КОНЕЧНОМЕРНАЯ ОПТИМИЗАЦИЯ. АЛГОРИТМ МЕТОДА ПРОЕКЦИИ ГРАДИЕНТА

УДК 519.853.3

Новиков М.Ю.
кафедра анализа систем и принятия решений
Уральский федеральный университет, ВШЭМ

Аннотация. Повышение эффективности экономического и технического управления в условиях кризисных явлений является одной из самых важных задач на этапе модернизации не только экономики, но и всего общества. В статье рассматривается алгоритм метода проекции градиента как один из способов решения задач оптимизации.

Ключевые слова: оптимизация, метод, алгоритм, градиент, функция.

Abstract. Improving the efficiency of the economic and technical management in the context of the crisis is one of the most important tasks at the stage of modernization not only the economy but also the whole society. This article dedicated to the algorithm of the method of gradient projection as a way of solving optimization problems.

Key words: optimization, method, algorithm, gradient, function.

Одним из наиболее интенсивно используемых, эффективных и важных инструментариев повышения эффективности экономического управления в настоящее время являются математические методы оптимизации. Использование математики в экономике позволяет определить и формально описать наиболее важные связи.

Можно выделить три основных этапа проведения математического моделирования в экономике:

1) определяются цели и задачи исследования, выполняется описание объекта в виде экономической модели;
2) формулируется математическая модель изучаемого объекта, определяется набор методов исследования;
3) обработка и анализ данных.

Современные методы оптимизации довольно часто трактуются при помощи высокого математического уровня. В то же время существуют довольно простые алгоритмы для поиска минимума (максимума) функции. При этом описание последовательности действий на алгоритмическом языке с элементами математики дает большую степень свободы для программиста, оставляя за ним возможность самостоятельного выбора языка программирования. Для решения задач оптимизации программист-математик должен учитывать реальные возможности вычислительной техники с учетом специфики алгоритма.

В теории оптимизации выделяют три основных класса задач: задачи безусловной оптимизации; задачи условной оптимизации; задачи оптимизации при неполных данных. В зависимости от вида функции, подлежащей оптимизации выделяют задачи линейного программирования и задачи выпуклого программирования (в этих задачах f и X выпуклы). До настоящего времени не найдены эффективные алгоритмы для невыпуклых и многоэкстремальных задач, а также для задач дискретной оптимизации.

Рассмотрим алгоритм решения задач нелинейного программирования (задачи выпуклого программирования). Математическая модель подобных задач в общем виде формулируется следующим образом:

$$f = (x_1, x_2, \ldots, x_n) \to min\ (max). \qquad (1)$$

При этом функция f выпукла вниз (вверх) и непрерывно дифференцируема на множестве X (выпуклое замкнутое множество).

Для подобных задач нет единого метода решения. В зависимости от вида целевой функции и системы ограничений разработаны специальные методы решения. Рассмотрим алгоритм метода проекции градиента в общем виде [1, с.269]. На основе анализа алгоритма можно сформулировать рекомендации для его реализации на одном из современных языков программирования. Ниже представлен алгоритм метода проекции градиента наиболее оптимальный, по мнению автора.

Задача. Найти для заданной функции и заданного множества .

Предположение. Функция выпукла вниз и непрерывно дифференцируема на множестве X (выпуклое замкнутое множество)

Алгоритм

Начало.

I. Выбрать начальное приближение $x^0 \in X$.

II. Присвоить k=0.

Основной цикл.

