DiffRed: Dimensionality Reduction guided by stable rank

Prarabdh Shukla
Department of CSE,
Indian Institute of Technology, Bhilai
Bhilai, India

Gagan Raj Gupta
Department of CSE,
Indian Institute of Technology, Bhilai
Bhilai, India

Kunal Dutta
Institute of Informatics,
University of Warsaw,
Warsaw, Poland

Abstract

In this work, we propose a novel dimensionality reduction technique, *DiffRed*, which first projects the data matrix, A, along first k_1 principal components and the residual matrix A^* (left after subtracting its k_1-rank approximation) along k_2 Gaussian random vectors. We evaluate *M1*, the distortion of mean-squared pair-wise distance, and *Stress*, the normalized value of RMS of distortion of the pairwise distances. We rigorously prove that *DiffRed* achieves a general upper bound of $O\left(\sqrt{\frac{1-p}{k_2}}\right)$ on *Stress* and $O\left(\frac{1-p}{\sqrt{k_2 * \rho(A^*)}}\right)$ on *M1* where p is the fraction of variance explained by the first k_1 principal components and $\rho(A^*)$ is the *stable rank* of A^*. These bounds are tighter than the currently known results for Random maps. Our extensive experiments on a variety of real-world datasets demonstrate that *DiffRed* achieves near zero *M1* and much lower values of *Stress* as compared to the well-known dimensionality reduction techniques. In particular, *DiffRed* can map a 6 million dimensional dataset to 10 dimensions with 54% lower *Stress* than PCA.

1 Introduction

High dimensional data is common in biological sciences, fin-tech, satellite imaging, computer vision etc. which make tasks such as machine learning, data visualization, similarity search, anomaly detection, noise removal etc. very difficult. Dimensionality reduction is a pre-processing step to obtain a low-dimensional representation while preserving its "structure" and "variation". In this work, our focus is on the development of efficient dimensionality reduction algorithms that map D-dimensional data in $\mathbb{R}^D$ to $\mathbb{R}^d$ where d, the target dimension is a small number. This decreases the amount of training time and computation resources required for the above tasks.

We consider two metrics to quantify distortion, which we aim to minimize. The first metric *M1* is the distortion of mean-squared pair-wise distances. Minimizing *M1* ensures that the low-dimensional representation has similar "Energy" or "total variance" as the original data. While this is important, we also need to preserve both short and long pair-wise distances for preserving importance structures such the nearest-neighbors and clusters in the data. This is accomplished by minimizing *Stress* [Kruskal, 1964], the normalized value of RMS distortion of the pairwise distance by the mapping. While *M1* may be minimized by a simple scaling of data points, doing so may distort other metrics such as *Stress.*

Traditional dimensionality reduction techniques such as PCA, SVD, MDS [Xu et al., 2008, Deegalla and Bostrom, 2008] use the structure of data to determine directions along which data should be projected. One identifies the "elbow" in a *scree plot* to choose the number of principal components. Beyond this, one gets diminishing returns and is forced to either choose a large target dimension or accept high distortion. In contrast to these approaches, one can use data-agnostic Gaussian random maps [Bingham and Mannila, 2001] which minimize distortion of pair-wise distances. It was generally thought that to guarantee low distortion, a large number of target dimensions are required by Gaussian random maps. In a recent work, [Bartal et al., 2019] obtained bounds of the form $O\left(\frac{1}{\sqrt{d}}\right)$ on *Stress* when using Gaussian random maps of any arbitrary dimension d. They also demonstrated that PCA can produce an embedding with *Stress* value being far from optimum and random maps can achieve better performance.

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

We propose a novel approach to dimensionality reduction that uses the *stable rank* (Def in Sec 3) of the data. The stable rank of a dataset gives an idea of directional spread in the data. It is always greater than 1 and less than the actual rank of the data. If the data is spread along various directions, its stable rank will be high, and if it is concentrated along a few directions only, then the stable rank will be low (refer Figure 1). Intuitively, for datasets with low stable rank, PCA is more effective. Our findings reveal a fresh perspective: Random Maps are more effective for high stable rank datasets, as opposed to the conventional belief that Random Maps are data-agnostic.

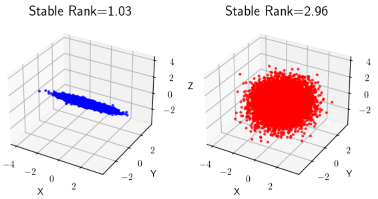


Figure 1: Stable rank as a measure of ”spread” in data in a 3-D example.

In this work, we prove rigorously using Hanson Wright inequality [Rudelson and Vershynin, 2013], that *M1* can be bounded by $O\left(\frac{1}{\sqrt{d\rho(A)}}\right)$ where $\rho(A)$ is the stable rank (Def in Sec 3) of the data matrix A. Thus, if stable rank is high, we can guarantee low distortion for small values of d. Empirically, we observe a similar behavior with respect to *Stress*. Since all input data matrices may not have high stable rank, we subtract the best k-rank approximation of input data matrix and obtain the residual matrix A^*. Empirically, we observe that for most common high-dimensional datasets, A^* has a higher stable rank than A and using random maps for dimensionality reduction will minimize its distortion. *DiffRed* leverages these insights and first projects the data along first k_1 principal components such that the fraction of variance, p explained by them is high. In the next step, it projects A^* along k_2 Gaussian random vectors. We make sure that these two projections lie in orthogonal subspaces, which plays a crucial role in obtaining a tighter upper bound of $O\left(\sqrt{\frac{1-p}{k_2}}\right)$ for *Stress*. This gives us a good analytical trade-off between the number of principal components and random vectors used to minimize *Stress* while keeping the target dimension $d = k_1 + k_2$ small. We demonstrate that *DiffRed* is effective on high dimensional datasets. It preserves global structure even with low target dimensions by carefully choosing k_1, such that the stable rank of the residual matrix is high and the theoretical bound is minimized.

To summarize, our contributions in this paper are as follows:

- We develop a new dimensionality reduction algorithm, *DiffRed* that combines Principal Components with Gaussian random maps in a novel way to achieve tighter upper bounds on both *M1* and *Stress* metrics.

- To the best of our knowledge, we are the first to have incorporated a metric involving the structure of the data matrix (Stable Rank) and impact of Monte-Carlo iterations in the bound of *M1* and *Stress* for *DiffRed* and Random maps. This allows d to be small and explains why random maps often work well in practice for high-dimensional datasets.

- Fast implementation of *DiffRed* and extensive experiments to demonstrate that it achieves better performance than various commonly used dimensionality reduction techniques on real-life datasets.

2 Related Work

Dimensionality reduction has been studied by [Cayton and Dasgupta, 2006, Censi and Scaramuzza, 2013, Fukumizu et al., 2004, Quist and Yona, 2004] in the context of machine learning. In the broader context of metric embedding, there is a large body of work in diverse research areas demonstrating the practicality of various dimensionality reduction and metric embedding techniques, e.g. [Ng and Zhang, 2002]. Dimensionality reduction techniques can be broadly classified as (i) linear and (ii) non-linear.

The most common linear dimensionality reduction technique is PCA (Principal Component Analysis) [F.R.S., 1901], although several other classical techniques such as factor analysis and multidimensional scaling, are also used [Spearman, 1904, Torgerson, 1952]. However, these linear techniques are not very good at handling non-linear data, e.g. when the data is lying on a low-dimensional manifold in a high-dimensional ambient space – often referred to as the *manifold hypothesis* [Niyogi et al., 2008].

In contrast, non-linear techniques such as Kernel PCA, Isomap, Diffusion maps, or Locally Linear Embedding, etc. can be quite effective at handling particular types of non-linear data, such as convex or Gaussian data, and are being used more and more in recent applications [Van Der Maaten et al., 2009]. However, in general, the technique used needs to be tailored to the application, as certain maps can be quite bad for certain types of datasets.

In this scenario, the method of random projections [Bingham and Mannila, 2001] is a linear dimensionality reduction technique, which has the advantages of genericity, low computational complexity, low memory requirement, and ability to handle some degree of non-linearity, e.g. data lying in low-dimensional manifolds – in contrast to PCA which, for high-dimensional data, requires significant computational time and memory, and cannot handle non-linear data. Various time-efficient randomized variants of PCA and SVD have been proposed, such as [Halko et al., 2010,Feng et al., 2018]. Similarly, faster variants of the Random Map have been proposed [Ailon and Chazelle, 2009]. Recently, [Fandina et al., 2022] have presented a fresh analysis of the Fast JL transform, showing an improvement in embedding time. [Schmidt, 2018] is one of the few comparative studies involving PCA and random projections.

The notion of *stable rank* (or numerical rank) of a matrix is a robust version of the rank of a matrix, and is not affected significantly by very small singular values. It was first introduced in [Rudelson and Vershynin, 2007] who used it to obtain low-rank approximations of matrices. Since then it has found several applications in numerical linear algebra e.g. [Indyk et al., 2019,Cohen et al., 2016,Kasiviswanathan and Rudelson, 2018]. However to the best of our knowledge, it has not yet been used to obtain stronger dimensionality reduction bounds. Moreover, in the known applications of stable rank, it is advantageous to have low stable rank, e.g. to obtain low rank approximations of matrices. On the other hand, our application utilizes *high* stable rank, which gives a stronger concentration bound for the mapped vectors, allowing us to choose a lower target dimension.

Stress as a metric has been used in a variety of applications such as MDS [Kruskal, 1964], psychology [Bor, 2005] and also surface matching [Bronstein et al., 2006] which is applied to 3D face recognition and medical imaging. Various quantitative studies of dimensionality reduction such as [Espadoto et al., 2019, Yin, 2007, Liu et al., 2017] have also considered *Stress* to be an important distortion metric to measure projection quality.

Stochastic embedding methods such as T-SNE [van der Maaten and Hinton, 2008] are also popular for visualization of datasets. However, they can cause large distortion and are rarely used for tasks such as machine learning, similarity search, anomaly detection, noise removal etc. UMAP [McInnes et al., 2020] is another useful visualization technique that performs manifold learning. Unlike T-SNE, it has no restriction on the target dimension.

In recent years, [Espadoto et al., 2019] is the most comprehensive survey of Dimensionality Reduction techniques. They work with 18 datasets, 44 techniques, and 7 quality metrics to create a projection assessment benchmark that helps answer which dimensionality reduction algorithm applies to a given context. In our experiments, we compare *DiffRed* to the best techniques reported in their survey.

3 Problem Formulation

Let us now formally define the problem of dimensionality reduction of a data matrix A to obtain embedding matrix $\tilde{A}$ while minimizing *M1* and *Stress*.

Definition 1 (Data Matrix)**.** A matrix in $\mathbb{R}^{n\times D}$ whose rows are n points $\mathbf{x}_1^\top, \dots, \mathbf{x}_n^\top$ in $\mathbb{R}^D$ is called a Data Matrix and is denoted by A. Without loss of generality, we will assume that A has rows with mean zero and unit variance[1].

Definition 2 (Embedding Matrix)**.** Given a data matrix $A \in \mathbb{R}^{n\times D}$, its corresponding embedding matrix $\tilde{A} \in \mathbb{R}^{n\times d}$ is a matrix whose rows $\tilde{\mathbf{x}}_1^\top, \dots, \tilde{\mathbf{x}}_n^\top$ are embeddings of the rows of A onto $\mathbb{R}^d$.

From now on, d shall denote the target dimension and D shall denote the original dimension unless specified otherwise.

Definition 3 (Stable Rank)**.** For a given matrix A, let $\sigma_1, \sigma_2, \dots$ be the singular values ordered from the highest to the lowest in magnitude. Then, the stable rank $\rho(A)$ of A is defined as

$$\rho(A) = \frac{\sum_{i=1}^{rank(A)} \sigma_i^2}{\sigma_1^2}$$

Definition 4 (*M1* Distortion)**.** For data matrix $A \in \mathbb{R}^{n\times D}$ (whose rows $\mathbf{x}_1^\top, \dots, \mathbf{x}_n^\top \in \mathbb{R}^D$ are the data points) and its corresponding embedding matrix $\tilde{A} \in \mathbb{R}^{n\times d}$ (whose rows $\tilde{\mathbf{x}}_1^\top, \dots, \tilde{\mathbf{x}}_n^\top$ in $\mathbb{R}^d$ are the low dimensional embeddings), the *M1* distortion (Λ_{M_1}) is given by:

$$\Lambda_{M1}(A, \tilde{A}) = \left|1 - \frac{||\tilde{A}||_F^2}{||A||_F^2}\right| = \left|1 - \frac{\sum_{i=1}^n ||\tilde{\mathbf{x}}||_2^2}{\sum_{i=1}^n ||\mathbf{x}||_2^2}\right|$$

Definition 5 (*Stress*)**.** For a set of points $\mathbf{x}_1, \dots, \mathbf{x}_n$ in D-dimensional space $\mathbb{R}^D$ and their respective low dimensional embeddings, $\tilde{\mathbf{x}}_1^\top, \dots, \tilde{\mathbf{x}}_n^\top$ in $\mathbb{R}^d$, we define the *Stress* Λ_S as:

$$\Lambda_S = \left(\frac{\sum_{i,j}(||\mathbf{d}_{ij}|| - ||\tilde{\mathbf{d}}_{ij}||)^2}{\sum_{i,j}||\mathbf{d}_{ij}||^2}\right)^{\frac{1}{2}}$$

where, $\mathbf{d}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ and $\tilde{\mathbf{d}}_{ij} = \tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j$

[1] This assumption will help us in proving Lemma 6

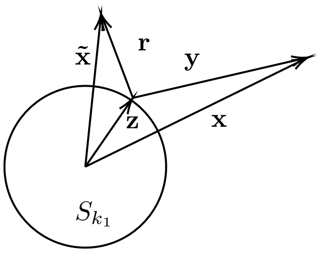


Figure 2: *DiffRed* algorithm maps vector $\mathbf{x} \in \mathbb{R}^D$ to $\tilde{\mathbf{x}} \in \mathbb{R}^{k_1+k_2}$ while preserving its component $\mathbf{z}$ in the best-fit-subspace S_{k_1}. $\mathbf{r}$ and $\mathbf{y}$ are orthogonal to $\mathbf{z}$.

