**Федеральное государственное образовательное**

**бюджетное учреждение**

**высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ**

**ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ**

**ФЕДЕРАЦИИ»**

**(Финансовый университет)**

**Факультет**

**информационных технологий и анализа больших данных**

**Направление «Прикладная математика и информатика»**

**Домашнее задание № 2**

«**Методы одномерной оптимизации**»

Студенты группы ПМ19-3:

Захаров Д. В.

Исмоилова М. В.

Константинов К. Л.

Мосолова К. Д.

Самофалова Т. А.

Руководитель:

Аксенов Дмитрий Андреевич

**Москва 2022**

Оглавление.

1. Физическая модель (постановка задачи)
2. Математическая модель
3. Алгоритмы
   1. Алгоритм 1
      1. Описание входных данных
      2. Описание алгоритма решения
      3. Описание выходных данных
   2. Алгоритм 2
      1. Описание входных данных
      2. Описание алгоритма решения
      3. Описание выходных данных
   3. Алгоритм 3
      1. Описание входных данных
      2. Описание алгоритма решения
      3. Описание выходных данных
4. Варианты использования системы
   1. ВИ 1
   2. ВИ 2
5. Архитектура решения
   1. Функции считывания информации
   2. Функции обработки информации
   3. Функции вывода информации
6. Тестирование
7. Выводы и заключение
8. **Постановка задачи (физическая модель)**

Финтех стартапу «Рога и копыта» необходимо определять качество выбранной аппроксимирующей функции для предсказания цены акций. Стартап имеет датасеты с несколькими факторами, влияющими на котировки, за определенный временной период.

Эту задачу можно решать через сравнения минимумов и максимумов выбранной функции на отрезке с минимумами и максимумами реальных данных.

Для решения задачи поиска экстремумов функции было решено выбрать следующие математические алгоритмы:

• метод золотого сечения;

• метод парабол;

• комбинированный метод Брента;

• Алгоритм неточной одномерной минимизации (Алгоритм Бройдена — Флетчера — Гольдфарба — Шанно)

Для определения оптимального алгоритма было решено упростить задачу и исключить факторы, влияющие на котировки, использовать только зависимость цены акции от времени. Таким образом, имеем задачу одномерной оптимизации.

Под каждый обозначенный выше алгоритм была написана отдельная функция. (их описание см. в разделе 5). Также была сделана визуализация метода парабол и проведена проверка на реальных данных (были замерены отклонения минимумов и максимумов, найденных алгоритмами от реальных значений в данных и найдено время работы каждого алгоритма). В следующих разделах отчета каждое действие рассмотрено более подробно. Также предложены варианты развития данного проекта.

1. **Математическая модель**

Пусть f(x) – унимодальная функция на отрезке [a, b]. Требуется найти приближенное значение точки минимума (максимума) этой функции x\* ≈ [a, b] с заданной точностью ε > 0.

1. **Алгоритмы**
   1. **Алгоритм 1**

Метод золотого сечения.

В основе метода лежит принцип деления отрезка в пропорциях золотого сечения.

* + 1. **Описание входных данных**

Обязательные параметры:

а) Функция в аналитическом виде;

б) Границы области оптимизации;

Необязательные параметры:

в) Тип экстремума (по умолчанию: min);

г) Точность оптимизации по аргументу (по умолчанию: 10-5);

д) Максимальное число итераций (по умолчанию: 500);

е) Флаг «вывод промежуточных результатов» (по умолчанию: False) – при установке TRUE выводит полученные значения результатов на каждой итерации);

ё) Флаг «запись промежуточных результатов в датасет» (по умолчанию: False) – при установке TRUE записывает номер итерации и полученные значения результатов на каждой итерации в pandas dataset).

* + 1. **Описание алгоритма решения**

1. По заданным пользователем границам области оптимизации рассчитываются начальные точки деления: , , где .

И значения в них целевой функции:

2.

Если (для поиска максимума: ), то

Иначе:

3. Если , то . Иначе: возврат к шагу 1.

* + 1. **Описание выходных данных**

а) Найденное значение координаты точки экстремума;

б) Значение функции в точке экстремума;

в) Отчет о работе алгоритма (например флаг: 0 – найдено значение с заданной точностью; 1 – достигнуто максимальное количество итераций; 2 – выполнено с ошибкой).

* 1. **Алгоритм 2**

Метод парабол.

В рамках данного метода целевая функция приближается параболой, соответственно, в качестве приближения к точке минимума выбирается вершина параболы.

* + 1. **Описание входных данных**

Обязательные параметры:

а) Функция в аналитическом виде;

б) Границы области оптимизации;

в) Тип экстремума;

Необязательные параметры:

г) Точность оптимизации по аргументу (по умолчанию: 10-5);

д) Максимальное число итераций (по умолчанию: 500);

е) Флаг «вывод промежуточных результатов» (по умолчанию: False) – при установке TRUE выводит полученные значения результатов на каждой итерации);

ё) Флаг «запись промежуточных результатов в датасет» (по умолчанию: False) – при установке TRUE записывает номер итерации и полученные значения результатов на каждой итерации в pandas dataset).

