西安电子科技大学
XIDIAN UNIVERSITY

IPIU
智能感知与图像理解

最优化理论
第五章：无约束最优化方法

人工智能学院
智能感知与图像理解实验室

无约束最优化方法

1. 最速下降法
2. 最速下降法的步长
3. Amijo’s法则
4. Rosenbrock函数上的例子



无约束最优化方法

1. 最速下降法
2. 最速下降法的步长
3. Amijo's法则
4. Rosenbrock函数上的例子



最速下降法

- 本章开始讨论多维无约束最优化问题

$$\min f(X) \qquad (5.1)$$

其中f: $R^n \to R^1$．这个问题的求解是指在R^n中找一点X^*，使得对于任意的$X \in R^n$ 都有

$$f(X^*) \le f(X) \qquad (5.2)$$

成立，则点 X^*就是问题（5. 1）的全局最优点



最速下降法

•无约束优化方法是优化技术中极为重要和基本的内容之一．它不仅可以直接用来求解无约束优化问题，而且很多约束优化问题也常将其转化为无约束优化问题，然后用无约束优化方法来求解.

•另外，有些无约束优化方法只需略加处理，即可用于求解约束优化问题.



最速下降法

- 无约束优化理论发展较早，比较成熟，方法也很多，新的方法还在陆续出现．把这些方法归纳起来可以分成两大类：一类是仅用计算函数值所得到的信息来确定搜索方向，通常称它为直接搜索法，简称为直接法，另一类需要计算函数的一阶或二阶导数值所得到的信息来确定搜索方向，这一类方法称为间接法（解析法）．
- 直接法不涉及导数、Hesse矩阵，适应性强，但收敛速度较慢；间接法收敛速度快，但需计算梯度，甚至需要计算Hesse矩阵.
- 一般的经验是,在可能求得目标函数导数的情况下还是尽可能使用间接方法；相反,在不可能求得目标函数的导数或根本不存在导数的情况下,当然就应该使用直接法.



最速下降法

- 对于问题（5.1）为了求其最优解，按最优化算法的基本思想是从一个给定的初始点 x_0 出发，通过基本迭代格式 $X_{k+1} = X_k + t_k P_k$ ，按照特定的算法产生一串点列 $\{X_k\}$，如果点列收敛，则该点列的极限点为问题（5.1）的最优解.



最速下降法

问题：在点X_k处，沿什么方向p_k，$f(x)$下降最快？



最速下降法

问题： 在点X_k处，沿什么方向P_k，$f(x)$下降最快？

分析：$f(X_k+\lambda p_k)=f(X_k)+\lambda g_k^T p_k+o(\lambda\|p_k\|),(\lambda>0)$

考查：$g_k^T p_k=\|g_k\|\|p_k\|\cos\theta$

显然当 $\cos\theta=-1$时，$g_k^T p_k$ 取极小值.

因此： $p_k=-g_k$

结论：负梯度方向使 $f(x)$ 下降最快，亦即最速下降方向.



最速下降法

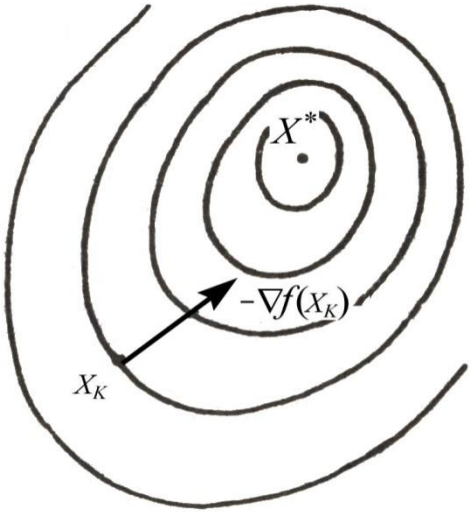
一、最速下降法基本原理

在基本迭代格式 $X_{k+1}=X_k+t_kP_k$ 中，每次迭代搜索方向 P_k 取为目标函数 $f(X)$ 的负梯度方向，即 $P_k=-\nabla f(X_k)$，而每次迭代的步长 t_k 取为最优步长，由此所确定的算法称为**最速下降法**.



最速下降法

•为了求解问题（5.1），如图所示，假定我们 已经迭代了次k获得了第k个迭代点 X_k ．现在从 X_k 出发，可选择的下降方向很多，一个非常自然的想法是沿最速下降方向（即负梯度方向）进行搜索应该是有利的，至少在 X_k 邻近的范围内是这样。因此，取搜索方向为 $P_k = -\nabla f(X_k)$



最速下降法

为了使目标函数在搜索方向上获得最多的下降，沿 P_k 进行一维搜索，由此得到第 $k+1$ 个迭代点 X_{k+1} 即，

$$X_{k+1} = X_k - t_k \nabla f(X_k)$$

其中步长因子 t_k 按下式确定

$$t_k = \arg\min_t f(X_k - t\nabla f(X_k))$$

也可记为 $$X_{k+1} = ls(X_k, -\nabla f(X_k)) \qquad (5.3)$$



最速下降法

显然,令 $k=0,1,2,\cdots$ 就可以得到一个点列 $X_0, X_1, X_2\cdots$，其中 X_0 是初始点,由计算者任意选定. 当 $f(X)$ 满足一定的条件时,由式(5.3)所产生的点列 $\{X_k\}$ 必收敛于 *f(x)* 的极小点 X^*.