Definition 6 (p)**.** $p = \frac{\sum_{i=1}^{k_1} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}$ represents the fraction of variance explained by k_1 principal components of A.

4 *DiffRed* Algorithm and Its Analysis

In this section, we formally describe the *DiffRed* algorithm, which uses a combination of principal components and Gaussian random maps to provide provable low distortion. In the pseudocode below, SVD has been employed for PCA and k-rank approximation. Alternatively, eigen-decomposition can also be used.

```
Algorithm 1: DiffRed Algorithm
Input: $A$, $k_1$, $k_2$, $\eta$
compute[2]SVD $A = U\Sigma V^\top$
compute $A_{k_1} \leftarrow \sum_{i=1}^{k_1} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ and $A^* \leftarrow A - A_{k_1}$
Let $V_{k_1}$ be the matrix with the $k_1$ leftmost
  columns of $V$
$Z \leftarrow AV_{k_1}$                    // Project $A$ along $V_{k_1}$
Initialize $min = \infty$
Initialize $T, T_{min} \in \mathbb{R}^{n \times k_2}$
                                 // $\eta$ Monte Carlo iterations
for $i = 0, \cdots, \eta$ do
    Sample $G \in \mathbb{R}^{D \times k_2}$ where $G_{ij} \sim \mathcal{N}(0,1)$ i.i.d.
    $G \leftarrow \frac{1}{\sqrt{k_2}} G$
    $T \leftarrow A^* G$
    if $\Lambda_{M_1}(A^*, T) < min$ then
        $T_{min} \leftarrow T$
$R \leftarrow T_{min}$
      // $T_{min}$ is the projection with least $\Lambda_{M_1}$
$\tilde{A} \leftarrow [Z|R]$
return $\tilde{A}$
```

Each vector $\mathbf{x} \in \mathbb{R}^D$ can be written as the following sum: $\mathbf{x} = \mathbf{z} + \mathbf{y}$. Here, $\mathbf{z} \in S_{k_1} = \text{SPAN}(\mathbf{v}_1, \ldots, \mathbf{v}_{k_1})$ where the v's are the k_1 principal components of the data matrix (which span the row space). $\mathbf{y}$ lies in the residual subspace $\mathbb{R}^D \setminus S_{k_1}$ and is orthogonal to $\mathbf{z}$ by definition. $\mathbf{z}$ is fully preserved during dimensionality reduction chosen by *DiffRed*. Only $\mathbf{y}$ undergoes a projection via random map to give $\mathbf{r}$. Finally, our embedded vector becomes $\tilde{\mathbf{x}} = \mathbf{z} + \mathbf{r}$. Our claim is that the square of the difference between length (norm) of $\mathbf{x}$ and $\tilde{\mathbf{x}}$ is less than that of between $\mathbf{r}$ and $\mathbf{y}$, i.e., $(|\mathbf{x}| - |\tilde{\mathbf{x}}|)^2 \leq (|\mathbf{y}| - |\mathbf{r}|)^2$. PCA and its variants attempt to preserve only $\mathbf{z}$ while neglecting $\mathbf{y}$ completely. *DiffRed* solves this problem elegantly. Increasing k_1 allows us to preserve "longer" $\mathbf{z}$ while increasing k_2 reduces the distortion of $\mathbf{y}$. In the proofs below, these insights are extended to the full data matrix, A.

Lemma 1 presents a tighter upper bound on *M1* for Gaussian random projections using the notion of stable rank and Theorem 2 does the same for *DiffRed*. Corollary 4 analyzes the importance of performing Monte Carle iterations in *DiffRed*. Then, we state a recent result on bounding *Stress* in Theorem 5 [Bartal et al., 2019]. Theorem 7 proves a tighter bound on *Stress* achieved by *DiffRed*.

Lemma 1. There exists a constant $c_1 > 0$, such that given a random matrix G as defined in the *DiffRed* Algorithm 1 and a data matrix A, for all $d \leq D$ and all $\varepsilon \in [0,1]$

$$\mathbb{P}\left[|\|AG\|_F^2 - \|A\|_F^2| \geq \varepsilon \cdot \|A\|_F^2\right] \leq 2. \exp\left(-c_1 \varepsilon^2 d \rho_A\right).$$

Theorem 2 (*M1* Distortion Bound)**.** Given a data matrix $A \in \mathbb{R}^{n \times D}$ and non-negative integers k_1 and k_2, let the application of the *DiffRed* algorithm on A with target dimensions k_1 and k_2 return the embedding matrix $\tilde{A} \in \mathbb{R}^{n \times d}$ where $d = k_1 + k_2$. Then,

$$\mathbb{P}\left[\Lambda_{M1}(A) \geq \varepsilon\right] \leq 2e^{(-\frac{c_1 \varepsilon^2 k_2 \rho(A^*)}{(1-p)^2})}$$

where $c_1 > 0$ is a constant.

The proof of Theorem 2 is provided in the supplementary material.

To minimize $\mathbb{P}\left[\Lambda_{M1} \geq \varepsilon\right]$ (failure probability), the argument in the exponent above needs to be large. This means we can achieve an ε of the order of $\frac{(1-p)}{\sqrt{k_2 \rho(A^*)}}$. Performing Monte Carlo iterations reduces the failure probability considerably and *M1* can be minimized. The following corollary is a direct consequence of Theorem 2

Corollary 3 (*M1* bound for RMap)**.** From Theorem 2, for the case of pure Random Maps($p = 0$,$k_1 = 0$, $k_2 = d$), we have the following bound:

$$\mathbb{P}\left[\Lambda_{M1}(A) \geq \varepsilon\right] \leq 2e^{(-c_1 \varepsilon^2 d \rho(A))}$$

Corollary 4 (*M1* Distortion, Monte Carlo Version)**.** Given a data matrix $A \in \mathbb{R}^{n \times D}$,$k_1$ and k_2, and given

[2] When U and V both are extremely large, a custom power iteration algorithm may be used to calculate only the top k_1 singular vectors and singular values

$\eta > 0$, let the application of the *DiffRed* algorithm on A with target dimensions k_1 and k_2 return the embedding matrix $\tilde{A} \in \mathbb{R}^{n \times d}$ where $d = k_1 + k_2$. Then, the probability that in η Monte Carlo iterations,

$$\begin{aligned} \mathbb{P}\left[\min\{\Lambda_{M1}(A)\} \geq \varepsilon\right] \leq \\ \delta_0^{\eta} \leq \\ \exp\left(-\eta\left(\frac{c_1\varepsilon^2 k_2 \rho(A^*)}{(1-p)^2} - \ln 2\right)\right) \end{aligned}$$

where $\delta_0 := 2\exp\left(-\frac{c_1\varepsilon^2 k_2\rho(A^*)}{(1-p)^2}\right)$.

The next two results analyze the *Stress* Metric, Λ_S. [Bartal et al., 2019] proved the following bound on *Stress* if pure Random Map is applied:

Theorem 5 (Bartal et al. [Bartal et al., 2019])**.** Let $P \subset \mathbb{R}^D$ be a finite point set, $q \geq 2$, and $G : \mathbb{R}^D \to \mathbb{R}^d$ be a Gaussian random map. Then with probability at least $1/2$, the q-norm stress of the point set P under the map G satisfies

$$\begin{aligned} \Lambda_S^{(q)}(P) &\leq 2\sqrt{\frac{3}{e} + \frac{3e^2}{2}}\sqrt{q/d} \\ &\leq 6.2\sqrt{q/d} \quad = \quad O(\sqrt{q/d}). \end{aligned}$$

In particular, the 2-norm stress, $\Lambda_S(P)$, satisfies $\Lambda_S(P) = O(\sqrt{1/d})$.

Lemma 6. Given points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^D$ and data matrix A. Let $\mathbf{d}_{ij} = \mathbf{x}_i - \mathbf{x}_j$, then:

$$\sum_{j<i}^{n} ||\mathbf{d}_{ij}||^2 = n\sum_{j<i}^{n} ||\mathbf{x}_i||^2 = n||A||_F^2$$

Theorem 7 (*Stress* Bound)**.** Given a set of points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, k_1 and k_2, let application of the *DiffRed* algorithm return the points $\tilde{\mathbf{x}_1}, \ldots, \tilde{\mathbf{x}_n}$. Then with probability at least $1/2$,

$$\Lambda_S = O\left(\sqrt{\frac{1-p}{k_2}}\right)$$

Proof. By definition, the value of *Stress* is:

$$\Lambda_S^2 = \frac{\sum_{i,j}(||\mathbf{d}_{ij}|| - ||\tilde{\mathbf{d}}_{ij}||)^2}{\sum ||\mathbf{d}_{ij}||^2}.$$

Now, since $\mathbf{d}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ and $\tilde{\mathbf{d}}_{ij} = \tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j$, i.e., we can break them into two components that are **orthogonal** to each other:

$$\mathbf{d}_{ij} = \mathbf{d}_{ij}^{(Z)} + \mathbf{d}_{ij}^{(Y)} \text{ and } \tilde{\mathbf{d}}_{ij} = \tilde{\mathbf{d}}_{ij}^{(Z)} + \tilde{\mathbf{d}}_{ij}^{(R)}.$$

Here $\mathbf{d}_{ij}^{(Z)}, \tilde{\mathbf{d}}_{ij}^{(Z)} \in S_{k_1}$, the best-fit Subspace of rank k_1 and $\mathbf{d}_{ij}^{(Y)}, \tilde{\mathbf{d}}_{ij}^{(R)} \in \mathbb{R}^d \setminus S_{k_1}$, the residual space. By using first k_1 principal components, *DiffRed* ensures that $\mathbf{d}_{ij}^{(Z)} = \tilde{\mathbf{d}}_{ij}^{(Z)}$. It follows that

$$\begin{aligned} ||\mathbf{d}_{ij}|| - ||\tilde{\mathbf{d}}_{ij}|| &= \\ \sqrt{||\mathbf{d}_{ij}^{(Z)}||^2 + ||\mathbf{d}_{ij}^{(Y)}||^2} - \sqrt{||\mathbf{d}_{ij}^{(Z)}||^2 + ||\tilde{\mathbf{d}}_{ij}^{(R)}||)^2}. \end{aligned}$$

In supplementary material we prove the following useful inequality:

$$(\sqrt{a^2 + b^2} - \sqrt{a^2 + c^2})^2 \leq (b - c)^2 \tag{1}$$

Plugging $a = ||\mathbf{d}_{ij}^{(Z)}|| = ||\tilde{\mathbf{d}}_{ij}^{(Z)}||$, $b = ||\mathbf{d}_{ij}^{(Y)}||$ and $c = ||\tilde{\mathbf{d}}_{ij}^{(R)}||$ helps us obtain the following bound on *Stress*:

$$\Lambda_S^2 \leq \frac{\sum_{i,j}(||\mathbf{d}_{ij}^{(Y)}|| - ||\tilde{\mathbf{d}}_{ij}^{(R)}||)^2}{\sum_{i,j}||\mathbf{d}_{ij}||^2} \tag{2}$$

Using Lemma 6,

$$\sum_{i,j}||\mathbf{d}_{ij}||^2 = n||A||_F^2 \text{ and}$$

$$\sum_{i,j}||\mathbf{d}_{ij}^{(Y)}||^2 = n||A^*||_F^2 = (1-p)||A||_F^2$$

because A^* is the residual matrix. Now, from and these relations, it follows that:

$$\frac{\sum_{i,j}||\mathbf{d}_{ij}||^2}{\sum_{i,j}||\mathbf{d}_{ij}^{(Y)}||^2} = \frac{1}{1-p}$$

Using this in equation 2 we get:

$$\Lambda_S^2 \leq (1-p)\left(\frac{\sum_{i,j}(||\mathbf{d}_{ij}^{(Y)}|| - ||\tilde{\mathbf{d}}_{ij}^{(R)}||)^2}{\sum_{i,j}||\mathbf{d}_{ij}^{(Y)}||^2}\right)$$

The RHS is simply now $(1-p)$ times $\Lambda_S^2(R)$ which is the *Stress* between the residual matrix $A - A_{k_1}$ and the matrix R. Now the statement of the Theorem follows from Theorem 5. ■

Corollary 8 (*Stress* Bound, Monte Carlo version)**.** Given a set of points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, k_1 and k_2, let application of the *DiffRed* algorithm return the points $\tilde{\mathbf{x}_1}, \ldots, \tilde{\mathbf{x}_n}$, and given $\eta > 0$, then the probability that in η Monte Carlo iterations, the *Stress* exceeds $O\left(\frac{1-p}{k_2}\right)$, is at most

$$\mathbb{P}\left[\Lambda_S \geq O\left(\sqrt{\frac{1-p}{k_2}}\right)\right] \leq \exp\left(-\eta \ln 2\right).$$

Complexity Analysis The complexity of the *DiffRed* algorithm 1 is $O(Dn \cdot \min\{D, n\} + \eta n k_2 D)$ which suggests that η can be chosen of the order of $\frac{\min\{D,n\}}{k_2}$ to avoid adding more complexity than what is needed for k_1-rank approximation. (ref. Supplementary Material Section 10)

5 Experiments

We have extensively evaluated *DiffRed* on various real-world datasets for stress and *M1* distortion metrics[3]. We first discuss the datasets, followed by the experimental setup, results, and various inferences.

