Раздел 2. Нелинейное программирование. Безусловная одномерная оптимизация

2.1. Алгоритм равномерного поиска

Рассмотрим следующую задачу условной оптимизации: найти минимум одномерной унимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Идея алгоритмов, относящихся к методу сокращения текущего интервала неопределенности, состоит в исключении в процессе поиска из рассмотрения тех подынтервалов, в которых в силу унимодальности функции $\Phi(x)$ точка x^* отсутствует.

Текущий интервал неопределенности будем обозначать ТИН, а его длину |ТИН|. Так что, если $\text{ТИН}=\left[a,b\right]$, то $|\text{ТИН}|=b-a$.

В **алгоритме равномерного поиска** испытания проводятся в точках, которые определяются путем равномерного деления интервала $\left[a,b\right]$ на N одинаковых подынтервалов. Из вычисленных значений функции $\Phi(x)$ выбирается наименьшее. Пусть это значение достигается в точке x_k. Тогда в связи с унимодальностью функции $\Phi(x)$ подынтервалы $\left[a,x_{k-1}\right]$, $\left[x_{k+1},b\right]$ можно исключить из рассмотрения, т.е. сделать очередным интервалом неопределенности интервал $\left[x_{k-1},x_{k+1}\right]$.

Алгоритм относится к классу пассивных методов поиска.

Алгоритм равномерного поиска.

1. Выполняем присваивания $r=1$, $a^1=a$, $b^1=b$, $\text{ТИН}_1=\left[a^1,b^1\right]$.

2. На текущем ТИН строим равномерную сетку с $N+1$ узлами (см. рис. 2.1.1).

3. Вычисляем значения функции $\Phi(x)$ в узлах построенной сетки $\Phi\left(x_0^r\right),...,\Phi\left(x_N^r\right)$.

4. Находим минимальное из этих значений: $\min\left(\Phi\left(x_0^r\right),...,\Phi\left(x_N^r\right)\right)=\Phi\left(x_k^r\right)$.

5. Выполняем присваивания $a^{r+1}=x_{k-1}^r$, $b^{r+1}=x_{k+1}^r$, $\text{ТИН}_{r+1}=\left[a^{r+1},b^{r+1}\right]$.

6. Если $\left|\text{ТИН}_{r+1}\right|\le\varepsilon_x$, то заканчиваем вычисления. Иначе - выполняем присваивание $r=r+1$ и переходим на п.2. Здесь ε_x – требуемая точность решения.

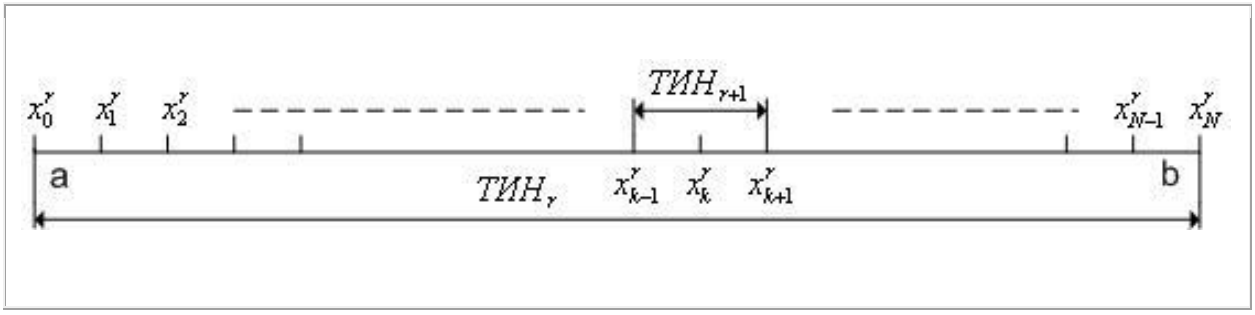

Рис. 2.1.1. Построение сетки на текущем интервале неопределенности

В качестве приближенного значения точки минимума x^* с равными основаниями может быть принята любая точка последнего текущего интервала неопределенности.

Первую итерацию приведенной схемы алгоритма равномерного поиска иллюстрирует рис.2.1.2. Легко видеть, что после одной итерации алгоритма равномерного поиска ТИН уменьшается в $\frac{N}{2}$ раз. Поэтому количество итераций r, необходимых для нахождения минимума функции с точностью ε_x, может быть найдено из условия $\left(\frac{2}{N}\right)^r\left(b-a\right)\le\varepsilon_x$.

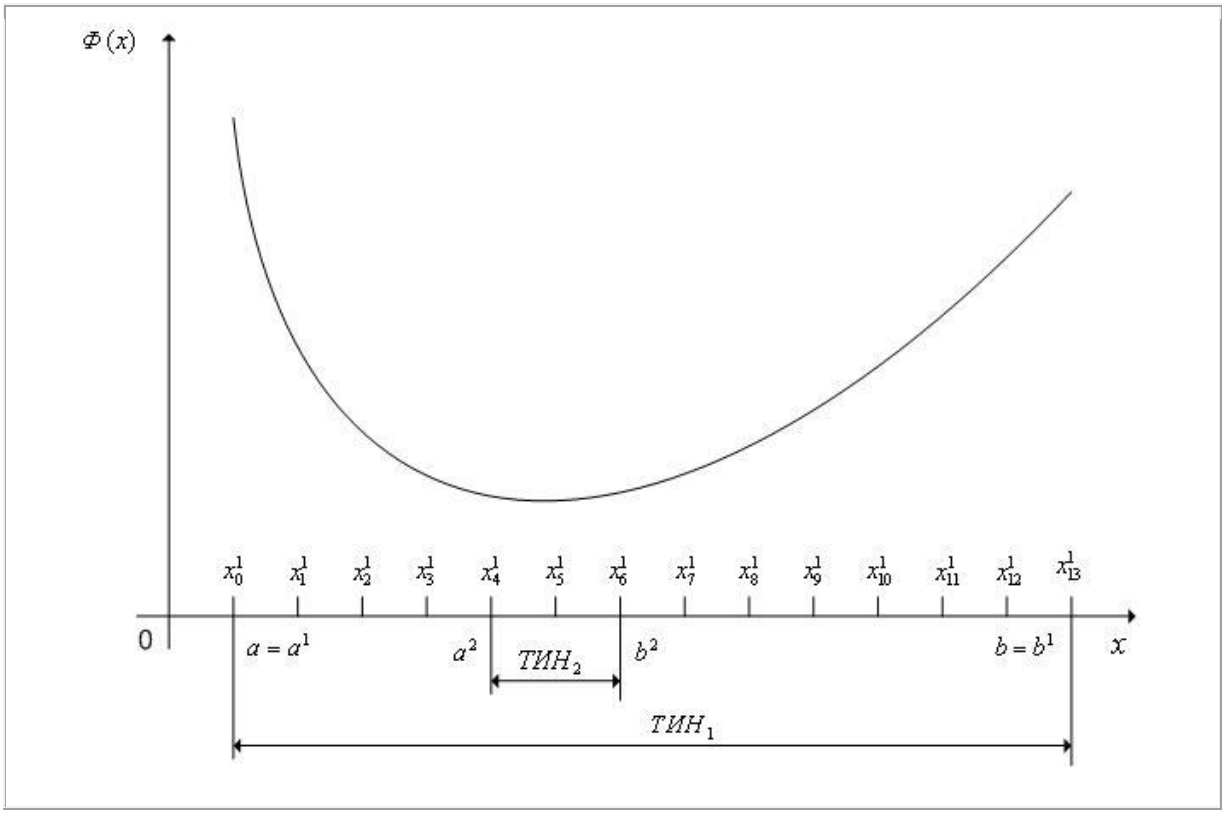

Рис. 2.1.2. Первая итерация поиск минимума одномерной унимодальной функции $\Phi(x)$ с помощью алгоритма равномерного поиска: $N=13$

2.2. Алгоритм деления пополам

Рассмотрим следующую задачу условной оптимизации: найти минимум одномерной унимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

В алгоритм деления пополам или алгоритме равномерного дихотомического поиска испытания проводятся парами. Координаты каждой последующей пары испытаний разнесены между собой на величину $\delta_x<\varepsilon_x$, где ε_x - требуемая точность решения. Испытания производятся в середине ТИН. По значениям $\Phi(x)$, полученным в этих точках, одна половина ТИН в силу унимодальности функции $\Phi(x)$ исключается из дальнейшего рассмотрения. Величина δ_x определяется требуемой точностью решения. Алгоритм относится к классу методов последовательного поиска.

Алгоритм деления пополам.

1. Выполняем присваивания $r=1$, $a^1=a$, $b^1=b$, $\text{ТИН}_1=\left[a^1,b^1\right]$.
2. Вычисляем величины (см. рис. 2.2.1) $x_0^r=\dfrac{a^r-b^r}{2}$, $x_1^r=x_0^r-\dfrac{\delta_x}{2}$, $x_2^r=x_0^r+\dfrac{\delta_x}{2}$.
3. Вычисляем значения $\Phi\left(x_1^r\right),\Phi\left(x_2^r\right)$ функции $\Phi(x)$.
4. Если $\Phi\left(x_1^r\right)<\Phi\left(x_2^r\right)$, то выполняем присваивания $a^{r+1}=a^r$, $b^{r+1}=x_0^r$, $\text{ТИН}_{r+1}=\left[a^{r+1},b^{r+1}\right]$. Иначе - выполняем присваивания $a^{r+1}=x_0^r$, $b^{r+1}=b^r$, $\text{ТИН}_{r+1}=\left[a^{r+1},b^{r+1}\right]$.
5. Если $\left|\text{ТИН}_{r+1}\right|\le\varepsilon_x$, то заканчиваем вычисления. Иначе - выполняем присваивание $r=r+1$ и переходим на п.2.

В качестве приближенного значения точки минимума с равными основаниями может быть принята любая точка последнего текущего интервала неопределенности.

