2.5. НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ (НП – ПРОГРАММИРОВАНИЕ)

Постановка НП-задачи формулируется как нахождения оптимума целевой функции *f(X)* при ограничениях и задаётся моделью вида

$$\begin{cases} f(x_1,x_2,...,x_n) \to \max, \\ g_1(x_1,x_2,...,x_n) \ge 0, \\ g_2(x_1,x_2,...,x_n) \ge 0, \\ ... \\ g_m(x_1,x_2,...,x_n) \ge 0. \end{cases} \qquad (2.67)$$

где функции $f(X)$ и $g_i(X)$, $i = 1, m$, в общем случае, нелинейные.

НП-задачи существенно отличаются от ЗЛП неформализованностью методов их решения. Нелинейность приводит к тому, что:

- область принятия решения может быть невыпуклая;
- область может иметь бесконечное число крайних точек.

Поэтому для решения НП-задач разработаны методы, которые ориентированы на классы задач в их конкретной постановке.

Общего подхода, являющегося универсальным во всех случаях, ***создать не удалось***.

2.5.1. Аналитические методы определения экстремумов

Указанные методы основываются на известных вам методах классического математического анализа, базируясь на ряд теорем [29].

Теорема 1 (о существовании экстремума). Если функция многих переменных $f(x_1, x_2, \dots, x_n)$ непрерывна и определена на замкнутом множестве $\mathfrak{R}$, то она достигает на этом множестве, ***по крайней мере, один раз*** своего минимального и максимального значений.

Теорема 2 (о местоположении экстремума). Если $f(x_1, x_2, \dots, x_n)$ является функцией нескольких переменных, определённой на допустимой области $\mathfrak{R}$, то экстремальное значение f (если оно существует) достигается в одной или нескольких точках, принадлежащих:

- множеству стационарных точек *S(X)*;
- множеству точек границы *G(X)*;

- множеству точек, в которых (где) функция $f(x_1, x_2, \dots, x_n)$ не дифференцируема.

Множество точек $S(X)$ функции $f(x_1, x_2, \dots, x_n)$ называется множеством стационарных точек, если его элементы удовлетворяю условию

$$\nabla f(X) = \left\{ \frac{\partial f(X)}{\partial x_j}, j = \overline{1,n} \right\} = 0. \qquad (2.68)$$

Вектор $\nabla f(X)$ – называют градиентом функции.

Находящий в стационарной точке ***минимум*** или ***максимум*** функции, может быть как ***абсолютным***, так и ***относительным***.

Относительный максимум функции $f(X)$ достигается в точке X^0 с координатами $(x_1^0, x_2^0, \dots, x_n^0)$, если для всех точек, лежащих в малой окрестности точки X^0, имеет место неравенство $f(X^0) \ge f(X^0 + H)$, где $H = \{h_1, h_2, \dots, h_n\}$.

Относительный максимум называется ещё ***локальным*** максимумом.

Абсолютный максимум функции $f(X)$ достигается в точке X^* с координатами $(x_1^*, x_2^*, \dots, x_n^*)$, если для всех точек, принадлежащих множеству ограничений $\mathfrak{R}$ справедливо неравенство $f(X^*) \ge f(X)$, где $X = \{x_1, x_2, \dots, x_n\} \in \mathfrak{R}$.

Абсолютный максимум называется ещё ***глобальным*** максимумом.

Аналогично, с точностью до знака неравенства, формулируются определения абсолютного и относительного минимумов.

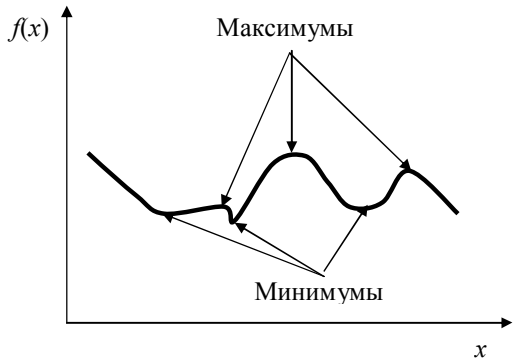

Рисунок 2.11 – Локальные и глобальные экстремумы

Характер экстремума, минимум или максимум, характеризуется выпуклостью или вогнутостью функции (рисунок 2.11).

Пусть $\mathfrak{R}$ – выпуклое множество точек n-мерного пространства.

Если для произвольного множителя $k \in [0, 1]$ и некоторого приращения ΔX выполняется неравенство

$$f(X + k\Delta X) \geq f(X) + k[f(X) - f(X + \Delta X)], \qquad (2.69)$$

то функция называется ***вогнутой*** (обращена выпуклостью вверх, в отличие от математического анализа!!!), а если

$$f(X + k\Delta X) \leq f(X) + k[f(X) - f(X + \Delta X)], \qquad (2.70)$$

то функция называется ***выпуклой***.

