# Лабораторная работа №3
# «Многомерная безусловная оптимизация» (7 часов)

## Лабораторная работа №3.1 «Методы прямого поиска» (4 часа)
## Лабораторная работа №3.2 «Методы многомерного поиска с использованием производных» (3 часа)

*Цель работы:* знакомство с методами многомерной безусловной оптимизации и сравнение эффективности применения этих методов для конкретных целевых функций.

# 1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

## 1.1 Общая постановка задачи

В общем виде задача многомерной безусловной оптимизации формулируется как: $\min f(x)$, при $x \in X$, где $x = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\}$ – точка в *n*-мерном пространстве $X = IR^n$.

Таким образом, целевая функция $f(x) = f(x^{(1)}, x^{(2)}, ..., x^{(n)})$ является функцией *n* аргументов. По аналогии с предыдущим анализом, будем решать задачу минимизации. Основная идея численных методов поиска минимума состоит в определении последовательности точек $\{x_k\}$, удовлетворяющих условию $f(x_0) > f(x_1) > ... > f(x_n)$. Методы построения таких последовательностей называются методами спуска. В этих методах точки последовательности $\{x_k\}$ вычисляются по формуле $x_{k+1} = x_k + \alpha_k p_k$, где $p_k$ – направление спуска, $\alpha_k$ – длина шага в этом направлении, $k = 0,1,2,...,n$. Различные методы спуска отличаются друг от друга способами выбора направления спуска $p_k$ и длины шага $\alpha_k$ вдоль этого направления.

Алгоритмы безусловной минимизации принято делить на классы, в зависимости от максимального порядка производных минимизируемой функции, вычисление которых предполагается. Так методы, использующие только значения целевой функции, относят к методам нулевого порядка или методами прямого поиска. Если в процессе поиска требуется вычисление первых производных целевой функции, то эти методы являются методами первого порядка. При использовании

для поиска минимума (максимума) вторых производных получаем методы второго порядка и т. д.

Основное достоинство методов нулевого порядка состоит в том, что они не требуют непрерывности целевой функции и существования производных. Однако прибегая к этим методам, надо быть уверенным в том, что метод другого типа применить нельзя, иначе можно дорого заплатить потерями машинного времени. Кроме того, общим их недостатком является плохая сходимость.

Простейшим методом поиска является метод покоординатного спуска. Рассмотрим функцию двух переменных (рисунок 1.1), минимум которой расположен точке $x = (x^{(1)*}, x^{(2)*})$. Из точки $x_0$ производим поиск минимума вдоль направления оси $x^{(1)}$ и, таким образом, находим точку $x_1$, в которой касательная к линии постоянного уровня параллельна оси. Затем, производя поиск из точки $x_1$ в направлении оси $x^{(2)}$, получаем точку $x_2$, производя поиск параллельно оси $x^{(1)}$, получаем точку $x_3$, затем $x_4$ и т. д. Таким образом, через некоторое число итераций находим точку минимума. Данная идея может быть применена и для функций *n* переменных.

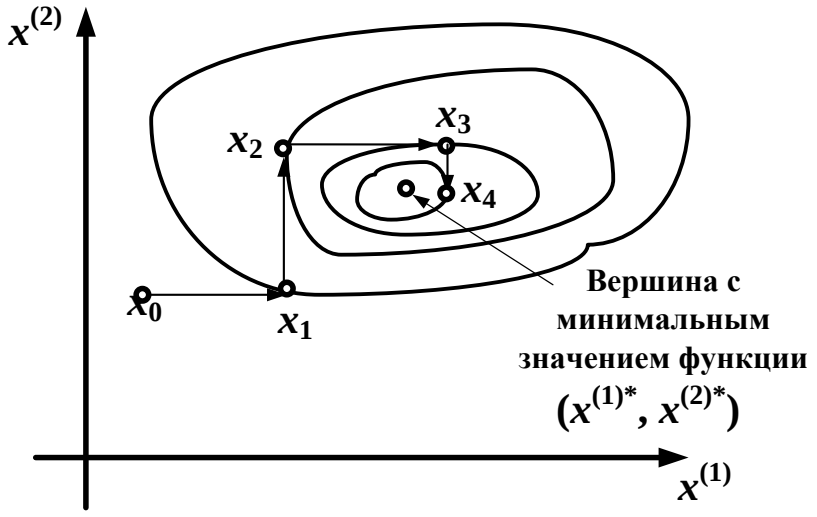


Рисунок 1.1 – Иллюстрация метода покоординатного спуска

Однако на практике данный метод оказался слишком медленным. Поэтому были разработаны более сложные методы, использующие больше информации на основании уже полученных значений функции. Среди методов прямого поиска рассмотрим поиск по симплексу (по многограннику) с модификацией Нелдера–Мида и метод Хука–Дживса.

### 1.1.2 Метод поиска по симплексу

Под симплексом понимается *n*-мерный выпуклый многогранник *n*-мерного пространства, имеющий $n+1$ вершину. Для $n=2$ это треугольник, а при $n=3$ это тетраэдр.

Идея метода состоит в сравнении значений функции в $n+1$ вершинах симплекса и перемещении симплекса в направлении лучшей точки. В рассматриваемом методе симплекс перемещается с помощью операций отражения. При анализе будем использовать следующие обозначения: $x_0(k)$, $x_1(k)$, ..., $x_n(k)$ – вершины симплекса (вектора в *n*-мерном пространстве), *к* – номер итерации.

Наименьшее число точек содержит регулярный симплекс (равноотстоящие вершины). В случае двух переменных (плоскость) это равносторонний треугольник, в трехмерном пространстве – тетраэдр. Метод наиболее эффективен при $n \le 6$.

Алгоритм начинается с построения регулярного симплекса в пространстве *n* переменных. При этом определяется вершина, которой соответствует наибольшее значение функции (рисунок 1.2). Эта вершина проецируется через центр тяжести остальных вершин в новую точку, которая будет вершиной нового симплекса. Итерации выполняются до тех пор, пока не будет накрыта точка минимума, либо не начнется циклическое движение (движение по двум или более симплексам).

