Методы оптимизации, ФКН ВШЭ, зима 2017

Практическое задание 3: Негладкая и условная оптимизация.

Срок сдачи: 7 апреля 2017 (23:59).
Язык программирования: Python 3.

1 Алгоритмы

1.1 Субградиентный метод

Субградиентный метод — это метод решения безусловной негладкой задачи оптимизации

$$\min_{x \in \mathbb{R}^n} \phi(x),$$

где $\phi : \mathbb{R}^n \to \mathbb{R}$ — выпуклая функция (не обязательно дифференцируемая).

Итерация субградиентного метода заключается в шаге из текущей точки x_k в направлении (любого) анти-субградиента $\phi'(x_k)$. При этом, поскольку для негладких задач норма субградиента $\|\phi'(x_k)\|_2$ не является информативной, субградиентный метод использует в качестве направления нормированный вектор $\phi'(x_k)/\|\phi'(x_k)\|_2$:

$$x_{k+1} = x_k - \alpha_k \frac{\phi'(x_k)}{\|\phi'(x_k)\|_2}.$$

Для сходимости метода необходимо, чтобы длины шагов α_k убывали к нулю, но не слишком быстро:

$$\alpha_k > 0, \quad \alpha_k \to 0, \quad \sum_{k=0}^{\infty} \alpha_k = \infty.$$

Обычно длины шагов выбирают по правилу $\alpha_k = \alpha/\sqrt{k+1}$, где $\alpha > 0$ — некоторая константа.

Нужно отметить, что последовательность $(x_k)_{k=0}^{\infty}$, построенная субградиентным методом, может не быть релаксационной последовательностью для функции ϕ, т. е. неравенство $\phi(x_{k+1}) \leq \phi(x_k)$ может не выполняться. Поэтому в субградиентном методе в качестве результата работы после N итераций метода вместо точки x_N возвращается точка $y_N := \operatorname{argmin}\{\phi(x) : x \in \{x_0, \ldots, x_N\}\}$ (т. е. из всех пробных точек x_k, построенных методом, выбирается та, в которой значение функции оказалось наименьшим).[1]

1.2 Проксимальный градиентный метод

Проксимальный градиентный метод используется для минимизации *композитных функций*:

$$\phi(x) := f(x) + h(x),$$

где функция $f : \mathbb{R}^n \to \mathbb{R}$ непрерывно дифференцируемая, а функция $h : \mathbb{R}^n \to \mathbb{R}$ выпуклая (не обязательно дифференцируемая) и достаточно простая.

Под *простотой* функции h подразумевается то, что для этой функции возможно эффективно вычислить проксимальное отображение

$$\operatorname{Prox}_h^{\alpha}(x) := \operatorname*{argmin}_{y \in \mathbb{R}^n} \left[\alpha h(y) + \frac{1}{2}\|y - x\|_2^2 \right],$$

[1] Заметим, что в практической реализации метода для вычисления результата y_N сами точки $x_0, \ldots, x_N$ хранить в памяти не нужно.

где $x \in \mathbb{R}^n,\ \alpha > 0$. Например, для $h(x) = \|x\|_1$ проксимальное отображение может быть вычислено аналитически независимо для каждой из координат $1 \le i \le n$:

$$[\mathrm{Prox}^{\alpha}_{\|\cdot\|_1}(x)]_i = \begin{cases} x_i + \alpha, & x_i < -\alpha, \\ 0, & |x_i| \le \alpha, \\ x_i - \alpha, & x_i > \alpha. \end{cases}$$

Итерация проксимального градиентного метода имеет следующий вид:

$$x_{k+1} = \mathrm{Prox}^{\alpha_k}_h(x_k - \alpha_k \nabla f(x_k)),$$

где $\nabla f(x_k)$ — градиент функции f точке x_k, а $\alpha_k > 0$ — (должным образом выбранная) длина шага.