Если неравенства (2.69) и (2.70) строгие, говорят о **строгой вогнутости** и ***строгой выпуклости***. Для одномерного случая указанные неравенства интерпретируются графически, как это представлено на рисунке 2.12.

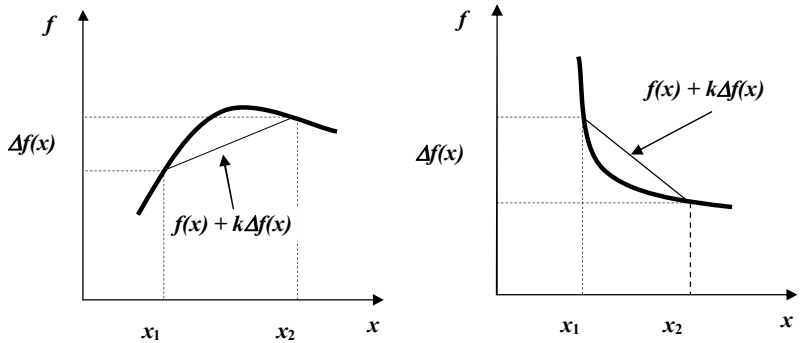

Рисунок 2.12 – Вогнутости и выпуклости функций

В многомерном случае непосредственное применение (2.69) или (2.70) является проблематичным. С этой целью применяется матрица Гёссе (или Гессе), элементы которой составляются из производных второго порядка и определяются так:

$$h_{i,j} = \left[\frac{\partial^2 f(X)}{\partial x_i \partial x_j} \right]_{x=x_0}, i = \overline{1,n}, j = \overline{1,m}. \qquad (2.71)$$

Матрица Гёссе (обозначается как $\nabla^2 f(X)$ и $H(X)$) называется ***положительно определённой***, если её главные угловые миноры

положительны, и ***отрицательно определённой***, если её главные угловые миноры имеют знак $(-1)^k$, k – номер углового минора.

В качестве напоминания [16, 18]: *минором элемента $a_{i,j}$ матрицы A называется определитель, построенный из элементов этой матрицы, оставшихся после вычёркивания i-ой строки и j-ого столбца.*

Главные угловые миноры матрицы соответствуют элементам, расположенным вдоль главной (с Северо-запада на Юго-восток) её диагонали.

$$\mu_1 = \begin{pmatrix} h_{1,1} & h_{1,2} & h_{1,3} \\ h_{2,1} & h_{2,2} & h_{2,3} \\ h_{3,1} & h_{3,2} & h_{3,3} \end{pmatrix} = \begin{vmatrix} h_{2,2} & h_{23} \\ h_{3,2} & h_{3,3} \end{vmatrix} = h_{2,2} \cdot h_{3,3} - h_{3,2} \cdot h_{2,3}.$$

Положительная определённость матрицы Гёссе соответствует ***выпуклости*** функции, отрицательная определённость – ***вогнутости***. Случай, когда знаки чередуются не по порядку, соответствует ***перегибу***.

Теорема 3. Для того, чтобы в точке X_0 достигался внутренний относительный максимум, достаточно ***равенства нулю всех первых*** производных и ***строгой вогнутости*** функции в окрестностях X_0.

Теорема 4. Для того, чтобы в точке X_0 достигался внутренний относительный минимум, достаточно ***равенства нулю всех первых*** производных и ***строгой выпуклости*** функции в окрестностях X_0.

Пример. Исследовать на экстремум функцию без ограничений

$$f(x_1, x_2, x_3) = x_1 + 2x_3 + x_2 \cdot x_3 - x_1^2 - x_2^2 - x_3^2.$$

Градиент этой функции есть

$$\nabla f(x_1, x_2, x_3) = \{1 - 2x_1; x_3 - 2x_2; 2 + x_2 - 2x_3\}.$$

Используя условие стационарной точки (2.68), получим её координаты

$$X_0 = \left\{ \frac{1}{2}; \frac{2}{3}; \frac{4}{3} \right\}.$$

Матрица Гёссе, построенная для рассматриваемого случая

$$\nabla^2 f(X) = \begin{pmatrix} -2 & 0 & 0 \\ 0 & -2 & 1 \\ 0 & 1 & -2 \end{pmatrix}.$$

Её угловые миноры суть

$$\mu_1 = \begin{vmatrix} -2 & 1 \\ 1 & -2 \end{vmatrix} = 3, \mu_2 = \begin{vmatrix} -2 & 0 \\ 0 & -2 \end{vmatrix} = 4, \mu_3 = \begin{vmatrix} -2 & 0 \\ 0 & -2 \end{vmatrix} = 4.$$

Они положительны, следовательно, матрица Гессе положительно определена, функция выпукла, и в точке X_0 с координатами $\left\{\frac{1}{2};\frac{2}{3};\frac{4}{3}\right\}$ достигается минимум.

