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

Методы оптимизации

Отчет по лабораторной работе №1

⟨Собрано 15 марта 2023 г.⟩

Работу выполнили:
Бактурин Савелий Филиппович M32331
Вереня Андрей Тарасович M32331
Сотников Максим Владимирович M32331

Преподаватель:
Свинцов Михаил

Задача 1

Постановка задачи

Реализуйте градиентный спуск с постоянным шагом (learning rate).

Решение

Поймем, сначала, что мы хотим добиться: мы хотели бы найти направление наискорейшего спуска с некоторой точки к минимуму на заданной плоскостью функцией f. Однако, при решении этой задачи возникает проблема с производительностью нахождения $\operatorname{argmin} f$ за счет появления тех или иных накладных расходов на подсчет не целочисленных значения, а также проблемой с нахождением такого шага λ, что наш алгоритм не «застрянет» в бесконечном поиске интересующей точки.

Введем обозначения, пусть α – есть некоторая константа, порядка 10^{-3}, $x_i = \{x_i^0, x_i^1, \ldots, x_i^{n-1}\}$ – некоторая координата в n-мерном пространстве, p_i – наше текущее направление.

Теперь рассмотрим идею *градиентного спуска*: оптимизацию нахождения необходимого минимум за k шагов мы будем осуществлять шаги в n-мерном пространстве в направлении, задаваемый как антиградиент функции f в точке, задаваемая предыдущем шагом, то есть

$$x_{i+1} = x_i - \alpha \cdot \nabla f(x_i),$$

где x_0 будет задаваться некоторым множеством $\text{INIT} = \{x_0^0, x_0^1, \ldots, x_0^{n-1}\}$ – то есть точка, от которой мы собираемся двигаться.

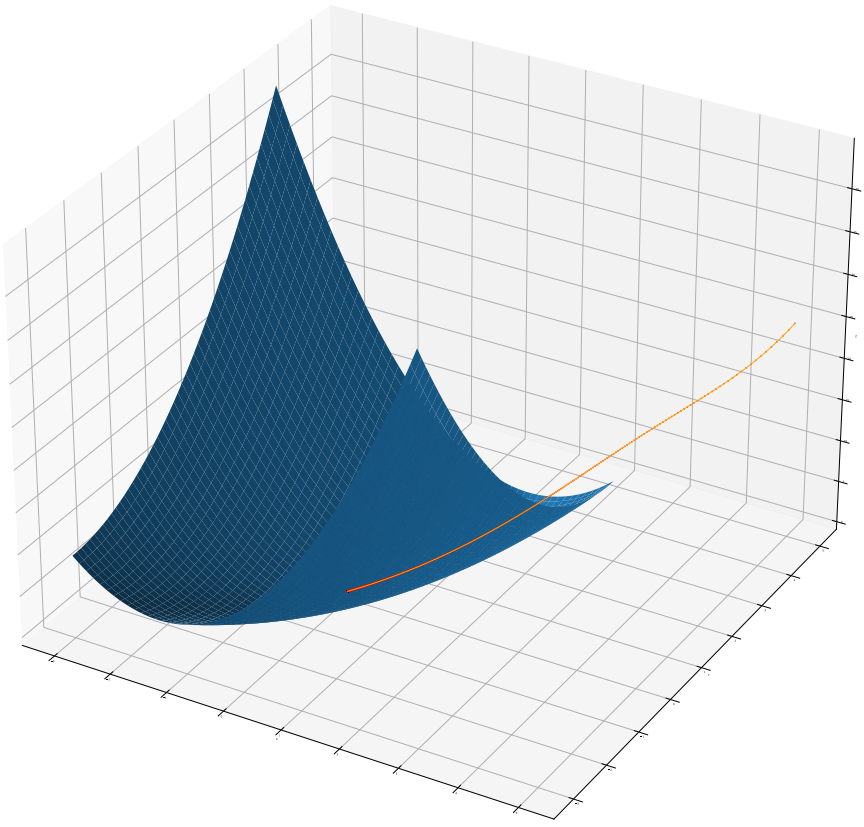
Итого, псевдо-алгоритм для этой задачи выглядит следующим образом:

```
function f(x):
        /*implementation defined*/

function ∇f(x):
        return [f(x)∂/∂x^0, f(x)∂/∂x^1, ..., f(x)∂/∂x^{n-1}]

function main:
        x_0 ← INIT
        α ← const
        forall i ∈ [1, k] do
                x_i ← x_{i-1} − α · ∇f(x_{i-1})
```

Разберем пример. Мы хотим найти методом градиентного спуска приближенный $\operatorname{argmin}_{\substack{x \in X \\ y \in Y}} f(x, y)$ функции $f(x, y) = x^2 - (x - y)^2$. Его направлением-антиградиентом будет $p_i = -\nabla f(x, y) = \{(-1) \cdot (2 \cdot x + 2 \cdot (x - y)), (-1) \cdot (-2 \cdot (x - y))\}$. В исходном коде `points` представляет из себя шаги, проделываемые алгоритмом от стартового состояния x_0 до некоторого приближенного x_k – являющийся минимумом. Наконец, представим всему миру полученную картину.