1.2.1 Схема Нестерова для подбора длины шага

Итерация проксимального градиентного метода имеет следующий геометрический смысл. Если градиент функции f удовлетворяет условию Липшица с константой $L_f > 0$, то для любых $x, y \in \mathbb{R}^n$ и $L \ge L_f$ справедлива оценка

$$\phi(y) \le f(x) + \nabla f(x)^T (y - x) + \frac{L}{2}\|y - x\|_2^2 + h(y) =: m_L(y; x), \tag{1}$$

которая является точной в точке $y = x$. Выбрав $x = x_k,\ L = L_k := 1/\alpha_k$, получаем, что итерация проксимального градиентного метода в точности соответствует минимизации функции $m_{L_k}(\cdot; x_k)$:

$$x_{k+1} = \operatorname*{argmin}_{y \in \mathbb{R}^n} m_{L_k}(y; x_k).$$

Если $L_k \ge L_f$, то, согласно оценке (1), значение функции ϕ в новой точке x_{k+1} уменьшается относительно значения в старой точке $\phi(x_k)$ по крайней мере на величину $m_{L_k}(x_{k+1}; x_k) - \phi(x_k)$:

$$\phi(x_{k+1}) \le m_{L_k}(x_{k+1}; x_k) \qquad (\le \phi(x_k)). \tag{2}$$

Оказывается, что именно это неравенство, а не условие $L_k \ge L_f$ отвечает за сходимость проксимального градиентного метода. Поэтому для подбора константы L_k (или, эквивалентно, длины шага α_k) на текущей итерации k можно использовать следующую простую процедуру линейного поиска: начать с некоторого значения L и увеличивать его в два раза, пока не выполнится неравенство (2). Заметим, что эта процедура определена корректно: как только значение L превысит L_f, неравенство (2) обязательно будет выполнено в силу липшицевости градиента функции f. О константе L_k удобно думать как о «локальной» константе Липшица градиента функции f.

Поскольку каждое пробное значение константы L требует полного вычисления функции ϕ, то число итераций линейного поиска необходимо, по возможности, сократить. Для этого имеет смысл инициализировать значение L_{k+1} с помощью уже найденного значения L_k. Естественным вариантом является инициализация $L_{k+1} = L_k$. Однако в этом случае константы L_k всегда будут только увеличиваться и никогда не уменьшаться; это плохо, поскольку при прочих равных лучше выбрать значение L как можно меньше, что будет соответствовать большему шагу и большему прогрессу в оптимизации (в некоторых областях пространства локальная константа Липшица может быть меньше, чем в области, содержащей начальную точку метода). Таким образом, чтобы константы Липшица на соседних итерациях были близки, но также со временем могли уменьшаться, в схеме Нестерова выполняется инициализация $L_{k+1} = L_k/2$.

Итоговый алгоритм проксимального градиентного метода с подбором длины шага по схеме Нестерова представлен в алгоритме 1. Здесь $L_0 > 0$ — параметр метода (нижняя оценка на глобальную константу Липшица L_f); его всегда можно выбрать равным единице ($L_0 = 1$) без принципиального ущерба для сходимости метода.

Можно показать, что, несмотря на то, что на отдельных итерациях проксимального градиентного метода может выполняться много шагов линейного поиска, среднее число вычислений функции ϕ за итерацию равно двум (задание 3.2).

Алгоритм 1 Проксимальный градиентный метод с подбором длины шага по схеме Нестерова

Вход: Начальная точка $x_0 \in \mathbb{R}^n$, начальная оценка константы Липшица $L_0 > 0$.