以后为书写方便, 记 $g(X)=\nabla f(X)$. 因此 $g(X_k)=\nabla f(X_k)$ 在不发生混淆时，再记 $g_k=g(X_k)=\nabla f(X_k)$.



最速下降法

- **二、最速下降法迭代步骤**

已知目标函数 $f(X)$ 及其梯度 $g(X)$ ， 终止限 ε

- （1）选定初始点 X_0 ， 计算 $f_0 = f(X_0), g_0 = g(X_0).$ 置 $k = 0$.
- （2）作直线搜索： $X_{k+1} = ls(X_k, -g_k)$ ； 计算

$$f_{k+1} = f(X_{k+1}),\ g_{k+1} = g(X_{k+1})$$

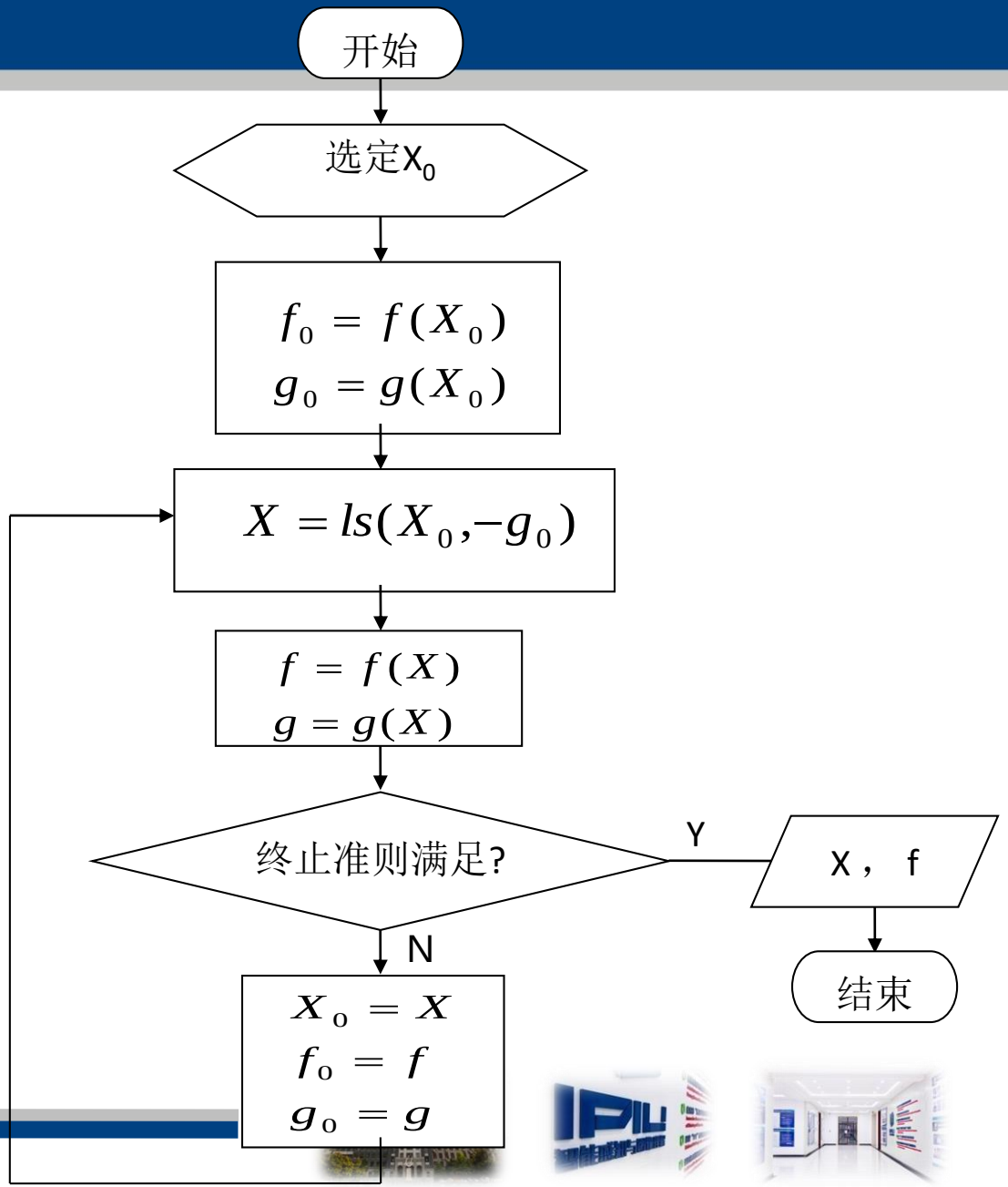
- （3）检测终止准则 $\|\nabla f(X_{k+1})\| \le \varepsilon$ 是否满足：

若满足，则打印最优解 $X_{k+1}, f(X_{k+1})$, 停止迭代；

否则，置 $k = k+1$ 转(2).



