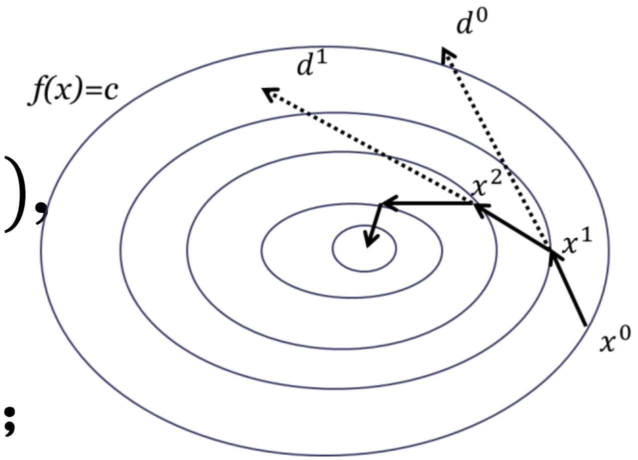
2019

最优化理论与方法

研究生学位课

陈军华（副教授、主任）



- **求解框架**
  - 1、从起点x^0开始，计算$f(x^0),\nabla f(x^0)$, $\nabla^2 f(x^0)$;
  - 2、选择一个从x^0出发的寻优方向 d^0;
  - 3、沿着方向 d^0找到新点（值） x^1，计算$f(x^1),\nabla f(x^1)$, $\nabla^2 f(x^1)$;
  - 4、重复以上步骤，产生新点$x^2, x^3, \ldots$,在每步迭代中取得更优的值。
- **关键问题**
  - 哪个方向？步长？算法能保证到达最优吗？多少步？

- 为什么是梯度方向（最速上升方向）

$f(x)$在x点处的梯度

$$\nabla f(x)=g(x)=\left[\frac{\partial f(x)}{\partial x_1},\frac{\partial f(x)}{\partial x_2},\cdots,\frac{\partial f(x)}{\partial x_n}\right]^T$$

  - 为什么梯度方向是寻优方向中较好的选择呢？

- **为什么是梯度方向**
  - 求$\min f(x), x$为 n维向量， $f(x)$可微。
  - 设x^0为起始搜索点，其邻接某点为 x^0+dx，使得：

$$f\left(x^0+dx\right)<f\left(x^0\right)$$

期望：$\max\{f\left(x^0\right)-f\left(x^0+dx\right)\}$

  - dx沿下降方向的距离增量 ds，可表达为：

$$(ds)^2=\sum_{i=1}^{n}(dx_i)^2$$

- **为什么是梯度方向**
  - 选择 $\frac{dx_i}{ds}$，以使$\frac{df}{ds}$ 最大化。

$$\frac{dx_i}{ds}(x) = -\frac{\frac{\partial f}{\partial x_i}(x)}{[\sum_{i=1}^{n}(\frac{\partial f}{\partial x_i}(x))^2]^{1/2}}$$

目标函数沿负梯度方向，所得变化率为：

$$\frac{df}{ds} = \sum_{i=1}^{n}\frac{\partial f}{\partial x_i}\frac{dx_i}{ds} = -\frac{\sum_{i=1}^{n}\frac{\partial f}{\partial x_i}\frac{\partial f}{\partial x_i}}{\left[\sum_{i=1}^{n}\left(\frac{\partial f}{\partial x_i}(x)\right)^2\right]^{\frac{1}{2}}} = -[\sum_{i=1}^{n}(\frac{\partial f}{\partial x_i})^2]^{1/2} \leq 0$$

  - →沿着负梯度方向 函数值是下降的。

- **为什么是梯度方向**
  - 沿（负）梯度方向在有限步内无法达到最优值。

$$\frac{df}{ds} = -\left[\sum_{i=1}^{n}\left(\frac{\partial f}{\partial x_i}\right)^2\right]^{\frac{1}{2}} \leq 0$$

  - 由于在最优点 x^* 处：$\frac{\partial f}{\partial x_i}(x^*) = 0$

故：$\frac{\partial f}{\partial x_i}(x) \to 0$ 时，$\frac{df}{ds} \to 0$

梯度算法

- $\frac{dx_i}{ds}(x) \quad = -\frac{\frac{\partial f}{\partial x_i}(x)}{[\sum_{i=1}^{n}(\frac{\partial f}{\partial x_i}(x))^2]^{1/2}}$
- $dx_i = -ds\frac{\partial f^n}{\partial x_i(x)}$
- 递推公式：

$$x^{p+1} = x^p + k\nabla f(x^p)$$

- $\nabla f(x^p)$应为归一化后的梯度。

- **例4_1**
  - 求$\boldsymbol{min f(x) = 3x_1} + 4x_2^2$

的梯度算法的递推式 $\boldsymbol{x^{p+1} = x^p + k\nabla f(x^p)}$

- **例4_1**
  - 求$\boldsymbol{minf}(\boldsymbol{x}) = \boldsymbol{3x_1} + \boldsymbol{4}x_2^2$

