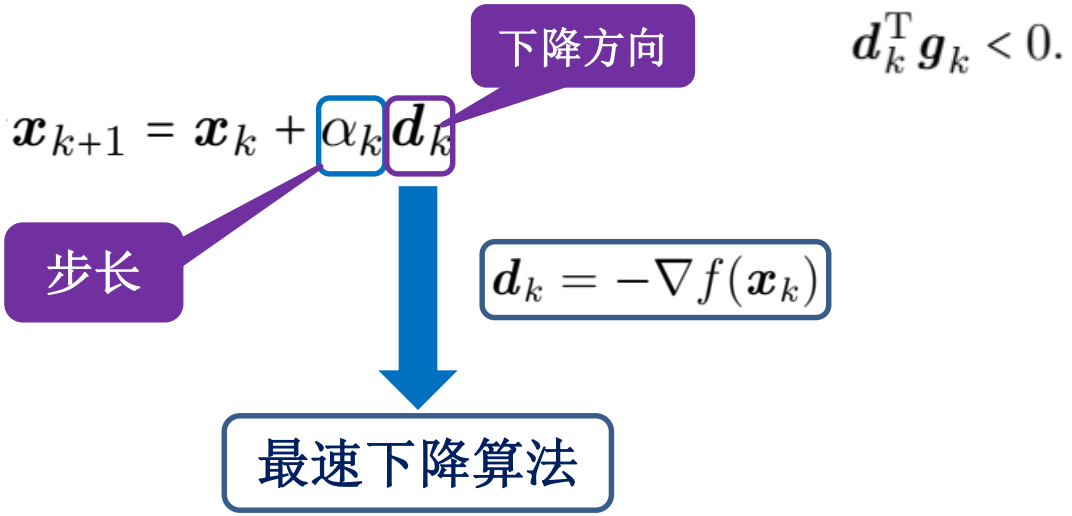
最速下降法与牛顿方法

一、最速下降法

优化模型

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x})$$

迭代算法



$$\boldsymbol{d}_k^{\mathrm{T}} \boldsymbol{g}_k < 0.$$

最速下降算法

步1、取初始点$\boldsymbol{x}_0$，参数$\varepsilon \geqslant 0$，令$k=0$.

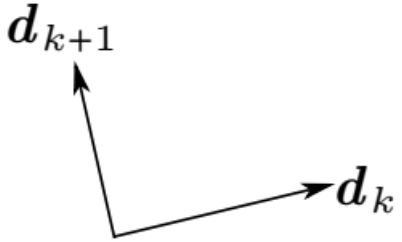
步2、若$\|\boldsymbol{g}_k\| \leqslant \varepsilon$，算法停止；否则，进入下一步.

步3、取搜索方向$\boldsymbol{d}_k=-\boldsymbol{g}_k$，利用精确或非精确线搜索产生步长$\alpha_k$.

步4、令$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k+\alpha_k \boldsymbol{d}_k$，$k=k+1$返回步2.

最优步长规则下的收敛性质

迭代方向的正交性 $\boldsymbol{d}_k^{\mathsf{T}}\boldsymbol{d}_{k+1}=0.$



证明： 记 $\varphi_k(\alpha)=f(\boldsymbol{x}_k+\alpha\boldsymbol{d}_k).$

利用最优步长规则，

$$\begin{aligned}0=\varphi_k'(\alpha_k)&=\langle\nabla f(\boldsymbol{x}_k+\alpha_k\boldsymbol{d}_k),\boldsymbol{d}_k\rangle\\&=\langle\nabla f(\boldsymbol{x}_{k+1}),\boldsymbol{d}_k\rangle\\&=-\langle\boldsymbol{d}_{k+1},\boldsymbol{d}_k\rangle.\end{aligned}$$

算法收敛性

定理 设目标函数连续可微。则最优步长规则下的最速下降算法产生的点列的任一聚点满足 $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$

证明： 将 $\boldsymbol{d}_k = -\boldsymbol{g}_k$ 应用于最优步长下降算法的收敛性结论。

收敛速度估计

定理 对严格凸二次函数 $f(\boldsymbol{x})=\frac{1}{2}\boldsymbol{x}^{\top}\boldsymbol{G}\boldsymbol{x}$，$\lambda_1\geqslant\lambda_2\geqslant\cdots\geqslant\lambda_n>0$ 为矩阵 $\boldsymbol{G}$ 的特征根。则最优步长最速下降算法线性收敛，即

$$\frac{f_{k+1}-f(\boldsymbol{x}^*)}{f_k-f(\boldsymbol{x}^*)}\leqslant\left(\frac{\lambda_1-\lambda_n}{\lambda_1+\lambda_n}\right)^2,$$