$L \leftarrow L_0$
for $k = 0, 1, \ldots$ **do**
  while True do
    $y \leftarrow \operatorname{argmin}_y m_L(y; x_k)$
    if $\phi(y) \leq m_L(y; x_k)$ **then**
      break
    end if
    $L \leftarrow 2L$
  end while
  $x_{k+1} \leftarrow y$
  $L \leftarrow \max\{L_0, L/2\}$
end for

Выход: Последняя вычисленная точка x_k

1.3 Метод барьеров

Метод барьеров (также называемый *методом внутренней точки* или *методом внутренних штрафов*) является методом решения условных задач оптимизации с ограничениями вида неравенств:

$$\min_x f(x) \qquad \text{s.t.} \quad g_i(x) \leq 0, \qquad i = 1, \ldots, m, \tag{3}$$

где $f, g_1, \ldots, g_m : \mathbb{R}^n \to \mathbb{R}$ — дважды непрерывно дифференцируемые функции, $x \in \mathbb{R}^n$ — целевая переменная.

1.3.1 Общая идея

Основная идея метода барьеров заключается в сведении исходной условной задачи (3) к последовательности специальным образом построенных безусловных задач, решения которых сходятся к решению исходной условной задачи. Это делается с помощью барьерной функции.

Пусть Q — множество, задаваемое функциональными ограничениями $g_1, \ldots, g_m$:

$$Q := \{x \in \mathbb{R}^n : g_i(x) \leq 0 \text{ для всех } 1 \leq i \leq m\}.$$

Барьерной функцией (или *функцией внутренних штрафов*) для множества Q называется любая функция $\Psi : \operatorname{int} Q \to \mathbb{R}$, определенная на внутренности множества Q и являющаяся достаточно гладкой, которая обладает *барьерным свойством*: $\Psi(x) \to +\infty$ при приближении x к границе множества Q изнутри.

Имея в распоряжении барьерную функцию, определим для каждого $t \geq 0$ вспомогательную функцию $\phi_t : \operatorname{int} Q \to \mathbb{R}$ по формуле

$$\phi_t(x) := tf(x) + \Psi(x).$$

Оказывается, что при $t \to +\infty$ точка $x(t)$ сходится к множеству решений исходной условной задачи (3). Обозначим $x(t) := \operatorname{argmin}_{x \in \operatorname{int} Q} \phi_t(x)$. Множество точек $\{x(t) : t \geq 0\}$ называется *центральным путем*. Таким образом, для решения исходной условной задачи (3), можно использовать следующую *схему отслеживания пути* (она же и есть метод барьеров).

Общая схема метода барьеров

1. Выбрать начальное приближение $x_0 \in \operatorname{int} Q$ и начальный параметр $t_0 > 0$.
2. На каждой итерации $k \geq 0$:
   (a) Вычислить точку
   $$x_{k+1} := \operatorname*{argmin}_{y \in \operatorname{int} Q} \phi_{t_k}(y), \tag{4}$$
   используя некоторый метод безусловной оптимизации с начальной точкой x_k.
   (b) Увеличить параметр t: выбрать $t_{k+1} > t_k$ (чтобы в итоге $t_k \to +\infty$).

1.3.2 Решение вспомогательной задачи

Заметим, что задача оптимизации, возникающая на каждой итерации метода барьеров в реальности является «безусловной», несмотря на присутствие в этой задаче условия $x \in \operatorname{int} Q$. Действительно, за счет барьерного свойства целевая функция ϕ_t стремится к $+\infty$ при приближении к границе множества Q, поэтому никакой «разумный» метод оптимизации достаточно близко к границе множества Q никогда не подойдет, и, с точки зрения самого метода, целевая функция как бы задана всюду. Тем не менее, здесь необходима определенная аккуратность.

Например, пусть для решения вспомогательной задачи (4) используется метод спуска: сначала $y_0 := x_k$, далее на каждой итерации $\ell \geq 0$ строится направление спуска $d_\ell \in \mathbb{R}^n$ для функции ϕ_{t_k} в точке y_ℓ, и выполняется шаг

$$y_{\ell+1} := y_\ell + \alpha_\ell d_\ell,$$

где $\alpha_\ell \geq 0$ — длина шага. Поскольку метод барьеров по построению всегда работает с внутренними точками множества Q (отсюда и альтернативное название — метод внутренней точки), то для всех *достаточно маленьких* длин шагов α_ℓ новая точка $y_{\ell+1}$ будет принадлежать множеству Q, а значение функции ϕ_{t_k} в этой точке уменьшится по сравнению с текущей точкой y_ℓ. Однако, если длина шага α_ℓ выбрана *слишком большой*, то точка $y_{\ell+1}$ может вообще оказаться за пределами множества Q — такое вполне может произойти при подборе шага в стандартном алгоритме линейного поиска. Чтобы избежать этой проблемы, стандартную процедуру линейного поиска необходимо модифицировать — запретить ей вычислять функцию в точках за пределами множества Q.

