Санкт-Петербургский Политехнический
Университет
Петра Великого

Институт прикладной математики и механики

Кафедра "Прикладная математика"

Отчёт
Курсовая работа
по дисциплине
"Методы оптимизации"

Выполнил студент:
Мальцов Д.
группа: 3630102/70401

Проверил:
к.ф-м.н.
Родионова Е.А.

Санкт-Петербург
2020 г.

Содержание

Стр.

1 Введение

В данной курсовой работе проводиться сравнительный анализ метода условного градиента и методоа случайного поиска.

2 Постановка задачи минимизации функции

2.1 Определения

Дадим основные определения:

Опр. Функция (f(x)) - это соответствие между элементами двух множеств, установленное по такому правилу, что каждому элементу первого множества соответствует один и только один элемент второго множества.

Опр. Градиент $\text{grad}(\phi) = \nabla\phi = \frac{\partial\phi}{\partial x_1}\overline{e_{x_1}} + \frac{\partial\phi}{\partial x_2}\overline{e_{x_2}} + ... + \frac{\partial\phi}{\partial x_n}\overline{e_{x_n}}$. Т.о. характеристика, показывающая направление наискорейшего возрастания некоторой величины, значение которой меняется от одной точки пространства к другой и называется градиентом.

Опр. Точка глобального экстремума $\overline{x} \in X \subset R^n$ - точка глобального экстремума, если f($\overline{x}$) <= f(x), $\forall$ x $\in$ X(в случае минимума) или если f($\overline{x}$) >= f(x), $\forall$ x $\in$ X(в случае максимума).
Опр. Точка локального экстремума $\overline{x} \in U \subset X \subset R^n$ - точка глобального экстремума, если f($\overline{x}$) <= f(x)(в случае минимума), $\forall$ x $\in$ X или f($\overline{x}$) >= f(x), $\forall$ x $\in$ X(в случае максимума).

2.2 Задача минимизации функции

Задача минимизации фунции заключается в нахождении точки локального или глобального минимума функции. Задача в данной работе заключается в нахождении условного экстремума(экстремум, который удовлетворяет дополнительным условиям наложенным на функцию).

3 Метод условного градиента[2]

Метод условного градиента уместно использовать в присутствии дополнительных ограничений в виде линейных функций.

3.1 Постановка задачи

1)f($x_1, x_2, ..., x_n$) -> min(max)
2)$g_i(x_1, x_2, ..., x_n)$ <= 0, i = $\overline{1, m}$.(2)
3)$x_1, x_2, ..., x_n$ >= 0.(3)
4)$\overline{\xi}$ = $(x_1^0, x_2^0, ..., x_n^0)$ - начальная точка приближения к экстремуму функции. 5)ϵ - точность нахождения экстремума.

3.2 Алгоритм метода

Расспишем алгоритм по шагам:
1)Нахождение градиента функции.

2)Составляем вспомогательную задачу:

$$Z = x_1 * \frac{\partial f}{\partial x_1}|_{\overline{\xi^k}}, x_2 * \frac{\partial f}{\partial x_2}|_{\overline{\xi^k}}, ..., x_n * \frac{\partial f}{\partial x_n}|_{\overline{\xi^k}}$$

3)Решаем задачу линейного программирования(в качестве метода решения будем использовать симплекс метод[3]):

$$\begin{gathered} \text{Z -> min(max)} \\ g_i(x_1, x_2, ..., x_n) <= 0, \text{ i} = \overline{1, m}. \\ x_1, x_2, ..., x_n >= 0. \end{gathered}$$

Пусть решение задачи ЛП следущее: $\overline{Z^k} = (z_1^k, z_2^k, ..., z_n^k)$, где z_j^k - значение переменных x_j на k-м шаге алгоритма.
4)Найдём новую точку приближения к экстремуму:
$\overline{\xi^{k+1}} = \overline{\xi^k} + \alpha(\overline{Z^k} - \overline{\xi^k})$
5)Нахождение α:
$x_i^{k+1} = x_i^k + \alpha(z_i^k - x_i^k)$, i = $\overline{1, n}$
Подставим x_i^{k+1} в функцию f.
Из уравнения $\frac{\partial f}{\partial \alpha} = 0$ найдем значение α(градиентным методом наискорейшего спуска), если $\alpha > 1$, то принимаем значения $\alpha = 1$.
6)Проверка критерия окончания счета:
$||x^{k+1} - x^k|| < \epsilon$, В случае ложного значения возврат к первому шагу алгоритма, в случае истинного значения экстремум найдем.

