MDS Mini Project

Generalization of Randomized Singular Value Decomposition

Nirmit Arora 200010034
Ninad Milind Khati 200020030
Kanishq Verma 200020021
Keshav Naram 200020029

April 7, 2023

1 Problem Statement

Singular Value Decomposition (SVD) is a popular technique used for analyzing data and extracting information from large matrices. However, computing the exact SVD can be computationally expensive, especially for large datasets. Randomized SVD (rSVD) offers a more efficient alternative for approximating the SVD using matrix-vector products with random vectors.

The aim of this research is to generalize the rSVD algorithm by incorporating prior knowledge and enabling the computation of accurate rank approximations for a wider range of linear operators. We propose a new algorithm for generalized rSVD that utilizes multivariate Gaussian vectors and operator-function products with functions drawn from a Gaussian process (GP). This allows us to explore the continuous analogue of the randomized SVD and construct a new covariance kernel based on weighted Jacobi polynomials. The proposed algorithm enables the rapid sampling of the GP and provides control over the smoothness of the randomly generated functions, leading to accurate rank approximations of linear operators.

We will evaluate the performance of the proposed algorithm on a range of linear operators and compare it with traditional SVD methods. The results of our study have the potential to significantly reduce the computational cost of analyzing large datasets and improve the accuracy of linear operator approximations in various applications.

1.1 Background: Randomised Singular Value Decomposition

Singular Value Decomposition (SVD) is a powerful tool for analyzing data and extracting information from large smatrices. However, computing the exact SVD can be computationally expensive, especially for large datasets. Randomized SVD (rSVD) offers a more efficient alternative for approximating the SVD using matrix-vector products with random vectors.

The standard procedure followed for the decomposition of a matrix using rsvd is as follows :

- Given an Input Matrix A . Generate a normally distributed two dimensional matrix Ω with number of columns = k that is the rank . Perform the QR decomposition using the equality $A\Omega = QR$.
- Generate matrix $Y{=}Q^T A$. Perfom the svd of Y . Retrieve the $\tilde{U}$, S , V matrices from the decomposition .
- Multiply the $\tilde{U}$ matrix with Q to attain the original U matrix .
- The final decomposed matrices of matrix A are U , S , V

2 Implementation of Code

2.1 Randomized SVD code

```python
def rsvd(matrix, k, oversample=10):
    """Perform randomized SVD on a given matrix.

    Args:
        matrix (np.ndarray): Input matrix to perform SVD on.
        k (int): Number of singular values/vectors to compute.
        oversample (int): Number of extra samples to take.

    Returns:
        Tuple of matrices U, S, and V^T such that
        matrix = U @ np.diag(S) @ V^T.
    """
    # Compute the shape of the input matrix.
    m, n = matrix.shape

    # Generate a random Gaussian matrix with shape (n, k + oversample).
    omega = np.random.randn(n, k + oversample)

    # Form the sample matrix Y.
    y = matrix @ omega

    # Compute the QR decomposition of Y.
    q, _ = np.linalg.qr(y)

    # Compute the matrix B = Q^T @ A.
    b = q.T @ matrix

    # Compute the SVD of B.
    u_tilde, s, v = svd(b, full_matrices=False)

    # Compute the matrix U = Q @ U_tilde.
    u = q @ u_tilde

    # Return the SVD factors.
    return u[:,:k], s[:k], v[:k,:]
```

