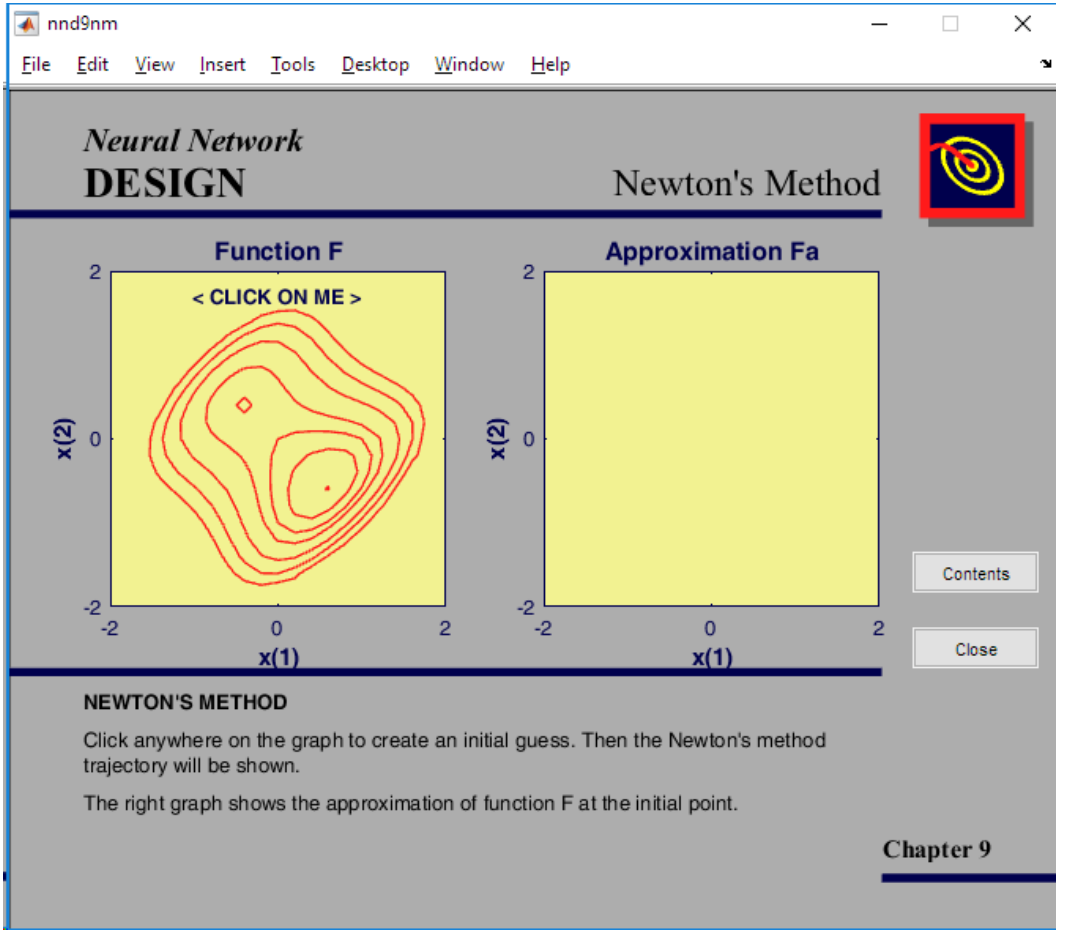
روش نیوتن

مثال از تابع غیر درجه دوم: شرایط آغازین متفاوت

DIFFERENT INITIAL CONDITIONS



نمودار کانتور تقریب درجه دوم در حوالی نزدیک به نقطه‌ی حدس اولیه، شبیه نمودار تابع کانتور اصلی است.



MATLAB

```
>> nnd9nm
```

روش نیوتن

NEWTON'S METHOD

روش نیوتن نسبت به روش تندترین شیب، سریع‌تر همگرا می‌شود
اما رفتار آن کاملاً پیچیده است.

- روش نیوتن مشکل احتمال همگرایی به نقطه‌ی زینی دارد.
- روش نیوتن امکان نوسان / واگرایی دارد (اما روش sd به شرط α تضمین همگرایی داشت.)
- روش نیوتن مشکل محاسبه و ذخیره‌ی ماتریس هسی $\mathbf{H}$ و معکوس آن $\mathbf{H}^{-1}$ را دارد.

