QUASI-SVD: LEARNING A LIE-CONSTRAINED MATRIX FACTORISATION FOR REAL-TIME IMAGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Singular Value Decomposition (SVD) underlies matrix factorisation tasks across many fields, with imaging applications demanding real-time processing. Yet SVD algorithms are inherently sequential, constraining real-time GPU throughput and limit online deployment in imaging pipelines. This study introduces a fully parallelized matrix factorization framework for GPUs by enforcing matrix orthogonality on left singular vectors via Lie-parametrised algebra and recovering the remaining components through soft constraints. This asymmetric constraint design enables an efficient parallel and provably valid decomposition, achieves high reconstruction fidelity and substantially accelerates computation relative to the exact SVD, with real-time throughput exceeding standard video frame rates. Performance is evaluated on multiple imaging tasks spanning complementary computational regimes: (1) spatio-temporal background subtraction for ultrasound localisation microscopy, requiring high-dimensional matrix separation, (2) Mueller matrix polarimetry for neurosurgical tissue characterisation, requiring massive batch processing of small matrices, and (3) an MNIST denoising benchmark at an intermediate scale with known ground truth. Across regimes and instruments, the proposed framework demonstrates robust domain transfer at various matrix scales, sufficient for live image-guided workflows that classical solvers cannot currently support in these settings. By prioritising downstream reconstruction fidelity over exact spectral recovery, the proposed SVD framework makes structured matrix factorisation practical for real-time processing.

1 INTRODUCTION

Today's imaging pipelines increasingly rely on GPU acceleration to achieve real-time processing and playback. Yet a standard matrix decomposition in scientific imaging, the singular value decomposition (SVD), remains difficult to parallelise. The SVD lies at the computational core of applications ranging from spatio-temporal filtering in ultrasound localisation microscopy (ULM) Demené et al. (2015); Errico et al. (2015); Baranger et al. (2018); Hahne et al. (2024) to Mueller matrix polarimetry (MMP) for surgical guidance Lu & Chipman (1996); Moriconi et al. (2024); Hahne et al. (2025a). The paradox is that the hardware has become massively parallel while the algorithms we rely on still employ fundamentally sequential and iterative procedures, leaving SVD as a bottleneck in pipelines that otherwise operate in real-time. For example, the SVD can throttle MMP to approximately 4 FPS Moriconi et al. (2024); Hahne et al. (2025a) or force ULM background suppression into an offline step Demené et al. (2015); Hahne et al. (2024), well below the throughput required for live image-guided decision-making.

This bottleneck is a result of the structural concept underlying the SVD. Classical SVD algorithms Golub & Reinsch (1970); Hansen (1987; 1990); Halko et al. (2011); Musco & Musco (2015); Łukasz Struski et al. (2024) rely on sequential transformations and iterative refinement that impose data dependencies between computational steps, preventing full use of the thousands of cores available on modern GPUs. The problem manifests at two computationally relevant extremes: (a) high-throughput batches of small matrix decompositions and (b) memory-intensive decompositions of high-dimensional data, as in temporal subspace filtering over hundreds of frames. Neither is well served by solvers designed around sequential bidiagonalisation, and the resulting latency is what currently keeps SVD-dependent imaging pipelines offline.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Existing GPU-parallel implementations improve throughput Lahabar & Narayanan (2009); NVIDIA Corporation (2026); Demmel & Veselić (1992); Novakovic (2015) but retain algorithmic dependencies that limit scalability, particularly across the range of matrix sizes relevant to imaging. Learning-based approaches offer a more GPU-friendly alternative Xu et al. (2022); Ryu et al. (2024), but typically enforce orthogonality through soft optimisation or avoid explicit factor recovery, leaving numerical validity insufficiently constrained. Conversely, Lie-theoretic parameterisations provide matrix orthogonality, but have not been applied to decomposition algebra until now.

This work closes that gap with a Lie-theoretic, fully differentiable matrix factorisation framework that resolves both limitations simultaneously. It enforces *exact* orthogonality on a single factor via the matrix exponential over a skew-symmetric Lie algebra, while recovering the remaining components through soft constraints. This asymmetric construction is provably sufficient for factorisation, and it deliberately trades exact spectral recovery for reconstruction fidelity and throughput, a trade-off validated directly on clinically motivated image regimes and generic benchmarks. This reframes the SVD as a deployment-oriented design choice rather than a purely numerical one.

This study makes the following contributions. **(i) Orthogonality via Lie theory**: the first application of matrix exponentials from Lie algebra to enforce orthogonality on a single factor of a learned SVD decomposition, eliminating the numerical drift that affects approaches enforcing orthogonality only through soft penalties on all factors. **(ii) GPU-native parallelisable architecture**: a fully differentiable, trainable framework that achieves > 3–$20\times$ speedup over algorithmic baselines while maintaining decomposition accuracy, demonstrated on real imaging data spanning complementary matrix dimensions (3×3 to 500×500). **(iii) Practical impact on real-time imaging**: validation across ULM and MMP shows throughput exceeding 25 FPS at clinical matrix scales, making previously offline or sub-real-time pipelines deployable, including on cost-constrained deployment.

2 Related work

Traditional SVD Methods Singular Value Decomposition decomposes a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ as $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U} \in \mathrm{O}(m)$ and $\mathbf{V} \in \mathrm{O}(n)$ are orthogonal matrices and $\mathbf{\Sigma}$ contains ordered singular values Golub & Van Loan (2013). The full SVD has cubic time complexity $\mathcal{O}(\min(m,n)\cdot mn)$ via Householder bidiagonalisation and QR iteration Golub & Reinsch (1970), which impose sequential dependencies for batched matrix factorisations on modern parallel hardware and thus limit the throughput Lahabar & Narayanan (2009). The computational burden is further compounded when moving to high-order arrays, where Tensor SVD Zhang & Xia (2018) encounters a fundamental statistical-computational gap that renders optimal recovery NP-hard under moderate signal-to-noise ratios. Approximation methods like Truncated SVD Hansen (1987; 1990); Stoll (2012) and Randomised SVD Halko et al. (2011); Musco & Musco (2015) reduce complexity but retain sequential bottlenecks that prevent efficient GPU parallelisation. Rank–revealing QR still hinges on pivoted column selection heuristics that require dependent branching and synchronization Chan (1987); Gu & Eisenstat (1996). Nyström methods only postpone the problem by reducing the effective matrix size Drineas et al. (2006); Musco & Musco (2017), but the subsequent SVD on the reduced core is still *sequential* and remains the runtime limit. As alternatives, Streaming SVD updates decomposition dynamically as new data arrives Brand (2006) and k-SVD employs an iterative framework to learn sparse representations by optimizing dictionary atoms Aharon et al. (2006); Scetbon et al. (2021). Despite their sophistication, these methods do not resolve the core structural barrier, which is to map cleanly to highly parallel computation architectures.

Parallel SVD Implementations To overcome sequential bottlenecks, CUDA-based implementations Lahabar & Narayanan (2009) accelerate SVD for large matrices using partial bidiagonalisation followed by QR or divide-and-conquer routines, although Householder transformations remain inherently sequential and limit parallelism Golub & Reinsch (1970); Novakovic (2015). Jacobi-based algorithms Demmel & Veselić (1992); Golub & Van Loan (2013); Novakovic (2015) offer a more parallelisable alternative via rotations over independent 2×2 pivot submatrices. Modern GPU libraries such as NVIDIA cuSOLVER implement Jacobi-type solvers (e.g., `gesvdj`) NVIDIA Corporation (2026), but these batched routines are practically restricted to small matrices ($m, n \leq 32$), falling back to sequential bidiagonalisation-based SVD for larger matrices.

Learning-Based Approaches Recent work has explored neural networks for SVD-related tasks. SV-Learn Xu et al. (2022) uses MLPs to predict singular values but suffers from numerical instability due to soft constraints and cannot recover singular vectors. Dictionary-learning methods such as Deep K-SVD Scetbon et al. (2021) produce sparse representations via iterative dictionary updates which rely on solvers with loop-heavy control flow that resists the parallelism targeted here. Operator-SVD Ryu et al. (2024) learns spectral decompositions of linear operators through nested low-rank approximation, achieving theoretical elegance but at significant computational cost unsuitable for real-time imaging. Beyond imaging, SVD also underlies spectral clustering layers, though the high computational cost of the decomposition often necessitates its replacement with dual autoencoder architectures to maintain end-to-end efficiency Yang et al. (2019). In the large language model (LLM) domain, SVD has attracted attention for activation whitening Yuan et al. (2023), low-rank weight approximation Wang et al. (2026), and truncation-aware model compression Yuan et al. (2023); Wang et al. (2025), underscoring the broad demand for efficient factorisation. More broadly, end-to-end (E2E) methods learn a desired reconstruction mapping directly from data offering fast inference capabilities for images, as for example Learned SVD (L-SVD) Boink & Brune (2019) or SVD-inspired filter networks Cole et al. (2022) included as learning-based baselines. In contrast, the proposed study retains an explicit, mathematically principled matrix-factorisation structure while approximating the decomposition to achieve the throughput required for real-time imaging.

Lie Groups and Orthogonal Constraints The matrix exponential and Cayley transform map Lie algebras onto orthogonal groups, providing hard matrix orthogonality constraints. These tools are established in visual odometry Whelan et al. (2015); Teed & Deng (2021); Hayoz et al. (2023); Matsuki et al. (2024) and in orthogonality-constrained layers Lezcano-Casado & Martınez-Rubio (2019); Li et al. (2020), yet no prior work uses them to enforce orthogonality in a learned decomposition of streaming matrices. Learned SVD methods instead impose orthogonality only implicitly through unconstrained optimisation Ryu et al. (2024) or omit it entirely Xu et al. (2022), accumulating numerical drift. No prior method therefore satisfies both requirements of parallelism and orthogonality guarantees simultaneously.

3 PROPOSED METHOD

This work targets an efficient, learnable SVD approximation that preserves reconstruction and downstream accuracy while minimizing GPU wall-clock latency and keeping training cost practical. The Quasi-SVD framework comprises a spectrum of module variants (illustrated in Fig. 1): analytic closed-form, unconstrained neural net (UNN), Lie-informed parametrisations, and end-to-end prediction where each trades off accuracy, latency, and training effort. Full derivations, work–span analyses, and implementation details appear in the supplement.

General notation In close analogy to the SVD, a factorisation of matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ is given by:

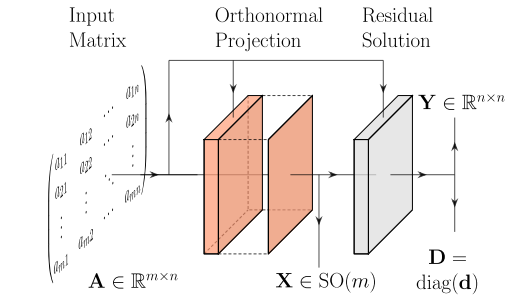
$$\mathbf{A} = \mathbf{X}\mathbf{D}\mathbf{Y}^\top, \qquad \text{where} \quad \mathbf{D} = \mathrm{pdiag}(\mathbf{d}) = \begin{bmatrix} \mathrm{diag}(\mathbf{d}) \\ \mathbf{0}_{(m-n)\times n} \end{bmatrix} \in \mathbb{R}^{m\times n} \tag{1}$$

with $\mathbf{X} \in \mathrm{SO}(m)$, $\mathbf{Y} \in \mathbb{R}^{n\times n}$, and $\mathbf{d} \in \mathbb{R}^n$ collecting the diagonal entries, i.e., $\mathbf{d} = \mathrm{pdiag}^{-1}(\mathbf{D}) = [d_1, \ldots, d_n]^\top$, ordered in descending magnitude. Here, $\mathrm{pdiag}(\cdot) : \mathbb{R}^n \to \mathbb{R}^{m\times n}$ zero-pads $\mathrm{diag}(\cdot)$ with $m - n$ rows below and $\mathrm{pdiag}^{-1}(\cdot)$ recovers the diagonal entries of the leading $n \times n$ block, while $m \geq n$ without loss of generality (if $m < n$, transpose $\mathbf{A}$ and swap the resulting factors back).