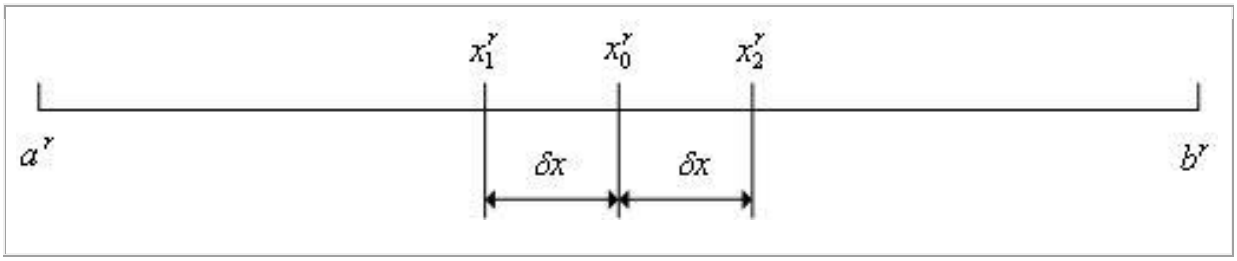

Рис. 2.2.1. К определению величин x_0^r, x_1^r, x_2^r

Приведенную схему алгоритма равномерного дихотомического поиска иллюстрирует рис. 2.2.2.

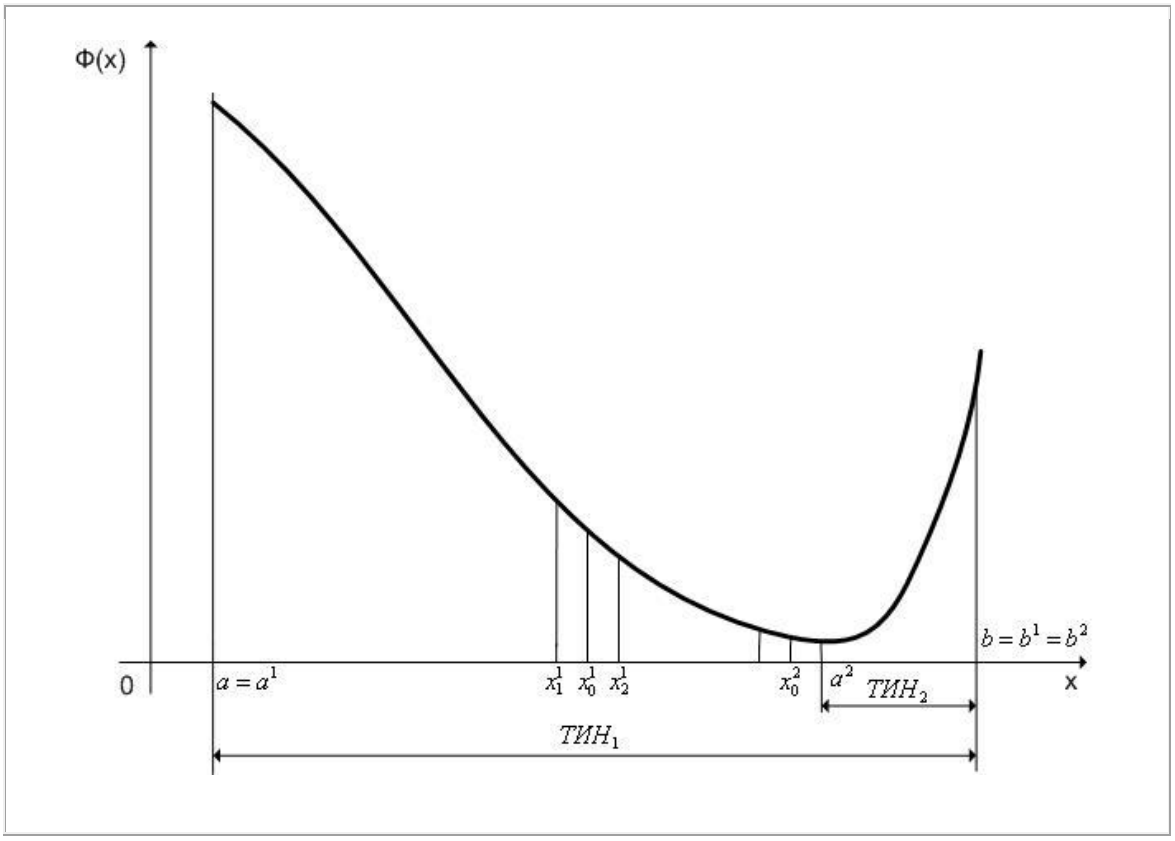

Рис. 2.2.2. Первые две итерации поиска минимума одномерной унимодальной функции с помощью алгоритма равномерного дихотомического поиска

Легко видеть, что после одной итерации алгоритма равномерного поиска ТИН уменьшается в 2 раза. Поэтому количество итераций r, необходимых для нахождения минимума функции с точностью ε_x, находится из условия $\frac{b-a}{2^r} \le \varepsilon_x$.

2.3. Алгоритм Фибоначчи

Рассмотрим следующую задачу условной оптимизации: найти минимум одномерной унимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Числа Фибоначчи задаются следующим рекуррентным уравнением:

$$i_N=i_{N-1}+i_{N-2},\ N\geq 2,\ i_0=i_1=1. \qquad (2.3.1)$$

Числа Фибоначчи $i_0,...,i_9$ приведены в таблице 2.3.1.

Таблица 2.3.1

N	0	1	2	3	4	5	6	7	8	9	...
i_N	1	1	2	3	5	8	13	21	34	55	...

Общее выражение для N-го числа Фибоначчи можно получить из решения уравнения (2.3.1):

$$i_N=\frac{\left(\frac{1}{\tau}\right)^{N+1}-(-\tau)^{N+1}}{\sqrt{5}},\ \text{где}\ \tau=\frac{\sqrt{5}-1}{2}\approx 0{,}618$$

При больших значениях N членом $(-\tau)^{N+1}$ можно пренебречь. При этом

$$i_N\approx\frac{\left(\frac{1}{\tau}\right)^{N+1}}{\sqrt{5}} \qquad (2.3.2)$$

Отсюда следует, что $\frac{i_{N-1}}{i_N}\approx\tau$. Т.е. отношение двух соседних чисел Фибоначчи примерно постоянно и равно τ.

Алгоритм Фибоначчи относится к классу поисковых методов оптимизации и включает в себя два этапа.

Алгоритм Фибоначчи.

Первый этап состоит из $(N-1)$-й итерации для $r=1,2,...,N-1$. Рассмотрим схему r-й итерации, когда $\text{ТИН}_r=\left[a^r,b^r\right]$:

1. Вычисляем величины $x_1^r=a^r+\left|\text{ТИН}_r\right|\frac{i_{N-1-r}}{i_{N+1-r}}$, $x_2^r=a^r+\left|\text{ТИН}_r\right|\frac{i_{N-r}}{i_{N+1-r}}$.

2. Вычисляем значения $\Phi\left(x_1^r\right),\Phi\left(x_2^r\right)$ функции $\Phi(x)$.

3. Если $\Phi\left(x_1^r\right)<\Phi\left(x_2^r\right)$, то выполняем присваивания $a^{r+1}=a^r$, $b^{r+1}=x_2^r$, $\text{ТИН}_{r+1}=\left[a^{r+1},b^{r+1}\right]$. Иначе - выполняем присваивания $a^{r+1}=x_1^r$, $b^{r+1}=b^r$, $\text{ТИН}_{r+1}=\left[a^{r+1},b^{r+1}\right]$.

Алгоритм Фибоначчи обладает тем свойством, что после выполнения $(N-1)$-й итерации имеет место следующая ситуация: $x_1^{N-1}=x_2^{N-1}=x^{N-1}$. Т.е. в результате $(N-1)$-й итерации сужение текущего интервала неопределенности не происходит: $\text{ТИН}_{N-1}=\left[a^{N-1},b^{N-1}\right]=\text{ТИН}_{N-2}=\left[a^{N-2},b^{N-2}\right]$.

Второй этап призван решить по какую сторону от точки x^{N-1} лежит точка минимума функции $\Phi(x)$.

Второй этап выполняется по следующей схеме:

1. Находим точку $x^N=x^{N-1}+\delta_x$, где $\delta_x\ll\text{ТИН}_{N-1}$ - свободный параметр алгоритма.

2. Вычисляем значение функции $\Phi\left(x^N\right)$.

3. Если $\Phi\left(x^N\right)>\Phi\left(x^{N-1}\right)$, то выполняем присваивания $\text{ТИН}_N=\left[a^{N-1},x^{N-1}\right]$ Иначе - выполняем присваивания $\text{ТИН}_N=\left[x^{N-1},b^{N-1}\right]$.

В качестве приближенного значения точки минимума x^* с равными основаниями может быть принята любая точка ТИН_N.

Некоторые свойства алгоритма Фибоначчи.

Утверждение 2.3.1. Для любого $r \in [1,...,N-2]$ алгоритм Фибоначчи обладает следующим свойством: одна из точек x_1^{r+1}, x_2^{r+1} совпадает с одной из точек x_1^r, x_2^r (см. рис. 2.3.1).