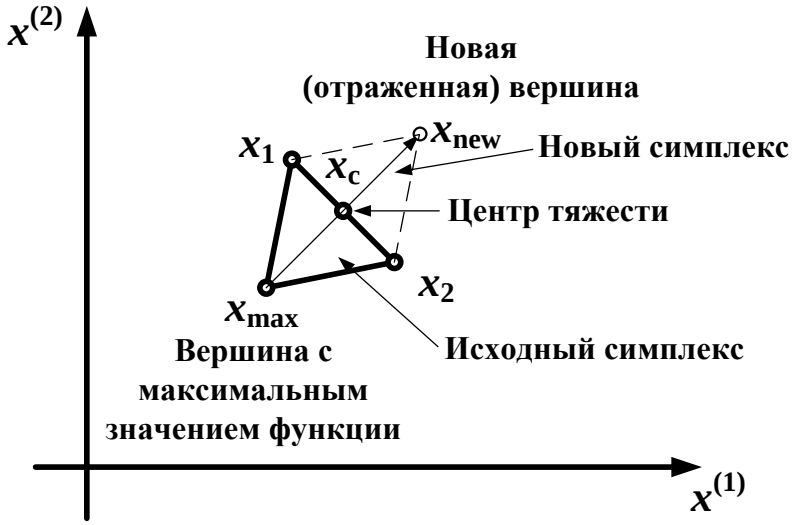


Рисунок 1.2 – Метод поиска по симплексу

*Схема алгоритма*

*Шаг*_1. Построение начального симплекса.

Задается начальная точка $x_0(0)$ и длина ребра симплекса $a$ (если $\alpha = 1$, то ребра единичной длины). Формируются остальные вершины симплекса: $x_i(k) = x_0(k) + \alpha e_i$, где $e_i$ – единичные векторы, $k = 0,1,2,...,n$.

*Шаг*_2. Определение направления улучшения решения.

Вычисляются значения целевой функции в каждой точке симплекса (на $k$-й итерации), среди которых находится максимальное значение. Пусть $f(x_i(k)) \le f(x_{\max}(k))$, для $i = 0,1,2,...,n$. Определяется центр тяжести всех точек, исключая точку $x_{\max}(k)$:

$$c_k = \frac{\sum_i x_i(k)}{n}. \qquad (1.1)$$

Тогда направление улучшения решения определяется вектором $c_k - x_{\max}(k)$.

*Шаг*_3. Построение отраженной точки.

Точки прямой, проходящей через $c_k$ и $x_{\max}(k)$, задаются формулой: $x = x_{\max}(k) + \alpha(c_k - x_{\max}(k))$.

При $\alpha = 0$ получим исходную точку $x_{\max}(k)$, при $\alpha = 1$ получаем точку $c_k$, при $\alpha = 2$ получаем новую вершину. Таким образом:

$x_{new}(k) = 2c_k - x_{\max}(k)$.

*Шаг*_4. Построение нового симплекса.

Вычисляем значение функции в новой точке. При этом возможны два случая: $f(x_{new}(k)) < f(x_{\max}(k))$ или $f(x_{new}(k)) \ge f(x_{\max}(k))$. В первом случае вершину $x_{\max}(k)$ заменяем на $x_{new}(k)$ и переходим на следующую итерацию.

Во втором случае производится пропорциональное уменьшение симплекса. Для этого находят вершину с минимальным значением целевой функции, делают ее базовой и строят новый симплекс с меньшими (например, в 2 раза) размерами.

*Шаг*_5. Проверка сходимости.

Если условие:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(f(x_i(k+1)) - f(x_0(k+1)))^2} \le \varepsilon$$

выполняется, то поиск заканчивается и полагается $\tilde{x} = x_0(k+1),\ \tilde{y} = f(x_0(k+1))$.

В противном случае $k = k+1$ и переход к шагу 2.

### 1.1.3 Метод Нелдера-Мида

Метод Нелдера-Мида или метод деформируемого симплекса обладает большей общностью и позволяет учитывать локальные свойства поверхности целевой функции. Симплексы вытягиваются в направлении наклона поверхности, их оси поворачиваются при встрече с оврагом на поверхности целевой функции, вблизи минимума они сжимаются. В данном методе симплекс перемещается с помощью трех основных операций над симплексом: нормальное отражение (рис.1.3,а), растяжение (рис.1.3,б) и сжатие (рис.1.3,в,г).

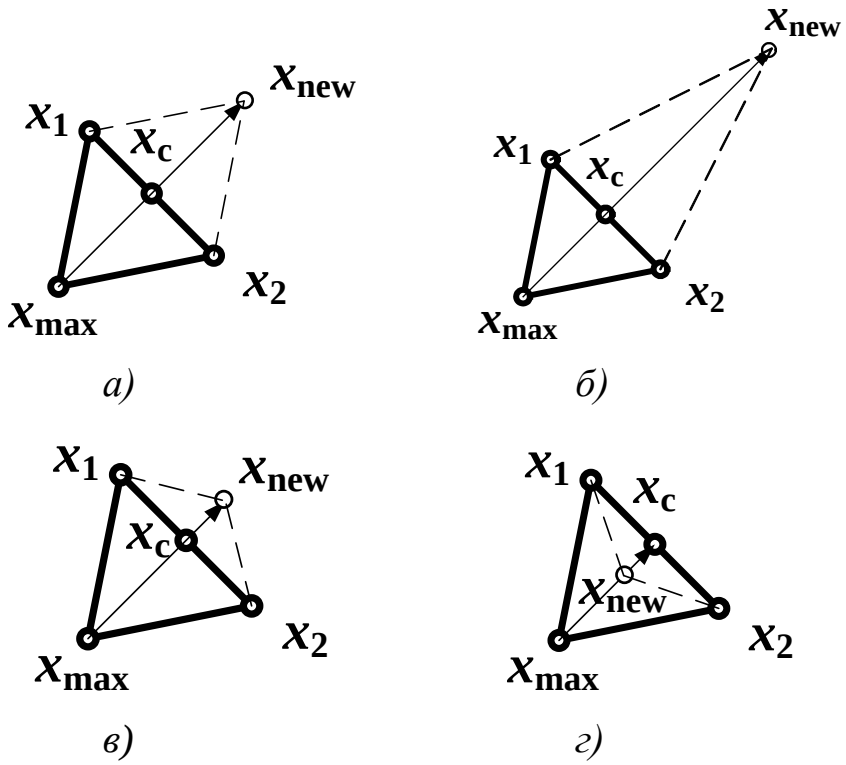


Рисунок 1.3 – Основные операции над симплексом

*Схема алгоритма.*

*Шаг*_1. Построение начального симплекса.

