No. 134/25, 56–70
ISSN 2657-6988 (online)
ISSN 2657-5841 (printed)
DOI: 10.26408/134.04

Submitted: 27.01.2025
Accepted: 14.04.2025
Published: 18.06.2025

SINGULAR VALUE VERSUS EIGENVALUE DECOMPOSITION EFFICIENCY IN COMPUTING PRINCIPAL COMPONENTS FOR DIMENSIONALITY REDUCTION OF LARGE DATASETS

Vadim Romanuke

Naval Academy of the Heroes of Westerplatte, 69 Śmidowicza St., 81-127 Gdynia, Poland, ORCID 0000-0003-3543-3087, e-mail: v.romanuke@amw.gdynia.pl

Abstract: Principal component analysis, being one of the best techniques for dimensionality reduction, is implemented by using one of the two high-accuracy algorithms: the singular value decomposition (SVD) and eigenvalue decomposition (EVD). The EVD is generally faster than the SVD, except for datasets with fewer observations or when the observation has fewer features. Apart from cases of shallower datasets consisting of just a few hundred double-precision observations, the EVD speeds up computing principal components by at least 4.5%, whereas the average speedup in 45% widely varies from 12% to 92%. The speedup on non-shallower single-precision datasets is roughly similar, but it nonetheless makes no sense due to EVD poor accuracy while operating on numeric data with single precision. The EVD is efficient if the dataset consists of no fewer than a few hundred observations (objects) having at least three double-precision features.

Keywords: Big Data, dimensionality reduction, principal component analysis, singular value decomposition, eigenvalue decomposition, performance, efficiency.

1. INTRODUCTION

In the present era of digitalization, Big Data has become not just a category of massive datasets, but also a scientific field of acquiring, storing, and analyzing such datasets for purposes of civilization evolution. Big Data has been extensively updating and revolutionizing logistics and transport, smart cities, media and entertainment, marketing and advertising, cybersecurity, healthcare, ecology, climate and earth science, industry, and education [Akter and Wamba 2023; Jamarani et al. 2024]. Preprocessing data includes filtering out messy and noisy data, transformation, and feature extraction, where possible simplification of a large dataset is often a primary task [Demirbaga et al. 2024]. Dimensionality reduction is the main paradigm to reasonably simplify large datasets without losing consistent information [Krishnan, Samaranayake and Jagannathan 2019; Oliveira and Cordeiro 2020]. Dimensionality reduction also allows preliminarily visualizing a high-

dimensional dataset, sometimes without further usage, just to see meaningful spots, trends, or clusters in the dataset in order to use them in further analysis of a simplified dataset having more than three features [Demirbaga et al. 2024; Olszewski 2025].

One of the best and most applicable techniques for dimensionality reduction is principal component analysis (PCA). The PCA linearly transforms the dataset onto a new coordinate system whose directions acquire the largest variation in the data [Olivieri 2024; Si-ahmed et al. 2025]. These directions sorted in descending order correspond to principal components. The principal components constitute an orthonormal basis in which the different individual dimensions of the data are linearly uncorrelated [Härdle, Simar and Fengler 2024]. Therefore, the PCA is defined as an orthogonal linear transformation on a real inner product space that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (being the first principal component), the second greatest variance lies on the second coordinate, and so on.

If the dataset is an $M \times N$ matrix

$$\mathbf{X} = \left[x_{mn} \right]_{M \times N} \tag{1}$$

representing M observations of N-dimensional objects (i. e., objects with N features), where usually $M > N$, the PCA returns N principal components

$$\left\{ \mathbf{P}_n = \left[p_{kn} \right]_{N \times 1} \right\}_{n=1}^{N} \tag{2}$$

that are unit vectors [Olivieri 2024] in the columns of matrix

$$\mathbf{P} = \left[p_{kn} \right]_{N \times N},$$

i. e.

$$\sqrt{\sum_{k=1}^{N} p_{kn}^2} = 1 \quad \forall n = \overline{1, N}. \tag{3}$$

The n-th vector, $n = \overline{2, N}$, is the direction of a line that best fits the data while being orthogonal to the first $n-1$ vectors [Jolliffe and Cadima 2016]. A best-fitting line minimizes the average squared perpendicular distance from the points to the line [Härdle, Simar and Fengler 2024]. The first principal component of a dataset represented by matrix (1) is constructed as a linear combination of the original N variables, and it explains the most variance of the data. The first principal component corresponds to a direction that maximizes the variance of the projected data. The second principal component explains the most variance in what is left once

the effect of the first principal component is removed. Thus every next principal component explains less variance. Together the N principal components explain all the variance. The n-th principal component, $n=\overline{2,N}$, can be taken as a direction orthogonal to the first $n-1$ principal components that maximizes the variance of the projected data.

With principal components (2), the dataset (1) is transformed into a dataset being an $M\times N$ matrix

$$\mathbf{Y}=\left[y_{mn}\right]_{M\times N}=\mathbf{X}\cdot\mathbf{P} \tag{4}$$

whose entries are:

$$y_{mn}=\sum_{k=1}^{N}x_{mk}\,p_{kn}\ \text{ for }\ m=\overline{1,M}\ \text{ and }\ n=\overline{1,N}\,. \tag{5}$$

In fact, columns of matrix (4), from left to right, are sorted in descending order by variation in the original data. If the first N^* principal components

$$\left\{\mathbf{P}_n=\left[p_{kn}\right]_{N\times N}\right\}_{n=1}^{N^*}\ \text{ by }\ N^*\in\left\{\overline{1,N-1}\right\} \tag{6}$$

explain a sufficiently high amount of variance (practically, around 90% or above), the remaining $N-N^*$ principal components

$$\left\{\mathbf{P}_n=\left[p_{kn}\right]_{N\times N}\right\}_{n=N^*+1}^{N}\ \text{ by }\ N^*\in\left\{\overline{1,N-1}\right\} \tag{7}$$

are ignored. Thus the initial dataset (1) is simplified to a dataset represented as an $M\times N^*$ matrix

$$\mathbf{Y}^*=\left[y_{mn}\right]_{M\times N^*}\,. \tag{8}$$

Obviously, matrix (8) is obtained from matrix (4) by cutting off its $N-N^*$ columns from the right.