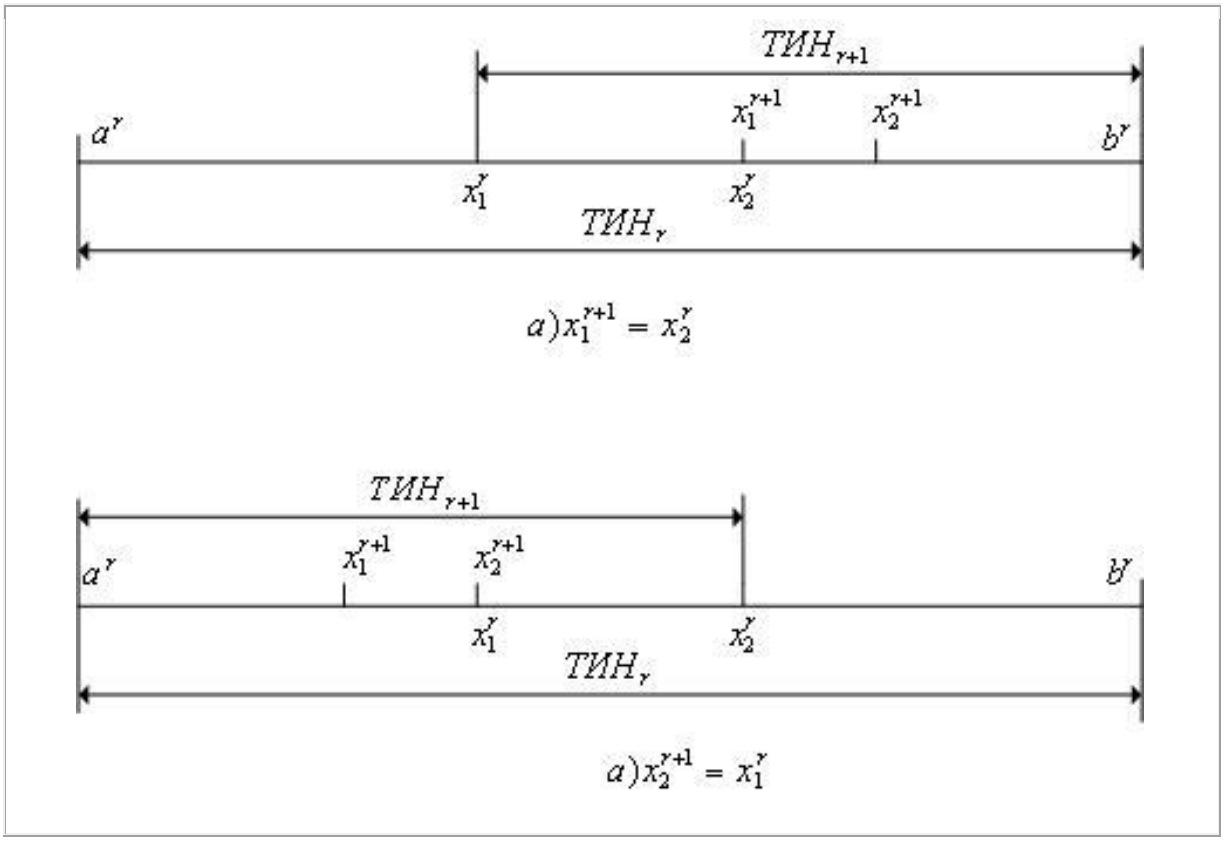

Рис. 2.3.1. К утверждению 2.3.1

Доказательство. Пусть на r-й итерации $\Phi\left(x_1^r\right) \le \Phi\left(x_2^r\right)$ -ситуация б на рис. 2.3.1. В соответствии с алгоритмом Фибоначчи $\text{ТИН}_{r+1} = \left[a^r, x_2^r\right]$ причем, очевидно, $x_1^r = \text{ТИН}_{r+1}$.

Рассмотрим точку

$$x_2^{r+1} = a^r + \left|\text{ТИН}_{r+1}\right| \frac{i_{N-1-r}}{i_{N-r}} = a^r + \left(x_2^r - a^r\right) \frac{i_{N-1-r}}{i_{N-r}}.$$

Подставим сюда значение координаты точки

$$x_2^r = a^r + \left|\text{ТИН}_r\right| \frac{i_{N-r}}{i_{N+1-r}},$$

$$x_2^{r+1} = a^r + \left(a^r + \left|\text{ТИН}_r\right| \frac{i_{N-r}}{i_{N+1-r}} - a^r \right) \frac{i_{N-1-r}}{i_{N-r}} = a^r + \left|\text{ТИН}_r\right| \frac{i_{N-1-r}}{i_{N-r}} = x_1^r.$$

Аналогично проводится доказательство для случая $\Phi\left(x_1^r\right) > \Phi\left(x_2^r\right)$ - ситуация а на рис. 2.3.1.

Указанное свойство алгоритма Фибоначчи позволяет на каждой итерации (кроме первой) производить испытания только в одной точке.

Утверждение 2.3.2. Точки x_1^r, x_2^r расположены симметрично относительно концов текущего интервала неопределенности $\text{ТИН}_r = \left[a^r, b^r \right]$, т.е. расстояние точки x_1^r до точки a^r равно расстоянию точки x_2^r до точки b^r или, что тоже самое, $x_2^r - a^r = b^r - x_1^r$.

Доказательство. В соответствии с алгоритмом Фибоначчи имеем:

$$x_1^r - a^r = a^r + \left(b^r - a^r\right) \frac{i_{N-1-r}}{i_{N+1-r}} - a^r = \left(b^r - a^r\right) \frac{i_{N-1-r}}{i_{N+1-r}},$$

$$b^r - x_2^r = b^r - a^r - \left(b^r - a^r\right) \frac{i_{N-r}}{i_{N+1-r}} - a^r = \left(b^r - a^r\right) \frac{i_{N-1-r} - i_{N-r}}{i_{N+1-r}}.$$

Но из формулы (2.3.1) следует, что $i_{N+1-r} = i_{N-r} + i_{N-1-r}$. Подставляя это в предыдущую формулу, получим $b^r - x_2^r = \left(b^r - a^r\right) \frac{i_{N-1-r}}{i_{N+1-r}} = x_1^r - a^r$.

Утверждение 2.3.3. В результате любой итерации $r \in \left[1, \ldots, N-2\right]$ алгоритма Фибоначчи длина текущего интервала неопределенности уменьшается в $\frac{i_{N-r}}{i_{N+1-r}}$ раз.

Доказательство. Поскольку $x_2^r - a^r = b^r - x_1^r$ (см. утверждения 2.3.2), достаточно рассмотреть только один из интервалов $\left[x_2^r - a^r\right], \left[b^r - x_1^r\right]$. Рассмотрим первый из указанных интервалов:

$$x_2^r - a^r = a^r + \left(b^r - a^r\right)\frac{i_{N-r}}{i_{N+1-r}} - a^r = \left(b^r - a^r\right)\frac{i_{N-r}}{i_{N+1-r}}.$$

Утверждение 2.3.4. При достаточно больших N в результате одной итерации алгоритма Фибоначчи длина текущего интервала неопределенности уменьшается примерно в τ раз.

Доказательство. Справедливость утверждения следует из утверждения 2.3.3 и из того факта, что при достаточно больших N имеем (см. (2.3.2)):

$$\frac{i_{N-r}}{i_{N+1-r}} \approx \frac{\left(\frac{1}{\tau}\right)^{N+1-r}}{\left(\frac{1}{\tau}\right)^{N+2-r}} = \tau$$

Из утверждения 2.3.4 следует, что при достаточно больших N алгоритм Фибоначчи практически идентичен алгоритму золотого сечения (см. следующий параграф 2.4).

Утверждение 2.3.5. В результате N итераций алгоритма Фибоначчи длина текущего интервала неопределенности становится равной

$$\left|\text{ТИН}_N\right| = \frac{b-a}{i_N}$$

Доказательство. Из утверждения 2.3.3 следует, что:

- после первой итерации длина ТИН равна $\left(b-a\right)\frac{i_{N-1}}{i_N}$;
- после второй итерации - $\left(b-a\right)\frac{i_{N-1}}{i_N}\frac{i_{N-2}}{i_{N-1}} = \left(b-a\right)\frac{i_{N-2}}{i_N}$;
- после итерации номер $\left(N-2\right)$ длина ТИН равна $\left(b-a\right)\frac{i_2}{i_N} = \left(b-a\right)\frac{2}{i_N}$
- после итерации номер $\left(N-1\right)$ длина ТИН не меняется;
- после итерации номер N длина ТИН уменьшается в два раза и становится равной $\frac{b-a}{i_N}$.

Поэтому количество итераций N, необходимых для нахождения минимума функции с точностью ε_x, находится из условия $\varepsilon_x \le \frac{b-a}{i_N}$.

Пример 2.3.1. С использованием $N=6$ итераций алгоритма Фибоначчи найдите минимум одномерной унимодальной функции $\Phi(x)=2(x-12)^2+3$, где $x\in[a,b]=[0,13]$. Положите $\delta_x=0.1$.

Решение. Первый этап

Итерация №1. $r=1$ $\text{ТИН}_1=[a_1,b_1]=[a,b]=[0,13]$:

$$\begin{cases} x_1^1=a_1+\left|\text{ТИН}_1\right|\frac{i_{N-2}}{i_N}=13\frac{5}{13}=5, \\ x_2^1=a_1+\left|\text{ТИН}_1\right|\frac{i_{N-1}}{i_{N+1-r}}=13\frac{8}{13}=8; \end{cases}$$

$$\begin{cases} \Phi\left(x_1^1\right)=2(5-12)^2+3=101 \\ \Phi\left(x_2^1\right)=2(8-12)^2+3=39 \end{cases} \Leftrightarrow \Phi\left(x_1^1\right)>\Phi\left(x_2^1\right)$$

Итерация №2. $r=2$ $\text{ТИН}_2=\left[x_1^1,b_1\right]=[5,13]=[a_2,b_2]$:

$$\begin{cases} x_1^2=a_2+\left|\text{ТИН}_2\right|\frac{i_{N-3}}{i_{N-1}}=5+8\frac{3}{8}=8=x_2^1, \\ x_2^2=a_2+\left|\text{ТИН}_2\right|\frac{i_{N-2}}{i_{N-1}}=5+8\frac{5}{8}=10; \end{cases}$$

$$\begin{cases} \Phi\left(x_1^2\right)=\Phi\left(x_2^1\right)=39 \\ \Phi\left(x_2^2\right)=2(10-12)^2+3=11 \end{cases} \Leftrightarrow \Phi\left(x_1^2\right)>\Phi\left(x_2^2\right)$$