3.3 Замечания

1) Условие (3) из постаовки задачи можно ослабить или убрать, для условия (2) можно изменить знак или записывать равенство(кроме линейности функций g_i), в случае написания алгоритма приведения матрицы условий к канонической форме для использования в решении задачи ЛП(что в следствие влечет за собой повышенную сложность написания кода и меньшую скорость работы программы).

3.4 Анализ метода

3.4.1 Скорость работы метода

Основная сложность алгоритма заключается в решении задачи ЛП, причем это делается в другом итерационном алгоритме. Симплекс метод имеет полиномиальную сложность выполнения в среднем случае. Таким образом, временная сложность каждой итерации составляет O(k^p)[1](k, p - показатели полиномиальной сложности).

4 Методы случайного поиска

Существует немало методов случайного поиска, из всего этого множества для исследования выберем 3 метода. Метод возврата при неудачном шаге, метод наилучшей пробы и метод комплексов. Обоснуем данный выбор методов: метод возврата при неудачном шаге является классическим методом случайного поиска, метод наилучшей пробы является улучшением метода возврата при неудачном шаге(выбираются лучшие точки вокруг текущей, при которой значение функции наименьшее), метод комплексов выбран, как сильно отличающийся от двух предыдущих метод. Так как все методы случайного поиска основанны на выборках, то следует выполнять метод порядка ста раз и брать среднее от этих вычислений.

4.1 Метод возврата при неудачном шаге[4]

4.1.1 Постановка задачи

1)f($x_1, x_2, ..., x_n$) -> min
2)$g_i(x_1, x_2, ..., x_n)$ <= 0, i = $\overline{1, m}$.
3)$x_1, x_2, ..., x_n >= 0$.
4)$\overline{\xi}$ = $(x_1^0, x_2^0, ..., x_n^0)$, λ_0 - начальная точка приближения к экстремуму функции и начальная длина шага соответственно.
5)ϵ - точность нахождения экстремума.

4.1.2 Алгоритм метода

1)Задаем начальное значение счетчика числа неудачных попыток k = 1.
2)x^{r+1} = $x^r + \lambda_r \frac{\Psi^r}{\|\Psi^r\|}$, где r - итерационный счётчик, Ψ^r - независимые случайные величины, равномерно распределённые в интервале [-1;1], $\Psi^r \in R^n$.
3)Если $f(x^{r+1}) < f(x^r)$ и x^{r+1} удовлетворяет условиям g_i возвращаемся к 1-у шагу, то x^r = x^{r+1}. Иначе k = k + 1.
4)Если k > K(K = 3n), то: $\lambda_r = \alpha\lambda_r$, $\alpha \in (0, 1)$. И возвращаемся к 1-у шагу.
5)Условие выхода из цикла: $|f(x^{r+1} - f(x^r))| < \epsilon$.

4.1.3 Скорость работы метода

Т.к. в методе не используется сложный математический аппарат , то скорость каждой итерации поряка O(1). Точность вычислений и их скорость зависит от выбора начального приближения, ϵ и λ_0(т.к. это прямо влияет на количество итераций).

4.2 Метод наилучшей пробы[4]

4.2.1 Постановка задачи

1)f($x_1, x_2, ..., x_n$) -> min
2)$g_i(x_1, x_2, ..., x_n)$ <= 0, i = $\overline{1, m}$.
3)$x_1, x_2, ..., x_n >= 0$.
4)$\overline{\xi}$ = $(x_1^0, x_2^0, ..., x_n^0)$, λ_0 - начальная точка приближения к экстремуму функции и начальная длина шага соответственно.
5)ϵ - точность нахождения экстремума.

4.2.2 Алгоритм метода

1)Делается M(внутренний параметр метода) пробных вычислений x^{r+1}, из которых выбирается то, при котором значение функции наименьшее. x^{r+1} = $x^r + \lambda_r \frac{\Psi^r}{\|\Psi^r\|}$, где r - итерационный счётчик, Ψ^r - независимые случайные величины, равномерно распределённые в интервале [-1;1], $\Psi^r \in R^n$.
2)Если $f(x^{r+1}) < f(x^r)$ и x^{r+1} удовлетворяет условиям g_i, то x^r = x^{r+1}, возвращаемся к 1-у шагу.
3)$\lambda_r = \alpha\lambda_r$, $\alpha \in (0, 1)$. И возвращаемся к 1-у шагу.
4)Условие выхода из цикла: $|f(x^{r+1} - f(x^r))| < \epsilon$.

4.2.3 Скорость работы метода

Т.к. в методе не используется сложный математический аппарат , то скорость каждой итерации поряка O(M). Точность вычислений и их скорость зависит от выбора начального приближения, M, ϵ и λ_0(т.к. это прямо влияет на количество итераций).

