优化模型与软件工具

主讲教师：董庆兴

华中师范大学 信息管理学院
qxdong@mail.ccnu.edu.cn

2017 年 11 月 30 日

大纲

下降迭代算法步骤

1. 给出初始点 $\mathbf{x}^{(0)}$，$k \leftarrow 0$
2. 判断 $\mathbf{x}^{(k)}$ 是否为极小点或者近似极小点，是则停止，否则转第 3 步
3. 按照某种规则确定搜索方向 $\mathbf{p}^{(k)}$
4. 按照某种规则确定搜索步长 λ_k, 得到 $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \lambda_k \mathbf{p}^{(k)}$，使得 $f(\mathbf{x}^{(k)} + \lambda_k \mathbf{p}^{(k)}) < f(\mathbf{x}^{(k)})$，转第 2 步

由上述步骤可知，确定搜索方向和搜索步长是非常关键的步骤，遵循不同的规则，就可以得到不同的算法

线性搜索

- 确定步长的一种符合直觉的做法是求使得目标函数值下降最多的 λ_k，也就是 $\lambda_k = \arg\min f(\mathbf{x}^{(k)} + \lambda \mathbf{p}^{(k)})$
- 由于这一过程是求解以 λ 为变量的一元函数 $f(\mathbf{x}^{(k)} + \lambda \mathbf{p}^{(k)})$ 的极小点 λ_k, 因此本方法被称作线性搜索或者一维搜索，这样确定的步长即为最优步长
- 线性搜索性质：在搜索方向上所得最优点处的梯度与搜索方向正交，即 $\nabla f(\mathbf{x}^{(k+1)})^{\mathrm{T}} \mathbf{p}^{(k)} = 0$

证明: 构造函数 $\phi(\lambda) = f(\mathbf{x}^{(k)} + \lambda \mathbf{p}^{(k)})$ ，从而由最优解的一阶条件可得 $\phi'(\lambda) = \nabla f(\mathbf{x}^{(k)} + \lambda \mathbf{p}^{(k)})^{\mathrm{T}} \mathbf{p}^{(k)} = \nabla f(\mathbf{x}^{(k+1)})^{\mathrm{T}} \mathbf{p}^{(k)} = 0$ ∎

终止条件

- 根据相继两次迭代的

  绝对误差

$$\left\|\mathbf{x}^{(k+1)}-\mathbf{x}^{(k)}\right\|<\epsilon_1 \tag{1}$$

$$\left\|f(\mathbf{x}^{(k+1)})-f(\mathbf{x}^{(k)})\right\|<\epsilon_2 \tag{2}$$

  相对误差

$$\frac{\|\mathbf{x}^{(k+1)}-\mathbf{x}^{(k)}\|}{\|\mathbf{x}^{(k)}\|}<\epsilon_3 \tag{3}$$

$$\frac{\|f(\mathbf{x}^{(k+1)})-f(\mathbf{x}^{(k)})\|}{\|f(\mathbf{x}^{(k)})\|}<\epsilon_4 \tag{4}$$

- 另一种则是根据一阶最优性条件，要求目标函数梯度的模足够小

$$\|\nabla f(\mathbf{x}^{(k)})\|\leq\epsilon_5 \tag{5}$$

收敛速度

- 一个好的算法，不仅仅要求它产生的点列能够收敛到问题的最优解，还要求快速收敛到最优解
- 设序列 $\{\mathbf{x}^{(k)}\}$ 收敛于 $\mathbf{x}^*$，若存在与迭代次数 k 无关的数 $0 < \beta < \infty$ 和 $\alpha \geq 1$，使得 k 从某个 $k_0 > 0$ 开始都有

$$\|\mathbf{x}^{(k+1)} - \mathbf{x}^*\| \leq \beta\|\mathbf{x}^{(k)} - \mathbf{x}^*\|^{\alpha} \tag{6}$$

  成立，就称 $\{\mathbf{x}^{(k)}\}$ 的收敛阶为 α，或 $\{\mathbf{x}^{(k)}\}$ 为 α 阶收敛
- 二阶收敛 $\alpha = 2$
  超线性收敛 $1 < \alpha < 2$
  线性收敛 $\alpha = 1$ 且 $0 < \beta < 1$
- 一般而言，线性收敛速度较慢，二阶收敛速度很快。若是一个算法有超线性收敛或者更高的收敛速度，就算是好算法了

单变量优化问题

单变量最优化问题又叫一维最优化问题:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathcal{R} \end{array}$$