روش‌های شبه–نیوتن

QUASI-NEWTON METHODS

با هدف رفع مشکل نیاز به محاسبه‌ی هسی:

روش‌های **شبه–نیوتن** (quasi-Newton) یا **سکانت تک–گام** (one-step-secant)

در این روش‌ها $\mathbf{A}_k^{-1}$ با ماتریس معین مثبت $\mathbf{H}_k$ جایگزین می‌شود
که در هر تکرار بدون نیاز به وارون‌سازی ماتریس، به‌هنگام می‌شود.

برای تابع درجه دوم، $\mathbf{H}_k$ به $\mathbf{A}^{-1}$ همگرا می‌شود.

شبکه‌های عصبی مصنوعی

بهینه‌سازی کارآیی

۳

روش گرادیان مزدوج

روش‌های گرادیان مزدوج

CONJUGATE GRADIENT METHOD

روش نیوتن، خاصیت خوبی به نام «خاتمه‌ی درجه دوم: quadratic termination» دارد:
تابع درجه دوم در تعداد محدودی تکرار می‌نیمم می‌شود.

اما مشکل این روش: نیاز به محاسبه‌ی هسی (مشتقات مرتبه دوم) است.

⇓

به روشی نیاز داریم که: ۱) خاتمه‌ی درجه دوم داشته باشد و ۲) تنها از مشتقات مرتبه اول استفاده کند.

↓

حرکت در مسیر بردارهای مزدوج (به‌جای بردارهای متعامد)

Conjugate Vectors

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{x} + c$$

A set of vectors is mutually conjugate with respect to a positive definite Hessian matrix **A** if

$$\mathbf{p}_k^T\mathbf{A}\mathbf{p}_j = 0 \qquad k \neq j$$

One set of conjugate vectors consists of the eigenvectors of **A**.

$$\mathbf{z}_k^T\mathbf{A}\mathbf{z}_j = \lambda_j\mathbf{z}_k^T\mathbf{z}_j = 0 \qquad k \neq j$$

(The eigenvectors of symmetric matrices are orthogonal.)

بردارهای مزدوج

CONJUGATE VECTORS

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{d}^T\mathbf{x} + c$$

هدف: می‌نیمم‌سازی $F(\mathbf{x})$

A set of vectors is mutually conjugate with respect to a positive definite Hessian matrix **A** if

$$\mathbf{p}_k^T\mathbf{A}\mathbf{p}_j = 0 \qquad k \neq j$$

مجموعه بردارهای دوبه‌دو مزدوج نسبت به ماتریس معین مثبت هسی **A**

One set of conjugate vectors consists of the eigenvectors of **A**.

یک مجموعه از بردارهای مزدوج، از بردارهای ویژه‌ی هسی **A** تشکیل می‌شود.

$$\mathbf{z}_k^T\mathbf{A}\mathbf{z}_j = \lambda_j\mathbf{z}_k^T\mathbf{z}_j = 0 \qquad k \neq j$$

مشکل: یافتن این بردارها به ماتریس هسی نیاز دارد.

(The eigenvectors of symmetric matrices are orthogonal.)

یک دنباله از جستجوهای خطی دقیق در راستای هر مجموعه از جهت‌های مزدوج $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ می‌نیمم دقیق هر تابع درجه دوم با n متغیر را حداکثر در n جستجو می‌یابد. **چگونه می‌توانیم این جهت‌های مزدوج را بیابیم؟**

For Quadratic Functions

$$\nabla F(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{d}$$

$$\nabla^2 F(\mathbf{x}) = \mathbf{A}$$

The change in the gradient at iteration k is

$$\Delta \mathbf{g}_k = \mathbf{g}_{k+1} - \mathbf{g}_k = (\mathbf{A}\mathbf{x}_{k+1} + \mathbf{d}) - (\mathbf{A}\mathbf{x}_k + \mathbf{d}) = \mathbf{A}\Delta \mathbf{x}_k$$

where

$$\Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$$

The conjugacy conditions can be rewritten

$$\alpha_k \mathbf{p}_k^T \mathbf{A} \mathbf{p}_j = \Delta \mathbf{x}_k^T \mathbf{A} \mathbf{p}_j = \Delta \mathbf{g}_k^T \mathbf{p}_j = 0 \qquad k \neq j$$

This does not require knowledge of the Hessian matrix.

