Курс «Вычислительные алгоритмы теории автоматического управления».

Лекция 6. Безусловная оптимизации систем. Методы нулевого, первого и второго порядков.

Методы моделирования случайных функций. Общая постановка задач оптимизации. Методы решения задач безусловной оптимизации. Сортировка одномерных массивов. Оптимизация одномерных унимодальных функций. Элементы выпуклого анализа. Методы локальной безусловной многомерной оптимизации. Методы прямого поиска (нулевого порядка). Методы оптимизации первого порядка. Методы безусловной оптимизации второго порядка.

Общая постановка задач оптимизации.

Пусть заданы множество $X \subseteq \Re^n$ и функция $f(x)$, определенная на множестве X. Требуется найти точки минимума или максимума $f(x)$ на множестве X. Условимся, в дальнейшем, рассматривать задачи на минимум, то есть задачи в следующей записи.

$$f(x) \underset{x \in X}{\rightarrow} \min .$$

При этом $f(x)$ будем называть целевой функцией, а X - допустимым множеством.

Следует подчеркнуть, что само понятие точки экстремума (минимума) неоднозначно и требует уточнения. Точка $x^* \in X$ называется:

- точкой глобального минимума функции $f(x)$ на множестве X или глобальным решением, если

$$f(x^*) \le f(x)$$

при всех $x \in X$;

- точкой локального минимума $f(x)$ на множестве X или локальным решением задачи $f(x) \underset{x \in X}{\rightarrow} \min$, если существует число $\varepsilon > 0$ такое, что $f(x^*) \le f(x)$ при $\forall x \in X \cap U_\varepsilon(x^*)$, где $U_\varepsilon(x^*) = \{x \in \Re^n : \| x - x^* \| \le \varepsilon\}$ - шар радиусом $\varepsilon > 0$ с центром в точке x^*. Если вышеуказанные неравенства выполняются как строгие, при любой точке $x \neq x^*$, то говорят, что точка x^* есть точка строгого минимума в глобальном или локальном смысле. Очевидно, что глобальное решение является и локальным; обратное неверно.

Множество всех точек глобального минимума обозначим с помощью следующего соотношения. $\arg \min_{x \in X}(f(x)) = \{x^* \in X : f(x^*) = f^*\}$, где $f^* = f(x^*) = \min_{x \in X}(f(x))$.

Из курса математического анализа известно следующее условие о существовании решения задачи оптимизации /Сухарев с9/.

***Теорема Вейерштрасса**. Пусть X – замкнутое ограниченное множество (компакт), $X \subset \Re^n$, а $f(x)$ непрерывная функция на множестве X. Тогда точка глобального минимума функции $f(x)$ на множестве X существует.*

Классификация основных задач оптимизации.

Прежде всего, все задачи оптимизации систем управления можно разделить на следующие три основных класса: задачи условной и безусловной оптимизации, а также на динамические задачи дискретной оптимизации (оптимального управления).

Рассмотрим теперь формальные постановки указанных выше задач оптимизации.

Задача безусловной оптимизации.

Задача $f(x) \underset{x \in X}{\rightarrow} \min$ называется задачей безусловной оптимизации, если $X = \Re^n$. То есть задачу можно записать в следующем виде $f(x) \rightarrow \min, \; x \in \Re^n$.

Задача условной оптимизации.

Задача $f(x) \underset{x \in X}{\to} \min$ называется задачей условной оптимизации, если $X \subset \Re^n$. То есть локальное решение x^* является внутренней точкой допустимого множества X, $x^* \in \text{int}(X)$. Во многих условных задачах минимум достигается именно на границе множества X, в силу чего для них классические результаты анализа являются неприменимыми. Обычно допустимое множество X задается с помощью конечного числа уравнений и неравенств. Так как любое равенство $g(x) = 0$ можно записать в виде двух неравенств вида $g(x) > 0$ и $g(x) \le 0$, то множество X часто записывают в следующем виде: $X = \{x \in \Re^n : g_i(x) \le 0, i = 1:m\}$.

В зависимости от вида функций $f(x)$ и $g_i(x)$, а также некоторых дополнительных ограничений на множество X внутри класса задач условной оптимизации различают следующие подклассы, для которых существуют свои, более эффективные, методы решения:

1. Задачи линейного программирования следующего вида:

$$C \cdot x \to \min,\ x \in \Re^n,\ A \cdot x = b,\ x \ge 0,$$

где $C^T \in \Re^n,\ b \in \Re^m,\ A \in \Re^{m \times n}$.

2. Задачи квадратичного программирования:

$$q \cdot x + x^T \cdot Q \cdot x \to \min,\ x \in \Re^n,\ A \cdot x = b,\ x \ge 0,$$

где $q^T \in \Re^n,\ b \in \Re^m,\ A \in \Re^{m \times n},\ Q \in \Re^{n \times n}$.

3. Задачи нелинейного программирования, когда функции $f(x)$ и $g_i(x)$ являются функциями общего вида.

4. Задачи стохастического программирования, когда множество X задается с помощью вероятностных ограничений вида $P\{x \in X(\omega)\} \ge \alpha$, $P(.)$ - вероятность нахождения точки x внутри ограничивающего множества X.

5. Задачи целочисленного программирования, когда $X \subset \mathbb{Z}^n$, где $\mathbb{Z} = \{..., -1, 0, 1, ...\}$.

Динамические задачи дискретной оптимизации.

Эти задачи можно формально записать в следующем виде:

$$\sum_{t=1}^{T} f_t(x_{t-1}, u_t) \to \min,\ x_{t+1} = G(x_t, u_t),\ x_t \in X_T \subset \Re^n,\ u_t \in U_T \subset \Re^m.$$

Следует также отметить, что все вышеперечисленные задачи могут рассматриваться, как в случае, когда целевая функция $f(x)$ имеет скалярный характер, так и в случае векторной целевой функции. То есть когда целевая функция имеет вид:

$$f(x) = \{f_1(x), f_2(x), ..., f_L(x)\}.$$

В случае векторной целевой функции задачи оптимизации называются многокритериальными.

<u>Методы решения задач безусловной оптимизации.</u>

Введем следующие обозначения: $\nabla f(x^*) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, ..., \frac{\partial f}{\partial x_n}\right)\Big|_{x=x^*}$ - вектор-строка частных производных функции $f(x)$ в точке x^*; $\nabla^2 f(x^*) = \left(\frac{\partial^2 f}{\partial x_i \cdot \partial x_j}\right)\Big|_{x=x^*}; i, j = 1:n$ - матрица вторых частных производных (гессиан) функции $f(x)$ в точке x^*.

Теорема */Зангвилл с30/* *Пусть функция* $f(x)$ *дифференцируема в точке* x^*. *Предположим, что имеется направление (вектор)* h, *для которого* $\nabla f(x^*)\cdot h<0$. *Тогда существует* $\alpha>0$ *такое, что для всех* $\beta;\ \alpha\geq\beta>0$, *что* $f(x^*+\beta\cdot h)<f(x^*)$.

Доказательство. Действительно, из предположения теоремы имеем

$\lim\limits_{\alpha\to 0}\dfrac{f(x^*+\alpha\cdot h)-f(x^*)}{\alpha}=\nabla f(x^*)\cdot h<0$. То есть, существует такое $\beta>0$, что

$\lim\limits_{\alpha\to 0}\dfrac{f(x^*+\beta\cdot h)-f(x^*)}{\beta}=\nabla f(x^*)\cdot h<0$. Отсюда получаем $f(x^*+\beta\cdot h)<f(x^*)$.

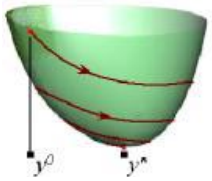
Теорема */Сухарев10/* *Пусть функция* $f(x)$ *дифференцируема в точке* x^*. *Если точка* x^* - *локальное решение задачи* $f(x)\to\min,\ x\in\Re^n$, *то* $\nabla f(x^*)=0$.

Доказательство. Пусть $\nabla f(x^*)\neq 0$ Выберем $h=-(\nabla f(x^*))^T$. Тогда $\nabla f(x^*)\cdot h=-\nabla f(x^*)\cdot(\nabla f(x^*))^T<0$. Отсюда, по теореме 1, существует точка $x=x^*+\beta\cdot h$ с меньшим значением функции $f(x)$, что противоречит исходному предположению теоремы 2.

Теорема 3. *Пусть функция* $f(x)$ *дважды дифференцируема в точке* x^*. *Если точка* x^* - *локальное решение задачи* $f(x)\to\min,\ x\in\Re^n$, *то матрица* $\nabla^2 f(x^*)=(\dfrac{\partial^2 f}{\partial x_i\cdot\partial x_j})|_{x=x^*};i,j=1:n$ *неотрицательно определена, то есть* $(\nabla^2 f(x^*)\cdot h,h)\geq 0$ *для всех* $h\in\Re^n$.

Доказательство. Запишем следующее равенство:

$f(x^*+\alpha\cdot h)-f(x^*)=\dfrac{1}{2}\cdot(\nabla^2 f(x^*)\cdot\alpha\cdot h,\alpha\cdot h)+o(\alpha^2)$. Тогда можно найти такое $\beta;\ \alpha\geq\beta>0$, что $(\nabla^2 f(x^*)\cdot\beta\cdot h,\beta\cdot h)\geq 0$ или $(\nabla^2 f(x^*)\cdot h,h)\geq 0$.



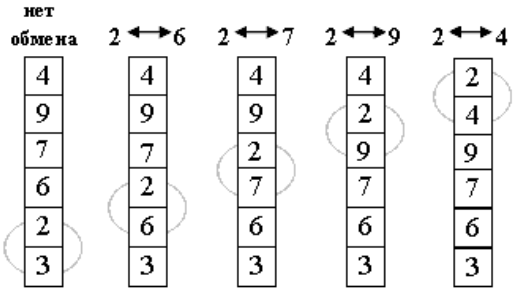
Теорема. *Пусть функция* $f(x)$ *дважды дифференцируема в точке* x^*. *Предположим, что* $\nabla f(x^*)=0$, *а матрица* $\nabla^2 f(x^*)=(\dfrac{\partial^2 f}{\partial x_i\cdot\partial x_j})|_{x=x^*};i,j=1:n$ *положительно определена, то есть* $(\nabla^2 f(x^*)\cdot h,h)\geq 0$ *для всех* $h\in\Re^n$. *Тогда точка* x^* *строго локальное решение задачи* $f(x)\to\min,\ x\in\Re^n$.

Сортировка одномерных массивов.

Одной из важнейших задач безусловной оптимизации является задача сортировки конечного количества элементов по их значению. Сортировка – это процесс упорядочивания конечного набора элементов в возрастающем или убывающем порядке. Несмотря на внешнюю простату такой задачи, эффективность многих численных алгоритмов оптимизации напрямую зависит от сложности алгоритмов решения такой задачи. Поэтому многие исследователи считают сортировку наиболее фундаментальной задачей при изучении алгоритмов. Существует несколько причин, обуславливающих такой вывод:
- практически нет ни одного реального приложения, где бы алгоритмы сортировки не применялись;
- во многих алгоритмах сортировка используется в качестве вспомогательной программы и, неэффективное ее решение, резко снижает эффективность основного алгоритма;
- сортировка: это задача, для которой можно доказать наличие нетривиальной нижней границы;
- в процессе реализации алгоритмов сортировки выявляются многие дополнительные задачи об организации взаимодействия аппаратных и программных средств, обеспечивающей существенное повышение быстродействия технического решения проектируемой системы /Кормен с175/.

Рассмотрим следующие алгоритмы сортировки:
- сортировка обменом (метод пузырька);
- сортировка вставками;
- сортировка делением (быстрая сортировка по Хоару).

Сортировка обменом основана на последовательном сравнении пары соседних элементов a_i и a_{i+1}. Если пара расположена не в требуемом порядке, то элементы переставляются. При сортировке по возрастанию после первого «прохода» массива от первого до последнего элемента на последнем месте окажется наибольший элемент массива. Далее сортируется оставшаяся часть массива. Наибольшее число проходов равно «n-1», причем число проверок при очередном проходе уменьшается на единицу.



