M05M11084

最优化理论、算法与应用

3 精确一维搜索方法

M05M11084

3 精确一维搜索方法

参考:

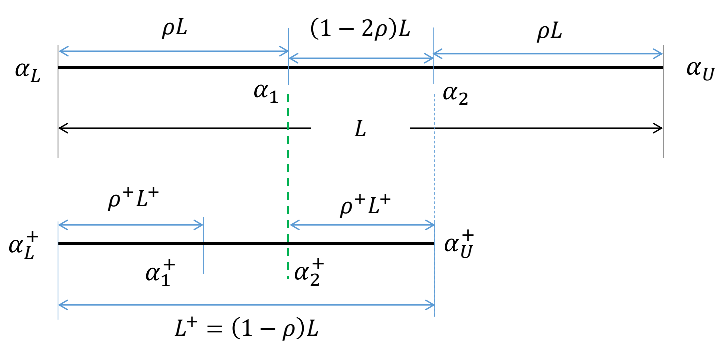
1. 应用最优化方法及MATLAB实现，刘兴高，第3章
2. 最优化导论，Edwin K.P., Chong著，孙志强等译，第7章

1. 引言
2. 区间消去法（搜索法）
   - 2.1 对分搜索法
   - 2.2 等间隔搜索法
   - **2.3 对称区间搜索法**
     - **基本思想**
     - Fibonacci法
     - 黄金分割法
3. 逼近方法
4. 划界法

对称区间搜索法

以上不难发现，在区间中取对称测试点，可以减少函数值计算次数

每次迭代，如果采用对称的两个测试点，那么，只需计算1次函数值

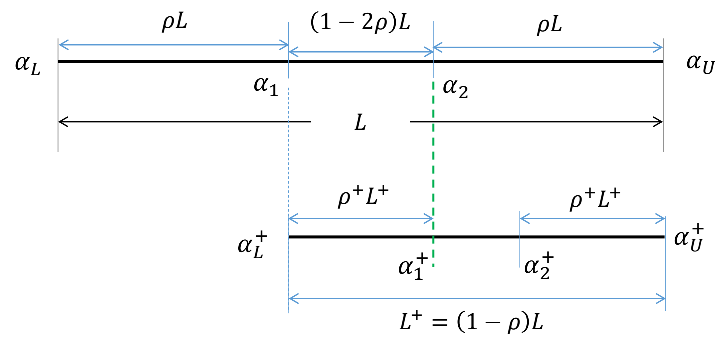


缩减率$\tau=\frac{L^+}{L}=1-\rho$

对称区间搜索法

以上不难发现，在区间中取对称测试点，可以减少函数值计算次数

每次迭代，如果采用对称的两个测试点，那么，只需计算1次函数值



缩减率$\tau = 1 - \rho$

对称区间消去算法 $\phi(\alpha) = f(x + \alpha d)$

Given $[\alpha_L^0 \quad \alpha_U^0]$, reduction rate sequence $\{\tau_k\}$, tolerance $tol > 0$, $k = 0$? $N_{max} = 100$

Compute $L_0 = \alpha_U^0 - \alpha_L^0$ $\quad \alpha_1 = \alpha_U^0 - \tau_0 L_0 \quad \phi_1 = \phi(\alpha_1)$

$\alpha_2 = \alpha_L^0 + \tau_0 L_0 \quad \phi_2 = \phi(\alpha_2)$

while $L_k > tol$

Compute $L_{k+1} = \tau_k L_k$ && $k < N_{max}$

If $\phi_1 < \phi_2$ Take $[\alpha_L^{k+1} \quad \alpha_U^{k+1}] = [\alpha_L^k \quad \alpha_2]$; $\alpha_2 := \alpha_1 \quad \phi_2 := \phi_1$

Compute $\alpha_1 = \alpha_U^{k+1} - \tau_{k+1} L_{k+1} \quad \phi_1 = \phi(\alpha_1)$

else Take $[\alpha_L^{k+1} \quad \alpha_U^{k+1}] = [\alpha_1 \quad \alpha_U^k]$; $\alpha_1 := \alpha_2 \quad \phi_1 := \phi_2$

Compute $\alpha_2 = \alpha_L^{k+1} + \tau_{k+1} L_{k+1} \quad \phi_2 = \phi(\alpha_2)$

end(if)

$k \leftarrow k + 1$

end (while)

$$a^* = \frac{1}{2}(\alpha_U^k + \alpha_L^k)$$

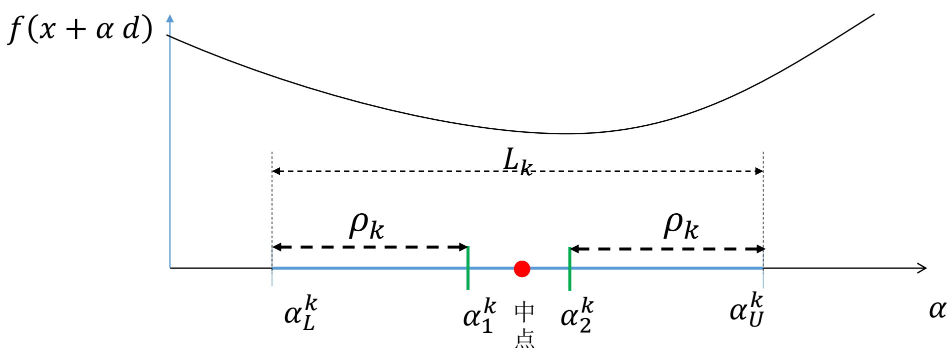
Generally, we give a maximum number of iterations.

