Лабораторная работа №2

Алгоритмы многомерной минимизации функции

Сысоев Александр, Зырянова Мария
Верблюжий случай

1 Постановка задания

Требуется реализовать алгоритмы поиска минимума функции нескольких переменных, исследовать их, оценить поведение:

- метод градиентного спуска
- метод наискорейшего спуска
- метод сопряженных градиентов

Ограничение на исследуемые функции – они должны быть квадратичными, то есть представимы в виде

$$f(x) = f(x_1, x_2, \cdots, x_n) = \frac{1}{2}\sum_{i,j=1}^{n} a_{ij}x_i x_j - \sum_{i=1}^{n} b_i x_i + c$$

.

В матричной форме эти функции имеют вид:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c, \text{ где } x = (x_1, \cdots, x_n)^T, b = (b_1, \cdots, b_n)^T$$

2 Предварительное исследование

Основная задача – найти минимум заданной функции. По теореме Ферма, если точка x^* – минимум функции, то $f'(x^*) = 0$. В многомерном случае можно применить этот критерий последовательно к каждой частной производной, и тогда получится критерий минимума функции нескольких переменных – равенство всех частных производных нулю. Величина, за это отвечающая – градиент, то есть если $x^* = (x_1^*, \cdots, x_n^*)$ – минимум функции, то

$$\nabla f(x^*) = 0$$

Рассмотрим частную производную по x_i квадратичной функции:

$$\frac{\delta}{\delta x_i} f(x_1, \cdots, x_n) = a_{ii}x_i + \sum_{j \neq i} \frac{1}{2}(a_{ij} + a_{ji})x_j - b_i$$

То есть в общем виде градиент квадратичной функции выглядит следующим образом:

$$\nabla f(x) = Ax - b$$

A – симметричная матрица ($\Rightarrow$ ее можно рассматривать, как треугольную), у которой на пересечении i строки и j столбца стоит величина $\frac{1}{2}(a_{ij} + aji)$.

В случае, если градиент в точке не равен нулю, он показывает направление наибольшего локального увеличения функции. Поэтому, если двигаться в направлении антиградиента (то есть $-\nabla f(x)$), то будет происходить движение в направлении убывания $f(x)$.

3 Исследуемые функции

Для анализа методов были выбраны следующие квадратичные функции:

1. $$f(x) = 64x_1^2 + 126x_1x_2 + 64x_2^2 - 10x_1 + 30x_2 + 13$$

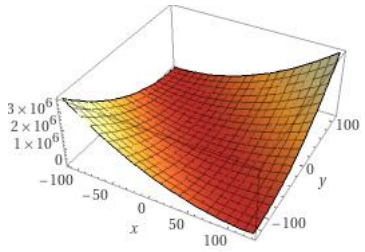

Рис. 1: График функции.

$$\min\{64x^2 + 126xy + 64y^2 - 10x + 30y + 13\} \approx$$
$$-187.39370078740157480314960 63 \text{ at } (x, y) \approx$$
$$(9.96062992125984251968503937, -10.039370078740157480314960 63)$$

Рис. 2: Минимум, найденный с помощью Wolphram Alpha.

4 Метод градиентного спуска

Как было написано ранее, двигаясь в направлении антиградиента, функция убывает. То есть по точке x^k можно получить точку x^{k+1} с меньшим значением функции f. Повторяя это нескольно раз, построится последовательность $x^{k+1} = x^k - \alpha^k \nabla f(x^k)$. При этом, α_k (величина шага) такова, что $f(x^{k+1}) < f(x^k)$.

Точка является миниумом, если в ней квадратичная форма положительно определена. Матрица A – симметричная, будем считать, что положительно определенная. Пусть L – наибольшее собственное значение A, тогда при любых $\alpha \in \left(0; \frac{2}{L}\right)$ и x из области определения фукнции $x^{k+1} = x^k - \alpha^k \nabla f(x^k)$, будет сходиться к единственной точке глобального минимума x^* линейно, со скоростью геометрической прогрессии. В этом случае последовательность x_k будет релаксационной и не будет происходить "проскакивания"стационарной точки.

Таким образом, посчитав максимальное собственное число A, примем $\alpha = \frac{2}{L}$, и затем будем выстраивать последовательность. Если на каком-то шаге $f(x^{k+1}) \geqslant f(x^k)$, то был сделан слишком большой шаг и следует уменьшить α. Примем новое $\alpha = \frac{\alpha}{2}$. Будем повторять до условия остановки: $\|\nabla f(x^k)\| \leqslant \epsilon$, либо до указанного колчиества итераций, если это было дано (в нашем исследовании – 2000).

ϵ	x_1	x_2	**Количество итераций**	**Значение минимума**	$x_{1_{min}}$	$x_{2_{min}}$
10^{-2}	10	10	876	-187.39370078722567	9.9606205571591	-10.039360714639415
10^{-2}	10	11	874	-187.393700787196	9.960619771803191	-10.039359929283508
10^{-2}	11	10	875	-187.39370078723766	9.9606208830298	-10.039361040510116
10^{-2}	11	11	854	-187.39370078704826	9.960616643413074	-10.03935680089339
10^{-3}	10	10	876	-187.39370078722567	9.9606205571591	-10.039360714639415
10^{-3}	10	11	874	-187.393700787196	9.960619771803191	-10.039359929283508
10^{-3}	11	10	875	-187.39370078723766	9.9606208830298	-10.039361040510116
10^{-3}	11	11	854	-187.39370078704826	9.960616643413074	-10.03935680089339
10^{-4}	10	10	876	-187.39370078722567	9.9606205571591	-10.039360714639415
10^{-4}	10	11	874	-187.393700787196	9.960619771803191	-10.039359929283508
10^{-4}	11	10	875	-187.39370078723766	9.9606208830298	-10.039361040510116
10^{-4}	11	11	854	-187.39370078704826	9.960616643413074	-10.03935680089339
10^{-5}	10	10	2000	-187.3937007873267	9.960623763053757	-10.039363920534072
10^{-5}	10	11	2000	-187.39370078732998	9.960623897700582	-10.039364055180897
10^{-5}	11	10	2000	-187.39370078732367	9.960623641492967	-10.039363798973282
10^{-5}	11	11	2000	-187.393700787349	9.960624748480237	-10.039364905960552
10^{-6}	10	10	2000	-187.3937007873267	9.960623763053757	-10.039363920534072
10^{-6}	10	11	2000	-187.39370078732998	9.960623897700582	-10.039364055180897
10^{-6}	11	10	2000	-187.39370078732367	9.960623641492967	-10.039363798973282
10^{-6}	11	11	2000	-187.393700787349	9.960624748480237	-10.039364905960552