2.5.2. Методы поиска экстремумов в задачах без ограничений или в случае ограничений с разделяющимися переменными

Ограничение $g_i(x_1, x_2, \ldots, x_n) = 0$ является функцией с ***разделяемыми переменными (сепарабельной)***, если его можно представить в виде

$$x_i = \varphi_i(\{x_j\}), \qquad i \neq j, \quad j = \overline{1,n}.$$

Общий алгоритм решения

- Отыскивается множество всех стационарных точек S функции $f(X)$ внутри допустимого множества $\mathfrak{R}$ и выбираются координаты точки, в наибольшей степени отвечающие направлению оптимизации задачи.
- Рассматривается множество точек границы G. Для этого выполняют разделение переменных по каждому из ограничений, с последующей подстановкой выражений переменных в функцию цели $f(X)$. Полученные функции исследуются на экстремумы. Выбираются координаты интересующего нас оптимума.
- Определяется и подвергается исследованию множество точек, принадлежащих $\mathfrak{R}$, где функция не дифференцируема.
- Из результатов предыдущих шагов выбирается наилучшее решение.

Замечания.

1. Метод требует значительных вычислительных затрат и аналитических преобразований.

2. Не отвечает должной формализации для использования вычислительной техники.

3. Применение ограничивается задачами, область поиска решений которых описывается функциями с сепарабельными переменными.

Тем не менее, были разработаны многочисленные методы, направлены на решение задачи и ориентированные на применение ЭВМ. “Движение” текущей точки к оптимуму может происходить различными способами. В зависимости от этого, имеет место следующая классификация, методы подразделяются на:

- прямые методы или методы, использующие лишь значения функции;
- методы первого порядка, использующие, наряду со значениями функции значения первых частных производных;
- методы второго порядка, использующие дополнительно к значениям функции и первых частных производных и прочие частные вторые производные и выше.

Причём производные могут вычисляться как аналитически, так и численно.

2.5.2.1. Прямые методы поиска

Одномерный поиск

При одномерном поиске функция *f(x)*, экстремум которой ищется, зависит от одной переменной. В дальнейшем будем предполагать, что ***решается задача минимизации***. Совершенно аналогичный алгоритм может быть использован и для случая поиска максимума. Отличия будут заключаться в знаках неравенств в проверках, производимых в процессе функционирования алгоритма.

Одномерный поиск является составной частью многих алгоритмов многомерного поиска в качестве вспомогательного. Алгоритмов одномерного поиска разработано изрядно, мы рассмотрим только самые интересные из них как с точки зрения скорости вычислений, так и их структуры.

Дихотомический поиск

Так же называется методом половинного деления. Это очень древний алгоритм, известный со времён раннего средневековья Ближнего Востока под названием “Поимка льва в пустыне”.

Есть пустыня, в ней лев. Делим пустыню пополам, лев окажется в одной из половинок. Часть пустыни, где находится лев, снова подвергаем делению… В конце концов, размеры пустыни окажутся чуть больше величины льва, и царя зверей остаётся только заключить в клетку.

Входные данные. Интервал поиска (сиречь, пустыня) [*a, b*]; допустимая конечная длина интервала, определяющая точность расчётов $l > 0$; константа различимости ε.

Условие окончания: $b\text{-}a \le l$.

Дадим пояснение алгоритма с использованием рисунков 2.11 и 2.12.

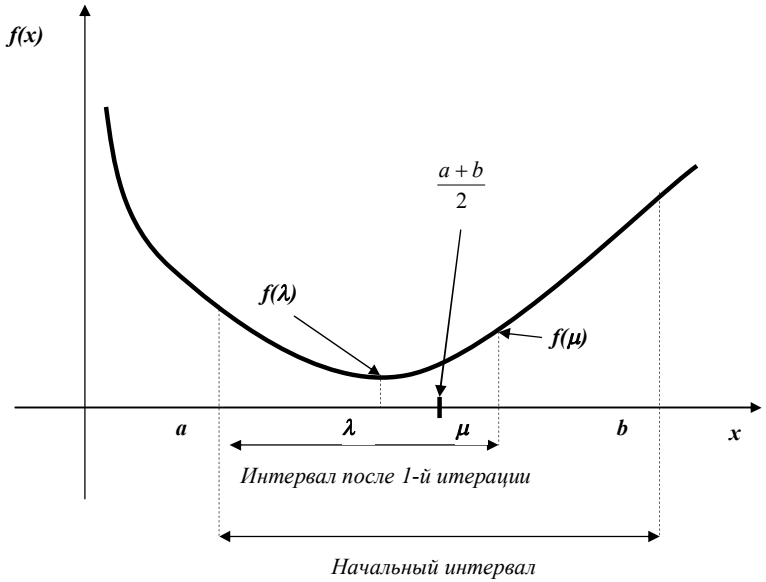

Рисунок 2.13 – Сужение интервала по мере дихотомии

Первоначально вычисляется пара величин: $\lambda = \frac{a+b}{2} - \varepsilon$ и $\mu = \frac{a+b}{2} + \varepsilon$, а тако же соответствующие им значения функций $f(\lambda)$ и $f(\mu)$. Полученные значения функций сравниваются между собой.