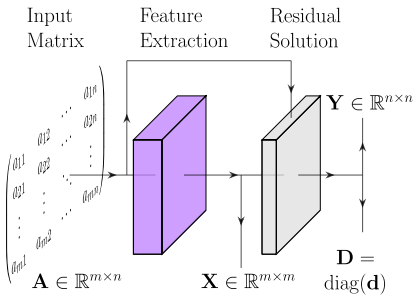
Analytic decomposition Orthogonality is central in many factorisation schemes underlying SVD. The present formulation enforces this structure directly inside the corresponding Lie group and parametrises orthogonal matrices through the Lie algebra $\mathfrak{so}(m)$. A skew-symmetric matrix $\mathbf{S} \in \mathfrak{so}(m)$ ensures that $\exp(\mathbf{S}) \in \mathrm{SO}(m)$, making the matrix exponential an analytic mechanism for generating valid left/right factors. Given a matrix $\mathbf{A}$, the construction proceeds as follows:

$$\mathbf{X} = \exp(\mathbf{S}) \ , \quad \text{where} \quad \mathbf{S} = \mathbf{A} - \mathbf{A}^\top \ , \tag{2}$$

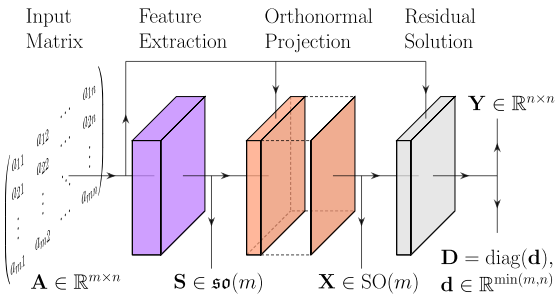
where $\mathbf{S} + \mathbf{S}^\top = \mathbf{0}$ enforces skew-symmetry. The exponential map $\exp(\cdot) : \mathfrak{so}(m) \mapsto \mathrm{SO}(m)$ guarantees $\mathbf{X}$ is orthogonal. For rectangular inputs ($m \neq n$), $\mathbf{A}$ is zero-padded by $m - n$ columns to form a square matrix $\mathbf{A}_{\mathrm{pad}} \in \mathbb{R}^{m\times m}$ prior to constructing $\mathbf{S}$.



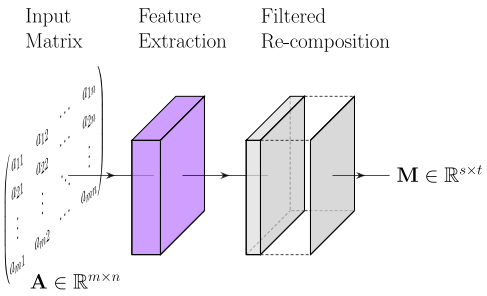
(a) Analytic model



(b) Unconstrained neural net (UNN)



(c) Lie-based neural net (LieNN)



(d) End-to-end network (E2E)

Figure 1: **Model variants.** The analytic model in (a) provides a closed-form solution from the matrix exponential for orthonormal projection without learnable modules. The unconstrained neural network (UNN) in (b) introduces trainable weights to factorise the input matrix. The Lie-based neural network (LieNN) in (c) predicts skew-symmetric parameters mapped to an orthogonal factor via the matrix exponential, combining learned flexibility with hard orthogonality guarantees. The end-to-end (E2E) network in (d) omits mathematical constraints entirely, producing only the composed matrix $\mathbf{M}$. While E2Es have been studied Boink & Brune (2019); Cole et al. (2022), models in (a) and (c) contain the proposed Lie constraint. All model variants are used for an ablation analysis.

For higher-dimensional cases, the computational burden of the matrix exponential can be reduced by using the order-p Taylor expansion $\exp_t(\cdot)$ as an approximation and, in the special case where $\mathbf{S} \in \mathfrak{so}(3)$, the Rodrigues' formula $\exp_3(\cdot)$ as a closed-form expression:

$$\exp_t(\mathbf{S}, p) = \sum_{r=0}^{p} \frac{\mathbf{S}^r}{r!}, \qquad \text{while} \qquad \exp_3(\mathbf{S}) \overset{\mathbf{S} \in \mathfrak{so}(3)}{=} \mathbf{I} + \frac{\sin\theta}{\theta}\mathbf{S} + \frac{1-\cos\theta}{\theta^2}\mathbf{S}^2, \tag{3}$$

where $\theta = \|\mathbf{S}\|_F / \sqrt{2}$ and $\|\cdot\|_F$ denotes the Frobenius norm. These yield a differentiable, algebraically grounded parameterisation of orthogonal matrix components with determinant $+1$.

From $\mathbf{X}$, the right factor $\mathbf{Y}$ and the diagonal entries $\mathbf{d}$ are estimated via:

$$\mathbf{Y} = \mathbf{A}^\top \mathbf{X}_n \mathbf{D}^\dagger, \quad \text{where} \quad \mathbf{D}^\dagger = \mathrm{diag}(\mathbf{d}^{-1}), \quad \text{and} \quad \mathbf{d} = \sqrt{\left|\, \mathrm{diag}^{-1}(\mathbf{X}_n^\top \mathbf{A}\mathbf{A}^\top \mathbf{X}_n)\right| + \varepsilon}\,, \tag{4}$$

with $\mathbf{X}_n \in \mathbb{R}^{m \times n}$ denoting the first n columns of $\mathbf{X}$. Here, $\varepsilon = 10^{-9}$ avoids a gradient singularity in $\sqrt{\cdot}$ near zero and $\mathbf{d}$ is separately clamped at 10^{-11} before inversion to prevent division by zero.

Unconstrained neural decomposition Extending on the work of SV-Learn Xu et al. (2022), a viable design choice for full neural SVD prediction is to let a network g_θ predict singular vectors:

$$\mathbf{X}_g = g_\theta(\mathbf{A}) \quad \text{where} \quad g_\theta : \mathbb{R}^{m \times n} \mapsto \mathbb{R}^{m \times m} \tag{5}$$

with $\mathbf{X}_g \in \mathbb{R}^{m \times m}$. To satisfy $\mathbf{A} = \mathbf{X}_g \mathbf{D}_g \mathbf{Y}_g^\top$, one borrows the concepts from Eq. (4) to compute $\mathbf{D}_g$ and $\mathbf{Y}_g$, respectively. This unconstrained model is illustrated in Fig. 1b.

Lie-informed neural decomposition To enforce matrix orthogonality in a learning context, a neural model f_θ predicts the upper triangle of the skew-symmetric matrix $\mathbf{s}$ via:

$$\mathbf{s} = f_\theta(\mathbf{A}) \quad \text{where} \quad f_\theta : \mathbb{R}^{m\times n} \mapsto \mathbb{R}^L \tag{6}$$

where $\mathbf{s} = [s_1, s_2, \ldots, s_L]^\top$ contains L elements that are scattered into the upper triangle of the skew-symmetric matrix through the mapping $F : \mathbb{R}^L \mapsto \mathfrak{so}(m)$, with $F(\mathbf{s})_{ij} = s_k$ for $i < j$ (and $F(\mathbf{s})_{ji} = -s_k$), k indexing the upper-triangular entries in row-major order. From this, the Lie-based neural network (LieNN) predicts the matrix $\mathbf{X}_f$ via:

$$\mathbf{X}_f = \exp\big(F(\mathbf{s})\big), \tag{7}$$

which is then used to compute $\mathbf{D}_f$ and $\mathbf{Y}_f$ analogous to Eq. (4). Implementation variants for the matrix exponential are detailed in Supp. Sec. B.1.

Lemma 1 (Asymmetric orthogonality suffices)**.** *For any SVD* $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ *with* $\det(\mathbf{U}) = -1$*, jointly flipping the sign of one corresponding column of* $\mathbf{U}$ *and* $\mathbf{V}$ *yields an equivalent factorisation* $\mathbf{A} = \tilde{\mathbf{U}}\mathbf{\Sigma}\tilde{\mathbf{V}}^\top$ *with* $\tilde{\mathbf{U}} \in \mathrm{SO}(m)$*. A left factor in* $\mathrm{SO}(m)$*, expressible as* $\exp(\mathbf{S})$ *for skew-symmetric* $\mathbf{S}$*, therefore always exists, while the corresponding right factor may lie in the determinant* (-1) *component of* $\mathrm{O}(n)$*. Full proof in Supp. Sec. A.3.*

This justifies treating $\mathbf{X}_f$ and $\mathbf{Y}_f$ asymmetrically during training: $\mathbf{X}_f$ is exactly orthogonal by construction, while $\mathbf{Y}_f$ is only encouraged towards orthogonality through a soft penalty. The mathematical properties on singular values and right-factor orthogonality are therefore left as soft constraints using the following learning objectives. (1) A singular value loss:

$$\mathcal{L}_\sigma = \left\|\mathrm{pdiag}^{-1}(\mathbf{\Sigma}) - \mathrm{pdiag}^{-1}(\mathbf{D}_f)\right\|_2^2 \tag{8}$$

(2) An off-diagonal zero loss with an operator $\mathrm{off}(\mathbf{Z}) = \mathbf{Z} - \mathrm{diag}\left(\mathrm{diag}^{-1}(\mathbf{Z})\right)$ is given by:

$$\mathcal{L}_{\text{off}} = \left\|\mathrm{off}\left(\mathbf{X}_f^\top \mathbf{A}\mathbf{A}^\top \mathbf{X}_f\right)\right\|_F^2 + \left\|\mathrm{off}\left(\mathbf{Y}_f^\top \mathbf{A}^\top \mathbf{A}\mathbf{Y}_f\right)\right\|_F^2 \tag{9}$$

(3) An orthogonality penalty on $\mathbf{Y}_f$:

$$\mathcal{L}_{\text{ort}} = \left\|\mathbf{Y}_f\mathbf{Y}_f^\top - \mathbf{I}\right\|_2^2 \tag{10}$$

where $\mathbf{I} \in \mathbb{R}^{n\times n}$ is an identity matrix. These losses are aggregated for a total loss given by

$$\mathcal{L}_T = \lambda_1\mathcal{L}_\sigma + \lambda_2\mathcal{L}_{\text{off}} + \lambda_3\mathcal{L}_{\text{ort}} \tag{11}$$

where $(\lambda_1, \lambda_2, \lambda_3)$ are the regularization constants.

End-to-end (E2E) learning Many applications of the SVD involve reconstructing a filtered version $\mathbf{M}$ of an input matrix $\mathbf{A}$. For instance, the Lu-Chipman decomposition in Mueller matrix imaging produces $\mathbf{M} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ after factorisation where $\mathbf{D} = \mathrm{diag}(\mathbf{d})$ and $\mathbf{d} \in \{-1, 1\}^m$ Lu & Chipman (1996). Scenarios like this offer the opportunity to learn the overarching objective of an SVD in an end-to-end (E2E) fashion by a function $E_\theta : \mathbb{R}^{m\times n} \mapsto \mathbb{R}^{s\times t}$. The loss then simplifies to:

$$\mathcal{L}_{\text{E2E}}(\mathbf{M}, \mathbf{A}) = \|\mathbf{M} - E_\theta(\mathbf{A})\|_2^2 \tag{12}$$

as the distance between the network's prediction and final reconstruction matrix $\mathbf{M}$ for which the SVD is used in the processing chain. A key advantage of this end-to-end approach is that operations beyond the SVD can be learned implicitly by $E_\theta(\mathbf{A})$, potentially improving computational efficiency. However, this comes at the cost of explicit factor recovery and interpretability, and risks poor generalisation from a potential inability to learn underlying patterns.

Neural module architectures Each learned model variant employs an MLP with a single hidden layer of size $h = \max(64, m/2)$, where m determines input and output dimensions. To emulate iterative convergence of classical SVD solvers, the MLP is optionally augmented with a three-step RNN that refines input matrix $\mathbf{A}$, where gated recurrent units (GRUs) are used due to balanced accuracy and efficiency Foucault & Meyniel (2021); Hassaan et al. (2025).

