ВМК

Задание 1. Изучение Python, NumPy

Практикум 317 группы, осень 2015

Начало выполнения задания: 24 сентября 2015 года.
Срок сдачи: **9 октября 2015 года, 23:59.**

Содержание

1 Задание

Данное задание направлено на освоение студентами языка Python и системы научных вычислений NumPy.

1. Для каждой из задач:
   (a) Написать на Python + NumPy несколько вариантов кода различной эффективности. Должно быть не менее трёх вариантов, в том числе как минимум один полностью векторизованный вариант и один вариант без векторизации. Третий вариант решения — на ваше усмотрение, например, это может быть наиболее хорошо читаемый способ решения или частично векторизованный вариант. Все варианты решения одной задачи должны содержаться в отдельном Python модуле.
   (b) Сравнить в IPython Notebook при помощи `%timeit` скорость работы на нескольких тестовых наборах разного размера (минимум 3).
   (c) Проанализировать полученные данные о скорости работы разных реализаций.
   (d) Получить выводы.
2. Написать отчёт о проделанной работе (формат PDF). Отчёт должен быть подготовлен в системе LaTeX. Полезные ссылки: [1] [2].
3. Выполненное задание (Python модули с решениями отдельных задач, IPython Notebook с проведением экспериментов, PDF-файл с отчётом о выполнении задания) прислать по почте преподавателю.

Советы по выполнению задания.

- Чтобы перезагрузить уже загруженный в IPython модуль, воспользуйтесь функцией `importlib.reload`.
- Сохранить результаты запуска `%timeit` в переменную можно так: `x = %timeit -o func(x)`.

2 Бонусные баллы

- Написанный код полностью соответствует style guide PEP 8 [3]. Часть требований можно проверить при помощи утилиты flake8 [4].
- Ко всем задачам присутствуют автоматические тесты, проверяющие совпадение результатов работы всех вариантов кода. Тесты должны использовать встроенный в Python фреймворк unittest. Краткое руководство по этому фреймворку: [5]. Обратите внимание на модель `numpy.testing`, облегчающий написание тестов для NumPy массивов.

3 Задачи

Предполагается, что модуль numpy импортирован под названием np.

1. Подсчитать произведение ненулевых элементов на диагонали прямоугольной матрицы. Для X = np.array([[1, 0, 1], [2, 0, 2], [3, 0, 3], [4, 4, 4]]) ответ 3.

2. Дана матрица X и два вектора одинаковой длины i и j. Построить вектор np.array([X[i[0], j[0]], X[i[1], j[1]], ..., X[i[N-1], j[N-1]]]).

3. Даны два вектора x и y. Проверить, задают ли они одно и то же мультимножество. Для x = np.array([1, 2, 2, 4]), y = np.array([4, 2, 1, 2]) ответ True.

4. Найти максимальный элемент в векторе x среди элементов, перед которыми стоит нулевой. Для x = np.array([6, 2, 0, 3, 0, 0, 5, 7, 0]) ответ 5.

5. Дан трёхмерный массив, содержащий изображение, размера (height, width, numChannels), а также вектор длины numChannels. Сложить каналы изображения с указанными весами, и вернуть результат в виде матрицы размера (height, width). Считать реальное изображение можно при помощи функции `scipy.misc.imread` (если изображение не в формате png, установите пакет pillow: `conda install pillow`). Преобразуйте цветное изображение в оттенки серого, использовав коэффициенты np.array([0.299, 0.587, 0.114]).

6. Реализовать кодирование длин серий (Run-length encoding). Дан вектор x. Необходимо вернуть кортеж из двух векторов одинаковой длины. Первый содержит числа, а второй - сколько раз их нужно повторить. Пример: x = np.array([2, 2, 2, 3, 3, 3, 5]). Ответ: (np.array([2, 3, 5]), np.array([3, 3, 1])).

7. Даны две выборки объектов - X и Y. Вычислить матрицу евклидовых расстояний между объектами. Сравнить с функцией `scipy.spatial.distance.cdist`.

8. Реализовать функцию вычисления логарифма плотности многомерного нормального распределения Входные параметры: точки X, размер (N, D), мат. ожидание m, вектор длины D, матрица ковариаций C, размер (D, D). Разрешается использовать библиотечные функции для подсчета определителя матрицы, а также обратной матрицы, в том числе в невекторизованном варианте. Сравнить с `scipy.stats.multivariate_normal(m, C).logpdf(X)` как по скорости работы, так и по точности вычислений.

Замечание. Можно считать, что все указанные объекты непустые (к примеру, в задаче №1 на диагонали матрицы есть ненулевые элементы).

Полезные функции NumPy: np.zeros, np.ones, np.diag, np.eye, np.arange, np.linspace, np.meshgrid, np.random.random, np.random.randint, np.shape, np.reshape, np.transpose, np.any, np.all, np.nonzero, np.where, np.sum, np.cumsum, np.prod, np.diff, np.min, np.max, np.minimum, np.maximum, np.argmin, np.argmax, np.unique, np.sort, np.argsort, np.bincount, np.ravel, np.newaxis, np.dot, np.linalg.inv, np.linalg.solve.

Многие из этих функций можно использовать так: x.argmin().

Список литературы

[1] К.В. Воронцов. Полезная информация для пользователей LaTeX. http://www.ccas.ru/voron/latex.html

[2] К.В. Воронцов. LaTeX2e в примерах. http://www.ccas.ru/voron/download/voron05latex.pdf

[3] PEP 8 — Style Guide for Python Code. http://legacy.python.org/dev/peps/pep-0008/

[4] Flake8. https://pypi.python.org/pypi/flake8

[5] Corey Goldberg. Python Unit Testing Tutorial. http://cgoldberg.github.io/python-unittest-tutorial/