* + 1. **Описание алгоритма решения**

1. Выбираются три точки из отрезка [a, b] так, чтобы выполнялись неравенства: .

В общем случае: .

2. Рассчитываются коэффициенты параболы:

.

3. Точка минимума: .

4. Если заданная точность не достигнута, пункты 1-3 повторяются, в качестве

* + 1. **Описание выходных данных**

а) Найденное значение координаты точки экстремума;

б) Значение функции в точке экстремума;

в) Отчет о работе алгоритма (например флаг: 0 – найдено значение с заданной точностью; 1 – достигнуто максимальное количество итераций; 2 – выполнено с ошибкой).

* 1. **Алгоритм 3**

Комбинированный метод Брента.

Метод Брента комбинирует в себе метод парабол и метод золотого сечения, совмещая в себе высокую скорость сходимость первого и стабильность второго метода.

* + 1. **Описание входных данных**

Обязательные параметры:

а) Функция в аналитическом виде;

б) Границы области оптимизации;

Необязательные параметры:

в) Точность оптимизации по аргументу (по умолчанию: 10-5);

г) Максимальное число итераций (по умолчанию: 500);

д) Флаг «вывод промежуточных результатов» (по умолчанию: False) – при установке TRUE выводит полученные значения результатов на каждой итерации);

е) Флаг «запись промежуточных результатов в датасет» (по умолчанию: False) – при установке TRUE записывает номер итерации и полученные значения результатов на каждой итерации в pandas dataset).

* + 1. **Описание алгоритма решения**

1. Задаются точки , где .

2.

3. Пока не достигнута точность сходимости:

g = e/2

e = d

Если точки x, w, v – разные, то по трем точкам находим параболу рассчитываем вершину параболы u

Если u принадлежит |a + e, b - e| и |u - x| < g/2:

d = |u-x|

иначе:

если x < (b - a)/2:

u = x – r(x - a)

d = x – a

если |u - x| < e то

u = x + sign(u-x)e

fu = f(u)

если fu<=f(x), то :

если u>=x, то:

a = x

иначе: c = x, v=w, w=x, x=u

иначе: если u>= x то:

c = u

иначе: a = u

если fu<=f(w) или w=x, то:

v = w, w = u

иначе если fu<=f(v) или v=x или v=w:

v=u

* + 1. **Описание выходных данных**

а) Найденное значение координаты точки экстремума;

б) Значение функции в точке экстремума;

в) Отчет о работе алгоритма (например флаг: 0 – найдено значение с заданной точностью; 1 – достигнуто максимальное количество итераций; 2 – выполнено с ошибкой).

* 1. **Алгоритм 4**

Алгоритм Бройдена-Флетчера\_Гольдфарба-Шанно.

Один из наиболее широко применяемых квазиньютоновских методов. Гессиан функции оценивается приближенно, исходя из сделанных до этого шагов.

* + 1. **Описание входных данных**

Обязательные параметры:

а) Функция в аналитическом виде;

б) Начальная точка;

в) Тип экстремума;

Необязательные параметры:

г) Параметр для первого условия Вольфе (по умолчанию: 10-4);

д) Параметр для второго условия Вольфе (по умолчанию: 0.1);

е) Максимально возможное значение аргумента функции (по умолчанию 100);

ё) порог выхода по длине интервала поиска (по умолчанию: 10-8);

ж) Максимальное число итераций (по умолчанию: 500);

з) Флаг «вывод промежуточных результатов» (по умолчанию: False) – при установке TRUE выводит полученные значения результатов на каждой итерации);

и) Флаг «запись промежуточных результатов в датасет» (по умолчанию: False) – при установке TRUE записывает номер итерации и полученные значения результатов на каждой итерации в pandas dataset).

* + 1. **Описание алгоритма решения**

1. Определяем начальное приближение H0 как единичную матрицу

2. Вычисляем pk = -Hk fk

3. Находим α с помощью линейного поиска. α должно удолетворять условиям Вольфе:

4. Вычисляются

5. Обновляется гессиан функции по формуле:

Где

* + 1. **Описание выходных данных**

а) Найденное значение координаты точки экстремума;

б) Значение функции в точке экстремума;

г) Отчет о работе алгоритма (например флаг: 0 – точка удовлетворяющая условию Вольфе найдена; 1 – точка удовлетворяющая условию Вольфе найдена с заданной точностью; 2 – достигнуто максимальное количество итераций; 3 – Достигнуто ограничение на максимально возможное значение аргумента; 4 – выполнено с ошибкой).

1. **Варианты использования системы**

Для вызова реализованных методов предусмотрено 4 функции:

1. golden\_ration

2. parabola\_method

3. BrantMethod

4. bfgs

Для работы пользователю следует передать соответствующие аргументы в функцию того метода, который он хочет использовать (подробнее см п. 5 отчета)

1. **Архитектура решения**

В библиотеке представлены 4 основные функции:

1. golden\_ration

2. parabola\_method

3. BrantMethod

4. bfgs

А также 2 функции, которые используются основными функциями:

1. get\_sympy\_subplots - позволяет встроить sympy.plolt в matplotlib

2. center\_point - находит вершину параболы, построенной по x, w, v

Основные функции:

5. 1. golden\_ration - метод золотого сечения

*Аргументы функции:*

- область поиска : tuple или list

[a, b], где (a<b) – интервал, в пределах которого максимальное и минимальная площадь поиска.