Нулевой проход, сравниваемые пары выделены

После нулевого прохода по массиву "вверху" оказывается самый "легкий" элемент - отсюда аналогия с пузырьком. Следующий проход делается до второго сверху элемента, таким образом, второй по величине элемент поднимается на правильную позицию.

```
for i=0; i < size;  i - номер прохода
  for  j = size-1; j > i; j-- )  -внутренний цикл прохода
    if ( a[j-1] > a[j] )
    x=a[j-1]; a[j-1]=a[j]; a[j]=x;
    end for j
  end for i
```

Среднее число сравнений и обменов имеют квадратичный порядок роста: $\Theta(n^2)$. Поэтому считается, что метод пузырька крайне медленный и для сортировки больших массивов обычно не используется.

Сортировка вставками также в основном используется для небольших массивов. Метод заключается в случайном выборе первого элемента и помещении его в буфер; далее из массива

берется по одному элементу, каждый которых помещается в нужное место среди других элементов буфера. Чтобы определить, куда нужно поместить очередной элемент, он сравнивается со всеми элементами буфера.

```
for j =2 to length [A]
        do key = A(j);
        // вставка элемента A(j) в отсортированную последовательность A[1..j-1]
  j=j-1;
  while i>0 &&  A(i)>key
        do A(i+1)=A(i);
  i=i-1;
   A(i+1)=key;
end for j
```

Среднее число сравнений и обменов для этого алгоритма также имеют квадратичный порядок роста: $\Theta(n^2)$.

Сортировка делением (быстрый алгоритм Хоара).

Быстрая сортировка построена на идее деления. Общая процедура заключается в том, чтобы выбрать некоторое значение, называемое *компарандом* (comparand), а затем разбить массив на две части. Все элементы, большие или равные компаранду, перемещаются на одну сторону, а меньшие — на другую. Потом этот процесс повторяется для каждой части до тех пор, пока массив не будет отсортирован. Например, если исходный массив состоит из символов **fedacb**, а в качестве компаранда используется символ **d**, первый проход быстрой сортировки переупорядочит массив следующим образом:

Начало **f e d a c b**
Проход 1 **b c a d e f**

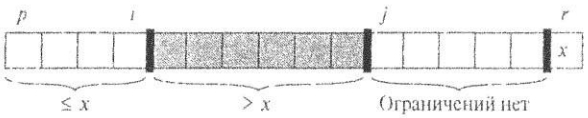
Затем сортировка повторяется для обеих половин массива, то есть **bca** и **def**. Этот процесс по своей сути рекурсивный, и, действительно, в чистом виде быстрая сортировка реализуется как рекурсивная функция, то есть на основе так называемой парадигме «разделяй и властвуй». Значение компаранда можно выбирать двумя способами — случайным образом либо усреднив небольшое количество значений из массива. Для оптимальной сортировки необходимо выбирать значение, которое расположено точно в середине диапазона всех значений. Однако для большинства наборов данных это сделать непросто. В худшем случае выбранное значение оказывается одним из крайних. Тем не менее, даже в этом случае быстрая сортировка работает правильно. Алгоритм быстрой сортировки условно реализуется следующей процедурой /Кормен с199/.

```
QuickSort(A,p,r)
        If p<r
          then q=Partition(A,p,r)
                QuickSort(A,p,q-1)
                QuickSort(A,q+1,r)
```

Ключевой частью рассматриваемого алгоритма сортировки является процедура Partition, изменяющая порядок элементов массива A[p,r] без привлечения дополнительной памяти.

```
Partition(A,p,r)
x=A(r)
i=p-1
for j=p to r-1
        do if A(j) ≤ x
                then i=i+1
```

// обменять A(i) с A(j)
// обменять A(i+1) с A(r)
return i+1



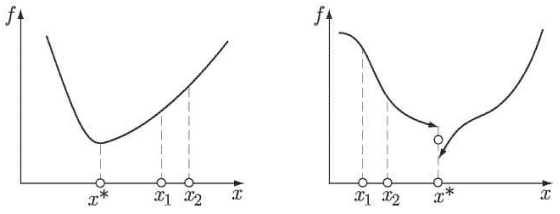
Четыре области, поддерживаемые процедурой PARTITION в подмассиве $A[p..r]$; все элементы подмассива $A[p..i]$ меньше либо равны x, все элементы подмассива $A[i+1..j-1]$ больше x и $A[r]=x$, а все элементы подмассива $A[j..r-1]$ могут иметь любые значения

Среднее количество сравнений для алгоритма Хоара равно $n \cdot \log(n)$, а среднее количество обменов примерно равно $\frac{n}{6} \cdot \log(n)$. Эти величины намного меньше соответствующих характеристик рассмотренных выше алгоритмов сортировки.

Оптимизация одномерных унимодальных функций.

Рассмотрим предварительно методы оптимизации унимодальных функций.

Функция $f(x)$ называется унимодальной функцией на интервале $X=[a,b]$, если существует такая точка $x^* \in X$, что: $f(x_1) > f(x_2)$, если $x_1 < x_2 < x^*$, $x_1, x_2 \in X$; $f(x_1) < f(x_2)$, если $x^* < x_1 < x_2$, $x_1, x_2 \in X$.



Унимодальные функции

***Лемма**. Пусть функция $f(x)$ унимодальна на отрезке $X=[a,b]$, и $x_1 < x_2$, $x_1, x_2 \in X$. Тогда, если $f(x_1) \le f(x_2)$, то $x^* \le x_2$; если же $f(x_1) \ge f(x_2)$, то $x^* \ge x_1$.*

Предположим, что число N вычислений минимизируемой функции является конечным и фиксированным. .

Алгоритм пассивного поиска минимума /Сухарев с46/ .

Если $N = 2K-1$, то $x^{(i)} = a + \frac{b-a}{N} \cdot i;\ i = 1,2,...,N$.

Если же $N = 2K$, то $x^{(2i)} = a + \frac{b-a}{K+1} \cdot i;\ x^{(2i-1)} = x^{(2i)} - \delta;\ i = 1,2,...,N$. Здесь δ некоторое малое положительное число.

Задача минимизации решается как задача нахождения минимального из чисел $f(x^{(1)}), f(x^{(2)}),..., f(x^{(N)})$. То есть $f(x^*) \approx \min\limits_{i=1,...,N} f(x^{(i)})$.

Говорят, что точка минимума x^* локализована в отрезке, если $x^* \in [x^{(i-1)}, x^{(i+1)}]$. Оценка длины отрезка локализации имеет следующий вид:

$$\max_{i=2,\dots,N-1}(x^{(i+1)} - x^{(i-1)}) = \begin{cases} 2 \cdot \dfrac{b-a}{N+1}, & N = 2K-1 \\ \dfrac{b-a}{1+\dfrac{N}{2}} + \delta, & N = 2k \end{cases}.$$

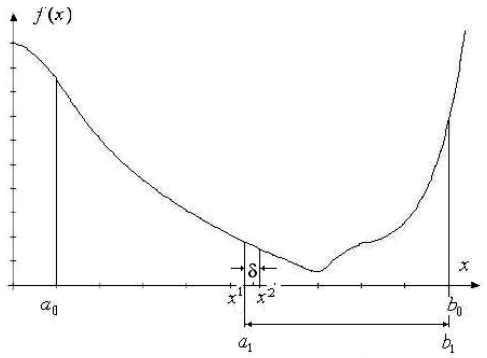
При этом значение $\delta > 0$ может быть выбрано сколь угодно малым.

Метод дихотомии.

Вычисляем две точки согласно следующим соотношениям /Лемешко с7/ :

$$a^{(0)} = a,\ b^{(0)} = b,\ \delta < \varepsilon,\ x^{(1)} = \frac{a^{(0)} + b^{(0)} - \delta}{2},\ x^{(2)} = \frac{a^{(0)} + b^{(0)} + \delta}{2}.$$

После этого вычисляем значения $f(x^{(1)}),\ f(x^{(2)})$.



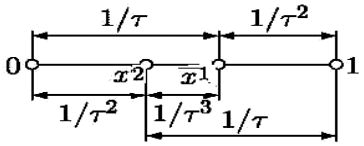
Если $f(x^{(1)}) < f(x^{(2)})$, то $a^{(1)} \leftarrow a^{(0)},\ b^{(1)} \leftarrow x^{(2)}$. Если же $f(x^{(1)}) > f(x^{(2)})$, то $a^{(1)} \leftarrow x^{(1)},\ b^{(1)} \leftarrow b^{(0)}$. Далее, с помощью найденных точек определяем новый интервал неопределенности. Поиск заканчивается, если длина интервала неопределенности $[a^{(k)}, b^{(k)}]$ на итерации k становится меньше заданной точности: $|b^{(k)} - a^{(k)}| < \varepsilon$.

В данном методе, на каждой итерации, минимизируемая функция $f(x)$ вычисляется дважды, а интервал неопределенности сокращается практически в два раза (при малых $\delta < \varepsilon$).

Метод «золотого сечения» /Сухарев с49 Лемешко с8/ :

Метод «золотого сечения» можно прояснить с помощью следующих рассуждений. Пусть $X = [0,1]$ и τ является решением уравнений $\tau^2 = \tau + 1$, ($\tau = \dfrac{1+\sqrt{5}}{2}$). Тогда справедливо следующее соотношение: $1 - \dfrac{1}{\tau} = \dfrac{1}{\tau^2}$. Выберем $x^{(1)} = \dfrac{1}{\tau}$. Тогда отношение длины всего отрезка X к длине большей из его частей $[0, x^{(1)}]$ равно отношению большей части $[0, x^{(1)}]$ к длине меньшей части $[x^{(1)}, 1]$: $\dfrac{1}{1/\tau} = \dfrac{1/\tau}{1/\tau^2} = \tau$. Деление отрезка в таком отношении носит название

«золотого сечения». Точку $x^{(2)}$ будем выбирать симметрично точке $x^{(1)}$ относительно середины отрезка $X=[0,1]$: $x^{(2)}=\dfrac{1}{\tau^2}$.



Взаимное расположение первых двух вычислений по методу золотого сечения

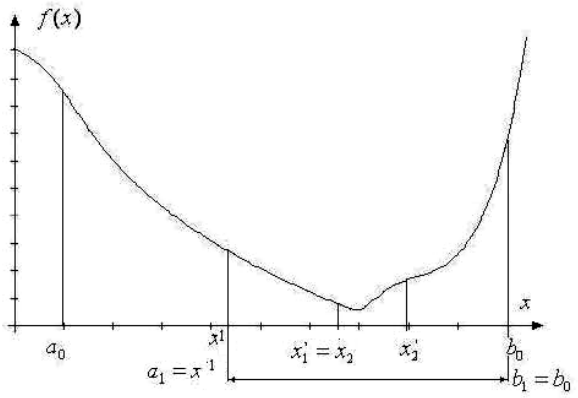
Тогда можно записать следующие соотношения. На первой итерации находим две точки по формулам:

$$a^{(0)}=a,\ b^{(0)}=b,$$

$$x^{(1)}=b^{(0)}+\frac{(\sqrt{5}-3)}{2}\cdot(b^{(0)}-a^{(0)})=b^{(0)}-0.38196\cdot(b^{(0)}-a^{(0)});$$

$$x^{(2)}=a^{(0)}+\frac{(3-\sqrt{5})}{2}\cdot(b^{(0)}-a^{(0)})=a^{(0)}+0.38196\cdot(b^{(0)}-a^{(0)})\ .$$

Затем вычисляем значения $f(x^{(1)}),\ \ f(x^{(2)})$.



Если $f(x^{(1)})<f(x^{(2)})$, то $a^{(1)}\leftarrow a^{(0)},\ b^{(1)}\leftarrow x^{(2)},\ x^{(2)}\leftarrow x^{(1)}$. Если же $f(x^{(1)})>f(x^{(2)})$, то $a^{(1)}\leftarrow x^{(1)},\ b^{(1)}\leftarrow b^{(0)},\ x^{(1)}\leftarrow x^{(2)}$.