4 EXPERIMENTS

4.1 DATASETS AND ASSESSMENT

The selected datasets represent complementary SVD workload regimes. The polarimetric dataset requires performing a very large number of small decompositions, whereas the ultrasound dataset involves fewer but substantially larger matrices and the MNIST dataset lies inbetween. Together, these cases capture the range of memory and performance trade-offs targeted by the proposed method.

Polarimetric data MMP is an imaging technique with growing diagnostic relevance in medical imaging Sampaio et al. (2023); Moriconi et al. (2024). For visualization, the Mueller matrix is typically decomposed via SVD following Lu and Chipman Lu & Chipman (1996), enabling interpretation of polarimetric features. In neuropathology, the per-pixel i azimuth angle $\varphi_i = {}^1\!/_2 \tan^{-1}(M_i^{(2,4)}/M_i^{(4,3)})$ derived from $\mathbf{M}_i$ in Eq. (12) encodes fiber orientation, a biomarker for tumor identification Hahne et al. (2025a). Although near real-time MMP has been demonstrated Moriconi et al. (2024), instant visualization of Lu–Chipman features remains challenging. The proposed Quasi-SVD model is evaluated on the NeuroPathoPol (NPP) dataset, with the public mouse uterine cervix (MUC) data Novikova et al. (2024); Pogudin (2024) reserved for domain-transfer analysis. NPP comprises 388×516-pixel images, each pixel associated with a 4×4 Mueller matrix $\mathbf{B}_i = B_i^{(u,v)}$. Training and validation use 12 tumor and 9 non-tumor brain samples, while testing employs 5 tumor and 3 healthy samples. For each pixel i, a normalised input matrix $\mathbf{A}_i \in \mathbb{R}^{3\times3}$ is derived from $\mathbf{B}_i$ as its bottom-right 3×3 block scaled by the $(1,1)$ entry, $\mathbf{A}_i = \mathbf{B}_i^{(2:4,\,2:4)}/B_i^{(1,1)}$. Data augmentation through random rotation and flipping increases the diversity of training samples Hahne et al. (2025b).

MNIST denoising data As a controlled benchmark with known ground truth, a denoising task is constructed from MNIST handwritten digits (28×28 grayscale). A 10×10 tile grid assembles 100 randomly permuted digits into a single 280×280 matrix, providing an intermediate scale regime. Each tile receives independent Gaussian noise ($\sigma = 0.2$), with per-epoch D4 dihedral transforms and a random background offset drawn from $[0, 0.5]$ applied to clean tiles *before* noise injection; tile order is shuffled per sample to prevent memorisation and penalise all-zero matrices.

Ultrasound data Many imaging applications employ the SVD for background removal Xin et al. (2015). In ULM, the SVD is a key processing step to suppress static reflectors (e.g., bone) and isolate microbubble flow, commonly realised through the Karhunen–Loève transform (KLT) Demené et al. (2015); Baranger et al. (2018); Heiles et al. (2022). Given a Casorati matrix $\mathbf{B}_c \in \mathbb{R}^{h\times n}$ constructed from n frames with h pixels, each $\mathbf{B}_c$ forms the Hermitian matrix $\mathbf{A} = \mathbf{B}_c^\top \mathbf{B}_c$ and uses its singular vectors for subspace denoising. Evaluation uses publicly available *in vivo* ULM data from the PALA study Heiles et al. (2022), comprising rat brain perfusion sequences acquired with a 15.6 MHz linear probe (128 elements, 0.1 mm pitch) at 1 kHz frame rate with multi-angle plane waves. B-mode data from *rat 18* is employed, containing 250 sequences of 500 frames each. The first 20 sequences are used for training and validation (0.9 split), and the final 5 for testing. Each B-mode image contains $h = 143 \times 167$ pixels, yielding $\mathbf{A} \in \mathbb{R}^{500\times500}$ from $n = 500$ stacked frames.

Metrics Reconstruction quality and decomposition accuracy are reported using both perceptual and numerical metrics. The structural similarity index (SSIM) measures local luminance, contrast, and structure using an 11-pixel Gaussian window. For matrix-level assessment, a relative Frobenius error is defined as $\|\mathbf{D}\|_F^{\text{rel}} = \|\mathbf{D} - \mathbf{D}^{(gt)}\|_F / \|\mathbf{D}^{(gt)}\|_F$ and $\|\mathbf{Y}\|_F^{\text{ort}} = \|\mathbf{Y}\mathbf{Y}^\top - \mathbf{I}\|_F / \|\mathbf{I}\|_F$ from the Frobenius norm $\|\cdot\|_F$ with $^{(gt)}$ denoting the ground truth (GT) and $\mathbf{I}$ the identity matrix.

Training and testing Models are trained using AdamW with gradient clipping and cosine-annealed learning rates converging to zero. Weights are initialised with Xavier (Glorot) uniform for linear layers, biases drawn from $\mathcal{N}(0, 10^{-6})$, and batch-norm weights/biases set to 1 and 0, respectively. Polarimetric data uses an Nvidia RTX 4090, batch size 776, 40 epochs, initial learning rate $1\text{e}{-3}$, with best checkpoint by lowest $\mathcal{L}_{\text{ort}}$ from Eq. (10) (or validation $\|\varphi_i - \varphi_i^{(gt)}\|_2^2$ when training with $\mathcal{L}_{\text{E2E}}$). MNIST data uses an Nvidia T500 mobile GPU, batch size 32, 200 epochs, initial learning rate $1\text{e}{-4}$, with best checkpoint by lowest validation SSIM. Ultrasound data uses an Nvidia H100, batch size 1, 5 epochs, initial learning rate $1\text{e}{-5}$, with best checkpoint by lowest $\mathcal{L}_T$ from

Eq. (11) (or validation $\mathcal{L}_{\text{E2E}}$ when training with $\mathcal{L}_{\text{E2E}}$). Runtimes are measured on identical batched inputs and hardware using synchronized wall-clock timing, including method-specific overheads.

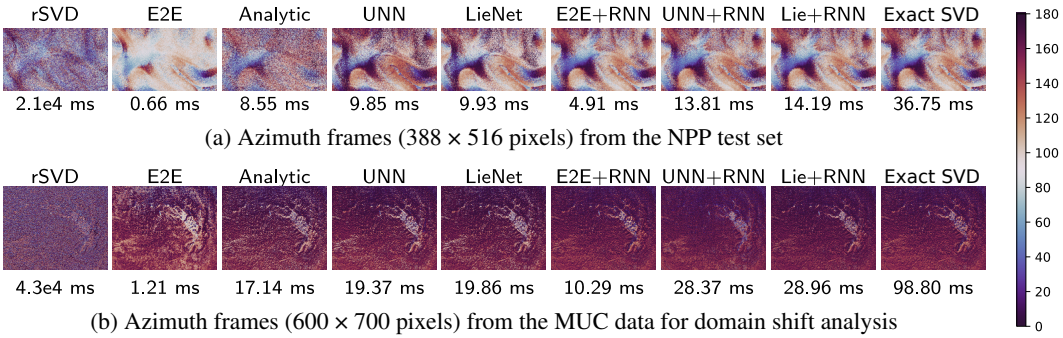
Comparison methods The proposed framework is evaluated against classical and learning-based SVD techniques: (i) Exact SVD, a reference implementation via cuSOLVER NVIDIA Corporation (2026) using one-sided Jacobi SVD for small matrices ($m, n \leq 32$) and traditional decomposition otherwise Golub & Reinsch (1970), serving as the numerical baseline for accuracy; (ii) SV-Learn, an MLP-based predictor of singular values only, without singular vector reconstruction Xu et al. (2022); (iii) end-to-end (E2E) networks, as in L-SVD Boink & Brune (2019); Cole et al. (2022), trained to directly reconstruct SVD-filtered outputs; and (iv) rSVD, a stochastic low-rank approximation Halko et al. (2011) (PyTorch), with default settings except oversampling (q capped at $\min(m, n)$, set to the SVD-optimal rank for MNIST) and zero-padding factors to full size for scoring. For ablation, each learned model is extended with an RNN, and the analytical and UNN variants are included.

4.2 Results

This section outlines general experimental trends followed by domain-specific outcomes. Numerical evaluations for both MMP and ULM are summarised in Tables 1, 3, and 4, with qualitative comparisons in Figs. 2, 3 and 4. Overall, RNN-enabled models deliver the highest reconstruction fidelity

Table 1: **NPP test set ($\mathbf{A}_i \in \mathbb{R}^{3\times3}$) for Lu-Chipman decomposition.** Mean ± std across frames (– for not applicable; std omitted when 0). **Bold** best, shaded second-best non-Exact-SVD score.

Method	RNN	$\overline{\text{SSIM}(\mathbf{M})}$ ↑	$\overline{\|\mathbf{X}\|_F^{\text{ort}}}$ ↓	$\overline{\|\mathbf{D}\|_F^{\text{rel}}}$ ↓	$\overline{\|\mathbf{Y}\|_F^{\text{ort}}}$ ↓	Time [ms] ↓
Exact SVD	✗	1.00	0.00	0.00	0.00	36.98 ± 1.81
rSVD	✗	0.15 ± 0.05	**0.00**	**0.00**	**0.00**	2.12e4±92.42
SV-Learn	✗	0.65 ± 0.08	–	0.50 ± 0.01	–	1.89
E2E	✗	0.75 ± 0.07	–	–	–	**0.66**
Analytic	✗	0.66 ± 0.08	**0.00**	0.09 ± 0.04	0.46 ± 0.08	8.48 ± 0.02
UNN	✗	0.82 ± 0.05	0.13 ± 0.04	0.03 ± 0.01	0.23 ± 0.07	9.85 ± 0.01
LieNN	✗	0.83 ± 0.06	**0.00**	0.03 ± 0.01	0.24 ± 0.07	9.93 ± 0.02
E2E	✓	**0.98 ± 0.01**	–	–	–	4.91
UNN	✓	0.85 ± 0.04	0.14 ± 0.03	0.02	0.08 ± 0.02	13.81 ± 0.01
LieNN	✓	0.94 ± 0.03	**0.00**	0.01	0.10 ± 0.03	14.19 ± 0.02



(a) Azimuth frames (388 × 516 pixels) from the NPP test set

(b) Azimuth frames (600 × 700 pixels) from the MUC data for domain shift analysis

Figure 2: **MMP benchmark comparison after Lu-Chipman decomposition ($\mathbf{A}_i \in \mathbb{R}^{3\times3}$).** The images show the per-pixel azimuth φ_i by the various models with Nvidia RTX 4090 GPU runtimes below. (a) shows an example from the NPP test set (more in Supp. Fig. 7); (b) shows a larger frame from the MUC data, using the same models to assess adaptation across instrument domain gaps.

across image domains, while LieNN best balances accuracy and orthogonality, halving runtime for MMP and achieving 5× (MNIST) to 20× (ULM) speed-ups at negligible overhead relative to UNN.

Results for MMP ($\mathbf{A}_i \in \mathbb{R}^{3\times3}$): Neural components learn to approximate the algebraic operations of the Lu–Chipman decomposition Lu & Chipman (1996), suggesting the potential to replace

Table 2: **MUC data ($\mathbf{A}_i \in \mathbb{R}^{3\times3}$) for domain gap analysis.** One frame results marked as in Table 1.