5.1 Datasets

Name	D	n	Type	ρ	Domain
Bank	17	45211	Low	1.48	Finance
Hatespeech	100	3221	Low	11.00	NLP
F-MNIST	784	60000	Low	2.68	Image
Cifar10	3072	50000	Medium	6.13	Image
geneRNASeq	20.53K	801	Medium	1.12	Biology
Reuters30k	30.92K	10788	Medium	14.50	NLP
APTOS 2019	509k	13000	High	1.32	Healthcare
DIV2K	6.6M	800	Very High	8.39	High Res Image

Table 1: Summary of the datasets used with their respective type based on dimensionality

Table 1 summarizes the datasets used for our experiments. Our datasets span a wide range of dimensionality, application domains and stable ranks. Bank [Moro et al., 2012] is a binary classification dataset of the marketing campaign of a Portuguese banking institution. Fashion MNIST [F-MNIST] [Xiao et al., 2017] is a multiclass classification dataset of grayscale images of 10 different kinds of fashion products. Cifar10 [Krizhevsky, 2009] is a dataset of RGB images of various objects. geneRNASeq [Fiorini, 2016] is a random extraction of gene expressions of patients having five different types of tumors. Reuters30k is the TF-IDF representation of the Reuters-21578 dataset [Lewis, 1997] which is a collection of documents consisting of financial news articles that appeared on Reuters newswire in 1987. APTOS 2019 [Karthik, 2019] is a dataset of retina images used for predicting the severity of diabetic retinopathy. DIV2k [Agustsson and Timofte, 2017a, Agustsson and Timofte, 2017b] is a collection of high-resolution 2K images.

5.2 Experimental Setup

For experimentation, we used a workstation with Intel(R) Xeon(R) Gold 5218 CPU @ 2.30GHz, with NVIDIA RTX A6000 GPU, and a shared commodity cluster. We used Python and slurm-based shell scripting to run our experiments. To speed up our experiments (especially computation of *Stress*), our entire codebase was written to leverage multiprocessing.

Pre-processing: We scale each dataset to zero mean and normalize the examples to be vectors of unit norm. We convert the datasets into their vector representations using various standard techniques like label encoding, tf-idf, etc. wherever applicable.

Computing Embeddings: We have compared *DiffRed* to various dimensionality reduction algorithms. Our choice of algorithms was based on the results presented by [Espadoto et al., 2019]. For each of these techniques, we have tried to use the most appropriate implementation wherever possible. For T-SNE, we used T-SNE CUDA [Chan et al., 2019]: a GPU version of T-SNE to compute the embeddings. For UMap, we used the official UMap implementation. For KernelPCA, SparsePCA and PCA we used *scikit-learn's* [Pedregosa et al., 2011] implementation. For RMap, we have used our own implementation. We have used $\alpha = 20$ Gaussian RMaps on each target dimension with the same multiplicative factor and hyperparameters as specified in the *DiffRed* algorithm [1]. We have used $\alpha = 20$ random maps to build and report our 95% confidence interval. For generating random Gaussian vectors and for our own *DiffRed*, we have mainly relied on numPy's routines. In our experiments, we have done hyperparameter tuning using a grid search to justify our theory and show various observations. We have taken the best hyperparameters reported by [Espadoto et al., 2019] as the starting point for the grid search. Finally, after hyperparameter tuning, we compare *DiffRed* to the best *Stress* found for each target dimension for each Dimensionality Reduction technique.

5.3 Experiment Results

To evaluate the performance of *DiffRed*, we compute the *Stress* and *M1* distortion metrics on different datasets for different target dimensions. As a part of our experiments, we perform a grid search on different values of k_1 and k_2. We use the results of the grid search to justify our method of choosing k_1 and k_2 for a given target dimension (discussed in Section 5.4) and to validate our theory (discussed in Sections 5.3.1 and 5.3.2).

5.3.1 Insights on *M1*

In this section, we discuss RMap and *DiffRed* in light of Corollary 3 and 4 respectively. The main observations are as follows:

Observation 1 In Figure 3, we see that for a fixed target dimension of 10, datasets[4] with higher stable rank have lower Λ_{M_1}. In Figure 4, we see that for Reuters30k (i.e., fixed stable rank of $\rho = 14.50$), higher target dimensions cause lesser *M1* distortion. These empirical observations are in agreement with Corollary 3, where $\mathbb{P}\left[\Lambda_{M1} \geq \varepsilon\right]$ depends on the negative exponent of $d \times \rho(A)$. Therefore, for minimization of Λ_{M_1} we require either a high stable rank or a high target dimension. Since stable rank incorporates the spread

[3] Code: https://github.com/S3-Lab-IIT/DiffRed

[4] except Bank, which has a low dimensionality to begin with.

Dataset	D	d	Λ_{M_1}						
			DiffRed	PCA	RMap	S-PCA	K-PCA	UMap	T-SNE ($d = 2$)
Bank	17	5	**2.82e-05**	0.54	0.38	0.58	0.95	94.89	2659.70
Hatespeech	100	10	**1.91e-04**	0.66	0.06	0.68	0.99	240.50	2298.09
FMnist	784	10	**1.92e-04**	0.60	0.11	0.64	1.00	241.35	829.54
Cifar10	3072	10	**1.31e-04**	0.49	0.09	0.54	1.00	166.84	604.71
geneRNASeq	20.5k	10	**7.96e-05**	0.94	0.31	0.95	1.00	328.72	8,761.41
Reuters30k	30.9k	10	**1.27e-04**	0.88	0.03	0.88	1.00	196.97	2393.31
APTOS 2019	509k	10	**4.09e-05**	0.81	0.24	-	-	-	-
DIV2k	6.6M	10	**7.07e-05**	0.66	0.05	-	-	-	-

Table 2: Comparison of the *M1* metric. Note that $k_1 = 2$ and $k_2 = 3$ for Bank and $k_1 = 6$ and $k_2 = 4$ for other datasets. For APTOS and DIV2k, *M1* is evaluated for only PCA, RMap, and *DiffRed* due to memory limitations.

in data, we observe that, **contrary to the popular belief, Random Maps are not data agnostic.**

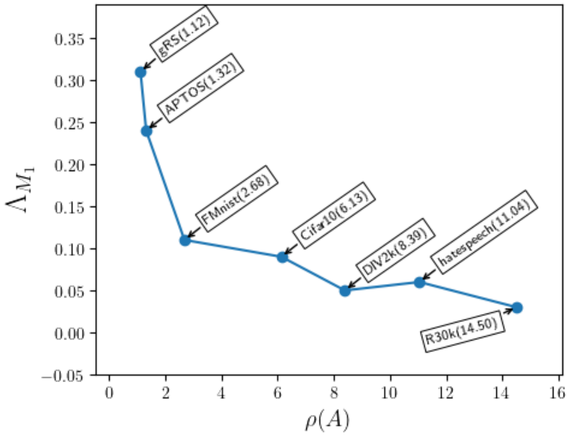


Figure 3: Dependence of *M1* for Random Maps on stable rank described in Corollary 3.($d = 10$) [*gRS*: geneRNASeq, *R30k*: Reuters30k]

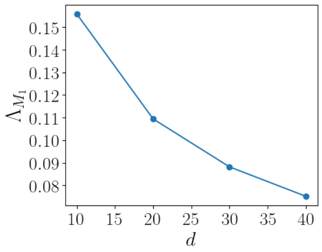


Figure 4: Variation of *M1* with target dimension for Reuters30k

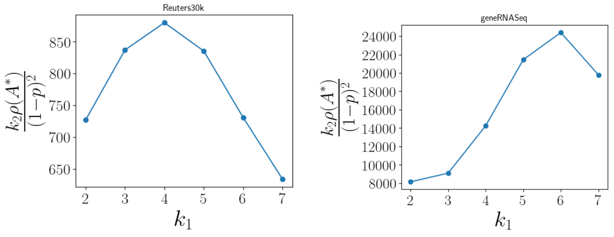
Observation 2 From Table 2, we observe that *DiffRed* has the best values for *M1* across all datasets. Detailed results on *M1* distortion are deferred to the supplementary material.

Observation 3 An interesting observation is that the *M1* metric is insensitive to the choice of k_1 and k_2. To measure the sensitivity of Λ_{M_1} on k_1 and k_2, we define the following quantity, β:

$$\beta = \underset{d \in \{10,20,30,40\}}{\text{Average}} \left(\underset{k_1,k_2}{Var}(\Lambda_{M_1}) \right)$$

β is the average (over target dimensions d) of the variance observed in Λ_{M_1} for different pairs of k_1 and k_2. In essence, β is a measure of the sensitivity of Λ_{M_1} w.r.t. k_1 and k_2 for a given dataset. We evaluated β for different values of k_1 and k_2, observed that the average of β across all datasets, $\langle\beta\rangle \approx 1.54 \times 10^{-6}$ (ref. Table 1 in the supplementary material).

The low sensitivity to k_1 and k_2 follows from Corollary 4, where for a constant η, $\mathbb{P}[\Lambda_{M1} \geq \varepsilon]$ depends on the negative exponent of $\frac{k_2\rho(A^*)}{(1-p)^2}$. We can consider two cases now: (i) k_2 is high and (ii) k_2 is low. As illustrated by Figure 5, the exponent term remains sufficiently high for different stable ranks if k_2 is high (i.e., Case (i) holds). Now, for the second case, we make another observation from Figure 11 that stable rank increases with k_1. Since a low k_2 value implies a high k_1 value ($k_1 = d - k_2$), the exponent term remains high because of the high stable rank.



(a) Reuters30k($\rho = 14.50$) (b) geneRNASeq ($\rho = 1.12$)

Figure 5: The exponent term $\frac{k_2\rho(A^*)}{(1-p)^2}$ remains high for different values of k_1. ($d = 10$)

5.3.2 Insights on *Stress*

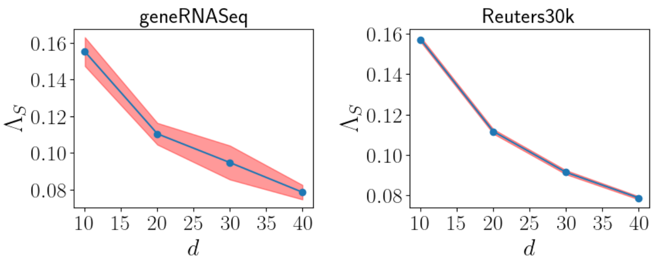
In this section, we discuss the various observations we make in context of the *Stress* metric. The major ob-

Dataset	D	d	Λ_S								
			DiffRed	PCA	RMap	S-PCA	K-PCA	UMap	UMap2	T-SNE ($d=2$)	T-SNE2 ($d=2$)
Bank	17	6	**0.02**	0.03	0.17	0.04	0.47	7.07	0.35	52.44	0.72
Hatespeech	100	10	**0.15**	0.36	0.16	0.36	0.65	5.29	0.46	32.86	0.38
FMnist	784	10	**0.12**	0.19	0.15	0.21	0.68	4.02	0.42	24.49	0.38
Cifar10	3072	10	**0.13**	0.21	0.16	0.24	0.69	1.26	0.60	16.88	0.31
geneRNASeq	20.5k	10	**0.13**	0.21	0.16	0.25	0.70	18.72	0.47	164.89	1.21
Reuters30k	30.9k	10	**0.155**	0.49	0.157	0.49	0.71	3.35	0.44	18.02	0.31
APTOS 2019	509k	10	**0.10**	0.12	0.16	-	-	-	-	-	-
DIV2k	6.6M	10	**0.14**	0.31	0.16	-	-	-	-	-	-

Table 3: Comparison of *DiffRed* with other dimensionality reduction algorithms in context of *Stress*. For *DiffRed*, the best *Stress* from grid search is reported. For APTOS and DIV2k, *Stress* is evaluated for only PCA, RMap, and *DiffRed* due to memory limitations.

servations are as follows:

Observation 1 We observe that *DiffRed* achieves the best values of *Stress* among all other algorithms (Table 3). We evaluated the *Stress* metric on all our datasets for target dimensions 10 to 40 for different values of k_1 and k_2. In Table 3 we compare the empirically obtained best values with the best values for other commonly used Dimensionality Reduction techniques. But we observe that a grid search to determine optimal k_1 and k_2 for a given target dimension and dataset is not required, as we will discuss in Section 5.4 (Choice of hyperparameters).
We note that among all datasets, *DiffRed* consistently achieves the lowest *Stress* values, even when the dimensionality is very high. Sparse-PCA remains close to PCA while Kernel-PCA has a higher *Stress* value. Techniques such as UMap (manifold approximation) and T-SNE (which preserves neighborhoods) do not perform well on distance based metrics. Therefore, to be fair to them, we have included versions UMap2 and T-SNE2 in Table 3. These versions are *energy-matched* with the original data, i.e., they have been re-scaled such that their Frobenius norm (energy) matches that of the original data (i.e., $\Lambda_{M_1} = 0$).



(a) geneRNASeq($\rho = 1.12$) (b) Reuters30k ($\rho = 14.50$)

Figure 6: Λ_S vs d for RMap (95% confidence interval in red).$\alpha = 20$ RMaps were used to generate confidence interval.

Observation 2 In accordance with Theorem 5, Random Maps preserve *Stress* better if more target dimensions are allowed. In Figure 6, we see that RMap benefits in context of the *Stress* metric if more target dimensions are allowed. This is the behavior one would expect from Theorem 5, which bounds *Stress* of random map as $O\left(\sqrt{\frac{1}{d}}\right)$.