На последующих итерациях производим расчет только той точки и значение функции в ней, которые необходимо обновить. В первом случае вычисляем $x^{(1)}$ и $f(x^{(1)})$. Во втором - $x^{(2)}$ и $f(x^{(2)})$. Поиск прекращается при выполнении условия $|b^{(k)}-a^{(k)}|<\varepsilon$. На k - ой итерации интервал неопределенности сокращается до величины $0.61803\cdot|b^{(k-1)}-a^{(k-1)}|$.

Взаимное расположение генерируемых алгоритмом точек (в случае нормированного интервала $X=[0,1]$) поясняет приведенная таблица, где через значения r_1 и r_2 обозначены расстояния точки до, соответственно, ближнего и дальнего концов отрезка.

Число проведенных вычислений	r_1	r_2	Длина отрезка локализации
1	$1/\tau^2$	$1/\tau$	1
2	$1/\tau^3$	$1/\tau^2$	$1/\tau$
...	...	...	...
i	$1/\tau^{i+1}$	$1/\tau^i$	$1/\tau^{i-1}$
...	...	...	...

Метод Фибоначчи /Сухарев с47 Лемешко с9/ .

Следует сразу отметить, что алгоритм Фибоначчи очень похож на метод «золотого сечения».

Последовательность чисел Фибоначчи подчиняется соотношению $F_{k+2} = F_{k+1} + F_k$, $k = 1, 2, \ldots$ и $F_1 = F_2$. Ряд чисел Фибоначчи имеет вид $1, 1, 2, 3, 5, 8.13, \ldots, k$. Имеется также формула Бинэ для прямого вычисления чисел Фибоначчи:

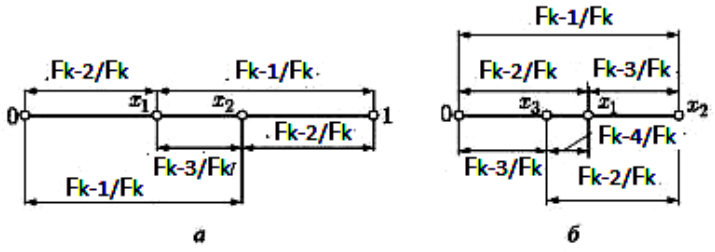
$$F_k = \frac{(\frac{1+\sqrt{5}}{2})^k - (\frac{1-\sqrt{5}}{2})^k}{\sqrt{5}}.$$

Приближенно, для больших значений n, можно записать следующее соотношение: $F_k \approx \frac{(\frac{1+\sqrt{5}}{2})^k}{\sqrt{5}}$.

Выберем некоторое значение k и, пусть $X = [0,1]$. Тогда первые два вычисления проводятся в точках:

$$x^{(1)} = \frac{F_{k-2}}{F_k},\ x^{(2)} = \frac{F_{k-1}}{F_k},$$

которые располагаются симметрично относительно середины отрезка $X = [0,1]$. Следующая же точка выбирается симметрично относительно середины отрезка локализации уже проведенного вычисления ($x^{(1)}$ или $x^{(2)}$).



Взаимное расположение первых трех вычислений по методу Фибоначчи

$$x^{(1)} = a^{(0)} + \frac{F_k}{F_{k+2}} \cdot (b^{(0)} - a^{(0)});\ x^{(2)} = a^{(0)} + \frac{F_{k+1}}{F_{k+2}} \cdot (b^{(0)} - a^{(0)}) = a^{(0)} + b^{(0)} - x^{(1)}.$$

На i - ой итерации, при фиксированном значении k, получаем точку с минимальным значением, которая совпадает с одной из точек, вычисляемых по формулам:

$$x^{(1)} = a^{(i)} + \frac{F_{k-i+1}}{F_{k-i+3}} \cdot (b^{(i)} - a^{(i)}) = a^{(i)} + \frac{F_{k-i+1}}{F_{k+2}} \cdot (b^{(0)} - a^{(0)});,$$

$$x^{(2)} = a^{(i)} + \frac{F_{k-i+2}}{F_{k-i+3}} \cdot (b^{(i)} - a^{(i)}) = a^{(i)} + \frac{F_{k-i+2}}{F_{k+2}} \cdot (b^{(0)} - a^{(0)}).$$

Данные точки расположены на отрезке $[a^{(i)}, b^{(i)}]$ симметрично относительно его середины. При условии, $i = k$ получим:

$$x^{(1)} = a^{(k)} + \frac{F_1}{F_{k+2}} \cdot (b^{(0)} - a^{(0)});, \quad x^{(2)} = a^{(k)} + \frac{F_2}{F_{k+2}} \cdot (b^{(0)} - a^{(0)}).$$

То есть точки совпадают и делят отрезок $[a^{(k)}, b^{(k)}]$ пополам. Отсюда:

$$\frac{b^{(k)} - a^{(k)}}{2} = \frac{b^{(0)} - a^{(0)}}{F_{k+2}} < \varepsilon.$$

Из этого условия обычно выбирают требуемое k и определяют необходимое число итераций. Возможно, из-за вычислительных погрешностей, потеря точности и, соответственно, потеря интервала с точкой минимизации.

Метод квадратичной интерполяции или метод парабол /Сухарев с51, Банди с26/.

Пусть известны значения функции $f(x)$ в трех различных точках α, β, γ: $f_\alpha, f_\beta, f_\gamma$. Тогда функцию $f(x)$ можно аппроксимировать многочленом $\varphi(x)$ второго порядка вида:

$$\varphi(x) = A \cdot x^2 + B \cdot x + C.$$

Параметры A, B, C можно определить из условий:

$$f_\alpha = \varphi(\alpha), \quad f_\beta = \varphi(\beta), \quad f_\gamma = \varphi(\gamma),.$$

Многочлен $\varphi(x)$ будет иметь минимум в точке $x = -\frac{B}{2A}$, если $A > 0$. То есть, значение точки минимума функции $f(x)$ можно аппроксимировать значением точки минимума многочлена $\varphi(x)$, которая определяется соотношением:

$$\delta = \frac{1}{2} \cdot \frac{(\beta^2 - \gamma^2) \cdot f_\alpha + (\gamma^2 - \alpha^2) \cdot f_\beta + (\alpha^2 - \beta^2) \cdot f_\gamma}{(\beta - \alpha) \cdot f_\alpha + (\gamma - \alpha) \cdot f_\beta + (\alpha - \beta) \cdot f_\gamma}.$$

Вычислительная процедура имеет следующие шаги.

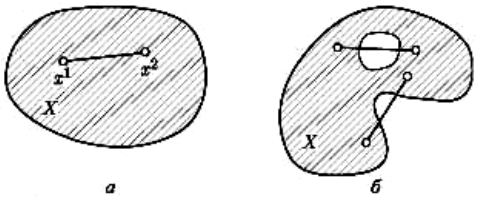
1. Вычислить $f(x^{(1)}), \ f(x^{(2)})$, где $x^{(2)} = x^{(1)} + h$.
2. Если $f(x^{(1)}) < f(x^{(2)})$, то взять в качестве третьей точки $x^{(3)} = x^{(1)} - h$ и вычислить значение функции $f(x^{(3)})$. В противном случае, в качестве третьей точки взять значение $x^{(3)} = x^{(1)} + 2h$ и вычислить значение функции $f(x^{(3)})$.
3. Используя полученные три точки, найти по формуле значение δ и вычислить значение функции $f(\delta)$.
4. Если разница между наименьшим значением функции на k - ой итерации и наименьшим значением функции на $(k+1)$ - ой итерации меньше заданной точности, то завершить процедуру поиска минимума.
5. Если процедура на шаге 4 не завершилась, то точка с наибольшим значением отбрасывается и процедура возвращается к шагу 3.

Метод квадратичной интерполяции также часто называют методом Пауэлла.

Элементы выпуклого анализа.

Определение. Множество $X \subseteq \Re^n$ *называется выпуклым, если* $\lambda \cdot x^{(1)} + (1-\lambda) \cdot x^{(2)} \in X$ *при* $\forall x^{(1)}, x^{(2)} \in X,\ \lambda \in [0,1]$.

То есть, множество X выпукло, если оно вместе с любыми своими точками $\forall x^{(1)}, x^{(2)}$, содержит соединяющий их отрезок /Сухарев с16,61 Банди с92/.



Выпуклое множество (*а*); невыпуклое множество (*б*)

Приведем примеры выпуклых множеств в пространстве: $X \subseteq \Re^n$. Это – само пространство $\Re^n$, его любое линейное подпространство, шар, отрезок.

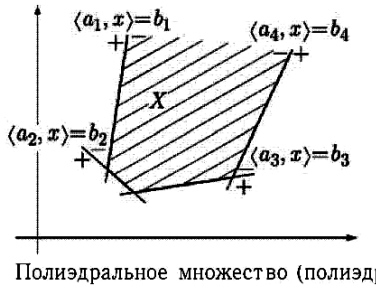
В выпуклом анализе важную роль играют также следующие множества:

$L_{x_0,h} = \{x \in \Re^n : x = x_0 + k \cdot h, k \in \Re\}$ - прямая линия, проходящая через точку x_0 в направлении вектора h;

$H_{p,\beta} = \{x \in \Re^n : (p,x) = \beta, \beta \in \Re, p \in \Re^n\}$ - гиперплоскость с нормалью p, где $(p,x) = p^T \cdot \beta$.

$H^+_{p,\beta} = \{x \in \Re^n : (p,x) \geq \beta, \beta \in \Re, p \in \Re^n\}$ и $H^-_{p,\beta} = \{x \in \Re^n : (p,x) \leq \beta, \beta \in \Re, p \in \Re^n\}$ - полупространства, порождаемые плоскостью $H_{p,\beta}$.

Всеми перечисленные множества (кроме шара) являются частными случаями класса выпуклых множеств вида: $X = \{x \in \Re^n : A \cdot x \leq b, b \in \Re^m, A \in \Re^{m \times n}\}$, которые называются полиэдрами.

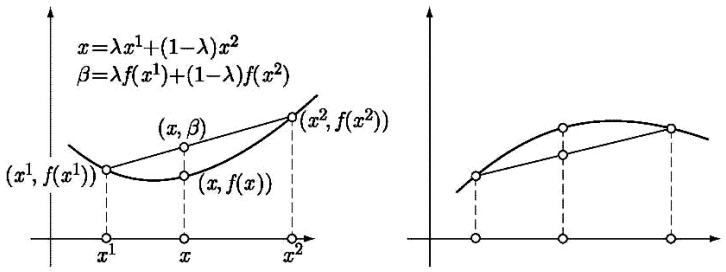


Полиэдральное множество (полиэдр)

Функция $f(x)$, определяемая на выпуклом множестве $X \subseteq \Re^n$, называется выпуклой, если:

$$f(\lambda \cdot x^{(1)} + (1-\lambda) \cdot x^{(2)}) \leq \lambda \cdot f(x^{(1)}) + (1-\lambda) \cdot f(x^{(2)}),\ \forall x^{(1)}, x^{(2)} \in X, \lambda \in [0,1].$$

Если для $\forall x^{(1)}, x^{(2)} \in X, x^{(1)} \neq x^{(2)}$ данное неравенство выполняется как строгое, то функция $f(x)$ называется строго выпуклой на $X \subseteq \Re^n$. Функция $f(x)$ называется (строго) вогнутой, если функция $-f(x)$ (строго) выпукла.



Выпуклая функция (*а*), вогнутая функция (*б*)

\

Пример.

Функция $y = \| x \|$ является выпуклой. Действительно, для любых $x_1, x_2 \in M \subset \Re^n$ имеем $y(\lambda x_1 + (1-\lambda)x_2) = \| \lambda x_1 + (1-\lambda)x_2 \| \le \lambda \| x_1 \| + (1-\lambda)\cdot \| x_2 \| = \lambda y(x_1) + (1-\lambda) y(x_2)$.

Квадратичная форма $y = x^T Bx + bx + c$, где B неотрицательно-определенная матрица, также является выпуклой функцией.

Свойства выпуклых функций.