2. PROBLEM STATEMENT

While the PCA is seemingly a very efficient method for dimensionality reduction of large datasets, it can be implemented by using one of the two high-accuracy algorithms — the singular value decomposition (SVD) [Li 2024; Zizler and La Haye 2024] and eigenvalue decomposition (EVD) [Golub and Van Loan 1996; Holmes 2023]. It is believed that the EVD algorithm is faster than the SVD algorithm when

the number of observations exceeds the number of variables, but may be less accurate. But, in fact, it is generally unclear when the EVD algorithm computes principal components faster than the SVD does. Another question is the computational time complexity of each of the SVD and EVD algorithms.

Therefore, the objective is to determine when each of the SVD and EVD is factually efficient for dimensionality reduction of large datasets. The two numeric types to be studied are double and single precision. To achieve the objective, the following tasks are to be fulfilled:

1. To generate random large datasets as matrices of a specified numeric type, where the generation should be done repeatedly to obtain statistically consistent and stable operation speed results upon averaging over a definite number of generations.

2. To measure computational time of the PCA by the SVD algorithm applied to generated matrices, and to measure computational time of the PCA by the EVD algorithm applied to the same generated matrices.

3. To carry out a comparative analysis of the averaged computational times, and to estimate the computational time complexity of each of the SVD and EVD algorithms.

4. To discuss the obtained results and findings from the comparative analysis of SVD versus EVD.

5. To conclude on the factual efficiency of the SVD and EVD in the PCA for the dimensionality reduction of large datasets, whereupon open questions and perspectives of further research are to be outlined.

3. DATASETS

A random dataset modeled as a matrix (1) of a specified numeric type is generated by using the standard normal distribution having zero mean and unit variance [Romanuke 2018]. Hence, each entry x_{mn} in the matrix is a value of the normally distributed random variable with zero mean and unit variance [Olivieri 2024].

As the numeric types to be studied are double and single precision for real numbers, there will be two series of random datasets. For these two types the number of observations M is sequentially set to every element of the 37-elemented set

$$\mathrm{M}=\left\{\left\{10^2\cdot k\right\}_{k=1}^{9},\left\{10^3\cdot k\right\}_{k=1}^{9},\left\{10^4\cdot k\right\}_{k=1}^{9},\left\{10^5\cdot k\right\}_{k=1}^{10}\right\} \tag{9}$$

and the number of initial variables (i. e., original features) N is sequentially set to every element of the 50-elemented set

$$\mathrm{N}=\left\{\left\{j+1\right\}_{j=1}^{9},\left\{2j+10\right\}_{j=1}^{5},\left\{5j+20\right\}_{j=1}^{16},\left\{10j+100\right\}_{j=1}^{10},\left\{20j+200\right\}_{j=1}^{10}\right\}. \tag{10}$$

So, by the specified numeric type, there are 1850 versions of the large dataset subject to dimensionality reduction. To obtain statistically consistent and stable results of operation speed (computational time), the dataset by every couple

$$\{M, N\} \text{ for } M \in \mathrm{M} \text{ and } N \in \mathrm{N} \tag{11}$$

is generated at 100 different pseudorandom number generator seeds. Thus, the grand total is 185,000 large datasets for double and single precision, separately. Computational times are stored as a $37 \times 50 \times 100$ array for each precision. They are averaged over the third dimension (generations), loosely being related to dimensionality reduction either, to achieve an average surface of two variables – the number of observations and the number of features.

4. EFFICIENCY AND COMPUTATIONAL TIME COMPLEXITY

Denote by $t_{\mathrm{SVD}}(M, N, i)$ and $t_{\mathrm{EVD}}(M, N, i)$ the time spans taken to compute the principal components by the SVD and EVD algorithms, respectively, for (11) by (9), (10) at generation i. The respective averaged computational times are

$$\overline{t}_{\mathrm{SVD}}(M, N) = \frac{1}{100} \cdot \sum_{i=1}^{100} t_{\mathrm{SVD}}(M, N, i) \tag{12}$$

and

$$\overline{t}_{\mathrm{EVD}}(M, N) = \frac{1}{100} \cdot \sum_{i=1}^{100} t_{\mathrm{EVD}}(M, N, i). \tag{13}$$

The spans

$$\left\{t_{\mathrm{SVD}}(M, N, i)\right\}_{i=1}^{100} \text{ and } \left\{t_{\mathrm{EVD}}(M, N, i)\right\}_{i=1}^{100} \tag{14}$$

are measured on a dual-core processor: Intel Core i5-7200U@2.50GHz, with 11.8 GB RAM in MATLAB R2018a.