最速下降法算法流程如图所示.



无约束最优化方法

1. 最速下降法
2. 最速下降法的步长
3. Amijo's法则
4. Rosenbrock函数上的例子



最速下降法的步长

将最速下降法应用于正定二次函数

$$f(X)=\frac{1}{2}X^TAX+b^TX+c \qquad (5.4)$$

可以推出显式迭代公式. 设 k 第次迭代点为 X_k, 我们来求 X_{k+1} 的表达式.对式（5.4）关于X求梯度，有

$$g(X)=AX+b \qquad (5.5)$$

因此，

$$g_k=g(X_k)=AX_k+b \qquad (5.6)$$



最速下降法的步长

现在从 X_k 出发沿 $-g_k$ 作直线搜索以确定 X_{k+1}，于是，

$$X_{k+1} = X_k - t_k g_k \qquad (5.7)$$

其中 t_k 是最优步长因子.



最速下降法的步长

回忆第四章一维搜索算法中，若从 X_k 出发，沿 P_k 方向进步一维搜索得极小点 $X_{k+1}=X_k+t_kP_k$，则该点 $X=X_{k+1}$ 处的梯度方向 $\nabla f(X_{k+1})$ 与搜索方向之间应满足

$$\nabla f(X_{k+1})^T P_k=0 \qquad (4.2)$$

再利用(5.6),(5.7)可得：

$$[A(X_k-t_kg_k)+b]^T g_k=0$$

或

$$[g_k-t_kAg_k]^T g_k=0$$



最速下降法的步长

由此解出:

$$t_k = \frac{g_k^T g_k}{g_k^T A g_k}$$

带入(5.7)中得到

$$X_{k+1} = X_k - \frac{g_k^T g_k}{g_k^T A g_k} g_k \qquad (5.8)$$

这就是最速下降法用于二次函数的显式迭代公式.



最速下降法的步长

课堂练习:

求 $\min\ f(x)=x_1^2-2x_1+x_2^2+4x_2+5$，取 $x_0=(0,0)^T,\varepsilon=0.1.$



最速下降法的步长

课堂练习:

求 $\min\ f(x)=x_1^2-2x_1+x_2^2+4x_2+5$，取 $X_0=(0,0)^T,\varepsilon=0.1.$

解: $\nabla f(x)=\begin{bmatrix}2x_1-2\\2x_2+4\end{bmatrix},A=\begin{bmatrix}2&0\\0&2\end{bmatrix}$

$$g_0=\nabla f(X_0)=\begin{bmatrix}-2\\4\end{bmatrix},\|g_0\|=\sqrt{20}\geq\varepsilon$$



最速下降法的步长

课堂练习：求 $\min\ f(x)=x_1^2-2x_1+x_2^2+4x_2+5$，取 $X_0=(0,0)^T,\varepsilon=0.1.$

$$步长 t_0=\frac{g_0^T g_0}{g_0^T A g_0}=\frac{1}{2}$$

$$X_1=X_0-\frac{1}{2}g_0=\begin{bmatrix}0\\0\end{bmatrix}-\frac{1}{2}\begin{bmatrix}-2\\4\end{bmatrix}=\begin{bmatrix}1\\-2\end{bmatrix}$$



最速下降法的步长

课堂练习：求 $\min\ f(x)=x_1^2-2x_1+x_2^2+4x_2+5$，取 $X_0=(0,0)^T,\varepsilon=0.1.$

$$\nabla f(x)=\begin{bmatrix}2x_1-2\\2x_2+4\end{bmatrix},A=\begin{bmatrix}2&0\\0&2\end{bmatrix}$$

$$X_1=\begin{bmatrix}1\\-2\end{bmatrix}$$

$$g_1=\nabla f(X_1)=\begin{bmatrix}0\\0\end{bmatrix},\|g_1\|=0<\varepsilon$$

$\begin{bmatrix}1\\-2\end{bmatrix}$ 为最优解



最速下降法的步长

例5.1 试用最速下降法求函数 $f(x_1, x_2) = x_1^2 + 4x_2^2$ 的极小点. 迭代两次，计算各迭代点的函数值，梯度及其模，并验证相邻两个搜索方向是正交的. 设初始点为$X_0 = [1,1]^T$.

解: 与（5.4）比较，得

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 8 \end{bmatrix}$$

梯度表达式是

$$g_0 = \nabla f(X_0) = \begin{bmatrix} 2 \\ 8 \end{bmatrix}$$



最速下降法的步长

- 由 $X_0=\begin{bmatrix}1\\1\end{bmatrix}$，计算

$$f(x_0)=1^2+4\times1^2=5$$

$$g_0=\nabla f(X_0)=\begin{bmatrix}2\\8\end{bmatrix}$$

$$\|g_0\|=8.24621$$

- 因为目标函数是二次的，可以使用式(5.8),所以有

$$X_1=X_0-\frac{g_0^Tg_0}{g_0^TAg_0}g_0=\begin{bmatrix}1\\1\end{bmatrix}-0.13077\begin{bmatrix}2\\8\end{bmatrix}=\begin{bmatrix}0.73846\\-0.04616\end{bmatrix}$$