- точность: float, необязательно

x критерий остановки допуска.

- maxiter : int

Максимальное количество итераций для выполнения.

- interim\_results : bool, необязательно

Если значение True, выведите промежуточные результаты.

- dataset\_rec : загрузка, необязательно

Если верно, то запись в pandas.Промежуточные результаты фрейма данных.

*Выход:*

Словарь с ключами:

- Ключ 'point' – аргумент функции, при котором функция принимает минимальное (максимальное) значение;

- Ключ 'value\_func' – значение функции в точке 'point'

- Ключ 'report' – отчёт о работе функции, при каком условии получен ответ

- Ключ 'interim\_results\_dataset' – объект pandas.DataFrame с промежуточными результатами (при - соответствующем флаге во время вызова, иначе – None)

2. parabola\_method – метод парабол

*Аргументы функции:*

Обязательные:

- func - функция в аналитическом виде

- limits - границы области оптимизации

Необязательные:

- accuracy - точность оптимизации по аргументу (по умолчанию: 10^-5);

- max\_iterations - максимальное количество итераций (по умолчанию: 500;

- intermediate\_results - флаг "вывод промежуточных результатов" (по умолчанию: False);

- intermediate\_writing - флаг "запись промежуточных результатов в набор данных" (по умолчанию: False);

*Выход:*

- x - значение точки x

- y - значение точки y

- str - строка с информацией о причине окончания работы (Достигнуть лимит итераций или уровень точности)

- pd.DataFrame with information about steps (if intermediate\_writing = True);

3. BrantMethod: - комбинированный метод Брента

Обязательные:

- func - функция в аналитическом виде

- limits - границы области оптимизации

Необязательные:

- accuracy - точность оптимизации по аргументу (по умолчанию: 10^-5);

- max\_iterations - максимальное количество итераций (по умолчанию: 500;

- intermediate\_results - флаг "вывод промежуточных результатов" (по умолчанию: False);

- intermediate\_writing - флаг "запись промежуточных результатов в набор данных" (по умолчанию: False);

*Выход:*

- x - значение точки x

- y - значение точки y

- str - строка с информацией о причине окончания работы (Достигнуть лимит итераций или уровень точности)

- pd.DataFrame with information about steps (if intermediate\_writing = True);

4. bfgs - Алгоритм Бройдена — Флетчера — Гольдфарба — Шанно:

- func : вызываемая функция (callable) ``f(x)``

Целевая функция, подлежащая минимизации или максимизации.

- diff\_func : вызываемая функция (callable) ``f'(x)``

Градиент целевой функции.

- x0 : float, int

Отправная точка.

extreme\_type : str, optional

Максимизация или минимизация.

accuracy : float, optional

Норма градиента (abs) в x0 должна быть меньше, чем "точность", прежде чем успешно

прекращение.

maxarg : float, int, optional

Максимальное значение функции аргумента.

firstW : int or ndarray, необязательно

Параметр для правила условий Armijo.

secondW : callable, необязательно

Параметр для правила условия кривизны.

maxiter : int, необязательно

Максимальное количество выполняемых итераций.

interim\_results : bool, необязательно

Если значение True, выведите промежуточные результаты.

dataset\_rec : bool, необязательно

Если верно, то запись в pandas.Промежуточные результаты фрейма данных.

*Выход:*

Словарь с ключами:

- Ключ 'point' – аргумент функции, при котором функция принимает минимальное (максимальное) значение;

- Ключ 'value\_func' – значение функции в точке 'point'

- Ключ 'report' – отчёт о работе функции, при каком условии получен ответ

- Ключ 'interim\_results\_dataset' – объект pandas.DataFrame с промежуточными результатами (при - соответствующем флаге во время вызова, иначе – None)

1. **Тестирование**

Исходя из полученных результатов, можем сделать выводы, что наиболее точными являются метод парабол и метод Бранта. Учитывая скорость выполнения алгоритма, предпочтительнее метод Бранта. Хоть и BFGS – можно назвать наиболее быстрым, точность вычислений оставляет нам возможность для выбора наиболее привлекательного.

1. **Заключение**

Исходя из всего вышеперечисленного, получили, что все описанные методы так или иначе решают поставленную задачу. При этом, исходя из тестов, наилучшим образом показал себя метод Брента.

Несмотря на более медленную скорость, чем у алгоритма Бройдена — Флетчера — Гольдфарба — Шанно, он показал лучшую точность. Поставленная задача не требует высокой скорости работы алгоритма, при этом важна точность при сравнении значений.

Дальнейшим развитием полученного результата будет реализация многомерного варианта выбранного алгоритма. Именно это и будет решать поставленную задачу.