Along with the principal components (2) computed within time spans (14), the PCA also returns [Jolliffe and Cadima 2016; Olivieri 2024] the principal component scores (PCSs), by which the original dataset without its N means

$$\left\{\frac{1}{M} \cdot \sum_{m=1}^{M} x_{mn}\right\}_{n=1}^{N}$$

is reconstructed via the matrix product of PCSs and (2); principal component

variances (PCVs), that is the eigenvalues of the covariance matrix of (1); Hotelling's T-squared statistic values (HT2Vs), which is the sum of squares of the standardized scores for each observation; percentages of the total variance explained by each principal component (TVEPs). The percentages of unsigned relative differences (URDPs) for principal components (2), PCSs, PCVs, HT2Vs, and TVEPs by the SVD and EVD algorithms are calculated to see the impact of the EVD algorithm on the PCA findings with respect to the SVD algorithm. Denote the maxima of those URDPs by

$$\delta_{\mathrm{PC}}(M,N),\ \delta_{\mathrm{PCS}}(M,N),\ \delta_{\mathrm{PCV}}(M,N),\ \delta_{\mathrm{HT2V}}(M,N),\ \delta_{\mathrm{TVEP}}(M,N), \quad (15)$$

respectively.

The double-precision SVD computational time (12) is shown in Figure 1. Except for a computational artifact at $M > 9 \cdot 10^5$ and $N > 300$, surface (12) smoothly increases as the double-precision dataset size (i. e. the number of its entries $M \cdot N$) increases. The double-precision EVD computational time (13) shown in Figure 2 has a similar surface. Datasets of a million observations by 400 double-precision features are handled by the EVD algorithm within 116 seconds, whereas the SVD algorithm takes longer than 200 seconds.

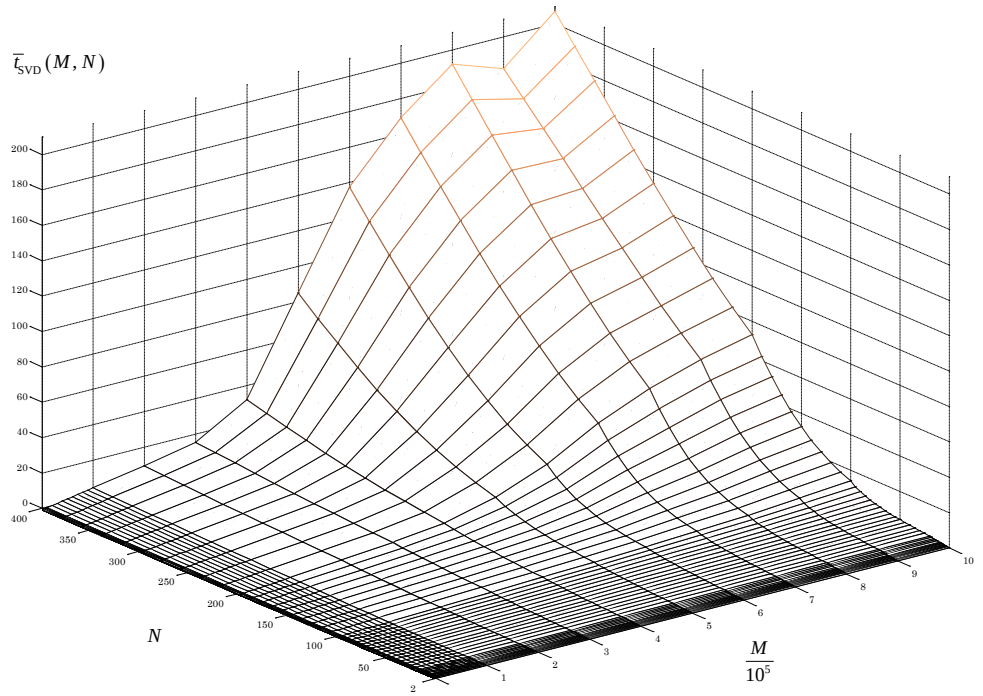

Fig. 1. The average computational time of the SVD algorithm for datasets with double precision

Nevertheless, it is hard to conclude on the EVD algorithm efficiency solely from Figure 2. Henceforward, the relative difference between (12) and (13) is

$$r(M,N)=\frac{\overline{t}_{\mathrm{SVD}}(M,N)}{\overline{t}_{\mathrm{EVD}}(M,N)}, \tag{16}$$

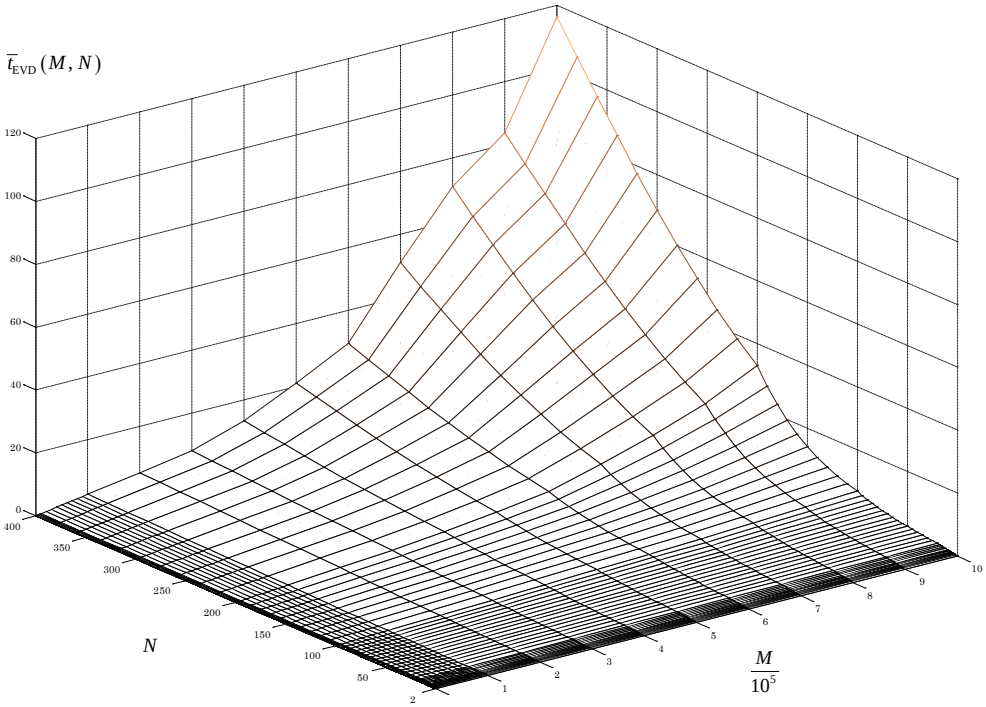

Fig. 2. The average computational time of the EVD algorithm for datasets with double precision

