# Сравнение эффективности алгоритмов SVD и QR with column pivoting для определения ранга матрицы

команда “Миньоны Тыртыша”
Оганов А. А.
Никишкин П. Г.
Давыдов К. Е.

# Математическая постановка задачи

Опр 1. Факторизация, раскрывающая ранг (RRF)

$A \in \mathbf{R}^{m \times n},\ X \in \mathbf{R}^{m \times p},\ D \in \mathbf{R}^{p \times p},\ Y \in \mathbf{R}^{p \times n},\ p \leq min(m, n)$

$A = XDY$, где D - диагональная, а X, Y - хорошо обусловленные

Примеры: SVD, QR разложение с поворотом столбцов

QR разложение с поворотом столбцов

$A\Pi = QR$, где $\prod$– матрица перестановок, $Q \in \mathbb{R}^{m \times n}$ имеет ортонормированные столбцы, $R \in \mathbb{R}^{n \times n}$является верхнетреугольной и удовлетворяет неравенствам:

$$|r_{kk}|^2 \geq \sum_{i=k}^{j} |r_{ij}|^2, j = k+1 \colon n, k = 1 \colon n, \quad |r_{11}| \geq |r_{22}| \geq \cdots \geq |r_{nn}|$$

# Постановка задачи

Сравнение методов оценки ранга матриц с помощью SVD и QR разложений.

Критерии сравнения:

1. Вычислительная сложность
2. Точность получаемых результатов
3. Поведение алгоритмов на реальных данных

Реализации:

QR - scipy.linalg.qr(pivoting=True)

SVD - np.linalg.svd()

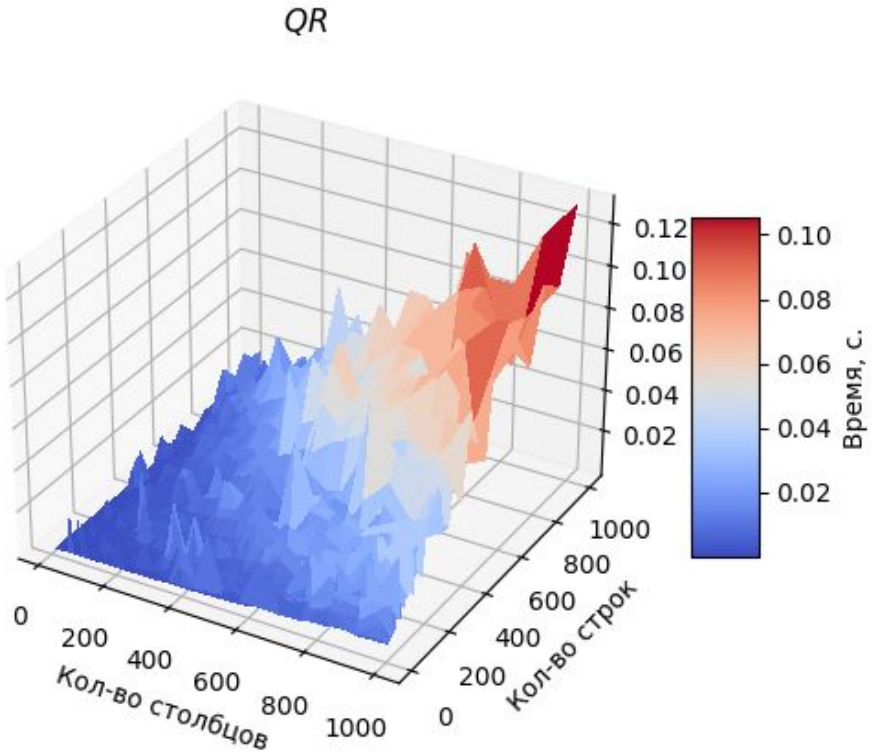
# Что такое rank?