Задается начальная точка $x_0(0)$ и длина ребра симплекса $\alpha = 1$. Формируются остальные вершины симплекса: $x_i^{\,0} = x_0^{\,(0)} + \alpha e_i$, где $e_i$ – единичные векторы, $i = 1,2,...,n$.

*Шаг*_2. Определение направления улучшения решения.

Вычисляются значения целевой функции в каждой точке симплекса (на *k*-й итерации), среди которых находятся максимальное и минимальное значение. Пусть $f(x_{\min}(k)) \le f(x_i(k)) \le f(x_{\max}(k))$. Определяется центр тяжести всех точек (1.1), исключая точку $x_{\max}(k)$. Тогда направление улучшения решения определяется вектором $c_k - x_{\max}(k)$.

*Шаг*_3. Построение отраженной точки.

Отражаем вершину $x_{\max}(k)$ с максимальным значением целевой функции через центр тяжести остальных точек и находим новую точку $x_{new}(k)$, координаты которой $x_{new}(k) = c_k + \alpha(c_k - x_{\max}(k))$, где $\alpha$ – коэффициент отражения.

*Шаг*_4. Построение нового симплекса.

Вычисляем значение функции в новой точке. При этом возможны три случая:

– $f(x_{new}(k)) < f(x_{\min}(k))$,

– $f(x_{new}(k)) > f(x_{\max}(k))$,

– $f(x_{\min}(k)) \le f(x_{new}(k)) \le f(x_{\max}(k))$.

В первом случае отражённая точка является точкой с наилучшим значением целевой функции. Поэтому направление отражение является перспективным и можно попытаться растянуть симплекс в этом направлении. Для этого строиться точка:

$x'_{new}(k) = c_k + \beta(x'_{new}(k) - c_k)$,

где $\beta > 1$ – коэффициент расширения.

Если $f(x'_{new}(k)) < f(x_{new}(k))$, то вершину $x_{\max}(k)$ заменяем на $x'_{new}(k)$, в противном случае $x_{\max}(k)$ заменяем на $x_{new}(k)$, и переходим на следующую итерацию.

Во втором случае отражённая точка является точкой с наихудшим значением целевой функции. Поэтому в этом случае производится сжатие симплекса. Для этого строится точка $x'_{new}(k)$, координаты которой определяются следующим образом:

$x'_{new}(k) = c_k + \gamma(x_{\max}(k) - c_k)$, если $f(x_{\max}(k)) \le f(x_{new}(k))$,

или

$x'_{new}(k) = c_k + \gamma(x_{new}(k) - c_k)$, если $f(x_{\max}(k)) > f(x_{new}(k))$,

где $0 < \gamma < 1$ – коэффициент сжатия.

Если $f(x'_{new}(k)) < \{f(x_{\max}(k)), f(x_{new}(k))\}$ вершину $x_{\max}(k)$ заменяем на $x'_{new}(k)$, в противном случае вершинам $x_i(k+1)$ $(i = 0,1,2,...,n)$ присваиваются средние значения $\tilde{x} = \frac{x_i(k) + x_{\min}(k)}{2}$, и переходят на следующую итерацию.

В третьем случае $x_{\max}(k)$ заменяем на $x_{new}(k)$, и переходим на следующую итерацию.

*Шаг*_5. Проверка сходимости.

Если условие $\sqrt{\frac{1}{n}\sum_{i=1}^{n}(f(x_i(k+1) - f(x_0(k+1)))^2} \le \varepsilon$ выполняется, то поиск минимума заканчивается и полагается $\tilde{x} = x_0(k+1), \tilde{y} = f(x_0(k+1))$. В противном случае $k = k+1$ и происходит переход к шагу 2.

В качестве параметров алгоритма (коэффициент отражения, растяжения и сжатия) целесообразно выбирать следующие значения: $\alpha = 1,\ \beta = 2,\ \gamma = 0{,}5$.

### 1.1.4 Метод Хука-Дживса

Метод Хука-Дживса (метод конфигураций) был разработан в 1961 году, но до сих пор является одним из самых эффективных методов многомерной безусловной оптимизации. Основная идея заключается в использовании исследующего поиска вокруг базисной точки, за которым в случае успеха следует поиск по образцу. Таким образом, алгоритм включает в себя два основных этапа поиска:

–исследуется окрестность выбранной базисной точки, в результате находится приемлемое направление спуска;

–в выбранном направлении находится точка с наименьшим значением целевой функции, то есть новая базисная точка.

Процедура продолжается пока в окрестностях базисных точек удается находить приемлемые направления спуска.

*Схема алгоритма*

*Шаг*_1. Задается начальное приближение (первая базисная точка) $x_0 = \{x_0^{(1)}, x_0^{(2)}, ..., x_0^{(n)}\}$, начальный *Шаг_h* для поиска направления спуска, точность решения $\delta$ (предельное значение для шага $h$), номер итерации $k = 0$.

*Шаг*_2. Исследующий поиск.

Определяется направление минимизации целевой функции $f(x) = f(x^{(1)}, x^{(2)}, ..., x^{(n)})$ в базисной точке $x_0 = \{x_0^{(1)}, x_0^{(2)}, ..., x_0^{(n)}\}$. Для этого последовательно дают приращение переменным $x^{(j)}$ в точке $x_k$.