Пусть, например, $f(\lambda) \le f(\mu)$ (условие № 1 на схеме алгоритма, рисунок 2.14), выход «да». В этом случае сдвигается граница b, как это показано на рисунке 2.13. В противном случае, выход «нет», граница a сдвигается по направлению к середине интервала $\frac{a+b}{2}$.

Длительность вычислительного процесса контролируется условием № 2: вычисления заканчиваются при возникновении ситуации $b - a \le l$, которая означает, что границы интервала поиска сузились до заданного точностного размера l.

Имеется выражение, которое позволяет определить число итераций, необходимое для получения решения с заданной точностью:

$$b_k - a_k = \frac{(b-k)}{2^k} + 2 \cdot \varepsilon \left(1 - \frac{1}{2^k}\right),$$

где k – число итераций.

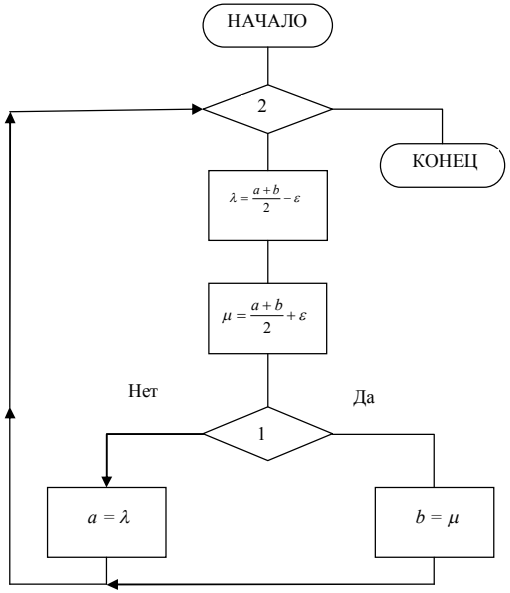

Рисунок 2.14 – Схема алгоритма дихотомического метода

Алгоритм, описанный выше, обладает самой меньшей скоростью сходимости, но, тем не менее, пользуется популярностью у программистов за простоту его реализации.

Метод золотого сечения

При функционировании метода выполняются условия:

- интервал поиска сужается равномерно;
- параметры точек сравнения и концов интервала соотносятся следующим образом

$$b - \lambda = \mu - a. \qquad (2.72)$$

Выполнение (2.72) достижимо, если значения λ и μ рассчитывать по формулам

$$\lambda = a + (1 - \alpha)\times(b - a) \text{ и} \quad (2.73)$$
$$\mu = a + \alpha\times(b - a), \quad (2.74)$$

где $|\alpha| < 1$.

Действительно, из (2.74) получается

$$\mu - a = \alpha\times(b - a). \quad (2.75)$$

Если λ, определяемое (2.73), подставить в левую часть (2.72), то

$$\boldsymbol{b - a} - (1 - \alpha)\times(\boldsymbol{b - a}) = \alpha\times(b - a). \quad (2.76)$$

Сопоставляя (2.75) и (2.76) видим, что условие (2.72) будет неизменно выполняться при определении λ и μ по формулам (2.73) и (2.74).

В ходе вычислений новые границы переставляются таким образом, чтобы либо $\lambda` = \mu$, либо $\mu` = \lambda$. Таким образом, пересчёт координат и функции в одной из этих точек не производится.

Коэффициент золотого сечения $\alpha = 0{,}618\ldots$ есть предел отношения соседних членов ряда Фибоначчи при их числе, стремящемся бесконечности.

Интервал поиска сужается равномерно, пропорционально α по закону

$$b_k - a_k = \alpha^k (b - a),$$

что позволяет оценить значение k – число итераций, необходимых до нахождения оптимума с заданной точностью.

Чертёж, поясняющий ход расчетов, и схема алгоритма представлены на рисунках 2.15 и 2.16.