Observation 3 From Figure 7, we make two observations: i. PCA benefits in context of *Stress* if more target dimensions are allowed and ii. The general trend of PCA is to perform better for datasets whose stable rank is low to begin with. (see Table 3). Complementary to this observation, we also note from Table 3 that the general trend for RMap is to perform better for datasets that have a high stable rank to begin with.

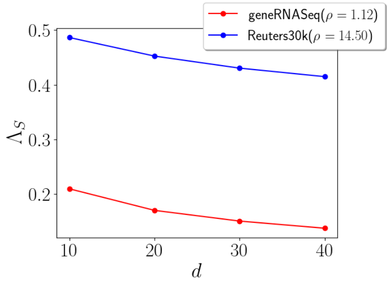
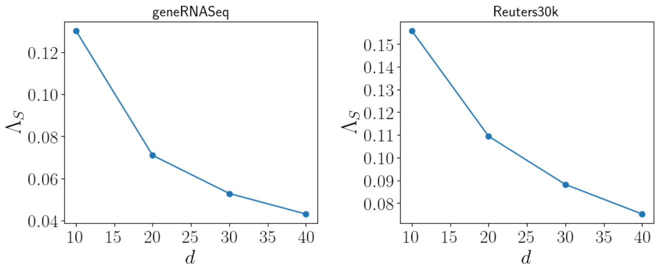


Figure 7: Plot showing how PCA benefits from more target dimensions (d) in context of *Stress*.

Observation 4 From Figure 8, we note that with *DiffRed*, we see an improvement in preserving *Stress* as more target dimensions are allowed (as suggested by Theorem 7, $O\left(\sqrt{\frac{1-p}{k_2}}\right)$). However, in our grid search experiments on *Stress* (as described in Observation 5.3.2 above), we observe that, unlike *M1*, *Stress* is, in fact, sensitive to the choice of k_1 and k_2 (For discussion on the choice of these hyperparameters, ref. Section 5.4 [Choice of hyperparameters]). In conclusion, if a particular downstream tasks benefits from lower

Stress, one may simply allow more target dimensions.



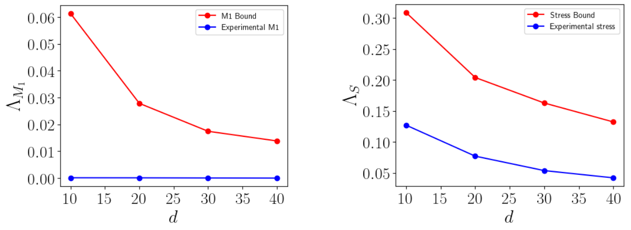
(a) geneRNASeq($\rho = 1.12$) (b) Reuters30k ($\rho = 14.50$)

Figure 8: Λ_S vs d when *DiffRed* is used.

5.4 Discussion

In this section, first we validate our theory results from our experiments and then we discuss the choice of hyperaparameters and possible applications of *DiffRed*

DiffRed generally performs better than RMap because of the additional $\sqrt{1-p}$ factor in the *Stress* bound [Theorem 7]. Additionally, Figure 9 below shows that both our bounds [Theorem 2 and Theorem 7] hold good in our experiments.



(a) Comparison of the *M1 Bound* [Theorem 2] and the experimentally observed *M1*. (b) Comparison of the *Stress bound* [Theorem 7] and the experimentally observed *Stress*.

Figure 9: Comparison between theoretical bounds and empirical observations.

Choice of hyperparameters As described in Section 5.3.1, the *M1* metric is not sensitive to values of k_1 and k_2, therefore, most values of k_1 and k_2 minimize the *M1* distortion. As for *Stress*, we have observed in our grid search experiments [Figure 10] that values of k_1 and k_2 that minimize the theoretical bound usually give *Stress* values that are close to the empirically observed minima. Therefore, one may simply choose the k_1, k_2 pair that minimizes the theoretical bound (i.e., the value of $\sqrt{\frac{1-p}{k_2}}$) for minimizing *Stress*. Computing the bound value is inexpensive, and therefore, one may simply iterate over all combinations of k_1 and k_2 for a given target dimension to find the minima.

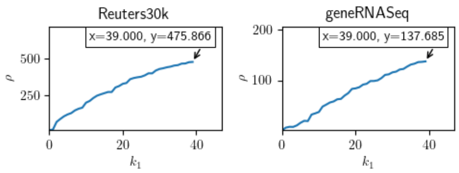


Figure 11: Plots of stable rank vs k_1. Plots for other datasets are in the supplementary material [Figure 13].

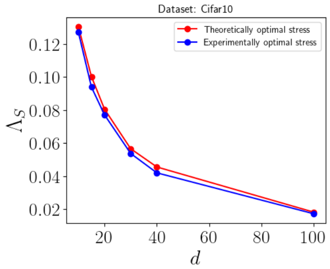


Figure 10: (For Cifar10) [Red]*Stress* for k_1, k_2 values that minimize the bound in Theorem 7. [Blue] *Stress* at empirically optimal k_1 and k_2 values.

Effect of Monte Carlo iterations In our experiments (ref. Supplementary Material) we found that by increasing η, we can find Random Maps that further reduce *M1* metric. It is very interesting to note that such Random Maps achieve lower *Stress*. This justifies the selection of Random Maps based on minimization of *M1* in the *DiffRed* algorithm.

6 Conclusion

In this paper, we design a new dimensionality reduction algorithm, *DiffRed* and obtain new bounds for *M1* and *Stress* metrics that are tighter than currently known results for Random Maps. *DiffRed* uses the notion of stable rank in choosing the directions for projecting the dataset. When the stable rank of a dataset is high to begin with, it emphasizes random maps. When the stable rank of the dataset is low to begin with, it first chooses enough number of principal components so that the stable rank of the residual matrix increases and then uses random maps. Therefore, by incorporating stable rank (structure of data) into our bound, we have shown how dimensionality reduction can be guided by stable rank, thereby reducing the required target dimension. Through extensive experiments on real-world datasets, we have shown that *DiffRed* obtains significant reduction in *M1* and *Stress* as compared to well known dimensionality reduction algorithms. As a part of future work, researchers can explore the effectiveness of *DiffRed* to various applications such as Clustering, Visualization, Nearest Neighbor Search, etc., where high dimensionality often becomes a bottleneck and a global-structure-preserving representation is required in lower dimensions.

Acknowledgement Kunal Dutta's work on this project was supported by the NCN SONATA grant nr. 2019/35/D/ST6/04525.

References

[Bor, 2005] (2005). *MDS as a Psychological Model*, pages 359–388. Springer New York, New York, NY.

[Agustsson and Timofte, 2017a] Agustsson, E. and Timofte, R. (2017a). Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131.

[Agustsson and Timofte, 2017b] Agustsson, E. and Timofte, R. (2017b). Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

[Ailon and Chazelle, 2009] Ailon, N. and Chazelle, B. (2009). The fast johnson–lindenstrauss transform and approximate nearest neighbors. *SIAM Journal on Computing*, 39(1):302–322.

[Bartal et al., 2019] Bartal, Y., Fandina, N., and Neiman, O. (2019). Dimensionality reduction: theoretical perspective on practical measures. In *NIPS*, volume 32.

[Bingham and Mannila, 2001] Bingham, E. and Mannila, H. (2001). Random projection in dimensionality reduction: Applications to image and text data. KDD '01, page 245–250.

[Bronstein et al., 2006] Bronstein, A. M., Bronstein, M. M., and Kimmel, R. (2006). Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences of the United States of America*, 103:1168 – 1172.

[Cayton and Dasgupta, 2006] Cayton, L. and Dasgupta, S. (2006). Robust euclidean embedding. ICML '06, page 169–176, New York, NY, USA.

[Censi and Scaramuzza, 2013] Censi, A. and Scaramuzza, D. (2013). Calibration by correlation using metric embedding from nonmetric similarities. *IEEE Trans. on PAMI*, 35(10):2357–2370.

[Chan et al., 2019] Chan, D. M., Rao, R., Huang, F., and Canny, J. F. (2019). Gpu accelerated t-distributed stochastic neighbor embedding. *Journal of Parallel and Distributed Computing*, 131:1–13.

[Cohen et al., 2016] Cohen, M. B., Nelson, J., and Woodruff, D. P. (2016). Optimal Approximate Matrix Product in Terms of Stable Rank. In Chatzigiannakis, I., Mitzenmacher, M., Rabani, Y., and Sangiorgi, D., editors, *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:14, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[Davidson et al., 2017] Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515.

[Deegalla and Bostrom, 2008] Deegalla, S. and Bostrom, H. (2008). Reducing high-dimensional data by principal component analysis vs. random projection for nearest neighbor classification. In *ICMLA'06*, pages 245–250.

[Espadoto et al., 2019] Espadoto, M., Martins, R. M., Kerren, A., Hirata, N. S. T., and Telea, A. C. (2019). Toward a quantitative survey of dimension reduction techniques. *IEEE Transactions on Visualization and Computer Graphics*, 27:2153–2173.

[Fandina et al., 2022] Fandina, O. N., Høgsgaard, M. M., and Larsen, K. G. (2022). The fast johnson-lindenstrauss transform is even faster.

[Feng et al., 2018] Feng, X., Xie, Y., Song, M., Yu, W., and Tang, J. (2018). Fast randomized pca for sparse data.

[Fiorini, 2016] Fiorini, S. (2016). gene expression cancer RNA-Seq. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5R88H.

[F.R.S., 1901] F.R.S., K. P. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

[Fukumizu et al., 2004] Fukumizu, K., Bach, F. R., and Jordan, M. I. (2004). Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *J. Mach. Learn. Res.*, 5:73–99.

[Halko et al., 2010] Halko, N., Martinsson, P.-G., and Tropp, J. A. (2010). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions.

[Indyk et al., 2019] Indyk, P., Vakilian, A., and Yuan, Y. (2019). Learning-based low-rank approximations. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

[Karthik, 2019] Karthik, Maggie, S. D. (2019). Aptos 2019 blindness detection.

[Kasiviswanathan and Rudelson, 2018] Kasiviswanathan, S. P. and Rudelson, M. (2018). Restricted eigenvalue from stable rank with applications to sparse linear regression. In Bubeck, S., Perchet, V., and Rigollet, P., editors, *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*, volume 75 of *Proceedings of Machine Learning Research*, pages 1011–1041. PMLR.

[Krizhevsky, 2009] Krizhevsky, A. (2009). Learning multiple layers of features from tiny images.

[Kruskal, 1964] Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27.

[Lewis, 1997] Lewis, D. (1997). Reuters-21578 Text Categorization Collection. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C52G6M.

[Liu et al., 2017] Liu, S., Maljovec, D., Wang, B., Bremer, P.-T., and Pascucci, V. (2017). Visualizing high-dimensional data: Advances in the past decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268.

[McInnes et al., 2020] McInnes, L., Healy, J., and Melville, J. (2020). Umap: Uniform manifold approximation and projection for dimension reduction.

[Moro et al., 2012] Moro, S., Rita, P., and Cortez, P. (2012). Bank Marketing. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5K306.

[Ng and Zhang, 2002] Ng, T. and Zhang, H. (2002). Predicting internet network distance with coordinates-based approaches. In *Proceedings.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pages 170–179 vol.1.

[Niyogi et al., 2008] Niyogi, P., Smale, S., and Weinberger, S. (2008). Finding the homology of submanifolds with high confidence from random samples. *Discrete and Computational Geometry*, 39(1-3):419–441.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Quist and Yona, 2004] Quist, M. and Yona, G. (2004). Distributional scaling: An algorithm for structure-preserving embedding of metric and nonmetric spaces. *J. Mach. Learn. Res.*, 5:399–420.

[Rudelson and Vershynin, 2007] Rudelson, M. and Vershynin, R. (2007). Sampling from large matrices: An approach through geometric functional analysis. *J. ACM*, 54(4):21–es.

[Rudelson and Vershynin, 2013] Rudelson, M. and Vershynin, R. (2013). Hanson-wright inequality and sub-gaussian concentration.

[Schmidt, 2018] Schmidt, B. (2018). Stable random projection: Lightweight, general-purpose dimensionality reduction for digitized libraries. *Journal of Cultural Analytics*, 3(1).

[Spearman, 1904] Spearman, C. (1904). "general intelligence", objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292.

[Torgerson, 1952] Torgerson, W. (1952). Multidimensional scaling i. theory and method. *Psychometrika*, 17(4):401–419.

[van der Maaten and Hinton, 2008] van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

[Van Der Maaten et al., 2009] Van Der Maaten, L., Postma, E., and Van den Herik, J. (2009). Dimensionality reduction: a comparative review. *J Mach Learn Res*, 10:66–71.

[Xiao et al., 2017] Xiao et al. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. https://github.com/zalandoresearch/fashion-mnist.

[Xu et al., 2008] Xu, H., Caramanis, C., and Mannor, S. (2008). Robust dimensionality reduction for high-dimension data. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 1291–1298.

[Yin, 2007] Yin, H. (2007). Nonlinear dimensionality reduction and data visualization: A review. *International Journal of Automation and Computing*, 4:294–303.