1. Выпуклые функции непрерывны во всех внутренних точках области определения.
2. Поставим в соответствие выпуклой функции $f(x)$ следующее множество $epi\{f(x)\} = \{(x,z) : x \in M \subset \Re^n, z \ge f(x)\}$, называемое надграфиком. Тогда анализ на выпуклость функции $f(x)$ можно заменить исследованием на выпуклость ее надграфика $epi\{f(x)\}$.
3. Неравенство Иенсена. Пусть $f(x)$ - выпуклая функция, $x \in M \subset \Re^n$. Тогда справедливо следующее соотношение: $f(\sum_i \alpha_i x_i) \le \sum_i \alpha_i f(x_i)$, $\sum_i \alpha_i = 1; \alpha_i \ge 0; x_i \in M; i = 1:m$
4. Если $f(x)$ - выпуклая функция, то множество $F = \{x : f(x) \le \gamma, \gamma = const\}$ является выпуклым. Для вогнутой функции $g(x)$ множество $G = \{x : g(x) \ge \gamma, \gamma = const\}$ также является выпуклым.
5. Пусть $h(x)$ - выпуклая функция, а $g(y)$ - выпуклая и неубывающая функция. Тогда функция $f(x) = g(h(x))$ выпуклая функция.

Теорема (критерий выпуклости). *Пусть M - непустое открытое выпуклое множество в $\Re^n$, $f(x)$ - дифференцируемая на M функция. Для того, чтобы функция $f(x)$ была выпуклой, необходимо и достаточно, чтобы при любых $x^{(1)}, x^{(2)} \in M \subset \Re^n$ выполнялось неравенство $[\nabla f(x^{(2)}) - \nabla f(x^{(1)})] \cdot (x^{(2)} - x^{(1)}) \ge 0$. Для строгой выпуклости $f(x)$ необходимо и достаточно, чтобы неравенство было строгим.*

Очевиден геометрический смысл этой теоремы. Дифференцируемая функция является выпуклой тогда и только тогда, когда график ее целиком лежит выше касательной гиперплоскости, проведенной в любой точке поверхности функции.

Теорема. *Пусть M - непустое открытое выпуклое множество в $\Re^n$, $f(x)$ - дважды дифференцируемая на M функция. Для того, чтобы функция $f(x)$ была выпуклой, необходимо и достаточно, чтобы матрица Гессе была неотрицательно-определенной в каждой точке M.*

В частности, если $f(x)$ - квадратичная функция, то матрица Гессе не зависит от точки, в которой она вычисляется. Следовательно, проверка на выпуклость сводится к проверке неотрицательной определенности матрицы при старшем члене.

Теорема (о глобальном экстремуме выпуклой задачи). *Если в задаче* $f(x) \to \min, x \in X$ *функция* $f(x)$ *выпукла, а* X *– выпуклое множество, то любое ее локальное решение является также глобальным.*

Доказательство. Пусть x^* - локальное решение. Тогда $\exists \varepsilon > 0$ такое, что $f(x^*) < f(x)$ для $\forall x \in X \cap U_\varepsilon(x^*)$, где $U_\varepsilon(x^*) = \{x \in \Re^n : \| x - x^* \| \le \varepsilon\}$. Так как множество X выпукло, то справедливо соотношение $\lambda \cdot x + (1-\lambda) \cdot x^* \in X$. Следовательно, $f(x^*) \le f(\lambda \cdot x + (1-\lambda) \cdot x^*) \le \lambda \cdot f(x) + (1-\lambda) \cdot f(x^*)$. Отсюда $f(x^*) < f(x)$ для $\forall x \in X$. То есть, x^* является глобальным решением задачи $f(x) \to \min, x \in X$.

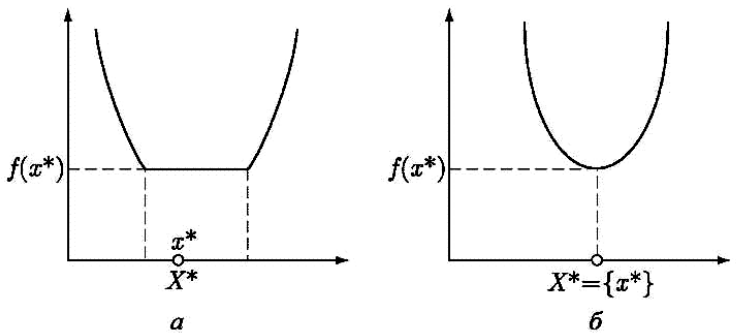
Теорема (о выпуклых задачах безусловной оптимизации). *Пусть функция* $f(x)$ *выпукла на пространстве* $\Re^n$ *и дифференцируема в точке* $x^* \in \Re^n$. *Если вектор градиента* $\nabla f(x^*) = 0$, *то* x^* *- точка минимума функции* $f(x)$ *на пространстве* $\Re^n$, *то есть является решением задачи безусловной оптимизации* $f(x) \to \min, x \in \Re^n$.

Доказательство. Для $\forall x \in \Re^n$ и $\lambda \in [0,1]$ справедливо соотношение: $f(\lambda \cdot x + (1-\lambda) \cdot x^*) \le \lambda \cdot f(x) + (1-\lambda) \cdot f(x^*)$. Тогда можно записать следующие соотношения:

$$f(x) - f(x^*) \ge \frac{f(\lambda \cdot x + (1-\lambda) \cdot x^*) - f(x^*)}{\lambda} = \frac{(\nabla f(x^*), \lambda \cdot (x - x^*)) + o(\lambda)}{\lambda} \overset{\nabla f(x^*)=0}{=} \frac{o(\lambda)}{\lambda}.$$

Отсюда, используя предельный переход, при $\lambda \to 0$, получим, что $f(x^*) \le f(x)$.

Теорема (о выпуклых задачах условной оптимизации). *Пусть задача* $f(x) \to \min, x \in X \subseteq \Re^n$ *выпукла и имеет решение. Тогда множество ее решений* $X^* = \arg\min_{x \in X}(f(x))$ *выпукло. Если при этом* $f(x)$ *строго выпукла на* X, *то решение задачи единственно, то есть* X^* *состоит из одной точки.*
(без доказательства)



Выпуклость множества решений выпуклой задачи оптимизации

Отметим еще несколько важных свойств выпуклых функций. Если $f(x)$ выпуклая функция на выпуклом множестве X и $x^{(1)}, x^{(2)} \in X$, то справедливо соотношение:

$$f(x^{(2)}) \geq f(x^{(1)}) + \nabla f(x^{(1)}) \cdot (x^{(2)} - x^{(1)}).$$

Функция $f(x)$ является выпуклой на выпуклом множестве X, если гессиан $H = (\frac{\partial^2 f}{\partial x_i \cdot \partial x_j})$ положительно определен. Для выпуклых функций это фактически означает, что первая производная является возрастающей функцией и, следовательно, может обращаться в нуль только в одной точке. То есть функция имеет одну точку минимума.

Методы безусловной локальной многомерной оптимизации.

Рассмотрим методы решения задач вида $f(x) \to \min, x \in \Re^n$. Обычно область поиска в таких задачах ограничивается некоторым параллелепипедом $D = \{x \in \Re^n : a \leq X \leq b\}$.

Данное ограничение учитывается, чаще всего, только при выборе шагового множителя. В остальном, при описании методов, этим ограничением пренебрегают.

Почти все методы безусловной оптимизации можно описать с помощью итерационных соотношений:

$x^{(k+1)} = x^{(k)} + \alpha^{(k)}((x^{(k)}) \cdot h^{(k)}) \cdot h^{(k)}$ или $x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot h^{(k)}$,

где $x^{(k)}$ - точка основных вычислений функции $f(x)$ на k - ой итерации, $h^{(k)}$ - направление смещения из точки $x^{(k)}$, $\alpha^{(k)}$ - шаговый множитель. Переход от точки $x^{(k)}$ к точке $x^{(k+1)}$ выполняется таким образом, чтобы обеспечить существенное убывание функции $f(x)$ на каждом шаге. То есть, практически все методы безусловной оптимизации реализуют элементы, так называемой, «жадной» стратегии. «Жадные» алгоритмы позволяют получить оптимальное (или приближенное к оптимальному) решение задачи путем выбора на каждом шаге такого локального выбора, который в данный момент выглядит наилучшим /Кормен с453/. Такие методы оптимизации, когда на каждом шаге $f(x^{(k+1)}) \leq f(x^{(k)})$ называют еще релаксационными.

В качестве одного из условий останова в методах локальной оптимизации часто используется оценка нормы градиента $\| \nabla f(x^{(k)}) \| \leq \varepsilon$. Конечно, в общем случае, такой критерий не гарантирует близость точки $x^{(k)}$ к оптимальной точке решения задачи по координатам. Для методов, не использующих вычисление градиента, правило останова строится на основе следующих критериев:

$\| x^{(k+1)} - x^{(k)} \| \leq \varepsilon$ или $\| f(x^{(k+1)}) - f(x^{(k)}) \| \leq \varepsilon$.

Методы безусловной оптимизации можно классифицировать следующим образом.

Если метод использует в вычислениях значения производных целевой функции для n - го порядка, то его относят к методам n - го порядка. Обычно выделяют методы нулевого, первого и второго порядков. Если метод нулевого порядка не использует предположений о гладкости целевой функции, то его называют методом прямого поиска.

Выбор коэффициента шагового множителя.

Важным элементом вычислений во всех методах является выбор стратегии формирования шагового множителя. Данный элемент в процессе выполнения алгоритма вызывается тысячи раз. Поэтому эффективность процедуры выбора данного элемента во многом обуславливает эффективность всего алгоритма. Выделяют обычно следующие подходы: постоянный шаговый множитель, когда шаг не зависит от значения $x^{(k)}$, определяется только классом принадлежности

функции $f(x)$ и рассчитывается перед началом итерационной процедуры; и переменный шаговый множитель, когда шаг изменяется со значением $x^{(k)}$.

При переменном шаге, обычно, при приближении к оптимальной точке по критерию близости, заданном в алгоритме, шаговый множитель уменьшается по некоторому закону. Такой закон подбирается из следующих условий: шаговый множитель не может быть большим, чтобы не нарушить сходимость алгоритма для большинства исследуемых функций $f(x)$, но, с другой стороны, не должен быть и слишком малым, чтобы не увеличить резко вычислительные затраты на решение задачи.

Рассмотрим более детально процедуры выбора шагового множителя. /Бертсекас с28/

1. *Одномерная минимизация.* Параметр $\alpha^{(k)}$ выбирается из условия:

$$\alpha^{(k)} = \arg\min_{\alpha \geq 0}(f(x^{(k)} + \alpha \cdot h^{(k)}).$$

Очевидно, что решить точно эту задачу нельзя. Поэтому необходимо связывать точность выбора шагового множителя с точностью решения одномерной минимизации и, соответственно, с точностью решения всей задачи многомерной задачи. Вычислительная сложность итогового алгоритма будет, при этом, значительно возрастать.

2. *Одномерная минимизация с ограничением.* Шаговый множитель $\alpha^{(k)}$ выбирается из условия:

$$\alpha^{(k)} = \arg\min_{\alpha=\{0,S\}}(f(x^{(k)} + \alpha \cdot h^{(k)}),$$

где S - некоторое множество ограничений на коэффициент α.

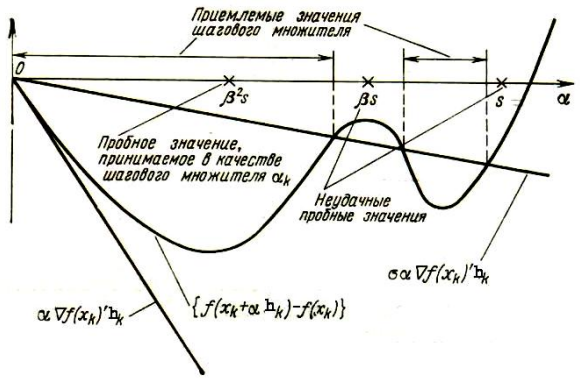
3. *Правило Армихо.* При фиксированных значениях параметров:

$$L > 0,\ \beta \in (0,1),\ \sigma \in (0,0.5),$$

полагают $\alpha^{(k)} = \beta^{m_k} \cdot L$, где m_k наименьшее из неотрицательных чисел, для которых справедливо неравенство:

$$f(x^{(k)}) - f(x^{(k)} + \beta^{m_k} \cdot L \cdot h^{(k)}) \geq -\sigma \cdot \beta^{m_k} \cdot L \cdot \nabla f(x^{(k)}) \cdot h^{(k)}.$$

При этом способе обе части неравенства вычисляются для $m_k = 0,1,2,...$ до тех пор, пока при некотором значении m_k неравенство не станет справедливым. Параметры процесса обычно выбирают из следующих диапазонов: $\beta \in [0.1, 0.5],\ \sigma \in [10^{-5}, 10^{-1}]$.