4.3 Метод комплексов [5]

4.3.1 Постановка задачи

1)f($x_1, x_2, ..., x_n$) -> min
2)$g_i(x_1, x_2, ..., x_n)$ <= 0, i = $\overline{1, m}$.
3)$x_1, x_2, ..., x_n >= 0$.
4)$\overline{\xi}$ = ($x_1^0, x_2^0, ..., x_n^0$), l_0 - начальная точка приближения к экстремуму функции и скаляр определяющий азмеры комплекса соответственно.
5)ϵ - точность нахождения экстремума.

4.3.2 Алгоритм метода

1)Генерируем N = 2 * n вершин комплекса, они должны соотвветствовать свойствам (2) и (3). Генерируем их по следующей формуле: $x_i = x_0 + l_0 * \Psi/\|\Psi\|$, Ψ - независимые величины из равномерного распределения на отрезке [0, 1], i = $\overline{1, N}$, $\Psi \in R^n$. 2)Находим индекс k такой, что f(x_k) = max(f(x_i)), i = $\overline{1, N}$.
3)Находим новую точку $x_k = x_{c_k} + \alpha * (x_{c_k} + x_k)(\alpha = 1.3)$, $x_{c_k} = \frac{1}{N-1} \sum_{i=1, i \neq k}^{N} x_i$. Если точка не удовлетворяет условиям g_i, то уменьшаем значение α в 2 раза и повторяем этот шаг. Если значение функции в новой точке x_k меньше, чем в предыдущей, то уменьшаем значение α в 2 раза и повторяем этот шаг.
4)По завершению 3-го шага, повторяем алгоритм начиная со 2-го шага. Если выполнен критерий выхода, то решением будет являться $x_c = \frac{1}{N} \sum_{i=1}^{N} x_i$.
5)Критерий выхода : $\frac{1}{N} \sqrt{\sum_{i=1}^{N} \|x_i - x_c\|} <= \epsilon, x_c = \frac{1}{N} \sum_{i=1}^{N} x_i$.

4.3.3 Скорость работы метода

Т.к. в методе не используется сложный математический аппарат , а используется проход по всему массиву, то временная сложность каждого шага = O(2 * n). Точность вычислений и их скорость зависит от выбора начального приближения, l, ϵ(т.к. это прямо влияет на количество итераций).

5 Сравнительный анализ

5.1 Тестовые примеры

Возьмём несколько тестовых примеров, на которых оценим работу методов:
1)$f(x) = -2x - 4y + x^2 + 2y^2 -> min$ Ограничения:

$$\begin{cases} 2x + 2y <= 8 \\ 3x - y <= 12 \\ x, y >= 0 \end{cases}$$

x_0 = [0, 0]
ϵ = 0.001
В данном примере ограничения не влияют на минимум функции.
2)$f(x) = -2x - 4y + x^2 + 2y^2 -> min$ Ограничения:

$$\begin{cases} 8x + 2y <= 8 \\ 9x - y <= 12 \\ x, y >= 0 \end{cases}$$

x_0 = [0, 0]
ϵ = 0.001

В данном примере из-за ограничений минимум функции лежит вне допустимой области.

5.1.1 Точные ответы на тестовые примеры

1)x = 1.0000, y = 1.0000

2)x = 0.7500, y = 1.0000

5.2 Ответы на тестовые примеры

Так как при использовании методов случайного поиска используется многократное вычисление(описано в "Методы случайного поиска"), то указано время не одного выполнения метода, а многократного(мы вызываем его 100 раз), а количество итераций берётся средним.

5.2.1 1 -й пример метод условного градиента

Ответ:x = 0.9996, y = 0.9995
Число итераций:14
Время работы:0.291сек

5.2.2 2 -й пример метод условного градиента

Ответ:x = 0.7518, y = 0.9562
Число итераций:140
Время работы:13.392сек

5.2.3 1 -й пример метод возврата при неудачном шаге

Ответ:x = 0.9997, y = 0.9857
Число итераций:40
Время работы:0.212сек

5.2.4 2 -й пример метод возврата при неудачном шаге

Ответ:x = 0.7601, y = 0.9603
Число итераций:78
Время работы:0.313сек

5.2.5 1 -й пример метод наилучшей пробы

Ответ:x = 0.9920, y = 0.9969
Число итераций:20
Время работы:0.326сек

5.2.6 2 -й пример метод наилучшей пробы

Ответ:x = 0.7512, y = 0.9662
Число итераций:240
Время работы:3.184сек

5.2.7 1 -й пример метод комплексов

Ответ:x = 1.0017, y = 1.0061
Число итераций:1390
Время работы:10.612сек

5.2.8 2 -й пример метод комплексов

Ответ:x = 0.7518, y = 0.9627
Число итераций:422
Время работы:3.225сек

5.3 Критерии для сравнения

Для проведения сравнительного анализа были выбраны следующие критерии:

1)Время работы и количество итераций метода(отдельно).