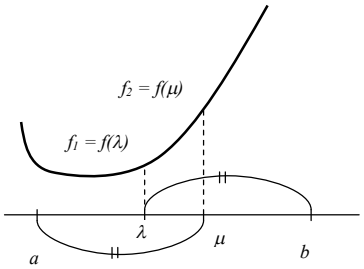

Рисунок 2.15 – Интервалы в методе золотого сечения

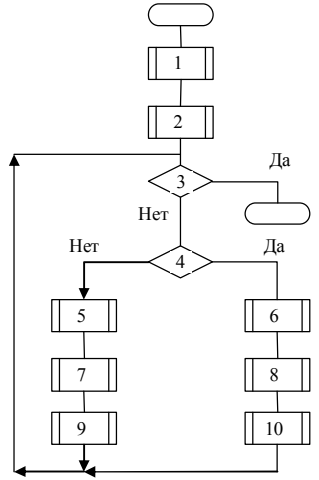

Рисунок 2.16 – Алгоритм метода золотого сечения

Входные данные. Интервал поиска $[a, b]$; точность расчётов $l > 0$.

Блоки № 1 рассчитывает параметры λ и μ по формулам (2.73) и (2.74), а блок № 2 определяет значения функции $f_1 = f(\lambda)$ и $f_2 = f(\mu)$, им соответствующие. Условие 3 $b - a \le l$ определяет остановку вычислительного процесса при достижении заданной погрешности. Условие 4 $f_1 \le f_2$ ответственно за сдвиги границ интервала поиска, аналогично дихотомическому методу. Блоки №№ 5, 7, 9 выполняют операции, связанные со сдвигом границы a.

- 5: $a = \lambda$; $\lambda = \mu$.
- 7: Расчёт μ по формуле (2.74)
- 9: Перестановка пересчёт значений функций $f_1 = f_2$; $f_2 = f(\mu)$

Блоки №№ 6, 8, 10 выполняют аналогичные операции, связанные со сдвигом границы b.

- 6: $b = \mu$; $\mu = \lambda$.
- 8: Расчёт λ по формуле (2.73)
- 10: Перестановка пересчёт значений функций $f_2 = f_1$; $f_1 = f(\lambda)$.

Метод Фибоначчи

В отличие от метода золотого сечения, указанный метод

- на каждой итерации изменяет коэффициент сжатия интервала и
- выполняется заранее известного числа шагов.

Основывается на использовании ряда Фибоначчи, элементы которого определяются рекуррентным соотношением

$$F_{n+1} = F_n + F_{n-1},\ F_0 = F_1 = 1, \qquad (2.77)$$

определяющим, что каждый последующий член ряда образован суммой двух предыдущих:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 ...

Входные данные. Интервал поиска [*a, b*], параметр, задающий точность расчётов $l > 0$, константа различимости $\varepsilon << l$.

Условие окончания: поиск заканчивается через определённое число шагов (итераций), определяемых по неравенству

$$\frac{b-a}{l} < F_n \Rightarrow n. \qquad (2.78)$$

Координаты точек λ_k и μ_k, где k – номер итерации, рассчитываются по формулам

$$\lambda_k = a_k + \frac{F_{n-k-1}}{F_{n-k+1}}(b_k - a_k), \qquad (2.79)$$

$$\mu_k = a_k + \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k). \qquad (2.80)$$

Согласно формулам (2.79) и (2.80), от итерации к итерации, интервал поиска меняется по закону

$$b_{k+1} - a_{k+1} = \frac{F_{n-k}}{F_{n-k+1}}(b_k - a_k) = b_k - \lambda_k = \mu_k - a_k, \qquad (2.61)$$

причём, перемещение границ интервала поиска и пересчёт $f(\lambda_k)$ и $f(\mu_k)$ происходит с соблюдением тех же принципов, что и в методе золотого сечения.

Константа различимости ε используется после окончания циклической части алгоритма для уточнения полученного результата. Схема алгоритма представлена рисунком 2.17.

Блок № 1 выполняет подготовительные операции: осуществляется расчёт ряда Фибоначчи (2.77), попутно определяется (2.78) – число членов ряда n, необходимых для расчёта; начальные значения

$$\lambda_0 = a + \frac{F_{n-2}}{F_n}(b-a),\ \ \mu_0 = a + \frac{F_{n-1}}{F_n}(b-a),\ \ f_1 = f(\lambda_0) \text{ и } f_2 = f(\mu_0).$$

Блок № 2 есть заголовок цикла изменения счётчика k от 1 до $n - 2$, тело цикла представляется блоками №№ 3 – 5.

Условие, управляющее сдвигом границ, $f_1 \le f_2$, представлено блоком № 3.

Сдвиг левой границы, выход «нет» блока № 3, выполняется в блоке № 4:

- $a = \lambda_k;\ \lambda_k = \mu_k$.
- Расчёт μ_k по формуле (2.80)
- Перестановка пересчёт значений функций $f_1 = f_2;\ f_2 = f(\mu_k)$.

