The Journal of Open Source Software

SVDlab: A Reproducible Toolkit for SVD-based Image Compression, Denoising, and PCA with Adaptive Rank Selection

Gayane Ghazaryan[1] **and Artashes Ghazaryan**[2]

1 Institute of Physics, Yerevan State University, Armenia **2** Provectus Research

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 11 October 2025
Published: unpublished

Summary

SVDlab is an open-source Python toolkit that demonstrates how the **Singular Value Decomposition (SVD)** can be applied in three classical yet powerful contexts: **image compression**, **image denoising**, and **principal component analysis (PCA)**. It is designed as a compact and reliable framework that connects the mathematical theory of SVD with practical, fully reproducible workflows.

A key question in these applications is how many singular values to retain. **SVDlab** implements an **adaptive rank-selection rule** that automatically balances reconstruction quality and efficiency by combining a cumulative-energy threshold with elbow detection (*Kneedle*). The rule chooses $k^* = \max(k_\tau, k_e)$, providing stable results across datasets and domains (Eckart & Young, 1936; Jolliffe, 2002; Satopaa et al., 2011).

All figures, tables, and metrics — including PSNR, SSIM, energy retention, and runtime — can be **reproduced from four one-line commands**. The toolkit records metadata to trace every artifact back to its code, parameters, and dependency versions. Benchmarks against eigenvalue decomposition (EVD) and pivoted thin QR demonstrate consistent, transparent performance across noise levels and hardware backends (Gu & Eisenstat, 1996; Wang et al., 2004).

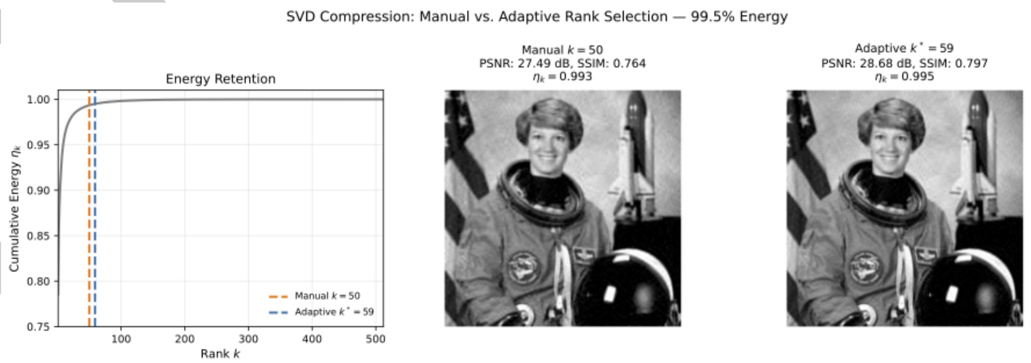

Figure 1: SVD compression comparison at 99.5 % retained energy. The adaptive rank (k*) selected by the Kneedle method achieves higher PSNR and SSIM than the fixed manual rank (k = 50). Reproducible via `code/svd_compression_merged.py`.

Statement of Need

Despite its central role in data analysis and imaging (Andrews & Patterson, 1976; Golub & Van Loan, 2013), the choice of rank in SVD-based methods remains inconsistent, hindering reproducibility and fair comparison across studies (Jolliffe, 2002).

SVDlab addresses this by offering a single, coherent toolkit that integrates preprocessing, adaptive rank selection, and reproducible artifact generation for multiple tasks. It aligns with the **FAIR principles** (Wilkinson et al., 2016) and current best practices in computational research transparency (Stodden et al., 2016).

The framework is intended for both researchers and educators who value auditable, theory-grounded experiments. It serves equally as a foundation for algorithmic research and for teaching laboratories where students can reproduce all results with minimal setup.

Novelty and Relation to Prior Work

While many libraries implement SVD or PCA, few unify them across tasks with a consistent design for **reproducibility and adaptive rank control**. **SVDlab** distinguishes itself through:

1. **Cross-task standardization** — one interface for compression, denoising, and PCA with identical preprocessing, metrics, and outputs.