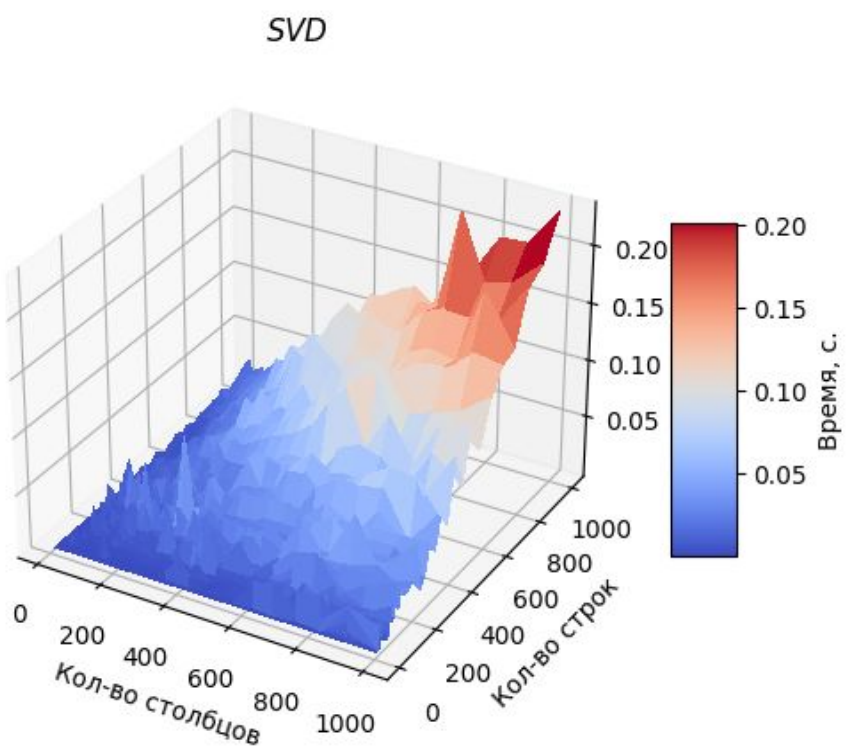
Будем использовать следующее определение ранга матрицы:

$$rank(A) = k \iff \|A - B\|_F \leq \varepsilon \|A\|_F$$

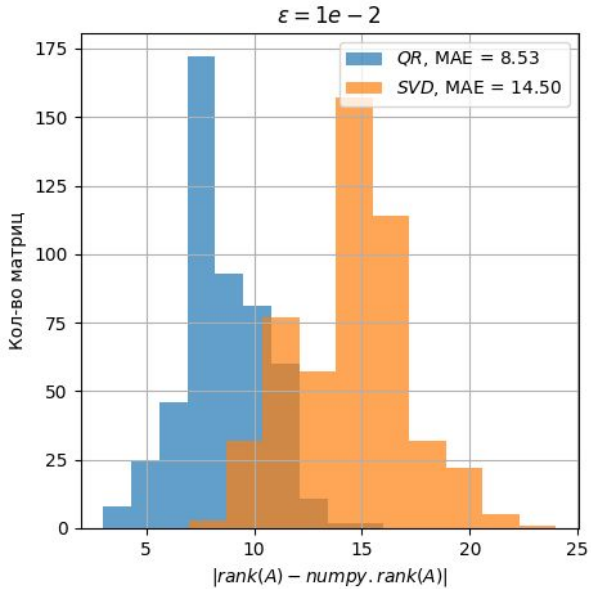
Матрица B определяется для каждого метода по своему. В случае QR мы оставляем первые k строк треугольной матрицы, в случае SVD - первые k сингулярных чисел.

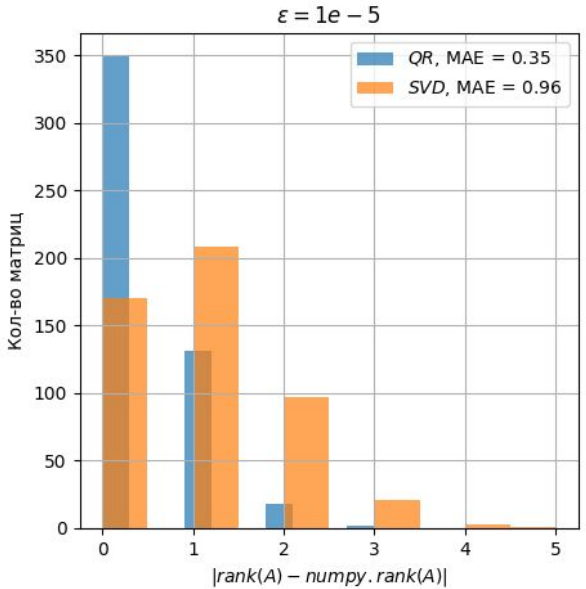
# Зависимость времени от размерности матрицы