Выбор длины шага по правилу Армихо

4. *Правило Голдстейна.* При фиксированном значении $\sigma \in (0, 0.5)$, шаговый множитель $\alpha^{(k)}$ выбирается из условия:

$$\sigma \le \frac{f(x^{(k)} + \alpha^{(k)} \cdot h^{(k)}) - f(x^{(k)})}{\alpha^k \cdot \nabla f(x^{(k)}) \cdot h^{(k)}} \le 1 - \sigma \quad \sigma \le \frac{f(x^{(k)} + \alpha^{(k)} h^{(k)}) - f(x^{(k)})}{\alpha^{(k)} (\nabla f(x^{(k)})^T h^{(k)}} \le 1 - \sigma .$$

5. *Постоянный шаговый множитель*. Шаговый множитель $\alpha^{(k)} = L$.

Методы прямого поиска (методы нулевого порядка).

Данные методы не требуют знания целевой функции в явном виде, ее регулярности, гладкости и существования производных. Однако эти методы требуют повышенных вычислительных затрат и, в общем случае, обеспечивают меньшую точность из-за трудностей определения правила останова /Лемешко с11/.

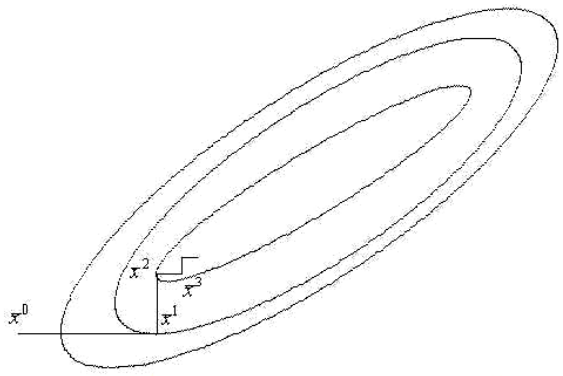
Алгоритм Гаусса.

Данный алгоритм заключается в минимизации на каждом шаге только одной компоненты вектора переменных $x \in \Re^n$. Пусть задана начальная точка $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})^T$. На первом шаге решается задача:

$$x_1^{(1)} = \arg\min_{x_1 \in \Re} (f(x_1, x_2^{(0)}, ..., x_n^{(0)}),$$

и формируется новая точка $x^{(1)} = (x_1^{(1)}, x_2^{(0)}, ..., x_n^{(0)})^T$. Далее ищется минимум по второй координате вектора $x^{(1)} \in \Re^n$ и так далее. Через n шагов получаем точку $x^{(n)} = (x_1^{(1)}, x_2^{(2)}, ..., x_n^{(n)})^T$, начиная с которой процесс возобновляется по первой переменной. В качестве условий прекращения поиска обычно используются следующие критерии:

$$\| x^{(k+1)} - x^{(k)} \| \le \varepsilon \quad \text{или} \quad \| f(x^{(k+1)}) - f(x^{(k)}) \| \le \varepsilon .$$



Пример траектории спуска в алгоритме Гаусса

Метод Гаусса не очень эффективен. Его сходимость очень низкая при, так называемой, «овражности» целевой функции. Поиск на таких функциях быстро «застревает» на дне оврага и алгоритм останавливается вдали от точки минимума целевой функции.

Алгоритм Хука-Дживса.

Алгоритм включает в себя два основных этапа:

- исследующий поиск в окрестности точки $x^{(k)}$;
- этап поиска по образцу в направлении, выбранном для минимизации.

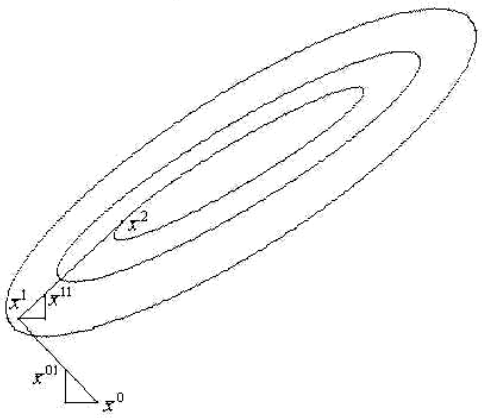
В рамках исследующего поиска делается пробный шаг по первой компоненте вектора $x^{(0)}$, то есть вычисляется функция в точке $x^{(1,1)} = (x_1^{(0)} + \Delta x, x_2^{(0)}, ..., x_n^{(0)})^T$. Если $f(x^{(1,1)}) > f(x^{(0)})$, то точку $x^{(0)}$ оставляют без изменений и делают новый пробный шаг, в противоположном направлении. То есть определяют $x^{(1,2)} = (x_1^{(0)} - \Delta x, x_2^{(0)}, ..., x_n^{(0)})^T$. Если значение функции и в точке $x^{(1,2)}$ больше $f(x^{(0)})$, то оставляют точку $x^{(0)}$ без изменений. Если выполняется условие $f(x^{(1,2)}) < f(x^{(0)})$, то полагают $x^{(0,1)} = x^{(1,2)}$. Из вновь полученной точки $x^{(0,1)}$ делают пробные шаги по оставшимся координатам, используя туже самую процедуру, что и для первой компоненты вектора $x^{(0)}$.

Если в процессе исследующего поиска не удалось сделать ни одного удачного пробного шага, то Δx уменьшают и повторяют исследующий поиск.

Рассмотрим теперь этап поиска по образцу. После исследующего поиска получают точку $x^{(0,n)}$ и определяют направление $(x^{(0,n)} - x^{(0)})$, в котором функция уменьшается. Для выбранного направления решают задачу одномерной оптимизации:

$$\lambda_{opt} = \arg(\min_{\lambda}(f(x^{(0)} + \lambda \cdot (x^{(0,n)} - x^{(0)}))).$$

Затем находится новое приближение $x^{(1)} = x^{(0)} + \lambda_{opt} \cdot (x^{(0,n)} - x^{(0)})$.



Пример траектории спуска в алгоритме Хука и Дживса

В точке $x^{(1)}$ начинают новый исследующий поиск и, цикл итераций повторяется.

Алгоритм Розенброка.

Алгоритм Розенброка называют также еще методом вращающихся координат. Этот метод особенно эффективен при нахождении точки минимума функций овражного типа.

Общая идея метода заключается в том, что выбирается в каждой k - ой точке итерации система ортогональных направлений $(S_1^{(k)}, S_2^{(k)}, ..., S_n^{(k)})$. Затем, в каждом из ортогональных направлений ищется минимальное значение целевой функции. Система координат поворачивается так, чтобы одна из осей совпала с направлением наибольшего убывания целевой функции, а остальные направления были ортогональными между собой. В качестве начального приближения системы координат на первом шаге обычно используется система ортонормированных векторов.

Пусть $x^{(0)}$ - вектор начальных приближений; $(S_1^{(0)}, S_2^{(0)}, ..., S_n^{(0)})$ - начальная система ортогональных направлений. Последовательно проводится минимизация функции $f(x)$ в ортогональных направлениях по следующим соотношениям:

$$\lambda_1 = \arg\min_{\lambda}(f(x^{(0)} + \lambda \cdot S_1^{(0)}); \;\; x_1^{(1)} = x_1^{(0)} + \lambda_1 \cdot S_1^{(0)};$$
$$.....$$
$$\lambda_n = \arg\min_{\lambda}(f(x^{(0)} + \lambda \cdot S_n^{(0)}); \;\; x_n^{(1)} = x_n^{(0)} + \lambda_1 \cdot S_n^{(0)};$$

Следующую итерацию начинают с точки $x^{(1)}$. Если не изменять направление, то такая процедура будет совпадать с алгоритмом Гаусса. Поэтому после некоторой k - ой итерации вычисляется новая система координат. Ортогональные направления поиска поворачиваются так, чтобы основное направление поиска совпадало с направлением наибольшего уменьшения функции – вдоль оврага (соответствует максимальным значениям λ_i). При этом для оценки целевой функции используется обычно ее квадратичная аппроксимация.

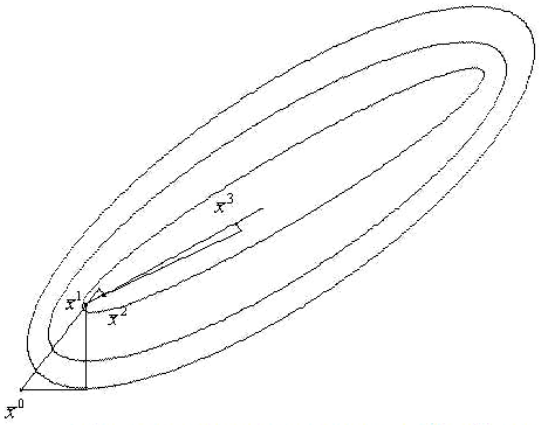
Рассмотрим процедуру изменения (вращения) координат.

Пусть в результате минимизации по каждому из ортогональных направлений получают систему параметров $(\lambda_1^{(k)}, \lambda_2^{(k)}, ..., \lambda_n^{(k)})$, на основе которой формируется система векторов $(A_1^{(k)}, A_2^{(k)}, ..., A_n^{(k)})$ по следующим правилам:

$$A_1^{(k)} = \lambda_1^{(k)} \cdot S_1^{(k)} + \lambda_2^{(k)} \cdot S_2^{(k)} + ... + \lambda_n^{(k)} \cdot S_n^{(k)};$$
$$A_2^{(k)} = \lambda_2^{(k)} \cdot S_2^{(k)} + ... + \lambda_n^{(k)} \cdot S_n^{(k)};$$
$$.....$$
$$A_n^{(k)} = \lambda_n^{(k)} \cdot S_n^{(k)}$$

С помощью векторов $(A_1^{(k)}, A_2^{(k)}, ..., A_n^{(k)})$ строится новая система ортогональных направлений $(S_1^{(k+1)}, S_2^{(k+1)}, ..., S_n^{(k+1)})$. Причем первый вектор направляется так, чтобы он совпадал с направлением общего перемещения на k - ом шаге, а остальные направления получаются с помощью процедуры ортогонализации Грама-Шмидта:

$$S_1^{(k+1)} = \frac{A_1^{(k)}}{\| A_1^{(k)} \|}, \;\; B_2^{(k)} = A_2^{(k)} - ((A_2^{(k)})^T \cdot S_1^{(k+1)}) \cdot S_1^{(k+1)};$$
$$S_2^{(k+1)} = \frac{B_2^{(k)}}{\| B_2^{(k)} \|}, \;\; .....$$
$$.....$$
$$B_j^{(k)} = A_j^{(k)} - \sum_{l=1}^{j-1} ((A_{l+1}^{(k)})^T \cdot S_l^{(k+1)}) \cdot S_l^{(k+1)}, \;\; S_j^{(k+1)} = \frac{B_j^{(k)}}{\| B_j^{(k)} \|}, \;\; j = 2,3,...,n$$



Пример траектории спуска в алгоритме Розенброка

Для эффективной работы алгоритма необходимо, чтобы числа $(\lambda_1^{(k)}, \lambda_2^{(k)}, ..., \lambda_n^{(k)})$ располагались в порядке убывания по абсолютному значению. При значениях $|\lambda_i^{(k)}| = 0$ направления соответствующих координат остаются без изменения. В качестве критериев останова алгоритма могут быть использованы такие же правила, как в алгоритмах Гаусса и Хука-Дживса.