确定缩减率

考虑不定区间长度的变化规律

设初始不定区间$[\alpha_L \quad \alpha_U]$，$k$次搜索后缩减为$[\alpha_L^k \quad \alpha_U^k]$

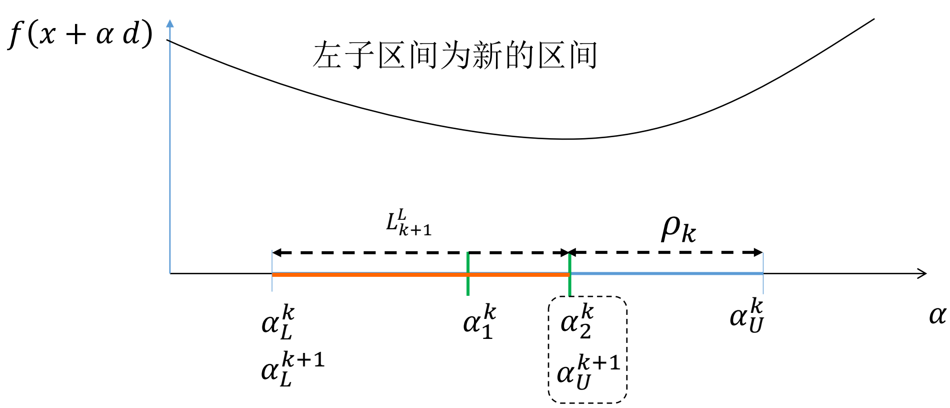
α_1^k和α_2^k关于区间中点对称



不定区间长度的变化规律 ① $L_{k+1}^L = L_{k+1}^R = L_{k+1}$

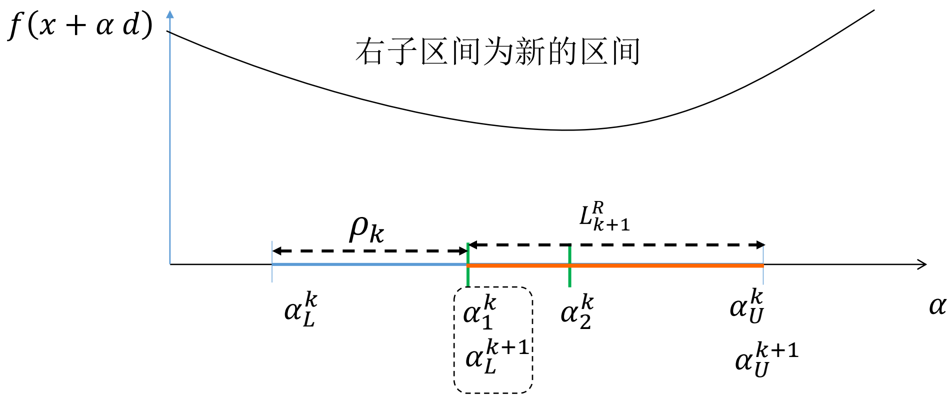
更新不定区间$k \to k+1$ 　**左**

$[\alpha_L^k \quad \alpha_U^k] \;\to\; [\alpha_L^{k+1} \quad \alpha_U^{k+1}]$



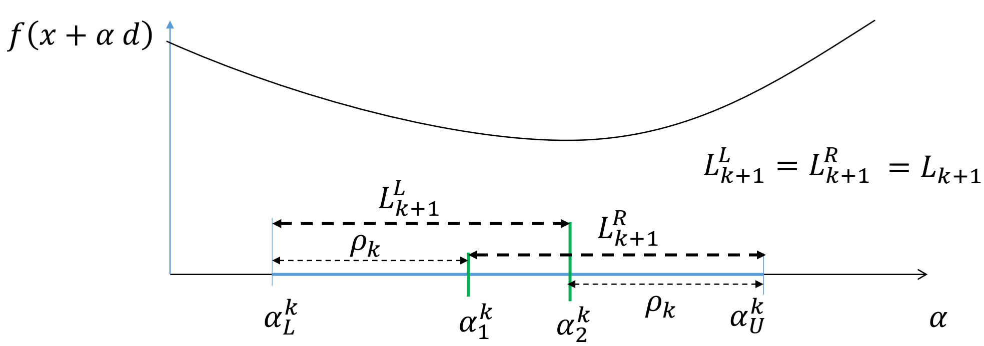
更新不定区间$k \to k+1$ ***右***

$[\alpha_L^k \quad \alpha_U^k] \ \to \ [\alpha_L^{k+1} \quad \alpha_U^{k+1}]$



无论左子区间还是右子区间成为新的区间，**新区间长度唯一**

$[\alpha_L^k \quad \alpha_U^k] \ \to \ [\alpha_L^{k+1} \quad \alpha_U^{k+1}]$

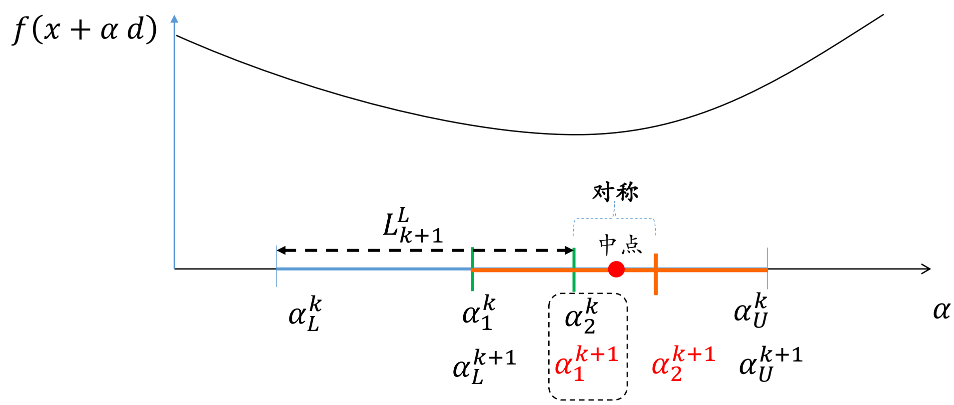


不定区间长度的变化规律　② $L_k = L_{k+1}^L + L_{k+2}^R$

不妨假定右子区间为新的区间$[\alpha_L^{k+1}\quad \alpha_U^{k+1}]$　根据对称性，确定新的试探点

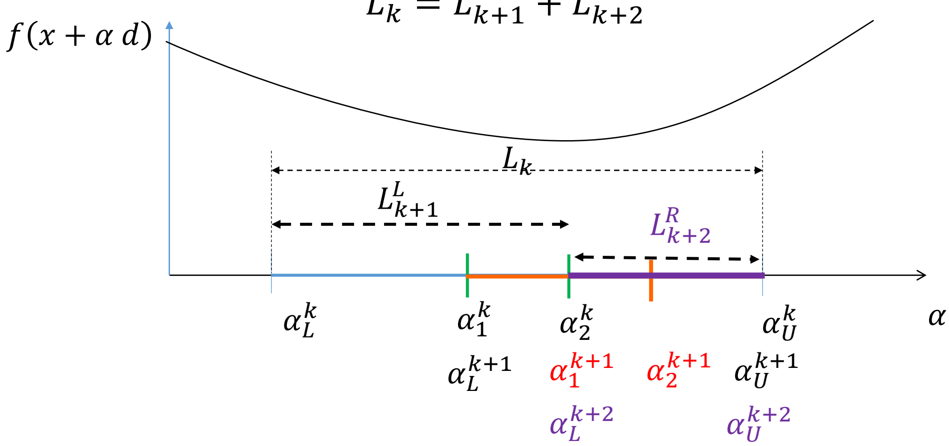
1.取左对称点$\alpha_1^{k+1} = \alpha_2^k$

2.构造α_1^{k+1}关于新区间中点的右对称点α_2^{k+1}



不妨考虑$[\alpha_L^{k+1}\quad \alpha_U^{k+1}]$**右子区间**为“更新的”区间$[\alpha_L^{k+2}\quad \alpha_U^{k+2}]$