目标函数值线性下降

$$\frac{\|\boldsymbol{x}_{k+1}-\boldsymbol{x}^*\|}{\|\boldsymbol{x}_k-\boldsymbol{x}^*\|}\leqslant\frac{\lambda_1-\lambda_n}{\lambda_1+\lambda_n}\sqrt{\frac{\lambda_1}{\lambda_n}}.$$

迭代点列线性收敛

$\boldsymbol{x}^*=\boldsymbol{0}$ 为目标函数的最小值点.

- 很强条件下(二次严格凸)最速下降算法的收敛速度估计；
- 线性收敛是一个很慢的收敛速度；
- 有例为证：该理论估计是准确的，没有提升的空间。

例 $\min\limits_{x\in R^2} f(x_1,x_2)=\frac{1}{3}x_1^2+\frac{1}{2}x_2^2$ **唯一最优解** $x^*=(0;0)$

取初始点 $x_0=(3;2)$

迭代点列 $x_k=\left(\frac{3}{5^k};(-1)^k\frac{2}{5^k}\right)\overset{k\to\infty}{\longrightarrow}(0;0)$

全局收敛，收敛速度线性！ $\|x_k-x^*\|=\sqrt{13}\left(\frac{1}{5}\right)^k.$

$$\frac{\|x_{k+1}-x^*\|}{\|x_k-x^*\|}=\frac{1}{5}<\frac{1}{5}\sqrt{\frac{3}{2}}=\frac{\lambda_1-\lambda_2}{\lambda_1+\lambda_2}\sqrt{\frac{\lambda_1}{\lambda_2}}$$

$$\frac{f(x_{k+1})-f(x^*)}{f(x_k)-f(x^*)}=\frac{1}{25}=\left(\frac{\lambda_1-\lambda_2}{\lambda_1+\lambda_2}\right)^2$$

算法分析

- 搜索方向(最速下降)——下降性是最好的

$$\nabla f(\boldsymbol{x}_k)^{\top}\boldsymbol{d}_k \leqslant \nabla f(\boldsymbol{x}_k)^{\top}\boldsymbol{d}, \quad \forall\ \boldsymbol{d} \in R^n,\ \|\boldsymbol{d}\| = \|\boldsymbol{d}_k\| = 1$$

- 迭代步长(精确搜索)——最优的

$$f(\boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k) \leqslant f(\boldsymbol{x}_k + \alpha \boldsymbol{d}_k), \quad \forall\ \alpha \geqslant 0$$

- 理论性质和数值结果均不好，较强条件下线性收敛！

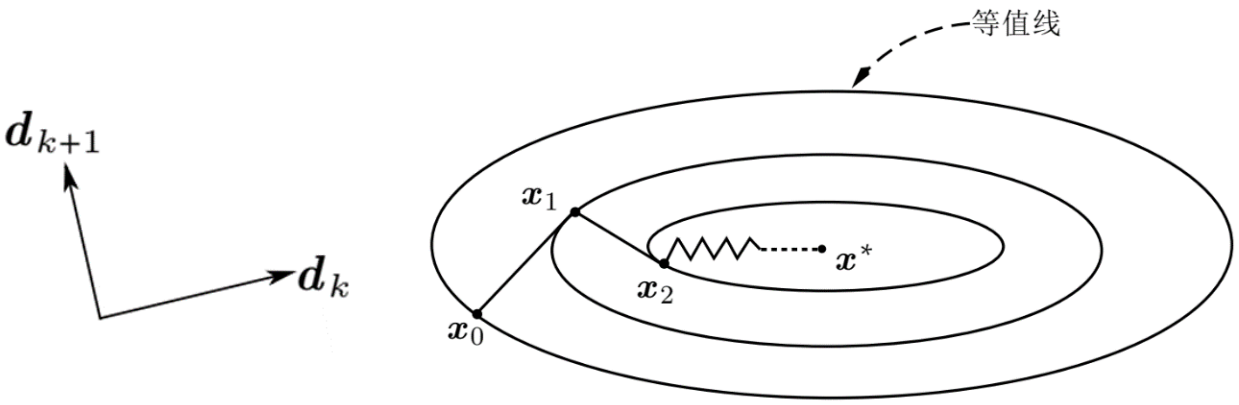
算法分析

优点 $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{d}_k \quad \boldsymbol{d}_k = -\boldsymbol{g}_k$