5 Метод наискорейшего спуска

Это модификация предудщего метода, в которой коэффициент α каждый раз пересчитывается. После вычисления в начальной точке градиента, движение в направлении антиградиента делается не маленькими шагами, а до тех пор, пока функция убывает. При достижении минимума на выбранном направлении, снова вычисляется градиент функции и описанные действия повторяются.

На каждом k-ом шаге коэффициент α_k находится из решения задачи одномерной оптимизации:

$$\Phi_k(\alpha_k) \to min, \quad \Phi_k(\alpha_k) = f(x^k - \alpha_k \nabla f(x^k))$$

При этом, $\alpha_k > 0$, поэтому оптимизируем на промежутке $\left[0; \frac{2}{L}\right]$.

Таким образом, на каждом шаге будем находить α_k, принимать $x^{k+1} = x^k - \alpha^k \nabla f(x^k)$, до тех пор, пока $\|\nabla f(x^k)\| \leqslant \epsilon$, либо до указанного колчиества итераций, если это было дано.

ϵ	x_1	x_2	**Количество итераций**	**Значение минимума**	$x_{1_{min}}$	$x_{2_{min}}$
10^{-2}	10	10	727	-187.39367612608606	9.957118417601245	-10.03585857508156
10^{-2}	10	11	731	-187.39367587698453	9.957100727449223	-10.035840884929538
10^{-2}	11	10	722	-187.39367595717147	9.957106412343867	-10.035846569824182
10^{-2}	11	11	727	-187.39367612608606	9.957118417601245	-10.03585857508156
10^{-3}	10	10	698	-187.39370053896903	9.96027747821606	-10.039017635696377
10^{-3}	10	11	702	-187.3937005439067	9.960280998069365	-10.03902115554968
10^{-3}	11	10	695	-187.39370054180165	9.96027949290635	-10.039019650386665
10^{-3}	11	11	698	-187.39370053825326	9.96027697033254	-10.039017127812855
10^{-4}	10	10	797	-187.39370078492178	9.96059470922056	-10.039334866700877
10^{-4}	10	11	800	-187.39370078491754	9.960594684372225	-10.039334841852535
10^{-4}	11	10	794	-187.39370078494346	9.960594870462446	-10.039335027942759
10^{-4}	11	11	797	-187.39370078492178	9.96059470922056	-10.039334866700877
10^{-5}	10	10	987	-187.39370078737676	9.960626416187417	-10.039366573667733
10^{-5}	10	11	984	-187.39370078737647	9.960626390312978	-10.039366547793293
10^{-5}	11	10	983	-187.3937007873775	9.960626399550419	-10.039366557030736
10^{-5}	11	11	980	-187.39370078737645	9.960626394538146	-10.03936655201846
10^{-6}	10	10	1250	-187.3937007874012	9.960629568354005	-10.039369725834321
10^{-6}	10	11	1256	-187.39370078740217	9.960629570877959	-10.039369728358274
10^{-6}	11	10	1264	-187.39370078740228	9.960629572116014	-10.039369729596329
10^{-6}	11	11	1250	-187.3937007874012	9.960629568354005	-10.039369725834321

6 Метод сопряженных градиентов

Отличительная особенность данного метода – он решает квадратичную задачу оптимизации за конечное число шагов (не более, чем n итераций, где n – размерность пространства).

Рассмотрим ненулевые векторы $p^1, \cdots, p^n$. Они называются сопряженными относительно матрицы $A_{n\times n}$ или A-ортогональными, если для всех $i, j : i \neq j$ выполняется условие $(Ap^i, p^j) = 0$. Система из таких векторов линейно независима и образует базис в E_n.

Тогда минимизация квадратичной функции $f(x) = \frac{1}{2}(Ax, x) + (b, x) + c$ (A – положительно определенная) сводится к итерационному процессу $x^k = x^{k-1} + \alpha_k p^k$, где p^k – A-ортогональные. Такой последовательный спуск по A-ортогональным направлениям приводит к точке минимума квадратичной формы не более чем за n шагов. На каждой итерации необходимо выбрать параметры, дающие наилучший многочлен, который можно построить, учитывая все сделанные до текущего шага измерения градиента.

Так как функция квадратичная и есть условие A-ортогональности, нахождение параметров сводится до следующих действий на каждом этапе итераций:

$$\alpha = \frac{\|\nabla f(x^k)\|^2}{(Ap^k, p^k)}, \quad x^{k+1} = x^k + \alpha_k p^k$$

$$\nabla f(x^{k+1}) = \nabla f(x^k) + \alpha_k A p^k$$

$$\beta_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2}$$

$$p^{k+1} = -\nabla f(x^{k+1}) + \beta_k p^k$$

7 Выводы

Реализация: GitHub.