2. **Adaptive, task-agnostic rank selection** — the hybrid cumulative-energy and elbow rule yields stable k^* across datasets, grounded in low-rank approximation theory (Eckart & Young, 1936; Satopaa et al., 2011).

3. **Reproducibility by design** — deterministic generation of all figures and tables from four simple commands (Stodden et al., 2016; Wilkinson et al., 2016).

4. **Methodological breadth** — standardized benchmarking of SVD, EVD, and QR factorizations with PSNR and SSIM evaluation (Gu & Eisenstat, 1996; Wang et al., 2004).

These elements turn linear algebra concepts into a transparent and hands-on computational toolkit suitable for both research and teaching.

Features

- **Adaptive rank selection** combining cumulative-energy and Kneedle-based elbow detection.

- **Four ready-to-use Python scripts** covering image compression, denoising, PCA, and benchmarking.

- **Deterministic results** ensured by fixed random seeds and pinned dependencies.

- **Automatic export** of all outputs as PDF, CSV, and LaTeX tables.

- **Cross-platform support** (Linux, macOS, Windows; Python 3.10).

- **Minimal execution effort** — all results from this paper can be reproduced with four simple commands.

- **Sensible defaults and fallbacks** — e.g., $\tau \approx 0.995$ for high-fidelity compression, $\tau \in [0.95, 0.99]$ for denoising; if the elbow is inconclusive, the energy rule applies.

- **PCA compatibility** — cumulative energy η_k coincides with explained variance for mean-centered data.

Adaptive rank selection summary: 1. Compute singular values $\{\sigma_i\}$ and cumulative energy $\eta_k = \sum_{i \le k} \sigma_i^2 / \sum_{i=1}^{r} \sigma_i^2$.

2. Threshold rule: $k_\tau = \min\{k : \eta_k \geq \tau\}$.
3. Elbow rule: detect k_e via Kneedle.
4. Default: $k^* = \max(k_\tau, k_e)$ (aggressive: $\min(k_\tau, k_e)$). If no elbow is detected, use k_τ and clip k to $[1, r]$.