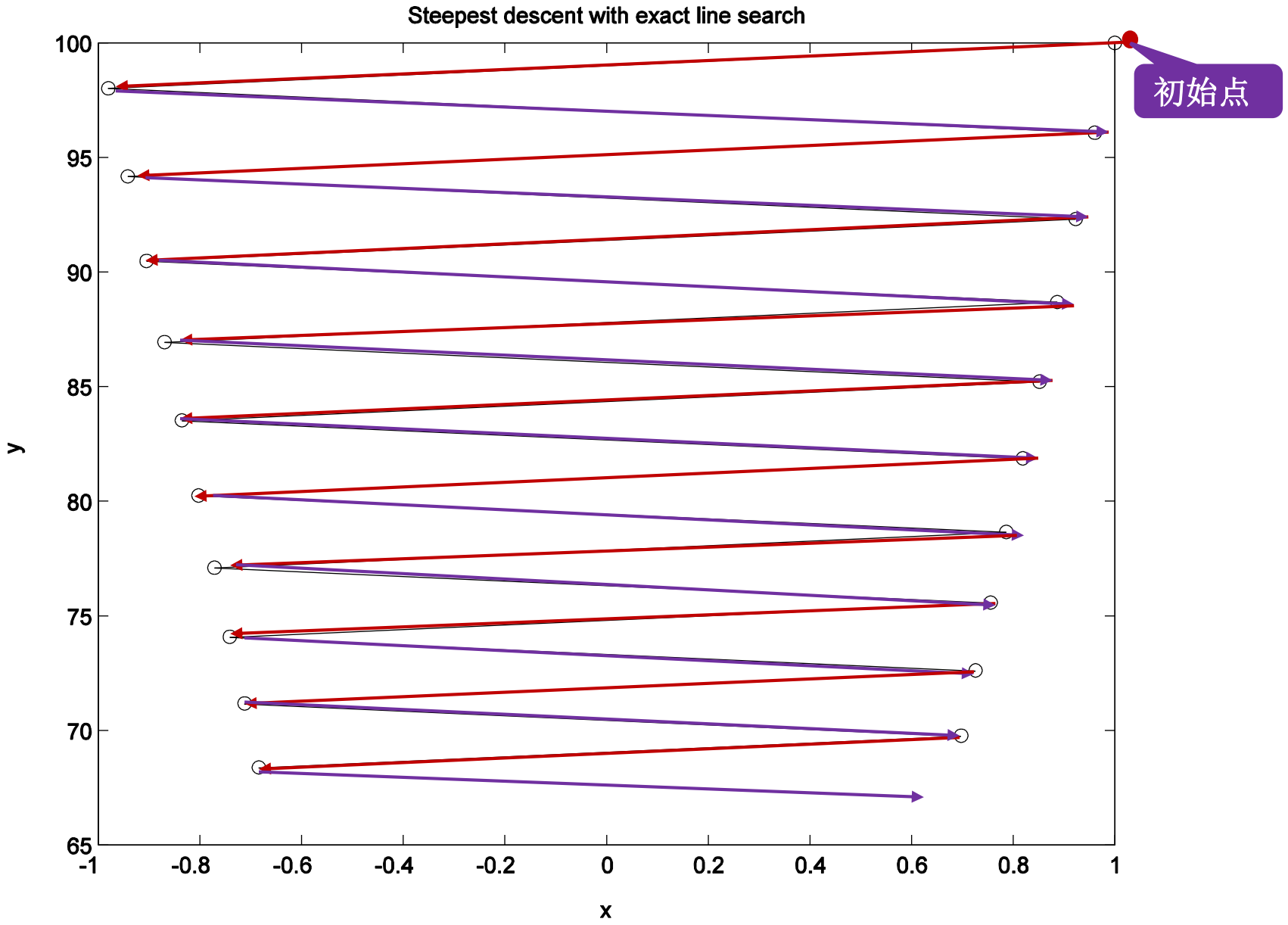
迭代过程简单，计算量与存储量小；

初始步函数值下降较快，可以快速靠近最优解。

缺陷 相邻两搜索方向正交, 导致锯齿现象：越靠近最优值点，靠近速度越慢。不适于算法收局。



最速下降算法的迭代过程



$$f(\boldsymbol{x}) = 100x_1^2 + x_2^2$$

原因分析

最速下降法的搜索方向基于目标函数的线性近似

$$f(\boldsymbol{x}_k + \boldsymbol{d}) = f(\boldsymbol{x}_k) + \boldsymbol{d}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k)$$

$$\min\{f(\boldsymbol{x}_k) + \boldsymbol{d}^{\mathrm{T}} \nabla f(\boldsymbol{x}_k) \mid \|\boldsymbol{d}\| \leqslant 1\}$$

Cauchy-Schwarz不等式

二阶近似

牛顿算法

二、牛顿算法

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \quad f(\boldsymbol{x})$$

利用目标函数 $f(\boldsymbol{x}_k+\boldsymbol{d})$ 在 $\boldsymbol{x}_k$ 点的二阶近似求新的迭代点：

$$m_k(\boldsymbol{d}) \triangleq f(\boldsymbol{x}_k)+\boldsymbol{d}^{\mathrm{T}}\boldsymbol{g}_k+\frac{1}{2}\boldsymbol{d}^{\mathrm{T}}\boldsymbol{G}_k\boldsymbol{d}.$$

利用最优性条件得其最小值点 $\boldsymbol{d}_k^N=-\boldsymbol{G}_k^{-1}\boldsymbol{g}_k.$

令 $\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\boldsymbol{G}_k^{-1}\boldsymbol{g}_k$ 即得**牛顿算法**。

牛顿步 $\boldsymbol{d}_k^N=-\boldsymbol{G}_k^{-1}\boldsymbol{g}_k$

算法框架

步1、取初始点 $\boldsymbol{x}_0$，参数 $\varepsilon \geqslant 0$，令 $k=0$.