Присвоим $z = x_k$. Для всех $j$ ( $j = 1,2,...,n$ ) циклически даем приращение переменным $x^{(j)}$ и формируем новое значение $z$:

$-z^{(j)} = x_k^{(j)} + h$, если $f(z) < f(x_k)$,

$-z^{(j)} = x_k^{(j)} - h$, если $f(z) > f(x_k)$,

$-z^{(j)} = x_k^{(j)}$, если $f(z) = f(x_k)$.

*Шаг*_3. Оценка результатов исследующего поиска.

Если не определилось подходящее направление (то есть $z = x_k$), то обследование окрестности базисной точки $x_k$ повторяется с меньшим шагом $h$ (например, $h = h/2$).

Проверяется текущее значение шага. Если $h > \delta$, то перейти к шагу 2, то есть повторить обследование точки $x_k$. Если $h \le \delta$, то поиск заканчивается, так как достигнуто предельное значение для шага $h$, и найти приемлемое направление спуска не удалось. В этом случае полагается $\tilde{x} = x_k$, $\tilde{y} = f(x_k)$.

*Шаг*_4. Поиск по образцу.

Если в результате исследующего поиска найдена точка с меньшим значением целевой функции ( $z \neq x_k$ ), то требуется найти новую базисную точку в направлении вектора $(z - x_k)$: $x_{k+1} = x_k + \lambda(z - x_k)$, где $\lambda$ – коэффициент «ускорения поиска». Определяется такое значение $\lambda = \lambda_k$, при котором достигается наименьшее значение целевой функции в выбранном направлении, то есть функции $\varphi(\lambda) = f(x_k + \lambda(z - x_k))$. В зависимости от способа выбора $\lambda_k$ возможны следующие варианты метода:

$-\lambda_k = const -$ постоянная для всех итераций;

–задается начальное $\lambda_0 = \lambda$, а далее $\lambda_k = \lambda_{k+1}$, если $f(x_{k+1}) < f(x_k)$, иначе дробим $\lambda_k$, пока не выполнится это условие;

$-\lambda_k$ определяется решением задачи одномерной минимизации функции $\varphi(\lambda)$.

В результате этого шага определяется новая базисная точка $x_{k+1} = x_k + \lambda(z - x_k)$, и поиск оптимального решения повторяется с шага 2, при этом $k = k + 1$.

## 1.2 Многомерный поиск с использованием производных

### 1.2.1 Градиентные методы

Как известно, градиент функции в некоторой точке $x_k$ направлен в сторону наискорейшего локального возрастания функции и перпендикулярен линии уровня (поверхность постоянного значения функции $f(x)$, проходящей через точку $x_k$). Вектор, противоположный градиенту $f'(x_k)$, называется антиградиентом, который направлен в сторону наискорейшего убывания функции $f(x)$. Выбирая в качестве направления спуска $p_k$ антиградиент $-f'(x_k)$ в точке $x_k$, приходим к итерационному процессу вида: $x_{k+1} = x_k - \alpha_k f'(x_k)$, где $\alpha_k > 0$, $k = 0,1,2,\ldots$ . В координатной форме этот процесс записывается следующим образом: $x_{k+1}^{(i)} = x_k^{(i)} - \alpha_k \frac{\partial f}{\partial x^i}(x_k)$, где $i = 1,2,\ldots,n.$

Все итерационные процессы, в которых направление движения на каждом шаге совпадает с антиградиентом функции, называются градиентными методами. Они отличаются друг от друга только способом выбора шага $\alpha_k$. Существует много различных способов выбора $\alpha_k$, но наиболее распространены: метод с постоянным шагом, метод с дроблением шага и метод наискорейшего спуска.

*Градиентный метод с постоянным шагом*

Основная проблема в градиентных методах – это выбор шага $\alpha_k$. Достаточно малый шаг $\alpha_k$ обеспечивает убывание функции, то есть выполнение неравенства $f(x_k - \alpha_k f'(x_k)) < f(x_k)$, но может привести к неприемлемо большому количеству итераций, необходимых для достижения точки минимума. С другой стороны, слишком большой шаг может вызвать неожиданный рост функции (невыполнение условия убывания) либо привести к колебаниям около точки

минимума. Однако проверка условия убывания на каждой итерации является довольно трудоемкой, поэтому в методе градиентного спуска с постоянным шагом задают $\alpha = \alpha_k$ постоянным и достаточно малым, чтобы можно было использовать этот шаг на любой итерации. При этом приходится мириться с возможно большим количеством итераций. Положительным фактором является то, что трудоемкость каждой итерации, минимальна (вычисляется только градиент $f'(x_k)$).

*Схема алгоритма*

*Шаг*_1. Задаются начальное приближение $x_0$, постоянный шаг $\alpha$, условия останова алгоритма $\varepsilon$. Вычисляется значение градиента $f'(x_k)$ – направление поиска. Присваивается $k = 0$.

*Шаг*_2. Определяется точка очередного эксперимента $x_{k+1} = x_k - \alpha f'(x_k)$, или, в координатной форме: $x_{k+1}^{(i)} = x_k^{(i)} - \alpha \dfrac{\partial f}{\partial x^i}(x_k^{(i)},...,x_k^{(n)})$, где $i = 1,2,...,n$.

*Шаг*_3. Вычисляется значение градиента $f'(x_{k+1})$, или, в координатной форме: $\{\dfrac{\partial f}{\partial x^{k+1}}(x_{k+1}^{(1)},...,x_{k+1}^{(n)}),...,\dfrac{\partial f}{\partial x^n}(x_{k+1}^{(1)},...,x_{k+1}^{(n)})\}$.

*Шаг*_4. Если $\left\| f'(x_{k+1}) \right\| \le \varepsilon$, то поиск заканчивается, при этом $\tilde{x} = x_{k+1}, \tilde{y} = f(x_{k+1})$, иначе $k = k+1$ и переход к шагу 2.