بردارهای مزدوج

برای توابع درجه دوم

FOR QUADRATIC FUNCTIONS

$$\nabla F(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{d}$$

$$\nabla^2 F(\mathbf{x}) = \mathbf{A}$$

The change in the gradient at iteration k is

$$\Delta \mathbf{g}_k = \mathbf{g}_{k+1} - \mathbf{g}_k = (\mathbf{A}\mathbf{x}_{k+1} + \mathbf{d}) - (\mathbf{A}\mathbf{x}_k + \mathbf{d}) = \mathbf{A}\Delta \mathbf{x}_k$$

where

$$\Delta \mathbf{x}_k = (\mathbf{x}_{k+1} - \mathbf{x}_k) = \alpha_k \mathbf{p}_k$$

α_k به‌گونه‌ای انتخاب می‌شود که $F(\mathbf{x})$ را در جهت $\mathbf{p}_k$ می‌نیمم نماید.

The conjugacy conditions can be rewritten

$$\alpha_k \mathbf{p}_k^T \mathbf{A}\mathbf{p}_j = \Delta \mathbf{x}_k^T \mathbf{A}\mathbf{p}_j = \Delta \mathbf{g}_k^T \mathbf{p}_j = 0 \qquad k \neq j$$

بیان شرط مزدوج بودن بدون نیاز به ماتریس هسی: برحسب تغییرات گرادیان

This does not require knowledge of the Hessian matrix.

جهت‌های جستجو مزدوج هستند، اگر بر تغییرات گرادیان **عمود** باشند.

Forming Conjugate Directions

Choose the initial search direction as the negative of the gradient.

$$\mathbf{p}_0 = -\mathbf{g}_0$$

Choose subsequent search directions to be conjugate.

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1}$$

where

$$\beta_k = \frac{\Delta\mathbf{g}_{k-1}^T \mathbf{g}_k}{\Delta\mathbf{g}_{k-1}^T \mathbf{p}_{k-1}} \quad \text{or} \quad \beta_k = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}} \quad \text{or} \quad \beta_k = \frac{\Delta\mathbf{g}_{k-1}^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}}$$

بردارهای مزدوج

تشکیل جهت‌های مزدوج

FORMING CONJUGATE DIRECTIONS

$\mathbf{p}_0$ دلخواه، $\mathbf{p}_1$ عمود بر $\Delta\mathbf{g}_0$، ... ، $\mathbf{p}_k$ عمود بر $\Delta\mathbf{g}_0, \Delta\mathbf{g}_1, \ldots, \Delta\mathbf{g}_{k-1}$
⇐ (مشابه متعامدسازی گرام–اشمیت)

Choose the initial search direction as the negative of the gradient.

$$\mathbf{p}_0 = -\mathbf{g}_0$$

Choose subsequent search directions to be conjugate.

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1}$$

where

$$\beta_k = \frac{\Delta\mathbf{g}_{k-1}^T \mathbf{g}_k}{\Delta\mathbf{g}_{k-1}^T \mathbf{p}_{k-1}} \quad \text{or} \quad \beta_k = \frac{\mathbf{g}_k^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}} \quad \text{or} \quad \beta_k = \frac{\Delta\mathbf{g}_{k-1}^T \mathbf{g}_k}{\mathbf{g}_{k-1}^T \mathbf{g}_{k-1}}$$

Conjugate Gradient algorithm

- The first search direction is the negative of the gradient.

$$\mathbf{p}_0 = -\mathbf{g}_0$$

- Select the learning rate to minimize along the line.

$$\alpha_k = -\frac{\nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k}\mathbf{p}_k}{\mathbf{p}_k^T\nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k}\mathbf{p}_k} = -\frac{\mathbf{g}_k^T\mathbf{p}_k}{\mathbf{p}_k^T\mathbf{A}_k\mathbf{p}_k}$$

(For quadratic functions.)

- Select the next search direction using

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k\mathbf{p}_{k-1}$$

- If the algorithm has not converged, return to second step.