III. Вычислить $\nabla f_0(x^k)$.

IV. Выбрать множитель β_k, удовлетворяющий условию:

$$0 < \beta' \leq \beta_k \leq \beta'' < +\infty \qquad (2)$$

V. Вычислить точку по формуле:

$$y^k = x^k - \beta_k \nabla f_0(x^k) \qquad (3)$$

VI. Вычислить точку y_X^k – проекцию точки y^k на выпуклое замкнутое множество X.

VII. Вычислить множитель ρ_k по формуле:

$$f_0\left(x^k - \rho_k\left(x^k - y_X^k\right)\right) = \min_{0 \leq \rho \leq 1} f_0(x^k - \rho\left(x^k - y_X^k\right) \qquad (4)$$

VIII. Присвоить:

$$x^{k+1} = x^k - \rho_k\left(x^k - y_X^k\right) \qquad (5)$$

$$k = k + 1 \qquad (6)$$

и перейти к шагу III.

Конец.

Наибольшую трудность при практической реализации алгоритма может доставить вычисление проекции точки на множество. Дело даже не в том, что далеко не для всех множеств можно найти точную проекцию. Исходя из определения, проекцией точки   на множество   называется вектор , удовлетворяющий условию:

$$\left\|y - P(y)\right\| = \min_{y \in X} \|y - a\| \tag{7}$$

Задача (7) является задачей минимизации строго выпуклой неотрицательной функции на множестве. Часто удобнее работать с эквивалентной ей задачей:

$$\left\|y - P(y)\right\|^2 = \min_{y \in X} \|y - a\|^2 \tag{8}$$

Очевидно, что задача отыскания проекции точки на выпуклое множество – это задача минимизации выпуклой функции на выпуклом множестве. Необходимо заметить, что, как правило, отыскать точно проекцию точки невозможно. Поэтому приходится пользоваться процедурами для нахождения приближенных значений проекции. Однако, при достаточно "простых" множествах X проекцию можно вычислить, пользуясь достаточно простыми "явными" формулами. Приведем несколько примеров таких множеств [2].

Пусть $a \in R^n$ произвольная точка.

- Шар

если $X = \{x \in R^n \mid \|x - x^0\| \le r$ - шар, то

$$\pi_X(a) = x^0 + \frac{a - x^0}{\|a - x^0\|} r \tag{9}$$

- Координатный параллелепипед

если $X = \{x \in R^n \mid b_j \le x_j \le c_j, \quad j = 1,\ldots,n\}$, то

$$(\pi_X(a))_j = \begin{cases} b_j & \\ a_j & \text{если} \\ c_j & \end{cases} \begin{matrix} a_j < b_j; \\ b_j \le a_j \le c_j; \\ a_j > c_j; \end{matrix} \tag{10}$$

- Неотрицательный ортант

если $X = R^n_+ = \{x \in R^n \mid x_j \ge 0, \quad j = 1,\ldots,n\}$, то

$$(\pi_X(a))_j = (\max(0, a_1), \ldots, \max(0, a_n)) \tag{11}$$

- Гиперплоскость

если $X = H^+_{p\beta} = \{x \in R^n \mid < p, x >= \beta\}$, ($p \ne 0$) то

$$(\pi_X(a))_j = a + (\beta - < p, a >) \frac{p}{\|p\|^2} \tag{12}$$

- Полупространство

если $X = H^+_{p\beta} = \{x \in R^n \mid < p, x > \ge \beta\}$, ($p \ne 0$), то

$$(\pi_X(a)) = a + \max(0; \beta - < p, a >) \frac{p}{\|p\|^2} \tag{13}$$

Перейдем к практическом применению метода проекции градиента. Предположим, что некая экономическая модель описана математическим языком в виде функции нелинейной функции:

$$f(x) = x_1^2 + x_2^2 \qquad (14)$$

при ограничениях, заданных системой следующих неравенств:

$$\begin{cases} -7 \le x_1 \le 5 \\ -5 \le x_2 \le -2 \end{cases} \qquad (15)$$

Следуя алгоритму, выбирается начальная точка x^0 , например: $x^0 = (-5, -3)$.

Значение градиента функции в этой точке будет равно: $\nabla f_0(x^0) = (-10, -6)$.