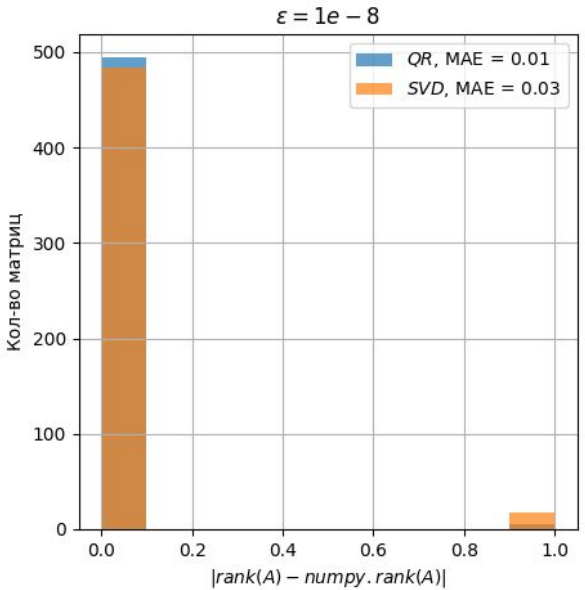


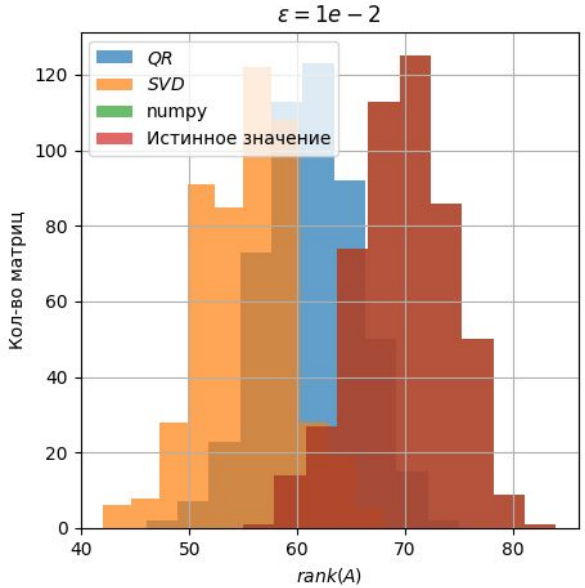
# Точность оценки в сравнении с numpy

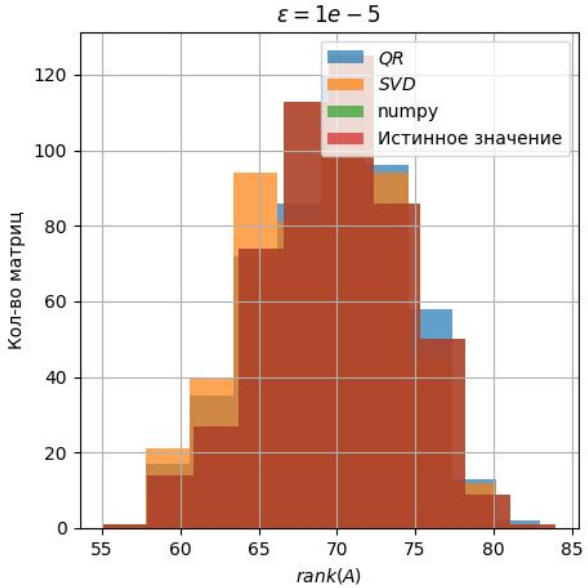


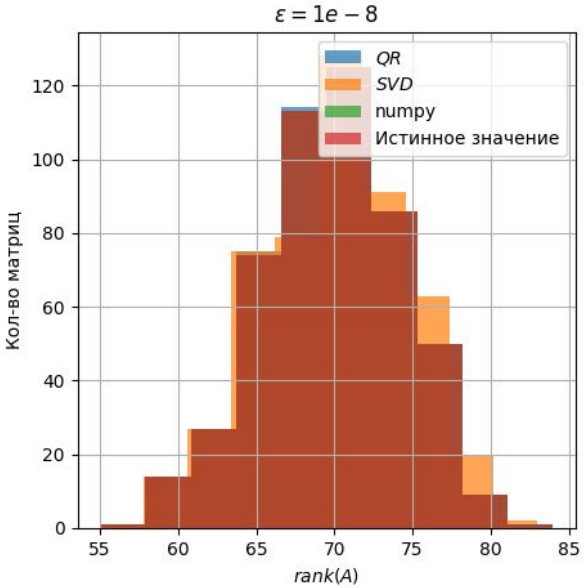



# Корректность определения ранга



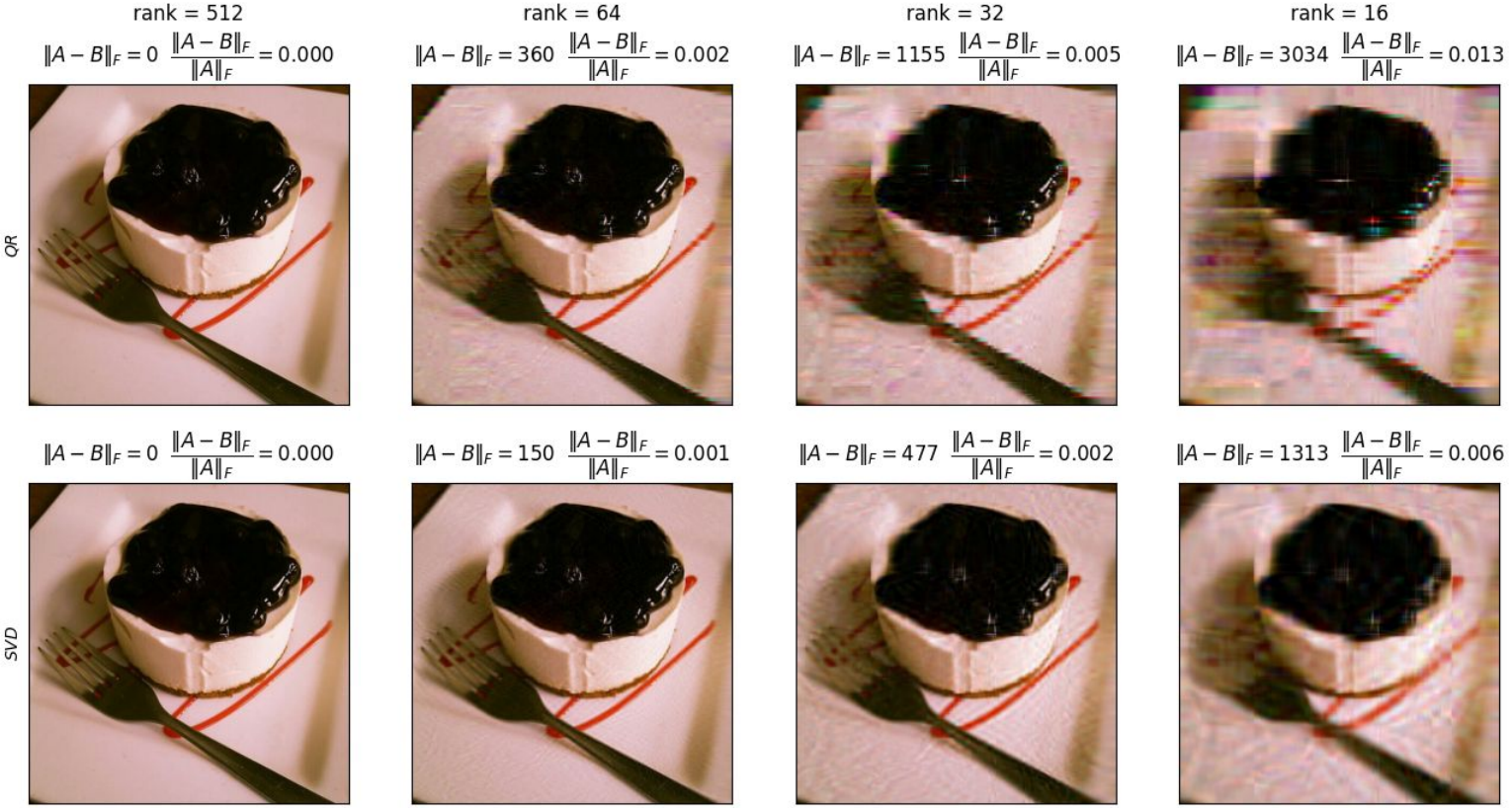



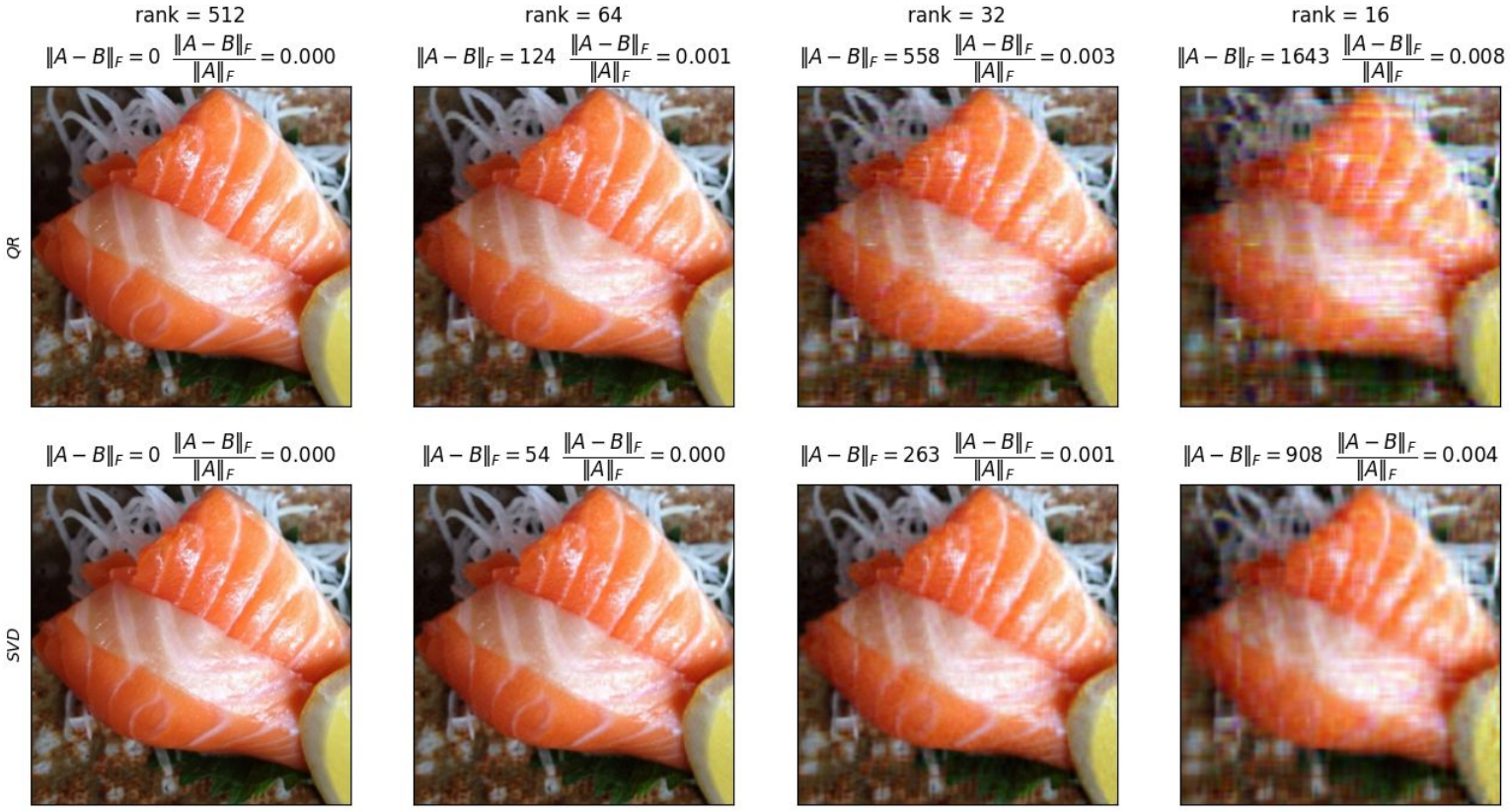



# Промежуточные выводы

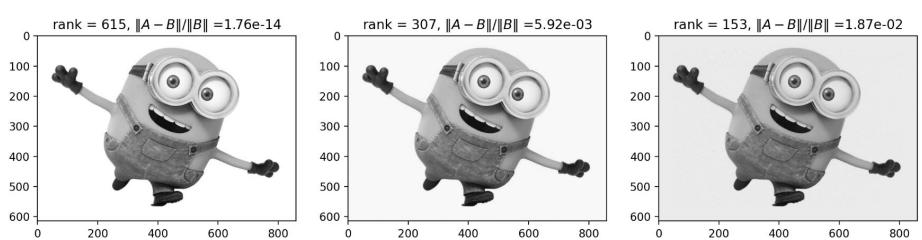
1. QR разложение работает быстрее SVD, что более заметно при увеличении размерности матрицы
2. При сравнении с numpy QR имеет меньше ошибку, чем SVD
3. Оба алгоритма и numpy считают число, которое похоже на истинное значение ранга
4. При использовании SVD получаются меньшие значения ранга