- A quadratic function will be minimized in *n* steps.

الگوریتم گرادیان مزدوج

CONJUGATE GRADIENT ALGORITHM

- The first search direction is the negative of the gradient.

$$\mathbf{p}_0 = -\mathbf{g}_0$$

- Select the learning rate to minimize along the line.

$$\alpha_k = -\frac{\nabla F(\mathbf{x})^T\Big|_{\mathbf{x}=\mathbf{x}_k}\mathbf{p}_k}{\mathbf{p}_k^T\nabla^2 F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_k}\mathbf{p}_k} = -\frac{\mathbf{g}_k^T\mathbf{p}_k}{\mathbf{p}_k^T\mathbf{A}_k\mathbf{p}_k}$$

(For quadratic functions.)

- Select the next search direction using

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k\mathbf{p}_{k-1}$$

- If the algorithm has not converged, return to second step.

- A quadratic function will be minimized in n steps.

با این الگوریتم تابع درجه دوم در n گام می‌نیمم می‌شود.

Example

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\mathbf{x} + \begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x} \qquad \mathbf{x}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix}$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix}\frac{\partial}{\partial x_1}F(\mathbf{x})\\ \frac{\partial}{\partial x_2}F(\mathbf{x})\end{bmatrix} = \begin{bmatrix}2x_1 + 2x_2 + 1\\ 2x_1 + 4x_2\end{bmatrix} \qquad \mathbf{p}_0 = -\mathbf{g}_0 = -\nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix}-3\\ -3\end{bmatrix}$$

$$\alpha_0 = -\frac{\begin{bmatrix}3 & 3\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}}{\begin{bmatrix}-3 & -3\end{bmatrix}\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}} = 0.2 \qquad \mathbf{x}_1 = \mathbf{x}_0 - \alpha_0\mathbf{g}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix} - 0.2\begin{bmatrix}3\\ 3\end{bmatrix} = \begin{bmatrix}-0.1\\ -0.1\end{bmatrix}$$

الگوریتم گرادیان مزدوج

مثال (۱ از ۳)

CONJUGATE GRADIENT ALGORITHM

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\mathbf{x} + \begin{bmatrix}1 & 0\end{bmatrix}\mathbf{x} \qquad \mathbf{x}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix}$$

$$\nabla F(\mathbf{x}) = \begin{bmatrix}\frac{\partial}{\partial x_1}F(\mathbf{x})\\ \frac{\partial}{\partial x_2}F(\mathbf{x})\end{bmatrix} = \begin{bmatrix}2x_1 + 2x_2 + 1\\ 2x_1 + 4x_2\end{bmatrix} \qquad \mathbf{p}_0 = -\mathbf{g}_0 = -\nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_0} = \begin{bmatrix}-3\\ -3\end{bmatrix}$$

$$\alpha_0 = -\frac{\begin{bmatrix}3 & 3\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}}{\begin{bmatrix}-3 & -3\end{bmatrix}\begin{bmatrix}2 & 2\\ 2 & 4\end{bmatrix}\begin{bmatrix}-3\\ -3\end{bmatrix}} = 0.2 \qquad \mathbf{x}_1 = \mathbf{x}_0 - \alpha_0\mathbf{g}_0 = \begin{bmatrix}0.5\\ 0.5\end{bmatrix} - 0.2\begin{bmatrix}3\\ 3\end{bmatrix} = \begin{bmatrix}-0.1\\ -0.1\end{bmatrix}$$

Example

$$\mathbf{g}_1 = \nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_1} = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} -0.1 \\ -0.1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}$$

$$\beta_1 = \frac{\mathbf{g}_1^T\mathbf{g}_1}{\mathbf{g}_0^T\mathbf{g}_0} = \frac{\begin{bmatrix} 0.6 & -0.6 \end{bmatrix}\begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}}{\begin{bmatrix} 3 & 3 \end{bmatrix}\begin{bmatrix} 3 \\ 3 \end{bmatrix}} = \frac{0.72}{18} = 0.04$$

$$\mathbf{p}_1 = -\mathbf{g}_1 + \beta_1\mathbf{p}_0 = \begin{bmatrix} -0.6 \\ 0.6 \end{bmatrix} + 0.04\begin{bmatrix} -3 \\ -3 \end{bmatrix} = \begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix}$$

$$\alpha_1 = -\frac{\begin{bmatrix} 0.6 & -0.6 \end{bmatrix}\begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix}}{\begin{bmatrix} -0.72 & 0.48 \end{bmatrix}\begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix}} = -\frac{-0.72}{0.576} = 1.25$$