  的梯度算法的递推式 $\boldsymbol{x^{p+1}} = \boldsymbol{x^p} + \boldsymbol{k\nabla f}(\boldsymbol{x^p})$

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x_1}} = \boldsymbol{3}, \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x_2}} = \boldsymbol{8x_2},$$

$$\boldsymbol{\nabla f}(\boldsymbol{x^p}) = \frac{\boldsymbol{1}}{\sqrt{\boldsymbol{9} + \boldsymbol{64}x_2^2}}\begin{pmatrix} \boldsymbol{3} \\ \boldsymbol{8x_2} \end{pmatrix}|_{\boldsymbol{x_p}}$$

$$\boldsymbol{x^{p+1}} = \begin{pmatrix} x_1^p \\ x_2^p \end{pmatrix} + \boldsymbol{k}\frac{\boldsymbol{1}}{\sqrt{\boldsymbol{9} + \boldsymbol{64}x_2^2}}\begin{pmatrix} \boldsymbol{3} \\ \boldsymbol{8x_2} \end{pmatrix}|_{\boldsymbol{x_p}}$$

$$= \begin{pmatrix} x_1^p + \frac{3k}{\sqrt{\boldsymbol{9+64}x_2^2}} \\ x_2^p + \frac{8kx_2^p}{\sqrt{\boldsymbol{9+64}x_2^2}} \end{pmatrix}$$

- **梯度步长？最优梯度**

  迭代式中 $\boldsymbol{x^{p+1}} = \boldsymbol{x^p} + \boldsymbol{k\nabla f(x^p)}$ k的值如何确定？

要使沿$\boldsymbol{\nabla f(x^p)}$方向推进后函数值最小，则$k$应满足：

$$\min_{\boldsymbol{k}} \boldsymbol{f(x^p + k\nabla f(x^p))}$$

- 上式成立的必要条件为：

$$\boldsymbol{\frac{df}{dk}\left(x^p + k\nabla f(x^p)\right) = 0}$$

- **最速下降法（梯度算法）**
  - 取初始点$x^0 \in E^n$，允许误差$\varepsilon > 0$.
  - 计算负梯度方向 $d^p = -\nabla f(x^p), \overline{d^p} = -\frac{\nabla f(x^p)}{\|\nabla f(x^p)\|}$（可省略）
  - 进行一维搜索$\min_k f(x^p + kd^p)$
  - 迭代： $x^{p+1} = x^p + kd^p$
  - 精度判断为$\|d^p\| \le \varepsilon$

- **例4_2**
  - 求 $\min f(x) = (x_1 - 2)^4 + (x_1 - 2x_2)^2$,
    $x^0 = (0.00,3.00)^T, \varepsilon = 0.1$

- **例4_2**
  - 求$\min f(x)=(x_1-2)^4+(x_1-2x_2)^2$,

$x^0=(0.00,3.00)^T, \varepsilon=0.1$

  - **解:**

$$\nabla f(x)=\begin{pmatrix} 4\left(x_1-2\right)^3+2(x_1-2x_2) \\ -4(x_1-2x_2) \end{pmatrix}$$

$$\nabla f\left(x^0\right)=\begin{pmatrix} -44 \\ 24 \end{pmatrix}$$

$$\left\|\nabla f\left(x^0\right)\right\|=\sqrt{(-44)^2+24^{\,2}}=50.11986\approx 50.12 \text{ (可略去此步)}$$

$$d^0=-\nabla f\left(x^0\right)=\begin{pmatrix} 44 \\ -24 \end{pmatrix}$$

- 例4_2

$$x^1 = x^0 + k^0 d^0 = \begin{pmatrix} 0 \\ 3 \end{pmatrix} + k^0 \begin{pmatrix} 44 \\ -24 \end{pmatrix} = \begin{pmatrix} 44k^0 \\ -24k^0 + 3 \end{pmatrix}$$

$$minf\left(x^1\right) = \left(44k^0 - 2\right)^4 + \left(44k^0 - 2(-24k^0 + 3)\right)^2$$

$$\widehat{k^0} = 0.062$$

$$x^1 = \begin{pmatrix} 44 \times 0.062 \\ -24 \times 0.062 + 3 \end{pmatrix} = \begin{pmatrix} 2.728 \\ 1.512 \end{pmatrix}$$

$$\vdots$$

$$\vdots$$

当进行第九次迭代时，$x^9 = \begin{pmatrix} 2.28 \\ 1.15 \end{pmatrix}$，此时$\left\|d^7\right\| = 0.09 < 0.1$