where $r(M,N)>1$ implies that the EVD algorithm is efficient at a given couple $\{M,N\}$. Ratio (16) is shown in Figure 3 for double precision as a surface mesh. It is noticeable that surface (16) falls off its peak at

$$M>6\cdot 10^5 \text{ and } N>200, \tag{17}$$

although ratio (16) is still greater than 1 (17). Another noticeable thing is ratio (16) drops below 1 at

$$M<500 \text{ and } N>80, \tag{18}$$

which is shown in Figure 4. Ratio (16) for double-precision datasets with only two

features approximates 1 from above, dropping to 1.02 at $M = 10^5$. Occasionally,

$$r(9000, 2) = 0.9829,$$

but this is the single point of the EVD inefficiency on datasets with only two features (moreover, such datasets most likely do not need any dimensionality reduction). The peak of ratio (16) at $M = 100$ and $N = 2$ (it is the smallest dataset; it is not claimed to fit the category of large datasets) is another computational artifact probably caused by the peculiarities of starting the computations (primary compilation and memory allocation, non-optimized executable code prior to the first run, etc.) [Kontoghiorghes 2020; Romanuke 2023; 2024]. The smaller peak at $M = 6000$ and $N = 2$ is caused due to similar reasons.

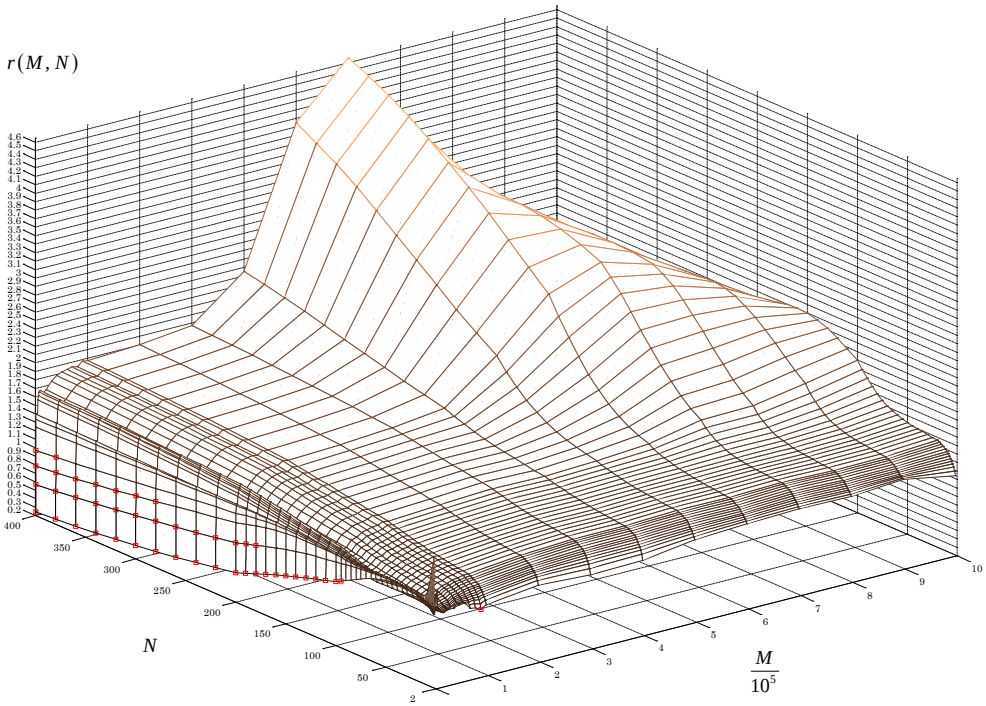

Fig. 3. The relative difference percentage (16) for datasets with double precision (the points of the EVD inefficiency are marked with squares)

URDPs (15) for double-precision datasets do not have any trends. The global maxima values of the maximal URDPs (15) among 185,000 double-precision datasets are

$$\max_{M \in \mathrm{M}} \max_{N \in \mathrm{N}} \delta_{\mathrm{PC}}(M, N) = 0.0031\%, \tag{19}$$

$$\max_{M\in\mathrm{M}}\max_{N\in\mathrm{N}}\delta_{\mathrm{PCS}}\left(M,N\right)=51.4355\,\%, \tag{20}$$