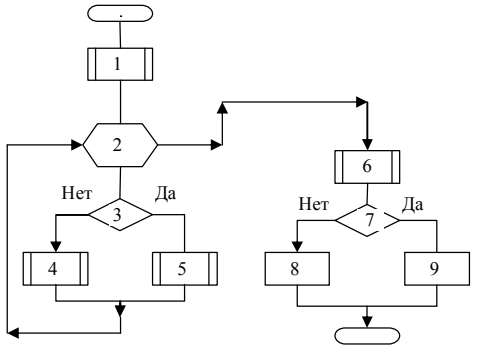

Рисунок 2.17 – Алгоритм метода Фибоначчи

Сдвиг правой границы, выход «да» блока № 3, выполняется в блоке № 5:

- $b = \mu_k;\ \mu_k = \lambda_k$.
- Расчёт λ_k по формуле (2.79)
- Перестановка пересчёт значений функций $f_2 = f_1;\ f_1 = f(\lambda_k)$.

Блоки №№ 6, 7, 8 и 9 осуществляют уточнение результата методом половинного деления. Блок № 6 выполняет операции $\mu = \lambda + \varepsilon$ и $f_2 = f(\mu)$. Блок № 7 проверяет условие $f_1 \le f_2$ и, в зависимости от его выполнения, сдвигает левую границу, выход «нет» блок № 8, $a = \lambda$, либо правую, выход «да», блок № 9, $b = \mu$.

Многомерный поиск

При многомерном поиске функция $f(X)$, экстремум которой ищется, зависит от нескольких переменных. Алгоритмы многомерного поиска эвристические. Хотя некоторые из них носят имена своих первооткрывателей, не факт, что они не переизобретаются заново многочисленными последователями, хотя и более невежественными в плане оптимизации, но, тем не менее, более подкованными в плане алгоритмизации.

Метод конфигураций

Метод конфигураций называется тако же методом траекторий или, по имени авторов, методом Хука и Дживса.

Идея метода состоит в следующем.

- Направления поиска ориентированы, так сказать, по сторонам света или вдоль осей координат в обе стороны.
- Вокруг текущей точки производится поиск направления, в котором функция убывает.
- Если такое направление найдено, то шаг поиска увеличивается, а поиск продолжается в выбранном направлении – устанавливается так называемый тренд поиска, и осуществляется до тех пор, пока функция в этом направлении убывает.
- Если факта убывания функции не обнаружено, то шаг поиска уменьшается.

Таким образом, делается попытка найти «овраг» (при минимизации) функции и двигаться вдоль него к точке минимума.

В задачи максимизации ищется направление возрастания функции и «хребет».

Алгоритм метода показан на рисунке 2.18.

Входные данные. Начальное значение шага λ, координаты начальной точки X_0 ускоряющий множитель α, точность нахождения решения ε, список возможных направлений поиска $S = \{s_1, s_2, \dots, s_n\}$. Для числа переменных равного двум, это орты: $s_1 = (0, 1)$ и $s_2 = (1,0)$.

Условие окончания: значение текущего шага укладывается в точность, $\varepsilon \geq \lambda$.

Блок № 1. Координаты начальной точки переприсваивается текущей переменной: $Y = X_0$.

Блок № 2. Организуется цикл перебора направлений поиска $j = 1, n$, тело цикла составляют блоки №№ 3 – 7.

Сравнение в блоке № 3 проверяется факт убывания функции в выбранном направлении: $f(Y + \lambda \times s_j) < f(Y)$. Если функция убывает, выход «да» блока № 3, то текущая точка перемещается в выбранном направлении $Y = Y + \lambda \times s_j$ (блок № 4).

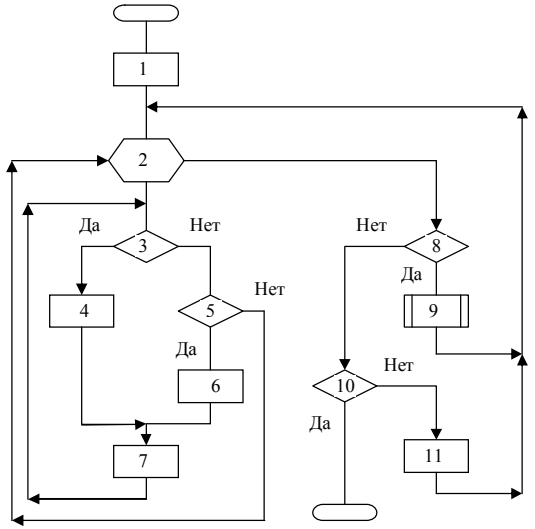

Рисунок 2.18 – Алгоритм метода Хука-Дживса

В противном случае, выход «нет» блока № 3, в блоке № 5 проверяется факт убывания функции в направлении, противоположном выбранному: $f(Y - \lambda \times s_j) < f(Y)$.