2.2 Generalized Randomized SVD code

```
def grsvd(matrix, k, oversample=10):
    """Perform randomized SVD on a given matrix.

    Args:
        matrix (np.ndarray): Input matrix to perform SVD on.
        k (int): Number of singular values/vectors to compute.
        oversample (int): Number of extra samples to take.

    Returns:
        Tuple of matrices U, S, and V^T such that
        matrix = U @ np.diag(S) @ V^T.
    """
    # Compute the shape of the input matrix.
    m, n = matrix.shape

    # Generate a random Gaussian matrix with shape (n, k + oversample).
    mu = np.mean(matrix, axis=0)

    # calculate the covariance matrix K using the formula
    imt_mat=(matrix - mu.reshape(1, -1))

    K = (imt_mat.T @ imt_mat)/m

    omega = np.random.multivariate_normal(mu,K,k+oversample).T

    # Form the sample matrix Y.
    y = matrix @ omega

    # Compute the QR decomposition of Y.
    q, _ = np.linalg.qr(y)

    # Compute the matrix B = Q^T @ A.
    b = q.T @ matrix

    # Compute the SVD of B.
    print(b.shape)
    u_tilde, s, v = svd(b, full_matrices=False)

    # Compute the matrix U = Q @ U_tilde.
    u = q @ u_tilde

    # Return the SVD factors.
    return u[:,:k], s[:k], v[:k,:]
```

3 Methodology and Results

3.1 Generalisation of Randomised SVD

3.1.1 Step 1: Compute Covariance Matrix

The first step is to compute the mean vector μ and the covariance matrix K of the input matrix A:

$$\mu = \frac{1}{m} \sum_{i=1}^{m} a_i \tag{1}$$

$$K = \frac{1}{m-1} \sum_{i=1}^{m} (a_i - \mu)(a_i - \mu)^T \tag{2}$$

where a_i denotes the i-th row of A, and m is the number of rows in A.

3.1.2 Step 2: Compute Omega Matrix

The next step is to compute an omega matrix Ω that satisfies the equality $A\Omega = QR$, where Q is an $m \times (k+p)$ orthonormal matrix and R is an $(k+p) \times n$ upper-triangular matrix. The omega matrix can be computed as follows:

$$\Omega = K^{1/2} V \tag{3}$$

where V is an $(n, k+p)$ matrix whose columns are drawn from a standard normal distribution, and $K^{1/2}$ denotes the matrix square root of K.

3.1.3 Step 3: Compute QR Decomposition

Using the computed omega matrix Ω, the QR decomposition of $A\Omega$ can be computed as follows:

$$Y = A\Omega = QR \tag{4}$$

where Q is an $m \times (k+p)$ orthonormal matrix and R is an $(k+p) \times (k+p)$ upper-triangular matrix.

3.1.4 Step 4: Compute Low-Rank Approximation

Finally, the low-rank approximation of A can be computed using the computed QR decomposition as follows:

$$A \approx Q_k R_k V_k^T \tag{5}$$

where Q_k is the first k columns of Q, R_k is the upper $k \times k$ block of R, and V_k is the $k \times n$ matrix that solves $R_k V_k = K^{1/2} Q_k^T A$.

In our implementation, we have computed the SVD of $B = Q^T A$, where Q is the orthogonal matrix from the QR decomposition of Y, and then computed U by multiplying Q with the left singular vectors of B. Finally, we have returned the SVD factors U, S, and V^T of the low-rank approximation of A.

3.2 GRSVD for Hilbert-Schmidt

Hilbert-Schmidt operators are a class of bounded linear operators on a Hilbert space. Given a kernel function $K(x, y)$, the corresponding Hilbert-Schmidt operator T_K is defined as:

$$T_K f(x) = \int K(x, y) f(y) dy \tag{6}$$

In other words, T_K maps a function $f(x)$ to a new function that is obtained by integrating $f(x)$ against the kernel $K(x, y)$.

One can draw an analogy between Hilbert-Schmidt operators and matrices. In particular, given a matrix A, we can view it as a linear operator that maps a vector v to another vector Av. Similarly, given a kernel function $K(x, y)$, we can view the corresponding Hilbert-Schmidt operator T_K as a linear operator that maps a function $f(x)$ to another function $T_K f(x)$.

To convert a kernel function K(x,y) into a matrix, we can evaluate the kernel function at all possible pairs of inputs (x_i, x_j) in our dataset. This results in an n x n matrix, where n is the number of data points in our dataset.