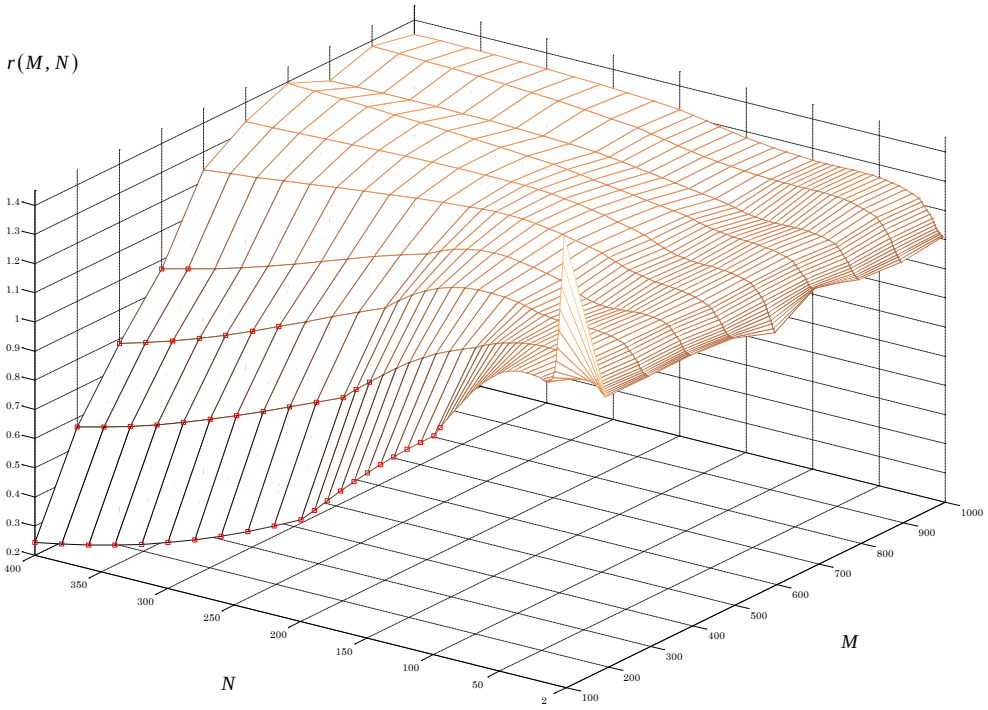

Fig. 4. A zoom-in on the relative difference percentage (16) in Figure 3 to show inefficiency of the EVD algorithm at (18) (the points of the EVD inefficiency are marked with squares)

$$\max_{M\in\mathrm{M}}\max_{N\in\mathrm{N}}\delta_{\mathrm{PCV}}\left(M,N\right)=5.2102\cdot10^{-9}\,\%, \tag{21}$$

$$\max_{M\in\mathrm{M}}\max_{N\in\mathrm{N}}\delta_{\mathrm{HT2V}}\left(M,N\right)=2.1451\cdot10^{-10}\,\%, \tag{22}$$

$$\max_{M\in\mathrm{M}}\max_{N\in\mathrm{N}}\delta_{\mathrm{TVEP}}\left(M,N\right)=5.2102\cdot10^{-9}\,\%, \tag{23}$$

where averaged maximal URDPs (15) are

$$\frac{1}{M\cdot N}\sum_{M\in\mathrm{M}}\sum_{N\in\mathrm{N}}\delta_{\mathrm{PC}}\left(M,N\right)=1.5617\cdot10^{-5}\,\%, \tag{24}$$

$$\frac{1}{M\cdot N}\sum_{M\in\mathrm{M}}\sum_{N\in\mathrm{N}}\delta_{\mathrm{PCS}}\left(M,N\right)=0.075004\,\%, \tag{25}$$

$$\frac{1}{M \cdot N} \sum_{M \in \mathrm{M}} \sum_{N \in \mathrm{N}} \delta_{\mathrm{PCV}}(M, N) = 6.6991 \cdot 10^{-12}\ \%, \tag{26}$$

$$\frac{1}{M \cdot N} \sum_{M \in \mathrm{M}} \sum_{N \in \mathrm{N}} \delta_{\mathrm{HT2V}}(M, N) = 1.0636 \cdot 10^{-12}\ \%, \tag{27}$$

$$\frac{1}{M \cdot N} \sum_{M \in \mathrm{M}} \sum_{N \in \mathrm{N}} \delta_{\mathrm{TVEP}}(M, N) = 6.6897 \cdot 10^{-12}\ \%. \tag{28}$$

Differences in PCVs, HT2Vs, and TVEPs are likely to occur for smaller-sized datasets [Jolliffe and Cadima 2016; Krishnan, Samaranayake and Jagannathan 2019; Romanuke 2019; Olivieri 2024]. The estimated probability of that PCVs, HT2Vs, and TVEPs computed by the EVD algorithm will differ from those by the SVD algorithm by 10^{-12} or more is less than 0.0023. Differences in principal components (2) and PCSs computed by the algorithms are far more probable. For instance, the estimated probability of that principal components (2) computed by the EVD algorithm will differ from those by the SVD algorithm by 10^{-9} or more is about 0.24. Such a probability for PCSs is about 0.33. This particularly explains the relatively huge URDP for PCSs in maximum percentages (19)–(23) and average percentages (24)–(28). Nevertheless, as the difference tolerance is relaxed from 10^{-9} up to 10^{-8}, the mentioned estimated probabilities for principal components (2) and PCSs become 0.0049 and 0.0081, respectively, i.e. they drop by 40 to 49 times versus raising the tolerance tenfold.

URDPs (15) for single-precision datasets do not have any trends either, but they are extremely poorer. For instance, the estimated probability of that principal components (2) computed by the EVD algorithm will differ from those by the SVD algorithm by 1% or more is about 0.205. This probability is about 0.555 and 0.91 for the difference tolerance of 0.1% and 0.01%, respectively. The single-precision SVD computational time (12) shown in Figure 5 is generally longer than that for EVD (Fig. 6), but it is shorter than the double-precision SVD computational time (Fig. 1).