Согласно определению множества Q, чтобы $y_\ell + \alpha d_\ell \in \operatorname{int} Q$, длина шага $\alpha \geq 0$ должна удовлетворять следующей системе неравенств:

$$g_i(y_\ell + \alpha d_\ell) < 0, \qquad 1 \leq i \leq m.$$

Наиболее просто эта система решается в том случае, когда множество Q является полиэдром, т. е. когда функции $g_1, \ldots, g_m$ являются аффинными: $g_i(x) = q_i^T x - s_i$. В этом случае соответствующая система неравенств выглядит следующим образом:

$$q_i^T(y_\ell + \alpha d_\ell) < s_i, \qquad 1 \leq i \leq m.$$

Раскрывая скобки, получаем

$$(q_i^T d_\ell)\alpha < s_i - q_i^T y_\ell, \qquad 1 \leq i \leq m. \tag{5}$$

Поскольку по построению точка y_ℓ — внутренняя, то все правые части в этих неравенствах строго положительные. Значит, если для некоторого $1 \leq i \leq m$ коэффициент $q_i^T d_\ell \leq 0$, то соответствующее неравенство бессодержательно (поскольку $\alpha \geq 0$). Введем обозначение

$$I_\ell := \{1 \leq i \leq m : q_i^T d_\ell > 0\}.$$

Тогда система (5) эквивалентна следующей системе:

$$(q_i^T d_\ell)\alpha < s_i - q_i^T y_\ell, \qquad i \in I_\ell.$$

Заметим, что теперь все коэффициенты при α строго положительные, поэтому законно разделить:

$$\alpha < \frac{s_i - q_i^T y_\ell}{q_i^T d_\ell}, \qquad i \in I_\ell.$$

Наконец, последнюю систему можно переписать в виде одного скалярного неравенства:

$$\alpha < \alpha_\ell^{\max}, \qquad \text{где } \alpha_\ell^{\max} := \min_{i \in I_\ell} \frac{s_i - q_i^T y_\ell}{q_i^T d_\ell}.$$

Итак, согласно проведенным выкладкам, все допустимые длины шагов α лежат в интервале $[0, \alpha_\ell^{\max})$, т. е. в процедуре линейного поиска никогда нельзя брать шаг $\alpha \geq \alpha_\ell^{\max}$. Например, если для подбора длины шага используется процедура бэктрекинга, то в этом случае единственная модификация, которую нужно сделать по сравнению со стандартным случаем, — это изменить начальную пробную длину шага: вместо прежде используемого значения α_ℓ^0 теперь нужно использовать значение

$$\tilde{\alpha}_\ell^0 := \min\{\alpha_\ell^0, \theta \alpha_\ell^{\max}\},$$

где $0 < \theta < 1$ — некоторая константа (например, 0.99).

1.3.3 Полная спецификация метода

К настоящему моменту схема метода барьеров все еще довольно абстрактная, и имеется несколько вопросов, на которые нужно ответить:

1. Как именно выбирать барьерную функцию Ψ?
2. Какой метод безусловной оптимизации нужно использовать для решения внутренней задачи, и какова должна быть точность ее решения?
3. Какая конкретно должна быть стратегия увеличения параметра t_k в схеме отслеживания пути?