# Ранг изображения

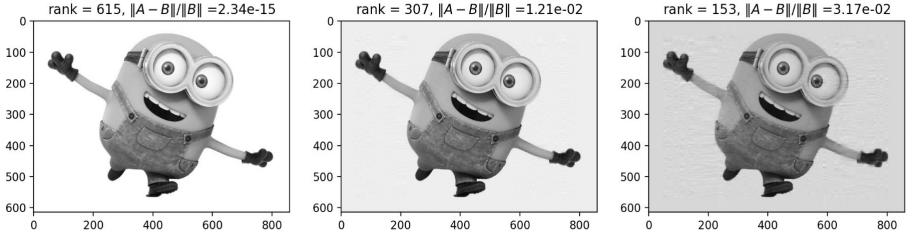



rank = 512
$\|A-B\|_F = 0$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.000$
rank = 64
$\|A-B\|_F = 124$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.001$
rank = 32
$\|A-B\|_F = 558$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.003$
rank = 16
$\|A-B\|_F = 1643$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.008$
QR
$\|A-B\|_F = 0$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.000$
$\|A-B\|_F = 54$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.000$
$\|A-B\|_F = 263$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.001$
$\|A-B\|_F = 908$ $\frac{\|A-B\|_F}{\|A\|_F} = 0.004$
SVD

# Результаты сжатия черно-белого изображения




с помощью SVD