Method	RNN	$\overline{\mathrm{SSIM}(\mathbf{M})}\uparrow$	$\overline{\|\mathbf{X}\|_F^{\mathrm{ort}}}\downarrow$	$\overline{\|\mathbf{D}\|_F^{\mathrm{rel}}}\downarrow$	$\overline{\|\mathbf{Y}\|_F^{\mathrm{ort}}}\downarrow$	Time [ms] $\downarrow$
Exact SVD	✗	1.00	0.00	0.00	0.00	98.80
rSVD	✗	0.10	**0.00**	**0.00**	**0.00**	4.33e4
SV-Learn	✗	0.41	–	0.55	–	3.14
E2E	✗	0.50	–	–	–	**1.21**
E2E	✓	**0.89**	–	–	–	10.29
UNN	✓	0.68	0.38	0.11	0.16	28.37
LieNN	✓	0.83	**0.00**	0.07	0.25	28.96

symbolic computations with accelerated abstractions. End-to-end modules most effectively cover these decomposition and filter steps, explaining their speed advantage over the non-iterative analytic method. However, these strategies do not produce explicit matrix factors, limiting their utility in applications requiring spectral analysis. Notably, rSVD attains near-exact factor errors yet the lowest SSIM, as near-degenerate Mueller spectra render the SVD basis non-unique, so an exact but differently rotated factorisation still misreconstructs $\mathbf{M}$. Figure 2 shows the per-pixel $\varphi_i \in \mathbb{R}$ azimuth images for the two polarimetric domains. The analytic model approximates GT well despite appearing noisy, showing a deterministic first-order SVD approximation can be competitive under tight budgets. Among learnable models without RNNs, the Lie-informed network performs best, though small local errors persist, and the end-to-end model shows a consistent offset. With RNNs, all learnable models closely track the Exact SVD, with only minor deviations in the UNN. On a different MMP instrument (Fig. 2b), the E2E model deviates most without RNNs, whereas activating the RNN best reproduces the Exact SVD patterns, supporting the trends in Tables 1 and 2. As MMP-trained model variants predict per-pixel components without spatial context, they transfer robustly across domains. Table 2 reports out-of-distribution results from a separate MMP instrument with a different optical setup Novikova et al. (2024); Pogudin (2024). This domain gap analysis indicates consistent relative performance, although all learned approaches sacrifice absolute accuracy.

Table 3: **MNIST test set ($\mathbf{A}_i \in \mathbb{R}^{280\times280}$) for denoising.** Mean $\pm$ std across frames (–: not applicable; std omitted when 0 or mean $> 10^1$). **Bold** best, shaded second-best non-GT score.

Method	RNN	k	$\overline{\mathrm{PSNR}}$ [dB]$\uparrow$	$\overline{\|\mathbf{X}\|_F^{\mathrm{ort}}}\downarrow$	$\overline{\|\mathbf{D}\|_F^{\mathrm{rel}}}\downarrow$	$\overline{\|\mathbf{Y}\|_F^{\mathrm{ort}}}\downarrow$	Time [ms]$\downarrow$
Noisy input	✗	–	15.40 ± 0.05	–	–	–	0.02
Exact SVD	✗	38	17.65 ± 0.08	**0.00**	**0.00**	**0.00**	64.23 ± 1.20
rSVD	✗	38	17.65 ± 0.07	0.87	0.33	0.87	19.35 ± 0.34
SV-Learn	✗	1	8.07 ± 0.14	–	0.11	–	$\mathbf{0.67 \pm 0.03}$
E2E	✗	–	12.85 ± 0.13	–	–	–	1.28 ± 0.09
Analytic	✗	280	2.97 ± 0.33	3.57×10^8	3.43×10^4	1.43×10^1	3.17 ± 0.07
UNN	✗	4	11.78 ± 0.24	0.94	0.07 ± 0.03	1.15×10^1	3.30 ± 0.08
LieNN	✗	96	17.78 ± 0.10	0.21 ± 0.02	0.18 ± 0.01	1.30 ± 0.03	6.59 ± 0.13
E2E	✓	–	12.55 ± 0.15	–	–	–	5.41 ± 0.05
UNN	✓	2	12.08 ± 0.14	0.96	0.02 ± 0.01	1.14×10^1	7.33 ± 0.05
LieNN	✓	97	$\mathbf{17.90 \pm 0.07}$	0.15	0.20	1.26 ± 0.02	10.33 ± 0.15

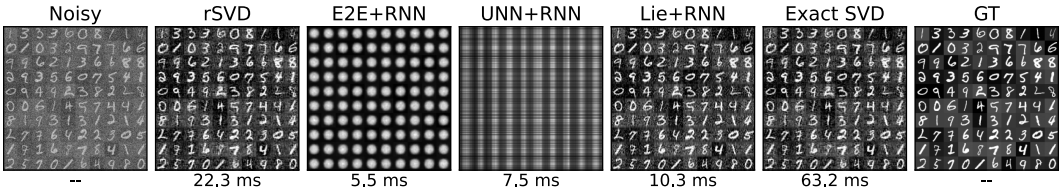


Figure 3: **MNIST patchwork denoising comparison** ($\mathbf{A}_i \in \mathbb{R}^{280\times280}$; noise $\sigma = 0.2$). Columns depict reconstructions by different methods with low-cost Nvidia T500 GPU runtimes shown below.

Results for MNIST denoising ($\mathbf{A}_i \in \mathbb{R}^{280\times280}$): Table 3 reports quantitative results for the MNIST patchwork denoising benchmark. LieNN with RNN achieves the highest PSNR (17.90 dB), marginally exceeding classical SVD truncation (17.65 dB, 0.51) at roughly $6\times$ the speed. Without RNN, LieNN matches SVD while being nearly $10\times$ faster. This denoising advantage over exact SVD is a by-product of imperfect fitting: the residual approximation error acts as implicit regularisation that suppresses noise components the exact SVD would retain. LieNN's optimal truncation rank ($k = 96$–97) is substantially higher than SVD's ($k = 38$), suggesting the learned basis concentrates signal energy differently. SV-Learn and E2E without RNN perform poorly and the analytic variant fails at 280×280 (PSNR 2.97 dB). A PSNR–rank sweep (Supp. Fig. 8) confirms that LieNN's advantage is robust across a wide range of k values.

Results for ULM using the PALA data ($\mathbf{A}_i \in \mathbb{R}^{500\times500}$): Table 4 reports results for ULM spatio-temporal decomposition. The LieNN with RNN achieves the highest SSIM, closely approximating the Exact SVD with low left factor error, confirming that Lie constraints yield physically meaningful decompositions even in highly dynamic ultrasound data, where the larger, quadratically-scaled $\mathbf{Y}_f$ and the heavy-tailed Casorati spectrum inflate $\overline{\|\mathbf{D}\|_F^{\text{rel}}}$ and $\overline{\|\mathbf{Y}\|_F^{\text{ort}}}$ relative to MMP without degrading $\overline{\text{SSIM}(\mathbf{M})}$. Unlike in MMP, the E2E model performs poorly, indicating ULM needs task structure.

Table 4: **ULM results for spatio-temporal filtering ($\mathbf{A}_i \in \mathbb{R}^{500\times500}$).** Mean ± std across frames (–: not applicable; std omitted when 0). **Bold** best, shaded second-best non-Exact-SVD score.

Method	RNN	$\overline{\text{SSIM}(\mathbf{M})}\uparrow$	$\overline{\|\mathbf{X}\|_F^{\text{ort}}}\downarrow$	$\overline{\|\mathbf{D}\|_F^{\text{rel}}}\downarrow$	$\overline{\|\mathbf{Y}\|_F^{\text{ort}}}\downarrow$	Time [ms] ↓
Exact SVD	✗	1.00	0.00	0.00	0.00	75.16 ± 0.96
rSVD	✗	0.72 ± 0.02	0.99	**0.00**	**0.99**	1.98 ± 0.02
Analytic	✗	0.67 ± 0.03	**0.00**	1.38	22.33 ± 0.25	**1.23 ± 0.01**
E2E	✓	0.71 ± 0.03	–	–	–	2.21
UNN	✓	0.77 ± 0.03	0.84	0.93	22.24 ± 0.04	2.49 ± 0.01
LieNN	✓	**0.89 ± 0.03**	**0.00**	0.97	20.80 ± 1.25	3.28 ± 0.04

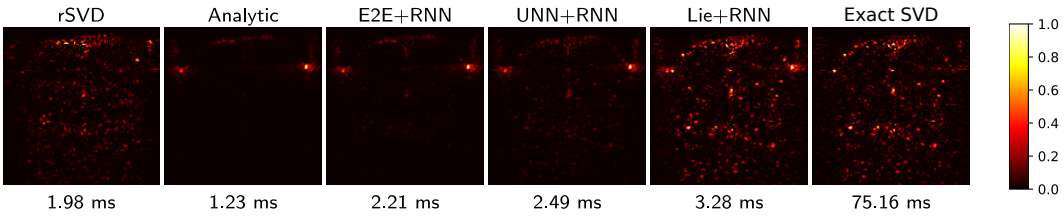


Figure 4: **ULM clutter removal ($\mathbf{A}_i \in \mathbb{R}^{500\times500}$).** Images show the B-mode intensity after spatio-temporal filtering (colour-encoded for better visibility). Columns represent methods and bottom numbers the per-stack Nvidia H100 GPU computation time.

Qualitative inspection of a ULM filtered B-mode frame in Fig. 4 shows that LieNN produces the closest match to the Exact SVD, retaining microbubble contrast and effective clutter suppression while operating at least 10-times faster. This suggests that online ULM is possible without compromising fidelity. Supp. Fig. 6 details how this advantage grows with matrix and batch size.

5 Conclusion

This work introduced Quasi-SVD, a Lie-informed differentiable approximation of SVD designed for massive GPU parallelism. By enforcing orthogonality analytically via skew-symmetric parametrization and the matrix exponential while relaxing singular-value exactness, it achieves guaranteed structural constraints at real-time speed. Across polarimetric, ultrasound, and MNIST denoising benchmarks Lie-constrained models exceed non-constrained baselines in reconstruction fidelity while substantially reducing computation cost, and the pixel-wise formulation yields strong domain transfer. Future extensions include additional constraints for improved singular-value and left-factor fidelity, and complex-valued implementations.

AI USE STATEMENT

In this work, we used generative AI tools for providing critical ingredients proving mathematical claims, assist in the writing of proofs, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods/models, assist with translation, clean and reformat dataset. We have reviewed all AI-assisted work. In particular, LLM-generated proofs and citation references have been manually double-checked for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

For fair use and reproducibility, the code developed and used in this study will be released as an online repository upon paper publication. We note that the MUC (MMP), PALA (ULM), and MNIST datasets used here are already publicly available while the NPP (MMP) dataset is currently under consideration of publication elsewhere and is expected to become accessible soon.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11): 4311–4322, 2006.

Jérôme Baranger, Bastien Arnal, Fabienne Perren, Olivier Baud, Mickael Tanter, and Charlie Demené. Adaptive spatiotemporal svd clutter filtering for ultrafast doppler imaging using similarity of spatial singular vectors. *IEEE transactions on medical imaging*, 37(7):1574–1586, 2018.

Guy E Blelloch. Programming parallel algorithms. *Communications of the ACM*, 39(3):85–97, 1996.

Yoeri E. Boink and Christoph Brune. Learned SVD: solving inverse problems via hybrid autoencoding. *CoRR*, abs/1912.10840, 2019. URL http://arxiv.org/abs/1912.10840.

Matthew Brand. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications*, 415(1):20–30, 2006. ISSN 0024-3795. doi: https://doi.org/10.1016/j.laa.2005.07.021. URL https://www.sciencedirect.com/science/article/pii/S0024379505003812. Special Issue on Large Scale Linear and Nonlinear Eigenvalue Problems.

Richard P Brent. The parallel evaluation of general arithmetic expressions. *Journal of the ACM (JACM)*, 21(2):201–206, 1974.

Tony F Chan. Rank revealing qr factorizations. *Linear Algebra and its Applications*, 88:67–82, 1987.

Elizabeth Cole, Qingxi Meng, John Pauly, and Shreyas Vasanawala. Learned compression of high dimensional image datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1748–1752, 2022.

Charlie Demené, Thomas Deffieux, Mathieu Pernot, Bruno-Félix Osmanski, Valérie Biran, Jean-Luc Gennisson, Lim-Anna Sieu, Antoine Bergel, Stéphanie Franqui, Jean-Michel Correas, Ivan Cohen, Olivier Baud, and Mickael Tanter. Spatiotemporal clutter filtering of ultrafast ultrasound data highly increases doppler and fultrasound sensitivity. *IEEE Transactions on Medical Imaging*, 34(11):2271–2285, 2015. doi: 10.1109/TMI.2015.2428634.

James Demmel and Krešimir Veselić. Jacobi's method is more accurate than qr. *SIAM journal on matrix analysis and applications*, 13(4):1204–1245, 1992.

Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, 36(1):158–183, 2006.