7 Checklist

1. For all models and algorithms presented, check if you include:
   - (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   - (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   - (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   - (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   - (b) Complete proofs of all theoretical results. [Yes]
   - (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   - (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   - (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   - (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   - (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   - (a) Citations of the creator If your work uses existing assets. [Yes]
   - (b) The license information of the assets, if applicable. [Yes]
   - (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   - (d) Information about consent from data providers/curators. [Not Applicable]
   - (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   - (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   - (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   - (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Supplementary Material

8 Proof of Theorem 2

Proof. From the *DiffRed* algorithm presented in Section 4, $\tilde{A} = [Z|R]$, so that $||\tilde{A}||_F^2 = ||Z||_F^2 + ||R||_F^2$. Using the identity that $||Z||_F^2 = \sum_{i=1}^{k_1} \sigma_i^2$ gives

$$\begin{aligned}\|\tilde{A}\|_F^2 =& \sum_{i=1}^{k_1} \sigma_i^2 + ||R||_F^2 \quad = \quad p\sum_{i=1}^{r} \sigma_i^2 + ||R||_F^2 \\ =& p\|A\|_F^2 + ||R||_F^2.\end{aligned} \tag{3}$$

Now, from Lemma 1, we know that with probability at least $1 - 2 \cdot \exp\left(-c_1\varepsilon^2 d\rho_A\right)$ we have:

$$|(\|R\|_F^2 - \|A^*\|_F^2)| \leq \varepsilon\|A^*\|_F^2$$

Let $\frac{\varepsilon}{1-p} = \varepsilon'$. This implies,

$$\mathbb{P}\left[(1-\varepsilon')\|A^*\|_F^2 \leq \|R\|_F^2 \leq (1+\varepsilon')\|A^*\|_F^2\right] \leq 2 \cdot \exp\left(-\frac{c_1\varepsilon^2 k_2 \rho(A^*)}{(1-p)^2}\right) \tag{4}$$

Now, using this in equation (3), we observe that with probability at least $1 - 2 \cdot \exp\left(-c_1\varepsilon^2 k_2\rho(A^*)/(1-p)^2\right)$:

$$\begin{aligned}\sum_{i=1}^{k_1} \sigma_i^2 + (1-\varepsilon') \sum_{i=k_1+1}^{r} \sigma_i^2 \leq \|\tilde{A}\|_F^2 \\ \leq \sum_{i=1}^{k_1} \sigma_i^2 + (1+\varepsilon') \sum_{i=k_1+1}^{r} \sigma_i^2\end{aligned} \tag{5}$$

Simplifying the upper bound of equation (5) gives us,

$$\|\tilde{A}\|_F^2 \quad \leq \quad \|A\|_F^2 + \varepsilon'(\|A\|_F^2 - \sum_{i=1}^{k_1} \sigma_i^2) \quad = \quad \|A\|_F^2(1+\varepsilon).$$

Simplifying the lower bound of equation (5) gives us $\|\tilde{A}\|_F^2 \geq \|A\|_F^2(1-\varepsilon)$. Thus we get that

$$\begin{aligned}\mathbb{P}\left[\|A\|_F^2(1-\varepsilon) \leq \|\tilde{A}\|_F^2 \leq \|A\|_F^2(1+\varepsilon)\right] \quad \geq \\ 1 - 2\exp\left(-\frac{c_1\varepsilon^2 k_2\rho(A^*)}{(1-p)^2}\right).\end{aligned}$$

Now applying the definition of Λ_{M_1}, we finally get:

$$\mathbb{P}\left[\Lambda_{M1} \geq \varepsilon\right] \leq 2 \cdot \exp\left(-\frac{c_1\varepsilon^2 k_2\rho(A^*)}{(1-p)^2}\right). \blacksquare$$

9 Other proofs

Theorem 9. [Rudelson and Vershynin, 2013] Let G be a $D \times d$ random matrix with entries being independent gaussian random variables G_{ij} with mean zero and variance $1/D$. Let B be an $n \times n$ matrix with entries $b_{ij} \in \mathbb{R}$. Then for every $t \geq 0$, we have

$$\mathbb{P}\left[\left|\mathrm{Tr}(GBG^\top) - \mathbb{E}\left[\mathrm{Tr}(GBG^\top)\right]\right| \geq t\right] \quad \leq \quad 2 \cdot \exp\left(-c \cdot \min\left(\frac{t^2}{\|B\|_F^2}, \frac{t}{\|B\|}\right)\right).$$

Proof (Proof of Lemma 1)**.** The main tool in our proof is the multi-dimensional variant of the *Hanson-Wright inequality* [Rudelson and Vershynin, 2013], stated in Theorem 9 which gives concentration bounds for certain quadratic forms of gaussian random variables.

Let $Z := \|AG\|_F^2 = \mathrm{Tr}(G^\top A^\top AG)$. We shall apply Theorem 9, with G as the $D \times d$ random matrix, and $B = A^\top A$ as a $D \times D$ matrix. We shall also require the following standard observations, which can be easily derived: $\|B\| = \|A\|^2$, and $\|B\|_F^2 \leq \|A\|^2 \cdot \|A\|_F^2$. This gives

$$\begin{aligned}\mathbb{P}\left[|Z - \mathbb{E}\left[Z\right]| \geq t\right] \quad &\leq \quad 2 \cdot \exp\left(-c \cdot \min\left(\frac{t^2}{D\|B\|_F^2}, \frac{t}{\|B\|}\right)\right) \\ &\leq \quad 2 \cdot \exp\left(-c \cdot \min\left(\frac{t^2}{D\|B\|_F^2}, \frac{t}{\|A\|^2}\right)\right).\end{aligned}$$

By linearity of Expectation over Gaussian random variables,
$\mathbb{E}\left[Z\right] = \mathbb{E}\left[\mathrm{Tr}(G^\top A^\top AG)\right] = \mathbb{E}\left[\mathrm{Tr}(AGG^\top A^\top)\right] = d \cdot \mathrm{Tr}(AA^\top)$ and thus,
$\mathbb{E}\left[Z\right] = d\sum_{r=1}^n a_r a_r^\top = \sum_{r=1}^n d\|a_r\|^2 = d\|A\|_F^2$
Now taking $t = \varepsilon\,\mathbb{E}\left[Z\right] = \varepsilon d\|A\|_F^2$, we get

$$\begin{aligned}\mathbb{P}\left[\left|Z - d\|A\|_F^2\right| \geq \varepsilon d\|A\|_F^2\right] \quad &\leq \quad 2 \cdot \exp\left(-c \cdot \min\left(\frac{\varepsilon^2 d^2\|A\|_F^4}{D\|A\|^2\|A\|_F^2}, \frac{\varepsilon d\|A\|_F^2}{\|A\|^2}\right)\right) \\ &= \quad 2 \cdot \exp\left(-c \cdot \min\left(\frac{\varepsilon^2 d^2\|A\|_F^2}{D\|A\|^2}, \frac{\varepsilon d\|A\|_F^2}{\|A\|^2}\right)\right) \\ &\leq \quad 2 \cdot \exp\left(-c\left(\frac{\varepsilon^2 d\|A\|_F^2}{\|A\|^2}\right)\right),\end{aligned}$$

where the last line follows by observing that $d/D \leq 1$ for $d \leq D$. ■

Proof (Proof of Inequality 1 (Theorem 7))**.** We have:

$$a^2(c^2 + b^2 - 2bc) = a^2(b-c)^2 \geq 0 \implies a^2c^2 + a^2b^2 \geq 2a^2bc$$
$$a^4 + a^2c^2 + a^2b^2 + b^2c^2 \geq a^4 + b^2c^2 + 2a^2bc = (a^2 + bc)^2$$

Since a, b, c are non negative, we can take a square root.

$$\sqrt{a^2+b^2}\sqrt{a^2+c^2} \geq a^2 + bc$$
$$2a^2 - 2\sqrt{a^2+b^2}\sqrt{a^2+c^2} \leq -2bc$$
$$a^2 + b^2 + a^2 + c^2 - 2\sqrt{a^2+b^2}\sqrt{a^2+c^2} \leq b^2 + c^2 - 2bc$$
$$\left(\sqrt{a^2+b^2} - \sqrt{a^2+c^2}\right)^2 \leq (b-c)^2$$

Proof (Proof of Lemma 6)**.** As the data is centered, the sum $\sum_{j=1}^n \mathbf{x}_j = 0$. We have: $\sum_{j<i}^n ||\mathbf{d}_{ij}||^2 = n\sum_{i=1}^n ||\mathbf{x}_i||^2 - \sum_{i=1}^n \mathbf{x}_i$

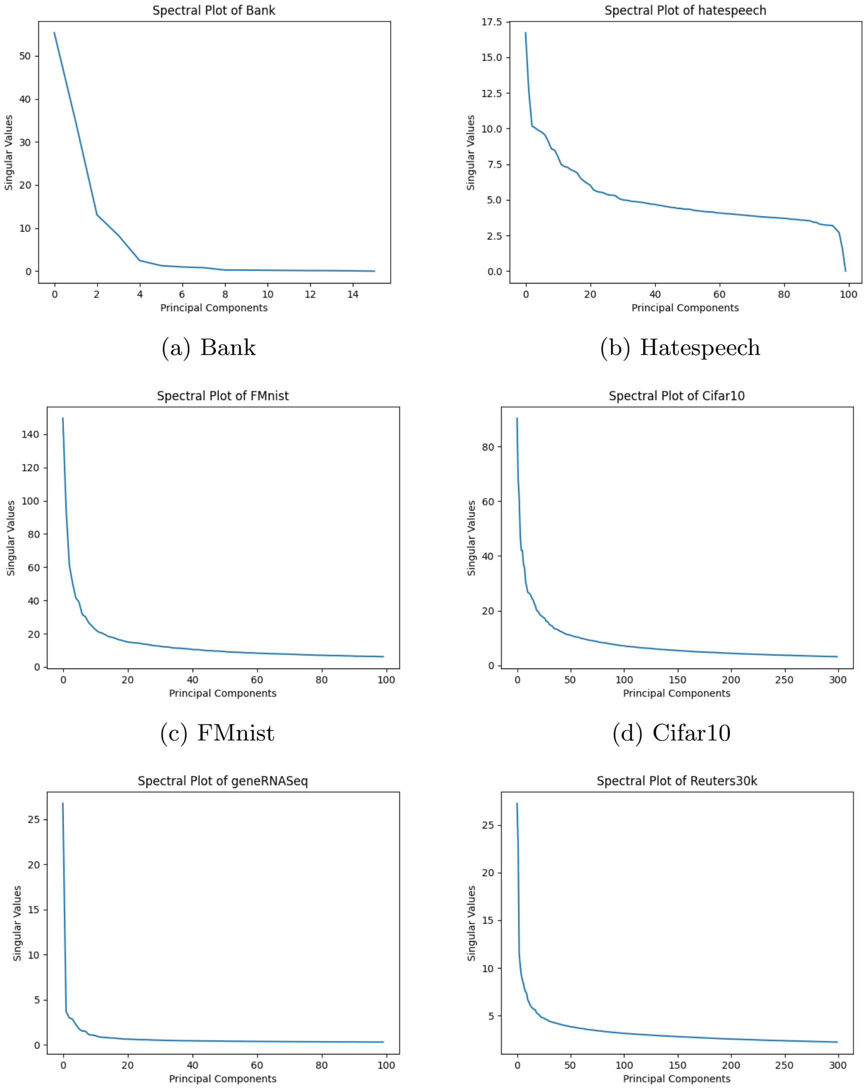
10 Complexity Analysis of the *DiffRed* Algorithm 1

Based on the algorithm description given previously, the running time complexity of *DiffRed* can be obtained as follows. We first obtain a k_1-rank approximation of the $n \times D$ data matrix using the singular value decomposition. This takes $O(nD^2)$ time. Next, we generate and apply a random $k_2 \times D$ Gaussian matrix, which can be done in time $O(nk_2D)$. For η Monte Carlo iterations, this becomes $O(\eta nk_2D)$. Thus, the total time complexity comes to $O(nD^2 + \eta nk_2D)$. For the case when $D \gg n$, we work with $A^\top$, and thus get a complexity of $O(Dn^2 + \eta nk_2D)$. So the overall complexity can be summarized as $O(Dn \cdot \min\{D, n\} + \eta nk_2D)$.

11 Detailed Experiment Results [From Section 5.3]

11.1 *Stress* and *M1*: Datasetwise results

In this section, we present a dataset wise summary of the results of our experiments on *M1* and *Stress* and in the later sections, we present the full grid search results. Figure 12 shows the plots of the singular values of all the datasets.



