# Одномерная оптимизация

## Постановка задачи

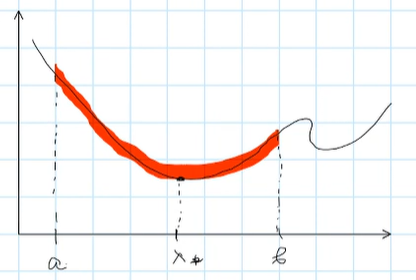
Для простоты будем искать минимум не на всей области определения (поиск глобального минимума), а на ограниченном отрезке

Самое важное в оптимизации – знать когда можно использовать какой-либо метод.

Все ниже рассматриваемые методы могут применяться только если

Функция называется унимодальной на , если существует такая точка

То есть функция унимодальна на отрезке, если она имеет на нём один единственный минимум

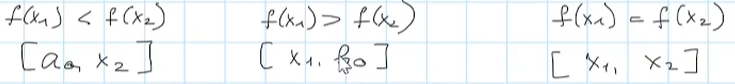
**

Пусть наша функция является непрерывной и унимодальной на отрезке, и мы хотим найти локальный минимум с заданной точностью

Почти все методы нахождения минимума на отрезке базируются на последовательном уменьшении интервала, который содержит точку минимума

Возьмем точки . Допустим, мы можем только вычислить значения функции в этих точках , а также на концах Из свойств унимодальной функции можем сказать, что минимум находится либо на отрезке либо на отрезке .

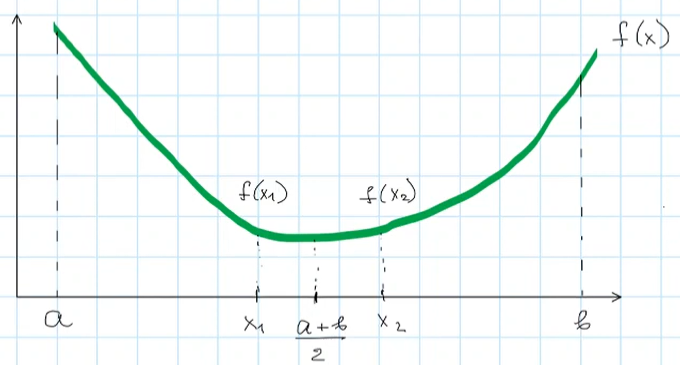
Этот алгоритм итеративный: на каждой итерации выбираем на отрезке какие-либо 2 точки, вычисляем в них значение функции , сравниваем их и сокращаем интервал в зависимости от сравнения



Алгоритм будет заканчиваться тогда, когда длина интервала, содержащего минимум станет .

Все методы одномерного поиска минимума отличаются только способом выбора точек . А об эффективности алгоритма можно судить по числу вычислений значения функции, которое нам необходимо для достижения запрошенной нами точности.

## Метод дихотомии



Есть функция заданная на отрезке . Мы хотим найти минимум с точностью

В чем заключается метод: выбираем середину отрезка . Задаётся некоторая – величина отступа, причём

Точки выбираются следующим образом: отступаем от найденной середины на расстояние , то есть

*Изображение выглядит как текст, доска

Автоматически созданное описание* . Далее вычисляем значения в точках и сокращаем исследуемый отрезок.

И так продолжаем сокращать отрезок до тех пор, пока его величина не станет меньше

В результате, за одну итерацию уменьшение интервала неопределённости происходит примерно в 2 раза

А за n итераций длина интервала неопределённости будет равна:

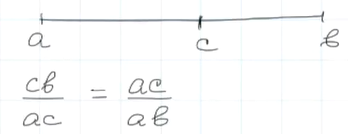
Соответственно, если мы хотим получить точность то нам необходимо совершить примерно итераций.

При этом на каждой итерации мы дважды вычисляем значение функции в точке.

## Золотое сечение

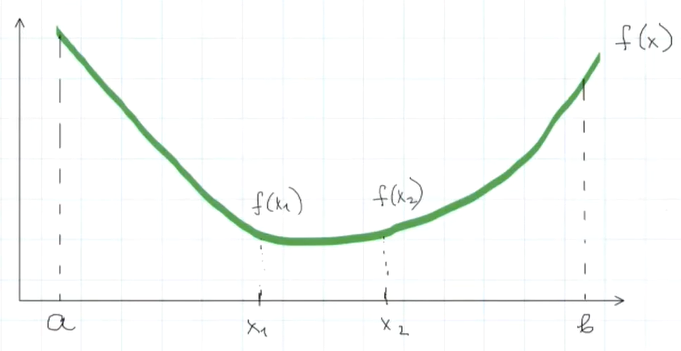
По умолчанию в методах оптимизации считается, что вычисление значения функции – это очень дорогостоящая операция, поэтому при сокращении интервала поиска до, например хотелось бы уже использовать точку для которой уже посчитано значение функции, в качестве одной из двух точек внутри нового интервала. Если мы это сделаем, то мы сократим количество обращений к вычислению функции на каждой итерации, то есть на каждой итерации нам нужно будет только одно вычисление функции.

Причем мы тогда на каждой итерации будем иметь 3 точки:

Метод золотого сечения позволяет применять вышесказанное

Золотое сечение – это такое деление отрезка, при котором отношение меньшей части к большей такое же как отношение большей части ко всей длине отрезка

Соответственно точки внутри интервалов поиска решений будут выбираться следующим образом:

1. На текущей итерации оба потенциальных интервала сокращения равны между собой
2. На каждой итерации интервал должен уменьшаться

Пусть точки в самом начале находятся симметрично относительно середины отрезка и эти точки будут делить этот отрезок в пропорции золотого сечения:

Не умаляя общности, будем считать, что (спроецируем его туда)

Тогда

Для нас подходит корень .

Значит положение первой точки нового интервала

Положение второй точки можно вычислить таким же образом, и эта точка уже будет одной из следующих точек следующего разбиения

Константа золотого сечения

Но интервал неопределенности будет уменьшаться не так быстро, как в методе дихотомии (там уменьшалось почти в 2 раза), но огромным плюсом является то, что значение функции мы вычисляем только 1 раз при каждой итерации

## Метод Фибоначчи

Данный метод – улучшение метода золотого сечения, служащего для нахождения минимума или максимума функции. Тут тоже на 1 итерации требуется вычислить значение функции дважды, а на всех последующих только 1 раз. Отличается он тем, что коэффициент сокращения интервала неопределенности будет меняться от итерации к итерации.

Допустим нам необходимо определить минимум как можно точнее, но мы можем произвести только n вычислений функции.

Пусть у нас интервал неопределенности и нам известно значение функции в некоторой точке . Тогда в какой точке следует вычислить значение функции, чтобы максимально сократить интервал неопределенности