满足要求。

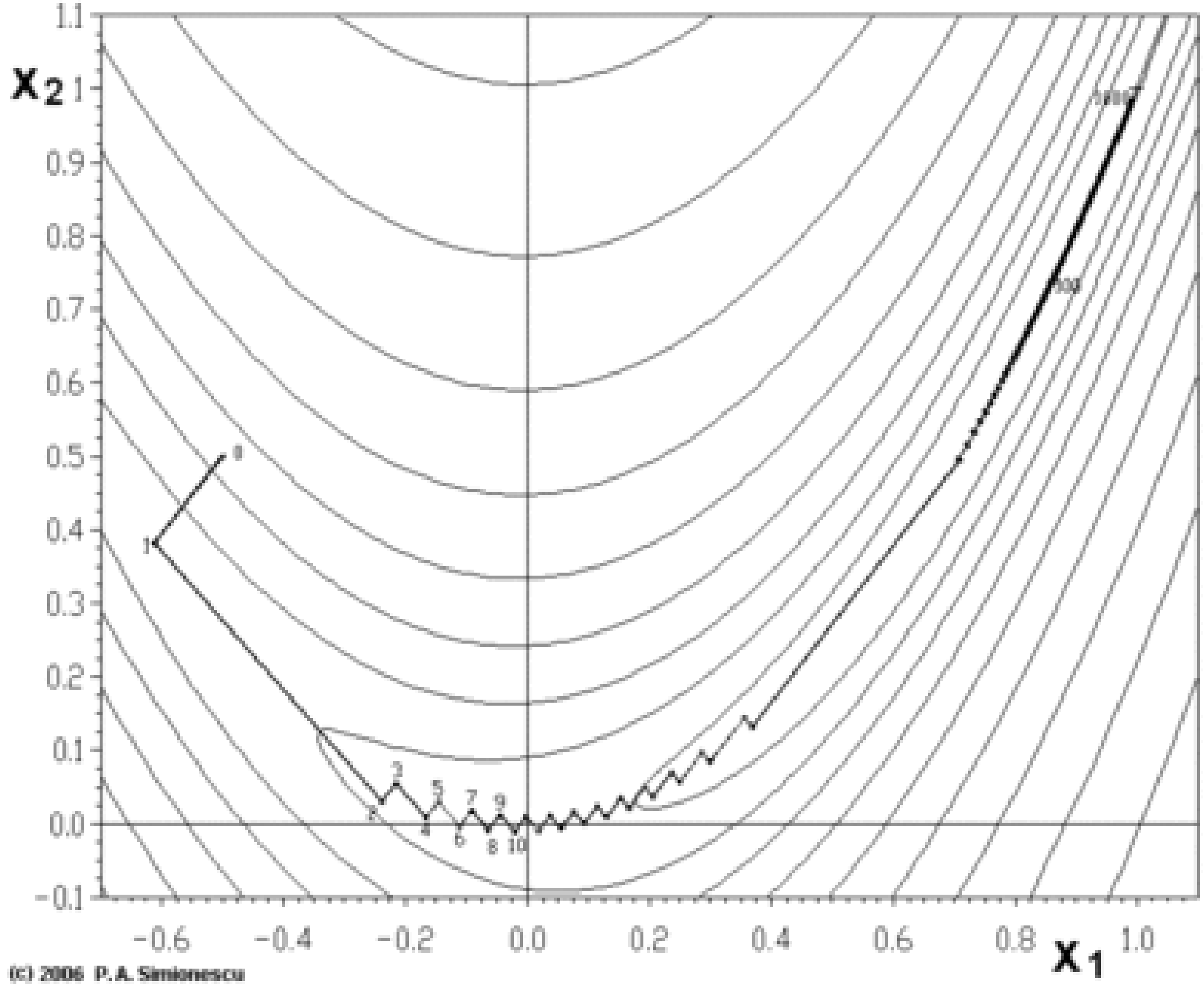
- **最速下降法的算法实现**
  - 几个难点：
  - 1）梯度计算；
  - 2）多维处理；
  - 3）一维搜索区间确定与极值取得。

- **最速下降法的算法的收敛速度**
  - 用程序分别求下面函数的极值点：
  - 1）$\min f(x) = 2(x_1 - 3.5)^2 + 4(x_2 - 4)^2 + 2$
  - 2）$\min f(x) = (1 - x_1)^2 + 100\left(x_2 - x_1^2\right)^2$