الگوریتم گرادیان مزدوج

مثال (۲ از ۳)

CONJUGATE GRADIENT ALGORITHM

$$\mathbf{g}_1 = \nabla F(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_1} = \begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} -0.1 \\ -0.1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}$$

$$\beta_1 = \frac{\mathbf{g}_1^T \mathbf{g}_1}{\mathbf{g}_0^T \mathbf{g}_0} = \frac{\begin{bmatrix} 0.6 & -0.6 \end{bmatrix}\begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}}{\begin{bmatrix} 3 & 3 \end{bmatrix}\begin{bmatrix} 3 \\ 3 \end{bmatrix}} = \frac{0.72}{18} = 0.04$$

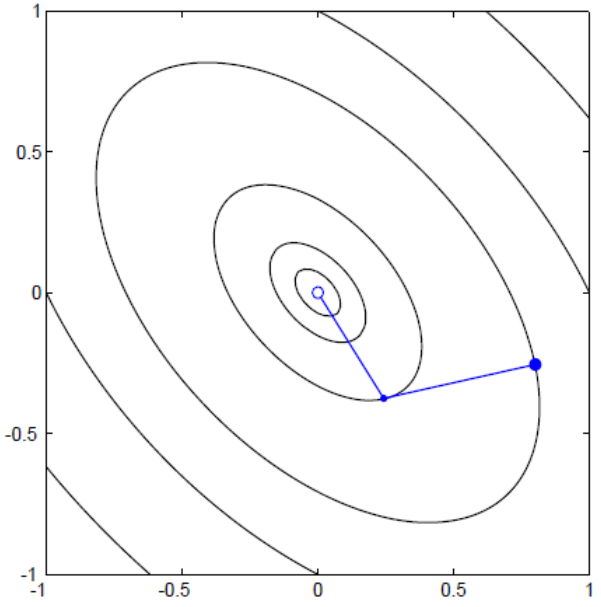
$$\mathbf{p}_1 = -\mathbf{g}_1 + \beta_1 \mathbf{p}_0 = \begin{bmatrix} -0.6 \\ 0.6 \end{bmatrix} + 0.04\begin{bmatrix} -3 \\ -3 \end{bmatrix} = \begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix}$$

$$\alpha_1 = -\frac{\begin{bmatrix} 0.6 & -0.6 \end{bmatrix}\begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix}}{\begin{bmatrix} -0.72 & 0.48 \end{bmatrix}\begin{bmatrix} 2 & 2 \\ 2 & 4 \end{bmatrix}\begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix}} = -\frac{-0.72}{0.576} = 1.25$$

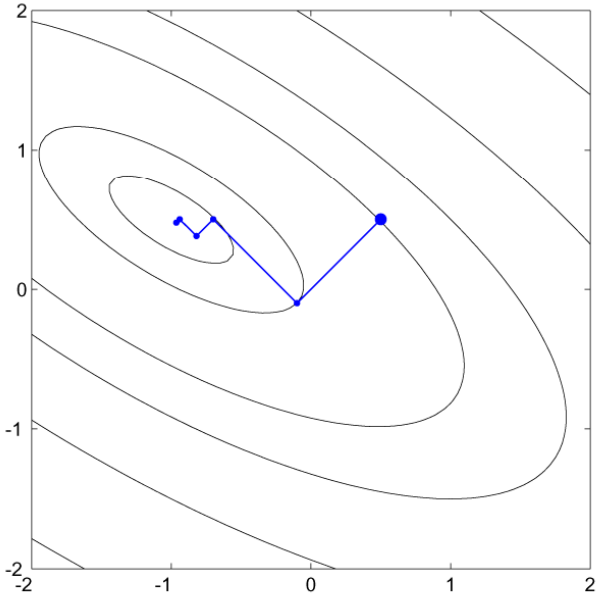
Plots

$$\mathbf{x}_2 = \mathbf{x}_1 + \alpha_1 \mathbf{p}_1 = \begin{bmatrix} -0.1 \\ -0.1 \end{bmatrix} + 1.25 \begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix} = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$$