Claudia Errico, Juliette Pierre, Sophie Pezet, Yann Desailly, Zsolt Lenkei, Olivier Couture, and Mickael Tanter. Ultrafast ultrasound localization microscopy for deep super-resolution vascular imaging. *Nature*, 527(7579):499–502, 2015.

Cédric Foucault and Florent Meyniel. Gated recurrence enables simple and accurate sequence prediction in stochastic, changing, and structured environments. *eLife*, 10:e71801, dec 2021. ISSN 2050-084X. doi: 10.7554/eLife.71801. URL https://doi.org/10.7554/eLife.71801.

G. H. Golub and C. Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420, 1970. ISSN 0945-3245. doi: 10.1007/BF02163027. URL https://doi.org/10.1007/BF02163027.

Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

Ming Gu and Stanley C Eisenstat. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing*, 17(4):848–869, 1996.

Christopher Hahne, Georges Chabouh, Arthur Chavignon, Olivier Couture, and Raphael Sznitman. Rf-ulm: Ultrasound localization microscopy learned from radio-frequency wavefronts. *IEEE Transactions on Medical Imaging*, 43(9):3253–3262, 2024. doi: 10.1109/TMI.2024.3391297.

Christopher Hahne, Ivan Diaz, Omar Rodriguez-Nunez, Elea Gros, Muriel Blatter, Theotim Lucas, David Hasler, Tatiana Novikova, Theoni Maragkou, Philippe Schucht, and Richard McKinley. Polarimetric feature analysis of mueller matrices for brain tumor image segmentation. *Opt. Express*, 33(20):43379–43392, Oct 2025a. doi: 10.1364/OE.561518. URL https://opg.optica.org/oe/abstract.cfm?URI=oe-33-20-43379.

Christopher Hahne, Omar Rodriguez-Nunez, Elea Gros, Théotim Lucas, Ekkehard Hewer, Tatiana Novikova, Theoni Maragkou, Philippe Schucht, and Richard McKinley. Physically consistent image augmentation for deep learning in mueller matrix polarimetry. *IEEE Transactions on Image Processing*, 34:6953–6962, 2025b. doi: 10.1109/TIP.2025.3618390.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53 (2):217–288, 2011.

Per Christian Hansen. The truncated svd as a method for regularization. *BIT Numerical Mathematics*, 27:534–553, 1987.

Per Christian Hansen. Truncated singular value decomposition solutions to discrete ill-posed problems with ill-determined numerical rank. *SIAM Journal on Scientific and Statistical Computing*, 11(3):503–518, 1990.

Zeinab A. Hassaan, Mohammed H. Yacoub, and Lobna A. Said. Gated recurrent unit accelerators for financial time series prediction on field-programmable gate array. *Engineering Applications of Artificial Intelligence*, 162:112534, 2025. ISSN 0952-1976. doi: https://doi.org/10.1016/j.engappai.2025.112534. URL https://www.sciencedirect.com/science/article/pii/S0952197625025655.

Michel Hayoz, Christopher Hahne, Mathias Gallardo, Daniel Candinas, Thomas Kurmann, Maximilian Allan, and Raphael Sznitman. Learning how to robustly estimate camera pose in endoscopic videos. *International journal of computer assisted radiology and surgery*, 18(7):1185–1192, 2023.

Baptiste Heiles, Arthur Chavignon, Vincent Hingot, Pauline Lopez, Eliott Teston, and Olivier Couture. Performance benchmarking of microbubble-localization algorithms for ultrasound localization microscopy. *Nature Biomedical Engineering*, 6(5):605–616, 2022.

Sheetal Lahabar and PJ Narayanan. Singular value decomposition on gpu using cuda. In *2009 IEEE international symposium on parallel & distributed processing*, pp. 1–10. IEEE, 2009.

Mario Lezcano-Casado and David Martınez-Rubio. Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group. In *International Conference on Machine Learning*, pp. 3794–3803. PMLR, 2019.

Jun Li, Fuxin Li, and Sinisa Todorovic. Efficient riemannian optimization on the stiefel manifold via the cayley transform. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJxV-ANKDH.

Shih-Yau Lu and Russell A Chipman. Interpretation of mueller matrices based on polar decomposition. *JOSA A*, 13(5):1106–1113, 1996.

Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18039–18048, 2024.

Stefano Moriconi, Omar Rodríguez-Núñez, Éléa Gros, Leonard A Felger, Theoni Maragkou, Ekkehard Hewer, Angelo Pierangelo, Tatiana Novikova, Philippe Schucht, and Richard McKinley. Near-real-time mueller polarimetric image processing for neurosurgical intervention. *International journal of computer assisted radiology and surgery*, 19(6):1033–1043, 2024.

Cameron Musco and Christopher Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. *Advances in neural information processing systems*, 28, 2015.

Cameron Musco and Christopher Musco. The random Nyström method. *Foundations and Trends in Machine Learning*, 10(6):453–566, 2017.

Vedran Novakovic. A hierarchically blocked jacobi svd algorithm for single and multiple graphics processing units. *SIAM journal on scientific computing*, 37(1):C1–C30, 2015.

Tatiana Novikova, Alexey Ovchinnikov, Gleb Pogudin, and Jessica C Ramella-Roman. Time-efficient filtering of imaging polarimetric data by checking physical realizability of experimental mueller matrices. *Bioinformatics*, 40(7):btae348, 06 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae348. URL https://doi.org/10.1093/bioinformatics/btae348.

NVIDIA Corporation. *cuSOLVER Library*, 2026. URL https://docs.nvidia.com/cuda/cusolver/. Version 13.1.

Gleb Pogudin. Github repository. https://github.com/pogudingleb/mueller_matrices, 2024.

Jongha Jon Ryu, Xiangxiang Xu, Hasan Sabri Melihcan Erol, Yuheng Bu, Lizhong Zheng, and Gregory W. Wornell. Operator SVD with neural networks via nested low-rank approximation. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42870–42905. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ryu24b.html.

Paulo Sampaio, Maria Lopez-Antuña, Federico Storni, Jonatan Wicht, Greta Sökeland, Martin Wartenberg, Pablo Márquez-Neila, Daniel Candinas, Brice-Olivier Demory, Aurel Perren, et al. Müller matrix polarimetry for pancreatic tissue characterization. *Sci. Rep.*, 13(1):16417, 2023.

Meyer Scetbon, Michael Elad, and Peyman Milanfar. Deep k-svd denoising. *IEEE Transactions on Image Processing*, 30:5944–5955, 2021.

Martin Stoll. A krylov–schur approach to the truncated svd. *Linear Algebra and its Applications*, 436(8):2795–2806, 2012. ISSN 0024-3795. doi: https://doi.org/10.1016/j.laa.2011.07.022. URL https://www.sciencedirect.com/science/article/pii/S0024379511005349. Special Issue dedicated to Danny Sorensen's 65th birthday.

Zachary Teed and Jia Deng. Tangent space backpropagation for 3d transformation groups. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10333–10342. IEEE, 2021.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. SVD-LLM: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=LNYIUouhdt.

Yutong Wang, Haiyu Wang, and Sai Qian Zhang. Qsvd: Efficient low-rank approximation for unified query-key-value weight compression in low-precision vision-language models. volume 38, pp. 1789–1820, 2026.

Thomas Whelan, Stefan Leutenegger, Renato Salas-Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics: science and systems*, volume 11, 2015.

Bo Xin, Yuan Tian, Yizhou Wang, and Wen Gao. Background subtraction via generalized fused lasso foreground modeling. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4676–4684, 2015. doi: 10.1109/CVPR.2015.7299099.

Derek Xu, William Shiao, Jia Chen, and Evangelos E Papalexakis. Sv-learn: Learning matrix singular values with neural networks. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 232–237. IEEE, 2022.

Xu Yang, Cheng Deng, Feng Zheng, Junchi Yan, and Wei Liu. Deep spectral clustering using dual autoencoder network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4066–4075, 2019.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. Asvd: Activation-aware singular value decomposition for compressing large language models, 2023.

Anru Zhang and Dong Xia. Tensor svd: Statistical and computational limits. *IEEE Transactions on Information Theory*, 64(11):7311–7338, 2018.

Łukasz Struski, Paweł Morkisz, Przemysław Spurek, Samuel Rodriguez Bernabeu, and Tomasz Trzciński. Efficient gpu implementation of randomized svd and its applications. *Expert Systems with Applications*, 248:123462, 2024. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2024.123462. URL https://www.sciencedirect.com/science/article/pii/S0957417424003270.

A MATHEMATICAL FOUNDATIONS

A.1 ORTHOGONALITY VIA MATRIX EXPONENTIAL

Theorem (Skew-Symmetric Exponential is Orthogonal) Let $\mathbf{S} \in \mathfrak{so}(m)$ be a real skew-symmetric matrix (i.e., $\mathbf{S}^\top = -\mathbf{S}$). Then the matrix exponential

$$\exp(\mathbf{S}) = \sum_{k=0}^{\infty} \frac{\mathbf{S}^k}{k!}$$

is an orthogonal matrix, meaning that

$$\exp(\mathbf{S})^\top \exp(\mathbf{S}) = \mathbf{I}.$$

Proof As $\mathbf{S}$ is skew-symmetric, we have $\mathbf{S}^\top = -\mathbf{S}$. Consider the matrix exponential:

$$\exp(\mathbf{S}) = \mathbf{I} + \mathbf{S} + \frac{\mathbf{S}^2}{2!} + \frac{\mathbf{S}^3}{3!} + \cdots.$$

Taking the transpose, and using the linearity of the transpose along with the fact that $(\mathbf{S}^k)^\top = (\mathbf{S}^\top)^k$, we obtain

$$\exp(\mathbf{S})^\top = \mathbf{I} + \mathbf{S}^\top + \frac{(\mathbf{S}^\top)^2}{2!} + \frac{(\mathbf{S}^\top)^3}{3!} + \cdots.$$

Substitute $\mathbf{S}^\top = -\mathbf{S}$:

$$\exp(\mathbf{S})^\top = \mathbf{I} - \mathbf{S} + \frac{\mathbf{S}^2}{2!} - \frac{\mathbf{S}^3}{3!} + \cdots = \exp(-\mathbf{S}).$$

Now, consider the product:

$$\exp(\mathbf{S})^\top \exp(\mathbf{S}) = \exp(-\mathbf{S}) \exp(\mathbf{S}).$$

Using the property of the exponential function on commuting matrices,

$$\exp(-\mathbf{S}) \exp(\mathbf{S}) = \exp(-\mathbf{S} + \mathbf{S}) = \exp(\mathbf{0}) = \mathbf{I}.$$

Thus, $\exp(\mathbf{S})^\top \exp(\mathbf{S}) = \mathbf{I}$, which shows that $\exp(\mathbf{S})$ is orthogonal.

Corollary This guarantees that $\mathbf{X} = \exp(\mathbf{S}) \in \mathrm{SO}(m)$ for any skew-symmetric $\mathbf{S} \in \mathfrak{so}(m)$.

A.2 SPECIAL ORTHOGONAL GROUP

The matrix exponential of a skew-symmetric matrix produces an element of the special orthogonal group,

$$\exp(\mathbf{S}) \in \mathrm{SO}(m) \subset \mathrm{O}(m),$$

which enforces the constraint $\det(\mathbf{X}) = 1$. In contrast, the full orthogonal group also contains matrices with negative determinant, i.e. $\det(\mathbf{X}') \in \{1, -1\}$ for $\mathbf{X}' \in \mathrm{O}(m)$.

In some applications, such as matching the sign structure of a target orthogonal factor, it is necessary to obtain an orthogonal matrix with either determinant. To allow this flexibility while still parameterising $\mathbf{X}$ through a skew-symmetric generator, we introduce a diagonal by

$$\mathbf{B} = \mathrm{diag}(\mathbf{b}), \qquad \mathbf{b} \in \{-1, 1\}^m,$$

which represents an element of $\mathrm{O}(m)$. The corrected orthogonal matrix is then computed as

$$\mathbf{X}' = \mathbf{B}\mathbf{X}.$$

where left-multiplication by $\mathbf{B}$ flips selected column directions of $\mathbf{X}$ and adjusts the determinant when required, while preserving orthogonality:

$$(\mathbf{X}')^\top \mathbf{X}' = \mathbf{X}^\top \mathbf{B}^2 \mathbf{X} = \mathbf{I}.$$

We evaluated the reflection correction during ablations and found that it neither improved approximation quality nor altered the stability of training. Because its effect was neutral and the method functions identically without it, we report the mechanism only in the supplementary to avoid diverting attention from components that materially influence performance. While this correction is theoretically available, we demonstrate hereafter that restricting to $\mathrm{SO}(m)$ is sufficient for valid decomposition.