с помощью QR

# Промежуточные выводы

1. При помощи SVD можно приблизить матрицу малоранговой более точно (как по норме, так и визуально), чем при QR.
2. Несмотря на то, что numpy оценивает изображение как полноранговую матрицу, ее можно представить как матрицу с меньшим рангом без потери визуального качества.

# Выводы

1. Было обнаружено существенное превосходство QR над SVD по времени работы на матрицах больших размеров, а также по точности определения ранга в сравнении с методом из библиотеки numpy. Однако SVD показывает лучший результат для малорангового приближения матрицы: сжатие получается более точным и визуально, и по значению ошибки.
2. Полученные выводы соответствуют высказанным заранее предположениям об эффективности работы алгоритмов.
3. Среди возможных путей дальнейшего развития проекта можно рассмотреть другие факторизации, раскрывающие ранг (LU, UTV), и сравнить их эффективность

# Что хотелось бы проверить?

1. Другой домен данных для плотных матриц, например, речь, аудио, различные сигналы
2. Проверить качество и время на случайных разреженных матрицах
3. Проверить QR для получения малорангового приближения матриц рекомендательных систем

# Основные ссылки


1. “What Is a Rank-Revealing Factorization?”, May 19, 2021 by Nick Higham
2. Golub, Gene H., and Charles F. Van Loan. *Matrix Computations*. 4th ed. Baltimore, MD: Johns Hopkins University Press, 2013, Sections 5.4.*
3. Project on GitHub