Курс: Методы оптимизации в машинном обучении, осень 2014

Задание 1. Одномерная оптимизация

Начало выполнения задания: 29 сентября
Срок сдачи: **13 октября (понедельник), 23:59.**
Среда для выполнения задания – MATLAB или PYTHON.

Содержание

Задание состоит из двух вариантов. Студент может выбрать любой вариант для выполнения.

Вариант 1

1. Реализовать алгоритмы одномерной минимизации функции без производной: метод золотого сечения, метод парабол и комбинированный метод Брента;
2. Протестировать реализованные алгоритмы на следующем наборе задач оптимизации:
   - $f(x) = -5x^5 + 4x^4 - 12x^3 + 11x^2 - 2x + 1$ на интервале $[-0.5, 0.5]$;
   - $f(x) = \log^2(x-2) + \log^2(10-x) - x^{0.2}$ на интервале $[6, 9.9]$;
   - $f(x) = -3x \sin 0.75x + \exp(-2x)$ на интервале $[0, 2\pi]$;
   - $f(x) = \exp(3x) + 5\exp(-2x)$ на интервале $[0, 1]$;
   - $f(x) = 0.2x \log x + (x - 2.3)^2$ на интервале $[0.5, 2.5]$;
3. Протестировать реализованные алгоритмы для задач минимизации многомодальных функций, например, на различных полиномах. Могут ли метод золотого сечения/Брента не найти локальный минимум многомодальной функции?
4. Реализовать метод кубических аппроксимаций (по значениям функции и производной в двух точках) и комбинированный метод Брента с производной, сравнить их работу с методами оптимизации без производной. Привести пример плохой работы метода кубических аппроксимаций;
5. Написать отчёт в формате PDF с описанием всех проведённых исследований. Данный отчёт должен содержать, в частности, количество обращений к оракулу при всех запусках итерационных процессов оптимизации.

Вариант 2

1. Придумать и реализовать (не пользуясь источниками в Интернете) алгоритм неточной одномерной оптимизации функции $\phi(\alpha)$ для поиска точки, удовлетворяющей сильным условиям Вольфа:
   - $\phi(\alpha) \le \phi(0) + c_1 \alpha \phi'(0)$;
   - $\phi'(\alpha) \ge c_2 \phi'(0)$;

   Здесь $\alpha > 0$, $\phi'(0) < 0$, $0 < c_1 \le c_2 < 1$.
2. Протестировать реализованный алгоритм на следующем наборе задач оптимизации:
   - $\phi(\alpha) = -\frac{\alpha}{\alpha^2 + \beta}$ для $\beta = 2$;
   - $\phi(\alpha) = (\alpha + \beta)^5 - 2(\alpha + \beta)^4$ для $\beta = 0.004$;

- $\phi(\alpha) = \phi_0(\alpha) + \frac{2(1-\beta)}{l\pi}\sin\left(\frac{l\pi}{2}\alpha\right)$, где

$$\phi_0(\alpha) = \begin{cases} 1-\alpha, & \alpha \le 1-\beta, \\ \alpha - 1, & \alpha \ge 1+\beta, \\ \frac{1}{2\beta}(\alpha-1)^2 + \frac{\beta}{2}, & \alpha \in [1-\beta, 1+\beta]. \end{cases}$$

  Здесь $\beta = 0.01$, $l = 39$;
- $\phi(\alpha) = \gamma(\beta_1)\sqrt{(1-\alpha)^2+\beta_2^2} + \gamma(\beta_2)\sqrt{\alpha^2+\beta_1^2}$, где $\gamma(\beta) = \sqrt{1+\beta^2} - \beta$, для следующих трёх пар (β_1, β_2): $(0.001, 0.001)$, $(0.01, 0.001)$, $(0.001, 0.01)$;
- Неограниченная снизу функция $\phi(\alpha)$.

3. Написать отчёт в формате PDF с описанием всех проведённых исследований. Данный отчёт должен содержать, в частности, подробное описание предлагаемого метода оптимизации, а также количество обращений к оракулу при всех запусках итерационных процессов оптимизации.

Оформление задания

Выполненное задание следует отправить письмом по адресу *bayesml@gmail.com* с заголовком письма

«[MOMO14] Задание 1, Фамилия Имя, Вариант №».

Убедительная просьба присылать выполненное задание только один раз с окончательным вариантом. Также убедительная просьба строго придерживаться заданных ниже прототипов реализуемых функций (для проверки задания используются, в том числе, автоматические процедуры, которые являются чувствительными к неверным прототипам).

Присланный вариант задания должен содержать в себе:

- Текстовый файл в формате PDF с указанием ФИО и номера варианта, содержащий описание всех проведённых исследований.
- Все исходные коды с необходимыми комментариями.