Если функция не убывает ни в выбранном, ни в противоположном направлениях (выход «нет», блок № 5), то выбирается очередное направление.

Если сравнение в блоке № 5 произошло удачно (выход «да»), то текущая точка решения перемещается в блоке № 6 по закону $Y = Y - \lambda \times s_j$.

Блок № 7 выполняет факультативное действие $\lambda = 2 \times \lambda$, которое увеличивает шаг, ускоряя тем самым поиск. В ряде реализаций алгоритма данное действие отсутствует.

Блок № 8 достигается в ходе работы алгоритма, после перебора всех направлений поиска и попадания в ситуацию, когда при заданном значении λ в окрестностях точки X_0 убывания функции не обнаружено.

Блок № 9 выполняет пересчёт координат текущей точки по формуле

$$Y = X_0 + \alpha \times (Y - X_0); \; X_0 = Y.$$

Блок № 10 осуществляет проверку окончания: $\varepsilon \geq \lambda$? Если «да» то алгоритм заканчивает работу, если «нет», то шаг поиска уменьшается

$$\lambda = \frac{\lambda}{2},$$

после чего начинается поиск вокруг текущей точки X_0 с уменьшенным шагом.

Пример поиска представлен ниже, на рисунке 2. 19.

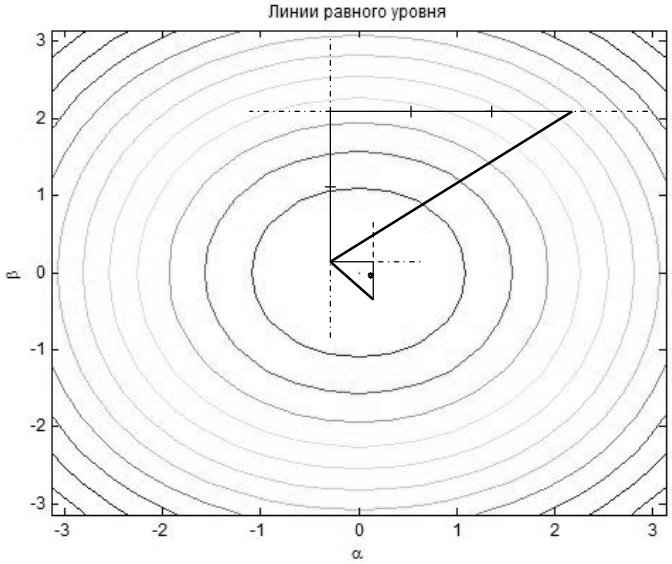

Рисунок 2.19 – Поиск решения методом траекторий

Для демонстрации использована функция вида $f(\alpha,\beta) = -\frac{\sin(\alpha+\beta)}{\alpha+\beta}$, которая достигает своего «дна» в точке с координатами (0, 0), это положение отмечено точечкой в середине рисунка, пунктиром показаны неудачные пробы, жирным – траектория спуска.

Замечания.

К безусловным ***достоинствам*** метода следует отнести следующие:

- способность восстанавливать направление движения, когда при искривлении «оврага» тренд поиска теряется;
- явного задания функции не требуется;
- легко учитываются ограничения на переменные и область поиска.

В качестве недостатка отмечается, что при попадании в область размытого локального минимума есть риск остановиться, не достигнув глобального минимума.

Метод Розенброка

Идея метода состоит в поиске по взаимно-ортогональным направлениям на каждом шаге алгоритма, в отличие от алгоритма Хука-Дживса, где поиск осуществляется вдоль координатных осей.

Если поиск, в каком либо из направлений, оказывается удачным, то шаг поиска по этому направлению увеличивается, в противном случае шаг уменьшается.

Поисковая процедура заканчивается, когда на каждом из направлений поиска одна проба окажется успешной, а одна – неудачной.

Данный метод ***особенно хорош*** при работе с функциями, обладающими длинными, узкими и искривлёнными гребнями и оврагами.

Топология алгоритма представлена на рисунке 2.20.

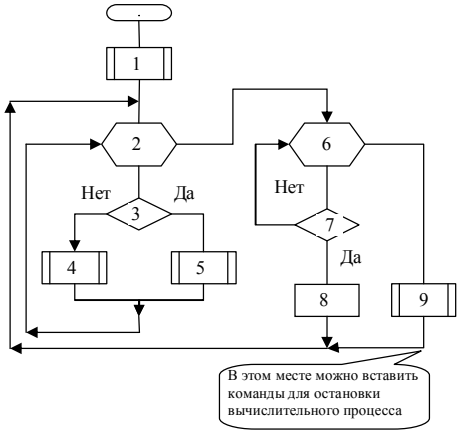

Рисунок 2.20 – Алгоритм метода Розенброка

Входные данные. Координаты начальной точки X_0; список возможных направлений поиска, первоначально набор ортогональных векторов $S = \{s_1, s_2, \ldots, s_n\}$; набор значений шагов поиска $\Lambda^T = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$; ускоряющий множитель α; замедляющий множитель β.