Итерация №3. $r=3$ $\text{ТИН}_3=\left[x_1^2,b_2\right]=[8,13]=[a_3,b_3]$:

$$\begin{cases} x_1^3 = a_3 + \left|\text{ТИН}_3\right| \dfrac{i_{N-4}}{i_{N-2}} = 8 + 5\dfrac{2}{5} = 10 = x_2^2, \\ x_2^3 = a_3 + \left|\text{ТИН}_3\right| \dfrac{i_{N-3}}{i_{N-2}} = 8 + 5\dfrac{3}{5} = 11; \end{cases}$$

$$\begin{cases} \Phi\left(x_1^3\right) = \Phi\left(x_2^2\right) = 11 \\ \Phi\left(x_2^3\right) = 2\left(11-12\right)^2 + 3 = 5 \end{cases} \Leftrightarrow \Phi\left(x_1^3\right) > \Phi\left(x_2^3\right)$$

Итерация №4. $r = 4$ $\text{ТИН}_4 = \left[x_1^3, b_3\right] = \left[10,13\right] = \left[a_4, b_4\right]$:

$$\begin{cases} x_1^4 = a_4 + \left|\text{ТИН}_4\right| \dfrac{i_{N-5}}{i_{N-3}} = 10 + 3\dfrac{1}{3} = 11 = x_2^3, \\ x_2^4 = a_4 + \left|\text{ТИН}_4\right| \dfrac{i_{N-4}}{i_{N-3}} = 10 + 3\dfrac{2}{3} = 12; \end{cases}$$

$$\begin{cases} \Phi\left(x_1^4\right) = \Phi\left(x_2^3\right) = 5 \\ \Phi\left(x_2^4\right) = 2\left(12-12\right)^2 + 3 = 3 \end{cases} \Leftrightarrow \Phi\left(x_1^4\right) > \Phi\left(x_2^4\right)$$

Итерация №5. $r = 5$ $\text{ТИН}_5 = \left[x_1^4, b_4\right] = \left[11,13\right] = \left[a_5, b_5\right]$:

$$\begin{cases} x_1^5 = a_5 + \left|\text{ТИН}_5\right| \dfrac{i_{N-6}}{i_{N-4}} = 11 + 1 = 12 = x_2^4, \\ x_2^5 = a_5 + \left|\text{ТИН}_5\right| \dfrac{i_{N-5}}{i_{N-4}} = 11 + 1 = 12 = 12; \end{cases}$$

$$\Phi\left(x_1^5\right) = \Phi\left(x_2^5\right) = \Phi\left(x_2^4\right) = 3$$

Второй этап.

Итерация №6. $r = 6$ $\text{ТИН}_5 = \text{ТИН}_6 = \left[11,13\right]$:

$$\begin{cases} x_1^6 = x_1^5 = x_2^5 = 12, \\ x_2^6 = x_1^5 + \delta_x = 12.1; \end{cases}$$

$$\begin{cases} \Phi\left(x_1^6\right) = \Phi\left(x_1^5\right) = 3 \\ \Phi\left(x_2^6\right) = 2(0.1)^2 + 3 = 3.02 \end{cases} \Leftrightarrow \Phi\left(x_2^6\right) > \Phi\left(x_1^6\right).$$

Ответ. ТИН=$[11,12]$.

2.4. Алгоритм золотого сечения

Рассмотрим следующую задачу условной оптимизации: найти минимум одномерной унимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Свойства золотого сечения.

Рассмотрим интервал $\left[a,b\right]$ (см. рис. 2.4.1).

Говорят, что точка с выполняет золотое сечение интервала $\left[a,b\right]$, если

$$\frac{c-a}{b-a}=\tau \qquad (2.4.1)$$

где $\tau=\frac{\sqrt{5}-1}{2}\approx 0,618$ - решение квадратного уравнения

$$\tau^2+\tau-1=0 \qquad (2.4.2)$$

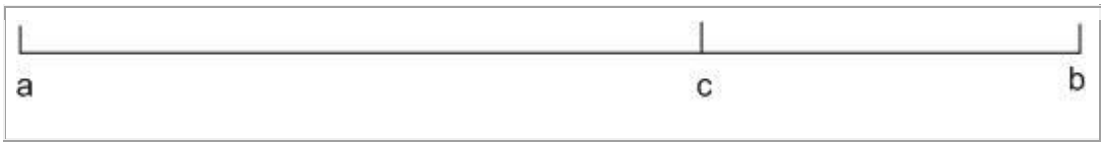

Рис. 2.4.1. К определению золотого сечения отрезка

Из определения золотого сечения следует, что $\frac{b-c}{b-a}=1-\tau$.

Действительно, $\frac{b-c}{b-a}=\frac{(b-a)-(c-a)}{b-a}=1-\frac{c-a}{b-a}=1-\tau$.

Алгоритм золотого сечения.

Алгоритм золотого сечения относится к классу последовательных методов поиска.

1. Выполняем присваивания $r=1$, $a^1=a$, $b^1=b$, ТИН$_1=\left[a^1,b^1\right]$.

2. Вычисляем величины (см. рис. 2.4.2)

$$x_1^r=b^r-\left(b^r-a^r\right)\tau,\quad x_2^r=a^r+\left(b^r-a^r\right)\tau \qquad (2.4.3)$$

3. Вычисляем значения $\Phi\left(x_1^r\right), \Phi\left(x_2^r\right)$ функции $\Phi(x)$.

4. Если $\Phi\left(x_1^r\right) < \Phi\left(x_2^r\right)$, то выполняем присваивания $a^{r+1} = a^r$, $b^{r+1} = x_2^r$, $\text{ТИН}_{r+1} = \left[a^{r+1}, b^{r+1}\right]$. Иначе - выполняем присваивания $a^{r+1} = x_1^r$, $b^{r+1} = b^r$, $\text{ТИН}_{r+1} = \left[a^{r+1}, b^{r+1}\right]$.

5. Если $\left|\text{ТИН}_{r+1}\right| \le \varepsilon_x$, то заканчиваем вычисления. Иначе - выполняем присваивание $r = r+1$ и переходим на п.2. Здесь ε_x – требуемая точность решения.

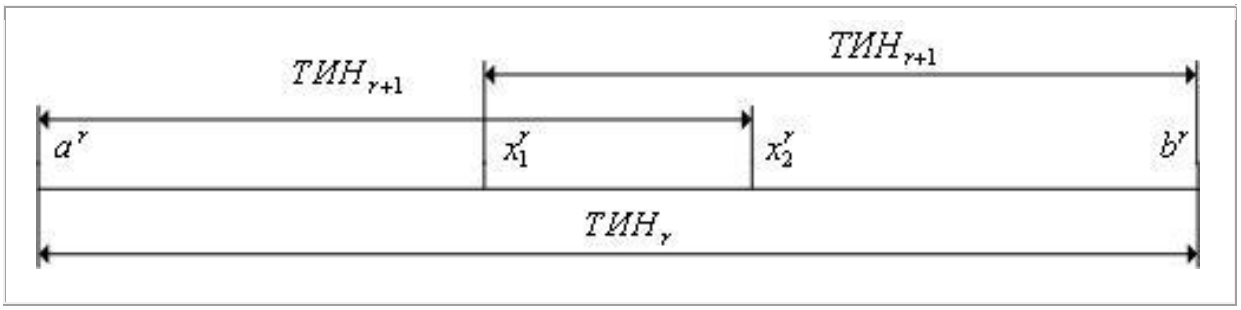

Рис. 2.4.2. К определению величин x_1^r, x_2^r

В качестве приближенного значения точки минимума x^* с равными основаниями может быть принята любая точка последнего текущего интервала неопределенности.

Некоторые свойства алгоритма золотого сечения.

Утверждение 2.4.1. Точки x_1^r, x_2^r расположены симметрично относительно концов текущего интервала неопределенности.

Действительно, из (2.4.3) следует, что точка x_1^r отстоит от точки b^r на величину $\left(b^r - a^r\right)\tau$; точка x_2^{r+1} отстоит от точки a^r на ту же величину.

Утверждение 2.4.2. Для любого $r \ge 1$ алгоритм золотого сечения обладает следующим свойством: одна из точек x_1^{r+1}, x_2^{r+1} совпадает с одной из точек x_1^r, x_2^r.

Доказательство. Пусть на r-й итерации $\Phi\left(x_1^r\right)<\Phi\left(x_2^r\right)$. В соответствии с алгоритмом золотого сечения $\text{ТИН}_{r+1}=\left[a^r, x_2^r\right]$ причем, очевидно, $x_1^r \in \text{ТИН}_{r+1}$. Для того, чтобы доказать справедливость утверждения достаточно показать, что верно отношение

$$\frac{x_1^r-a^r}{x_2^r-a^r}=\tau \qquad (2.4.4)$$

Из соотношений (2.4.3) следует, что

$$b^r-x_1^r=\left(b^r-a^r\right)\tau$$
$$b^r-x_1^r-a^r+a^r=\left(b^r-a^r\right)\tau$$
$$\left(b^r-a^r\right)-\left(x_1^r-a^r\right)=\left(b^r-a^r\right)\tau$$
$$x_1^r-a^r=\left(b^r-a^r\right)(1-\tau).$$

Аналогично имеем $x_2^r-a^r=\left(b^r-a^r\right)\tau$.

Разделив первый из этих результатов на второй, получим

$$\frac{x_1^r-a^r}{x_2^r-a^r}=\frac{1-\tau}{\tau}. \qquad (2.4.5)$$

Из уравнения (2.4.2) следует, что $1-\tau=\tau^2$. Отсюда и из (2.4.5) следует справедливость (2.4.4).

Аналогично проводится доказательство для случая $\Phi\left(x_1^r\right)>\Phi\left(x_2^r\right)$.

Указанное свойство алгоритма золотого сечения позволяет на каждой итерации (кроме первой) производить испытания только в одной точке.

Из схемы алгоритма золотого сечения имеем.

Утверждение 2.4.3. В результате одной итерации алгоритма золотого сечения длина текущего интервала неопределенности сокращается в τ раз. Поэтому количество итераций N, необходимых для нахождения минимума функции с точностью ε_x, находится из условия $\varepsilon_x \le \left(b-a\right)\tau^N$.

Из утверждения 2.4.3 и результатов предыдущего параграфа следует, что при достаточно больших N алгоритм Фибоначчи практически идентичен алгоритму золотого сечения (см. параграф 2.3).