$$L_k = L_{k+1}^L + L_{k+2}^R$$



不定区间长度的变化规律 ③ $L_k = L_{k+1} + L_{k+2}$

记L_k为区间$[\alpha_L^k \quad \alpha_U^k]$的长度

$$L_k = L_{k+1}^L + L_{k+2}^R$$

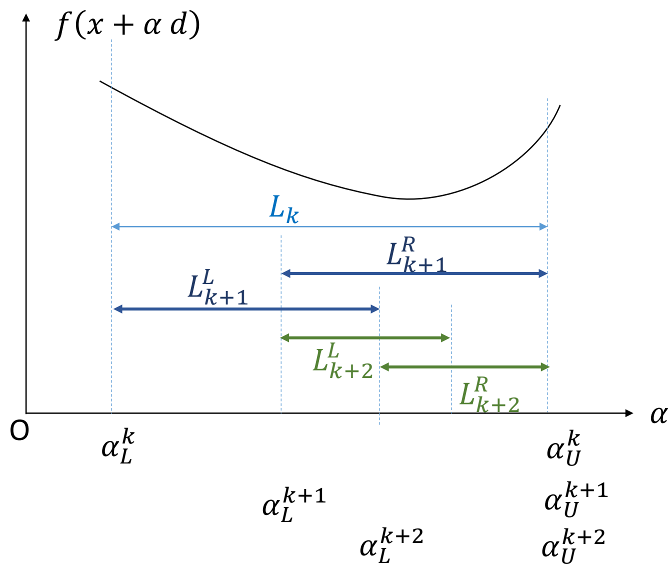
由对称性，得

$$L_{k+1}^L = L_{k+1}^R = L_{k+1}$$

$$L_{k+2}^L = L_{k+2}^R = L_{k+2}$$

得区间长度的递推公式

$$L_k = L_{k+1} + L_{k+2}$$



对称区间搜索法的缩减率 $\tau_{k+1}\tau_k = 1 - \tau_k$

$$\alpha_U^{k+1} - \alpha_L^{k+1} = \tau_k\left(\alpha_U^k - \alpha_L^k\right)$$

$$L_{k+1} = \tau_k L_k \quad \Longrightarrow L_{k+2} = \tau_{k+1}L_{k+1} = \tau_{k+1}\tau_k L_k$$

$$L_k = L_{k+1} + L_{k+2} \quad \Longrightarrow L_k = \tau_k L_k + \tau_{k+1}\tau_k L_k$$

$$1 = \tau_k + \tau_{k+1}\tau_k \quad \Longrightarrow \boxed{\tau_{k+1}\tau_k = 1 - \tau_k}$$

方程的解τ_k有许多，其中典型的有两个:

- Fibonacci法 $\tau_k \neq \tau_{k+1}$，每次迭代，缩减率不同
- 黄金分割法 $\tau_k = \tau$， 每次迭代，缩减率一样

1. 引言
2. 区间消去法（搜索法）
   - 2.1 对分搜索法
   - 2.2 等间隔搜索法
   - 2.3 对称区间搜索法
     - 基本思想
     - Fibonacci法
     - 黄金分割法
3. 逼近方法
4. 划界法

利用Fibonacci序列定义缩减率τ_k

$$L_0 = \alpha_U - \alpha_L$$
$$L_k = L_{k+1} + L_{k+2}, k = 0,1,\dots,n-1$$
$$L_{n+1} = 0$$

$$\begin{cases} L_0 = L_1 + L_2 \\ L_1 = L_2 + L_3 \\ \quad\vdots \\ L_{n-1} = L_n + L_{n+1} \end{cases}$$

$$\begin{cases} L_n = L_{n-1} - L_{n+1} = 1L_{n-1} \equiv F_0 L_{n-1} \\ L_{n-1} = L_n + L_{n+1} = 1L_{n-1} \equiv F_1 L_{n-1} \end{cases}$$

$$\begin{aligned}
L_{n-2} &= L_{n-1} + L_n = (1+1)L_{n-1} \equiv F_2 L_{n-1} \\
L_{n-3} &= L_{n-2} + L_{n-1} = (2+1)L_{n-1} = (F_2 + F_1)L_{n-1} \equiv F_3 L_{n-1} \\
&\vdots \\
L_{k+1} &= L_{k+2} + L_{k+3} = (F_{n-k-2} + F_{n-k-3})L_{n-1} \equiv F_{n-k-1} L_{n-1} \\
L_k &= L_{k+1} + L_{k+2} = (F_{n-k-1} + F_{n-k-2})L_{n-1} \equiv F_{n-k} L_{n-1} \\
L_{k-1} &= L_k + L_{k+1} = (F_{n-k} + F_{n-k-1})L_{n-1} \equiv F_{n-k+1} L_{n-1} \\
&\vdots \\
L_1 &= L_2 + L_3 = (F_{n-3} + F_{n-2})L_{n-1} \equiv F_{n-1} L_{n-1} \\
L_0 &= L_1 + L_2 = (F_{n-2} + F_{n-1})L_{n-1} \equiv F_n L_{n-1}
\end{aligned}$$

Fibonacci序列$\{F_i\}$

$$F_0 = 1$$
$$F_1 = 1$$
$$F_i = F_{i-1} + F_{i-2}, \qquad i = 2,3,\dots$$

$$\tau_k = \frac{L_{k+1}}{L_k} = \frac{F_{n-k-1}}{F_{n-k}}$$

$$L_{k+1} = \frac{F_{n-k-1}}{F_{n-k}} L_k$$

$$L_n = L_{n-1} = \frac{L_0}{F_n}$$

利用Fibonacci序列定义缩减率τ_k　　小结

Fibonacci序列$\{F_0, F_1, F_2, F_3, F_4, \cdots\} = \{1,1,2,3,5, \ldots\}$

$$\begin{cases} F_0 = 1 \\ F_1 = 1 \\ F_k = F_{k-1} + F_{k-2} \quad k = 2, \ldots, n \end{cases}$$

不定区间长度关系

$$L_{k+1} = \frac{F_{n-k-1}}{F_{n-k}} L_k$$

$$L_n = L_{n-1} = \frac{L_0}{F_n} \quad \leq tol$$

缩减率

$$\tau_k = \frac{L_{k+1}}{L_k} = \frac{F_{n-k-1}}{F_{n-k}}$$

确定迭代次数n

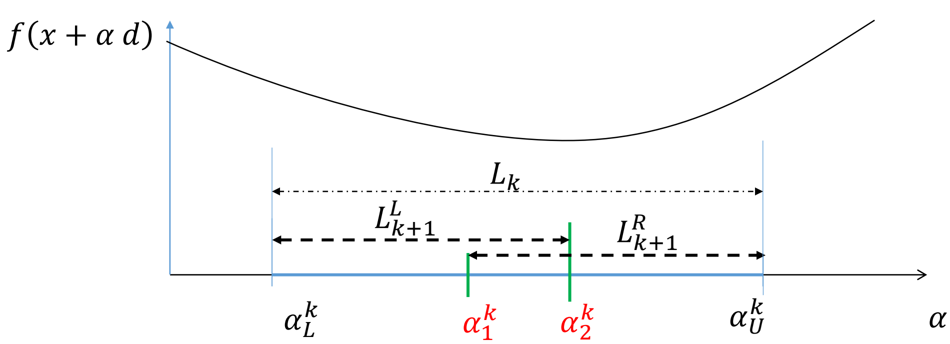
$$F_n \geq \frac{L_0}{tol}$$

由缩减率确定测试点

$$L_{k+1} = \tau_k L_k$$

$$\alpha_1^k = \alpha_U^k - L_{k+1}^R = \alpha_U^k - \tau_k L_k \qquad \tau_k = \frac{F_{n-k-1}}{F_{n-k}}$$

$$\alpha_2^k = \alpha_L^k + L_{k+1}^L = \alpha_L^k + \tau_k L_k$$



最后一次迭代采用对分搜索法

$k=n-2$时，不定区间为$[\alpha_L^{n-2} \quad \alpha_U^{n-2}]$

$$\tau_{n-2}=\frac{F_1}{F_2}=\frac{1}{2}$$

两个测试点重合 $\alpha_1^{n-2}=\alpha_2^{n-2}=\dfrac{\alpha_L^{n-2}+\alpha_U^{n-2}}{2}$

$$\alpha_M^{n-2}=\frac{\alpha_L^{n-2}+\alpha_U^{n-2}}{2}$$

做一次对分搜索, 扰动值$\varepsilon=0.1tol$

$$\alpha_1^{n-2}=\alpha_M^{n-2}-\varepsilon$$
$$\alpha_2^{n-2}=\alpha_M^{n-2}+\varepsilon$$