*Градиентный метод с дроблением шага*

В методе градиентного спуска с дроблением шага величина шага $\alpha_k$ выбирается так, чтобы выполнялось неравенство $f(x_k - \alpha_k f'(x_k)) - f(x_k) \le -\delta\alpha_k \left\| f'(x_{k+1}) \right\|^2$, где $0 \le \delta \le 1$ – произвольно выбранная постоянная (одна и та же для всех итераций). Это требование на выбор шага $\alpha_k$ более жесткое, чем условие убывания, но имеет тот же смысл: функция должна убывать от итерации к итерации. Однако при выполнении неравенства функция будет уменьшаться на гарантированную величину, определяемую правой частью неравенства.

Процесс выбора шага протекает следующим образом. Выбираем число $\alpha>0$, одно и то же для всех итераций. На *k*-й итерации проверяем выполнение неравенства при $\alpha_k = \alpha$. Если оно выполнено, полагаем $\alpha_k = \alpha$ и переходим к следующей итерации. Если нет, то шаг $\alpha_k$

дробим, например, уменьшаем каждый раз в два раза, до тех пор, пока оно не выполнится.

*Схема алгоритма*

*Шаг*_1. Задаются $x_0$, $\varepsilon$, $\delta$ и начальное значение шага $\alpha$, $k = 0$. Вычисляется значение градиента $f'(x_k)$ – направление поиска.

*Шаг*_2. Проверяется условие $f(x_k - \alpha f'(x_k)) \le -\delta\alpha_k \left\| f'(x_k) \right\|^2$. Если оно выполняется, то переходим к шагу 3, иначе дробим значение $\alpha$ ($\alpha = \alpha / 2$) и повторяем *Шаг*_2.

*Шаг*_3. Определяется точка очередного эксперимента: $x_{k+1} = x_k - \alpha f'(x_k)$.

*Шаг*_4. Вычисляется значение градиента в точке $x_{k+1}$: $f'(x_{k+1})$.

*Шаг*_5. Проверка сходимости.

Если $\left\| f'(x_{k+1}) \right\| \le \varepsilon$, то поиск заканчивается, при этом $\tilde{x} = x_{k+1}, \tilde{y} = f(x_{k+1})$. Иначе $k = k + 1$ и переход к шагу 2.

*Метод наискорейшего спуска.*

В градиентном методе с постоянным шагом величина шага, обеспечивающая убывание функции $f(x)$ от итерации к итерации, оказывается очень малой, что приводит к необходимости проводить большое количество итерации для достижения точки минимума. Поэтому методы спуска с переменным шагом являются более экономными. Алгоритм, на каждой итерации которого шаг $\alpha_k$ выбирается из условия минимума функции $f(x)$ в направлении движения, то есть

$$f(x_k - \alpha_k f'(x_k)) = \min_{\alpha \ge 0} f(x_k - \alpha f'(x_k)),$$

называется методом наискорейшего спуска.

Реализация метода наискорейшего спуска предполагает решение на каждой итерации довольно трудоемкой вспомогательной задачи одномерной минимизации. Как правило, метод наискорейшего спуска дает выигрыш в числе машинных операций, поскольку обеспечивает движение с самым выгодным шагом. Решение задачи одномерной минимизации связано с дополнительными вычислениями только самой функции $f(x)$, тогда как основное машинное время тратится на вычисление ее градиента $f'(x_k)$. Следует иметь в виду, что одномерную минимизацию можно производить любым методом одномерной оптимизации, что порождает различные варианты метода наискорейшего спуска.

*Схема алгоритма*

*Шаг*_1. Задаются $x_0$, $\varepsilon$, $k=0$. Вычисляется градиент $f'(x_0)$ – направление поиска.

*Шаг*_2. Определяется точка очередного эксперимента: $x_{k+1}=x_k-\alpha f'(x_k)$, где $\alpha_k$ – минимум задачи одномерной минимизации: $\alpha_k=\arg\min\limits_{\alpha\geq 0} f(x_k-\alpha f'(x_k))$.

*Шаг*_3. Вычисляется значение градиента в точке $x_{k+1}$: $f'(x_{k+1})$.

*Шаг*_4. Если $\left\| f'(x_{k+1})\right\| \leq \varepsilon$, то поиск заканчивается, при этом $\tilde{x}=x_{k+1}, \tilde{y}=f(x_{k+1})$. Иначе $k=k+1$ и переход к шагу 2.

### 1.2.2 Метод покоординатного спуска

Желание уменьшить объем вычислительной работы, требуемой для осуществления одной итерации метода наискорейшего спуска, привело к созданию методов покоординатного спуска. Пусть $x_0=\{x_0^{(1)}, x_0^{(2)},..., x_0^{(n)}\}^T$ – начальное приближение. Вычислим частную производную по первой координате и примем: $x_1=x_0-\alpha_0\dfrac{\partial f}{dx^1}(x_0)e_1$, где $e_1=\{1,0,...,0\}^{\mathrm{T}}$ – единичный вектор оси $x^{(1)}$. Следующая итерация состоит в вычислении точки $x_2$ по формуле: $x_2=x_0-\alpha_0\dfrac{\partial f}{dx^2}(x_0)e_2$, где $e_2=\{0,1,...,0\}^{\mathrm{T}}$ – единичный вектор оси $x^{(2)}$ и т. д. Таким образом, в методах покоординатного спуска поиск происходит по ломаной линии, состоящей из отрезков прямых, параллельных координатным осям (см. рис.1.1).

Спуск по всем координатам составляет одну «внешнюю» итерацию. Пусть $k$ – номер очередной внешней итерации, а $j$ – номер той координаты, по которой производится спуск. Тогда формула, определяющая следующее приближение к точке минимума, имеет вид: $x_{kn+j}=x_{kn+j-1}-\alpha_{kn+j-1}\dfrac{\partial f}{\partial x^j}(x_{kn+j-1})e_j$, где $k=0,1,2,...,\ j=1,2,...,n$.

В координатной форме формула имеет вид: $x_{kn+j}^{(i)}=x_{kn+j-1}^{(i)}$, если $i=j$, и $x_{kn+j}^{(i)}=x_{kn+j-1}^{(i)}-\alpha_{kn+j-1}\dfrac{\partial f}{\partial x^j}(x_{kn+j-1})$, если $i\neq j$.