2.5. Метод хорд (метод секущих)

Рассмотрим следующую задачу условной оптимизации: найти минимум одномерной унимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Необходимым условием минимума функции $\Phi(x)$ является условие

$$\Phi'(x)=0,\ x\in\left[a,b\right]. \qquad (2.5.1)$$

Рассматриваемый в данном параграфе метод первого порядка решения задачи основан на поиске стационарной точки функции $\Phi(x)$, т.е. на решении задачи (2.5.1), которая представляет собой задачу нахождения корней функции $\Phi'(x)$, принадлежащих интервалу $\left[a,b\right]$.

Метод хорд ориентирован на нахождение корня уравнения (2.5.1) в случае, когда на границах интервала $\left[a,b\right]$ знаки производной $\Phi'(x)$ различны. Такая ситуация, очевидно, возможна, если точка минимума функции $\Phi(x)$ является внутренней точкой интервала $\left[a,b\right]$- см. рис. 2.5.1.

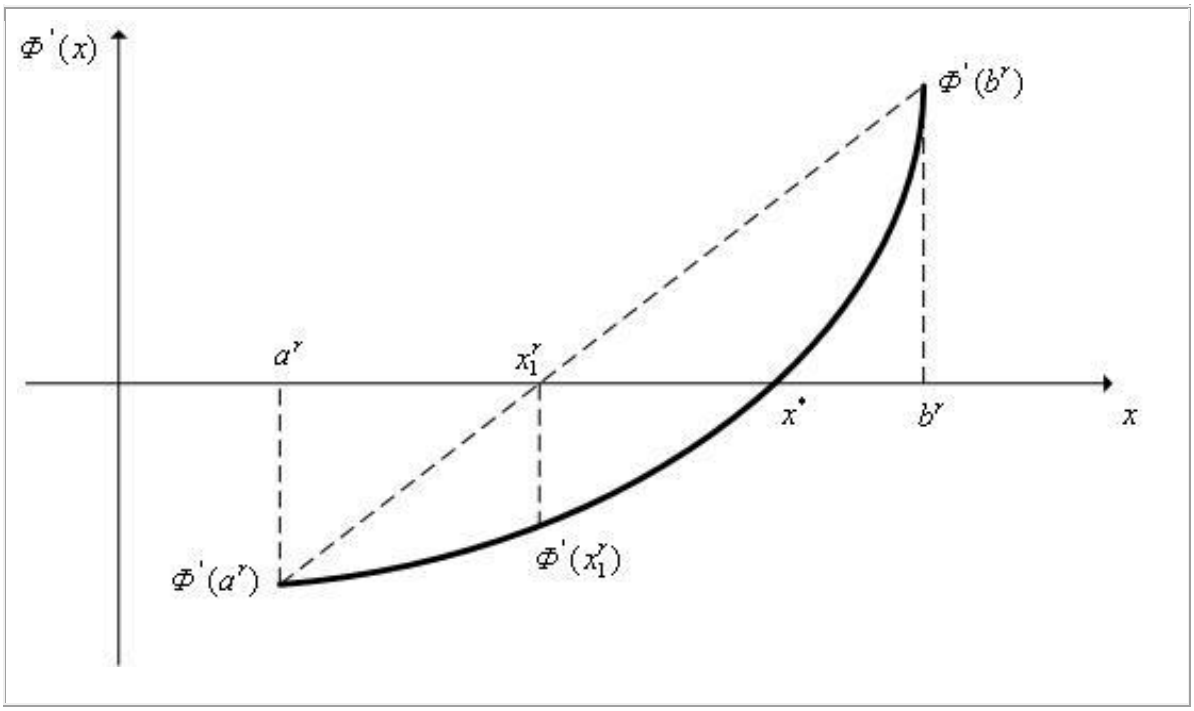

Рис. 2.5.1. К схеме метода хорд

Нам далее понадобится значение x_1^r. Из подобия треугольника $a^r, x_1^r, \Phi'(a^r)$ и треугольника $b^r, x_1^r, \Phi'(b^r)$ имеем $\frac{\Phi'(b^r)}{\Phi'(a^r)} = \frac{b^r - x_1^r}{x_1^r - a^r}$.

Отсюда следует, что

$$x_1^r = b^r - \frac{\Phi'(b^r)(b^r - a^r)}{\Phi'(b^r) - \Phi'(a^r)} \qquad (2.5.2)$$

Алгоритм метода хорд.

1. Выполняем присваивания $r=1,\ a^1 = a,\ b^1 = b$.
2. Вычисляем значения производных $\Phi'(a^r),\ \Phi'(b^r)$.
3. Если производные $\Phi'(a^r),\ \Phi'(b^r)$ имеют одинаковые знаки – завершаем вычисления (точки a, b выбраны неверно).
4. По формуле (2.5.2) вычисляем приближение x_1^r к стационарной точке функции $\Phi(x),$ значение производной $\Phi'(x_1^r)$.
5. Если $|\Phi'(x_1^r)| \le \varepsilon$, где ε – требуемая точность решения, то принимаем $x^* \approx x_1^r$ и заканчиваем вычисления.
6. Если производные $\Phi'(a^r),\ \Phi'(x_1^r)$ имеют разные знаки, то выполняем присваивания $a^{r+1} = a^r,\ b^{r+1} = x_1^r,\ r = r+1$ и переходим на п.4.
7. Если производные $\Phi'(x_1^r),\ \Phi'(b^r)$ имеют разные знаки, (как на рис. 2.5.1.) то выполняем присваивания $a^{r+1} = x_1^r,\ b^{r+1} = b^r,\ r = r+1$ и переходим на п.4.

В случае квадратичной функции $\Phi(x)$ производная этой функции $\Phi'(x)$ линейна. Поэтому метод хорд гарантирует нахождение стационарной точки функции $\Phi(x)$ всего за одну итерацию.

Поскольку поиск заканчивается при выполнении условия $\left|\Phi'\left(x_1^r\right)\right| \le \varepsilon$, возможно появление ложных корней. Например, для уравнения $x^2 + 0.0001 = 0$ ложный корень $x = 0$ появляется в том случае, если $\varepsilon > 0.0001$. В подобных случаях увеличивая точность поиска, можно избавить от ложных корней. Однако, возможны уравнения, для которых такой подход не приводит к успеху. Например, уравнение $\frac{1}{x} = 0$ не имеет действительных корней, однако для сколь угодно малого ε найдется точка, удовлетворяющая условию окончания поиска.

Возможна модификация метода хорд, когда значения производной $\Phi'(x)$ вычисляются приближенно с использованием первых разностей. В этом случае метод становится прямым (нулевого порядка).

2.6. Метод касательных (метод Ньютона решения нелинейных уравнений)

Рассмотрим следующую задачу условной оптимизации: найти минимум одномерной унимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Метод касательных ориентирован на нахождение корня уравнения $\Phi'(x)=0, x\in\left[a,b\right]$ в случае, когда на границах интервала $\left[a,b\right]$ знаки производной $\Phi'(x)$ различны. Такая ситуация, очевидно, возможна, если точка минимума функции $\Phi(x)$ является внутренней точкой интервала $\left[a,b\right]$ (см. рис. 2.6.1), метод требует, чтобы функция $\Phi(x)$ была определена и дважды дифференцируема в области допустимых значений $D=\left[a,b\right]$.

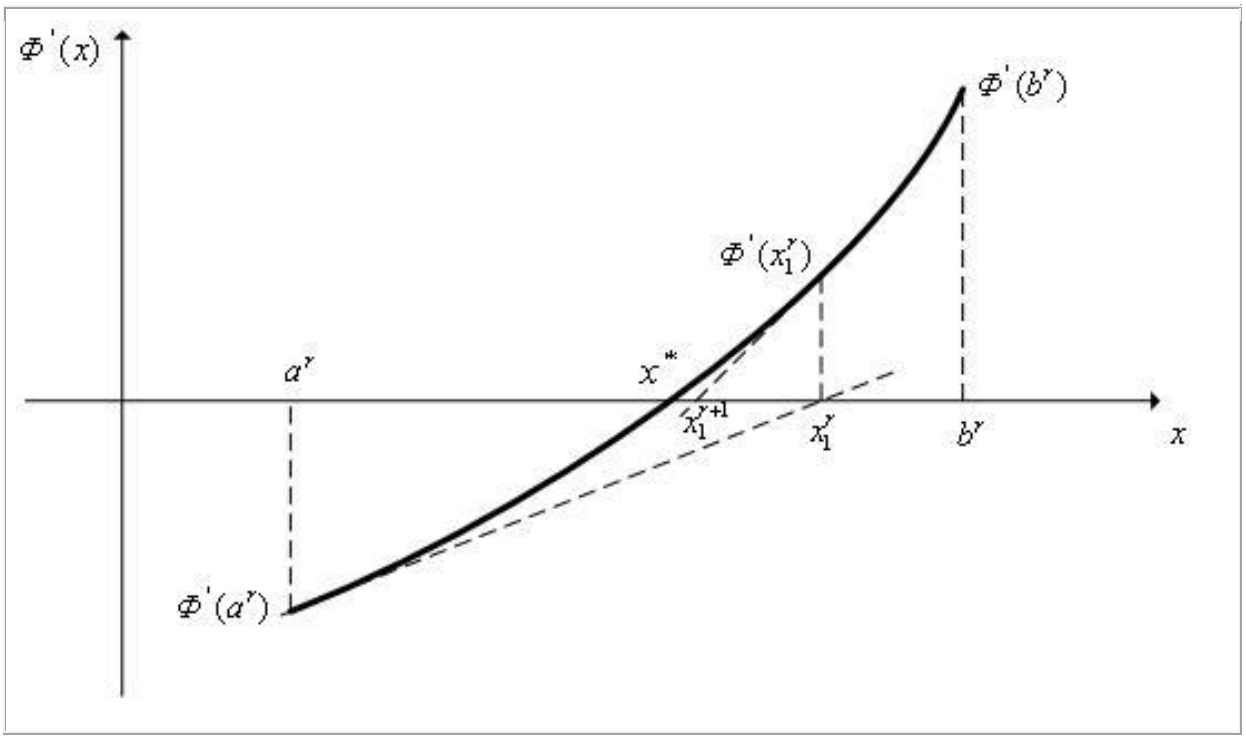

Рис. 2.6.1. К схеме метода касательных

Нам далее понадобится значение x_1^r. Линейная функция, аппроксимирующая функцию $\Phi'(x)$ в точке a^r, записывается в виде

$$\tilde{\Phi}'(x,a^r)=\Phi'(a^r)+\Phi''(a^r)(x-a^r). \qquad (2.6.1)$$

Приравняв правую часть уравнения (2.6.1) к нулю, получим

$$x_1^r=a^r-\frac{\Phi'(a^r)}{\Phi''(a^r)} \qquad (2.6.2)$$

Алгоритм метода касательных.