The GRSVD algorithm for Hilbert-Schmidt operators is similar to the RSVD algorithm for matrices. Given a kernel function $K(x, y)$ and a positive integer k, the algorithm proceeds as follows:

1. Generate a matrix Ω of size $n \times (k + p)$ as a multivariate normaly distributed matrix in a similar way as we did in the GRSVD for matrices , using covariance matrix and mean function
2. Compute $Y = K \cdot G$.
3. Compute the QR decomposition of $Y = Q \cdot R$.
4. Compute $B = Q^T \cdot K$.
5. Compute the SVD of $B = U \cdot \Sigma \cdot V^T$.
6. Compute $U_k = Q \cdot U$.

Here, p is a user-specified oversampling parameter, and U_k denotes the matrix containing the first k columns of U. The output of the RSVD algorithm for Hilbert-Schmidt operators is U_k, Σ, and V^T, which are approximations to the left singular vectors, singular values, and right singular vectors of the Hilbert-Schmidt operator T_K.

Note that, unlike matrices, Hilbert-Schmidt operators are infinite-dimensional, and so it is not possible to compute their singular values exactly. However, the RSVD algorithm provides a computationally efficient way to approximate the leading singular vectors and values of Hilbert-Schmidt operators in practice.

3.3 Results

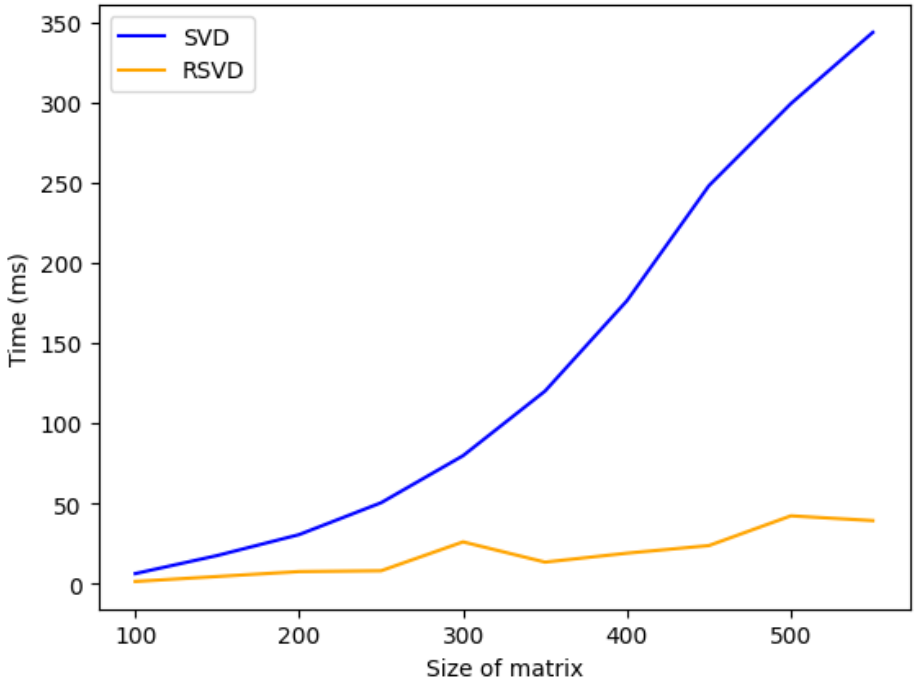

Figure 1: Comparison of time taken by RSVD and SVD

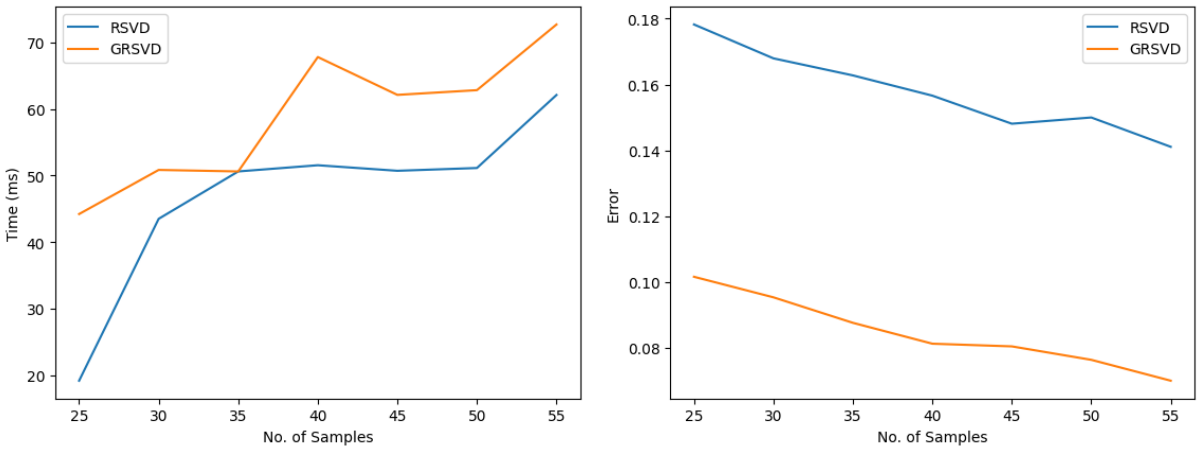

Figure 2: Comparison of Time Taken and Error b/w RSVD and GRSVD

All the results we have shown can be found with the code in the GitHub repository that is linked here.

4 Relevance of the work with the course

The Generalization of Randomized Singular Value Decomposition research paper heavily relies on linear algebra concepts. The paper uses the randomized SVD algorithm, which involves factorizing a given matrix into two matrices, typically referred

to as U and V, and a diagonal matrix, known as the singular value matrix. The singular value matrix contains the singular values of the original matrix and captures its intrinsic properties.

Linear algebra concepts such as matrix factorization and eigenvalue decomposition are used to obtain the randomized SVD. Specifically, the randomized SVD algorithm approximates the singular value matrix using a subset of the original matrix's columns. The approximation is based on the fact that the columns of the original matrix span a subspace of lower dimensionality, which can be approximated using a smaller subset of the columns.