После $j = n$ итераций счетчик внешних итераций $k$ увеличивается на единицу, а $j$ принимает значение равное единице.

Величина шага $\alpha_k$ выбирается на каждой итерации аналогично тому, как это делается в градиентных методах. Например, если $\alpha_k = \alpha$, то имеем покоординатный спуск с постоянным шагом.

*Схема алгоритма покоординатного спуска с постоянным шагом*

*Шаг*_1. При $k = 0$ вводятся исходные данные $x_0$, ε, α.

*Шаг*_2. Осуществляется циклический покоординатный спуск по $j$ ( $j = 1,2,...,n$ ) из точки $x_{kn}$ по формуле:

$$x_{kn+j} = x_{kn+j-1} - \alpha \frac{\partial f}{\partial x^j}(x_{kn+j-1})e_j \ .$$

*Шаг*_3. Если $\left\|(x_{(k+1)n} - x_{kn})\right\| \le \varepsilon$, то поиск минимума заканчивается – $\tilde{x} = x_{(k+1)n}$, $\tilde{y} = f(x_{(k+1)n})$. Иначе $k = k + 1$ и переход к шагу 2.

Если же шаг $\alpha_k$ выбирается из условия минимума функции $\varphi(\alpha) = f(x_{kn+j-1} - \alpha \frac{\partial f}{\partial x^j}(x_{kn+j-1})e_j)$, то мы получаем аналог метода наискорейшего спуска, называемый методом Гаусса – Зейделя.

*Схема алгоритма Гаусса – Зейделя*

*Шаг*_1. При $k = 0$ вводятся исходные данные $x_0$, ε.

*Шаг*_2. Осуществляется циклический покоординатный спуск по $j$ ( $j = 1,2,...,n$ ) из точки $x_{kn}$ по формуле

$$x_{kn+j} = x_{kn+j-1} - \alpha_{kn+j-1} \frac{\partial f}{\partial x^j}(x_{kn+j-1})e_j \ ,$$

где $\alpha_{kn+j-1}$ является решением задачи одномерной минимизации функции $\varphi(\alpha) = f(x_{kn+j-1} - \alpha \frac{\partial f}{\partial x^j}(x_{kn+j-1})e_j)$.

*Шаг*_3. Если $\left\|(x_{(k+1)n} - x_{kn})\right\| \le \varepsilon$, то поиск минимума заканчивается, иначе $k = k + 1$ и переход к шагу 2.

### 1.2.3 Методы оврагов

Градиентные методы медленно сходятся в тех случаях, когда поверхности уровня целевой функции $f(x)$ сильно вытянуты. Этот

факт известен в литературе как «эффект оврагов». Суть эффекта в том, что небольшие изменения одних переменных приводят к резкому изменению значений функции – эта группа переменных характеризует «склон оврага», а по остальным переменным, задающим направление «дно оврага», функция меняется незначительно. На рисунке 1.4 изображены линии уровня «овражной» функции траектория градиентного метода характеризуется довольно быстрым спуском на «дно оврага», и затем медленным зигзагообразным движением в точку минимума.

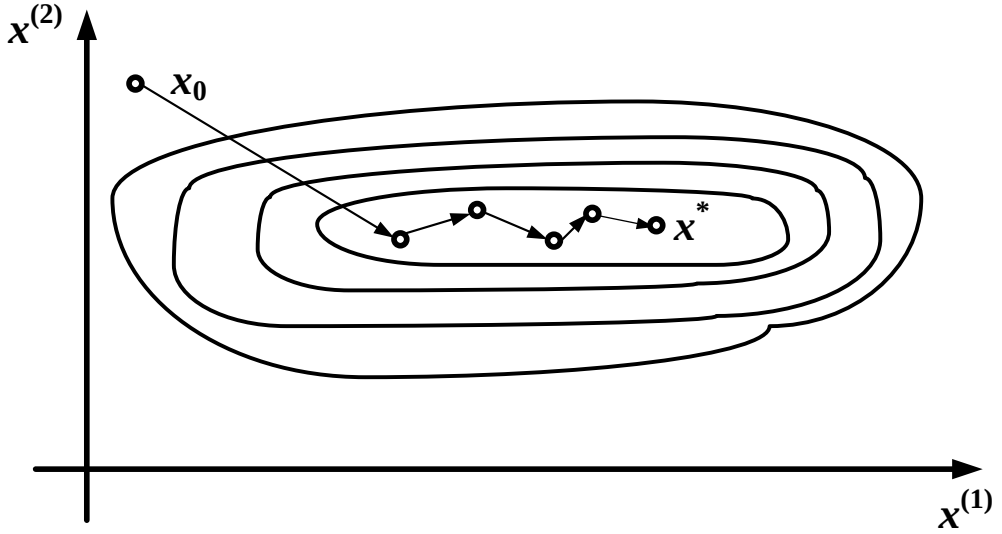


Рисунок 1.4 – Пример «эффекта оврагов»

Существуют различные подходы для определения точки минимума функции $f(x)$ в овражной ситуации. Большинство из этих методов основаны на эвристических подходах. Их можно применять в ситуациях, когда применение более совершенных методов невозможно или нецелесообразно, например, значение целевой функции вычисляется со значительными погрешностями, информация о ее свойствах недостаточна, и т. д. Эти методы просты в реализации и довольно часто применяются на практике, позволяя в ряде случаев получить удовлетворительное решение задачи.

Рассмотрим эвристическую схему, которая содержит два основных этапа. Оба этапа представляют собой аналоги градиентного спуска с постоянным шагом. Только вместо градиента $f'(x_k)$ используется вектор $g(x)$, формируемый из координат $f'(x_k)$, но на каждом из этапов по разным правилам.