Ratio (16) for the single-precision datasets shown in Figure 7 resembles those for double-precision datasets (Fig. 3), but it does not have a drop at (17) or anything like that. To the contrary, drops below 1 at (18) are very similar to those in Figure 3 (with the zoom-in shown in Fig. 4). Similar computational artifacts [Kontoghiorghes 2020; Romanuke 2023; 2024] are also noticeable at fewer observations (a few thousand) and nearly the fewest number of features (the front corner of the surface mesh).

Both the SVD and EVD algorithms have the same polynomial time complexity, regardless of precision [Zimand 2004; Enderton 2011; Downey 2024; Posthoff 2024]. Due to the PCA the computational time is statistically really unstable, it is

hard to ascertain the best degree of the fitting polynomial. It is either cubic or the fourth-degree polynomial at best, though. However, due to poor accuracy of the EVD algorithm in the computing principal components of single-precision datasets, its efficiency for dimensionality reduction of such datasets is unquestionably devastated.

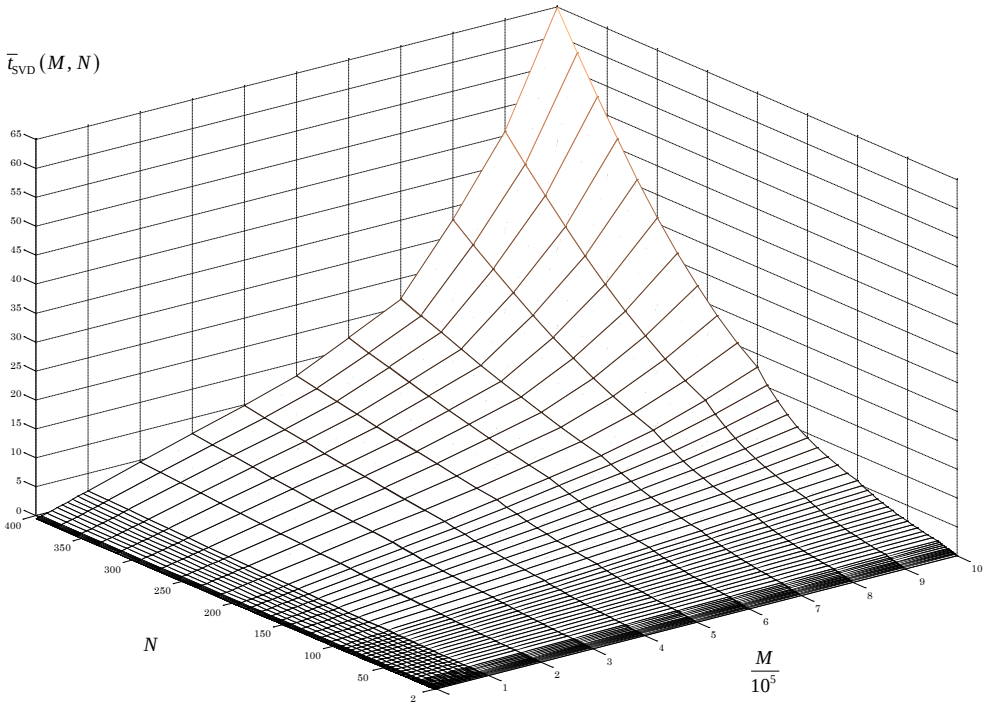

Fig. 5. The average computational time of the SVD algorithm for datasets with single precision

5. DISCUSSION

The obtained results visualized in Figures 1–7 confirm that, in computing principal components, the EVD algorithm is generally faster than the SVD algorithm, except for datasets of fewer observations or when the observation has fewer features (Fig. 3, 4, 7). In more detail, the EVD algorithm can be slower at (18), and it is scarcely faster if there are no more than 10^5 observations having two or three features. However, the task of the dimensionality reduction of a (large) dataset with two or three features is trivial, if not meaningless [Li et al. 2025; Meepaganithage, Nicolescu and Nicolescu 2025].

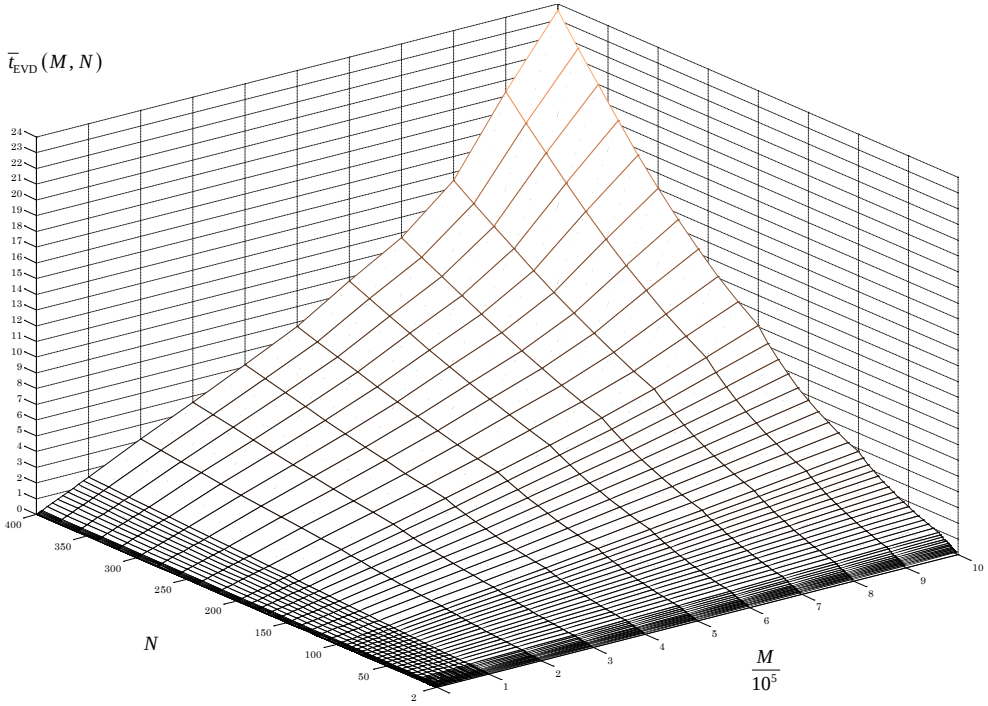

Fig. 6. The average computational time of the EVD algorithm for datasets with single precision