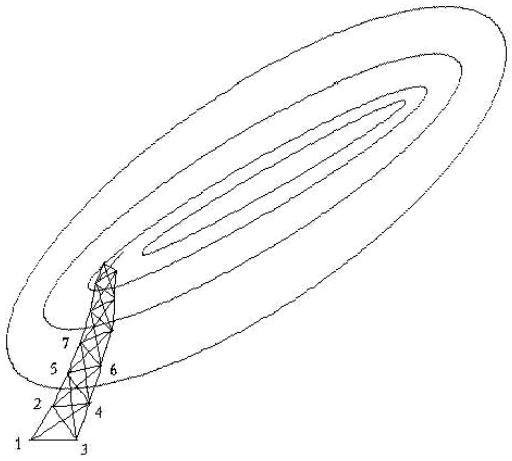
Симплексный метод Нелдера-Мида (поиск по деформируемому многограннику).

Координаты вершин регулярного симплекса в пространстве $\Re^n$ обычно определяются матрицей $A \in \Re^{n\times(n+1)}$, в которой столбцы представляют собой вершины симплекса, пронумерованные от 1 до $(n+1)$, а строки - координаты вершин от 1 до n:

$$A = \begin{pmatrix} 0 & a_1 & a_2 & ...a_2 \\ 0 & a_2 & a_1 & ...a_2 \\ . & . & . & ... \\ 0 & a_2 & a_2 & ...a_1 \end{pmatrix},$$

где $a_1 = \frac{L}{n\sqrt{2}} \cdot (\sqrt{(n+1)} + n - 1),\ a_2 = \frac{L}{n\sqrt{2}} \cdot \frac{1}{(\sqrt{(n+1)} - 1)}$, L - расстояние между вершинами.

На первом шаге алгоритма строится регулярный симплекс. Далее из вершины, в которой функция $f(x)$ имеет максимальное значение (точка 1), проводится проектирующая прямая через центр тяжести симплекса. Затем точка 1 исключается из рассмотрения и строится новый отраженный симплекс из оставшихся старых точек и одной новой точки, расположенной на проектирующей прямой, на некотором расстоянии от центра тяжести. В продолжение работы алгоритма, каждый раз исключается вершина, где целевая функция максимальна. Кроме того, используются правила, предусматривающие уменьшение размера симплекса и предотвращение циклических движений в окрестности точки оптимальности. К сожалению, для функций овражного типа такой метод не очень эффективен.



Рассмотрим k - ю итерацию алгоритма. Пусть $x_i^{(k)} = (x_1^{(i,k)}, x_2^{(i,k)}, ..., x_n^{(i,k)}),\ i = 1:(n+1)$ - является i -ой вершиной симплекса в пространстве $\Re^n$ на k - ом шаге поиска. Отметим вершины с максимальным и минимальным значениями:

$$f(x_h^{(k)}) = \max\{f(x^{(1,k)}), f(x^{(2,k)}), ..., f(x^{(n,k)})\},$$

$$f(x_l^{(k)}) = \min\{f(x^{(1,k)}), f(x^{(2,k)}), ..., f(x^{(n,k)})\}.$$

Обозначим через $x^{((n+2),k)}$ - центр тяжести вершин симплекса без точки $x_h^{(k)}$ с максимальным значением целевой функции. Координаты точки $x^{((n+2),k)}$ вычисляются по формуле:

$$x_j^{((n+2),k)} = \frac{1}{n} \cdot (\sum_{i=1}^{n+1} (x_j^{(i,k)} - x_j^{(n,k)})),\ \ j = 1, 2, ..., n.$$

Обычно на первом шаге начало координат помещают в центр тяжести исходного симплекса.

Алгоритм Нелдера-Мида состоит из следующих основных этапов:

- отражения;
- растяжения;
- сжатия;
- редукции.

1. Отражение представляет собой проектирование точки $x_h^{(k)}$ через центр тяжести $x^{((n+2),k)}$ по следующему правилу: $x^{((n+3),k)} = x^{((n+2),k)} + \alpha \cdot (x^{((n+2),k)} - x^{(n,k)})$, где $\alpha > 0$ – коэффициент отражения.

Если $f(x^{((n+3),k)}) \geq f(x_h^{(k)})$, то переходят к этапу редукции алгоритма. В противном случае, если справедливы соотношения:

$$f(x^{((n+3),k)}) < f(x_h^{(k)}) \ \text{ и } \ f(x^{((n+3),k)}) \geq f(x_l^{(k)}),$$

то выполняют этап сжатия.

2. Этап растяжения заключается в следующем. Если $f(x^{((n+3),k)}) < f(x_l^{(k)})$ (то есть меньше минимального значения на k - ом шаге), то вектор $(x^{((n+3),k)} - x^{((n+2),k)})$ растягивают в соответствии со следующей формулой:

$$x^{((n+4),k)} = x^{((n+2),k)} + \gamma \cdot (x^{((n+3),k)} - x^{((n+2),k)}),$$

где $\gamma > 1$ - коэффициент растяжения.

Если $f(x^{((n+4),k)}) < f(x_l^{(k)})$, то точку $x_l^{(k)}$ заменяют на точку $x^{((n+4),k)}$ и алгоритм продолжается с операции отражения при $k = k+1$. В противном случае точку $x_l^{(k)}$ заменяют на точку $x^{((n+3),k)}$ и затем алгоритм переходит к этапу отражения.

3. Этап сжатия начинается с проверки условия $f(x^{((n+3),k)}) > f(x_l^{(k)})$ для всех $i = 1:(n+1); i \neq h$. Если условие выполняется, то вектор $(x_h^k - x^{((n+2),k)})$ сжимается по следующей формуле:

$x^{((n+5),k)} = x^{((n+2),k)} + \beta \cdot (x_h^{(k)} - x^{((n+2),k)})$,

где $0 < \beta < 1$ - коэффициент сжатия. После этого, точка $x_h^{(k)}$ заменяется на точку $x^{((n+5),k)}$ и алгоритм переходит к этапу отражения при $k = k+1$. При этом, заново ищется точка $x_h^{(k+1)}$.

4. На этапе редукции, если $f(x^{((n+3),k)}) > f(x_h^{(k)})$, то все векторы $(x^{(i,k)} - x_l^{(k)}), i = 1:(n+1)$ уменьшаются в два раза с отсчетом от точки $x_l^{(k)}$ по формуле:

$x^{(i,k)} = x_l^{(k)} + 0.5 \cdot (x^{(i,k)} - x_l^{(k)}), i = 1:(n+1)$ и, осуществляется переход к этапу отражения с $k = k+1$.

В качестве критерия останова могут быть использованы правила, используемые в других алгоритмах прямого поиска. Можно также использовать следующее правило:

$\frac{1}{n+1} \cdot (\sum_{i=1}^{n+1} (f(x^{(i,k)} - f(x^{((n+2),k)}))^2)^{\frac{1}{2}} \leq \varepsilon$. Чаще всего рекомендуется использовать следующие параметры алгоритма: $\alpha = 1,\ 0.4 \leq \beta \leq 0.6,\ 2 \leq \gamma \leq 3$.

<u>Методы оптимизации первого порядка (градиентные методы).</u>

Алгоритм наискорейшего спуска

Градиентные методы предельно просты, а поэтому пользуются большой популярностью. Их динамика, в общем виде, описывается разностным уравнением вида:

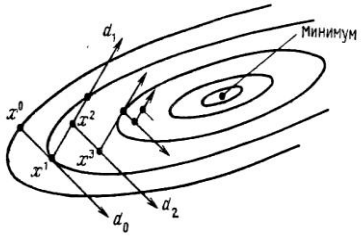
$$x^{(k+1)} = x^{(k)} + \lambda^{(k)} \cdot d^{(k)},$$

где $d^{(k)}$ - вектор направления, совпадающий с направлением $(-\nabla f(x^{(k)}))^T$, а $\lambda^{(k)}$ - масштабный коэффициент шага. Параметр $\lambda^{(k)}$ выбирается на каждом шаге путем решения одномерной задачи оптимизации следующего вида:

$$\lambda^{(k)} = \arg\min_{\lambda}(f(x^{(k)} + \lambda \cdot d^{(k)}).$$

В зависимости от выбора параметра $\lambda^{(k)}$ траектории поиска могут существенно различаться. При большом значении коэффициента шага траектория спуска будет представлять собой колебательный процесс, который может, в некоторых случаях, и расходиться. При выборе малых значений коэффициента шага траектория поиска (спуска) будет плавной, но и процесс поиска будет очень медленным. Каждый последующий шаг итерации будет ортогонален предыдущему, что следует из предыдущих соотношений:

$$\frac{df(x^{(k)} + \lambda \cdot d^{(k)})}{d\lambda} = (-\nabla f(x^{(k)} + \lambda \cdot d^{(k)}) \cdot d^{(k)} = (-\nabla f(x^{(k+1)}) \cdot d^{(k)} = (d^{(k+1)})^T \cdot d^{(k)} = 0.$$



Два последовательных направления перемещения в методе наискорейшего спуска ортогональны

Вообще говоря, алгоритм наискорейшего спуска для целевой функции общего вида может закончиться в любой точке, где $\nabla f(x^{(k)}) = 0$, в том числе и в точке седла. Алгоритм очень плохо сходится для функций овражного типа. Поэтому метод наискорейшего спуска используют обычно в комбинации с другими методами, в том числе и с методами прямого поиска.

Рассмотрим работу метода наискорейшего спуска для квадратичных функций вида:

$$f(x) = \frac{1}{2} \cdot x^T \cdot Q \cdot x,$$

где Q - симметричная положительно-определенная матрица.

Теорема (о сходимости) */Городецкий Прак с22/.* *Для квадратичной функции вида* $f(x) = \frac{1}{2} \cdot x^T \cdot Q \cdot x$ *с симметричной положительно-определенной матрицей* Q *метод наискорейшего градиентного спуска сходится из любой начальной точки* $x^{(0)}$ *со скоростью геометрической прогрессии. При этом справедливы следующие оценки:* $\exists a \geq 0,\ T > 0$, *что выполняются соотношения:*

$$0 \leq a \leq q = \frac{(\frac{\lambda_{\min}}{\lambda_{\max}} - 1)^2}{(\frac{\lambda_{\min}}{\lambda_{\max}} + 1)^2}, \quad |f(x^{(k)} - f(x^*)| \leq a^* \cdot |f(x^{(0)} - f(x^*)|;$$

$$\| x^{(k)} - x^* \| \leq T \cdot a^{\frac{k}{2}} \cdot \| x^{(0)} - x^* \|; \quad ,$$

где $\lambda_{\min}, \lambda_{\max}$ *минимальное и максимальное собственные числа матрицы* $H = (\frac{\partial^2 f}{\partial x_i \cdot \partial x_j})$. *При* $\lambda_{\min} << \lambda_{\max}$ *получим, что* $q \to 1$ *и, следовательно, скорость сходимости алгоритма к решению может оказаться крайне низкой. При* $\lambda_{\min} \approx \lambda_{\max}$ *алгоритм может сходиться за конечное число шагов.*

Довольно часто в методах наискорейшего спуска вычисление градиента целевой функции заменяют его приближенным вычислением с помощью разностных отношений /Гилл с176/ :

$$\frac{\partial f(x)}{\partial x_i} \approx \frac{f(x + h \cdot e_i) - f(x)}{h}$$

где h достаточно малый шаг, а e_i - единичный вектор с единицей на i - ом месте. Однако, такой подход увеличивает количество вычислительной работы для каждой итерации в $(n+1)$ раз.

Пусть $\max_i | f''(\xi) | \le M$, где $\xi \in [x, x+h]$. Тогда, можно показать, что оценка ошибки вычисления по формуле разностных отношений достигает минимума при $h_0 = 2\sqrt{eps \cdot M}$, где eps ошибка вычисления (задания) функций f. Поэтому выбор очень маленького шага не приведет к росту точности. Выбор же оптимального шага h_0 будет давать погрешность порядка $O(eps)$. Такие алгоритмы, где используются разностные оценки градиента, называют субградиентными методами.

Критерии плохой обусловленности (овражности) задач оптимизации

Траектория $x(t)$ градиентного наискорейшего спуска нахождения минимума функции $f(x)$ задается в пределе векторным дифференциальным уравнением: $\frac{dx(t)}{dt} = -\nabla f(x(t))$. При решении задач квадратичной оптимизации с плохо обусловленной матрицей гессиана в движении точки по траектории можно выделить два участка с различным поведением решения. Первый, сравнительно непродолжительный, характеризуется большими значениями вектора производных $\nabla f(x)$ и означает спуск на дно оврага линий уровня функции $f(x)$. На дне оврага норма вектора градиента становится относительно малой и изменение траектории $x(t)$ становится незначительным. Поэтому был предложен следующий критерий плохой обусловленности функции оптимизации.