Сперва ответим на первые два вопроса. Несмотря на то, что гипотетически в методе барьеров можно использовать любую комбинацию барьерной функции и метода безусловной минимизации, наиболее эффективной (как с точки зрения теории, так и практики) оказывается связка *логарифмического* барьера и *метода Ньютона*. Логарифмический барьер в общем виде имеет вид

$$\Psi(x) := -\sum_{i=1}^{m} \ln(-g_i(x)).$$

В качестве *критерия остановки* в методе Ньютона в этом задании Вам нужно будет использовать стандартный критерий по норме градиента целевой функции (который использовался в предыдущих заданиях):

$$\|\nabla \phi_{t_k}(y_\ell)\|_2^2 \leq \varepsilon_{\text{inner}} \|\nabla \phi_{t_k}(x_k)\|_2^2,$$

где $\varepsilon_{\text{inner}} > 0$ — некоторый параметр, который обычно выбирается довольно маленьким (например, 10^{-7}), что соответствует практически точному решению внутренней задачи (в выпуклом случае). Заметим, что поскольку в качестве алгоритма минимизации используется метод Ньютона, а стартовая точка выбирается достаточно «хорошей», то маленькое значение $\varepsilon_{\text{inner}}$ совсем не представляет никакой проблемы для метода (напомним, что метод Ньютона имеет локальную квадратичную сходимость).

Что касается последнего вопроса о стратегии увеличения параметров t_k, то здесь имеется несколько эффективных стратегий. Самая простая из них, которую Вам нужно будет использовать в этом задании, состоит в постоянном *линейном увеличении*:

$$t_{k+1} := \gamma t_k,$$

где $\gamma > 1$ — постоянная константа (обычно порядка 10–100).

2 Модели

2.1 Модель LASSO

Модель LASSO является одной из стандартных моделей линейной регрессии. Имеется обучающая выборка $((a_i, b_i))_{i=1}^m$, где $a_i \in \mathbb{R}^n$ — вектор признаков i-го объекта, а $b_i \in \mathbb{R}$ — его регрессионное значение. Задача заключается в прогнозировании регрессионного значения b_{new} для нового объекта, представленного своим вектором признаков a_{new}.

В модели LASSO, как и в любой модели линейной регрессии, прогнозирование выполняется с помощью линейной комбинации компонент вектора a с некоторыми фиксированными коэффициентами $x \in \mathbb{R}^n$:

$$b(a) := a^T x.$$

Коэффициенты x являются параметрами модели и настраиваются с помощью решения следующей оптимизационной задачи:

$$\phi(x) := \frac{1}{2}\sum_{i=1}^{m}(a_i^T x - b_i)^2 + \lambda \sum_{j=1}^{n}|x_j| =: \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1 \to \min_{x\in\mathbb{R}^n}. \tag{6}$$

Здесь $\lambda > 0$ — коэффициент регуляризации (параметр модели). Особенностью LASSO является использование именно ℓ_1-регуляризации (а не, например, ℓ_2-регуляризации). Такая регуляризация позволяет получить разреженное решение. В разреженном решении x^* часть компонент равна нулю. (Можно показать, что при $\lambda \geq \|A^T b\|_\infty$ все компоненты будут нулевыми). Нулевые веса соответствуют исключению соответствующих признаков из модели (признание их неинформативными).

2.2 Двойственная задача и критерий остановки

Перепишем задачу (6) в следующем эквивалентном виде:

$$\min_{x\in\mathbb{R}^n,\, z\in\mathbb{R}^n}\left\{\frac{1}{2}\|z\|_2^2 + \lambda\|x\|_1 : Ax - b = z\right\}. \tag{7}$$

Можно показать, что двойственной задачей к задаче (7) является

$$\max_{\mu\in\mathbb{R}^n}\left\{-\frac{1}{2}\|\mu\|_2^2 - b^T\mu : \|A^T\mu\|_\infty \leq \lambda\right\}. \tag{8}$$