最速下降法的步长

- 计算

$$f(X_1)=0.73846^2+4\times0.04616^2=0.55385,$$

$$g_1=\nabla f(X_1)=\begin{bmatrix}1.47692\\-0.36923\end{bmatrix},\qquad \|g_1\|=1.52237.$$

$$X_2=X_1-\frac{g_1^Tg_1}{g_1^TAg_1}g_1=\begin{bmatrix}0.73846\\-0.04616\end{bmatrix}-0.42500\begin{bmatrix}1.47692\\-0.36923\end{bmatrix}=\begin{bmatrix}0.11076\\0.11076\end{bmatrix},$$

$$f(X_2)=0.06134,\quad g_2=\nabla f(X_2)=\begin{bmatrix}0.22152\\0.88008\end{bmatrix},\|g_2\|=0.91335.$$



最速下降法的步长

- 因为 $g_1^T g_0 = 0.0000,\ g_2^T g_1 = 0.0000,$
- 由此说明相邻两个搜索方向

  $P_1 = -g_1$ 与 $P_0 = -g_0$，$P_2 = -g_2$ 与 $P_1 = -g_1$ 是正交的.



最速下降法

三、最速下降法有关说明

- 最速下降法的优点是算法简单，每次迭代计算量小，占用内存量小，即使从一个不好的初始点出发，往往也能收敛到局部极小点.但它有一个严重缺点就是收敛速度慢.
- 沿负梯度方向函数值下降很快的说法，容易使人们产生一种错觉，认为这一定是最理想的搜索方向，沿该方向搜索时收敛速度应该很快，然而事实证明，梯度法的收敛速度并不快.



最速下降法

• 特别是对于等值线（面）具有狭长深谷形状的函数，收敛速度更慢．其原因是由于每次迭代后下一次搜索方向总是与前一次搜索方向相互垂直，如此继续下去就产生所谓的锯齿现象.

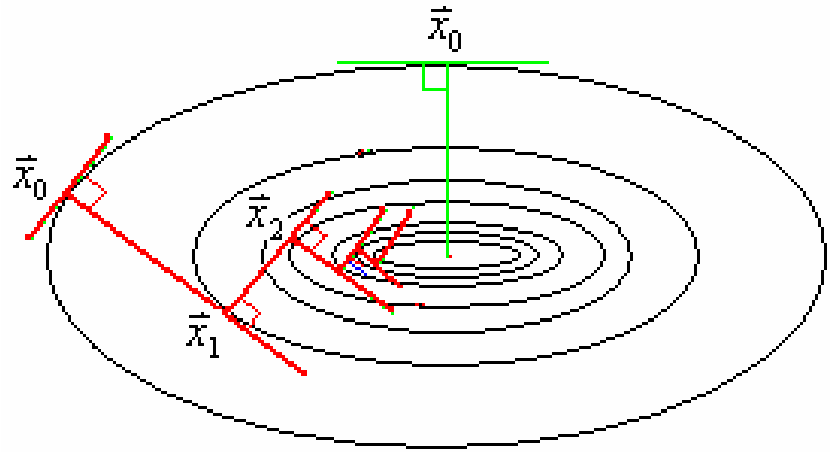
•即从直观上看，在远离极小点的地方每次迭代可能使目标函数有较大的下降，但是在接近极小点的地方，由于锯齿现象，从而导致每次迭代行进距离缩短，因而收敛速度不快.



最速下降法

锯齿现象

最速下降法的迭代点在向极小点靠近的过程中，走的是曲折的路线：后一次搜索方向 $\vec{p}_{k+1}$ 与前一次搜索方向 $\vec{p}_k$ 总是相互垂直的，称它为锯齿现象。除极特殊的目标函数（如等值面为球面的函数）和极特殊的初始点外，这种现象一般都要发生。





最速下降法

1 最速下降法

2 最速下降法的步长

3 Amijo's法则

4 Rosenbrock函数上的例子



Amijo's法则

Amijo's法则

$$f(\mathbf{x}_k + t_k P_k) \le f(\mathbf{x}_k) + t_k \rho \nabla f(\mathbf{x}_k)^T P_k \quad (1)$$

$$f(\mathbf{x}_k + t_k P_k) \ge f(\mathbf{x}_k) + t_k (1-\rho) \nabla f(\mathbf{x}_k)^T P_k \quad (2)$$

where, $\rho \in (0, \frac{1}{2})$

[1] Bazaraa, Mokhtar S., Hanif D. Sherali, and Chitharanjan M. Shetty. Nonlinear Programming: Theory and Algorithms. John Wiley & Sons, 2013.