Apart from cases of shallower datasets consisting of just a few hundred double-precision observations, the EVD algorithm speeds up computing principal components at least by 4.5%, whereas the average speedup in 45% varies widely from 12% to 92%. The maximal speedup for 2.222 times varies between 1.266 and 4.655 times. The speedup on non-shallower single-precision datasets is roughly similar, but it nonetheless makes no sense due to EVD poor accuracy while operating on numeric data with single precision. Contrariwise, the EVD algorithm performs very accurately on double-precision datasets, so it is absolutely recommended to be applied to the dimensionality reduction of double-precision datasets larger than a few hundred double-precision observations.

It is noteworthy that the speedup falloff at (17) is plausibly explained by that the dataset size becomes too large for a RAM of 11.8 GB. The comparative analysis of SVD versus EVD has been carried out by two parallel processor workers [Kontoghiorghes 2020; Romanuke 2023], so it is naturally expected that the EVD algorithm will be more efficient for more parallel processor workers and larger RAM [Romanuke 2024].

It is worth remembering that the PCA is a linear dimensionality reduction technique, so the PCA relies on a linear model. If a dataset has a hidden nonlinear

pattern, then the PCA and subsequent dimensionality reduction of the dataset can devastate its information and the related knowledge from it [Jolliffe and Cadima 2016; Olivieri 2024]. Therefore, the EVD algorithm efficiency on non-shallower double-precision datasets exists by an additional condition of model linearity.

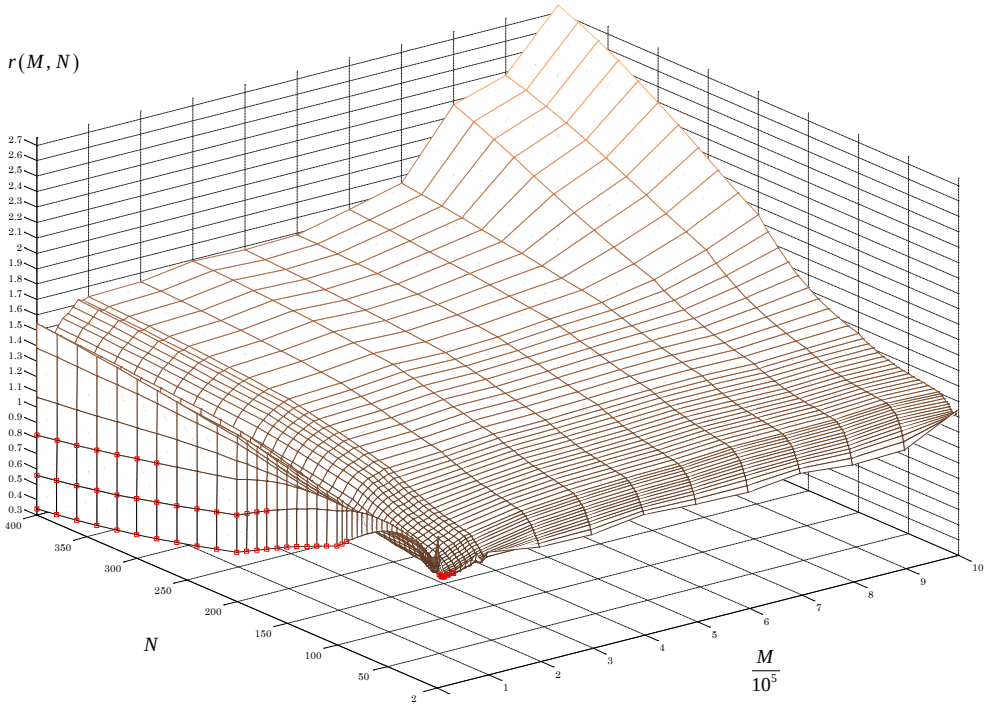

Fig. 7. The relative difference percentage (16) for datasets with single precision (the points of the EVD inefficiency are marked with squares)

6. CONCLUSION

In computing the principal components for dimensionality reduction of large datasets, the EVD algorithm is efficient if the dataset consists of no fewer than a few hundred observations (objects) having at least three double-precision features. The efficiency implies that the EVD algorithm computes faster than the SVD algorithm, without losing in accuracy. When a dataset has single-precision entries, the EVD algorithm is inaccurate and occasionally may be extremely inaccurate compared to SVD, so the latter remains the only exact algorithm to compute principal components for dimensionality reduction of single-precision datasets.

An open question, not answered yet, is how to efficiently parallelize both or at least one of the algorithms. The matter is their computational time complexity is

more like a polynomial, and extremely large datasets consisting of billions of entries can devastate the EVD algorithm efficiency. Having handled the largest dataset of 400 million entries with 11.8 GB RAM, the devastation already gradually starts for such a memory configuration at 120 million entries by 0.6 million observations. This open question may be particularly answered by applying the Tall Array approach, if only the dataset is stored on disk [Paluszek and Thomas 2024]. However, another open question arises about the Tall Array efficiency, where it is unclear when a dataset can be treated as large enough to be efficiently converted into a tall array. Besides, the parallelization of computing principal components should be studied also on how to implement the PCA on graphic processing units (GPUs). Therefore, a perspective for further research is to study parallelization of the PCA and, in particular, both SVD and EVD algorithms. Another perspective is to justify conditions, by which a dataset does not hide any significant nonlinear patterns and thus can be subject to linear dimensionality reduction by applying the PCA.