Таким образом, имея в распоряжении допустимую двойственную точку $\mu \in \mathbb{R}^n$, т. е. такую что $\|A^T\mu\|_\infty \leq \lambda$, можно вычислить следующую оценку для невязки в задаче (6):

$$\phi(x) - \phi^* \leq \frac{1}{2}\|Ax - b\|_2^2 + \lambda\|x\|_1 + \frac{1}{2}\|\mu\|_2^2 + b^T\mu =: \eta(x, \mu). \tag{9}$$

Величина $\eta(x, \mu)$ называется *зазором двойственности* и обращается в ноль в оптимальных решениях x^* и μ^* задач (6) и (8). Заметим, что решения x^* и μ^* связаны между собой следующим соотношением: $Ax^* - b = \mu^*$. Поэтому для фиксированного x естественным выбором соответствующего μ будет

$$\mu(x) := \min\left\{1, \frac{\lambda}{\|A^T(Ax - b)\|_\infty}\right\}(Ax - b).$$

Такой выбор обеспечивает стремление зазора двойственности $\eta(x, \mu(x))$ к нулю при $x \to x^*$, что позволяет использовать условие $\eta(x, \mu(x)) < \varepsilon$ в качестве критерия остановки в любом итерационном методе решения задачи (6).

В этом задании все рассматриваемые методы должны использовать критерий остановки по зазору двойственности $\eta(x, \mu(x))$.

2.3 Сведение к гладкой условной задаче оптимизации

Выполнив эквивалентное преобразование задачи через надграфик, задачу (6) можно переписать в виде гладкой условной задачи:

$$\min_{x,u} \left\{ \frac{1}{2}\|Ax - b\|_2^2 + 1_n^T u \right\} \qquad \text{s.t.} \quad x \preceq u, \quad x \succeq -u. \tag{10}$$

где $x, u \in \mathbb{R}^n$ и $1_n := (1, \ldots, 1) \in \mathbb{R}^n$. Эта задача является задачей QP, и ее можно решать, например, с помощью метода барьеров.

3 Формулировка задания

1 Скачайте коды, прилагаемые к заданию:

`https://github.com/mihaha/hse-optimization-course/tree/master/task3`

Эти файлы содержат прототипы функций, которые Вам нужно будет реализовать. Некоторые процедуры уже частично или полностью реализованы.

2 Реализуйте субградиентный метод (функция `subgradient_method` в модуле `optimization`).

3 Реализуйте негладкий оракул для функции (6) (классы `LeastSquaresOracle` и `LassoNonsmoothOracle` в модуле `oracles`).

4 Реализуйте процедуру вычисления зазора двойственности (9) (функция `lasso_duality_gap` в модуле `oracles`).

5 Реализуйте проксимальный градиентный метод (функция `proximal_gradient_descent` в модуле `optimization`).

6 Реализуйте композитный оракул для функции (6) (классы `L1RegOracle` и `LassoProxOracle` в модуле `oracles`).

7 Выпишите для задачи (10) вспомогательную функцию $\phi_t(x, u)$, минимизируемую в методе барьеров. Выпишите систему линейных уравнений, задающую ньютоновоское направление $d_k = (d_k^x, d_k^u)$. Предложите свой способ решения этой системы и прокомментируйте его достоинства и недостатки. Для текущей точки (x, u) и направления d_k определите максимальную допустимую длину шага α. Какую начальную точку (x_0, u_0) можно выбрать для метода барьеров?

8 Реализуйте метод барьеров для задачи (10) (функция `barrier_method_lasso` в модуле `optimization`). Для подбора длины шага на внутренних итерациях используйте бэктрекинг. (Не забудьте выбрать начальное значение длины шага равным единице, если оно допустимо.)

9 Проведите эксперименты, описанные ниже. Напишите отчет.

3.1 Эксперимент: Выбор длины шага в субградиентном методе

Исследуйте работу субградиентного метода в зависимости от выбора константы α_0 в формуле для длины шага. Обязательно попробуйте различные начальные точки.