Amijo's法则

法则一: 目标函数应该有足够的下降

$$\nabla f(\mathbf{x}_k)^T P_k < 0, \text{如果} P_k \text{为下降方向}$$

$$f(\mathbf{x}_k + t_k P_k) \le f(\mathbf{x}_k) + t_k \rho \nabla f(\mathbf{x}_k)^T P_k \le f(\mathbf{x}_k)$$



Amijo's法则

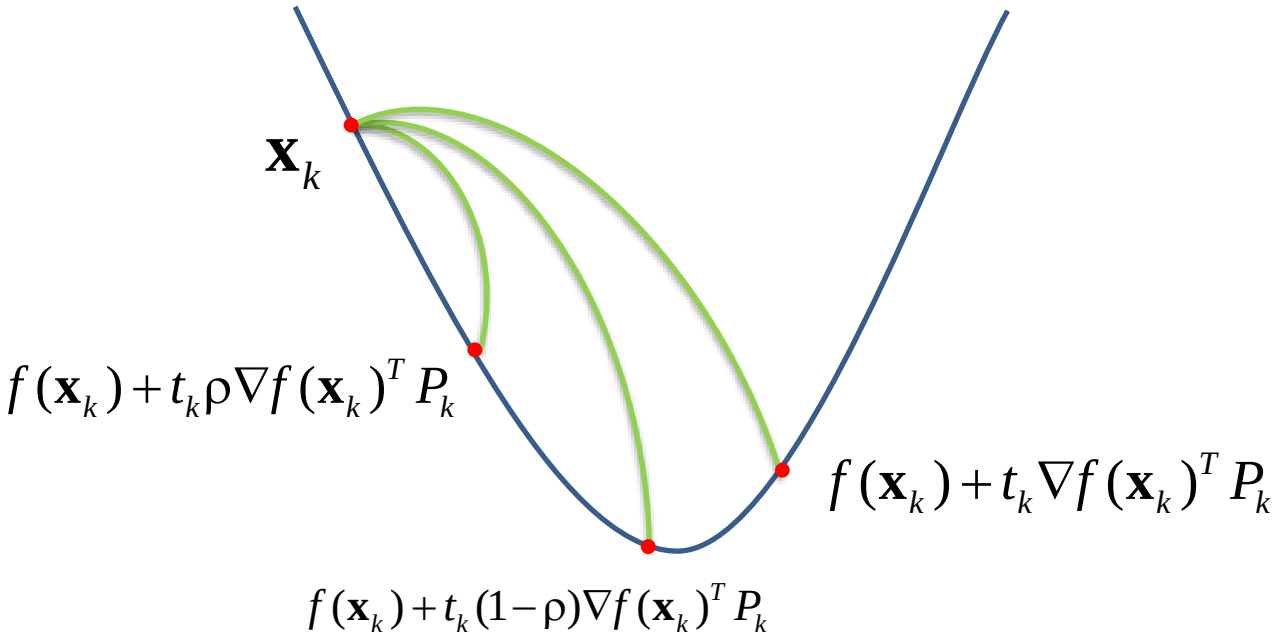
法则二: 步长应该足够大

$$f(\mathbf{x}_k + t_k P_k) = f(\mathbf{x}_k) + t_k \nabla f(\mathbf{x}_k)^T P_k + O(t_k)$$
$$\geq f(\mathbf{x}_k) + t_k (1-\rho) \nabla f(\mathbf{x}_k)^T P_k$$



Amijo’s法则

Amijo’s法则理解



无约束最优化方法

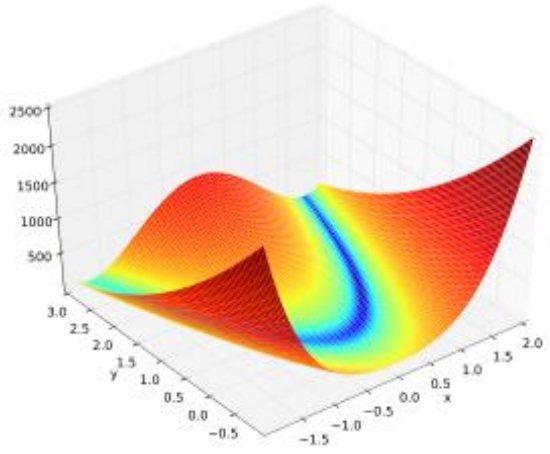
1. 最速下降法
2. 最速下降法的步长
3. Amijo's法则
4. Rosenbrock函数上的例子