Conjugate Gradient



Steepest Descent



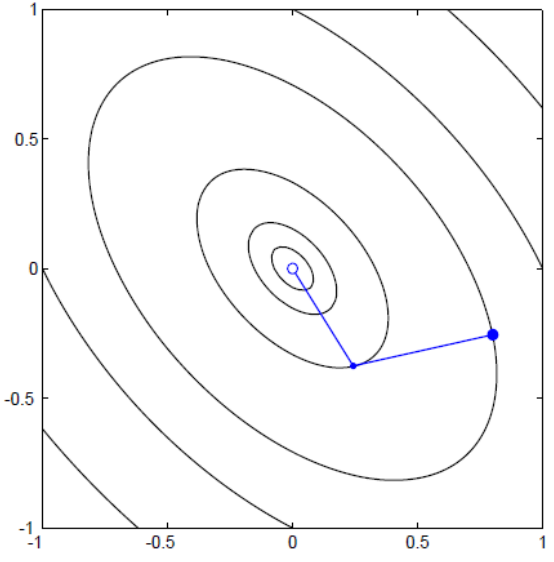
الگوریتم گرادیان مزدوج

مثال (۳ از ۳)

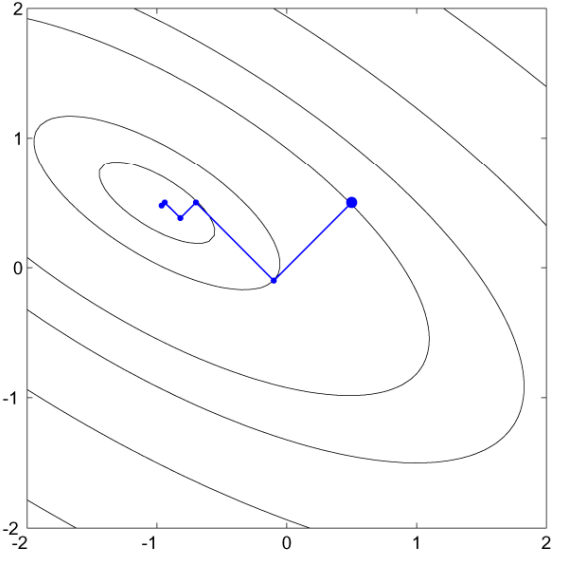
CONJUGATE GRADIENT ALGORITHM

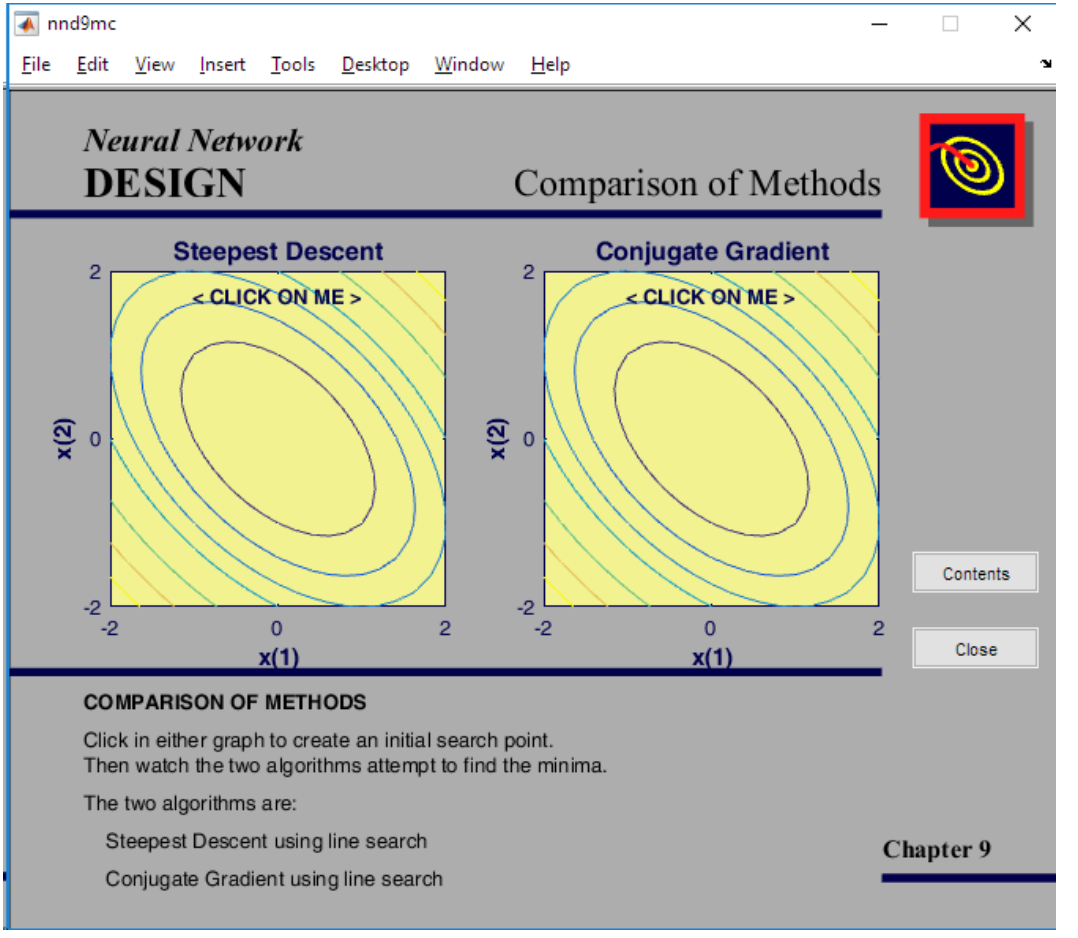
$$\mathbf{x}_2 = \mathbf{x}_1 + \alpha_1 \mathbf{p}_1 = \begin{bmatrix} -0.1 \\ -0.1 \end{bmatrix} + 1.25 \begin{bmatrix} -0.72 \\ 0.48 \end{bmatrix} = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}$$

Conjugate Gradient



Steepest Descent





MATLAB

```
>> nnd9mc
```

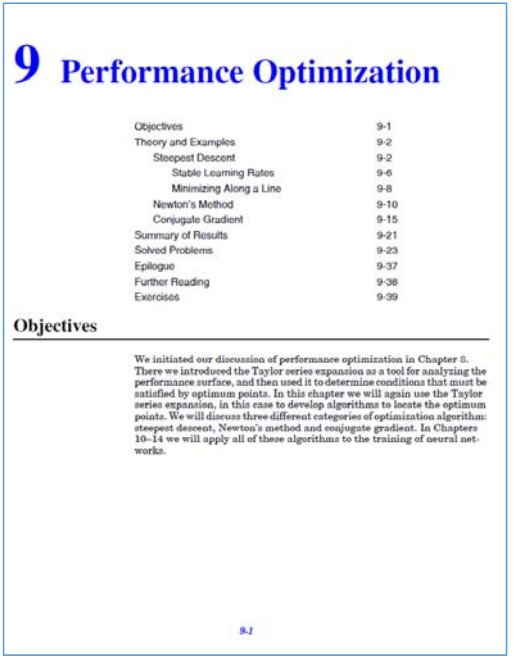
شبکه‌های عصبی مصنوعی

بهینه‌سازی کارآیی

۴

منابع

منبع اصلی



9 Performance Optimization

Objectives

We initiated our discussion of performance optimization in Chapter 8. There we introduced the Taylor series expansion as a tool for analyzing the performance surface, and then used it to determine conditions that must be satisfied by optimum points. In this chapter we will again use the Taylor series expansion, in this case to develop algorithms to locate the optimum points. We will discuss three different categories of optimization algorithm: steepest descent, Newton's method and conjugate gradient. In Chapters 10–14 we will apply all of these algorithms to the training of neural networks.

9-1

Martin T. Hagan, Howard B. Demuth, Mark H. Beale, Orlando De Jesus,
Neural Network Design,
2nd Edition, Martin Hagan, 2014.
Chapter 9

Online version can be downloaded from: http://hagan.okstate.edu/nnd.html