Задача 2

Постановка задачи

Реализуйте метод одномерного поиска (метод дихотомии, метод Фибоначчи, метод золотого сечения) и градиентный спуск на его основе.

Решение

Задача 3

Постановка задачи

Проанализируйте траекторию градиентного спуска на примере квадратичных функций. Для этого придумайте две-три квадратичные функции от двух переменных, на которых работа методов будет отличаться.

Решение

Задача 4

Постановка задачи

Для каждой функции:

(a) исследуйте сходимость градиентного спуска с постоянным шагом, сравните полученные результаты для выбранных функций;

(b) сравните эффективность градиентного спуска с использованием одномерного поиска с точки зрения количества вычислений минимизируемой функции и ее градиентов;

(c) исследуйте работу методов в зависимости от выбора начальной точки;

(d) исследуйте влияние нормализации (scaling) на сходимость на примере масштабирования осей плохо обусловленной функции;

(e) в каждом случае нарисуйте графики с линиями уровня и траекториями методов;

Решение

Задача 5

Постановка задачи

Реализуйте генератор случайных квадратичных функций n переменных с числом обусловленности k.

Решение

Изначально поймем, что такое *число обусловленности*. По своей сущности, это нечто, что может показать насколько может измениться значение функции при небольшом изменении аргумента. Для нахождения такого числа и, в следствии, нахождения некоторого вектора чисел, которые будут являться коэффициентами квадратичной формы, существует несколько способов: через матричный нормы – это исходит напрямую из рассматриваемого линейного уравнения вида $\mathfrak{A}x = \mathbf{b}$, где $\mathfrak{A}$ – линейный

оператор, $\mathbf{b}$ – вектор и x – переменная. Такой способ подошел бы нам, если б мы знали заранее нормы матриц. Из-за чего нам понадобится более сильное средство, именуемое как *сингулярное разложение*, а именно: возьмем некоторую матрицу A, тогда его число обусловленности будет равно отношению максимального и минимального из диагональных элементов; остальные же элементы на диагональной части матрицы будут коэффициентами разложения квадратичной формы.

Итак мы хотим сгенерировать матрицу такую, что $k = \frac{\max_{\forall i \in [0,n]} x_{i,i \in [0,n]}}{\min_{\forall i} x_{i,i}}$. Для этого мы получим максимальный элемент $\mathrm{MAX} = k \cdot \mathrm{MIN}$, а в качестве минимального возьмем случайное число из ограниченного операционной системой диапазоном, например $[0, 2^{64} - 1]$. Тогда как все остальные элементы следует брать из диапазона $[\mathrm{MIN} + 1, \mathrm{MIN} \cdot k)$.

Итого, псевдо-алгоритм для этой задачи выглядит следующим образом:

```
function random(l, r):
        return randomized R ∈ [l, ..., r)

function main(n, k):
        MIN ← random(0, 2^32)
        MAX ← MIN · k
        q ← [MIN, MAX, x_0 ..., x_{n-3}], ∀x_i = 0
        forall i ∈ [2, n] do
                q_i ← random(MIN + 1, MAX)
```

Для примера мы рассмотрим задачу. Необходимо сгенерировать квадратичную функцию с $n = 10$ переменными и $k = 5$ числом обусловленности. Воспользуемся заготовленной программой, которая выводит функцию в TEXвиде, и получим, как один из результатов вот такой:

$$\begin{aligned} f(x_0, x_1, \ldots, x_9) = {} & 55881652088902977 \cdot x_0^2 + 279408260444514885 \cdot x_1^2 \\ & + 168040993445098276 \cdot x_2^2 + 98341599851142922 \cdot x_3^2 \\ & + 126881448626083312 \cdot x_4^2 + 112063718763017541 \cdot x_5^2 \\ & + 167683450954662177 \cdot x_6^2 + 104734621927684905 \cdot x_7^2 \\ & + 196100745993640030 \cdot x_8^2 + 199634427735860479 \cdot x_9^2 \end{aligned}$$

Задача 6

Постановка задачи

Исследуйте зависимость числа итераций $T(n, k)$, необходимых градиентному спуску для сходимости в зависимости от размерности пространства $2 \leqslant n \leqslant 10^3$ и числа обусловленности оптимизируемой функции $1 \leqslant k \leqslant 10^3$.

Решение

Дополнительное задание

Постановка задачи

Реализуйте одномерный поиск с учетом условий Вольфе и исследуйте его эффективность. Сравните полученные результаты с реализованными ранее методами.

Решение