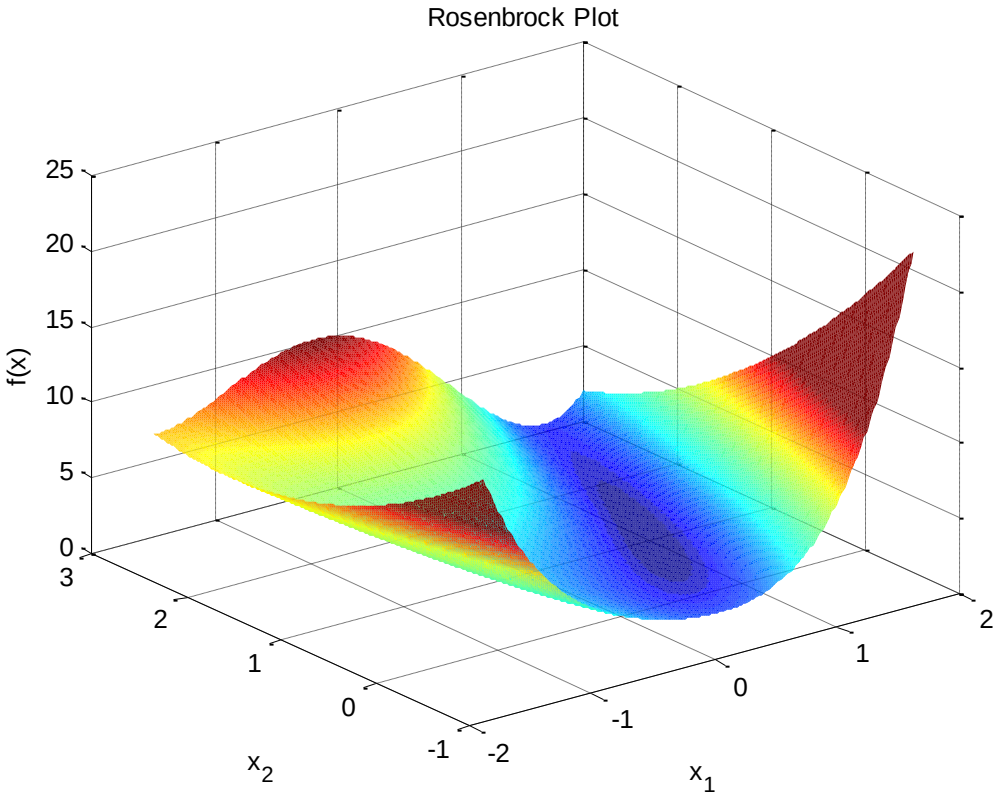
Rosenbrock函数上的例子

Rosenbrock Function[2]

$$f(x) = (x_1^2 - x_2)^2 + (1 - x_1)^2$$



Rosenbrock Valley



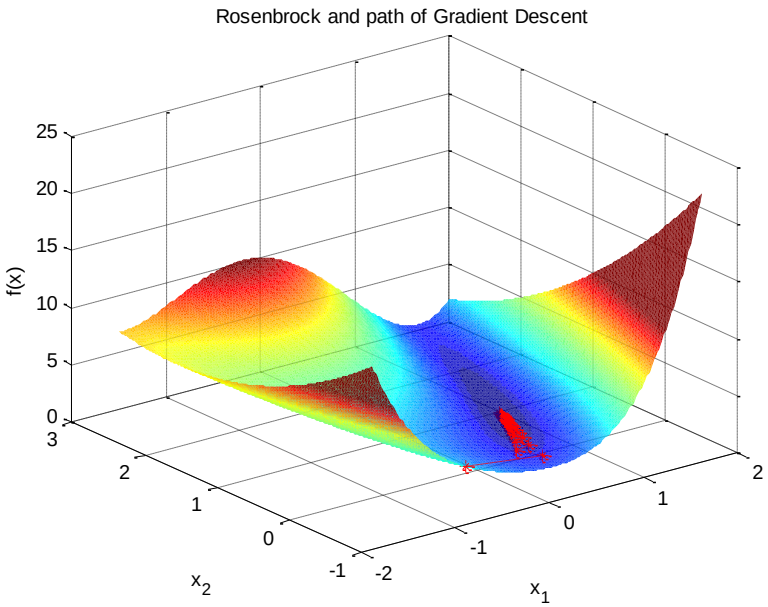
Plot of Rosenbrock

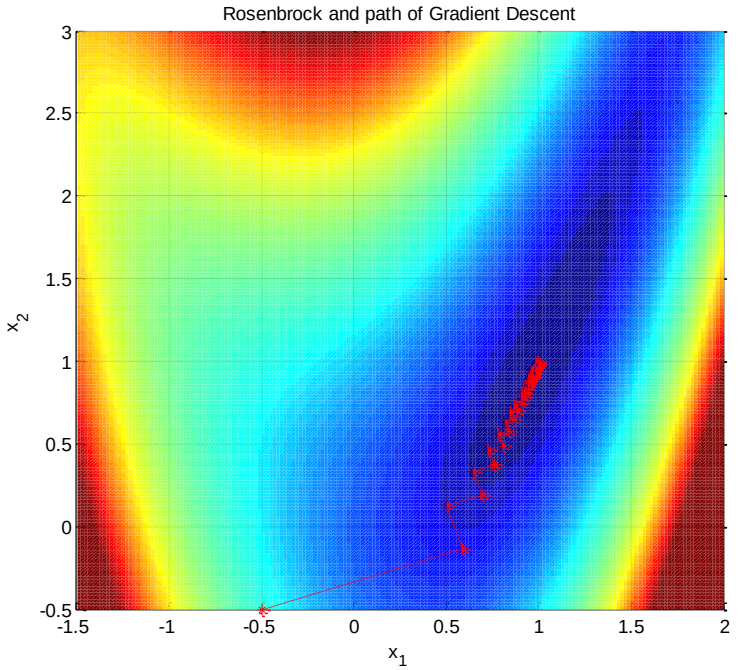
[2] Rosenbrock, H.H. (1960). "An automatic method for finding the greatest or least value of a function". The Computer Journal 3: 175–184.