步2、若 $\|\boldsymbol{g}_k\| \leqslant \varepsilon$，算法停止；否则，进入下一步.

步3、令 $\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\boldsymbol{G}_k^{-1}\boldsymbol{g}_k$，返回步2.

算法特点：步长恒取1，无线搜索；

收敛性与收敛速度

定理 设目标函数二阶连续可微,Hesse阵在最优值点 x^* 非奇异.

则 (1) 若初始点充分靠近最优值点,则算法超线性收敛,

$$\lim_{k\to\infty} x_k = x^*;$$

(2) 若Hesse阵在最优值点附近Lipschitz连续, 则二阶收敛。

证明要点：利用目标函数二阶展式

$$0 = g(x^*) = g_k + G_k(x^* - x_k) + o(\|x_k - x^*\|).$$

左乘 G_k^{-1}，

$$x_k - x^* - G_k^{-1} g_k = o(\|x_k - x^*\|)$$

$$x_{k+1} - x^* = o(\|x_k - x^*\|).$$ 得结论（1）。

对后一结论，由目标函数Hesse阵的Lipschitz连续性

$$
\begin{aligned}
\|\boldsymbol{x}_{k+1}-\boldsymbol{x}^*\| &= \|\boldsymbol{x}_k-\boldsymbol{x}^*-\boldsymbol{G}_k^{-1}\boldsymbol{g}_k\| \\
&= \|\boldsymbol{G}_k^{-1}\big[\boldsymbol{G}_k(\boldsymbol{x}_k-\boldsymbol{x}^*)-\boldsymbol{g}_k+\boldsymbol{g}(\boldsymbol{x}^*)\big]\| \\
&\leqslant M\|\int_0^1[\boldsymbol{G}_k-\boldsymbol{G}(\boldsymbol{x}^*+\tau(\boldsymbol{x}_k-\boldsymbol{x}^*))](\boldsymbol{x}_k-\boldsymbol{x}^*)\mathrm{d}\tau\| \\
&\leqslant LM\|\boldsymbol{x}_k-\boldsymbol{x}^*\|^2\int_0^1(1-\tau)\mathrm{d}\tau \\
&= \frac{1}{2}LM\|\boldsymbol{x}_k-\boldsymbol{x}^*\|^2.
\end{aligned}
$$