A.3 Why Asymmetric Constraints Suffice

Lemma (Existence of a Decomposition with $\mathbf{X} \in \mathrm{SO}(m)$) Consider a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $\mathbf{A} = \mathbf{X}\mathbf{D}\mathbf{Y}^\top$, where $\mathbf{X}, \mathbf{Y}$ contain the left and right singular vectors of $\mathbf{A}$ and $\mathbf{D} = \mathrm{diag}(\mathbf{d})$ is the (possibly rectangular) diagonal matrix containing the singular values, such that for the column vectors $\mathbf{x}_k, \mathbf{y}_k$ holds

$$\mathbf{A}\mathbf{A}^\top \mathbf{x}_k = \sigma_k^2 \mathbf{x}_k, \quad \mathbf{A}^\top \mathbf{A}\mathbf{y}_k = \sigma_k^2 \mathbf{y}_k.$$

Now, if $\mathbf{X} \in \mathrm{O}(m)$ with $\det(\mathbf{X}) = -1$, let $\mathbf{B}_X = \mathrm{diag}(-1, 1, \ldots, 1)$ and $\mathbf{B}_Y = \mathrm{diag}(-1, 1, \ldots, 1)$ denote sign matrices that flip the same corresponding singular-vector pair, and define

$$\tilde{\mathbf{X}} = \mathbf{X}\mathbf{B}_X, \qquad \tilde{\mathbf{Y}} = \mathbf{Y}\mathbf{B}_Y.$$

These $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ remain orthogonal because they are obtained by right multiplication with orthogonal sign matrices. For the corresponding sign flip, the diagonal structure of $\mathbf{D}$ gives

$$\mathbf{B}_X \mathbf{D} \mathbf{B}_Y^\top = \mathbf{D},$$

and hence

$$\begin{aligned}\tilde{\mathbf{X}}\mathbf{D}\tilde{\mathbf{Y}}^\top &= \mathbf{X}\mathbf{B}_X\mathbf{D}\mathbf{B}_Y^\top\mathbf{Y}^\top \\ &= \mathbf{X}\mathbf{D}\mathbf{Y}^\top \\ &= \mathbf{A}.\end{aligned}$$

The matrices $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ still contain the singular vectors corresponding to the same singular values, since right multiplication by the sign matrices only changes the signs of their columns. In particular, if $\mathbf{x}_k$ and $\mathbf{y}_k$ correspond to σ_k, then replacing them by $-\mathbf{x}_k$ and $-\mathbf{y}_k$ preserves their singular-vector relations and leaves σ_k unchanged.

Moreover, $\tilde{\mathbf{X}} \in \mathrm{SO}(m)$:

$$\begin{aligned}\det(\tilde{\mathbf{X}}) &= \det(\mathbf{X})\det(\mathbf{B}_X) \\ &= (-1)(-1) = 1.\end{aligned}$$

Note that this sign flexibility (discussed in A.2) is a theoretical convenience for the proof; the practical method does not require explicit sign matching.

This lemma motivates the asymmetric design: an equivalent SVD representation always exists with $\mathbf{X} \in \mathrm{SO}(m)$, so enforcing orthogonality on $\mathbf{X}$ via the exponential map does not exclude any valid SVD representation. In contrast, the corresponding right factor may lie in the determinant (-1) component of $\mathrm{O}(n)$ and is therefore not constrained to $\mathrm{SO}(n)$. In the proposed method, $\mathbf{Y}$ is recovered residually and encouraged toward orthogonality through the soft constraint $\mathcal{L}_{\text{ort}}$.

A.4 Approximation Bounds

A.4.1 Gershgorin Bounds on Singular Values

Gershgorin Circle Theorem Let $\mathbf{A} \in \mathbb{R}^{p\times p}$ with entries denoted as a_{ij}. Then, the eigenvalues are contained in the circles whose radii are bounded by the sums of the matrix entries:

$$\lambda \in \{z \mid |z - a_{ii}| < R_i\} \quad \text{where} \quad R_i = \sum_{\substack{j=1 \\ j\neq i}}^{p} |a_{ij}|.$$

Bounds on Singular Values via Coherence Consider the symmetric squared matrix $\mathbf{B} := \mathbf{A}^\top\mathbf{A} \in \mathbb{R}^{n\times n}$ such that $b_{ik} = \sum_{j=1}^{n} a_{ji}a_{jk} = \langle \mathbf{a}_{\cdot i}, \mathbf{a}_{\cdot k}\rangle$, where $\mathbf{a}_{\cdot i}, \mathbf{a}_{\cdot k}$ are the i-th and k-th columns of $\mathbf{A}$. Applying the Gershgorin Circle Theorem to $\mathbf{B}$, the eigenvalues of $\mathbf{B}$ (i.e., the squared singular values of $\mathbf{A}$) satisfy

$$\lambda \in \left\{z \mid |z - \|\mathbf{a}_{\cdot i}\|^2| < S_i\right\}, \qquad S_i := \sqrt{n-1}\cdot \mu(\mathbf{A})\|\mathbf{A}\|_F\|\mathbf{a}_{\cdot i}\|,$$

where the coherence of $\mathbf{A}$ is given by

$$\mu(\mathbf{A}) = \max_{i\neq k} \frac{|\langle \mathbf{a}_{\cdot i}, \mathbf{a}_{\cdot k}\rangle|}{\|\mathbf{a}_{\cdot i}\|\|\mathbf{a}_{\cdot k}\|}$$

.

Proof By the Gershgorin Circle Theorem applied to $\mathbf{B}$, the Gershgorin radius for row i is

$$R_i^{\mathbf{B}} := \sum_{\substack{k=1 \\ k \neq i}}^{n} |b_{ik}|.$$

The diagonal elements satisfy

$$|b_{ii}| = \left| \sum_{j=1}^{n} a_{ji}^2 \right| = \|\mathbf{a}_{\cdot i}\|^2,$$

confirming the center of the i-th Gershgorin disc. For the off-diagonal radius, using the coherence bound $|\langle \mathbf{a}_{\cdot i}, \mathbf{a}_{\cdot k} \rangle| \leq \mu(\mathbf{A}) \|\mathbf{a}_{\cdot i}\| \|\mathbf{a}_{\cdot k}\|$:

$$\begin{aligned} R_i^{\mathbf{B}} &= \sum_{\substack{k=1 \\ k \neq i}}^{n} |b_{ik}| = \sum_{\substack{k=1 \\ k \neq i}}^{n} |\langle \mathbf{a}_{\cdot i}, \mathbf{a}_{\cdot k} \rangle| \\ &\leq \mu(\mathbf{A}) \|\mathbf{a}_{\cdot i}\| \sum_{\substack{k=1 \\ k \neq i}}^{n} \|\mathbf{a}_{\cdot k}\| \leq \mu(\mathbf{A}) \|\mathbf{a}_{\cdot i}\| \sqrt{n-1} \|\mathbf{A}\|_F = S_i, \end{aligned}$$

where the final step applies Cauchy–Schwarz,

$$\sum_{k=1}^{n} \|\mathbf{a}_{\cdot k}\| \leq \sqrt{n} \|\mathbf{A}\|_F,$$

restricted to the $n-1$ terms $k \neq i$. Thus $R_i^{\mathbf{B}} \leq S_i$, establishing S_i as a valid Gershgorin radius bound.

B Work–Span Complexity Analysis and Proofs

B.1 Computational complexity

The following analysis considers Quasi-SVD under the work–span model Blelloch (1996) (EREW PRAM) for $m \leq n$, highlighting its suitability for SIMT (Single Instruction, Multiple Threads) architectures. Unlike classical solvers limited by sequential steps, Quasi-SVD exploits a parallel computation graph comprising multiple stages: MLP of width h, skew-symmetric scattering, matrix exponentiation, and a one-sided orthogonal decomposition. Table 5 summarises the comparison with classical baselines. While the Cayley-transform's LU solve dictates the span for general cases,

Table 5: Work–span complexity Blelloch (1996). Full proofs are found in Supp. B.2.

Method	**Work**	**Span**
Golub–Reinsch Golub & Reinsch (1970)	$\mathcal{O}(mn^2)$	$\mathcal{O}(m \log m)$
Truncated SVD (via bidiagonalisation) Hansen (1987)	$\mathcal{O}(mnK)$	$\mathcal{O}(K \log n)$
Randomised SVD (rSVD) Halko et al. (2011)	$\mathcal{O}(mn\ell + n\ell^2)$	$\mathcal{O}(\ell \log n)$
Lanczos (rank K) Golub & Van Loan (2013)	$\mathcal{O}(mnK)$	$\mathcal{O}(K \log n)$
Subspace (block) iteration Musco & Musco (2015)	$\mathcal{O}(mnK)$	$\mathcal{O}(K \log n)$
Quasi-SVD: LieNN by Cayley	$\mathcal{O}(m^3)$	$\mathcal{O}(m \log m)$
Quasi-SVD: LieNN by Rodrigues, $m = n = 3$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Quasi-SVD: LieNN by Taylor	$\mathcal{O}(m^3)$	$\mathcal{O}(\log m)$

employing a constant-degree Taylor expansion or the closed-form Rodrigues formula ($m = n = 3$) reduces the span to $\mathcal{O}(\log m)$. By default, a 9-term Taylor expansion is employed for general matrices, whereas the Rodrigues formula is used for $\mathbf{A} \in \mathbb{R}^{3\times 3}$.

B.2 Complexity Proofs for Lie-based SVD

This section analyses Quasi-SVD under the work–span model and provides formal derivations for all claims in Table 5. Throughout, we write $n = \min(m, n)$, corresponding to the implicit rank used in the main paper's complexity table. We adopt the EREW PRAM model Blelloch (1996): processors may not simultaneously read or write the same memory cell. All reductions over p elements use balanced binary fan-in trees.

Notation and Model Assumptions

- $\mathbf{A} \in \mathbb{R}^{m\times n}$, $m \geq n$ (enforced by transpose branch), $n = \min(m, n)$.
- MLP hidden width $h = \mathcal{O}(m)$.
- *Work* W: total arithmetic operations.
- *Span* S: depth of the critical path in the DAG of operations.
- Matrix multiply $\mathbb{R}^{a\times b} \cdot \mathbb{R}^{b\times c}$: work $\mathcal{O}(abc)$, span $\mathcal{O}(\log b)$ (parallel dot products, each a reduction tree).
- By **Brent's theorem** Brent (1974), wall-clock time on P processors satisfies $T_P = \mathcal{O}(W/P + S)$.

  Batch dimension note. The work–span bounds in Table 5 and Propositions 2–9 are defined per single matrix instance $\mathbf{A} \in \mathbb{R}^{m\times n}$. Across a batch of size B, SIMT execution scales total work linearly as $\mathcal{O}(B \cdot W)$ while ideal span remains S due to parallel evaluation across threads. Empirical scaling over batch dimensions is evaluated in Figure 6b.

Stage 1: Pre-processing MLP

Proposition 2 (Pre-processing work and span)**.** *A two-layer MLP $f : \mathbb{R}^{mn} \to \mathbb{R}^{mn}$ with hidden width h satisfies*

$$W_{\text{pre}} = \mathcal{O}(mnh), \qquad S_{\text{pre}} = \mathcal{O}(\log(mn)).$$

Proof. The MLP computes $f(x) = W_2\,\sigma(W_1 x + b_1) + b_2$ with $W_1 \in \mathbb{R}^{h\times mn}$ and $W_2 \in \mathbb{R}^{mn\times h}$.