Rosenbrock函数上的例子

最速下降法





Rosenbrock函数上的例子

最速下降法

Command Window

```
Iteration=1;current_f_value=2.8125;x_1=-0.5;x_2=-0.5;
Iteration=2;current_f_value=0.4031;x_1=0.59125;x_2=-0.13625;
Iteration=3;current_f_value=0.26404;x_1=0.50192;x_2=0.12561;
Iteration=4;current_f_value=0.17813;x_1=0.69462;x_2=0.19117;
Iteration=5;current_f_value=0.1323;x_1=0.64604;x_2=0.33362;
Iteration=6;current_f_value=0.099682;x_1=0.75933;x_2=0.37223;
Iteration=7;current_f_value=0.078308;x_1=0.72707;x_2=0.46685;
Iteration=8;current_f_value=0.062012;x_1=0.80452;x_2=0.49298;
Iteration=9;current_f_value=0.050283;x_1=0.78105;x_2=0.56163;
Iteration=10;current_f_value=0.040977;x_1=0.83795;x_2=0.58085;
Iteration=11;current_f_value=0.033903;x_1=0.82008;x_2=0.63338;
```

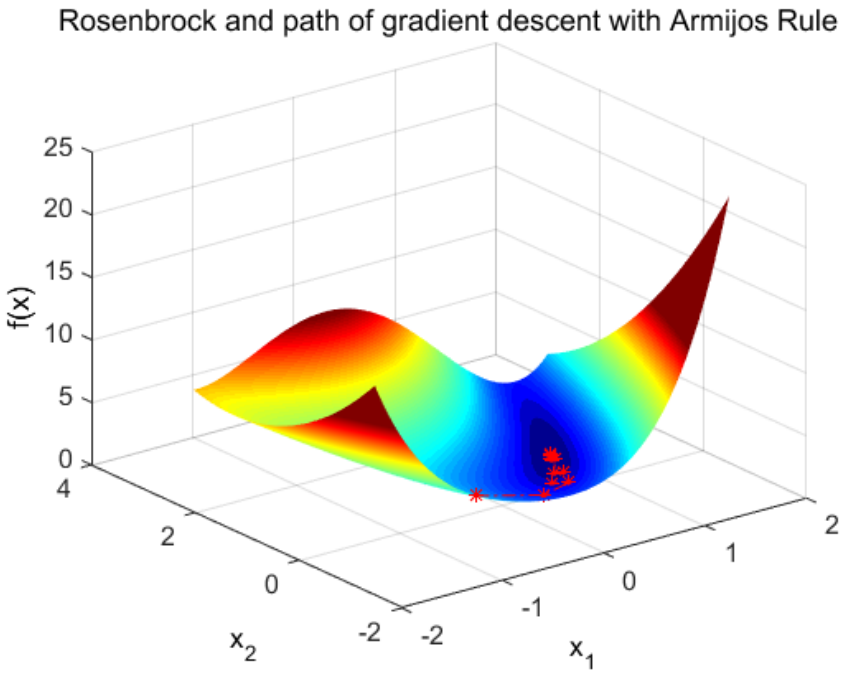
…… ……

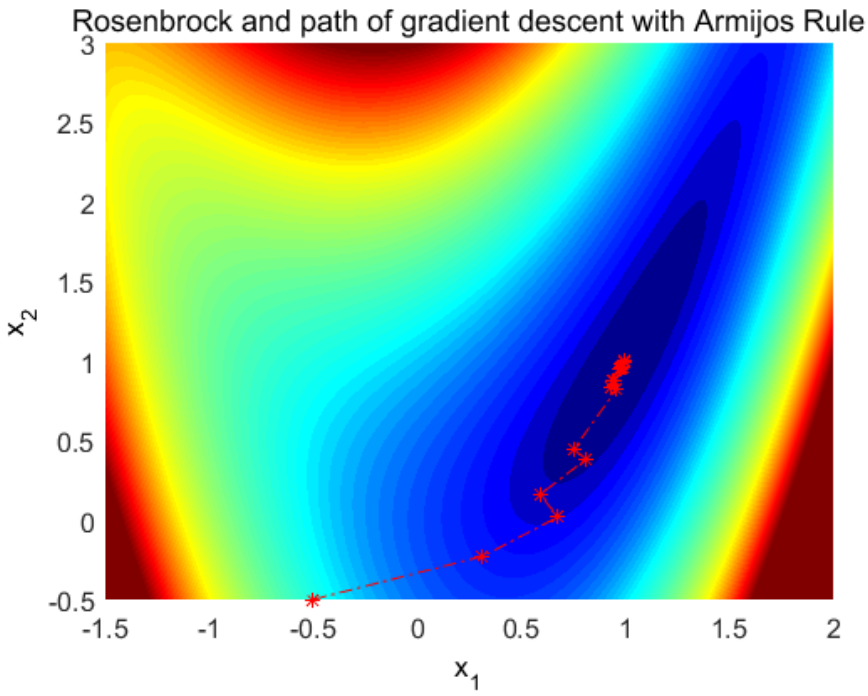
```
Iteration=115;current_f_value=6.6655e-08;x_1=0.99975;x_2=0.99944;
Iteration=116;current_f_value=5.9235e-08;x_1=0.99979;x_2=0.99946;
Iteration=117;current_f_value=5.2644e-08;x_1=0.99978;x_2=0.9995;
Iteration=118;current_f_value=4.6787e-08;x_1=0.99982;x_2=0.99952;
Iteration=119;current_f_value=4.1583e-08;x_1=0.9998;x_2=0.99956;
Iteration=120;current_f_value=3.6956e-08;x_1=0.99984;x_2=0.99957;
Iteration=121;current_f_value=3.2844e-08;x_1=0.99982;x_2=0.99961;
Iteration=122;current_f_value=2.9188e-08;x_1=0.99986;x_2=0.99962;
```