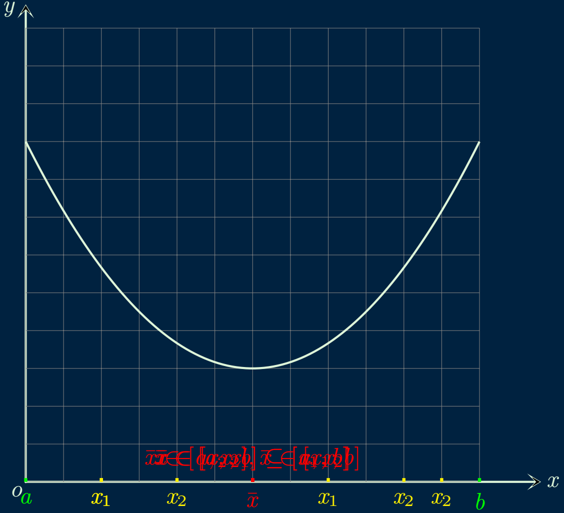
解析的方法(必要条件):$f'(x)$

单峰函数

假设 f 在 $\mathbf{dom}\ f$ 是区间 $[a,b]$ 上的一元函数,$\bar{x}$ 是 f 在 $[a,b]$ 上的极小值点,若对任意的 $x_1, x_2 \in [a,b], x_1 < x_2$,有:

1. $x_2 \le \bar{x}$,则有 $f(x_1) > f(x_2)$
2. $x_1 \ge \bar{x}$,则有 $f(x_2) > f(x_1)$

单峰函数

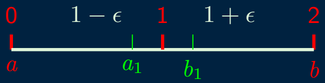


区间缩小

- 上图说明，只要在区间 $[a,b]$ 内取两不同点，并比较其函数值，则可将搜索空间由 $[a,b]$ 降为 $[a,x_2]$(如果 $f(x_2)>f(x_1)$) 或者 $[x_1,b]$（如果 $f(x_1)>f(x_2)$）
- 如果要继续缩小搜索空间 $[a,x_2]$（或者 $[x_1,b]$），只需在上述区间内再取一点，与 $f(x_1)$（或 $f(x_2)$）比较即可
- 只要缩小后的区间包含极值点 $\bar{x}$，则区间缩小的越小，就越接近于函数的极小点
- 但是，这也意味着计算次数随之上升，这说明区间的缩短率和函数值的计算次数有关
- 那么，为了考虑效率，我们想知道：计算函数值 n 次，能把原来多大的一个区间缩小成长度为一个单位的区间呢?

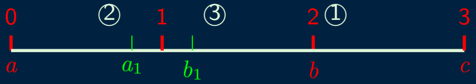
缩短率计算 1

- 如果用 F_n 表示计算 n 个函数值能缩短为单位区间的最大原区间长度，显然有 $F_0 = F_1 = 1$
- 对于 $n = 2$，有



$[a, b_1] + [a_1, b] > [a, b]$

- 由于 ϵ 可以选为任意小的数，因此缩短后的区间接近于一个单位长度，由此可得 $F_0 = F_1 = 1, F_2 = 2$
- 对于 $n = 3$，有



- 由此可得 $F_0 = F_1 = 1, F_2 = 2, F_3 = 3$

缩短率计算 2

- 对于 $n=4$，有

- 由此可得 $F_0=F_1=1, F_2=2, F_3=3, F_4=5$
- 对于 $n=5$，有

- 由此可得 $F_0=F_1=1, F_2=2, F_3=3, F_4=5, F_5=8$. 由上述计算过程可知，有递推公式 $F_n=F_{n-1}+F_{n-2}$，也就是常说的 Fibonacci 数

n	0	1	2	3	4	5	6	7	8	9	10	11	12
F_n	1	1	2	3	5	8	13	21	34	55	89	144	233

相对精度 vs 绝对精度

- 计算 n 次函数值所能获得的最大缩短比率（缩短后的区间长度与原区间长度之比）为 $\frac{1}{F_n}$
- 现在想要计算 n 个函数值，而把区间 $[a_0, b_0]$ 的长度缩短为原来长度的 δ 倍 (正小数，称为区间缩短的相对精度)，即缩短后的区间长度为 $b_{n-1} - a_{n-1} \leq \delta(b_0 - a_0)$，则只需要 n 足够大，使得

$$F_n \geq \frac{1}{\delta}$$

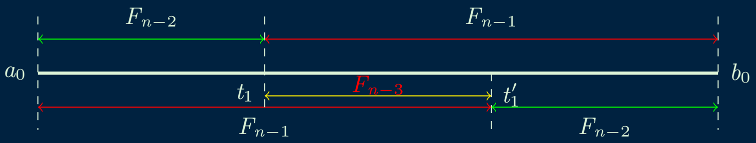
- 有时候是给出绝对精度 η，即要求 $b_{n-1} - a_{n-1} \leq \eta$。显然，绝对精度与相对精度之间存在着如下关系：