На первом этапе задается малое число $\delta_1 << 1$ и используется градиентный спуск, где вместо градиента $f'(x_k)$ берется вектор $g(x) = \{g^{(1)}(x),..., g^{(n)}(x)\}$, который определяется следующим образом:

$$g^{(i)}(x) = \frac{\partial f}{\partial x^i}(x), \text{ если } |\frac{\partial f}{\partial x^i(x)}| > \delta_1,$$

$$g^{(i)}(x) = 0, \text{ если } |\frac{\partial f}{\partial x^i(x)}| \le \delta_1, \qquad (1.2)$$

где $i = 1,2,...n$.

Таким образом, спуск производится лишь по тем переменным, в направлении которых производная целевой функции достаточно велика. Это позволяет быстро спуститься на «дно оврага». Спуск продолжается до тех пор, пока метод не зациклится, то есть до тех пор, пока каждая следующая итерация позволяет найти точку, в которой значение функции меньше, чем значение, найденное в предыдущей итерации. После этого переходим к следующему этапу.

На втором этапе задается некоторое большое число $\delta_2 >> 1$ и используется процедура спуска, где вместо градиента $f'(x_k)$ берется вектор $g(x) = \{g^{(1)}(x),..., g^{(n)}(x)\}$, который определяется следующим образом:

$$g^{(i)}(x) = \frac{\partial f}{\partial x^i}(x), \text{ если } |\frac{\partial f}{\partial x^i(x)}| < \delta_2,$$

$$g^{(i)}(x) = 0, \text{ если } |\frac{\partial f}{\partial x^i(x)}| \ge \delta_2, \qquad (1.3)$$

где $i = 1,2,...n$.

В этом случае перемещение происходит по «берегу» оврага вдоль его «дна». Как и на первом этапе, спуск продолжается до тех пор, пока метод не зациклится.

После выполнения первого и второго этапов принимается решение о завершении работы или продолжении. Для этого сравнивается норма разности предыдущей точки, то есть точки, которую мы имели до применения первого и второго этапов, с текущей точкой, то есть полученной после применения с точностью решения задачи $\varepsilon_1$. Если эта норма меньше $\varepsilon_1$ и норма градиента в текущей точке меньше $\varepsilon_3$, то поиск заканчивается и последняя вычисленная точка принимается за приближенное решение задачи. Иначе для текущей точки вновь повторяем первый и второй этапы и т. д.

*Схема алгоритма*

*Шаг*_1. Задаются $x_0$, $\varepsilon_1$, $\varepsilon_3$, $\delta_1$, $\delta_2$, $\alpha_1$ – постоянный шаг первого этапа и $\alpha_2$ – постоянный шаг второго этапа ($\alpha_1 < \alpha_2$), $k = 0$.

*Шаг*_2. Первый этап.

Из точки $x_к$ осуществляется спуск на «дно оврага» с постоянным шагом $\alpha_1$. При спуске очередная точка вычисляется по формуле: $x_{j+1} = x_j - \alpha_1 g(x_j)$, где $g(x) = \{g^{(1)}(x),...,g^{(n)}(x)\}$, $g^{(i)}(x)$ вычисляются по формуле (3.2). Предположим, что процесс остановится в точке $x_l$. Тогда переходим ко второму этапу.

*Шаг*_3. Второй этап.

Из точки $x_l$ осуществляется спуск вдоль «дна оврага» с постоянным шагом $\alpha_2$. При спуске очередная точка вычисляется по формуле $x_{j+1} = x_j - \alpha_2 g(x_j)$, где $g(x) = \{g^{(1)}(x),...,g^{(n)}(x)\}$, а $g^{(i)}(x)$ вычисляются по формуле (3.3). Предположим, что процесс остановится в точке $x_m$.

*Шаг*_4. Если $\|x_k - x_m\| \le \varepsilon_1$ и $\|f'(x_k)\| \le \varepsilon_3$, то полагаем $\tilde{x} = x_m, \tilde{y} = f(x_m)$ и поиск минимума заканчивается. Иначе $k = m$ и переходим к шагу 2.

*Метод Гельфанда*

Следующая эвристическая схема, предложенная И.М. Гельфандом, состоит в следующем. Пусть $x_0$ и $\tilde{x}_0$ – две произвольные близкие точки. Из $x_0$ совершают обычный градиентный спуск с постоянным шагом и после нескольких итераций с малым шагом $\alpha$ попадают в точку $u_0$. То же самое делаем для точки $\tilde{u}_0$, получая точку $\tilde{x}_1$. Две точки $u$, $\tilde{x}_0$ лежат в окрестности «дна оврага». Соединяя их прямой, делаем «большой шаг» $\lambda$ в полученном направлении, перемещаясь «вдоль дна оврага» (шаг $\lambda$ называют овражным шагом). В результате получаем точку $x_1$. В ее окрестности выбираем точку $\tilde{u}_0$ и повторяем процедуру.

*Схема алгоритма*

*Шаг*_1. Вводятся начальное приближение $x_0$, точность решения $\varepsilon_1$ и $\varepsilon_3$, шаг $\alpha$ для градиентного спуска, начальное значение $\lambda$ для овражного шага, $k = 0$. Из точки $x_0$ осуществляется градиентный

спуск с постоянным шагом $\alpha$ на дно оврага. В результате получается точка $u_0$.

*Шаг*_2. В окрестности $x_k$ берется точка $\tilde{x}_k$ и из нее осуществляется градиентный спуск. В результате получается точка $\tilde{u}_k$.

*Шаг*_3. Новая точка $x_{k+1}$ определяется следующим образом. По формуле $x'_{k+1} = u_k + \lambda \frac{(\tilde{u}_k - u_k)}{\|\tilde{u}_k - u_k\|}$ при $f(\tilde{u}_k) < f(u_k)$, или $x'_{k+1} = \tilde{u}_k + \lambda \frac{(u_k - \tilde{u}_k)}{\| u_k - \tilde{u}_k \|}$ при $f(\tilde{u}_k) > f(u_k)$. Из нее осуществляем градиентный спуск и получаем точку $u'_{k+1}$. Если $f(u'_{k+1}) > f(u_k)$, то полагаем $x_{k+1} = x'_{k+1}$ и $u_{k+1} = u'_{k+1}$. Иначе уменьшаем овражный шаг $\lambda$ (например, в два раза – $\lambda = \lambda / 2$) и повторяем *Шаг*_3.