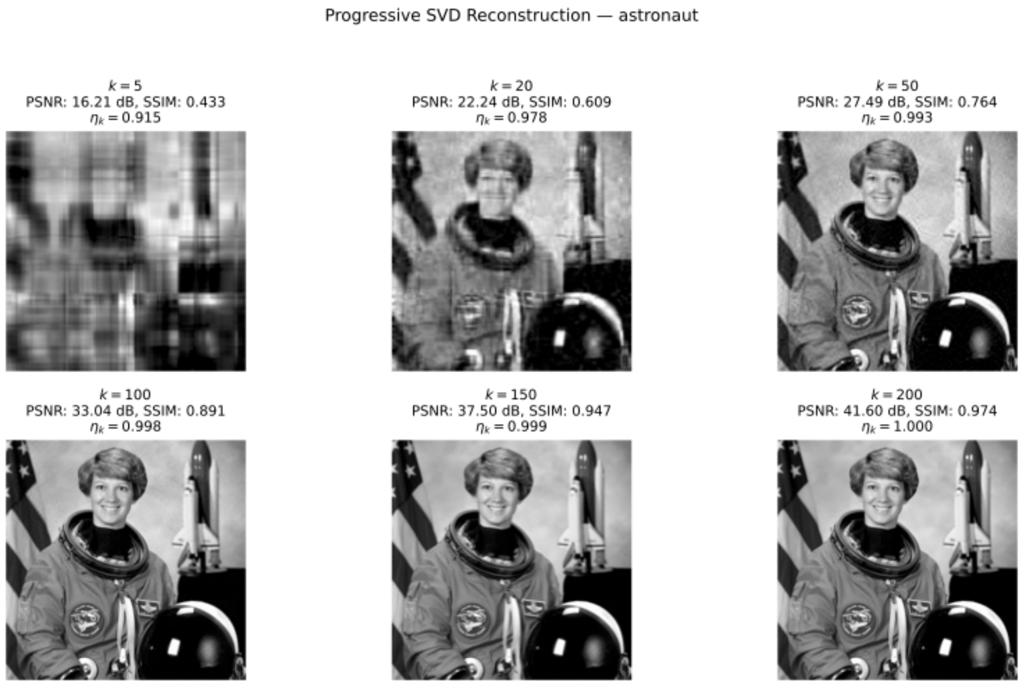

Figure 2: Progressive SVD reconstructions of the "astronaut" image at increasing ranks (k = 5 … 200), showing the trade-off between compression efficiency and visual fidelity. Generated via `code/svd_compression_merged.py`.

Example Usage

```
# 1) Image compression
python3 code/svd_compression_merged.py

# 2) Image denoising
python3 code/svd_denoising.py

# 3) Factorization benchmarks
python3 code/benchmark_and_plots.py

# 4) PCA with adaptive component selection
python3 code/pca_adaptive_combined.py
```

Each script automatically saves results under results/Figures/ and results/Tables/, allowing readers to reproduce every figure and table from a clean environment.

Limitations and Future Work

The current release focuses on exact (non-randomized) factorizations for moderate-scale problems. Future work will extend SVDlab with randomized and streaming variants (Halko et al., 2011), GPU acceleration, and support for color images and video — while maintaining full reproducibility and auditability.

Acknowledgements

We thank the open-source communities behind **NumPy**, **SciPy**, **scikit-image**, **scikit-learn**, and **Matplotlib**, whose work made this toolkit possible.
We also appreciate the support of colleagues at **Yerevan State University** and **Provectus** for their helpful feedback and collaboration.

For citation and reproducibility, please refer to the archived version of the software (Ghazaryan & Ghazaryan, 2025).
The complete source code and Zenodo release are available at
https://doi.org/10.5281/zenodo.17313445.

.

References

Andrews, H. C., & Patterson, C. L. (1976). *Singular value decomposition and digital image processing*. Cornell University, School of Electrical Engineering.

Eckart, C., & Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*(3), 211–218. https://doi.org/10.1007/BF02288367

Ghazaryan, G., & Ghazaryan, A. (2025). *SVDlab: A reproducible toolkit for SVD-based image compression, denoising, and PCA with adaptive rank selection* (Version 1.0.2). Zenodo. https://doi.org/10.5281/zenodo.17313445

Golub, G. H., & Van Loan, C. F. (2013). *Matrix computations* (4th ed.). Johns Hopkins University Press. https://doi.org/10.1201/9781420035827-8

Gu, M., & Eisenstat, S. C. (1996). Efficient algorithms for computing the rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, *17*(4), 848–869. https://doi.org/10.1137/S1064827592240555

Halko, N., Martinsson, P.-G., & Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, *53*(2), 217–288. https://doi.org/10.1137/090771806

Jolliffe, I. T. (2002). *Principal component analysis* (2nd ed.). Springer. https://doi.org/10.1007/b98835

Satopaa, V., Albrecht, J., Irwin, D., & Raghavan, B. (2011). Finding a "kneedle" in a haystack: Detecting knee points in system behavior. *Proceedings of the 31st International Conference on Distributed Computing Systems Workshops*, 166–171. https://doi.org/10.1109/ICDCSW.2011.20

Stodden, V., McNutt, M., Bailey, D. H., Deelman, E., Gil, Y., Hanson, B., Heroux, M. A., Ioannidis, J. P. A., & Taufer, M. (2016). Enhancing reproducibility for computational methods. *Science*, *354*(6317), 1240–1241. https://doi.org/10.1126/science.aah6168

Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612. https://doi.org/10.1109/TIP.2003.819861

Wilkinson, M. D., Dumontier, M., Aalbersberg, Ij. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., Silva Santos, L. B. da, Bourne, P. E., & others. (2016). The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18