Условие окончания: автором не предусмотрено, ибо, скорее всего, считал вручную и вычислительный процесс контролировал визуально. При компьютерной реализации используют один из возможных подходов:

- выполняют несколько фиксированных шагов, после чего алгоритм прекращает свою работу;
- модуль вектора приращений (расстояние между смежными текущими точками на итерации) $|\Delta| \le \varepsilon$ несколько раз подряд.

Розенброком были предложены следующие значения констант: $\alpha = 3$, $\beta = -0{,}5$.

Блок № 1 осуществляет подготовительные операции, $Y = X_0$.

Блоки №№ 2 – 5 выполняют сканирование направлений: заголовок цикла перебора (блок № 2); вычисление точки в окрестностях текущей точки в выбранном направлении, значения функции и сравнение значений : $f(Y + \lambda_i s_i) < f(Y)$(блок № 3).

Если значение функции в выбранном направлении увеличивается или неизменно, выход «нет», то текущее направление поиска штрафуется, путём умножения $\lambda_i = \beta \times \lambda_i$ и запоминается соответствующие координаты в окрестностях текущей точки $P_i = Y + \lambda_i s_i$ – блок № 4.

Если функции в выбранном направлении убывает, выход «да», то направление поиска поощряется $\lambda_i = \alpha \times \lambda_i$, и также запоминаются координаты $P_i = Y + \lambda_i s_i$ – блок № 5.

Блоки №№ 6 и 7 осуществляют поиск удачного направления среди просканированных направлений. Блок № 6 – организует циклический перебор, блок № 7 проверяет условие $f(P_i) < f(Y)$.

Если первое же попавшееся направление удачное, текущая точка перемещается блоком № 8, $X_0 = P_i$, после чего вычислительный процесс возобновляется со сканирования направлений.

Если удачное направление поиска не найдено, то необходимо модифицировать систему направлений, что осуществляется блоком № 9:

- вычисляется приращение $\Delta = X_0 - Y$;
- переопределяется новая точка начала итерации $Y = X_0$;
- новое направление s_1 берется совпадающим с направлением вектора Δ (единичные значения по ненулевым координатам), а остальные направления пересчитываются с учётом условия ортогональности.

2.5.2.2. Градиентные методы поиска

Эти методы основываются на использовании значений градиентов целевой функции, как следует из их названия.

В зависимости от порядка используемых градиентов бывают первого и второго порядков. Далее мы рассмотрим

- метод наискорейшего спуска (подъёма) – 1-го порядка;
- метод Ньютона (вторых производных) – 2-го порядка;
- метод сопряжённых направлений (сопряжённого градиента), предложенный Флетчером и Ривсом, 1-го порядка;
- методы переменной метрики, квазиньютоновские, 1-го порядка.

Общая идея, положенная в основу этих методов, для случая функции одной переменной, пояснена с помощью рисунка. 2.21.

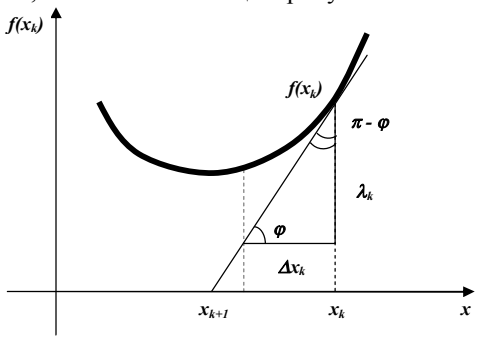

Рисунок 2.21 – Использование градиента

По рисунку видно, что $x_{k+1} = x_k - \Delta x_k$ и $\tan\varphi = f'(x_k)$ – геометрический смысл первой производной функции одной переменной.

Рассмотрение прямоугольного треугольника позволяет найти $\Delta x_k = \lambda_k \cdot f'(x_k)$ при заданном значении λ_k. В этом случае нами использованы формулы связи значений тригонометрических функций при изменении аргумента на величину $\pm\pi$.

Для многомерного случая справедливо аналогичное выражение

$$x_{k+1} = x_k - \lambda_k \times \nabla f(x_k), \qquad (2.81)$$

где $\lambda_k > 0$ – величина шага на k-ой итерации. Все особенности градиентных методов заключаются в приёмах определения λ_k на каждой итерации.