对称区间消去算法 Fibonacci搜索法

Given $[\alpha_L^0 \quad \alpha_U^0]$, tolerance $tol>0$, $k=0$

$N_{max}=100$

Compute $L_0=\alpha_U^0-\alpha_L^0$ Compute Fibonacci sequence $\{F_k\}_0^n$

Evaluate n by $tol \geq L_0/F_n$ reduction rate sequence $\{\tau_k\}$

$\alpha_1=\alpha_U^0-\tau_0 L_0 \quad \phi_1=\phi(\alpha_1)$

$\alpha_2=\alpha_L^0+\tau_0 L_0 \quad \phi_2=\phi(\alpha_2)$

while $L_k>tol$ && $k<N_{max}$

Compute $L_{k+1}=\tau_k L_k$

If $k=n-2$

$\alpha_1^{n-2}=\alpha_M^{n-2}-0.1tol, \alpha_2^{n-2}=\alpha_M^{n-2}+0.1tol \qquad \alpha_M^{n-2}=\dfrac{\alpha_L^{n-2}+\alpha_U^{n-2}}{2}$

else

If $\phi_1<\phi_2$ Take $[\alpha_L^{k+1} \quad \alpha_U^{k+1}]=[\alpha_L^k \quad \alpha_2]$; $\alpha_2:=\alpha_1 \quad \phi_2:=\phi_1$

Compute $\alpha_1=\alpha_U^{k+1}-\tau_{k+1}L_{k+1} \quad \phi_1=\phi(\alpha_1)$

else Take $[\alpha_L^{k+1} \quad \alpha_U^{k+1}]=[\alpha_1 \quad \alpha_U^k]$; $\alpha_1:=\alpha_2 \quad \phi_1:=\phi_2$

Compute $\alpha_2=\alpha_L^{k+1}+\tau_{k+1}L_{k+1} \quad \phi_2=\phi(\alpha_2)$

end(if)

end(if)

$k\leftarrow k+1$

end (while)

$a^*=\dfrac{1}{2}(\alpha_U^k+\alpha_L^k)$

Fibonacci_search.m

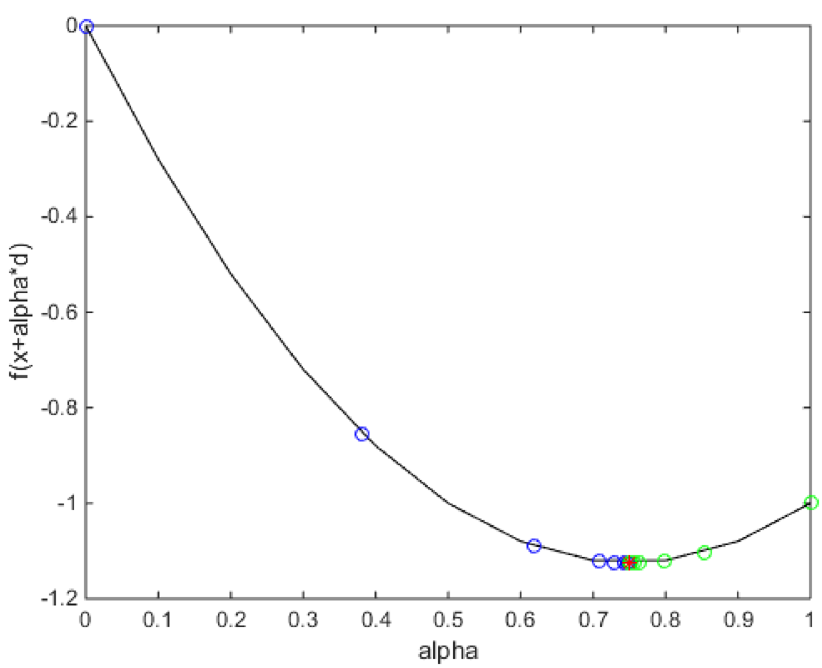
实例测试

例3.7 已知函数$f(x)=2x^2-x-1$在当前点$x=-0.5$处的一个下降方向$d=1$和不定区间$\alpha^*\in[0\quad 1]$，用Fibonacci搜索法获取最佳步长（取$tol=1\times10^{-4}$）

解：函数$f(x)$是凸函数，且$f(0.25)=-1.125$。

alpha_star =0.7500
x_next =0.2500
f_next =-1.1250
k =19

example_3_7_CH03.m



ple_3_7_CH03.m

例3.8 已知函数$f(\boldsymbol{x}) = x_1^2 + x_2^2 - 1$在当前点$\boldsymbol{x} = [2 \quad 2]$处的一个下降方向$d = [-1 \quad -1]$和不定区间$\alpha^* \in [0 \quad 2]$，用Fibonacci搜索法获取最佳步长（取$tol = 1 \times 10^{-6}$）。

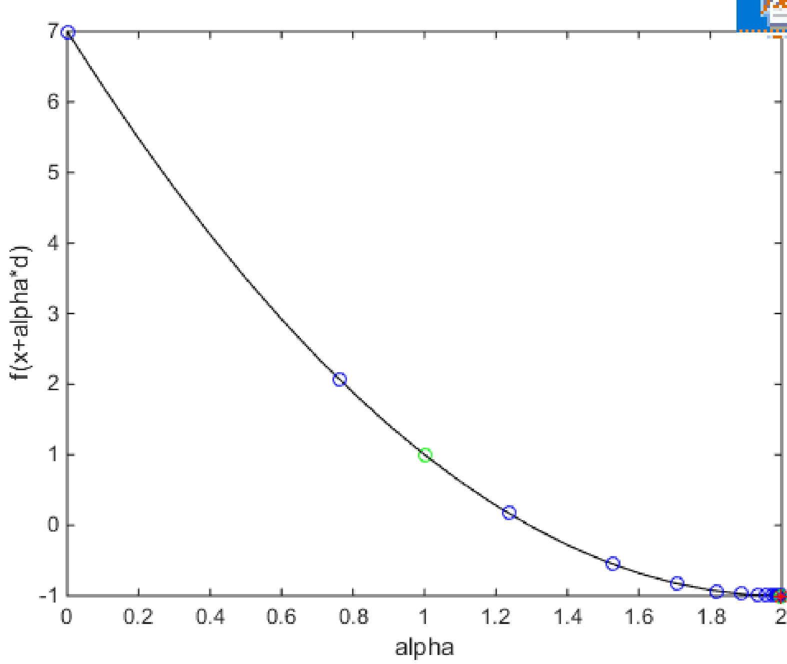
解：函数$f(\boldsymbol{x})$是凸函数，且$f(0,0) = -1$。