$$\eta = \delta(a_0 - b_0)$$

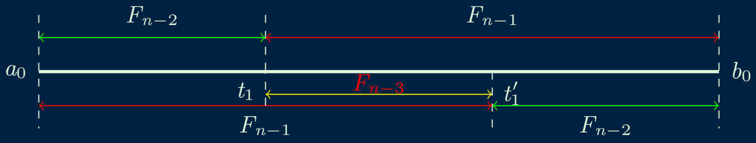
Fibonnaci 法步骤

1. 确定试点个数 n：根据相对精度 δ 确定 F_n, 然后查表得 n
2. 选取前两个试算点的位置：第一次缩短时的两个试算点的位置分别为

$$\begin{cases} t_1 = a_0 + \dfrac{F_{n-2}}{F_n}(b_0 - a_0) \\ \quad = b_0 - \dfrac{F_{n-1}}{F_n}(b_0 - a_0) \\ t_1' = a_0 + \dfrac{F_{n-1}}{F_n}(b_0 - a_0) \end{cases} \tag{7}$$



Fibonnaci 法步骤



3. 计算函数值 $f(t_1)$ 和 $f(t_1')$，并比较他们的大小
   - 若 $f(t_1) < f(t_1')$，则取

$$a_1 = a_0, b_1 = t_1', t_2' = t_1$$

$$t_2 = b_1 - \frac{F_{n-2}}{F_{n-1}}(b_1 - a_1)$$

   - 否则，$f(t_1) > f(t_1')$，则有

$$a_1 = t_1, b_1 = b_0, t_2 = t_1'$$

$$t_2' = a_1 + \frac{F_{n-2}}{F_{n-1}}(b_1 - a_1)$$

Fibonnaci 法步骤

4. 计算函数值 $f(t_2)$ 和 $f(t_2')$（其中一个已知），如第 3 步那样一步步迭代，计算试点的一般公式为

$$\begin{cases} t_k = b_{k-1} - \dfrac{F_{n-k}}{F_{n-k+1}}(b_{k-1} - a_{k-1}) \\ t_k' = a_{k-1} + \dfrac{F_{n-k}}{F_{n-k+1}}(b_{k-1} - a_{k-1}) \end{cases} \tag{8}$$

其中 $k = 1, \cdots, n-1$

Fibonnaci 法步骤

5. - 当进行到 $k = n - 1$ 时，有

$$t_{n-1} = t'_{n-1} = \frac{a_{n-2} + b_{n-2}}{2}$$

- 此时无法比较函数值 $f(t_{n-1})$ 和 $f(t'_{n-1})$ 的大小以确定最终区间，为此，取

$$\begin{cases} t_{n-1} = \dfrac{a_{n-2} + b_{n-2}}{2} \\ t'_{n-1} = a_{n-2} + \left(\dfrac{1}{2} + \epsilon\right)(b_{n-2} - a_{n-2}) \end{cases} \tag{9}$$

其中 ϵ 为任意小的数

- 在 t'_{n-1} 和 t_{n-1} 这两点中，以函数值较小者为近似极小点，相应的函数值为近似极小值，最终区间为 $[a_{n-2}, t'_{n-1}]$ 或 $[t_{n-1}, b_{n-2}]$

Fibonnaci 法求解 1

例题

试用 Fibonacci 法求函数 $f(t) = t^2 - t + 2$ 的近似极小点和极小值，要求缩短后的区间长度不大于区间 $[-1, 3]$ 的 0.08 倍

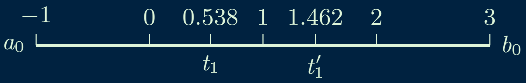
- 容易验证，$f(t) = t^2 - t + 2$ 为严格凸函数，我们可以给出其精确解为：$t^* = 0.5, f(t^*) = 1.75$
- 已知 $\delta = 0.08$，可知 $F_n = \frac{1}{0.08} = 12.5$，由此可知 $n = 6$
- 由 $a_0 = -1, b_0 = 3$，可计算得到

$$\begin{cases} t_1 = b_0 - \dfrac{F_5}{F_6}(b_0 - a_0) = & 3 - \dfrac{8}{13}(3 - (-1)) = 0.538 \\ t_1' = a_0 + \dfrac{F_5}{F_6}(b_0 - a_0) = & -1 + \dfrac{8}{13}(3 - (-1)) = 1.462 \end{cases}$$

Fibonnaci 法求解 2

- 求解可得

$$\begin{cases} f(t_1) = 0.538^2 - 0.538 + 2 & = 1.751 \\ f(t_1') = 1.462^2 - 1.462 + 2 & = 2.675 \end{cases}$$