算法特点

无全局收敛性。仅局部收敛，即要求初始点靠近最优值点；

二阶收敛 ：迭代点越靠近最优值点，收敛速度越快！

具有二次终止性：对严格凸二次函数，一步即得最优解

严格凸二次函数 $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\top}\boldsymbol{G}\boldsymbol{x} + \boldsymbol{g}^{\top}\boldsymbol{x} + c$ ⇒ $\boldsymbol{x}^{*} = -\boldsymbol{G}^{-1}\boldsymbol{g}$

任意初始点 $\boldsymbol{x}_0 \in R^n$

$$\begin{aligned}\boldsymbol{x}_1 &= \boldsymbol{x}_0 - \boldsymbol{G}^{-1}\nabla f(\boldsymbol{x}_0)\\ &= \boldsymbol{x}_0 - \boldsymbol{G}^{-1}(\boldsymbol{G}\boldsymbol{x}_0 + \boldsymbol{g})\\ &= -\boldsymbol{G}^{-1}\boldsymbol{g} = \boldsymbol{x}^{*}.\end{aligned}$$

例：用牛顿算法求解 $\min f(x) = \sqrt{1+x^2}$

解：0为最优解。

目标函数导数 $f'(x) = \dfrac{x}{\sqrt{1+x^2}}$ $f''(x) = \dfrac{1}{(1+x^2)^{3/2}}$

迭代过程

$$\begin{aligned} x_{k+1} &= x_k - \frac{f'(x_k)}{f''(x_k)} \\ &= x_k - x_k(1+x_k^2) \\ &= -x_k^3 \end{aligned}$$

当 $|x_0| < 1$ 时，算法快速收敛到最优解；

当 $|x_0| \geqslant 1$ 时，算法不收敛.

例：用牛顿算法求解 $\min f(\boldsymbol{x}) = 4x_1^2 + x_2^2 - x_1^2 x_2$

解： **最优解** $x^* = (0;0)$ **鞍点** $(2\sqrt{2};4)$ $(-2\sqrt{2};4)$

目标函数梯度信息

$$\nabla f(\boldsymbol{x}) = \begin{pmatrix} 8x_1 - 2x_1x_2 \\ 2x_2 - x_1^2 \end{pmatrix} \qquad \nabla^2 f(\boldsymbol{x}) = \begin{pmatrix} 8 - 2x_2 & -2x_1 \\ -2x_1 & 2 \end{pmatrix}$$

取精度 $\varepsilon = 10^{-3}$ 和不同初始点用牛顿算法求解

$$\min f(\boldsymbol{x}) = 4x_1^2 + x_2^2 - x_1^2 x_2$$

初始点 $x_0 = (1;1)$

目标函数值非单调

k	$x^{(k)}$	$f\left(x^{(k)}\right)$	$\nabla f\left(x^{(k)}\right)$	$\left\|\nabla f\left(x^{(k)}\right)\right\|$	$\nabla^2 f\left(x^{(k)}\right)$
1	1.0000 1.0000	4.000	6.0000 1.0000	6.0828	6.0000 – 2.0000 – 2.000 2.0000
2	– 0.7500 – 1.2500	4.5156	– 7.8750 – 3.0625	8.4495	10.500 1.5000 1.5000 2.0000
3	– 0.1550 – 0.1650	0.1273	– 1.2911 – 0.3540	1.3388	8.3300 0.3100 0.3100 2.0000
4	– 0.0057 – 0.0111	0.0003	– 0.0459 – 0.0223	0.0511	8.0222 0.0115 0.0115 2.0000
5	– 0.0000 – 0.0000	0.0000	– 0.0001 – 0.0000	0.0001	8.0000 0.0000 0.0000 2.0000

最优解

初始点 $x_0 = (3; 4)$

k	$x^{(k)}$	$f\left(x^{(k)}\right)$	$\nabla f\left(x^{(k)}\right)$	$\left\|\nabla f\left(x^{(k)}\right)\right\|$	$\nabla^2 f\left(x^{(k)}\right)$
1	3.0000 4.0000	16.000	0.0000 −1.0000	1.0000	0.0000 −6.0000 −6.000 2.0000
2	2.8333 4.0000	16.0000	0.0000 −0.2078	0.0278	0.0000 −5.6667 −5.6667 2.0000
3	2.8284 4.0000	16.0000	0.0000 0.0000	0.0000	0.0000 −5.6569 −5.6569 2.0000

