Задание №1 Изучение Python, NumPy

Задание основано на задании курса “Математические методы прогнозирования” (кафедра ВМиК МГУ)

Для каждой из задач:

1) Написать на Python + NumPy несколько вариантов кода различной эффективности. Один полностью векторизованный вариант и один вариант без векторизации. Варианты решения одной задачи должны содержаться в отдельном Python модуле.
2) Сравнить в IPython при помощи %timeit скорость работы на нескольких тестовых наборах разного размера (минимум 3)
3) Проанализировать полученные данные о скорости работы разных реализаций.

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).

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_idx и j_idx. Построить вектор np.array([X[i_idx[0], j_idx[0]], X[i_idx[1], j_idx[1]], . . . , X[i_idx[N-1], j_idx[N-1]]]). Для X = np.array(range(4 * 5)).reshape(4, 5) + 1, i_idx = np.array([1, 3, 0, 2]), j_idx = np.array([0, 2, 3, 1]) ответ: [6 18 4 12]
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]). Для вывода используйте scipy.misc.imshow или matplotlib.pyplot.imshow

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])).

Бонусные баллы

- Написанный код соответствует style guide PEP 8. Часть требований можно проверить при помощи утилиты flake8.
- 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: http://cgoldberg.github.io/python-unittest-tutorial/. Обратите внимание на модель numpy.testing, облегчающий написание тестов для NumPy массивов.

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().