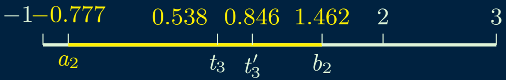


- 由于 $f(t_1) < f(t_1')$，故取 $a_1 = -1. b_1 = 1.462, t_2' = 0.538,$

$$\begin{cases} t_2 = b_1 - \dfrac{F_4}{F_5}(b_1 - a_1) = 1.462 - \dfrac{5}{8}(1.462 - (-1)) = -0.077 \\ f(t_2) = (-0.777)^2 - (-0.777) + 2 = 2.083 \end{cases}$$

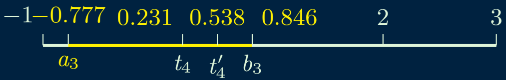
Fibonnaci 法求解 3

- 由于 $f(t_2) > f(t_2') = 1.751$，故取 $a_2 = -0.777, b_2 = 1.462, t_3 = 0.538$，



$$\begin{cases} t_3' = a_2 + \dfrac{F_3}{F_4}(b_2 - a_2) = -0.777 + \dfrac{3}{5}(1.462 - (-0.777)) = 0.846 \\ \hfill f(t_3') = (0.846)^2 - (-0.846) + 2 = 1.870 \end{cases}$$

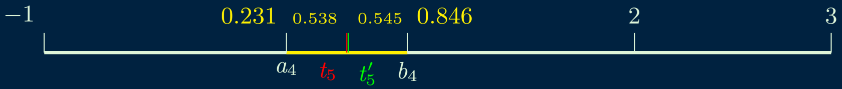
- 由于 $f(t_3') > f(t_3) = 1.751$，故取 $a_3 = -0.777, b_3 = 0.846, t_4' = 0.538$,



$$\begin{cases} t_4 = b_3 - \dfrac{F_2}{F_3}(b_3 - a_3) = 0.846 - \dfrac{2}{3}(0.846 - (-0.777)) = 0.231 \\ \hfill f(t_4) = (-0.231)^2 - 0.231 + 2 = 1.822 \end{cases}$$

Fibonnaci 法求解 4

- 由于 $f(t_4) > f(t_4') = 1.751$，故取 $a_4 = 0.231, b_4 = 0.846, t_5 = 0.538$，



$$\begin{cases} t_5' = a_4 + \left(\dfrac{1}{2} + \epsilon\right)(b_4 - a_4) = 0.231 + (0.5 + 0.01)(0.846 - 0.231) = 0.545 \\ f(t_5') = (0.545)^2 - 0.545 + 2 = 1.752 > f(t_5) = 1.751 \end{cases}$$

- 故取 $a_5 = 0.231, b_5 = 0.545$，以 t_5 为近似极小点，近似极小值为 1.751，缩短后的区间长度为 $0.545 - 0.231 = 0.314$，缩短比例为 $\frac{0.314}{4} = 0.0785$

0.618 法理论基础

- 由前述可知，Fibonacci 法是以 n 个试点来缩短某一区间时，区间长度的缩短率依次为 $\frac{F_{n-1}}{F_n}, \frac{F_{n-2}}{F_{n-1}}, \frac{F_{n-3}}{F_{n-2}} \cdots, \frac{F_2}{F_1}$
- 现将上述数列分为奇数项 $\frac{F_{2k-1}}{F_{2k}}$ 和偶数项 $\frac{F_{2k}}{F_{2k+1}}$，可以证明，这两个数列收敛于同一极限
- 设 $k \to \infty$ 时，$\frac{F_{2k-1}}{F_{2k}} \to \lambda, \frac{F_{2k}}{F_{2k+1}} \to \mu$
- 由于

$$\frac{F_{2k-1}}{F_{2k}} = \frac{F_{2k-1}}{F_{2k-1} + F_{2k-2}} = \frac{1}{1 + \frac{F_{2k-2}}{F_{2k-1}}}$$

- 所以有 $\lim\limits_{k\to\infty} \frac{F_{2k-1}}{F_{2k}} = \frac{1}{1+\mu} = \lambda$，同理有 $\mu = \frac{1}{1+\lambda}$，联立方程可得

$$\mu = \frac{1+\mu}{2+\mu}$$

- 可求得 $\lambda = \mu = \frac{\sqrt{5}-1}{2} = 0.618$

0.618 法

- 现用不变的区间缩短率 0.618，代替 Fibonacci 法中每次不同的缩短率，就得到了黄金分割法（0.618 法）
- 由于每次的缩短率为 μ，则最后的区间长度为

$$(b_0 - a_0)\mu^{n-1}$$

- 当已知缩短精度为 δ 时，可用下式计算试点个数 n:

$$\mu^{n-1} \leq \delta$$

- 当然也可以不预先计算试算点数目，而在计算过程中逐次判断
- 0.618 法是一种等速对称的试探方法