Rosenbrock函数

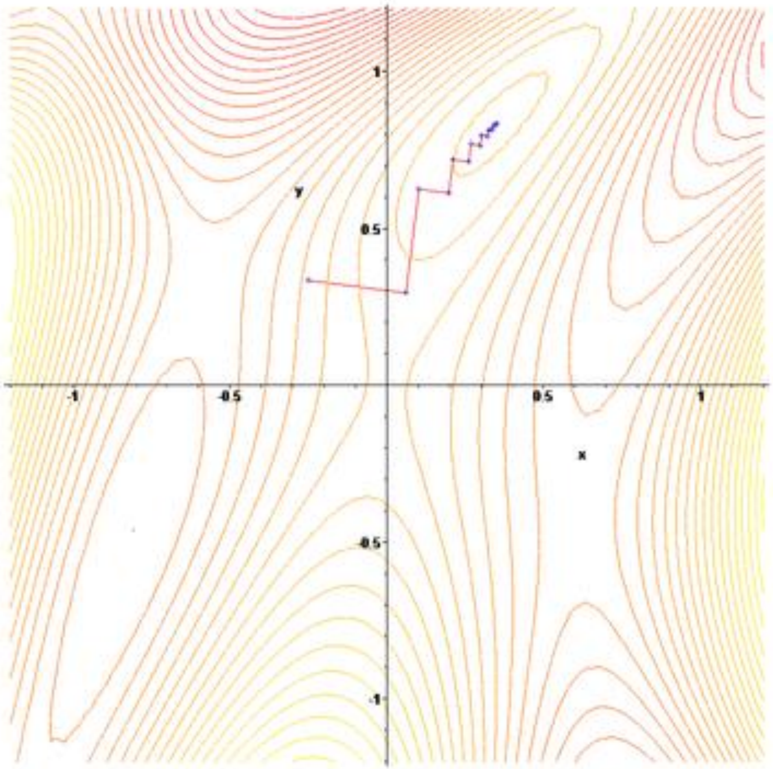
  - 3）$\min f(x) = -\sin\left(\frac{1}{2}x_1^2 - \frac{1}{4}x_2^2 + 3\right) \cos(2x_1 + 1 - e^{x_2})$

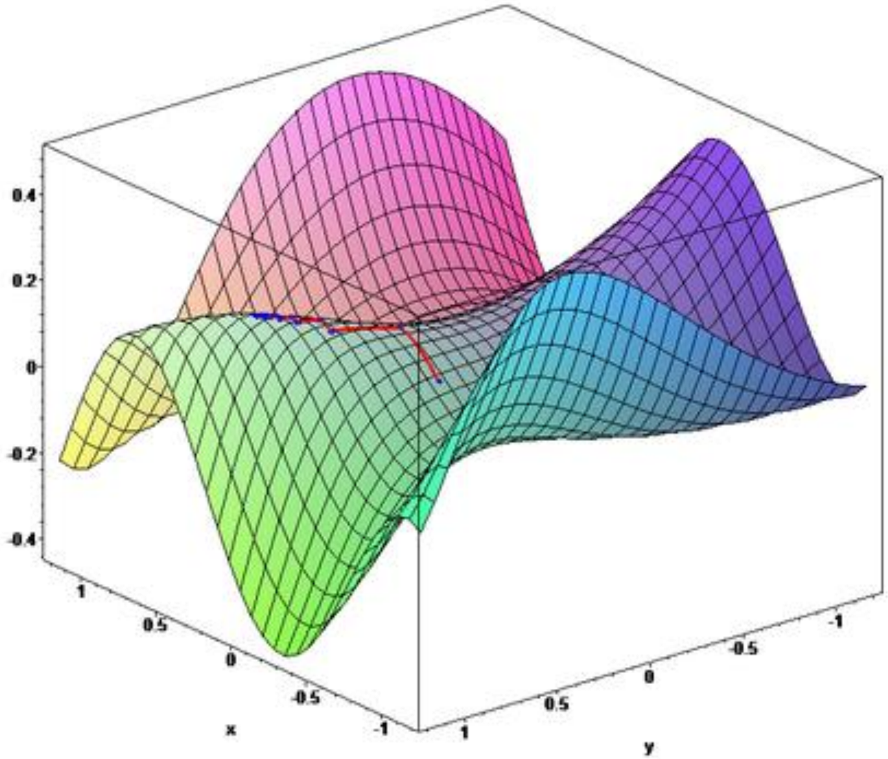
- **最速下降法的算法的收敛速度**
  - 2）$\min f(x)=(1-x_1)^2+100\left(x_2-x_1^2\right)^2$



最速下降法的算法的收敛速度

- 3） $\min f(x) = -\sin\left(\frac{1}{2}x_1^2 - \frac{1}{4}x_2^2 + 3\right)\ \cos(2x_1\ + 1 - e^{x_2}\ )$





