Национальный исследовательский университет ИТМО
Факультет информационных технологий и программирования
Прикладная математика и информатика

Методы оптимизации
Отчет по лабораторной работе №1

Работу выполнили:
Булавко Тимофей Евгеньевич M3237
Илляхунов Ансар Адылович M3237

Преподаватель:
Андреев Юрий Александрович

г. Санкт-Петербург
2023 г.

Постановка задач:

1) Реализация метода градиентного спуска с постоянным шагом (learning rate)

2) Реализация любого метода одномерного поиска и градиентного спуска на его основе

3) Реализация метода Нелдера-Мида используя готовую реализацию в Python библиотеке scipy.optimize. Изучить возможности библиотеки scipy.optimize.

4) Провести исследование. Для исследования выбрать 2-3 квадратичные функции двух переменных, на которых эффективность методов будет явно отличаться; Сравнить методы на каждой из этих функциях:

a) исследовать сходимость и сравнить эффективность методов на выбранных функциях, с учетом количества итераций и количества вычислений значений минимизируемой функции и ее градиентов, в зависимости от желаемой точности;

b) исследовать работу методов в зависимости от выбора начальной точки;

c) проиллюстрировать примеры. Нарисовать графики рассматриваемых функций (3D), нарисовать графики с линиями уровня и траекториями методов (2D, в области задания). Вычисленные значения нужно оформить в виде сравнительных таблиц.

Доп. Задание:

Реализовать и исследовать на эффективность метод покоординатного спуска.

Задача 1: градиентный спуск с постоянным шагом

При градиентном спуске с постоянным шагом последовательность приближений определяется формулой:

x_new = x — h * grad (x), где h > 0

Алгоритм реализуется следующим образом:
1. Выбрать начальную точку ***X0***
2. Вычислить градиент текущей точки
3. Перейти в следующую точку по формуле

Реализацию алгоритма можно найти по ссылке.

Функция ***gradient_descent_constant_step***:
df — градиент искомой функции.
X0 — начальная точка нашего приближения.
eps - минимальная погрешность.
learning_rate — длинна шага.
max_iters — максимальное количество итераций метода.

Задача 2: градиентный спуск с золотым сечением

Так как мне был предоставлен выбор я решил использовать метод золотого сечения.

В данном алгоритме реализация остается такой же только теперь мы будем вычислять длину шага с помощью золотого сечения.

Для поиска минимума функции (в нашем случае мы будем перебирать коэффициенты на который будет домножен наш градиент) мы будем действовать так:

1) Заведем переменную ***phi = (1 + sqrt(5)) / 2***

2) У нас будут границы поиска ***l*** и ***r*** (левая и правая границы соответственно). После мы вычислим значение функции в следующих точках:

x = l + (r — l) / (phi + 1)
y = r — (r — l) / (phi + 1)

и посчитаем значение в этих точках (***f1*** и ***f2*** соответственно)

3) если ***f1 < f2***

r = y
y = x
f2 = f1
x = l + (r — l) / (phi + 1)
f2 = f(y)

в противном случае

l = x
x = y
f1 = f2
y = r — (r — l) / (phi + 1)
f2 = f(y)

4) Остановим итерацию алгоритма когда нас будет устраивать точность то есть ***r-l*** по модулю будет меньше ***eps***

Реализацию можно найти по ссылке.

Задача 3: метод Нелдера-Мида

Метод Нелдера-Мида является одним из методов оптимизации без использования градиента. Он особенно эффективен для решения задач безусловной оптимизации на малых размерностях.

Рассмотрим реализацию метода Нелдера-Мида:

1. Задаем начальный симплекс и вычисляем значения функции в его вершинах. Сортируем вершины симплекса по значению функции в них.

2. Пробуем расширить симплекс если мы не смогли улучшить ответ, то нам нужно сжать симплекс. Расширение симплекса — отражаем самую худшую вершину, если ответ улучшен то заменяем ее. Сжатие симплекса — сжимаем симплекс , оставляя пропорции, относительно лучшей вершины, после сжатия перезаписываем новые вершины.

3. Выполняем шаги пока не достигнем нужной точности.

4. Как ответ берем лучшую вершину.

Реализацию с помощью ***scipy*** можно найти по ссылке.

Задача 4: исследование

Для исследования были выбраны функции:

1. f(x, y) = 2x^2+2y^2+2xy+6x
2. f(x, y) = 0.6x^2 + 0.7y^2 + 0.12xy

Метод градиентного спуска с фиксированным шагом:

Сходимость данного метода будет сильно зависеть от выбора шага. Если шаг был выбран слишком большой, то метод разойдется, так как будет ходить вдоль отрезка. На больших значениях будет большая производная, а значит размер реального шага будет увеличиваться, поэтому траектория точек будет ходить вдоль прямой с потенциальным минимумом, но на каждой следующей итерации будет только отдаляться от него. При достаточно маленьком шаге метод будет сходиться, но для этого потребуется достаточно много итераций. Особенно заметно, что в точках близких к минимуму сдвиг точки будет невелик, так как длина градиента начинает постепенно уменьшаться. Это создает более плавную траекторию, что может хорошо работать для более сложных функций с большим количеством стационарных точек, так как траектория не будет "кидаться" по сторонам, но это приводит к малой эффективности.

Метод градиентного спуска с золотым сечением

Данный метод будет эффективнее метода с фиксированным шагом так как итераций приближения будет меньше, но при подсчетах длинны нашего шага, будет много обращений к нашей функции, поэтому если у нас запрос к функции это дорогостоящая операция этот метод будет не эффективен.

Метод Нелдера-Мида

Данный метод не требует вычисления градиентов. Не требует гладкости функции: этот метод хорошо работает даже для не гладких функций, которые могут вызывать проблемы для методов, основанных на градиенте. Метод Нелдера-Мида устойчив к выбору начального приближения и может находить локальные оптимумы. Из минусов это медленная сходимость: метод Нелдера-Мида может быть медленным для сходимости к оптимальному решению, особенно для функций с большим количеством переменных. Также из минусов большое количество обращений к фукнции.

Все графики по которым происходили исследования можно найти по ссылке

Доп задание (метод покоординатного спуска):

Метод покоординатного спуска является одним из простых методов оптимизации без градиентов, который основан на последовательной минимизации функции по одной переменной за раз. В нашем случае для вычисления шага мы использовали метод золотого сечения. Вычисление следующего приближения происходит как одномерное приближение по каждой координате. Из преимуществ этот метод прост в реализации и не требует вычисления градиента. Из недостатков это медленная сходимость и частое обращение к функции.

Как и в прошлом случае графики исследования функций находятся по ссылке