example_3_8_CH03.m

alpha_star =2.0000
x_next =1.0e-06 *[0.5091 0.5091]
f_next =-1.0000
k =30

改变tol的值，发现tol由大至小，数值解趋于解析解

example_3_8_CH03.m



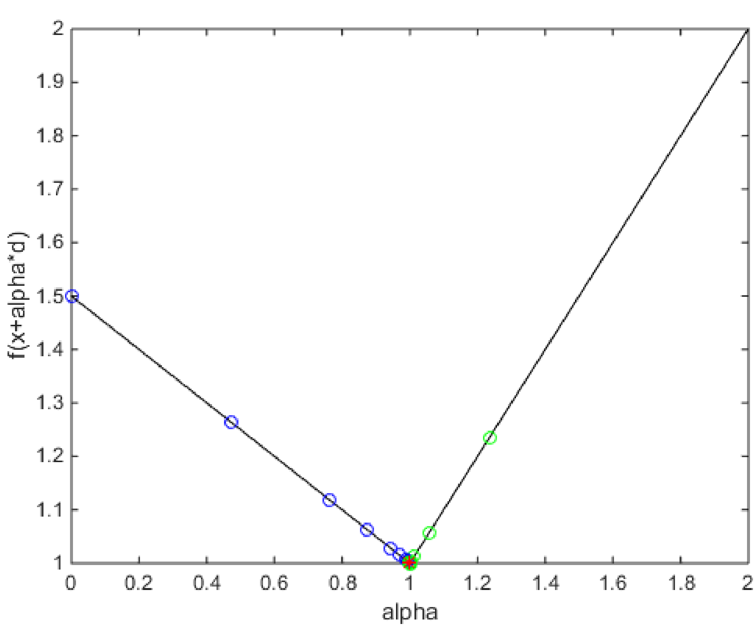
例3.9 已知函数$f(\boldsymbol{x})=\begin{cases}-x+3 & x\le 2\\ \frac{1}{2}x & x>2\end{cases}$在当前点$x=3$处的一个下降方向$d=-1$和不定区间$\alpha^*\in[0\quad 2]$，用Fibonacci搜索法获取最佳步长（取$tol=1\times10^{-4}$）。

解：函数$f(\boldsymbol{x})$是凸函数，且$f(2)=1$。

alpha_star = 1.0000
x_next =2.0000
f_next =1.0000
k =21

example 3 9 CH03.m

mple 3 9 CH03.m



黄金分割的基本思想 与 特点

对称不定区间缩减率：

$$\tau_{k+1}\tau_k = 1 - \tau_k，\ \tau_k = \beta$$

$$\beta^2 = 1 - \beta$$

$$\beta = \frac{-1+\sqrt{5}}{2} \approx 0.618$$

$$1 - \beta = \frac{3-\sqrt{5}}{2} \approx 0.382$$

黄金分割与Fibonacci搜索法的关系

①随着搜索次数的增加， Fibonacci搜索法的不定区间缩减率趋近于黄金分割法的，$\tau_k \to \beta$

②黄金分割法的效率略低于Fibonacci搜索法的效率

$$\frac{[\alpha_L^{k-1} \quad \alpha_U^{k-1}]_F}{[\alpha_L^{k-1} \quad \alpha_U^{k-1}]_G} = \frac{\tau_1^2}{\sqrt{5}} = \frac{3\sqrt{5}+5}{10} \approx 1.17$$

当搜索次数非常大时，黄金分割法的最终区间的长度比Fibonacci搜索法的长17%

③当指定精度改变时， Fibonacci搜索法需要重新计算n以及试探步长对应的函数值；

而黄金分割法只需在当前搜索结果的基础之上继续搜索

对称区间消去算法　黄金分割法

Given $[\alpha_L^0 \quad \alpha_U^0]$, reduction rate $\tau = \frac{-1+\sqrt{5}}{2}$, tolerance $tol > 0$, $k = 0$　　$N_{max} = 100$

Compute $L_0 = \alpha_U^0 - \alpha_L^0$　$\alpha_1 = \alpha_U^0 - \boldsymbol{\tau} L_0$　$\phi_1 = \phi(\alpha_1)$

$\alpha_2 = \alpha_L^0 + \boldsymbol{\tau} L_0$　$\phi_2 = \phi(\alpha_2)$

while $L_k > tol$ && $k < N_{max}$

Compute $L_{k+1} = \boldsymbol{\tau} L_k$

If $\phi_1 < \phi_2$　Take $[\alpha_L^{k+1} \quad \alpha_U^{k+1}] = [\alpha_L^k \quad \alpha_2]$;　$\alpha_2 := \alpha_1$　$\phi_2 := \phi_1$

Compute $\alpha_1 = \alpha_U^{k+1} - \boldsymbol{\tau} L_{k+1}$　$\phi_1 = \phi(\alpha_1)$

else　Take $[\alpha_L^{k+1} \quad \alpha_U^{k+1}] = [\alpha_1 \quad \alpha_U^k]$;　$\alpha_1 := \alpha_2$　$\phi_1 := \phi_2$

Compute $\alpha_2 = \alpha_L^{k+1} + \boldsymbol{\tau} L_{k+1}$　$\phi_2 = \phi(\alpha_2)$

end(if)

$k \leftarrow k + 1$

end (while)

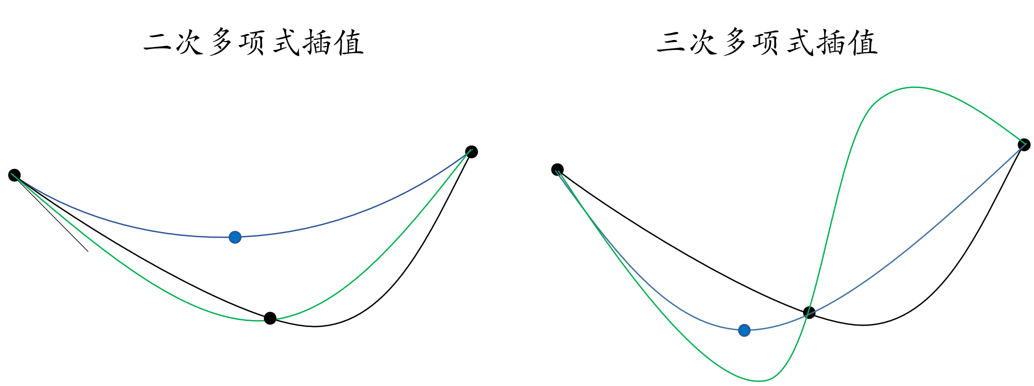
$a^* = \frac{1}{2}(\alpha_U^k + \alpha_L^k)$

学生练习编写

1. 引言
2. 区间消去法（搜索法）
3. 逼近方法
   3.1 曲线插值法（略）
   - 二次逼近法
   - 三次逼近法

   3.2 牛顿法
   3.3 割线法
4. 划界法

逼近方法



• 二次插值法

基本思想:

①根据函数在子区间2~3点（如端点、试探步长点）的函数值、导数值，构造一个二次多项式函数;

②用这个二次多项式的极值点作为原函数的极值点近似点;

③以此点判定得到新的包含原函数极值点的新的子区间，如此迭代，直至子区间长度足够小

Lagrange插值函数（三点二次插值函数）

$$L_2(x)=\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}y_0+\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}y_1+\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}y_2$$