Layer 1. Each of the h output neurons computes an inner product over mn inputs. With unlimited processors, all h inner products run in parallel; each inner product is a reduction tree of depth $\lceil \log_2(mn) \rceil$. Hence $W_1 = \mathcal{O}(h \cdot mn)$, $S_1 = \mathcal{O}(\log(mn))$.

Activation. Elementwise σ on h values: $W = \mathcal{O}(h)$, $S = \mathcal{O}(1)$.

Layer 2. Each of the mn outputs is an inner product over h values: $W_2 = \mathcal{O}(mn \cdot h)$, $S_2 = \mathcal{O}(\log h)$.

Summing, and noting $\log h \leq \log(mn)$: $W_{\text{pre}} = \mathcal{O}(mnh)$, $S_{\text{pre}} = \mathcal{O}(\log(mn))$. □

Stage 2: Reduction MLP

Proposition 3 (Reduction work and span)**.** *The reduction MLP $g : \mathbb{R}^{mn} \to \mathbb{R}^{\binom{m}{2}}$ with hidden width h satisfies*

$$W_{\text{red}} = \mathcal{O}(m^2 h), \qquad S_{\text{red}} = \mathcal{O}(\log m).$$

Proof. Identical argument to Proposition 2. The output dimension is $|\mathcal{T}| = m(m-1)/2$ with $\mathcal{O}(m^2)$. Since $m \geq n$, $m^2 \geq mn$, so the output layer cost $\mathcal{O}(|\mathcal{T}| \cdot h) = \mathcal{O}(m^2 h)$ exceeds the input layer cost $\mathcal{O}(mnh)$ by $W_{\text{red}} = \mathcal{O}(m^2 h)$. □

Stage 3: Skew-Symmetric Scatter

Proposition 4 (Fill skew-symmetric work and span)**.** *Scattering $|\mathcal{T}|$ values into an $m \times m$ skew-symmetric matrix satisfies*

$$W_{\text{skew}} = \mathcal{O}(m^2), \qquad S_{\text{skew}} = \mathcal{O}(1).$$

Proof. Each of the $|\mathcal{T}| = m(m-1)/2$ entries is written to two locations (i,j) and (j,i) with sign flip; there are no dependencies between writes (EREW is satisfied since all target cells are distinct). Total writes: $m^2 - m = \mathcal{O}(m^2)$. All writes execute in one parallel step, so $S_{\text{skew}} = \mathcal{O}(1)$. □

Stage 4: Matrix Exponential

4a. Cayley Transform

Proposition 5 (Cayley transform work and span)**.** *The Cayley map $(I-S)^{-1}(I+S)$ for $S \in \mathbb{R}^{m\times m}$ skew-symmetric satisfies*

$$W_{\text{Cay}} = \mathcal{O}(m^3), \qquad S_{\text{Cay}} = \mathcal{O}(m \log m).$$

Proof. **LU decomposition of $(I-S)$.** Gaussian elimination with partial pivoting proceeds through $m-1$ sequential elimination steps. At step k, pivot search over $m-k$ rows costs span $\mathcal{O}(\log m)$, and row updates for the remaining $m-k$ rows cost span $\mathcal{O}(\log m)$ (parallel scalar-multiply-add across $m-k$ columns). Across $m-1$ steps the span accumulates to $\mathcal{O}(m \log m)$. Total work for LU: $\mathcal{O}(m^3)$.

Triangular solves. Forward and back substitution each require m sequential steps with $\mathcal{O}(\log m)$ span per step: $S = \mathcal{O}(m \log m)$, $W = \mathcal{O}(m^2)$.

Matrix multiply $(I-S)^{-1}(I+S)$**.** $\mathbb{R}^{m\times m} \cdot \mathbb{R}^{m\times m}$: $W = \mathcal{O}(m^3)$, $S = \mathcal{O}(\log m)$.

Combining (LU dominates): $W_{\text{Cay}} = \mathcal{O}(m^3)$, $S_{\text{Cay}} = \mathcal{O}(m \log m)$. □

4b. Rodrigues Formula ($m = n = 3$)

Proposition 6 (Rodrigues work and span)**.** *For $m = n = 3$, the Rodrigues rotation formula satisfies*

$$W_{\text{Rod}} = \mathcal{O}(1), \qquad S_{\text{Rod}} = \mathcal{O}(1).$$

Proof. With $m = 3$, a skew-symmetric S has exactly 3 free parameters $(\theta_1, \theta_2, \theta_3)$. The Rodrigues formula

$$X = I + \frac{\sin\theta}{\theta} S + \frac{1-\cos\theta}{\theta^2} S^2, \quad \theta = \|\mathbf{v}\|_2,\ \mathbf{v} = (\theta_1, \theta_2, \theta_3)^\top,$$

involves a fixed number of scalar operations ($\|\cdot\|$, sin, cos, divisions) and two 3×3 matrix-scalar products, all independent of any variable-size dimension. Hence both work and span are $\mathcal{O}(1)$. □

4c. Taylor Series (degree p)

Proposition 7 (Taylor exponential work and span)**.** *A degree-p Horner-form Taylor approximation $e^S \approx \sum_{k=0}^{p} S^k/k!$ for $S \in \mathbb{R}^{m\times m}$ satisfies*

$$W_{\text{Tay}} = \mathcal{O}(pm^3), \qquad S_{\text{Tay}} = \mathcal{O}(p \log m).$$

For constant $p = \mathcal{O}(1)$ this reduces to $W = \mathcal{O}(m^3)$, $S = \mathcal{O}(\log m)$.

Proof. Horner evaluation $(((\frac{1}{p!}S + \frac{1}{(p-1)!}I)S + \cdots)S + I)$ requires p sequential matrix multiplications, each costing $W = \mathcal{O}(m^3)$ and $S = \mathcal{O}(\log m)$. Sequentiality of the p steps gives total span $p \cdot \mathcal{O}(\log m) = \mathcal{O}(p \log m)$. □

Stage 5: One-Sided Decomposition

Proposition 8 (Decomposition work and span)**.** *Given $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{X} \in \mathbb{R}^{m\times m}$ orthogonal, computing $(\mathbf{X}, \mathbf{D}, \mathbf{Y})$ satisfies*

$$W_{\text{dec}} = \mathcal{O}(m^2 n), \qquad S_{\text{dec}} = \mathcal{O}(\log m).$$

Proof. The routine performs the following operations in order.

(i) $\mathbf{AA}^\top \in \mathbb{R}^{m\times m}$**:** $W = \mathcal{O}(m^2 n)$, $S = \mathcal{O}(\log n)$.

(ii) $(\mathbf{AA}^\top)\mathbf{X}_n \in \mathbb{R}^{m\times n}$**:** $W = \mathcal{O}(m^2 n)$, $S = \mathcal{O}(\log m)$.

(iii) Diagonal extraction: Diagonal extraction over n columns, m rows each: $W = \mathcal{O}(mn)$, $S = \mathcal{O}(\log m)$; n square roots.

(iv) Right factor $\mathbf{Y} = \mathbf{A}^\top \mathbf{X}_n \mathbf{D}^\dagger \in \mathbb{R}^{n\times n}$**:** $\mathbf{A}^\top \mathbf{X}_n \to n \times n$: $W = \mathcal{O}(mn^2)$, $S = \mathcal{O}(\log m)$; then $\mathbf{D}^\dagger$ scaling: $W = \mathcal{O}(n^2)$, $S = \mathcal{O}(1)$.

Since $m \geq n$, $m^2 n \geq mn^2$, so the dominant work term is $\mathcal{O}(m^2 n)$, and the dominant span is $\mathcal{O}(\log m)$. □

END-TO-END THEOREM

Proposition 9 (End-to-end complexity)**.** *Under EREW PRAM with $h = \mathcal{O}(m)$ and $m \geq n$, the full neural Lie-SVD forward pass satisfies:*

$$\begin{aligned}
&\textit{Cayley, default: } W = \mathcal{O}(m^3),\ S = \mathcal{O}(m \log m).\\
&\textit{Rodrigues, } m{=}n{=}3\textit{: } W = \mathcal{O}(1),\ S = \mathcal{O}(1).\\
&\textit{Taylor, degree } p{=}\mathcal{O}(1)\textit{: } W = \mathcal{O}(m^3),\ S = \mathcal{O}(\log m).
\end{aligned}$$

Proof. Sum Propositions 2–8 for each variant, noting that stages execute sequentially so spans add while works add.

Cayley. $W = \mathcal{O}(mn^2 + m^2 n + m^2 + m^3 + m^2 n) = \mathcal{O}(m^3)$. Since $h = \mathcal{O}(m)$, the preprocessing MLP costs $mnh = \mathcal{O}(mn^2)$; the reduction MLP costs $\mathcal{O}(m^2 h) = \mathcal{O}(m^2 n)$; Stage 5 costs $\mathcal{O}(m^2 n)$. Since $m \geq n$, $m^3 \geq m^2 n \geq mn^2$, so the Cayley exponential (Stage 4) dominates: $W = \mathcal{O}(m^3)$. Span: $\mathcal{O}(\log(mn)) + \mathcal{O}(\log m) + \mathcal{O}(1) + \mathcal{O}(m \log m) + \mathcal{O}(\log m) = \mathcal{O}(m \log m)$ (LU dominates).

Rodrigues. With dimensions fixed at $m = n = 3$, all five pipeline stages consume constant work and execution depth, yielding $W = \mathcal{O}(1)$ and $S = \mathcal{O}(1)$.

Taylor ($p = \mathcal{O}(1)$). As in the Cayley case, m^3 dominates $\mathcal{O}(m^2 n)$ from Stage 5. Stage 4 span becomes $\mathcal{O}(\log m)$; Stage 5 span $\mathcal{O}(\log m)$ also contributes but does not exceed it, so $S = \mathcal{O}(\log m)$. Work as in the Cayley case: $\mathcal{O}(m^3)$. □

Corollary 10 (GPU speedup via Brent's theorem)**.** *On a GPU with P SIMD lanes, the wall-clock time satisfies*

$$T_P = \mathcal{O}\left(\frac{W}{P} + S_{\text{total}}\right).$$

The speedup relative to single-threaded Golub–Reinsch ($T_1^{\text{GR}} = \mathcal{O}(mn^2)$, $S^{\text{GR}} = \mathcal{O}(m \log m)$) is

$$\frac{T_1^{\text{GR}}}{T_P} = \mathcal{O}\left(\frac{mn^2}{W/P + S_{\text{total}}}\right)$$

which is meaningful when $P \gg r$, i.e. the GPU is sufficiently wide relative to the matrix rank.

> **Memory-access patterns.** The span bounds above assume unit-cost memory access. In practice, GPU coalesced access requires column-major layout for the matrix multiply in Stage 5 and row-major for the MLP weight matrices. Cache-miss penalties can inflate effective span by a constant factor that depends on matrix tile size relative to shared-memory capacity; this does not affect asymptotic span but is important for constant-factor comparisons.

C ATTEMPTS THAT DID NOT WORK

C.1 TWO-SIDED ORTHOGONALITY CONSTRAINT

The proposed orthogonality constraint can be simultaneously applied to $\mathbf{X}$ and $\mathbf{Y}$. This *two-sided* constraint requires matching orthogonal singular vectors in $\mathbf{X}$ and $\mathbf{Y}$ to satisfy $\mathbf{A} = \mathbf{XDY}^\top$. In

practice, the neural architectures used in this study fail to reconstruct valid combinations when enforcing both $\mathbf{X} \in \mathrm{SO}(m)$ and $\mathbf{Y} \in \mathrm{SO}(n)$ such that predicted components result in $\mathbf{A} \neq \mathbf{X}\mathbf{D}\mathbf{Y}^\top$. Consequently, the inability to reconstruct $\mathbf{A}$ from $\mathbf{X}\mathbf{D}\mathbf{Y}^\top$ does not qualify to be a valid decomposition technique. However, it is worth mentioning that this *two-sided* constraint achieved SSIM scores similar to the *one-sided* equivalent when trained using the $\mathcal{L}_{\mathrm{E2E}}$ loss from Eq. (12).