2)Приближение к ожидаемой точности и абсолютная погрешность методов.

3)Значение начальных условий в методе.

4)Сложность программной реализации методов.

5.4 Сравнительный анализ метода условного градиентта и методов случайного поиска

Рассмотрим первый пункт:
По количеству итераций метод условного градиента лучше, а по времени хуже, из-за особенностей, описанных в пунктах "скорость работы метода"(это наблюдается для двух примеров), это наблюдается при сравнениями с методами наилучшей пробы и возврата при неудачном шаге, а метод комплексов проигрывает по скорости всем методам(в 1-м примере), из-за того, что сходимость данного метода быстрее, когда условный минимум не совпадает с глобальным.

Рассмотрим второй пункт:
Видно, что метод условных градиентов в первом примере точно определяет решение до знаков задаваемой точности. А метод возврата при неудачном шаге даёт наихудший результат и при заданной точности имеет погрешность до 2-х знаков после запятой. Методы наилучшей пробы и комплексов в свою очередь дают точность порядка ожидаемой.
Во 2-м примере все методы дают большую погрешность, по сравнению с 1-м примером. Это связано с повышением количества итераций у каждого из методов(кроме метода

комплексов).

У всех реализованных методов случайного поиска есть значительный недостаток перед методом условного градиента. Этот недостаток отображается в 3-м пункте критериев.

Все методы случайного поиска очень сильно зависят от выбора начальной точки и параметров метода. Причины этого следующие:
1)Если начальная точка такая, что условия g_i невыполняются, то метод не найдет решения ,даже приблизительно.
2)При большой удалённости начальной точки от решения, метод может не сойтись или, если сходится к решению, то погрешность увеличивается, как и количество итераций во много раз.

В свою очередь для метода условного градиента, в зависимости от начальной точки изменяется количество итераций и это изменение не значительно, и на скорости работы не сказывается, также важно, чтобы начальная точка была внутри области(она может быть отрицательной, а в методах случайного поиска отрицательное начальное приближение может привести к неверному ответу).

Для пункта 4 оценим сложность написания методов.
Упорядочим методы, от самого сложного, к самому простому: метод условного градиента -> метод комплексов -> метод наилучшей пробы -> метод возврата при неудачном шаге.

Обоснуем данное утверждение. 1)Метод условного градиента является самым сложным из-за необходимости дополнительно реализовать метод для решения задачи ЛП и метод для решения задачи одномерной минимизации.2)Метод комплексов нуждается не только в выборке равномерного распределения, но и в построении комплекса и последующих манипуляциях над ним.3) Метод наилучшей пробы является значительно проще 2-х предыдущих, а из-за дополнительного условия для поиска точки, в которой будет минимальное значение функции он сложнее, чем метод возврата при неудачном шаге.

При том, метод условного градиента можно модернизировать, что было описано в пункте "скорость работы метода"(и это повысило бы сложность реализации).

6 Вывод

Сравнив данные методы можно утверждать, что лучше использовать методы случайного поиска, когда известно примерное местонахождение точки и важна скорость работы. Если же пользователю неизвестно примерное местонахождение точки минимума и ему важна точность результата, то наилучшим выбором среди этих методов будет метод условного градиента.

7 Литература

1) Simplex алгоритм - Simplex algorithm // https://ru.qwe.wiki/ URL: https://ru.qwe.wiki/wiki/Simplex_algorit
(дата обращения: 13.05.2020).

2) https://studfile.net URL: https://studfile.net/preview/488257/ (дата обращения: 13.05.2020).

3) Петухов Л. В. Методы оптимизации. Задачи выпуклого программирования: учебное пособие / Л. В. Петухов, Г. А. Серегин, Е. А. Родионова — Санкт-Петербург: Изд-во Политехн. ун-та, 2014 – 98 с.

4) http://bigor.bmstu.ru URL: http://bigor.bmstu.ru/?cnt/?doc=MO/ch0801.mod/?cou=MO/base.cou (дата обращения: 13.05.2020).

5)https://hub.exponenta.ru URL: https://hub.exponenta.ru/post/postanovka-zadachi-optimizatsii-i-chislennye-metody-ee-resheniya356#3_3_2 (дата обращения: 13.05.2020).