1. Выполняем присваивания $r=1,\ a^1=a,\ b^1=b$.

2. Вычисляем значения производных $\Phi'(a^r),\ \Phi'(b^r)$.

3. Если производные $\Phi'(a^r),\ \Phi'(b^r)$ имеют одинаковые знаки – завершаем вычисления (точки a,b выбраны неверно).

4. По формуле (2.6.2.) вычисляем приближение x_1^r к стационарной точке функции $\Phi(x)$ и значение $\Phi'(x_1^r)$.

5. Если $\left|\Phi'(x_1^r)\right|\le\varepsilon$, где ε – требуемая точность решения, то принимаем $x^*\approx x_1^r$ и заканчиваем вычисления.

6. Если разные знаки имеют производные $\Phi'(a^r),\ \Phi'(x_1^r)$, (как на рис. 2.6.1) то выполняем присваивания $a^{r+1}=a^r,\ b^{r+1}=x_1^r,\ r=r+1$ и переходим на п.4.

7. Если разные знаки имеют производные $\Phi'(x_1^r),\ \Phi'(b^r)$, то выполняем присваивания $a^{r+1}=x_1^r,\ b^{r+1}=b^r,\ r=r+1$ и переходим на п.4.

В случае квадратичной функции $\Phi(x)$ производная этой функции $\Phi'(x)$ линейна. Поэтому метод касательных гарантирует нахождение стационарной точки функции $\Phi(x)$ всего за одну итерацию.

Возможна модификация метода касательных, когда значения производной $\Phi'(x)$ вычисляются приближенно с использованием первых разностей. В этом случае метод становится прямым (нулевого порядка).

2.7. Метод перебора

Рассматривается следующая одномерная задача условной глобальной оптимизации): найти минимум, вообще говоря, многоэкстремальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

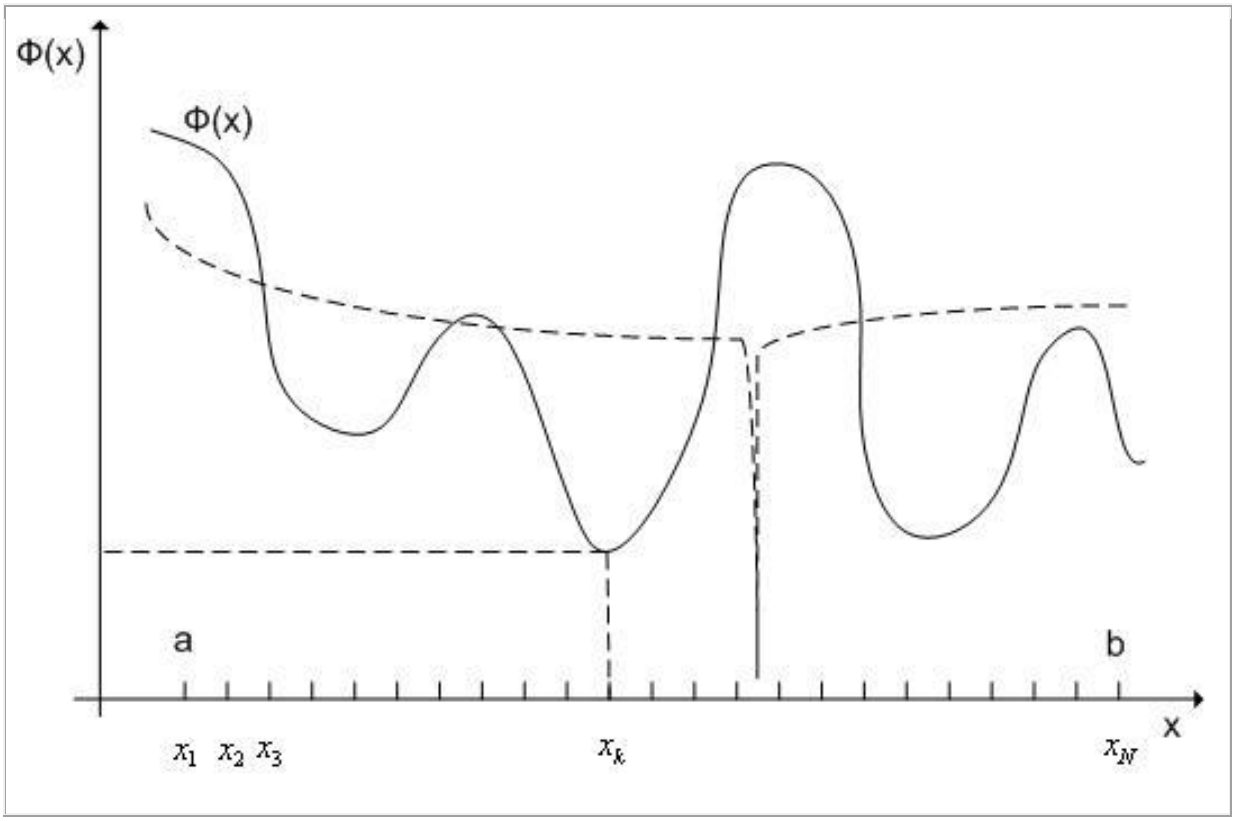

Рис. 2.7.1. К схеме метода перебора

Алгоритм метода перебора.

1. Покрываем интервал $\left[a,b\right]$ некоторой сеткой с узлами $x_i, i\in\left[1,...,N\right]$.

2. Производим испытание в точке $x_1=\tilde{x}^*$, т.е. вычисляем значения $\tilde{\Phi}^*$ функции $\Phi(x)$ в этой точке.

3. Полагаем $r=2$.

4. Производим испытание в точке x_r - вычисляем значение $\Phi_r=\Phi(x_r)$ функции $\Phi(x)$ в этой точке.

5. Если $\Phi_r < \tilde{\Phi}^*$, то выполняем присваивания $\tilde{x}^* = x_r$, $\tilde{\Phi}^* = \Phi_r$.

6. Если $r < N$, то выполняем присваивание $r = r + 1$ и переходим на п.4. Иначе - заканчиваем вычисления.

7. Принимаем $\tilde{x}^*$ в качестве приближенного значения точки глобального минимума функции $\Phi(x)$ на интервале $[a,b]$ или каким-либо из рассмотренных одномерных методов локальной оптимизации организуем в окрестности точки $\tilde{x}^*$ поиск локального минимума этой функции.

При выборе количества узлов сетки $x_i, i \in [1,...,N]$ можно исходить из требуемой точности решения ε_x – максимальный шаг сетки принять равным этой величине.

Отметим, что метод перебора, как и любой другой метод глобальной оптимизации, при отсутствии априорной информации о свойствах минимизируемой функции не гарантирует нахождение глобального минимума (см. пунктирный график на рис. 2.7.1).

2.8. Одномерный метод Монте-Карло

Рассматривается следующая одномерная задача условной глобальной оптимизации): найти минимум, вообще говоря, многоэкстремальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Алгоритм метода Монте-Карло.

1. Генерируем с помощью программного генератора случайных чисел, равномерно распределенных в интервале $\left[a,b\right]$, случайное число $x_1=\tilde{x}^*$.
2. Производим испытание в точке $x_1=\tilde{x}^*$ - вычисляем значения $\tilde{\Phi}^*$ функции $\Phi(x)$ в этой точке.
3. Полагаем $r=2$.
4. Аналогично п. 1 генерируем случайное число $x_r\in\left[a,b\right]$.
5. Производим испытание в точке x_r - вычисляем значение $\Phi_r=\Phi(x_r)$ функции $\Phi(x)$ в этой точке.
6. Если $\Phi_r<\tilde{\Phi}^*$, то выполняем присваивания $\tilde{x}^*=x_r$, $\tilde{\Phi}^*=\Phi_r$.
7. Если $r<N$, то выполняем присваивание $r=r+1$ и переходим на п. 4. Иначе - заканчиваем вычисления. Здесь N – количество испытаний.
8. Принимаем $\tilde{x}^*$ в качестве приближенного значения точки глобального минимума функции $\Phi(x)$ на интервале $\left[a,b\right]$ или каким-либо из рассмотренных одномерных методов локальной оптимизации организуем в окрестности точки $\tilde{x}^*$ поиск локального минимума этой функции.

При достаточно большом N метода гарантирует нахождение глобального минимума с высокой вероятностью.

2.9. Метод выделения интервалов унимодальности

Рассмотрим одномерную задачу условной глобальной оптимизации: найти минимум одномерной многоэкстремальной функции $\Phi(x)$, определенной в **замкнутой** области допустимых значений $D=\left[a,b\right]$ и имеющей в этой области конечное число минимумов $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

Метод выделения интервалов унимодальности функции $\Phi(x)$ требует априорного знания оценки $d>0$ минимального расстояния между локальными минимумами этой функции.