Определение */Черноруцкий с40/. Задача* $f(x) \to \min, x \in \Re^n$ *является плохо обусловленной или «овражной», если отвечающая решению методом наискорейшего спуска система дифференциальных уравнений «жесткая».*

В соответствии с данным определением, для определения «овражности» задачи оптимизации необходимо исследовать на плохую обусловленность матрицу уравнений наискорейшего спуска в зоне поиска, как это уже было указано выше в разделе, посвященном исследованию «жестких» дифференциальных уравнений.

Другой подход связан с исследованием плохой обусловленности матрицы гессиана. Однако следует отметить, что встречаются задачи с «плохими» матрицами якобиана и гессиана, но которые не являются «овражными». Поэтому, часто сложность профиля оптимизируемой функции исследуют предварительно с помощью метода градиентного спуска с постоянным шагом:

$$x^{(k+1)} = x^{(k)} + \lambda \cdot \nabla f(x^{(k)}).$$

Принадлежность $f(x)$ к классу овражных функций проявляется в необходимости применения относительно малых значений λ. Попытки увеличения λ вызывают потерю свойства релаксации (то есть монотонного убывания) последовательности значений функции $f(x^{(k)})$. При этом значения функции начинают, или резко возрастать, или колебаться с достаточно большой амплитудой. Если выбрать граничное значение λ такое, что процесс снова станет релаксационным, то по найденной величине шагового множителя можно судить о степени «овражности» в окрестности исследуемой точки $x^{(k)}$. В частности, в качестве такой меры, можно принять следующее значение $\| f(x^{(k+1)}) \| / \| f(x^{(k)}) \|$. Шкала для такой меры обычно формируется на основе решения известных модельных задач.

Оценки скорости сходимости итерационных процессов поиска.

Линейной скоростью сходимости итерационного процесса поиска называют сходимость, при которой некоторая мера $\varepsilon^{(k)}$ близости к решению убывает по закону геометрической прогрессии. То есть значения $\varepsilon^{(k+1)}$ и $\varepsilon^{(k)}$ являются бесконечными малыми одного порядка:

$$\frac{\| \varepsilon^{(k+1)} \|}{\| \varepsilon^{(k)} \|} = O(1), \ k \to \infty .$$

Сверхлинейной сходимостью итерационного процесса поиска называют сходимость более быструю, чем у любой геометрической прогрессии. При сверхлинейной сходимости значение $\varepsilon^{(k+1)}$ имеет более высокий порядок малости по отношению к значению $\varepsilon^{(k)}$ при $k \to \infty$. То есть:

$$\lim_{k\to\infty} \frac{\| \varepsilon^{(k+1)} \|}{\| \varepsilon^{(k)} \|} = 0 .$$

Таким образом, если последовательность $\{x^{(k)}\}$ сходится к значению x^* при $k \to \infty$ сверхлинейно, то выполняется неравенство:

$$\| x^{(k+m)} - x^* \| \le C \cdot \| x^{(k)} - x^* \|, \ \forall m > 0 \ ,$$

где $C > 0$ – некоторая константа.

Квадратичной сходимостью итерационного процесса поиска называется сходимость, при которой некоторая мера погрешности на следующем шаге $\varepsilon^{(k+1)}$ является величиной не менее, чем второго порядка малости по отношению к значению $\varepsilon^{(k)}$ ошибки на предыдущем шаге. То есть существует такая постоянная $C > 0$, что:

$\| \varepsilon^{(k+1)} \| \le C \cdot \| \varepsilon^{(k)} \|^2$ при некотором значении $k > k_0$. Таким образом, если последовательность $\{x^{(k)}\}$ сходится к x^* квадратично при $k \to \infty$, то $\exists \varepsilon > 0, \ C > 0$, что $\forall x^{(0)}$ из шара $U_\varepsilon(\| x^* - x^{(0)} \| < \varepsilon)$ справедливо неравенство: $\| x^{(k+1)} - x^* \| \le C \cdot \| x^{(k)} - x^* \|^2$.

Методы сопряженных градиентов.

В классе методов сопряженных направлений (градиентов), в отличие от метода наискорейшего спуска, используется информация о направлениях поиска на предыдущих этапах оптимизации.

Обычно направление поиска $h^{(k)}$ на k - ом шаге строится как линейная комбинация значения градиента $-\nabla f(x^{(k)})$ и направлений спуска $h^{(k-1)}, h^{(k-2)}, ..., h^{(k-s)}$, на предыдущих s шагах. Весовые коэффициенты выбираются таким образом, чтобы сделать эти направления сопряженными.

Определение. *Система линейно-независимых векторов $h^{(0)}, h^{(1)}, ..., h^{(n-1)}$, для симметричной матрицы $Q \in \Re^{n\times n}$ называется Q - сопряженной (или просто сопряженной), если выполняются следующие соотношения:*

$\forall i, j = 0,1,...,(n-1); i \ne j; \ (h^{(i)}, Q \cdot h^{(j)}) = 0 .$

Здесь $(h^{(i)}, Q \cdot h^{(j)}) = (h^{(i)})^T \cdot Q \cdot h^{(j)}$.

Лемма. *Если все вектора* $h^{(0)}, h^{(1)}, ..., h^{(n-1)}$ *отличны от нуля,* Q *– симметричная и положительно определенная матрица, то из условия:*

$(h^{(i)}, Q \cdot h^{(j)}) = 0; \ \forall i, j = 0,1,...,(n-1); i \neq j$,

следует их нелинейная независимость.

Суть метода сопряженных направлений заключается в том, что требуется найти n направлений $h^{(0)}, h^{(1)}, ..., h^{(n-1)}$ таких, что последовательность n одномерных минимизаций вдоль этих направлений будут приводить к отысканию минимума функции $f(x) = \frac{1}{2} \cdot x^T \cdot Q \cdot x + c^T \cdot x$, то есть:

$$f(x^{(k)}) = \min_{x \in \Re^n}(f(x)), \ \forall x^{(0)} \in \Re^n ,$$

где $x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot h^{(k)}; \ f(x^{(k)} + \alpha^{(k)} \cdot h^{(k)}) = \min_{\alpha \in \Re}(f(x^{(k)} + \alpha \cdot h^{(k)}), \ k = 0.1.2.....$

Оказывается, что указанным свойством обладает система взаимно сопряженных, относительно матрицы Q, направлений. /Сухарев с226/

Рассмотрим поисковую процедуру по следующим правилам:

$$h^{(0)} = -\nabla f(x^{(0)}); \ h^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k-1)} \cdot h^{(k-1)}; \ k = 0,1,... .$$

Из условия сопряженности векторов $h^{(k)}$ и $h^{(k-1)}$ получим:

$$0 = (h^{(k)}, Q \cdot h^{(k-1)}) = (-\nabla f(x^{(k)}), Q \cdot h^{(k-1)}) + \beta^{(k-1)} \cdot (h^{(k-1)}, Q \cdot h^{(k-1)}) .$$

Отсюда справедливо соотношение:

$$\beta^{(k-1)} = \frac{(\nabla f(x^{(k)}, Q \cdot h^{(k-1)})}{(h^{(k-1)}, Q \cdot h^{(k-1)})} .$$

Легко доказать, что вектора $\nabla f(x^{(k)})$ и $\nabla f(x^{(k-1)})$, как и методе наискорейшего спуска, ортогональны, то есть $(\nabla f(x^{(k)}), \nabla f(x^{(k-1)})) = 0; \ k = 1,2,.....$

Лемма */Сухарев с230/* . *Пусть* $x^{(0)} \in \Re^n$, *точки* $x^{(1)}, x^{(2)}, ..., x^{(n-1)}$ *и векторы* $h^{(0)}, h^{(1)}, ..., h^{(n-1)}$ *получены в соответствии с поисковой процедурой по формулам*

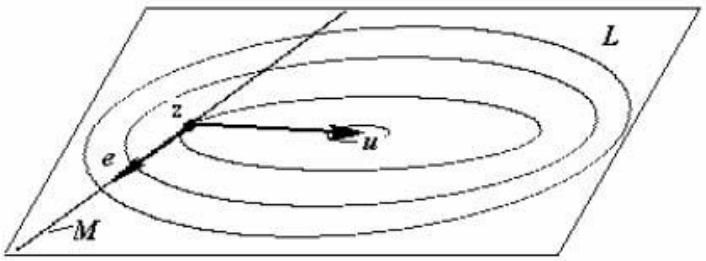
$$h^{(0)} = -\nabla f(x^{(0)}); \ h^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k-1)} \cdot h^{(k-1)}; \ \beta^{(k-1)} = \frac{(\nabla f(x^{(k)}, Q \cdot h^{(k-1)})}{(h^{(k-1)}, Q \cdot h^{(k-1)})}, k = 0,1,...,$$

и $\nabla f(x^{(k)}) \neq 0, \ k = 0,1,...,(n-1)$. *Тогда векторы* $h^{(0)}, h^{(1)}, ..., h^{(n-1)}$ *взаимно сопряжены, а градиенты* $\nabla f(x^{(0)}), \nabla f(x^{(1)}), ..., \nabla f(x^{(n-1)})$ *взаимно ортогональны.*

Теорема *Метод сопряженных направлений (градиентов) в соответствии с процедурой поиска по формулам*,

$$h^{(0)} = -\nabla f(x^{(0)}); \ h^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k-1)} \cdot h^{(k-1)}; \ \beta^{(k-1)} = \frac{(\nabla f(x^{(k)}, Q \cdot h^{(k-1)})}{(h^{(k-1)}, Q \cdot h^{(k-1)})}, k = 0,1,...$$

отыскания минимума квадратичной целевой функции $f(x)=\frac{1}{2}\cdot x^T\cdot Q\cdot X+c^T\cdot x$ *сходится не более, чем за* n *- шагов.*
(без доказательства).



Полученный метод находит минимум любой квадратичной целевой функции с положительно определенной матрицей Гессе не более, чем за n шагов. Однако, для вычисления точек $x^{(k)}$ должен быть использован «точный» одномерный поиск.

Запишем следующее соотношение:

$$f(x^{(k)}+\alpha\cdot h^{(k)})=\frac{1}{2}\cdot(Q(x^{(k)}+\alpha\cdot h^{(k)}),x^{(k)}+\alpha\cdot h^{(k)})+(c,x^{(k)}+\alpha\cdot h^{(k)})=$$
$$=\frac{1}{2}\cdot(Q\cdot h^{(k)},h^{(k)})\cdot\alpha^2+(Q\cdot x^{(k)}+c,h^{(k)})\cdot\alpha+(\frac{1}{2}\cdot Q\cdot x^{(k)},x^{(k)}).$$

Минимум значения $f(x^{(k)}+\alpha\cdot h^{(k)})$ достигается, очевидно, в точке:

$$\alpha^{(k)}=-\frac{(Q\cdot x^{(k)}+c,h^{(k)})}{(Q\cdot h^{(k)},h^{(k)})}\geq 0.$$

Очевидно, что это значение параметра будет точное решение одномерной оптимизации задачи $f(x)=\frac{1}{2}\cdot x^T\cdot Q\cdot x+c^T\cdot x\to\min$.

Построим теперь процедуру метода сопряженных градиентов для минимизации неквадратичной функции. Для этого, необходимо, преобразовать формулу $f(x^{(k)}+\alpha\cdot h^{(k)})$ так, чтобы в ней не фигурировала матрица Q.