*Шаг*_4. Если $\|u_{k+1} - u_k\| \le \varepsilon_1$ и $\|f'(u_{k+1})\| \le \varepsilon_3$, то полагаем $\tilde{x} = u_{k+1}$, $\tilde{y} = f(u_{k+1})$ и поиск минимума заканчивается. Иначе $k = k + 1$ и переходим к шагу 2.

## 2 ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

1. Изучить изложенные методы многомерной безусловной оптимизации:

ЛР 3.1 – Методы прямого поиска.

ЛР 3.2 – Методы многомерного поиска с использованием производных.

2. Составить программы реализующие методы многомерной безусловной минимизации и найти точку минимума целевой функции $f(x) = f(x^{(1)}, x^{(2)})$ с заданной точностью $\varepsilon$ указанными методами. Начальное приближение $x_0$ и точность $\varepsilon$ приведены в таблице 2.1. Сравнить результаты, полученные разными методами для одной и той же целевой функции (в частности, сравнить число вычислении целевой функции, понадобившихся для получения заданной точности). Для каждого применяемого метода построить траекторию промежуточных точек, получаемых на очередных шагах метода и сходящихся к точке минимума.

3. Оформить отчет о выполнении задания с приведением условия задачи, алгоритмов и программ, указанных в задании методов минимизации, графиков траекторий промежуточных приближений

(файл – *MathCAD*), таблицы результатов сравнения рассмотренных методов, заключения по результатам сравнения методов.

Целевая функция зависит от двух аргументов:

$$f(x) = (x_1 - a)^2 + (x_2 - b)^2 + e^{c \cdot x_1^2 + d \cdot x_2}$$

Таблица 2.1 – Варианты задания

| № | Целевая функция | | | | Начальное приближение | Точность решения |
|---|---|---|---|---|---|---|
| | *a* | *b* | *c* | *d* | | |
| 1 | 1 | -1,4 | 0,01 | 0,11 | (1;0) | 0,0001 |
| 2 | 2 | -1,3 | 0,04 | 0,12 | (0;1) | 0,00005 |
| 3 | 10 | -0,5 | 0,94 | 0,2 | (0;0) | 0,0001 |
| 4 | 15 | 0 | 1,96 | 0,25 | 1,96 | 0,025 |
| 5 | 3 | -1,2 | 0,02 | 1,3 | (0;-1) | 0,00005 |
| 6 | 11 | -0,4 | 1 | 0,21 | (-1;0) | 0,0001 |
| 7 | 10 | -1 | 1 | 2 | (1;0) | 0,0003 |
| 8 | 15 | -0,5 | 2,25 | 2,5 | (0;0) | 0,0002 |
| 9 | 20 | 0,4 | 0,3 | 0,3 | (0;-1) | 0,0001 |
| 10 | 25 | 0,9 | 0,35 | 0,35 | (1;0) | 0,0004 |
| 11 | 15 | -1 | 2 | 2,5 | (0;0) | 0,0002 |
| 12 | 7 | 0,4 | 0,1 | 0,2 | (0;-1) | 0,0001 |
| 13 | 5 | 0,3 | 0,35 | 0,35 | (1;0) | 0,0001 |
| 14 | 20 | 3 | 1,96 | 0,25 | 1,96 | 0,005 |
| 15 | 3 | -1,2 | 0,02 | 1,3 | (0;-1) | 0,00005 |
| 16 | 11 | -0,4 | 1 | 0,21 | (-1;0) | 0,0001 |
| 17 | 10 | -1 | 1 | 2 | (1;0) | 0,0003 |
| 18 | 15 | -0,5 | 2,25 | 2,5 | (0;0) | 0,0002 |
| 19 | 20 | 0,4 | 0,3 | 0,3 | (0;-1) | 0,0001 |
| 20 | 3 | -1,2 | 0,02 | 1,3 | (0;-1) | 0,00005 |
| 21 | 25 | 0,9 | 0,35 | 0,35 | (1;0) | 0,0004 |
| 22 | 15 | -1 | 2 | 2,5 | (0;0) | 0,0002 |
| 23 | 7 | 0,4 | 0,1 | 0,2 | (0;-1) | 0,0001 |
| 24 | 5 | 0,3 | 0,35 | 0,35 | (1;0) | 0,0001 |
| 25 | 20 | 3 | 1,96 | 0,25 | 1,96 | 0,005 |
| 26 | 3 | -1,2 | 0,02 | 1,3 | (0;-1) | 0,00005 |
| 27 | 10 | -0,5 | 0,94 | 0,2 | (0;-1) | 0,0001 |
| 28 | 15 | 0 | 1,96 | 0,25 | (1;0) | 0,0001 |
| 29 | 3 | -1,2 | 0,02 | 1,3 | 1,96 | 0,005 |
| 30 | 15 | -1,2 | 0,01 | 1,3 | (0;-1) | 0,005 |

## 3 ТРЕБОВАНИЕ К ОТЧЕТУ

В отчете должны быть отображены следующие пункты:

1. Задание.
2. Краткие теоретические сведения.
3. График функции в MathCAD.
4. Схемы алгоритмов.
5. Листинги основных частей программы.
6. Таблица результатов сравнения рассмотренных методов.
7. Заключение по результатам сравнения методов.

## 4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое экстремум функции многих переменных?
2. Как аналитически найти экстремум функции многих переменных?
3. Что такое симплекс?
4. Что такое отраженная точка?
5. Какие условия применяются для останова алгоритмов?
6. Метод поиска по симплексу.
7. Метод Хука-Дживса.
8. Метод деформируемого симплекса.
9. Достоинства и недостатки методов прямого поиска.
10. Градиентные методы.
11. Методы оврагов.
12. Метод покоординатного спуска.
13. Метод Гельфанда.
14. Что такое «эффект оврагов»?
15. Что такое «овражный шаг»?