(a) Bank

(b) Hatespeech

(c) FMnist

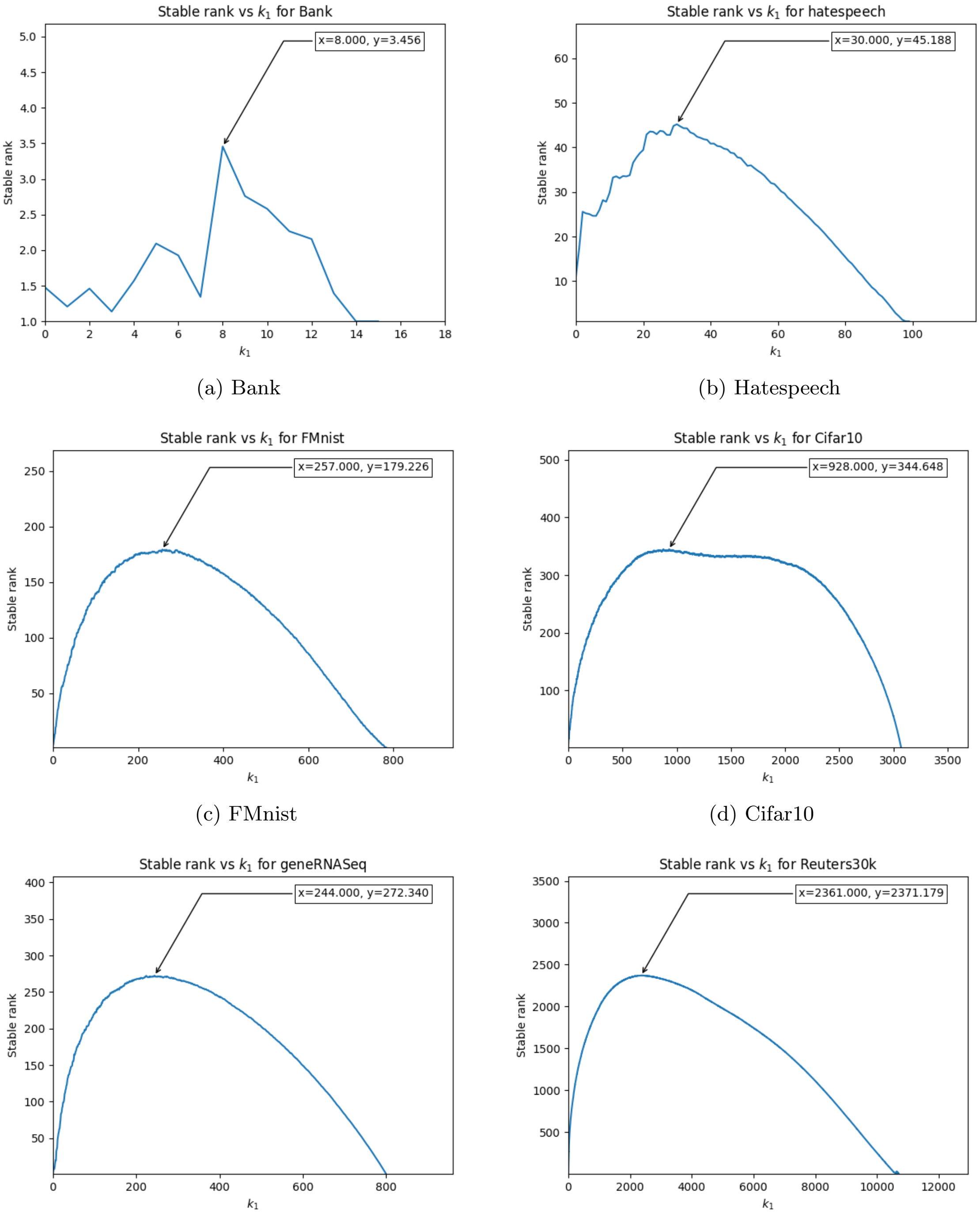
(d) Cifar10

(e) geneRNASeq

(f) Reuters30k

Figure 12: Plots showing the spectral plots of all the datasets

Figure 13 shows how the stable rank of the residual matrix for all datasets increases as more directions of variance are removed (i.e., k_1) so long as the number of components removed remains well within the range of a practically required dimensionality (< 100 for high dimensionality datasets). From our datasets, we note that for Bank and hatespeech, the starting dimensionality itself is low (17 and 100 respectively) and therefore the peak of the curve occurs earlier.



(a) Bank

(b) Hatespeech

(c) FMnist

(d) Cifar10

(e) geneRNASeq

(f) Reuters30k

Figure 13: Plots showing the stable rank vs. k_1 for all the datasets

11.1.1 Bank

Bank [Moro et al., 2012] is a binary classification dataset of a Portugese bank's marketing campaign. The goal of the classification is to predict if a client will subscribe to a term deposit or not. The dataset has a low dimensionality of 17 which puts it out of the curse of dimensionality regime. We account for this low dimensionality in our experiments by exploring only low target dimensions in the range 1 to 8. It also has a relatively low stable rank of 1.48. Figure 12a shows the singular value plot for Bank and Figure 13a shows the stable rank plot. Table 6 and Table 14 show the results of our grid search experiments on *Stress* and *M1* metrics to find the optimal values of k_1 and k_2. The rows having the minimum metric value are marked in bold.

11.1.2 Hatespeech

Hatespeech [Davidson et al., 2017] is a dataset of tweets labelled according to different types of offensive content. For our experiments we use the `.npy` files provided by [Espadoto et al., 2019] on their paper website [5]. The dataset has a dimensionality of 100 and a stable rank of 11. Figure 12b shows the plot of singular values and Figure 13b shows the stable rank plot. Tables 7 and 15 show the results of our experiments on *M1* and *Stress*.

11.1.3 FMnist

Fashion-MNIST [Xiao et al., 2017] (abbreviated as FMnist in our paper) is an image dataset consisting of 60,000 images of fashion images belonging to 10 different classes. Each image is a 28 by 28 grayscale image which can be represented as a 784 dimensional vector. The dataset has a stable rank of 2.68. Figure 12b shows the singular value plot and Figure 13b shows the stable rank plot for FMnist. Tables 8 and 16 show our experiment results.

11.1.4 Cifar10

Cifar10 [Krizhevsky, 2009] is a dataset of 60,000 color images belonging to 10 classes. Each image is a 32 by 32 image with 3 channels, therefore, each image can be represented as a 3072 dimensionality vector. The stable rank of the dataset is 6.13. Figures 12d and 13d are the relevant spectral and stable rank plots. Tables 9 and 17 show the results of our experiments on *Stress* and *M1*.

11.1.5 geneRNASeq

geneRNASeq [Fiorini, 2016] is a dataset of gene expressions with the aim of classifying 5 types of tumor. It has a dimensionality of 20531 and a stable rank of 1.12 which is also the lowest stable rank among all datasets. Figures 12e and 13e show the spectral and stable rank plots respectively. Tables 10 and 18 show the results of our grid search experiments on k_1 and k_2.

11.1.6 Reuters30k

Reuters30k is a TF-IDF vector respresentation of the Reuters newswires dataset [Lewis, 1997] which is a collection of news articles belonging to different topics. We use this[6] huggingface version of the dataset. To generate TF-IDF representation, we use scikit-learn's `TfidVectorizer`. The dimensionality of the dataset becomes 30,916 after this preprocessing. It also has the highest stable rank (14.50) among all datasets. Figures 12f and 13f show the relevant spectral and stable rank plots. Tables 11 and 19 show the results of our experiments on *Stress* and *M1*. Another interesting observation is that for target dimension 40, we achieve 81.87% reduction in *Stress* as compared to PCA (marked in red).

11.2 Very High Dimensionality Datasets

We chose two very high dimensionality datasets- APTOS 2019 (509k) and DIV2k (6.6M)- one with a low stable rank and one with a high stable rank. We only evaluated these datasets on PCA and RMap other than *DiffRed* because their high dimensionality made other algorithms very slow.

11.2.1 APTOS 2019

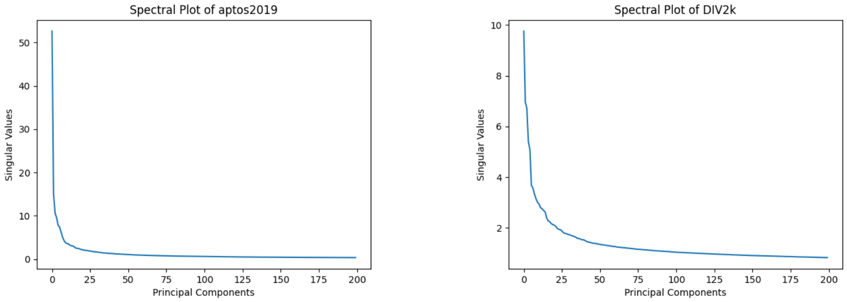
APTOS 2019 [Karthik, 2019] is a Kaggle dataset of 13,000 retina images taken using fundus photography. It is a multiclass-classification dataset where each image is labelled as belonging to one of the five levels of severity of diabetic retinopathy. For our purpose, we resized each image to size of 474 by 358 yielding vectors of dimensionality 509,076 (as each image has 3 channels). This is one of our datasets in the 'very high dimensionality' category. It has a low stable rank of 1.32. Figures 14a and 15a show the spectral and stable rank plots for APTOS 2019 and Tables 12 and 20 show the results of the grid search experiments on k_1 and k_2 for *Stress* and *M1* metrics.

11.2.2 DIV2k

DIV2k [Agustsson and Timofte, 2017a, Agustsson and Timofte, 2017b] is a dataset of 800 2K high resolution image dataset from the NTIRE 2017 challenge. For our purposes, we rescale every image to 1080 by 2048 which

[5] https://mespadoto.github.io/proj-quant-eval/

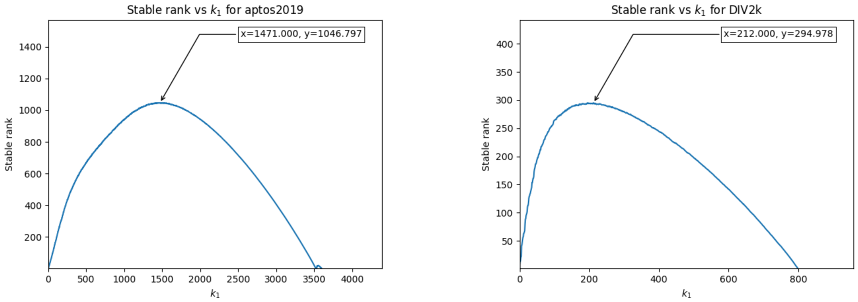
[6] https://huggingface.co/datasets/reuters21578



(a) APTOS 2019

(b) DIV2k

Figure 14: Spectral plots of very high dimensionality datasets.



(a) APTOS 2019

(b) DIV2k

Figure 15: Stable Rank plots of very high dimensionality datasets.

means that each image can be represented as a 6,635,520 dimensional vector (3 channels of color). The dataset has a high stable rank of 8.39. Figures 14b and 15b are the respective spectral and stable rank plots. Tables 13 and 21 show the results of our grid search experiments to find optimal k_1 and k_2 for *Stress* and *M1* metrics.

11.3 Low sensitivity of Λ_{M_1} to k_1 and k_2

The following table shows the Average Variance of Λ_{M_1} over different k_1 and k_2 values for different dimensions (described in Section 5.3.1, **Observation 2**).

Dataset	β
Bank	3.85e-06
Hatespeech	3.45e-07
FMnist	9.85e-07
Cifar10	3.25e-07
geneRNASeq	3.59e-06
Reuters30k	2.17e-08
APTOS 2019	3.07e-06
DIV2k	9.80e-08

Table 4: Variance β (defined in Sec. 5.3.1, **Observation 2**) observed in Λ_{M_1} for different combinations of k_1 and k_2 averaged over all target dimensions.

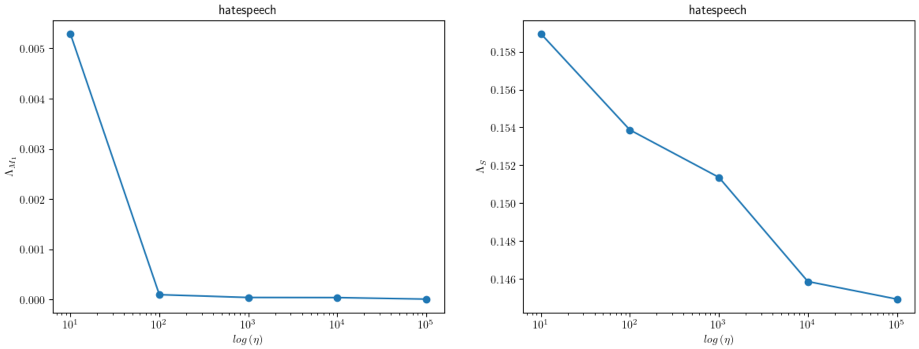
11.4 Hyperparameter tuning

For our experiments on hyperparameter tuning for other dimensionality reduction techniques, we have presented the most optimal values of *Stress* and *M1* in our tables. For full results, please refer to the files in the following directories in our repository[7]:

- Full results: `Experiments/dimensionality_reduction_metrics/results/other_dr_techniques/`
- Code: `Experiments/dimensionality_reduction_metrics/other_dr_techniques/`

11.5 Effect of Monte Carlo iterations [From Section 5.4]

Figure 16 shows that performing Monte Carlo iterations helps in finding good random directions. With more Monte Carlo iterations, we find Random Maps that further reduce the *M1* metric. We note that such Random Maps (which minimize *M1*), also further reduce *Stress*. All the results presented in the paper have been computed at $\eta = 100$.



(a) M1

(b) Stress

Figure 16: Log scale plot of Stress and M1 metrics against the number of Monte Carlo iterations η showing how diminishing improvements over the metrics are obtained with increasing η

11.6 Application: *DiffRed* as a precursor to visualization

PCA has been widely used to reduce the dimensionality of high-dimensional datasets before applying T-SNE/UMap to mitigate the slow computation for high dimensions. Using *DiffRed*, we can reduce the data to an intermediate dimension while preserving *Stress* (global structure) and then apply T-SNE/UMap for visualization. The following Table 5 shows that using *DiffRed* as a preprocessing step causes significant improvement in the *Stress* of the final T-SNE/UMap visualization for the Reuters30k dataset.

Method	Λ_S
PCA + T-SNE	0.55
DiffRed + T-SNE	**0.32**
PCA + UMap	0.56
DiffRed + UMap	**0.45**

Table 5: *Stress* of T-SNE and UMap after using PCA & *DiffRed* as pre-processing step for Reuters30k with intermediate dimension 10. Final *Stress* for the T-SNE2 and UMap2 versions described in the main paper are presented here.

12 Tables

In this section, we provide data from various hyper-parameter tuning experiments. For *DiffRed*, we varied the target dimension and the values of k_1 and k_2. It is clear from these experiments, that by increasing the target dimension, we can reduce the *Stress* metric. The *M1* metric is not sensitive to the choice of k_1 and k_2. However, the values of k_1 and k_2 have to be chosen carefully to minimize *Stress*. The optimal choice can be made by using the theoretical bound as discussed in the main text of the paper in Section 5.3.2.