Выбираем множитель β_0=0.5

Вычисляем точку $y^k = x^k - \beta_k \nabla f_0(x^k)$

Проекцию точки на множество вычисляем по явной формуле для координатного параллелепипеда *(формула 10).*

Выход из основного цикла осуществляется по достижению необходимой точности. Как правило, выбирается значение модуля разности между текущим и предыдущим значением функции.

В данном примере на некоторой итерации алгоритм найдет решение: функция принимает минимальное значение, равное 4 в точке (0,-2).

Практические примеры задач нелинейного программирования широко встречаются в экономической деятельности. Приведем пример подобной задачи:

«Предприниматель решил выделить на расширение своего дела 150 тыс. руб. известно, что если на приобретение нового оборудования затратить *x* тыс. руб., а на зарплату вновь принятых работников *y* тыс. руб., то прирост объёма продукции составит . Как следует распределить выделенные денежные ресурсы, чтобы прирост объёма продукции был максимальным?»

Метод проекции градиента не лишен особенностей и специфики применения. Тем не менее, использование данного алгоритма эффективно по отношению к задачам выпуклого программирования, где система условий выступает в виде достаточно простого выпуклого множества.

Среди множества современных языков программирования, одним из наиболее удобных для научных вычислений, является интерпретируемый язык Python. Язык близок с MATLAB и хорош для программирования математических вычислений. К тому же Python умеет работать с такими языками как Fortran, C и C++, которые уже широко используются в научных расчетах. Имеется большое количество модулей, написанных на Python, которые упрощают процесс вычисления (NumPy, SymPy [4] и другие).

Сочетание математического аппарата и современных информационных технологий позволяет с достаточной точностью и быстротой решать задачи на оптимизацию не только в экономике, но и в других областях.

Список литературных источников

1. И.В. Бейко. Методы и алгоритмы решения задач оптимизации. Бейко И.В., Бублик Б.Н., Зинько П.Н. – К.: Вища школа. Головное издательство, 1983 – 512 с.
2. Интернет-учебник по курсу «Методы оптимизации» http://kek.ksu.ru/EOS/MO/uchebnik.asp
3. Статья «Программирование и научные вычисления на языке Python». http://ru.wikiversity.org/wiki/
4. Документация SymPy http://sympy.org/en/index.html

ЗАКОН БЕНФОРДА И АТРИБУЦИЯ ТЕКСТОВ

УДК: 51-78, 519.234.3, 519.257, 81-139

Зенков А. В. к.ф.-м.н.доцент
кафедра моделирования управляемых систем
Уральский федеральный университет, ВШЭМ
Казанцев М.В. студент
кафедра моделирования управляемых систем
Уральский федеральный университет, ВШЭМ

Аннотация. Исследовано распределение первой значащей цифры в числительных связных текстов. Обнаружено, что закон Бенфорда приближённо выполняется для них. Отклонения от закона Бенфорда являются статистически устойчивыми авторскими особенностями, позволяющими при некоторых условиях различить части текста с разным авторством.

Ключевые слова: закон Бенфорда, статистическая проверка гипотез, критерий Манна-Уитни.

Abstract. The distribution of the first significant digit in numerals of connected texts is considered. Benford's law is found to hold approximately for them. Deviations from Benford's law are statistically significant author peculiarities that allow, under certain conditions, to distinguish between parts of the text with a different authorship.

Keywords: Benford’s law, Statistical hypothesis testing, Mann – Whitney U-test.

Введение

Своеобразное проявление Закона больших чисел – известный уже более ста лет закон Бенфорда [1] – в последние десятилетия из статистического курьёза превращается в полезное средство анализа данных. Этот закон описывает вероятность появления определённой первой значащей цифры в разнообразных распределениях величин, взятых из реальной жизни. Вопреки здравому предположению о том, что частоты появления любой первой значащей цифры должны быть равными, для многих массивов данных в качестве первой значащей цифры чаще других встречается единица! Согласно закону Бенфорда вероятность появления цифры d в качестве первой значащей