

Практическое задание 1, вариант 2. Неточная одномерная оптимизация.

Начало выполнения задания: 14 сентября 2016 г.

Срок сдачи: 28 сентября (среда), 23:59.

Среда для выполнения задания: Python 3.

1 Формулировка задания

1. Придумать и реализовать алгоритм неточной одномерной оптимизации функции $\phi : \mathbf{R}_+ \rightarrow \mathbf{R}$ для поиска точки $\alpha > 0$, удовлетворяющей сильным условиям Вульфа:

$$\begin{aligned}\phi(\alpha) &\leq \phi(0) + c_1\alpha\phi'(0), \\ |\phi'(\alpha)| &\leq c_2|\phi'(0)|,\end{aligned}$$

где $\phi'(0) < 0$, $0 < c_1 < c_2 < 1$.

Большая просьба придумывать алгоритм самостоятельно и не пользоваться уже готовыми решениями. Тем не менее, если у Вас возникли сложности, то нет ничего страшного в том, чтобы почерпнуть пару идей в литературе.

2. Протестировать реализованный алгоритм на следующем наборе функций:

(a) $\phi_1(\alpha) := -\alpha/(\alpha^2 + \beta)$ при $\beta = 2$.

(b) $\phi_2(\alpha) := (\alpha + \beta)^5 - 2(\alpha + \beta)^4$ при $\beta = 0.004$.

(c) $\phi_3(\alpha) := \phi_0(\alpha) + 2[(1 - \beta)/(l\pi)] \sin(l\pi\alpha/2)$ при $\beta = 0.01$, $l = 39$, где

$$\phi_0(\alpha) := \begin{cases} 1 - \alpha, & \alpha < 1 - \beta, \\ \alpha - 1, & \alpha > 1 + \beta, \\ [1/(2\beta)](\alpha - 1)^2 + \beta/2, & \alpha \in [1 - \beta, 1 + \beta]. \end{cases}$$

(d) $\phi_4(\alpha) := \gamma(\beta_1)((1 - \alpha)^2 + \beta_2^2)^{1/2} + \gamma(\beta_2)(\alpha^2 + \beta_1^2)^{1/2}$, где $\gamma(\beta) := \sqrt{1 + \beta^2} - \beta$.

i. $\phi_{4a}(\alpha) := \phi_4(\alpha)$ при $\beta_1 = 0.001$, $\beta_2 = 0.001$.

ii. $\phi_{4b}(\alpha) := \phi_4(\alpha)$ при $\beta_1 = 0.01$, $\beta_2 = 0.001$.

iii. $\phi_{4c}(\alpha) := \phi_4(\alpha)$ при $\beta_1 = 0.001$, $\beta_2 = 0.01$.

Сравнить реализованный метод с библиотечным `scipy.optimize.linesearch.scalar_search_wolfe2` для $\alpha_0 = 1$, $c_1 = 10^{-4}$ и двух значений c_2 : 1) $c_2 = 0.1$, 2) $c_2 = 0.9$.

3. Для реализованного алгоритма исследовать, как меняется число вызовов оракула функции ϕ для нахождения требуемой точки в зависимости от значений констант c_1 и c_2 , а также начального приближения α_0 .
4. Написать отчет в формате PDF с описанием всех проведенных исследований.

2 Оформление задания

Результатом выполнения задания являются 1) pdf-отчет с описанием предлагаемого алгоритма и всех проведенных исследований¹ и 2) текстовый файл `optim1d.py`, содержащий исходный код алгоритма нахождения точки, удовлетворяющей сильным условиям Вульфа. Выполненное задание следует отправить письмом по адресу `bayesml@gmail.com` с заголовком

«[ВМК МОМО16] Задание 1 (вариант 2), Фамилия Имя».

Убедительная просьба присылать выполненное задание только один раз с окончательным вариантом.

Отчет должен содержать подробное описание предлагаемого алгоритма вместе с математически строгим доказательством всех необходимых утверждений. Пример возможного утверждения: «если в точке a интервала $[a, b]$ выполнены такие условия, а в точке b такие, то интервал $[a, b]$ обязательно содержит точку, удовлетворяющую сильным условиям Вульфа».

Поскольку проверка реализованных алгоритмов будет осуществляться в полуавтоматическом режиме, все реализованные функции должны строго соответствовать приведенным ниже прототипам и корректно запускаться в Python 3 (а не Python 2!). Проверить наличие всех необходимых функций, а также их соответствие требуемым прототипам можно с помощью специального скрипта `check_submission_v2.py`, выдаваемого вместе с текстом задания.

3 Прототипы функций

1. Метод поиска точки, удовлетворяющей сильным условиям Вульфа:

Модуль:	<code>optim1d</code>
Функция:	<code>search_wolfe(phi, alpha0=1, c1=1e-4, c2=0.9, max_n_evals=100, disp=False)</code>
Параметры:	<code>phi: callable phi(x)</code> Оракул функции ϕ . Принимает: <code>x: float</code> Точка вычисления. Возвращает: <code>f: float</code> Значение функции в точке x . <code>g: float</code> Значение производной в точке x . <code>alpha0: float</code> , опционально Начальное приближение. <code>c1: float</code> , опционально Значение константы c_1 в условиях Вульфа. <code>c2: float</code> , опционально Значение константы c_2 в условиях Вульфа. <code>max_n_evals: int</code> , опционально Максимальное допустимое число вызовов оракула. <code>disp: bool</code> , опционально

¹При этом допускается возможность сделать отчет в IPython-блокноте, а затем сконвертировать его в формат PDF.

	Отображать информацию для отладки (текущая итерация, стадия bracketing/sectioning, тип интерполяции/экстраполяции и пр.) или нет.
Возврат:	alpha: float Найденная точка. status: int Статус выхода, число: 0: найдена точка, удовлетворяющая сильным условиям Вульфа; 1: превышено максимальное возможное число вызовов оракула. n_evals: int Суммарное число выполненных вызовов оракула.

ССЫЛКИ

- [1] Roger Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2013. Гл. 2.
- [2] Jorge Nocedal и Stephen Wright. *Numerical Optimization*. Springer Science & Business Media, 2006. Гл. 3.