[7] https://github.com/S3-Lab-IIT/DiffRed

12.1 *Stress*

Table 6: Bank:Λ_S

Target Dimension	k1	k2	Stress	PCA Stress	RMap Stress (α=20)		S-PCA Stress	K-PCA Stress	UMAP Stress
					μ	σ			
1	0	1	0.278474	0.175070	0.40	0.128	0.189710	0.507601	12.799170
2	0	2	0.413185	0.085241	0.28	0.095	0.099255	0.481864	8.930940
2	**1**	**1**	**0.310159**						
3	0	3	0.250078	0.038763	0.28	0.060	0.057670	0.470851	7.912646
3	1	2	0.169713						
3	**2**	**1**	**0.056258**						
5	0	5	0.117427	0.003634	0.22	0.098	0.036128	0.465844	7.402985
5	1	4	0.098115						
5	2	3	0.050027						
5	3	2	0.010721						
5	**4**	**1**	**0.004978**						
6	0	6	0.082337	0.003634	0.17	0.059	0.037417	0.465480	7.069356
6	1	5	0.073995						
6	2	4	0.02745						
6	3	3	0.012201						
6	4	2	0.004184						
6	**5**	**1**	**0.00235**						
7	0	7	0.1341	0.001219	0.17	0.091	0.036316	0.465270	7.107705
7	1	6	0.053662						
7	2	5	0.011971						
7	3	4	0.00538						
7	4	3	0.002543						
7	5	2	0.00159						
7	**6**	**1**	**0.00109**						
8	0	8	0.109656	0.000674	0.18	0.070	0.037008	0.465107	7.131875
8	2	6	0.032214						
8	3	5	0.007498						
8	4	4	0.002741						
8	5	3	0.002032						
8	6	2	0.00073						
8	**7**	**1**	**0.000424**						

Table 7: Hatespeech: Λ_S

Target Dimension	k1	k2	Stress	PCA Stress	RMap Stress (α=20)		S-PCA Stress	K-PCA Stress	UMAP Stress
					μ	σ			
10	0	10	0.154463	0.36	0.16	0.01	0.36	0.65	5.29
10	**1**	**9**	**0.152249**						
10	2	8	0.161939						
10	3	7	0.159242						
10	4	6	0.167399						
10	5	5	0.167598						
10	6	4	0.182932						
10	7	3	0.207632						
20	0	20	0.108961	0.26	0.11	0.00	0.27	0.64	5.21
20	2	18	0.107507						
20	3	17	0.098006						
20	4	16	0.098426						
20	**5**	**15**	**0.097116**						
20	8	12	0.099934						
20	10	10	0.107269						
20	12	8	0.115551						
20	15	5	0.127951						
20	18	2	0.181814						
30	0	30	0.089438	0.20	0.09	0.00	0.22	0.63	5.17
30	2	28	0.085617						
30	3	27	0.083632						
30	5	25	0.079905						
30	6	24	0.082452						
30	8	22	0.076772						
30	10	20	0.07936						
30	**12**	**18**	**0.073679**						
30	15	15	0.076703						
30	18	12	0.086769						
30	20	10	0.090678						
30	25	5	0.103606						
30	27	3	0.127849						
40	0	40	0.078129	0.16	0.08	0.00	0.17	0.63	5.09
40	2	38	0.071711						
40	4	36	0.07244						
40	5	35	0.069092						
40	8	32	0.064041						
40	10	30	0.064503						
40	11	29	0.066805						
40	15	25	0.059473						
40	16	24	0.058154						
40	**20**	**20**	**0.058028**						
40	25	15	0.062881						
40	30	10	0.071768						
40	35	5	0.086825						

Table 8: FMnist: Λ_S

Target Dimension	**k1**	**k2**	**Stress**	**PCA Stress**	**RMap Stress (α=20)**		**S-PCA Stress**	**K-PCA Stress**	**UMAP Stress**
					μ	σ			
10	0	10	0.149077	0.19	0.15	0.009	0.21	0.68	4.02
10	2	8	0.12508						
10	3	7	0.117871						
10	4	6	0.120748						
10	**5**	**5**	**0.117036**						
10	6	4	0.124043						
10	7	3	0.134036						
20	0	20	0.111262	0.14	0.11	0.009	0.16	0.68	4.34
20	2	18	0.085441						
20	4	16	0.073317						
20	5	15	0.068579						
20	6	14	0.069169						
20	**8**	**12**	**0.066389**						
20	10	10	0.067899						
20	12	8	0.070051						
20	15	5	0.080917						
20	18	2	0.112871						
30	0	30	0.095465	0.12	0.09	0.008	0.14	0.68	4.98
30	3	27	0.06347						
30	5	25	0.053878						
30	8	22	0.048813						
30	10	20	0.047958						
30	12	18	0.048057						
30	**15**	**15**	**0.047662**						
30	18	12	0.050517						
30	20	10	0.052338						
30	25	5	0.067108						
30	27	3	0.083181						
40	0	40	0.085391	0.10	0.08	0.006	0.13	0.68	4.18
40	4	36	0.048749						
40	5	35	0.045706						
40	8	32	0.040424						
40	10	30	0.039411						
40	**15**	**25**	**0.037091**						
40	16	24	0.037105						
40	20	20	0.037701						
40	25	15	0.039636						
40	30	10	0.045351						
40	35	5	0.058315						

Table 9: Cifar10: Λ_S

Target Dimension	**k1**	**k2**	**Stress**	**PCA Stress**	**RMap Stress (α=20)**		**S-PCA Stress**	**K-PCA Stress**	**UMAP Stress**
					μ	σ			
10	0	10	0.150986	0.21	0.16	0.009	0.24	0.69	1.26
10	2	8	0.130287						
10	**3**	**7**	**0.127005**						
10	4	6	0.131711						
10	5	5	0.134101						
10	6	4	0.13699						
10	7	3	0.149397						
20	0	20	0.10698	0.15	0.11	0.005	0.18	0.69	1.25
20	2	18	0.088584						
20	4	16	0.080418						
20	5	15	0.079099						
20	**8**	**12**	**0.076988**						
20	10	10	0.07744						
20	12	8	0.080066						
20	15	5	0.091193						
20	18	2	0.125555						
30	0	30	0.087177	0.12	0.09	0.004	0.15	0.69	1.27
30	3	27	0.067666						
30	5	25	0.060986						
30	8	22	0.056502						
30	12	18	0.054123						
30	**15**	**15**	**0.053645**						
30	18	12	0.055647						
30	20	10	0.057995						
30	25	5	0.073301						
30	27	3	0.088451						
40	0	40	0.072628	0.11	0.08	0.003	0.13	0.69	1.27
40	4	36	0.053967						
40	5	35	0.052083						
40	8	32	0.046417						
40	10	30	0.045487						
40	15	25	0.042958						
40	16	24	0.042068						
40	**20**	**20**	**0.041934**						
40	25	15	0.043438						
40	30	10	0.047964						
40	35	5	0.062596						

Table 10: geneRNASeq: Λ_S

Target Dimension	k1	k2	Stress	PCA Stress	RMap Stress (α=20)		S-PCA Stress	K-PCA Stress	UMAP Stress
					μ	σ			
10	0	10	0.154996	0.21	0.16	0.008	0.25	NA	18.72
10	2	8	0.156133						
10	3	7	0.138719						
10	4	6	0.132871						
10	**5**	**5**	**0.130126**						
10	6	4	0.133581						
10	7	3	0.147557						
20	0	20	0.103773	0.17	0.11	0.006	0.24	0.70	18.60
20	2	18	0.091928						
20	4	16	0.082728						
20	5	15	0.080249						
20	6	14	0.077367						
20	**8**	**12**	**0.071007**						
20	10	10	0.075024						
20	12	8	0.078858						
20	15	5	0.094134						
20	18	2	0.134547						
30	0	30	0.092378	0.15	0.09	0.009	0.25	0.70	18.72
30	3	27	0.070738						
30	5	25	0.059465						
30	8	22	0.054121						
30	**10**	**20**	**0.052897**						
30	12	18	0.055643						
30	15	15	0.057122						
30	18	12	0.059765						
30	20	10	0.062337						
30	25	5	0.083422						
30	27	3	0.101695						
40	0	40	0.090848	0.14	0.08	0.004	0.25	0.70	18.22
40	4	36	0.055797						
40	5	35	0.050136						
40	8	32	0.045299						
40	**10**	**30**	**0.043141**						
40	15	25	0.043549						
40	16	24	0.044106						
40	20	20	0.045514						
40	25	15	0.049506						
40	30	10	0.05636						
40	35	5	0.077122						

Table 11: Reuters30k: Λ_S

Target Dimension	k1	k2	Stress	PCA Stress	RMap Stress (α=20)		S-PCA Stress	K-PCA Stress	UMAP Stress
					μ	σ			
10	**0**	**10**	**0.155841**	0.49	0.16	0.001	0.49	0.71	3.35
10	2	8	0.162356						
10	3	7	0.170477						
10	4	6	0.183243						
10	5	5	0.197407						
10	6	4	0.216498						
10	7	3	0.244457						
20	0	20	0.110339	0.45	0.11	0.001	0.46	0.70	3.27
20	**2**	**18**	**0.109416**						
20	4	16	0.114293						
20	5	15	0.11665						
20	8	12	0.127054						
20	10	10	0.136838						
20	12	8	0.150428						
20	15	5	0.184754						
20	18	2	0.27384						
30	0	30	0.090478	0.43	0.09	0.001	0.44	0.70	3.21
30	**2**	**28**	**0.088198**						
30	3	27	0.088585						
30	5	25	0.089782						
30	8	22	0.094184						
30	10	20	0.097164						
30	12	18	0.101738						
30	15	15	0.109841						
30	18	12	0.120367						
30	20	10	0.130392						
30	25	5	0.179204						
30	27	3	0.225727						
40	0	40	0.079027	0.41	0.08	0.001	0.43	0.70	3.12
40	2	38	0.075794						
40	**4**	**36**	**0.075186**						
40	5	35	0.076406						
40	8	32	0.077578						
40	10	30	0.079557						
40	15	25	0.085225						
40	16	24	0.086591						
40	20	20	0.093946						
40	25	15	0.10561						
40	30	10	0.127771						
40	35	5	0.175076						

Table 12: APTOS 2019: Λ_S

Target Dimension	**k1**	**k2**	**Stress**	**PCA Stress**	**RMap Stress (α=20)**	
					μ	σ
10	0	10	0.179073	0.12	0.16	0.016
10	1	9	0.148839			
10	2	8	0.123974			
10	3	7	0.122061			
10	4	6	0.1052			
10	5	5	0.097674			
10	6	4	0.101736			
10	**7**	**3**	**0.097584**			
20	0	20	0.095128	0.08	0.11	0.014
20	2	18	0.085237			
20	3	17	0.075839			
20	4	16	0.06693			
20	5	15	0.058859			
20	**8**	**12**	**0.045718**			
20	10	10	0.047303			
20	12	8	0.046251			
20	15	5	0.05158			
20	18	2	0.06905			
30	0	30	0.091059	0.06	0.09	0.008
30	2	28	0.060186			
30	3	27	0.060056			
30	5	25	0.046503			
30	8	22	0.034993			
30	10	20	0.032974			
30	11	19	0.031663			
30	12	18	0.030824			
30	**15**	**15**	**0.030116**			
30	18	12	0.030801			
30	20	10	0.030715			
30	25	5	0.038002			
30	27	3	0.045107			
40	0	40	0.086019	0.05	0.08	0.012
40	2	38	0.055444			
40	4	36	0.042809			
40	5	35	0.038921			
40	8	32	0.029978			
40	10	30	0.026612			
40	15	25	0.023963			
40	16	24	0.024174			
40	**20**	**20**	**0.022142**			
40	25	15	0.022693			
40	30	10	0.025446			
40	35	5	0.031453			

Table 13: DIV2k: Λ_S

Target Dimension	**k1**	**k2**	**Stress**	**PCA Stress**	**RMap Stress (α=20)**	
					μ	σ
10	0	10	0.156703	0.31	0.16	0.003
10	**1**	**9**	**0.144361**			
10	2	8	0.144838			
10	3	7	0.146517			
10	4	6	0.149547			
10	5	5	0.157161			
10	6	4	0.174424			
10	7	3	0.185904			
20	0	20	0.109547	0.26	0.11	0.003
20	2	18	0.099879			
20	3	17	0.09588			
20	4	16	0.093435			
20	**5**	**15**	**0.091908**			
20	8	12	0.096338			
20	10	10	0.102482			
20	12	8	0.109068			
20	15	5	0.132754			
20	18	2	0.190162			
30	0	30	0.09143	0.23	0.09	0.002
30	2	28	0.081431			
30	3	27	0.076753			
30	5	25	0.072565			
30	8	22	0.073153			
30	**10**	**20**	**0.072457**			
30	12	18	0.073059			
30	15	15	0.078949			
30	18	12	0.083008			
30	20	10	0.090443			
30	25	5	0.121266			
30	27	3	0.147061			
40	0	40	0.080119	0.21	0.08	0.002
40	2	38	0.069251			
40	4	36	0.064546			
40	5	35	0.061345			
40	8	32	0.061543			
40	**10**	**30**	**0.05954**			
40	15	25	0.062452			
40	16	24	0.061158			
40	20	20	0.064559			
40	25	15	0.06917			
40	30	10	0.083609			
40	35	5	0.109629			