鞍点

初始点 $x_0 = (2;0)$

$$\nabla^2 f(\boldsymbol{x}_0) = \begin{pmatrix} 8 & -4 \\ -4 & 2 \end{pmatrix}$$

目标函数Hesse阵奇异，不能进行下一步迭代。算法终止

牛顿方法的缺陷

- 初始点远离最优解时，算法可能不收敛；
- Hesse矩阵奇异时，算法不可行；
- 计算量和存储量大：需计算目标函数的梯度和Hesse阵

一些补救和改进措施：

- 引入线搜索，阻尼牛顿法，保证收敛性

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\boldsymbol{G}_k^{-1}\boldsymbol{g}_k \quad \Rightarrow \quad \boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\alpha_k\boldsymbol{G}^{-1}\boldsymbol{g}_k$$

$$f(\boldsymbol{x}_{k+1})<f(\boldsymbol{x}_k)$$

- 结合最速下降法，设计“杂交” 牛顿法，保证下降性

$$\boldsymbol{x}_{k+1}=\boldsymbol{x}_k-\boldsymbol{G}^{-1}\boldsymbol{g}_k \qquad f(\boldsymbol{x}_{k+1}) \overset{?}{<} f(\boldsymbol{x}_k)$$

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

- 求牛顿方程近似解, 建立非精确牛顿算法, 降低计算量

$$\boldsymbol{G}_k\boldsymbol{d}=-\boldsymbol{g}_k \quad \Rightarrow \quad \boldsymbol{G}_k\boldsymbol{d}\approx-\boldsymbol{g}_k$$

$$\boldsymbol{G}_k\boldsymbol{d}=-\boldsymbol{g}_k+\boldsymbol{r}_k, \quad \boldsymbol{r}_k\approx\boldsymbol{0}$$

$$\min_{x\in\mathbb{R}^n} \quad f(x)$$

牛顿算法：求解梯度方程 $g(x)=0$

最终得到的是目标函数的稳定点：$\nabla f(x^*)=0$

算法产生点列的聚点，可能是目标函数的最小值点，也可能是最大值点.

将其用于求解恰定方程组 $F(x)=0$ 得方程组问题的牛顿算法。最常见和最基本的方程组求解算法！

$$\begin{cases} f_1(x_1, x_2, \cdots, x_n) = 0 \\ f_1(x_1, x_2, \cdots, x_n) = 0 \\ \cdots\cdots \\ f_m(x_1, x_2, \cdots, x_n) = 0 \end{cases}$$

$$\begin{cases} \text{恰定方程组，} m = n \\ \text{超定方程组，} \ m > n \\ \text{欠定方程组，} \ m < n \end{cases}$$

一元非线性方程 $f(x)=0$

取初始点 x_0，将函数 $f(x)$ 在该点线性Taylor展开:

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0) = 0$$

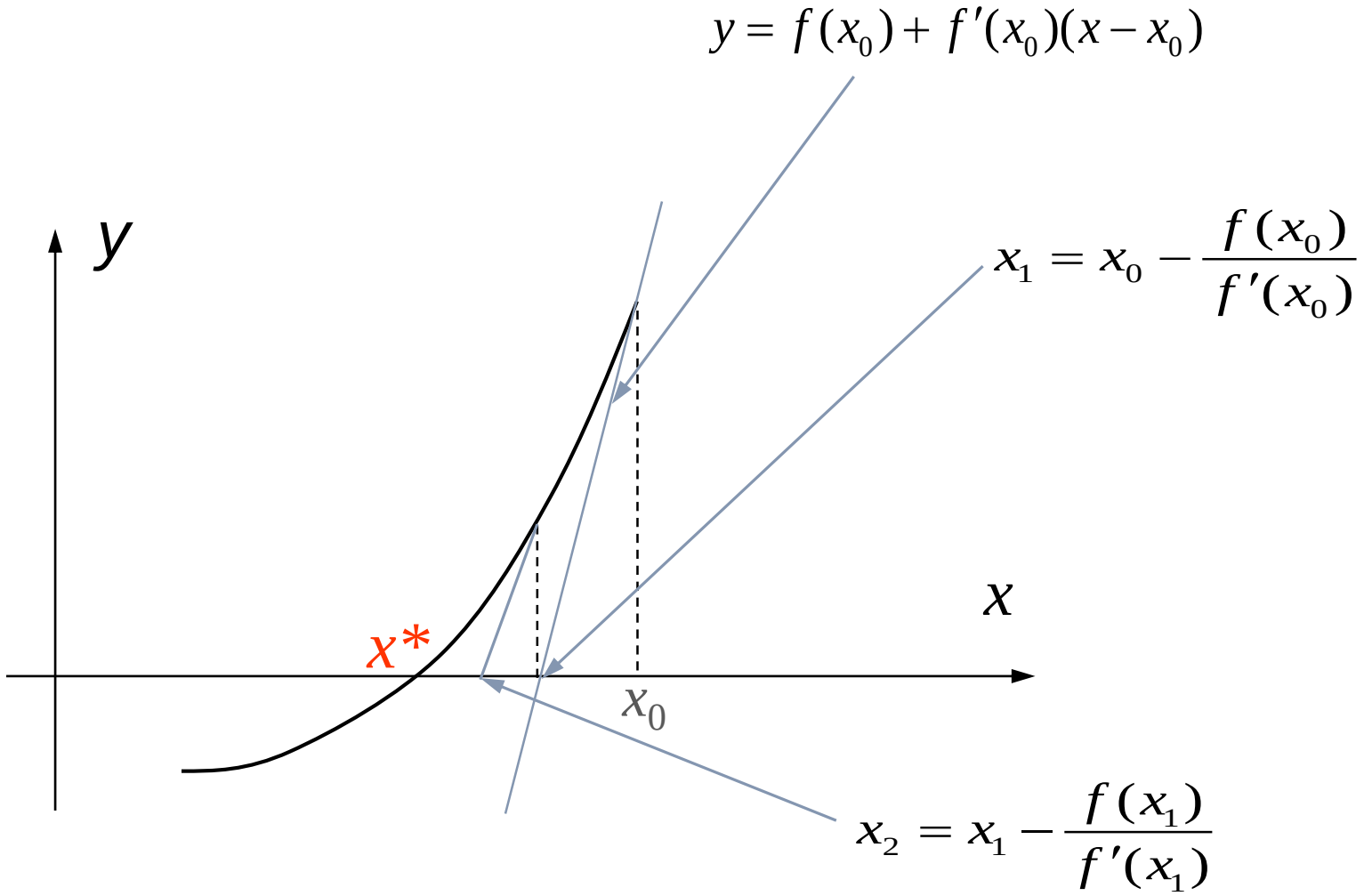
$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

重复上述过程

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

定理 一元方程的牛顿迭代算法局部收敛, 在单根附近具有较高(二阶)的收敛速度.

条件: $f'(x^*) \neq 0$



牛顿法，即切线法

$$\begin{cases} f_1(x_1, x_2, \cdots x_n) = 0 \\ f_2(x_1, x_2, \cdots x_n) = 0 \\ \vdots \\ f_n(x_1, x_2, \cdots x_n) = 0 \end{cases} \Rightarrow F(x) = 0$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \nabla F(\boldsymbol{x}_k)^{-1} F(\boldsymbol{x}_k)$$

$$\nabla F(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

定理 多元恰定方程的牛顿迭代算法局部收敛, 在根附近具有较高(二阶)的收敛速度.

条件：Jacob矩阵非奇异。

$$\nabla F(\boldsymbol{x}) = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_2} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \dfrac{\partial f_2}{\partial x_1} & \dfrac{\partial f_2}{\partial x_2} & \cdots & \dfrac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial f_n}{\partial x_1} & \dfrac{\partial f_n}{\partial x_2} & \cdots & \dfrac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases}$$

系数矩阵可逆

$$Ax = b \longrightarrow x = A^{-1}b$$

两种算法的比较

	最速下降算法	牛顿算法
收敛性	全局收敛 （初始点任意）	局部收敛 （初始点靠近最优值点）

最速下降算法：无论身在何处都能找回家，只是速度慢点，特别到了家门口。

牛顿算法：只能在家附近才能找到家，而且此时能快速回到家，只是累点。