- **最速下降法的算法的收敛速度**
  - 实际计算表明：梯度法的收敛速度并不快。一般情况下，若初始点离极点远时，效果较好，到后期接近极点时，尤其当函数的等高线是狭窄长时，收敛速度很慢，迭代的路线往往是锯齿状的。
  - 有没有更好的方法呢？

- **例4_3**
  - 求$f(x) = 4x_1 + 6x_2 - 2x_1^2 - 2x_1x_2 - 2x_2^2$ 的极值, $x^0 = (1,1)^T, \varepsilon = 0.5$

- **例4_3**
  - 求$f(x) = 4x_1 + 6x_2 - 2x_1^2 - 2x_1x_2 - 2x_2^2$的极值, $x^0 = (1,1)^T, \varepsilon = 0.5$
  - $\nabla f(x) = \begin{pmatrix} 4 - 4x_1 - 2x_2 \\ 6 - 2x_1 - 4x_2 \end{pmatrix}$

    $\nabla^2 f(x) = \begin{pmatrix} -4 & -2 \\ -2 & -4 \end{pmatrix}$
  - $AC - B^2 > 0$，且 $A < 0, C < 0$具有极大值。

- **例4_3**
  - 求$f(x) = 4x_1 + 6x_2 - 2x_1^2 - 2x_1x_2 - 2x_2^2$ 的极值, $x^0 = (1,1)^T, \varepsilon = 0.5$
  - 解：令$f(x) = -f(x) = -4x_1 - 6x_2 + 2x_1^2 + 2x_1x_2 + 2x_2^2$
  - $\nabla f(x) = \begin{pmatrix} -4 + 4x_1 + 2x_2 \\ -6 + 2x_1 + 4x_2 \end{pmatrix}$，$\|\nabla f(x_0)\| = 2 > \varepsilon$
  - 1）第一次迭代

$$d^0 = -\nabla f(x_0) = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$$

$$x^1 = x^0 + k^0 d^0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} + k^0 \begin{pmatrix} -2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 - 2k^0 \\ 1 \end{pmatrix}$$

- **例4_3**
  - $g(k) = f(x^1) = -4(1 - 2k^0) - 6 \times 1 + 2 \times (1 - 2k^0)^2 - 2 \times (1 - 2k^0) + 2$

$$\frac{dg(k)}{dk} = -16k^0 + 4$$

$$\widehat{k^0} = 1/4$$

$$x^1 = \begin{pmatrix} 1 - 2 \times \frac{1}{4} \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ 1 \end{pmatrix}$$

$$\nabla f(x_1) = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \|\nabla f(x_0)\| = 1 > \varepsilon$$

- **例4_3**
  - 2）第二次迭代

$$d^1=-\nabla f(x_1)=\begin{pmatrix}0\\1\end{pmatrix}$$

$$x^2=x^1+k^1d^1=\begin{pmatrix}\frac{1}{2}\\1\end{pmatrix}+k^1\begin{pmatrix}0\\1\end{pmatrix}=\begin{pmatrix}\frac{1}{2}\\1+k^1\end{pmatrix}$$

  - $g(k)=f\left(x^2\right)=-4\left(\frac{1}{2}\right)-6\times\left(1+k^1\right)+2\times\left(\frac{1}{2}\right)^2+2\times$

$\left(1+k^1\right)\times\frac{1}{2}+2\times(1+k^1)^2$

$$\frac{dg(k)}{dk}=4k^1-1$$

$$\widehat{k^1}=1/4$$

- **例4_3**

$$\widehat{k^1} = 1/4,\quad d^1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$x^2 = \begin{pmatrix} \frac{1}{2} \\ 1 + \frac{1}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{4}{5} \end{pmatrix}$$

$$\nabla f\left(x^2\right) = \begin{pmatrix} -1/2 \\ 0 \end{pmatrix},\quad \left\|\nabla f\left(x^2\right)\right\| = 0.25 < \varepsilon$$

算法终止

- **梯度法对于二次函数的讨论**
  - 求$\min f(x) = \frac{1}{2}x^T Q x$

- **梯度法对于二次函数的讨论**
  - 求$\min f(x)=\frac{1}{2}x^TQx$
  - $\nabla f(x^p)=Qx^p \qquad x^{p+1}=x^p+kQx^p$
  - 故：