Rosenbrock函数上的例子

Gradient descent with Armijo's Rule

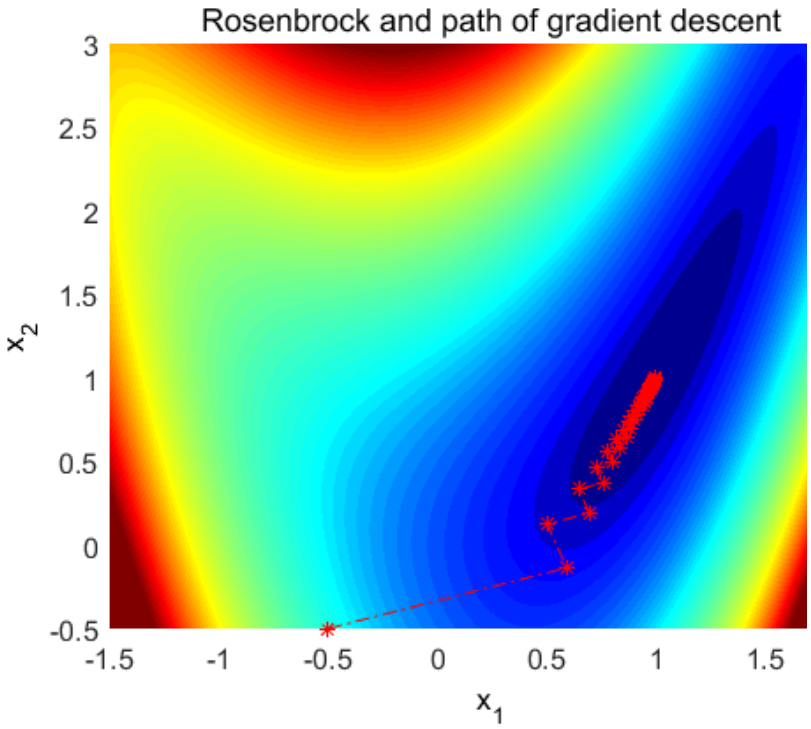




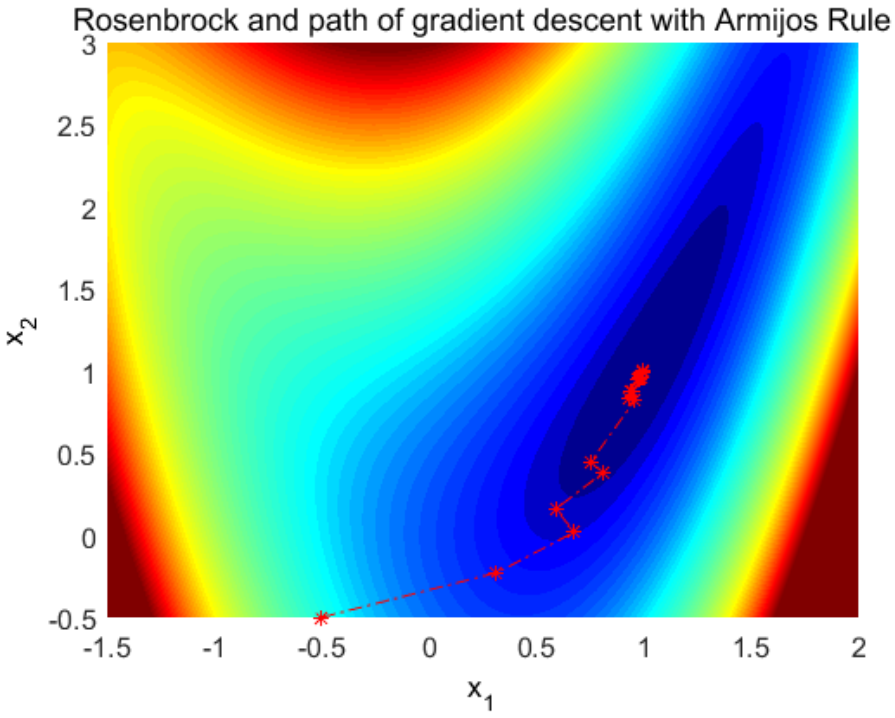


Amijo's法则

Comparison



Gradient



Armijo's Rule



西安电子科技大学
XIDIAN UNIVERSITY

IPIU
智能感知与图像理解

Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education

THE END

Thanks for your participation!