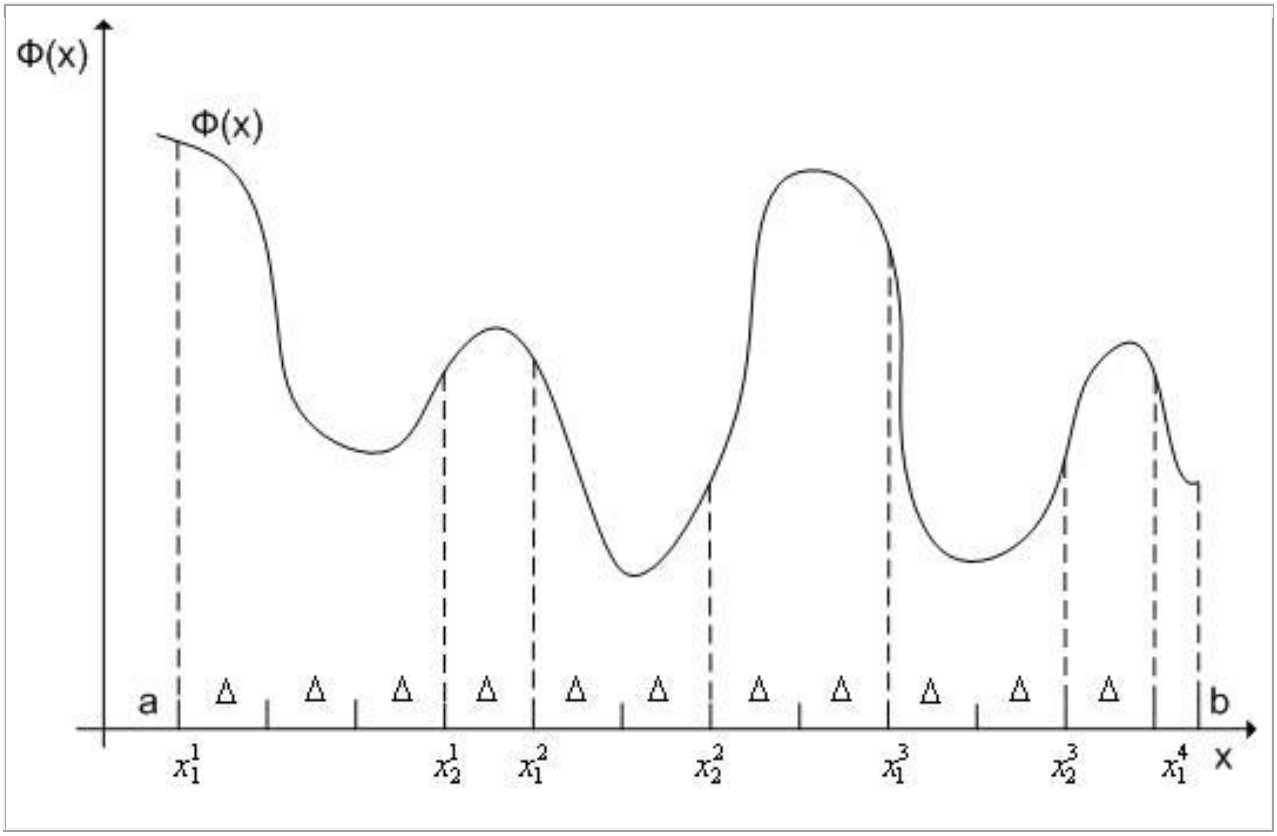

Рис. 2.9.1. К схеме выделения интервалов унимодальности

Алгоритм метода выделения интервалов унимодальности.

1. Полагаем $r=1$.
2. Если функция $\Phi(x)$ в точке a возрастает, то полагаем $x_0^r=a$ переходим на п. 4.

3. Если функция $\Phi(x)$ в точке a убывает, то полагаем $x_1^r = a$ и переходим на п. 5.
4. Последовательно увеличивая x на величину Δ по формуле $x = x_0^r + i\Delta \le b$, $i = 1,2,...$, находим первую точку x_1^r, в которой функция $\Phi(x)$ убывает. Здесь и далее Δ - величина в несколько раз меньшая величины d.
5. Последовательно увеличивая x на величину Δ по формуле $x = x_1^r + i\Delta \le b$, $i = 1,2,...$, находим первую точку x_2^r, в которой функция $\Phi(x)$ возрастает.
6. В качестве r-го интервала унимодальности принимаем интервал $\left[x_1^r, x_2^r\right]$.
7. Если интервал $\left[a,b\right]$ исчерпан, переходим на п.8, иначе полагаем $x_0^{r+1} = x_2^r$, $r = r+1$ и переходим на п.4.
8. Положим, что общее количество найденных интервалов унимодальности функции $\Phi(x)$ равно $N-1$. Каким-либо одномерным методом локальной оптимизации находим локальный минимум функции $\Phi(x)$ в каждом из $N-1$ интервалов унимодальности. Обозначим точки этих минимумов $\tilde{x}_i^*, i \in \left[1,...,N-1\right]$. Соответствующие значения функции $\Phi(x)$ обозначим $\tilde{\Phi}_i^*, i \in \left[1,...,N-1\right]$. Добавим к точкам $\tilde{x}_i^*$ точки $\tilde{x}_0^*$, $\tilde{x}_N^*$ и вычислим соответствующие значения $\tilde{\Phi}_0^*$, $\tilde{\Phi}_N^*$ функции $\Phi(x)$.
9. Найдем минимальную из величин $\tilde{\Phi}_i^*, i \in \left[1,...,N\right]$ и соответствующее значение аргумента: $\min\limits_{i \in [0,N]} \tilde{\Phi}_i^* = \tilde{\Phi}_k^* = \Phi\left(\tilde{x}_k^*\right)$.
10. В качестве решения задачи глобальной оптимизации примем точку $\left(\tilde{x}_k^*, \tilde{\Phi}_k^*\right)$.

На рис. 2.9.1 интервалами унимодальности являются интервалы $\left[x_1^1, x_2^1\right]$, $\left[x_1^2, x_2^2\right]$, $\left[x_1^3, x_2^3\right]$.

Для определения того, возрастает или убывает в данной точке функция $\Phi(x)$, может использоваться ее первая разность в этой точке $\Delta\Phi_x = \Phi(x) - \Phi(x + \Delta x)$, где Δx - некоторая малая величина. А именно, если $\Delta\Phi_x > 0$, то функция возрастает в точке x; иначе – убывает. Заметим, что при этом в каждой точке требуется выполнить дополнительное испытание функции $\Phi(x)$.

Если функция $\Phi(x)$ непрерывно дифференцируема в интервале $\left[a,b\right]$, то для определения того, возрастает или убывает в данной точке эта функция, можно, очевидно, использовать значения первой производной функции $\Phi(x)$ в этой точке. А именно, если $\Phi'(x) > 0$, то в точке x функция $\Phi(x)$ возрастает; в противном случае – убывает.

Замечание 2.9.1. Если априорная оценка d минимального расстояния между локальными минимумами функции $\Phi(x)$ отсутствует, то никаких оснований полагать, что в интервалах, выделенных с помощью рассмотренного алгоритма, функция $\Phi(x)$ является унимодальной функцией. Пусть, например, функция $\Phi(x)$ на интервале $\left[c,d\right]$ постоянна (см. рис. 2.9.2). Если к такой функции применить алгоритм выделения интервалов унимодальности с любым $\Delta > 0$, то в качестве интервала унимодальности будет выделен интервал $\left[x_1^1, x_2^1\right] \supset \left[c,d\right]$, на котором функция $\Phi(x)$ имеет бесконечное количество минимумов, т.е. не является унимодальной функцией.

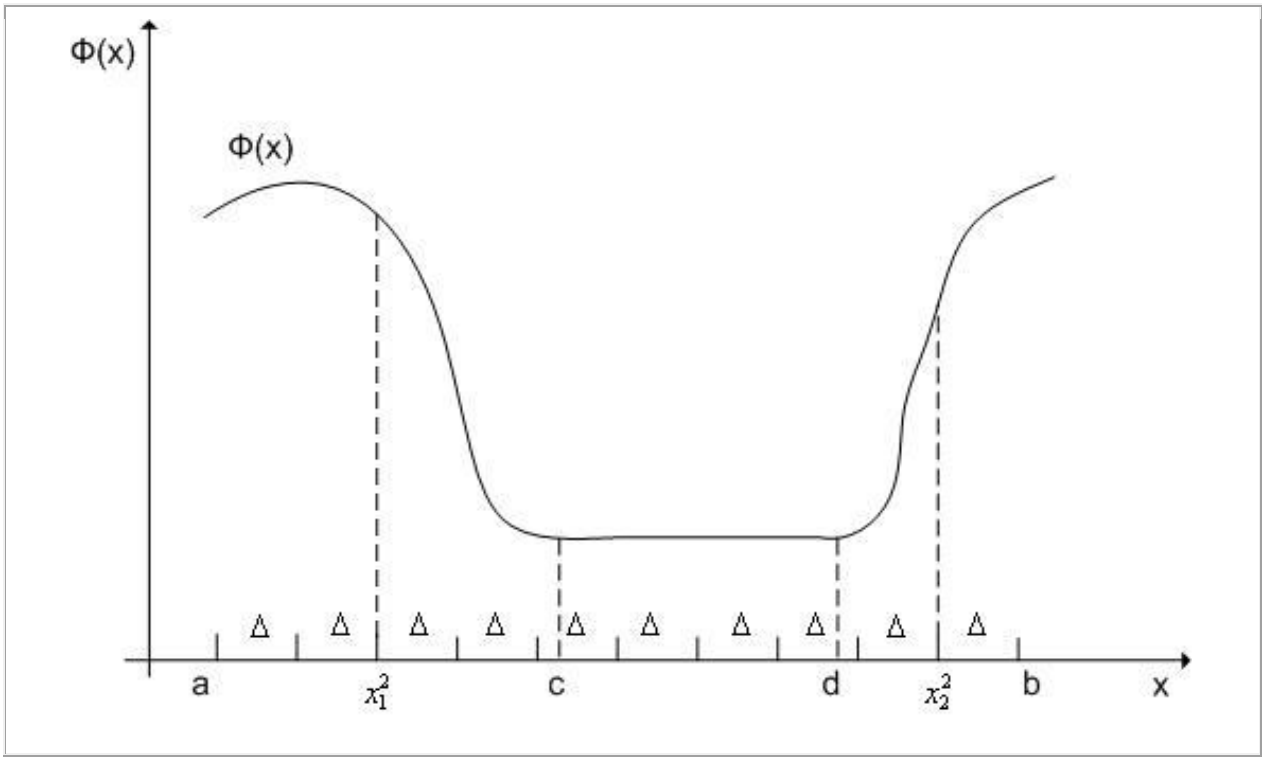

Рис.2.9.2. К замечанию 2.9.1

2.10. Метод аппроксимирующих моделей

Рассмотрим одномерную задачу условной глобальной оптимизации): найти минимум одномерной мультимодальной функции $\Phi(x)$, определенной в замкнутой области допустимых значений $D=\left[a,b\right]$, $\min\limits_{x\in[a,b]}\Phi(x)=\Phi(x^*)$.