The paper further extends the randomized SVD algorithm by introducing probabilistic guarantees for its performance. These guarantees rely on the concentration of measure phenomenon, which is a concept from linear algebra that relates to the concentration of random variables around their mean values. The guarantees ensure that the randomized SVD algorithm can approximate the singular values of a given matrix with high probability.This paper also mentioned multivariate Gaussian random input vectors that have a general symmetric positive semi-definite covariance matrix and Positive semi-definite matrices are taught during the course.

Overall, the Generalization of Randomized Singular Value Decomposition research paper demonstrates the importance of linear algebra concepts in solving real-world problems in data science. The randomized SVD algorithm and its extensions rely heavily on matrix factorization, eigenvalue decomposition, and other linear algebra concepts.

5 Future Extensions

There are several potential directions for future work based on the research presented in the *Generalization of Randomized Singular Value Decomposition* paper. Some possible areas for extending this work include:

1. Improving the performance of the randomized SVD algorithm: While the randomized SVD algorithm is computationally efficient and effective in many cases, there may be ways to further optimize its performance. Researchers could investigate new sampling techniques or modifications to the algorithm that improve its accuracy or reduce its computational complexity.

2. Applying the randomized SVD algorithm to new domains: The randomized SVD algorithm has been successfully applied in a range of domains, from recommender systems to image processing. Researchers could explore new applications of the algorithm in fields such as natural language processing, signal processing etc.,

3. Developing new algorithms for low-rank matrix approximation: The randomized SVD algorithm is just one of many techniques for approximating low-rank matrices. Researchers could explore other algorithms or combinations of algorithms that improve upon the randomized SVD's performance or applicability in certain domains.

4. Investigating the theoretical properties of the algorithm: While the paper presents probabilistic guarantees for the randomized SVD algorithm, there may be other theoretical properties of the algorithm that could be explored. For example, researchers could investigate the algorithm's stability, its convergence properties, or its sensitivity to noise.

Overall, the research presented in the *Generalization of Randomized Singular Value Decomposition* paper provides a strong foundation for future work in the field of low-rank matrix approximation and related domains. By building upon this work, researchers can continue to develop new algorithms and applications that push the boundaries of what is possible with data analysis and machine learning.