MATLAB

Таблица 1: Метод золотого сечения

[x_min, f_min, exitflag, results] = min_golden(func, bounds, param_name1, param_value1, ...)
ВХОД *func* – указатель на оптимизируемую функцию; *bounds* – границы интервала оптимизации, вектор типа double длины 2; (*param_name*, *param_value*) – необязательные параметры, следующие названия и значения возможны: 'tol_x' – точность оптимизации по аргументу, число, по умолчанию = 1e-5; 'max_iter' – максимальное число итераций, число, по умолчанию = 500; 'display' – режим отображения, true или false, если true, то отображаются номер итерации, текущее значение функции, аргумента, текущая точность и др. показатели, по умолчанию = false;
ВЫХОД *x_min* – найденное значение минимума, число; *f_min* – значение функции в точке минимума, число; *exitflag* – результат оптимизации, число: '0' – значение минимума найдено с заданной точностью tol_x; '1' – достигнуто максимальное число итераций; *results* – подробные результаты оптимизации, структура со следующими полями: 'num_oracle' – количество обращений к оракулу;

В структуру results можно включать любые дополнительные параметры. Прототипы функций min_parabolic для метода парабол, min_cubic для кубических аппроксимаций и min_brent для метода Брента выглядят аналогично. В методе Брента добавляется параметр 'use_gradient' с возможными значениями true и false для учета

случая оптимизации с производной и без. При отображении в методе Брента необходимо указывать способ выбора очередной точки на каждой итерации (golden/parabolic или bisection/parabolic).

Таблица 2: Метод неточной одномерной минимизации

[alpha_min, phi_min, exitflag, results] = min_wolfe(func, alpha0, param_name1, param_value1, ...)
ВХОД $func$ – указатель на оптимизируемую функцию; $alpha0$ – начальная точка, число; $(param\_name, param\_value)$ – необязательные параметры, следующие названия и значения возможны: 'c1' – параметр для первого условия Вольфа, число, по умолчанию = 1e-4; 'c2' – параметр для второго условия Вольфа, число, по умолчанию = 0.1; 'alpha_max' – максимальное возможное значение alpha, по умолчанию = 100; 'tol_interval' – порог выхода по длине интервала поиска, по умолчанию = 1e-8; 'max_iter' – максимальное число итераций, число, по умолчанию = 500; 'display' – режим отображения, true или false, если true, то отображаются номер итерации, текущее значение функции, аргумента и др. показатели, по умолчанию = false;
ВЫХОД $alpha\_min$ – найденное значение минимума, число; phi_min – значение функции в точке минимума, число; $exitflag$ – результат оптимизации, число: '0' – точка, удовлетворяющая условиям Вольфа, найдена; '1' – точка, удовлетворяющая условиям Вольфа, найдена с точностью tol_interval; '2' – достигнуто максимальное число итераций; '3' – достигнуто значение alpha_max, функция не ограничена снизу, либо в интервале [0, alpha_max] нет требуемой точки; $results$ – подробные результаты оптимизации, структура со следующими полями: 'num_oracle' – количество обращений к оракулу;

Можно добавлять свои входные параметры, контролирующие оптимизационный процесс. Также можно включать дополнительные параметры в структуру results.

PYTHON

Все коды реализаций методов оптимизации должны располагаться в одном модуле под названием *minfuncs*. Прототипы функций min_parabolic для метода парабол, min_cubic для кубических аппроксимаций и min_brent для метода Брента выглядят аналогично прототипу в табл. 3. В методе Брента добавляется параметр 'use_gradient' с возможными значениями 0 и 1 для учета случая оптимизации с производной и без. При отображении в методе Брента необходимо указывать способ выбора очередной точки на каждой итерации (golden/parabolic или bisection/parabolic).

Таблица 3: Метод золотого сечения

x_min, f_min, exitflag, num_oracle = min_golden(func, bounds, tol_x = 1e-5, max_iter = 500, display = 0)
ВХОД *func* – оптимизируемая функция; *bounds* – границы интервала оптимизации, кортеж (x_min, x_max); *tol_x* – точность оптимизации по аргументу; *max_iter* – максимальное число итераций; *display* – режим отображения, равен 0 или 1, если 1, то отображаются результаты на промежуточных итерациях: номер итерации, значение функции, длина текущего интервала поиска и проч.;
ВЫХОД *x_min* – найденное значение минимума; *f_min* – значение функции в точке минимума; *exitflag* – результат оптимизации, число: '0' – значение минимума найдено с заданной точностью tol_x; '1' – достигнуто максимальное число итераций; *num_oracle* – количество обращений к оракулу;

Таблица 4: Метод неточной одномерной минимизации

alpha_min, phi_min, exitflag, num_oracle = min_wolfe(func, alpha0, c1 = 1e-4, c2 = 0.1, alpha_max = 100, tol_interval = 1e-8, max_iter = 500, display = 0)
ВХОД *func* – оптимизируемая функция; *alpha*0 – начальная точка, число; *c*1 – параметр для первого условия Вольфа; *c*2 – параметр для второго условия Вольфа; *alpha_max* – максимальное возможное значение alpha; *tol_interval* – порог выхода по длине интервала поиска; *max_iter* – максимальное число итераций; *display* – режим отображения, 0 или 1, если 1, то отображаются номер итерации, текущее значение функции, аргумента и др. показатели, по умолчанию = false;
ВЫХОД *alpha_min* – найденное значение минимума; *phi_min* – значение функции в точке минимума; *exitflag* – результат оптимизации, число: '0' – точка, удовлетворяющая условиям Вольфа, найдена; '1' – точка, удовлетворяющая условиям Вольфа, найдена с точностью tol_interval; '2' – достигнуто максимальное число итераций; '3' – достигнуто значение alpha_max, функция не ограничена снизу, либо в интервале [0, alpha_max] нет требуемой точки; *num_oracle* – количество обращений к оракулу;