3.2 Эксперимент: Среднее число итераций линейного поиска в схеме Нестерова

Для проксимального метода построить график суммарного (куммулятивного) числа итераций линейного поиска в зависимости от номера итерации. Действительно ли среднее число итераций линейного поиска не превышает (примерно) двух?

3.3 Эксперимент: Сравнение методов

Провести экспериментальное сравнение трех реализованных методов для различных значений числа переменных n; рассмотреть, как минимум, случаи $n \leq 100$ и $n \geq 1000$. Какие методы быстрее достигают низкой точности ($\varepsilon = 10^{-2}$), а какие высокой ($\varepsilon = 10^{-10}$)?

Значение коэффициента регуляризации λ выбрать стандартным образом: $\lambda = 1$.

Для сравнения методов нарисуйте графики 1) гарантируемая точность по зазору двойственности (9) против числа итераций и 2) гарантированная точность по зазору двойственности против реального времени работы. Для гарантированной точности по зазору двойственности используйте логарифмическую шкалу.

4 Оформление задания

Результатом выполнения задания являются

1. Файлы `optimization.py` и `oracles.py` с реализованными методами и оракулами.
2. Полные исходные коды для проведения экспериментов и рисования всех графиков. Все результаты должны быть воспроизводимыми. Если вы используете случайность — зафиксируйте `seed`.
3. Отчет в формате PDF о проведенных исследованиях.

Выполненное задание следует отправить письмом на почту своей группы[2] с заголовком

Практическое задание 3, Фамилия Имя.

Задание следует присылать только один раз с окончательным вариантом.

Каждый проведенный эксперимент следует оформить как отдельный раздел в PDF документе (название раздела — название соответствующего эксперимента). Для каждого эксперимента необходимо сначала написать его описание: какие функции оптимизируются, каким образом генерируются данные, какие методы и с какими параметрами используются. Далее должны быть представлены результаты соответствующего эксперимента — графики, таблицы и т.д. Наконец, после результатов эксперимента должны быть написаны Ваши выводы — какая зависимость наблюдается и почему.

Важно: Отчет не должен содержать никакого кода. Каждый график должен быть прокомментирован — что на нем изображено, какие выводы можно сделать из этого эксперимента. Обязательно должны быть подписаны оси. Если на графике нарисовано несколько кривых, то должна быть легенда. Сами линии следует рисовать достаточно толстыми, чтобы они были хорошо видимыми.

5 Проверка задания

Перед отправкой задания обязательно убедитесь, что Ваша реализация проходит автоматические *предварительные* тесты `presubmit_tests.py`, выданные вместе с заданием. Для этого запустите команду

```
python presubmit_tests.py
```

Важно: Файл с тестами может измениться. Перед отправкой обязательно убедитесь, что ваша версия `presubmit_tests.py` — последняя.

Важно: Решения, которые не будут проходить тесты `presubmit_tests.py`, будут автоматически оценены в 0 баллов. Проверяющий не будет разбираться, почему Ваш код не работает и читать Ваш отчет.

Оценка за задание будет складываться из двух частей:

[2]Для 141 группы: opt.homework+141@gmail.com. Для 142 группы: opt.homework+142@gmail.com. Для 145 группы: opt.homework+145@gmail.com.

1. Правильность и эффективность реализованного кода.

2. Качество отчета.

Правильность и эффективность реализованного кода будет оцениваться автоматически с помощью независимых тестов (отличных от предварительных тестов). Качество отчета будет оцениваться проверяющим. При этом оценка может быть субъективной и аппеляции не подлежит.

За реализацию модификаций алгоритмов и хорошие дополнительные эксперименты могут быть начислены дополнительные баллы. Начисление этих баллов является субъективным и безапелляционным.

Важно: Практическое задание выполняется самостоятельно. Если вы получили ценные советы (по реализации или проведению экспериментов) от другого студента, то об этом должно быть явно написано в отчёте. В противном случае «похожие» решения считаются плагиатом и все задействованные студенты (в том числе те, у кого списали) будут сурово наказаны.