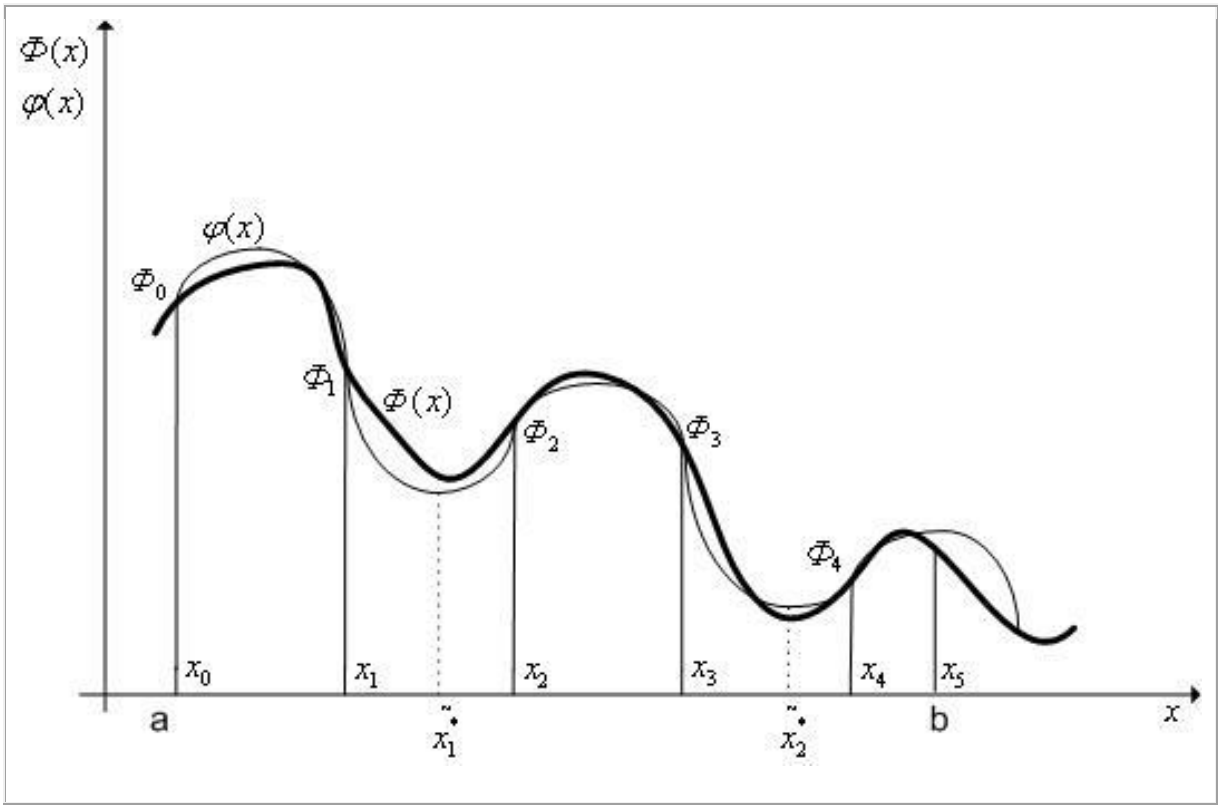

Рис. 2.10.1. К схеме метода аппроксимирующих моделей. $N=5$

Алгоритм метода аппроксимирующих моделей.

1. Покрываем интервал $\left[a,b\right]$ некоторой сеткой с узлами $x_i\in\left[a,b\right], x_i\neq x_j,\ i,j\in\left[0,...,N\right]$ и производим испытания в точках $x_i, i\in\left[0,...,N\right]$, т.е. вычисляем значения функции $\Phi(x)$ в этих точках $\Phi(x_i)=\Phi_i, i\in\left[0,...,N\right]$.

2. Строим аппроксимирующую функцию $\varphi(x)$, проходящую через точки $(x_i,\Phi_i), i\in\left[0,...,N\right]$. Эту функцию принято называть **математической моделью минимизируемой функции** $\Phi(x)$ или модельной функцией.

3. Оцениваем адекватность построенной модели $\varphi(x)$. Для этого:

- производим дополнительные испытания функции $\Phi(x)$ в некоторых точках $x_k \in [a,b], k \in [1,...,M]$;
- вычисляем значения модельной функции $\varphi(x)$ и функции $\Phi(x)$ в этих точках $\varphi(x_k), \Phi(x_k), k \in [1,...,M]$;
- вычисляем погрешность аппроксимации, например, $\max\limits_{k \in [1,...,M]} \left| \varphi(x_k) - \Phi(x_k) \right|$.

4. Если погрешность аппроксимации превышает заданную, то по результатам всех предшествующих испытаний строим новую модельную функцию $\varphi(x)$ и переходим на п. 3.

5. Определяем положение глобального минимума модельной функции $\varphi(x)$, который или принимается в качестве глобального минимума функции $\Phi(x)$, или уточняется с помощью какого-либо метода локальной оптимизации.

На рис. 2.10.1 x_1^*, x_2^* - точки локального минимума модельной функции $\varphi(x)$; точка $\tilde{x}_2^*$ - приближенное значение точки глобального минимума функции $\Phi(x)$ на интервале $[a,b]$.

В качестве модельных функций $\varphi(x)$ чаще всего используют полиномы.

Рассмотрим использование в качестве модельной функции полиномов.

Аппроксимирующий полином Лагранжа

Будем искать аппроксимирующий полином в виде

$$\varphi(x) = \Phi(x_0)\psi_0(x) + \Phi(x_1)\psi_1(x) + ... + \Phi(x_N)\psi_N(x) = \sum_{i=0}^{N} \Phi(x_i)\psi_i(x),$$

где $\psi_i(x), i \in [0,...,N]$ - неизвестные полиномы от x, независящие от аппроксимируемой функции $\varphi(x)$.

Из того условия, что модельная функция $\varphi(x)$ должна совпадать с аппроксимируемой функцией $\Phi(x)$ в узлах сетки $x_i, i \in [0,...,N]$, имеем систему из $N+1$ равенств

$$\varphi(x_i) = \Phi(x_i) = \Phi(x_0)\psi_0(x_i) + \Phi(x_1)\psi_1(x_i) + ... + \Phi(x_N)\psi_N(x_i), \quad i \in [0,...,N] \quad (2.10.1)$$

Для выполнения равенств (2.10.1) полиномы $\psi_i(x), i \in [0,...,N]$, очевидно, должны удовлетворять условиям

$$\psi_i(x_j) = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases}, \quad (2.10.2)$$

или, другими словами, полином $\psi_i(x), i \in [0,...,N]$ должен иметь в качестве корней все числа $x_i, i \in [0,...,N]$, кроме числа x_j, а при $x = x_j$ должен иметь значение, равное единице.

Условию (2.10.2) удовлетворяют только полиномы вида

$$\psi_i(x) = A_i \cdot (x_i - x_0)(x_i - x_1)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_N),$$

где A – неизвестная константа.

Найдем эту константу из условия $\psi_i(x_i) = 1$:

$$1 = A_i \cdot (x_i - x_0)(x_i - x_1)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_N);$$

$$A_i = \frac{1}{(x_i - x_0)(x_i - x_1)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_N)}.$$

Таким образом,

$$\psi_i(x) = \frac{(x - x_0)(x - x_1)...(x - x_{i-1})(x - x_{i+1})...(x - x_N)}{(x_i - x_0)(x_i - x_1)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_N)} = \prod_{j=0, j \neq i}^{N} \frac{(x - x_j)}{(x_i - x_j)} \quad (2.10.3)$$

и искомый аппроксимирующий полином определяют выражением

$$\varphi(x) = \sum_{i=0}^{N} \Phi(x_i)\psi_i(x) = L_N(x). \quad (2.10.4)$$

Полином (2.10.4) называется аппроксимирующим полиномом Лагранжа.

Использование аппроксимирующего полинома Лагранжа (2.10.4) в качестве модельной функции идейно очень просто, но обладает существенным недостатком. Пусть после построения этого полинома на сетке $x_i, i \in [0,...,N]$ и проверке его адекватности выясняется, что погрешность аппроксимации превышает заданную. Тогда, в соответствии с рассмотренной выше схемой метода, необходимо построить новый полином Лагранжа на сетке, полученной объединением сеток $x_i, i \in [0,...,N]$, $x_k, k \in [0,...,M]$, что требует пересчета всех посчитанных ранее функций $\psi_i(x), i \in [0,...,N]$. От этого недостатка свободна модификация аппроксимирующего полинома Лагранжа – аппроксимирующий полином Ньютона.

Нахождение стационарных точек аппроксимирующего полинома.

После построения аппроксимирующего полинома возникает задача нахождения стационарных точек функции $\varphi(x)$ (см. схему метода).

Поскольку аппроксимирующий полином непрерывен и, по крайней мере, один раз непрерывно дифференцируем, его стационарные точки удобно искать как нули первой производной – т.е. как корни уравнения

$$\varphi'(x) = 0 \qquad (2.10.5)$$

Для поиска корней уравнения чаще всего используют метод хорд и метод касательных, использующие линейную интерполяцию функции $\varphi'(x)$ (см. параграфы 2.5 и 2.6), а также другие методы.