函数的极值点 $L_2'(x)=0$

$$x^*=\frac{1}{2}(x_0+x_1)+\frac{1}{2}\cdot\frac{(y_1-y_0)(x_0-x_2)(x_1-x_2)}{(x_1-x_2)y_0-(x_0-x_2)y_1+(x_0-x_1)y_2}$$

设初始不定区间$[\alpha_L \quad \alpha_U]$，经过$k$次搜索后缩减为$[\alpha_L^k \quad \alpha_U^k]$，$\alpha^k\in[\alpha_L^k \quad \alpha_U^k]$;

取

参数α	α_L^k	α^k	α_U^k	x_i
函数值	$f(x+\alpha_L^k d)$	$f(x+\alpha^k d)$	$f(x+\alpha_U^k d)$	y_i

$$\alpha^{k+1}=\frac{1}{2}\left(\alpha_L^k+\alpha^k\right)+\frac{1}{2}\cdot\frac{\left(f\left(x+\alpha^k d\right)-f\left(x+\alpha_L^k d\right)\right)\left(\alpha_L^k-\alpha_U^k\right)\left(\alpha^k-\alpha_U^k\right)}{\left(\alpha^k-\alpha_U^k\right)f\left(x+\alpha_L^k d\right)-\left(\alpha_L^k-\alpha_U^k\right)f\left(x+\alpha^k d\right)+\left(\alpha_L^k-\alpha^k\right)f\left(x+\alpha_U^k d\right)}$$

四种不同情况下，子区间的确定

情况	若		则	
			$[\alpha_L^{k+1} \quad \alpha_U^{k+1}]$	α^{k+1}
①	$\alpha_L^k<\alpha^{k+1}<\alpha^k$	$f\left(x+\alpha^{k+1}d\right)\le f\left(x+\alpha^k d\right)$	$[\alpha_L^k \quad \alpha^k]$	α^{k+1}
②		$f\left(x+\alpha^{k+1}d\right)>f\left(x+\alpha^k d\right)$	$[\alpha^{k+1} \quad \alpha_U^k]$	α^k
③	$\alpha^k<\alpha^{k+1}<\alpha_U^k$	$f\left(x+\alpha^{k+1}d\right)\le f\left(x+\alpha^k d\right)$	$[\alpha^k \quad \alpha_U^k]$	α^{k+1}
④		$f\left(x+\alpha^{k+1}d\right)>f\left(x+\alpha^k d\right)$	$[\alpha_L^k \quad \alpha^{k+1}]$	α^k

当$\left|\alpha^{k+1}-\alpha^k\right|\le tol$，终止迭代，$\alpha^*\approx\alpha^{k+1}$

两点二次插值法

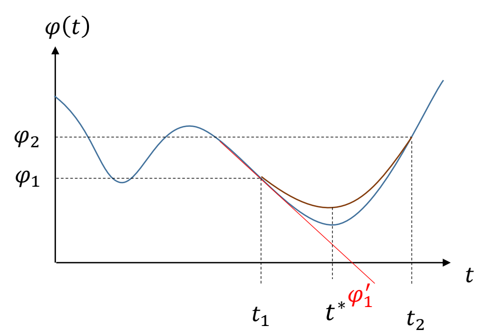
内插法

①已知函数一点的函数值和导数及另一点的函数值

设 函数$\varphi(t)$，已知$\varphi_i=\varphi(t_i)$，$i=1,2$，$\varphi_1'=\varphi'(t_1)$

$P(t)=a(t-t_1)^2+b(t-t_1)+c$，满足$P(t_i)=\varphi_i$，$i=1,2$；$\varphi_1'=P'(t_1)$

$$t^*=t_1-\frac{\varphi_1'(t_2-t_1)^2}{2[\varphi_2-\varphi_1-\varphi_1'(t_2-t_1)]}$$



外插法

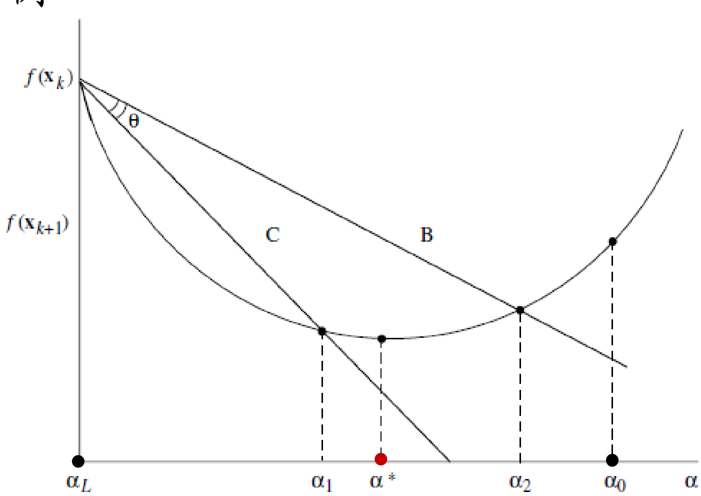
②已知函数一点的函数值和两点的导数

设 函数$\varphi(t)$，已知$\varphi_1=\varphi(t_1)$，$\varphi_i'=\varphi'(t_i)$, $i=1,2$

$P(t)=a(t-t_1)^2+b(t-t_1)+c$，满足$P(t_1)=\varphi_1$； $\varphi_1'=P'(t_1)$，$i=1,2$

$$t^* = t_1-\frac{\varphi_1'(t_2-t_1)}{\varphi_2'-\varphi_1'}$$

例



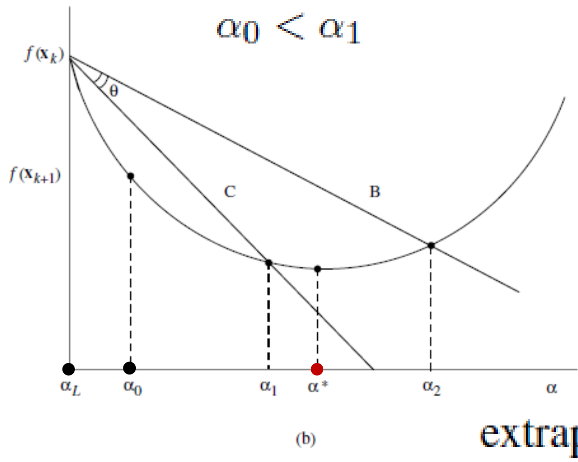
(a)

$$\alpha_0>\alpha_2$$

the interpolation formula

$$\breve{\alpha}_0=\alpha_L+\frac{(\alpha_0-\alpha_L)^2 f_L'}{2[f_L-f_0+(\alpha_0-\alpha_L)f_L']}$$

例



extrapolation formula

$$\breve{\alpha}_0 = \alpha_0 + \frac{(\alpha_0 - \alpha_L) f_0'}{(f_L' - f_0')}$$

1. 引言
2. 区间消去法（搜索法）
3. 逼近方法
   3.1 曲线插值法（略）
   3.2 牛顿法
   3.3 割线法
4. 划界法

An Introduction to Optimization 4th ed. - E. Chong, S. Zak 第7章

Newton's Method

At x^k, using $f(x^k)$, $f'(x^k)$, and $f''(x^k)$

define a quadratic, $q(x) = f(x^k) + f'(x^k)(x - x^k) + \frac{1}{2} f''(x^k)(x - x^k)^2$

$$x^{k+1} = \operatorname{argmin} q(x)$$

FONC for a minimizer of q yields

$$0 = q'(x) = f'(x^k) + f''(x^k)(x - x^k)$$

$$\boxed{x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}}$$

Newton recurrence formula

Example 7.4

Find the minimizer of

$$f(x) = \frac{1}{2}x^2 - \sin x$$

$x^0 = 0.5$, $\varepsilon = 10^{-5}$, in the sense that we stop when $|x^{k+1} - x^k| < \varepsilon$.