$$f\left(x^{p+1}\right)=\frac{1}{2}(x^p+kQx^p)^TQ(x^p+kQx^p)$$

$$=\frac{1}{2}[x^TQx\ +kx^TQ^2x+kx^TQ^TQx+k^2x^TQ^TQ^2x]_{x^p}$$

$$\frac{df}{dk}=\frac{1}{2}[x^TQ^2x+x^TQ^TQx+2kx^TQ^TQ^2x]_{x^p}=0$$

$$k^*=-[\frac{x^TQ^2x}{x^TQ^3x}]_{x^p}$$

- **例4_4**
  - 求$\min f(x)=\frac{1}{2}(x_1^2+2x_2^2),\ x^0=(4,4)^T$

- **例4_4**
  - 求$\min f(x) = \frac{1}{2}(x_1^2 + 2x_2^2),\ x^0 = (4,4)^T$

$$Q = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$$

$$k^* = -\left[\frac{x^T Q^2 x}{x^T Q^3 x}\right]_{x^p}$$

$$k^* = -\frac{5}{9}$$

$$x^1 = x^0 + k\nabla f(x^0) = \begin{pmatrix} 4 \\ 4 \end{pmatrix} - \frac{5}{9}\begin{pmatrix} 4 \\ 8 \end{pmatrix} = \begin{pmatrix} \frac{16}{9} \\ -\frac{4}{9} \end{pmatrix}$$

求 $(\nabla f(x^0), \nabla f(x^1))$?

- 使用梯度法相邻两步寻优方向互相垂直

- **证明：梯度法相邻两步寻优方向互相垂直**
  - 设x^p和 x^{p+1}为相邻两点，$\nabla f(x^p)$为第 p步的梯度方向

则：$x^{p+1}=x^p+k\nabla f(x^p)$,且满足

$$\frac{df}{dk}\left(x^p+k\nabla f(x^p)\right)=0$$

$$\frac{\partial f[x^p+k\nabla f(x^p)]}{\partial[x^p+k\nabla f(x^p)]}\cdot\frac{\partial\,[x^p+k\nabla f(x^p)]}{\partial k}=0$$

即：

$$\nabla^T f[x^p+k\nabla f(x^p)]\quad\cdot(\nabla f(x^p))$$

$$\nabla^T f(x^{p+1})\quad\cdot(\nabla f(x^p))=0$$

- **作业：**
  - 用梯度法（最速下降法）求解：

$$\min f(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2,\ x^0 = (2,-2,1,-1)^T$$

至少迭代2步。

- **梯度法收敛性分析**
  - 使用最速下降法，收敛性为线性：

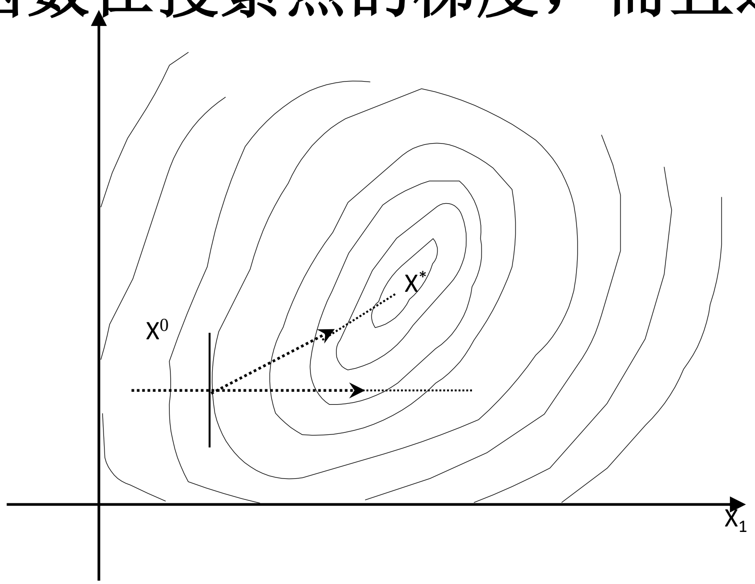
$$\lim_{p\to\infty}\frac{f\left(x^{p+1}\right)-f\left(x^{*}\right)}{f\left(x^{p}\right)-f\left(x^{*}\right)}=a<1$$

求得局部最优，与初始点有关。

迭代次数也与初始点有关，初始点离最优值越接近越好。

有无更快收敛速度的梯度算法呢？

- **二阶导数法（广义牛顿法）**
  - 梯度法的寻优途径，在搜索点，负梯度方向是最优方向，即它是局部最优方向，我们已验证，它并非是全局最优方向。现在我们能否调整一下梯度法的搜索方向，使得它对于全局来说也是最优方向，也就是说从起始搜索点就瞄准极小值点的方向去搜索，其速度不是更快吗？
  - 二阶导数法不仅考虑了目标函数在搜索点的梯度，而且还考虑了梯度的变化趋势