C.2 Energy-based optimisation

Building on the notation of Eqs. (8), (9), and (10) of the main paper, one can construct the following overall objective:

$$\mathbf{s}^\star = \underset{\mathbf{s}\in\mathbb{R}^L}{\arg\min}\ \left\{\lambda_1\mathcal{L}_\sigma + \lambda_2\mathcal{L}_{\text{off}} + \lambda_3\mathcal{L}_{\text{ort}}\right\}, \tag{13}$$

for energy-based optimisation. In this work, the optimisation objective is solved using the limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (LBFGS) as an efficient Hessian-based optimiser. From the optimal estimate $\mathbf{s}^\star$, the components $\mathbf{X}^\star$, $\mathbf{D}^\star$, and $\mathbf{Y}^\star$ are obtained analogous to Section 3 of the main paper.

Table 6 supplements the MMP benchmark results in Table 1 of the main paper. The energy-based solutions are close to the corresponding initialization while incurring additional computational cost.

Table 6: Energy-based optimisation results complementing the NPP experiment from Table 1.

Method	RNN	$\overline{\mathrm{SSIM}(\mathbf{M})}\uparrow$	$\overline{\lVert\mathbf{X}\rVert_F^{\mathrm{ort}}}\downarrow$	$\overline{\lVert\mathbf{D}\rVert_F^{\mathrm{rel}}}\downarrow$	$\overline{\lVert\mathbf{Y}\rVert_F^{\mathrm{ort}}}\downarrow$	Time [ms] $\downarrow$
Analytic (Baseline)	✗	0.66 ± 0.08	0.00	0.09 ± 0.04	0.46 ± 0.08	8.48 ± 0.02
LieNN (Baseline)	✓	0.94 ± 0.03	0.00	0.01	0.10 ± 0.03	14.19 ± 0.02
Energy (Analytic init)	✗	0.69 ± 0.08	0.00	0.08 ± 0.03	0.44 ± 0.08	28.13 ± 0.04
Hybrid (LieNN init)	✓	0.94 ± 0.03	0.00	0.01	0.10 ± 0.03	33.69 ± 0.08

C.3 Hybrid learning and energy optimisation

Using the energy-based optimisation as an extension for the neural decompositions in Section 3 of the main paper did not improve on the results but added a large computational overhead.

C.4 Learned recurrent optimiser

While the RNN is used as $\mathbf{A}^{(t+1)} = \mathrm{RNN}\left(\mathbf{A}^{(t)}\right)$ with step t in the presented results, an alternative approach implies computing residuals $\mathbf{r}$ from:

$$\mathbf{r} = \mathbf{d}_\mathbf{X} - \mathbf{d}_\mathbf{Y}$$

with diagonal vectors $\mathbf{d}_\mathbf{X}$ and $\mathbf{d}_\mathbf{Y}$ from Eq. (4) in the main paper for $\mathbf{X}$ and $\mathbf{Y}$, respectively. The intuition is to then let networks be aware of singular value distances by:

$$\begin{aligned}\mathbf{D}_\mathbf{X}^{(t+1)} &= \mathbf{D}_\mathbf{X}^{(t)} + \mathrm{RNN}_\mathbf{X}(\mathbf{r}^{(t)}),\\ \mathbf{D}_\mathbf{Y}^{(t+1)} &= \mathbf{D}_\mathbf{Y}^{(t)} + \mathrm{RNN}_\mathbf{Y}(\mathbf{r}^{(t)}),\end{aligned}$$

to achieve a matching set of left and right singular vectors.

The results of such learned optimisers remain below 0.5 SSIM indicating that the network architectures struggle to find underlying data patterns using this scheme.

D Additional Experimental Results

This appendix provides extended results that exceed the space constraints of the main paper.

D.1 Numerical error growth

We analyze how the metrics in Section 4.1 of the main paper evolve with the matrix dimension m. Since publicly available datasets rarely contain matrices sampled at uniform size intervals, we generate synthetic data with known ground-truth factors, as detailed below. Figure 5 shows how error metrics evolve with the square matrix dimension m.

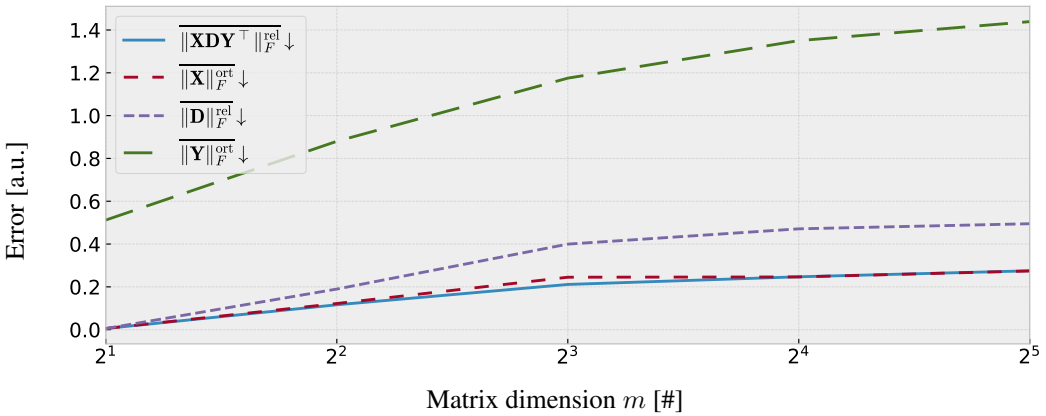
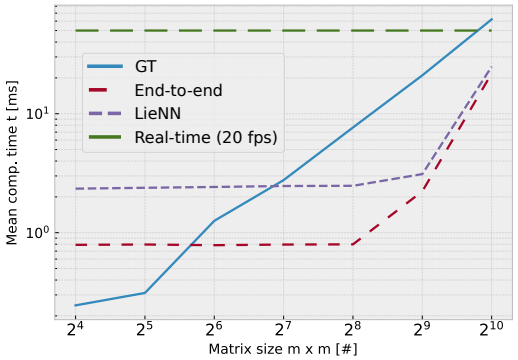


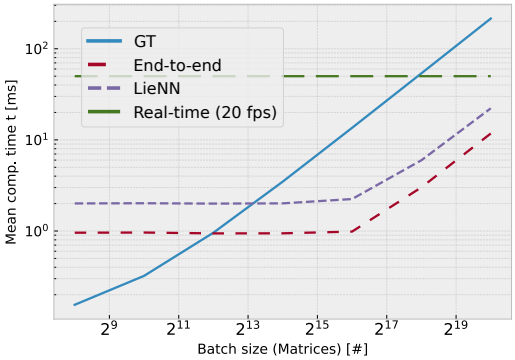
Figure 5: **Error growth for varying dimension m in square matrices.** The curves depict the metrics from Section 4.1 of the main paper, representing the orthogonality errors $\overline{\|\mathbf{X}\|_F^{\text{ort}}}$ and $\overline{\|\mathbf{Y}\|_F^{\text{ort}}}$ as well as deviations for singular values in $\overline{\|\mathbf{D}\|_F^{\text{rel}}}$ and a matrix reconstruction error $\overline{\|\mathbf{XDY}^\top\|_F^{\text{rel}}}$.

D.2 Scalability across matrix and batch sizes

To complement the per-benchmark runtimes reported in the main paper, we evaluate how the best-performing approaches scale with problem size. Figure 6 plots measured execution times as a function of matrix dimension and batch size. The matrix-dimension sweep (Fig. 6a) reflects the increasing number of temporal frames utilised in ULM clutter removal, where deep temporal stacks are often preferred Demené et al. (2015); Baranger et al. (2018); Heiles et al. (2022). Conversely, the batch-size sweep (Fig. 6b) corresponds to the spatial pixel count in a Mueller matrix image, where each pixel requires an independent 3×3 decomposition.



(a) Varying matrix size $m = n$ at constant batch dimension 1



(b) Varying batch size at constant matrix dimension $m = n = 3$

Figure 6: **Computation time plots across matrix and batch sizes.** Timings were measured on an Nvidia H100 GPU to accommodate large memory demands and averaged over 1000 iterations after 100 preceding warm-ups each. Runtime increases sharply with data size, amplifying the gap between the classical SVD and the proposed methods. The plots use logarithmic axes (base-10 on y, base-2 on x), visually compressing exponential growth in runtime.

The plots use logarithmic axes (base-10 on the vertical and base-2 on the horizontal axis), which compress the rapid growth of absolute runtimes and let the curves appear deceptively linear. Up to moderate sizes ($\approx 2^6$ in matrix size; $\approx 2^{13}$ in batch size), presented methods behave similarly while the cuSOLVER-based SVD benefits from the Jacobi routine, which is optimal for small matrices ($m, n \leq 32$). Beyond these ranges, the absolute gaps widen substantially although the log scaling masks their steep divergence. This underscores the practical advantage of the proposed approach as data scales toward high-throughput regimes.

D.3 Additional polarimetric test results

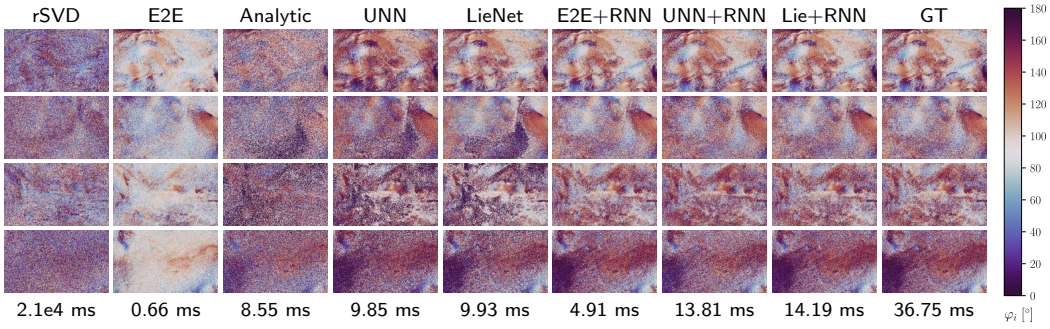


Figure 7: **MMP benchmark comparison (pt. 2).** The images show the remaining NPP test samples of the per-pixel azimuth φ_i decomposed by the various models.

Figure 7 provides the remaining azimuth images from the NPP test set, complementing those shown in Fig. 2 of the main paper. The visual trends mirror the quantitative findings reported earlier.

D.4 Additional MNIST test results

Figure 8 shows the PSNR of each method as a function of truncation rank k on the MNIST patchwork test set. All rank-truncated methods are evaluated at every integer rank from 1 to 280. The per-method optima reported in Table 3 of the main paper correspond to the peak of each curve on the validation split. LieNN maintains a consistent advantage over classical SVD across a broad range of k values, indicating that its benefit is not sensitive to precise rank tuning.

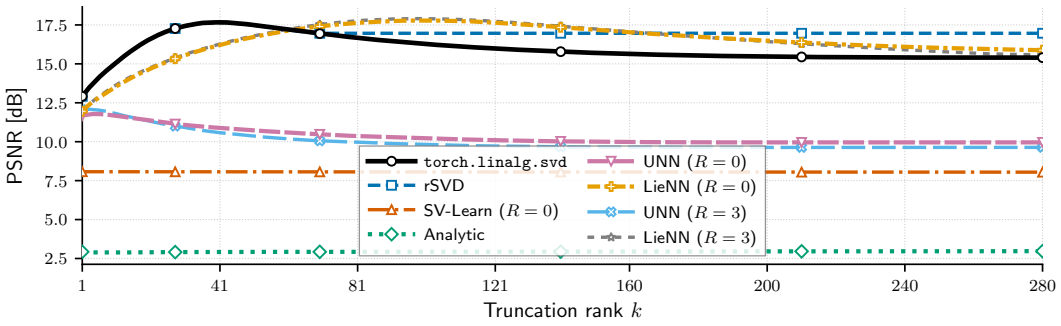


Figure 8: **MNIST patchwork PSNR rank sweep** ($\mathbf{A}_i \in \mathbb{R}^{280\times280}$). PSNR relative to the clean ground truth as a function of truncation rank k for all rank-truncated methods. R denotes the RNN iteration number, while $R = 0$ indicates that the RNN is absent. Curves are evaluated at every integer rank from 1 to 280 on the test split; per-method optima are selected on the validation split.