

Численные методы, осень 2022

Задание 1 [Векторные и матричные нормы. Ортогональные матрицы. NumPy] Всего баллов: 60 Срок сдачи: 7 октября

РЕКОМЕНДОВАННАЯ ЛИТЕРАТУРА

- Лекции 1-3 из [1]
- Лекции 1-2 из [2]
- [От Python к NumPy](#)
- [100 упражнений на NumPy](#)

УПРАЖНЕНИЯ

- (5) Рассмотрим матрицу $H(v) = 1 - 2vv^*$, где v — единичный вектор-столбец. Каков ранг матрицы $H(v)$? Докажите, что она унитарна.
- (5) Пусть x — вектор размерности m , а A — матрица $m \times n$. Докажите следующие неравенства и приведите примеры таких x и A , при которых неравенства обращаются в равенства:
 - $\|x\|_2 \leq \sqrt{m} \|x\|_\infty$
 - $\|A\|_\infty \leq \sqrt{n} \|A\|_2$
- (5) Пусть u и v — векторы размерности m . Рассмотрим матрицу $A = 1 + uv^*$, которая отличается от единичной на возмущение ранга 1. Может ли она быть вырожденной? Предположив, что нет, вычислите обратную матрицу. Вы можете искать её в виде $A^{-1} = 1 + \alpha uv^*$, где α — число, которое нужно найти.
- (5) Докажите, что для любой унитарной матрицы U (и для произвольной матрицы A) имеет место равенство $\|UA\|_F = \|AU\|_F = \|A\|_F$, где $\|\cdot\|_F$ — норма Фробениуса.
- (5) Рассмотрите двумерное векторное пространство $r = (x, y)$ и постройте единичный круг $\|r\|_p \leq 1$ при $p = 1, 2, 3$ (используйте библиотеку `matplotlib`).
- (15) Рассмотрим функцию, отображающую шесть тензоров на один тензор: $Z(\lambda^{(1)}, \lambda^{(2)}, \lambda^{(3)}, \Gamma^{(1)}, \Gamma^{(2)}, U)$:

$$Z_{ahij} = \sum_{bcdefg} \lambda^{(1)}_{ab} \Gamma^{(1)}_{cbd} \lambda^{(2)}_{de} \Gamma^{(2)}_{feg} \lambda^{(3)}_{gh} U_{ijcf}$$

Предположив, что все индексы пробегает значения от 1 до χ , проведите эксперимент и сравните скорость различных реализаций функции Z . Исследуйте значения χ в диапазоне 3–50.

- В файле `convolution.ipynb` вы можете найти реализацию *глупого* способа вычисления этой свертки, который требует $\chi^4 \times \chi^6 = \chi^{10}$ операций. На самом деле это можно вычислить гораздо быстрее!
 - С помощью функции `numpy.einsum` (нужно использовать аргумент `optimize`), можно добиться намного большей производительности. Чтобы понять, что происходит под капотом, воспользуйтесь функцией `numpy.einsum_path`. Какое минимальное количество операций требуется для вычисления Z ?
 - Посмотрев на вывод функции `numpy.einsum_path`, реализуйте алгоритм для вычисления Z , который столь же эффективен, как `numpy.einsum`, но использует более элементарные `numpy.dot` и `numpy.tensor_dot`.
- (10) В этом упражнении вашей целью будет изучение и ускорение реализации [алгоритма K-средних](#), который используется для кластеризации. В блокноте `kmeans.ipynb` вы можете найти наивную реализацию. Изучите код, убедитесь, что вы его поняли. Вы найдете там две функции `dist_i` и `dist_ij`, которые (намеренно) реализованы довольно неэффективно. Улучшите их, избавившись от циклов с помощью векторизации из `numpy`, и измерьте ускорение алгоритма в целом при $N = 10000$.

8. (10) Некоторые вещи просто не могут быть векторизованы, но все же могут быть ускорены по сравнению с наивной реализацией. Например, рассмотрим вычисление [последовательности Хофштадтера-Конвея](#):

$$\begin{aligned} a(1) &= 1 \\ a(2) &= 1 \\ a(n) &= a(a(n-1)) + a(n - a(n-1)), \quad n > 2 \end{aligned}$$

Напишите три функции, вычисляющие последовательность до n -го элемента разными способами:

- i) предварительно выделяя массив `numpy` и заполняя его с помощью цикла `for`
- ii) добавляя элементы по одному в `python`-список и преобразуя его в `numpy`-массив в конце
- iii) тоже, что и (i), но в скомпилированной (`jit`) версии

Выберите самую быструю реализацию и вычислите с помощью неё $a(10^8)$.

-
- [1] L. Trefethen and D. Bau, *Numerical Linear Algebra*, Other Titles in Applied Mathematics (SIAM, 1997).
 [2] E. Tyrtyshnikov, *A Brief Introduction to Numerical Analysis* (Birkhäuser Boston, 2012).