REFERENCES

Akter, S., Wamba, S.F., 2023, *Handbook of Big Data Research Methods*, Edward Elgar Publishing.

Demirbaga, Ü., Aujla, G.S., Jindal, A., Kalyon, O., 2024, *Big Data Analytics. Theory, Techniques, Platforms, and Applications*, Springer, Cham.

Downey, R., 2024, *Computational Complexity*, [in:] *Computability and Complexity. Undergraduate Topics in Computer Science*, Springer, Cham, pp. 159–176.

Enderton, H.B., 2011, *7 – Polynomial-Time Computability*, [in:] *Computability Theory*, Academic Press, pp. 139–149.

Golub, G.H., Van Loan, C.F., 1996, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, USA.

Härdle, W.K., Simar, L., Fengler, M.R., 2024, *Principal Component Analysis*, [in:] *Applied Multivariate Statistical Analysis*, Springer, Cham, pp. 309–345.

Holmes, M.H., 2023, *Eigenvalue Problems*, [in:] *Introduction to Scientific Computing and Data Analysis. Texts in Computational Science and Engineering*, vol. 13, Springer, Cham, pp. 129–204.

Jamarani, A., Haddadi, S., Sarvizadeh, R., Kashani, M.H., Akbari, M., Moradi, S., 2024, *Big Data and Predictive Analytics: A Systematic Review of Applications*, Artificial Intelligence Review, vol. 57, no. 176.

Jolliffe, I.T., Cadima, J., 2016, *Principal Component Analysis: A Review and Recent Developments*, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 374, no. 2065.

Kontoghiorghes, E.J., 2020, *Handbook of Parallel Computing and Statistics*, Chapman & Hall.

Krishnan, R., Samaranayake, V.A., Jagannathan, S., 2019, *A Hierarchical Dimension Reduction Approach for Big Data with Application to Fault Diagnostics*, Big Data Research, vol. 18, no. 100121.

Li, B., Liu, Z., Xu, Z., Li, J., Wang, B., Song, M., 2025, *Vehicle Appearance Dataset*, in: *Pattern Recognition and Computer Vision. PRCV 2024. Lecture Notes in Computer Science*, vol. 15039, Springer, Singapore, pp. 407–421.

Li, H., 2024, *Singular Value Decomposition*, [in:] *Machine Learning Methods*, Springer, Singapore, pp. 311–336.

Meepaganithage, A., Nicolescu, M., Nicolescu, M., 2025, *Enhanced Maritime Safety Through Deep Learning and Feature Selection*, [in:] *Advances in Visual Computing, ISVC 2024*, Lecture Notes in Computer Science, vol. 15047, Springer, Cham, pp. 309–321.

Oliveira, J.J. M., Cordeiro, R.L.F., 2020, *Unsupervised Dimensionality Reduction for Very Large Datasets: Are We Going to the Right Direction?* Knowledge-Based Systems, vol. 196, no. 105777.

Olivieri, A.C., 2024, *Principal Component Analysis*, [in:] *Introduction to Multivariate Calibration*, Springer, Cham, pp. 71–87.

Olszewski, D., 2025, *Asymmetry Index for Data and Its Verification in Dimensionality Reduction and Data Visualization*, Information Sciences, vol. 689, no. 121405.

Paluszek, M., Thomas, S., 2024, *Data for Machine Learning in MATLAB*, [in:] *MATLAB Machine Learning Recipes*, Apress, Berkeley, CA, USA, pp. 21–48.

Posthoff, C., 2024, *Polynomial and Exponential Complexity*, [in:] *Artificial Intelligence for Everyone*, Springer, Cham, pp. 41–62.

Romanuke, V.V., 2018, *Decision Making Criteria Hybridization for Finding Optimal Decisions' Subset Regarding Changes of the Decision Function*, Journal of Uncertain Systems, vol. 12, no. 4, pp. 279–291.

Romanuke, V.V., 2019, *Generator of a Toy Dataset of Multi-Polygon Monochrome Images for Rapidly Testing and Prototyping Semantic Image Segmentation Networks*, Electrical, Control and Communication Engineering, vol. 15, no. 2, pp. 54–61.

Romanuke, V.V., 2023, *Speedup of the k-means Algorithm for Partitioning Large Datasets of Flat Points by a Preliminary Partition and Selecting Initial Centroids*, Applied Computer Systems, vol. 28, no. 1, pp. 1–12.

Romanuke, V.V., 2024, *Deep Clustering of the Traveling Salesman Problem to Parallelize its Solution*, Computers & Operations Research, vol. 165, no. 106548, pp. 1–20.

Si-ahmed, I., Hamdad, L., Agonkoui, C.J., Kande, Y., Dabo-Niang, S., 2025, *Principal Component Analysis of Multivariate Spatial Functional Data*, Big Data Research, vol. 39, no. 100504.

Zimand, M., 2004, *Chapter 3 – Polynomial Time, Nondeterministic Polynomial Time, and Exponential Time*, [in:] North-Holland Mathematics Studies, vol. 196, North-Holland, The Netherlands, pp. 51–108.

Zizler, P., La Haye, R., 2024, *Singular Value Decomposition*, [in:] *Linear Algebra in Data Science. Compact Textbooks in Mathematics*, Birkhäuser, Cham, pp. 73–101.