- **二阶导数法（广义牛顿法）**
  - 取初始点$\boldsymbol{x}^0 \in \boldsymbol{E}^n$，允许误差 $\boldsymbol{\varepsilon} > \mathbf{0}$.
  - 计算梯度方向 $\boldsymbol{d}^p = -[\nabla^2 \boldsymbol{f}(\boldsymbol{x}^p)]^{-1}\nabla \boldsymbol{f}(\boldsymbol{x}^p)$
  - 进行一维搜索$\min_{\boldsymbol{k}} \boldsymbol{f}(\boldsymbol{x}^p + \boldsymbol{k}\boldsymbol{d}^p)$
  - $\boldsymbol{x}^{p+1} = \boldsymbol{x}^p + \boldsymbol{k}\boldsymbol{d}^p$
  - 精度判断为$||\boldsymbol{d}^p|| \le \boldsymbol{\varepsilon}$
  - 证明$\boldsymbol{d}^p$为函数下降方向。

- **二阶导数法（广义牛顿法）**
  - 证明：$\boldsymbol{d^p}$为函数下降方向。即证 $\nabla f(x^p)\cdot d^p < 0$
  - $$\nabla^T f(x^p)\cdot d^p = \nabla^T f(x^p)\cdot -[\nabla^2 f(x^p)]^{-1}\nabla f(x^p)$$ $$= -\nabla^T f(x^p)\cdot [\nabla^2 f(x^p)]^{-1}\nabla f(x^p)$$

可知当$[\nabla^2 f(x^p)]^{-1}$正定时，$\nabla^T f(x^p)\cdot d^p < 0$

故只需$f(x)$为凸函数，即为下降方向。

二阶导数法对于二次函数可以一步达优。

- **例4_5 用广义牛顿法求解**

  $\min f(x) = x_1^2 + 25x_2^2,\ x^0 = (2,2)^T$,精度为0.01

- **例4_5**

$$min f(x) = x_1^2 + 25x_2^2, x^0 = (2,2)^T$$,精度为0.01

- 解：$\nabla f(x) = \begin{pmatrix} 2x_1 \\ 50x_2 \end{pmatrix}$

$$\nabla^2 f(x) = \begin{pmatrix} 2 & 0 \\ 0 & 50 \end{pmatrix} > 0$$

$$\nabla f(x^0) = \begin{pmatrix} 4 \\ 100 \end{pmatrix}, [\nabla^2 f(x)]^{-1} = \begin{pmatrix} \frac{1}{2} & 0 \\ 1 & \frac{1}{50} \end{pmatrix}$$

$$\|\nabla f(x^0)\| = 50.04 > 0.01$$

- **例4_5**

$$d^0 = -\left[\nabla^2 f\left(x^0\right)\right]^{-1}\nabla f\left(x^0\right) = -\begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{50} \end{pmatrix}\begin{pmatrix} 4 \\ 100 \end{pmatrix} = -\begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

$$\min_k f\left(x^0 + kd^0\right) = (2-2k)^2 + 25 \times (2-2k)^2 = 26 \times (2-2k)^2$$

$$\frac{df}{dk} = -104(2-2k) = 0 \rightarrow k = 1$$

$$x^1 = x^0 + kd^0 = (2,2)^T + (-2,-2)^T = (0,0)^T$$

$\left\|\nabla f\left(x^1\right)\right\| = 0$，故达最优。

- **例4_6**

$$\boldsymbol{minf(x) = (x_1 - 1)^4 + x_2^2,\ x^0 = (0,1)^T}$$

解 $\nabla f(x) = \begin{bmatrix} 4(x_1-1)^3 \\ 2x_2 \end{bmatrix}$， $\nabla^2 f(x) = \begin{bmatrix} 12(x_1-1)^2 & 0 \\ 0 & 2 \end{bmatrix}$。