Для целевой функции $f(x)=\frac{1}{2}\cdot x^T\cdot Q\cdot x+c^T\cdot x$, как нетрудно заметить, справедливо соотношение: $\nabla f(x)=Q\cdot x+c$. Тогда, в соответствии со схемой решения, можно записать следующее соотношение:

$$\nabla f(x^{(k)})=\nabla f(x^{(k-1)})+\alpha^{(k-1)}\cdot Q\cdot h^{(k)}.$$

На основание вышеизложенного получим:

$$\beta^{(k-1)}=\frac{(\nabla f(x^{(k)},Q\cdot h^{(k-1)})}{(h^{(k-1)},Q\cdot h^{(k-1)})}=\frac{\|\nabla f(x^{(k)})\|^2}{\|\nabla f(x^{(k-1)}\|^2}.$$

Полученный метод также носит название метода сопряженных градиентов и может применяться для минимизации неквадратичных функций. Однако, при этом, метод перестает быть конечным, вырабатываемые им направления $h^{(0)},h^{(1)},...,h^{(n-1)}$ не являются, вообще говоря,

взаимно сопряженными относительно какой-либо матрицы, а определение величин $\alpha^{(k)}$ в задачах одномерной минимизации по направлениям $h^{(i)}, i = 0,1,2,....$ приходится решать численно.

Метод сопряженных градиентов Флетчера-Ривса.

Шаг 0. Задают следующие величины: параметр останова $\varepsilon > 0,\ k = 0$, параметры одномерного поиска и начальную точку $x^{(0)}$.

Шаг 1. Вычисляют $f^{(0)} = f(x^{(0)}),\ \nabla f^{(0)} = \nabla f(x^{(0)}),\ h^{(0)} = -\nabla f(x^{(0)}),\ k1 = 0$.

Шаг 2. Если $(\nabla f^{(k)}, h^{(k)}) \geq 0$, то направление $h^{(k)}$ не является направлением локального убывания функции, поэтому заменяем $h^{(k)} = -\nabla f^{(k)}$. В противном случае направление $h^{(k)}$ сохраняем. Значения $k1,\ x^{(0)}$ оставляем без изменения. Переходим к шагу 3.

Шаг 3. Вычисляем величину коэффициента масштаба шага $\alpha^{(k)}$ одномерного поиска. Определяем следующие значения: $x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot h^{(k)},\ f^{(k+1)} = f(x^{(k+1)}),\ \nabla f^{(k+1)} = \nabla f(x^{(k+1)})$.

Полагаем $k = k+1,\ k1 = k1+1$. Переходим к шагу 4.

Шаг 4. Проверяем критерий останова. Если выполняется неравенство $\| \nabla f(x^{(k)}) \| < \varepsilon$, то поиск прекращаем и значение $x^{(k)}$ принимается, как оценка решения. Если $\| \nabla f(x^{(k)}) \| > \varepsilon$, то переходим к шагу 5.

Шаг 5. Если $k1 = n$, то полагаем $x^{(0)} = x^{(k)}$ и переходим к шагу 1. Если $k1 < n$, то переходим к шагу 6.

Шаг 6. Вычисляем $\beta^{(k)} = \dfrac{\| \nabla f(x^{(k+1)}) \|^2}{\| \nabla f(x^{(k)}) \|^2}$ и $h^{(k+1)} = -\nabla f(x^{(k+1)}) + \beta^{(k)} \cdot h^{(k)}$. Переходим к шагу 2.

Можно показать, что при применении методов сопряженных градиентов (в том числе и метода Флетчера-Ривса) для минимизации ограниченной снизу функции f, градиент которой удовлетворяет условию Липшица, для любой начальной точки $x^{(0)} \in \Re^n$, получаем $\| \nabla f(x^{(k)}) \| \underset{k \to \infty}{\to} 0$. Для сильно - выпуклой гладкой функции, удовлетворяющей некоторым дополнительным ограничениям, последовательность $\{x^{(k)}\}$ сходится к точке минимума x^* со сверхлинейной скоростью: $\| x^{(k+n)} - x^* \| \leq C \cdot \| x^{(k)} - x^* \|^2,\ k = \{0, n, 2n, 3n, ...\}$. То есть метод обладает высокой скоростью сходимости. При этом его трудоемкость лишь незначительно превосходит метод наискорейшего спуска.

Методы второго порядка безусловной оптимизации.

В методах второго порядка при поиске минимума используют информацию, как о функции, так и ее производных до второго порядка включительно.

Метод Ньютона.

Пусть функция f выпукла и дважды дифференцируема на $\Re^n$ и матрица $\nabla^2 f(x)$ является невырожденой. Тогда справедливо соотношение:

$$f(x)-f(x^{(k)})=(\nabla f(x^{(k)},(x-x^{(k)}))+\frac{1}{2}\cdot(\nabla^2 f(x^{(k)})\cdot(x-x^{(k)}),(x-x^{(k)}))+o(\|x-x^{(k)}\|^2)\ .$$

Для определения следующей точки $x^{(k+1)}$ решается задача минимизации функции $\varphi^{(k)}(x)$ вида:

$$\varphi^{(k)}(x)=(\nabla f(x^{(k)},(x-x^{(k)}))+\frac{1}{2}\cdot(\nabla^2 f(x^{(k)})\cdot(x-x^{(k)}),(x-x^{(k)}))\ .$$

Ясно, что $\nabla^2\varphi^{(k)}(x)=\nabla^2 f(x^{(k)})$.

Так как функция $f(x)$ выпукла, тогда и только тогда, когда матрица ее вторых производных неотрицательна, то функция $\varphi^{(k)}(x)$ - выпукла. Тогда необходимое и достаточное условие минимума функции $\varphi^{(k)}(x)$ в точке x имеет вид:

$$\nabla\varphi^{(k)}(x)=\nabla f(x^{(k)})+\nabla^2 f(x^{(k)})\cdot(x-x^{(k)})=0\,.$$

Решая полученную систему линейных уравнений и принимая найденную точку минимума за точку $x^{(k+1)}$, получим: $x^{(k+1)}=x^{(k)}+h^{(k)}$,
где $h^{(k)}=-(\nabla^2 f(x^{(k)}))^{-1}\cdot\nabla f(x^{(k)})$.

Теорема (О сходимости метода Ньютона) */Сухарев с220/.*

Пусть функция f дважды дифференцируема, сильно выпукла с константой $\theta>0$ на $\Re^n$ и удовлетворяет условию:

$$\|\nabla^2 f(x)-\nabla^2 f(x^{(0)})\|\le C\cdot\|x-x^{(0)}\|,$$

где $C>0$, а начальная точка $x^{(0)}$ такова, что $\|\nabla f(x^{(0)})\|\le\frac{8\cdot\theta^2\cdot q}{C}$, $q\in(0,1)$. Тогда последовательность $\{x^{(k)}\}$ сходится к точке минимума x^* с квадратичной скоростью:

$$\|x^{(k)}-x^*\|\le\frac{4\cdot\theta\cdot q^{2^k}}{C}.$$

Несложно показать, что последовательность $\{x^{(k)}\}$ сходится к точке минимума x^* для целевой функции вида $f(x)=\frac{1}{2}\cdot x^T\cdot Q\cdot x+c^T\cdot x$ за один шаг. Однако для функции, отличной от квадратичной формы, возникают определенные сложности с выбором необходимого начального приближения, что является одним из недостатков метода Ньютона. Кроме того, метод обладает высокой трудоемкостью, обусловленной необходимостью вычисления и обращения на каждом шаге матрицы вторых производных.

Квазиньютоновские методы (методы переменной метрики).

Идея, положенная в основу квазиньютоновских методов, состоит в том, чтобы по результатам измерения градиентов функции f в точках $x^{(k)}$ траектории поиска построить некоторую матрицу $H^{(k)}$, которая является некоторой оценкой «кривизны» поверхности $f(x)$. То есть, по

сути, найти оценку $(\nabla^2 f)^{-1}$. Так как оцениваемая матрица симметрична, то на матрицу $H^{(k)}$ накладывают также, обычно, дополнительные условия симметрии $H^{(k)} = (H^{(k)})^T$.

Рассмотрим итерационный процесс поиска следующего вида:

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \cdot h^{(k)},$$

где $h^{(k)} = -H^{(k)} \cdot \nabla f(x^{(k)})$.

Так как справедливо соотношение:

$$\nabla f(x^{(k+1)}) - \nabla f(x^{(k)}) = \nabla^2 f(x^{k+1}) \cdot (x^{(k+1)} - x^{(k)}) + o(\| x^{(k+1)} - x^{(k)} \|),$$

то приближенно можно записать:

$$(\nabla^2 f(x^{(k)}))^{-1} \cdot (\nabla f(x^{(k+1)} - \nabla f(x^{(k)})) \approx (x^{(k+1)} - x^{(k)}) \ .$$

Это соотношение носит название квазиньютоновского условия. Приближение к матрице $(\nabla^2 f(x^{(k)}))^{-1}$ пересчитывают, обычно, от шага к шагу по формуле:

$$H^{(k+1)} = H^{(k)} + \Delta H^{(k)}.$$

В зависимости от способа пересчета различают и различные квазиньютоновские методы.

Введем обозначения:

$$z^{(k)} = \nabla f^{(k+1)} - \nabla f^{(k)}, \ h^{(k)} = x^{(k+1)} - x^{(k)}.$$

Поправки $\Delta H^{(k)}$ строятся обычно в виде матриц. Наиболее известными являются схемы Бройдена (B-формула) и Дэвидона-Флетчера-Пауэлла (DFP-формула). /Городецкий Прак с42/.

Схема Бройдена:

$$H^{(k+1)} = H^{(k)} + \frac{(h^{(k)} - H^{(k)} \cdot z^{(k)}) \cdot (h^{(k)} - H^{(k)} \cdot z^{(k)})^T}{(h^{(k)} - H^{(k)} \cdot z^{(k)})^T \cdot z^{(k)}} \ .$$

Последовательность шагов алгоритма Бройдена.

Шаг 1. Задается начальное приближение $x^{(0)}$ и некоторая положительно определенная матрица $H^{(0)}$ (например, единичная диагональная матрица).

Шаг 2. Вычисляется $x^{(k+1)} = x^{(k)} - \alpha^{(k)} \cdot H^{(k)} \cdot \nabla f(x^{(k)})$, где

$$\alpha^{(k)} = \arg\min_{\alpha}(f(x^{(k)}) - \alpha \cdot H^{(k)} \cdot \nabla f(x^{(k)}) .$$

Шаг 3. Находится очередное приближение матрицы

$$H^{(k+1)} = H^{(k)} + \frac{(h^{(k)} - H^{(k)} \cdot z^{(k)}) \cdot (h^{(k)} - H^{(k)} \cdot z^{(k)})^T}{(h^{(k)} - H^{(k)} \cdot z^{(k)})^T \cdot z^{(k)}}$$ по B-формуле.

Шаг 4. Проверяется критерий останова $\| \nabla f(x^{(k)}) \| < \varepsilon$. Если он не выполняется, то осуществляется переход к шагу 2. Если выполняется неравенство $\| \nabla f(x^{(k)}) \| < \varepsilon$, то поиск прекращается и $x^{(k)}$ выдается, как оценка решения.

Если целевая функция является квадратичной, то для определения минимума оказывается достаточным сделать n шагов. В случае минимизации неквадратичной функции возможны нежелательные явления, связанные с возможностью нарушения положительной определенности матрицы $H^{(k)}$.

Схема Дэвидона-Флетчера-Пауэлла:

$$H^{(k+1)} = H^{(k)} - \frac{(H^{(k)} \cdot z^{(k)} \cdot (z^{(k)})^T \cdot H^{(k)}}{(z^{(k)})^T \cdot H^{(k)} \cdot z^{(k)}} + \frac{h^{(k)} \cdot (h^{(k)})^T}{(h^{(k)})^T z^{(k)}}.$$

В этой схеме, как и в схеме Бройдена, начальное значение матрицы $H^{(0)}$ выбирается, как единичная диагональная матрица. Хотя в ряде случаев является более предпочтительным задание начального значения в виде матрицы конечно-разностных приближений вторых частных производных функции f в начальной точке $x^{(0)}$. Этот алгоритм является одним из наиболее эффективных алгоритмов переменной метрики. В ходе процесса оптимизации этим методом происходит постепенный переход от градиентного направления спуска к ньютоновскому. При этом, при квадратичной целевой функции направления поиска становятся сопряженными, что обеспечивает его высокую эффективность. Метод также устойчив к ошибкам определения масштабного коэффициента $\alpha^{(k)}$ шага поиска при одномерной оптимизации.