Solution $f'(x) = x - \cos x$, $f''(x) = 1 + \sin x$

$$x^1 = x^0 - \frac{f'(x^0)}{f''(x^0)} = 0.5 - \frac{-0.3775}{1.479} = 0.7552$$

$$x^2 = x^1 - \frac{f'(x^1)}{f''(x^1)} = 0.7552 - \frac{0.0271}{1.685} = 0.7391$$

$$x^3 = x^2 - \frac{f'(x^2)}{f''(x^2)} = 0.7391 - \frac{9.461 \times 10^{-5}}{1.673} = 0.7390$$

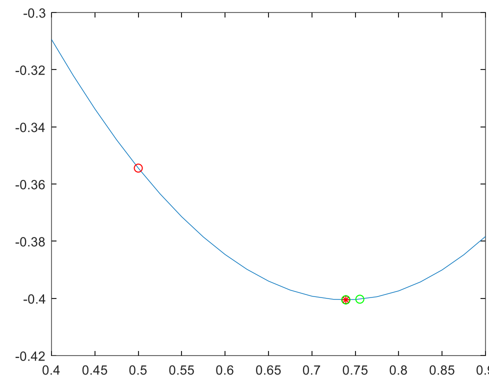
$$x^4 = x^3 - \frac{f'(x^3)}{f''(x^3)} = 0.7390 - \frac{1.170 \times 10^{-9}}{1.673} = 0.7390$$

Example 7.4 cntd

$x^3 = 0.7390, \qquad x^4 = 0.7390$

$$|x^4 - x^3| < \varepsilon = 10^{-5}$$
$$f'(x^4) = -8.6 \times 10^{-6} \approx 0$$
$$f''\left(x^{(4)}\right) = 1.673 > 0$$

$x^* \approx x^4$ is a strict minimizer

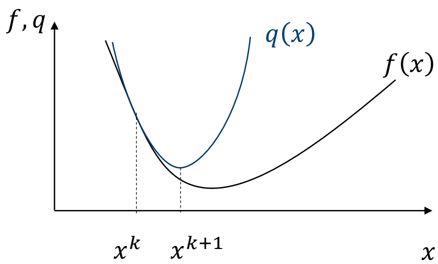


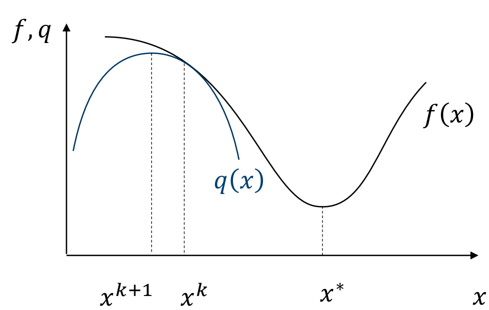
P01_Chong_CH07_Example_7_4.m

property

Newton's method works well if $f''(x) > 0$ everywhere.

If $f''(x) < 0$ for some x, Newton's method may fail to converge to the minimizer





Newton's Failure

The number of iterations required can not be determined before the algorithm begins.

The algorithm will not work if $f(x)$ is not differentiable.

The algorithm will halt (program termination by division by zero if not checked for) if a horizontal tangent line is encountered.

Newton's method will sometimes find an extraneous root.

An Introduction to Optimization 4th ed. - E. Chong, S. Zak 第7章

Secant Method

Newton's method for minimizing f

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

approximate $f''(x^k)$ with

$$f''(x^k) \approx \frac{f'(x^k) - f'(x^{k-1})}{x^k - x^{k-1}}$$

First Difference

$$\boxed{x^{k+1} = x^k - \frac{x^k - x^{k-1}}{f'(x^k) - f'(x^{k-1})} f'(x^k)}$$

Requires two initial points x^{-1} and x^0

$$x^{k+1} = \frac{f'(x^k)x^{k-1} - f'(x^{k-1})x^k}{f'(x^k) - f'(x^{k-1})}$$

$$\begin{array}{cc} x^{k-1} & f'(x^{k-1}) \\ x^k & f'(x^k) \end{array}$$

It does not involve $f(x^k)$

It tries to drive $f' \to 0$

Stop Condition

$$|f'(x^k)| < \varepsilon$$

$$|f'(x^k) - f'(x^{k-1})| < \varepsilon$$

$$|f'(x^k) - f'(x^{k-1})| < \varepsilon |f'(x^k)|$$

Bracketing

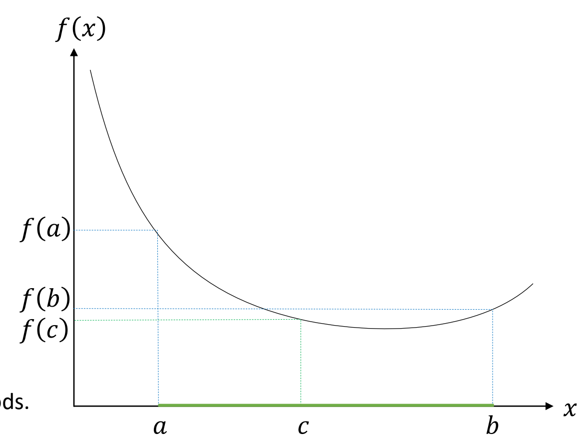
划界法，包围盒，区间逼近法，区间缩减法，向前向后搜索法

The interval in which the minimizer lies is also called a *bracket*, and procedures for finding such a bracket are called *bracketing* methods.

To find a bracket [*a, b*] containing the minimizer, assuming unimodality

find three points $a < c < b$

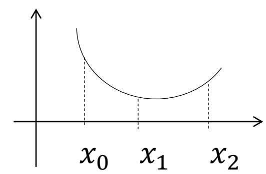
$$f(c) < f(a)\,,\ f(c) < f(b)$$



The desired bracket can initialize any search method, including the golden section, Fibonacci, and bisection methods.

Bracketing – Simple Procedure

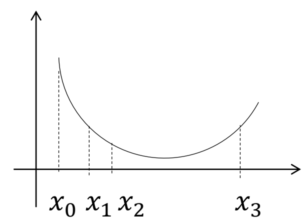
Pick three arbitrary points $x_0 < x_1 < x_2$
If $f(x_1) < f(x_0)$ and $f(x_1) < f(x_2)$,
the desired bracket is $[x_0 \quad x_2]$



Forward

If not, say $f(x_0) > f(x_1) > f(x_2)$,
then pick a point $x_3 > x_2$ and $f(x_2) < f(x_3)$
the desired bracket is $[x_1 \quad x_3]$

Otherwise,
continue with this process until the function increases.

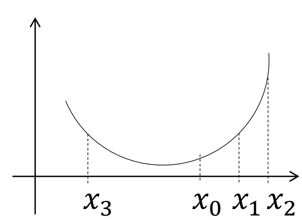


The minimizer is to the right of the rightmost point.