12.2 *M1*

Table 14: Bank: Λ_{M_1}

Target Dimension	k1	k2	M1	PCA M1	RMap M1 (α=20)		S-PCA M1	K-PCA M1	UMAP M1
					μ	σ			
1	0	1	0.006966	0.651017	0.61	0.425	0.683462	0.960012	171.177790
2	0	2	0.012508	0.584043	0.42	0.229	0.614852	0.951856	95.028671
2	**1**	**1**	**0.012201**						
3	0	3	0.001685	0.550702	0.43	0.265	0.577230	0.947818	94.888941
3	1	2	0.003476						
3	**2**	**1**	**0.000357**						
5	0	5	0.002059	0.535537	0.38	0.446	0.564456	0.945817	122.448199
5	1	4	2.51e-5						
5	2	3	2.82e-5						
5	3	2	0.000148						
5	**4**	**1**	**5.82e-6**						
6	0	6	0.002623	0.535173	0.23	0.204	0.576306	0.945657	182.743756
6	1	5	0.004556						
6	2	4	3.60e-5						
6	3	3	0.000192						
6	4	2	2.92e-5						
6	**5**	**1**	**4.57e-7**						
7	0	7	0.011784	0.534967	0.26	0.234	0.575984	0.945562	190.630670
7	1	6	0.003607						
7	2	5	1.94e-5						
7	3	4	0.000287						
7	4	3	3.31e-6						
7	**5**	**2**	**2.59e-6**						
7	6	1	9.22e-6						
8	0	8	0.001774	0.534825	0.23	0.202	0.575932	0.945489	256.766924
8	2	6	0.000117						
8	3	5	2.54e-5						
8	4	4	1.01e-5						
8	5	3	7.06e-6						
8	6	2	3.57e-6						
8	**7**	**1**	**3.49e-7**						

Table 15: Hatespeech: Λ_{M_1}

Target Dimension	k1	k2	M1	PCA M1	RMap M1 (α=20)		S-PCA M1	K-PCA M1	UMAP M1
					μ	σ			
10	0	10	0.001827	0.66	0.06	0.04	0.68	0.99	240.50
10	1	9	0.001306						
10	2	8	4.47e-4						
10	3	7	0.001135						
10	4	6	0.00197						
10	5	5	0.002305						
10	**6**	**4**	**0.000191**						
10	7	3	0.000364						
20	0	20	1.64e-3	0.50	0.04	0.03	0.53	0.99	569.90
20	2	18	0.000502						
20	3	17	1.36e-3						
20	**4**	**16**	**9.59e-6**						
20	5	15	0.000354						
20	8	12	6.57e-4						
20	10	10	0.000718						
20	12	8	0.000512						
20	15	5	0.000965						
20	18	2	0.000668						
30	0	30	0.00033	0.41	0.03	0.02	0.44	0.99	828.16
30	**2**	**28**	**1.76e-6**						
30	3	27	0.000218						
30	5	25	2.21e-3						
30	6	24	3.64e-4						
30	8	22	6.39e-4						
30	10	20	1.82e-4						
30	12	18	4.03e-4						
30	15	15	5.34e-5						
30	18	12	1.54e-4						
30	20	10	3.40e-6						
30	25	5	7.39e-5						
30	27	3	3.76e-4						
40	0	40	1.26e-3	0.33	0.03	0.02	0.36	0.99	1095.06
40	2	38	0.00123						
40	4	36	3.35e-4						
40	5	35	2.42e-4						
40	8	32	7.41e-5						
40	10	30	3.36e-5						
40	11	29	2.31e-4						
40	15	25	2.14e-4						
40	16	24	0.000412						
40	**20**	**20**	**5.38e-6**						
40	25	15	0.000328						
40	30	10	0.000132						
40	35	5	8.17e-5						

Table 16: FMnist: Λ_{M_1}

Target Dimension	**k1**	**k2**	**M1**	**PCA M1**	**RMap M1 (α=20)**		**S-PCA M1**	**K-PCA M1**	**UMAP M1**
					μ	σ			
10	0	10	0.005264	0.60	0.11	0.059	0.64	1.00	241.35
10	2	8	0.000153						
10	**3**	**7**	**3.74e-5**						
10	4	6	0.000303						
10	5	5	0.000181						
10	6	4	0.000192						
10	7	3	0.000181						
20	0	20	0.001009	0.55	0.11	0.077	0.59	1.00	487.89
20	2	18	4.94e-5						
20	4	16	0.000164						
20	5	15	3.21e-5						
20	**6**	**14**	**1.79e-5**						
20	8	12	0.000104						
20	10	10	3.71e-5						
20	12	8	0.000122						
20	15	5	0.000289						
20	18	2	0.000163						
30	0	30	0.001011	0.52	0.09	0.054	0.56	1.00	781.17
30	3	27	0.00012						
30	5	25	4.63e-5						
30	8	22	0.000116						
30	10	20	2.21e-5						
30	12	18	5.55e-5						
30	15	15	2.46e-5						
30	18	12	4.09e-5						
30	20	10	9.53e-6						
30	**25**	**5**	**2.73e-7**						
30	27	3	9.15e-6						
40	0	40	0.000826	0.49	0.09	0.061	0.54	1.00	1030.35
40	4	36	0.000212						
40	5	35	2.01e-5						
40	8	32	5.20e-5						
40	10	30	0.000129						
40	**15**	**25**	**1.10e-6**						
40	16	24	1.68e-6						
40	20	20	2.02e-5						
40	25	15	2.39e-5						
40	30	10	3.05e-6						
40	35	5	2.16e-5						

Table 17: Cifar10: Λ_{M_1}

Target Dimension	**k1**	**k2**	**M1**	**PCA M1**	**RMap M1 (α=20)**		**S-PCA M1**	**K-PCA M1**	**UMAP M1**
					μ	σ			
10	0	10	0.002759	0.49	0.09	0.062	0.54	1.00	166.84
10	2	8	0.000142						
10	3	7	0.000154						
10	4	6	0.000156						
10	5	5	0.001149						
10	**6**	**4**	**0.000131**						
10	7	3	0.000799	0.38	0.04	0.035	0.43	1.00	485.22
20	0	20	0.000342						
20	2	18	0.000517						
20	4	16	0.000547						
20	5	15	0.001161						
20	**8**	**12**	**0.00011**						
20	10	10	0.000162						
20	12	8	0.000167						
20	15	5	0.000416						
20	18	2	0.000236						
30	0	30	0.001148	0.32	0.05	0.031	0.38	1.00	753.84
30	3	27	0.000331						
30	5	25	0.000243						
30	8	22	0.000126						
30	12	18	0.000454						
30	**15**	**15**	**5.66e-5**						
30	18	12	0.00086						
30	20	10	6.23e-5						
30	25	5	9.57e-5						
30	27	3	0.000344						
40	0	40	0.000331	0.28	0.04	0.028	0.34	1.00	1008.78
40	4	36	0.001214						
40	5	35	0.000229						
40	8	32	0.000165						
40	10	30	2.01e-5						
40	15	25	2.84e-5						
40	16	24	2.03e-4						
40	20	20	4.94e-5						
40	25	15	0.000398						
40	30	10	1.12e-4						
40	**35**	**5**	**1.35e-5**						

Table 18: geneRNASeq: Λ_{M_1}

Target Dimension	k1	k2	M1	PCA M1	RMap M1 (α=20)		S-PCA M1	K-PCA M1	UMAP M1
					μ	σ			
10	0	10	0.009101	0.94	0.31	0.246	0.95	1.00	328.72
10	2	8	0.000113						
10	3	7	0.000175						
10	4	6	3.68e-5						
10	**5**	**5**	**1.69e-5**						
10	6	4	7.96e-5						
10	7	3	6.20e-5						
20	0	20	0.004859	0.93	0.18	0.149	0.95	1.00	586.64
20	2	18	8.93e-5						
20	4	16	0.000114						
20	**5**	**15**	**2.59e-6**						
20	6	14	8.39e-6						
20	8	12	2.39e-5						
20	10	10	1.58e-5						
20	12	8	4.68e-5						
20	15	5	8.67e-6						
20	18	2	0.000121						
30	0	30	0.001417	0.93	0.19	0.143	0.95	1.00	826.54
30	**3**	**27**	**1.25e-6**						
30	5	25	2.02e-5						
30	8	22	1.51e-5						
30	10	20	6.49e-6						
30	12	18	1.53e-6						
30	15	15	5.31e-6						
30	18	12	2.33e-5						
30	20	10	6.10e-6						
30	25	5	2.02e-6						
30	27	3	8.15e-6						
40	0	40	0.001658	0.93	0.12	0.081	0.95	1.00	1089.52
40	4	36	6.77e-6						
40	5	35	1.25e-5						
40	8	32	1.02e-5						
40	10	30	1.38e-5						
40	**15**	**25**	**4.61e-6**						
40	16	24	1.61e-5						
40	20	20	1.07e-5						
40	25	15	2.02e-5						
40	30	10	4.68e-6						
40	35	5	1.48e-5						

Table 19: Reuters30k: Λ_{M_1}

Target Dimension	k1	k2	M1	PCA M1	RMap M1 (α=20)		S-PCA M1	K-PCA M1	UMAP M1
					μ	σ			
10	0	10	0.00062	0.88	0.03	0.018	0.88	1.00	196.97
10	2	8	7.17e-5						
10	3	7	0.000112						
10	4	6	0.000445						
10	**5**	**5**	**9.61e-6**						
10	6	4	0.000127						
10	7	3	6.91e-5						
20	0	20	0.000577	0.85	0.03	0.020	0.85	1.00	394.25
20	2	18	2.30e-5						
20	4	16	6.20e-5						
20	5	15	0.000112						
20	8	12	2.88e-5						
20	10	10	8.41e-5						
20	**12**	**8**	**2.10e-5**						
20	15	5	0.000129						
20	18	2	0.000133						
30	0	30	1.83e-5	0.83	0.02	0.017	0.84	1.00	679.72
30	2	28	NA						
30	**3**	**27**	**1.50e-6**						
30	5	25	1.20e-4						
30	8	22	3.25e-5						
30	10	20	1.99e-5						
30	12	18	2.53e-5						
30	15	15	7.05e-5						
30	18	12	1.60e-5						
30	20	10	7.19e-5						
30	25	5	7.04e-5						
30	27	3	2.16e-5						
40	0	40	4.49e-5	0.81	0.02	0.014	0.83	1.00	913.86
40	2	38	NA						
40	4	36	1.72e-4						
40	5	35	8.02e-5						
40	8	32	4.53e-6						
40	10	30	5.61e-5						
40	**15**	**25**	**5.17e-7**						
40	16	24	1.04e-5						
40	20	20	4.85e-5						
40	25	15	9.50e-6						
40	30	10	4.60e-5						
40	35	5	7.85e-5						

Table 20: APTOS 2019: Λ_{M_1}

Target Dimension	**k1**	**k2**	**M1**	**PCA M1**	**RMap M1 (α=20)**	
					μ	σ
10	0	10	0.006698	0.81	0.24	0.15
10	1	9	0.000345			
10	2	8	0.000173			
10	3	7	2.47e-5			
10	4	6	2.98e-5			
10	5	5	3.82e-5			
10	6	4	4.09e-5			
10	**7**	**3**	**1.58e-5**			
20	0	20	1.95e-3	0.79	0.15	0.12
20	2	18	8.24e-5			
20	3	17	1.27e-5			
20	4	16	5.36e-5			
20	5	15	2.09e-5			
20	8	12	2.58e-5			
20	**10**	**10**	**2.76e-6**			
20	12	8	4.15e-5			
20	15	5	7.27e-5			
20	18	2	8.11e-5			
30	0	30	0.008812	0.78	0.15	0.08
30	2	28	6.57e-5			
30	3	27	9.98e-5			
30	5	25	9.81e-5			
30	8	22	3.99e-5			
30	10	20	4.82e-5			
30	11	19	1.57e-5			
30	12	18	9.66e-5			
30	15	15	1.87e-5			
30	18	12	1.83e-5			
30	20	10	4.97e-5			
30	25	5	9.97e-6			
30	**27**	**3**	**8.88e-6**	0.77	0.13	0.14
40	0	40	0.002634			
40	2	38	2.69e-4			
40	4	36	6.84e-6			
40	5	35	1.38e-4			
40	8	32	7.96e-5			
40	10	30	3.79e-5			
40	15	25	2.26e-5			
40	16	24	9.11e-6			
40	20	20	6.41e-6			
40	**25**	**15**	**2.25e-6**			
40	30	10	7.04e-6			
40	35	5	2.02e-5			

Table 21: DIV2k: Λ_{M_1}

Target Dimension	**k1**	**k2**	**M1**	**PCA M1**	**RMap M1 (α=20)**	
					μ	σ
10	0	10	0.001538	0.66	0.05	0.029
10	1	9	0.00092			
10	2	8	0.000219			
10	3	7	0.000219			
10	4	6	0.0007			
10	5	5	0.000289			
10	**6**	**4**	**7.07e-5**			
10	7	3	0.000231			
20	**0**	**20**	**3.01e-6**			
20	2	18	7.61e-5	0.58	0.04	0.027
20	3	17	9.13e-5			
20	4	16	3.25e-5			
20	5	15	9.80e-5			
20	8	12	0.000158			
20	10	10	0.000147			
20	12	8	9.74e-5			
20	15	5	4.49e-5			
20	18	2	0.00063			
30	0	30	0.000279	0.54	0.02	0.020
30	2	28	0.000965			
30	3	27	8.80e-5			
30	5	25	0.000117			
30	8	22	0.00059			
30	10	20	0.000402			
30	12	18	0.000163			
30	15	15	0.000301			
30	**18**	**12**	**7.46e-5**			
30	20	10	0.000432			
30	25	5	0.000187			
30	27	3	0.000164			
40	0	40	0.000696	0.51	0.03	0.018
40	2	38	0.000442			
40	4	36	5.37e-5			
40	5	35	9.36e-5			
40	8	32	6.96e-5			
40	10	30	0.00034			
40	15	25	8.92e-5			
40	16	24	6.33e-5			
40	20	20	8.28e-6			
40	25	15	0.000162			
40	**30**	**10**	**4.00e-6**			
40	35	5	0.000191			