第 1 次迭代：

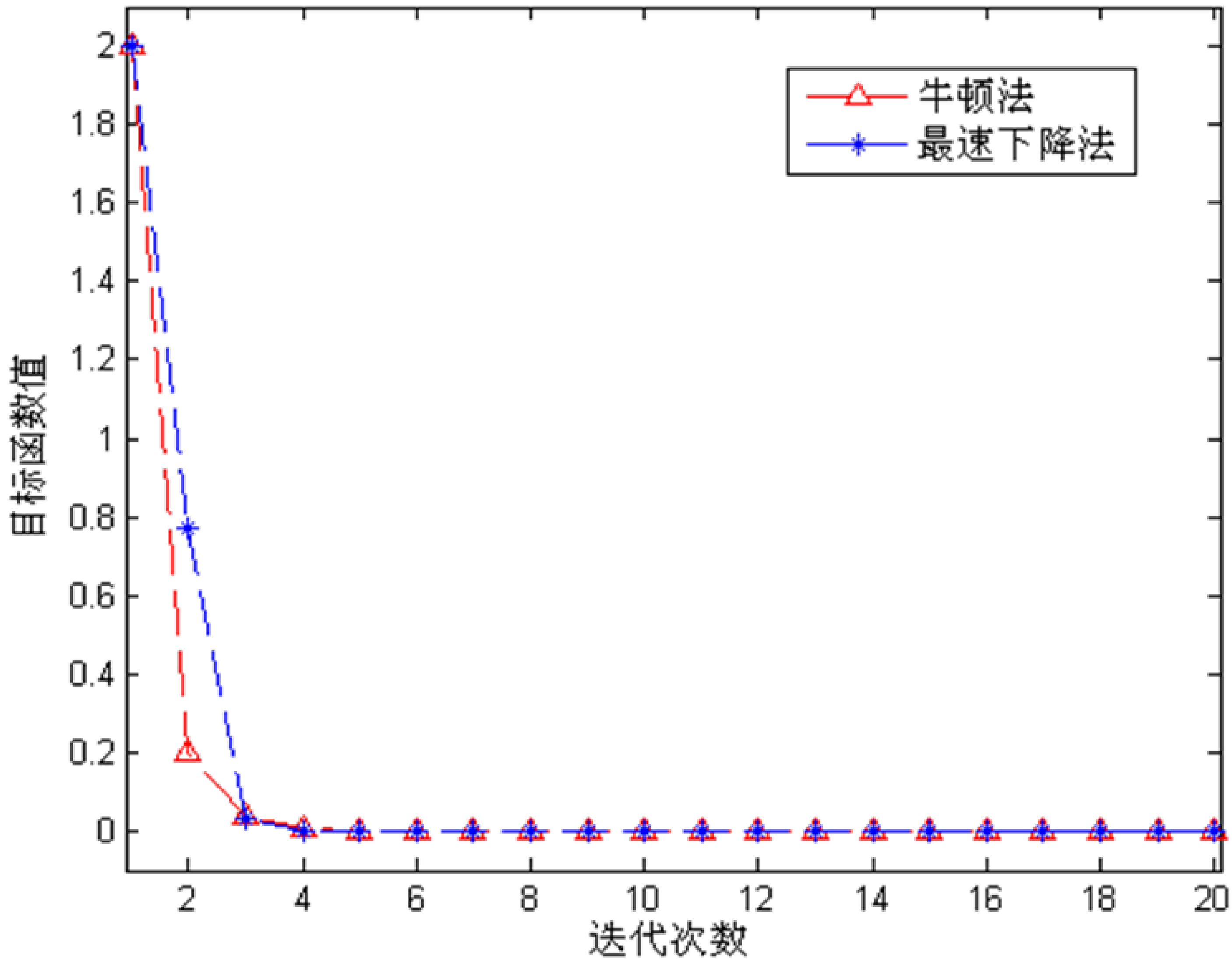
$$g_0 = \begin{bmatrix} -4 \\ 2 \end{bmatrix},\ G_0 = \begin{bmatrix} 12 & 0 \\ 0 & 2 \end{bmatrix},\ x^{(1)} = x^{(0)} - G_0^{-1} g_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 12 & 0 \\ 0 & 2 \end{bmatrix}^{-1} \begin{bmatrix} -4 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} \\ 0 \end{bmatrix}。$$

第 2 次迭代：

$$g_1 = \begin{bmatrix} -\frac{32}{27} \\ 0 \end{bmatrix},\quad G_1 = \begin{bmatrix} \frac{48}{9} & 0 \\ 0 & 2 \end{bmatrix}$$

$$x^{(2)} = x^{(1)} - G_1^{-1} g_1 = \begin{bmatrix} \frac{1}{3} \\ 0 \end{bmatrix} - \begin{bmatrix} \frac{48}{9} & 0 \\ 0 & 2 \end{bmatrix}^{-1} \begin{bmatrix} -\frac{32}{27} \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{5}{9} \\ 0 \end{bmatrix}。$$

- 二阶导数法（广义牛顿法）


牛顿法
最速下降法
目标函数数值
迭代次数
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
2
4
6
8
10
12
14
16
18
20

- 二阶导数法（广义牛顿法）
  - 1)牛顿法是局部收敛的，即初始点选择不当，可能会导致不收敛;
  - 2)牛顿法不是下降算法，当二阶Hesse阵非正定时，不能保证是下降方向;
  - 3)二阶Hesse阵必须可逆，否则算法将无法进行下去;
  - 4)对函数分析性质要求苛刻，计算量大，仅适合小规模优化问题。

  - 该方法需要求二阶导数，以及矩阵的逆，计算速度会很慢。
  - 是否有算法能够既加速收敛，又不会太占计算机资源呢...