Backward

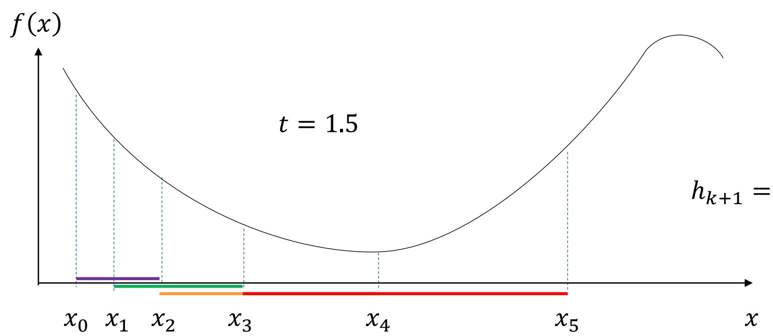
If $f(x_0) < f(x_1) < f(x_2)$,
then pick a point $x_3 < x_2$ and $f(x_2) < f(x_3)$
the desired bracket is $[x_3 \quad x_1]$.

Otherwise,
continue with this process until the function increases.



The minimizer is to the left of the leftmost point.

Simple Procedure *Forward*

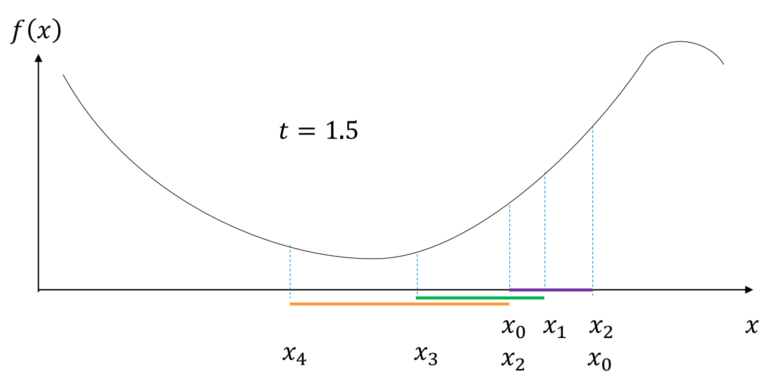


$h_{k+1} = th_k \quad x_{k+1} = x_k + h_k$

k Number of iteration $\quad h_k$ Step $\quad t$ Coefficient for increasing Step

x_k Current Point $\quad x_{k-1}$ Last Point $\quad x_{k+1}$ Next Point

Simple Procedure *Backward*



k Number of iteration $\quad h_k$ Step $\quad t$ Coefficient for increasing Step

x_k Current Point $\quad x_{k-1}$ Last Point $\quad x_{k+1}$ Next Point

Bracketing Algorithm

Step 0 $k = 1$

Given an initial point x_0, step length h_0;

$h_1 = th_0, x_1 = x_0 + h_1 \quad h_2 = th_1, x_2 = x_1 + h_2$

Step 1 If $f(x_k) < f(x_{k-1})$ and $f(x_k) < f(x_{k+1})$, the desired bracket is $[x_{k-1}, x_{k+1}]$. Stop.

Step 2 If $f(x_{k-1}) > f(x_k) > f(x_{k+1})$, $k := k + 1$

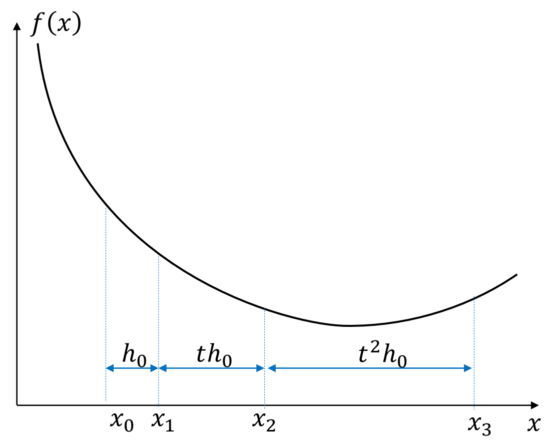
$h_{k+1} = th_k \quad x_{k+1} = x_k + h_k$; GoTo Step1.

Else GoTo Step 3

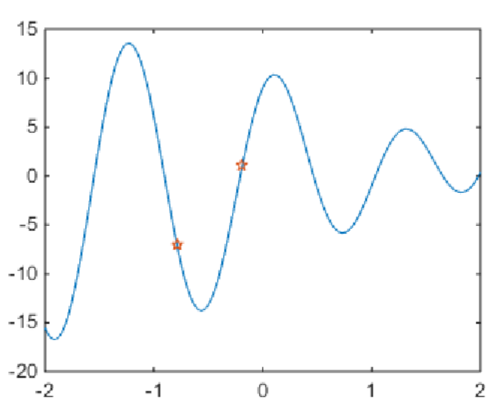
Step 3 $x_{k-1} \leftrightarrows x_{k+1}$, $\quad k := k + 1$

$h_{k+1} = -th_k \quad x_{k+1} = x_k + h_k$

GoTo Step1.




```
fun = @(x) x+6*sin(4*x)+9*cos(5*x);
x0=0;
h0=0.0125;
[lb, ub] = Opt_AdvanceRetreat(fun, x0, h0)
```



```
lb =

   -0.7875

ub =

   -0.1875
```

Opt_AdvanceRetreat.m

Single_peak_interval.m

Write a Routine with MATLAB

MATLAB常用符号与函数

点连成线
plot(x,y, ‘LineStyle’,’-.’'color','m')

点坐标 x vector $n \times 1$;
y vector $n \times 1$;

线型 “点划线” 线色“粉红”

线型		点标记				颜色			
−	实线	.	点	v	下三角	b	蓝色	m	棕色
:	点线	o	小圆圈	^	上三角	g	绿色	y	黄色
−.	点划线	x	叉号	<	左三角	r	红色	k	黑色
--	虚线	+	加号	>	右三角	c	青色	w	白色
		•	星号	p	五角星				
		s	方格	h	六角星				
		d	菱形						

'LineStyle'

'Color' — Line color

x=0:0.01:1;

区间：[0,1]

在域[0,1]以0为起点，步长为0.01，终点为1，得离散点列$\boldsymbol{x}_{n\times 1}$

y=x.^2-sin(x);

$y = x^2 - \sin x$

离散点列相应的函数值点阵$y_{n\times 1}$

n=101

每个离散点的函数值，存入列矢y

plot(x,y, ‘LineStyle’,’-.’'color','m')

具体可以使用matlab的help查询如何使用，例

>>help plot

```
>> help plot
plot - 2-D line plot

  This MATLAB function creates a 2-D line plot of the data in Y versus the
  corresponding values in X. If X and Y are both vectors, then they must have equal
  length.

  plot(X,Y)
  plot(X,Y,LineSpec)
  plot(X1,Y1,...,Xn,Yn)
  plot(X1,Y1,LineSpec1,...,Xn,Yn,LineSpecn)
  plot(Y)
  plot(Y,LineSpec)
  plot(___,Name,Value)
  plot(ax,___)
  h = plot(___)

  plot 的参考页
```

作业

P61

3-3

3-6