Mixed-Precision SVD on GPUs via Ogita–Aishima Iterative Refinement

Angelika Schwarz, Rasmus Munk Larsen, Samuel Rodriguez, Christopher Baker, and Lev Kruglyak

NVIDIA Corporation
{abschwarz,rlarsen,srodriguezbe,cbaker,lkruglyak}@nvidia.com

Abstract

The Ogita–Aishima iterative refinement algorithm for SVD is based on matrix–matrix multiplication, making it an attractive approach for GPUs. We extend the algorithm to complex arithmetic, and introduce robust cluster handling based on Wedin's perturbation theory and a per-pair version of Ogita–Aishima's convergence analysis. Our implementation refines an FP32-quality SVD computed with cuSolver's `cusolverDnXgesvdp` to an FP64-quality SVD. We validate the correctness of our mixed-precision implementation by testing on a variety of singular value distributions. In all cases the accuracy is comparable to the FP64 SVD routines `[D,Z]GESVD`. We observe speedups of up to 5.2× over FP64-quality `cusolverDnXgesvdp` on the RTX PRO 6000 Blackwell Workstation Edition GPU.

Keywords mixed-precision, GPU, clustered singular values, cuSolver, numerical linear algebra

1 Introduction

GPUs were adopted for general-purpose scientific computing on the strength of successive generational gains in single- and double-precision (FP32, FP64) throughput. The subsequent growth of artificial-intelligence (AI) workloads on GPUs has since driven two dominant hardware trends [4]: specialized units for matrix and tensor operations (e.g., NVIDIA Tensor Cores), and support for reduced-precision floating-point types (e.g., FP16, BF16, FP8).

Taking full advantage of AI-era GPUs in traditional scientific applications therefore requires exploiting this low-precision hardware while still achieving the accuracy a given application demands. Two approaches are increasingly pursued to this end, often in concert: floating-point emulation, the implicit use of low-precision hardware; and mixed-precision computing, the explicit use of low-precision data types.

Ozaki-style floating-point emulation [12, 16, 23, 17], now productized in vendor libraries [18], lets applications built on matrix multiplication (GEMM) and other level-3 BLAS exploit high-throughput, low-precision units with minimal changes. It decomposes a high-precision (e.g., FP64, FP32) matrix into low-precision "slices" that are multiplied on specialized matrix/tensor hardware and reassembled into an accurate, high-precision result; given a sufficient low-to-high throughput ratio, this yields significant speedups over native arithmetic. Because dense matrix factorizations have long been designed to be rich in level-3 BLAS, real-world speedups in dense linear algebra can be obtained through emulation alone, all the more so for algorithms structured around matrix multiplication.

Mixed-precision algorithms, by contrast, aim for an accurate result while explicitly employing low-precision floating point to gain throughput or memory efficiency [1, 11, 9]. Approaches include low-precision preconditioners [26, 8] and *iterative-refinement* schemes that use low-precision computations to progressively improve a high-precision solution. Iterative refinement has been deployed successfully on GPUs, e.g., for solving linear systems [7] and for selected eigenpairs of the symmetric eigenproblem [21].

In this paper we target the singular value decomposition (SVD) $A = U\Sigma V^*$ of a matrix $A \in \mathbb{C}^{m\times n}$, where $U \in \mathbb{C}^{m\times m}$, $V \in \mathbb{C}^{n\times n}$ are unitary and $\Sigma \in \mathbb{R}^{m\times n}$ is diagonal, real, and non-negative. We extend Ogita–Aishima's refinement algorithm [15, 24], which is attractive for GPUs: it is an iterative-refinement scheme that admits low-precision computation and is dominated by GEMM, so it directly exploits Ozaki-style emulation.

Uchino et al. [24] introduced the first GPU implementation of Ogita–Aishima's GEMM-based SVD refinement, but left one impediment to general applicability as future work: an inability to handle clustered singular values. This is the main issue we address. We present a library-quality GPU implementation with the following contributions:

- **Wedin-union cluster-detection criterion.** Uchino et al. [24] detect clusters based on a consecutive-pair-based test. We use the union of singular value clusters.
- **Cluster-safe refinement loop.** Without cluster handling, close singular values corrupt the basis vectors in the Ogita–Aishima refinement loop and cause divergence. We adapt the loop so that singular values clustered in the sense of our criterion converge to a safe state. This lets us refine well-separated singular values in the loop while refining the clusters in a postprocessing step.
- **Complex-valued extension.** The Ogita–Aishima refinement SVD is stated for a real-valued input matrix [15]. Applied to a complex matrix, the diagonal of the refined projection retains an arbitrary per-vector phase, $t_{jj} = \sigma_j e^{\mathrm{i}\phi_j}$; we add an imaginary diagonal term to the refinement step that drives this phase to zero *in-step*, so the singular values emerge on the real axis without a separate postprocessing pass.
- **Per-pair convergence theory.** The global convergence bound for the Newton-style Ogita–Aishima iteration is inversely proportional to m, limiting the matrix sizes for which the original method is guaranteed convergent. We derive a per-pair bound without this restriction; combined with the cluster-detection criterion, it makes our implementation robust regardless of matrix size and singular-value distribution.

2 Related Work

Mixed-precision algorithms for the singular value decomposition fall into two families: iterative-refinement methods, which improve a low-precision factorization through Newton-style updates; and Jacobi-based methods, which diagonalize through plane rotations driven by mixed-precision computations. The two families differ in their reliance on matrix multiplication and consequently in how they exploit modern accelerators.

2.1 Ogita–Aishima Iterative Refinement for Eigendecomposition

Ogita and Aishima introduced a Newton-style iterative refinement algorithm for the symmetric eigenvalue decomposition [13]. The algorithm couples the orthogonality condition $U^T U = I$ with the diagonality condition $U^T A U = \Lambda$ to obtain correction matrices for an approximate eigendecomposition. Each step is dominated by GEMMs, and the iteration converges quadratically when the initial approximation is sufficiently accurate and the eigenvalues are well separated.

The original algorithm assumes simple, well-separated eigenvalues. To handle clustered eigenvalues, Ogita and Aishima extended the method in a follow-up paper [14]. Their extension uses the Davis–Kahan theorem to detect clusters, solves a small high-precision eigendecomposition on the projected subspace, and exploits the shift invariance of eigenvectors to refine clusters by introducing a diagonal shift that increases the relative gap.

The framework has since been deployed at scale and extended to related problems. Uchino et al. [22] integrate Ogita–Aishima refinement into EigenExa [10], exploiting its GEMM-dominated structure on large-scale CPU systems. For non-Hermitian problems, Bujanović et al. [2] develop a Newton-like refinement of the Schur decomposition that reduces, in a low-/high-precision setting, to four high-precision GEMMs per iteration; Terao [19] generalizes the Ogita–Aishima refinement to diagonalizable non-Hermitian eigendecompositions in a similar spirit. Concurrently with the present work, Terao et al. [20] combine Ogita–Aishima refinement with the Ozaki scheme on CPU for the symmetric eigenproblem, demonstrating the algorithmic affinity between iterative refinement and low-precision GEMM emulation.

2.2 Mixed-precision SVD methods

Ogita and Aishima adapted their refinement framework to the singular value decomposition [15]. As in the eigenvalue case, the corrections are derived from the orthogonality and diagonality conditions, and the per-step cost is dominated by GEMMs. The SVD case differs in that each off-diagonal pair (i, j) requires solving a 2×2 linear system that couples the orthogonality conditions on the left and right factors with the diagonality of $U^T AV$. The iteration converges quadratically under the same separation assumption, now applied to the singular values. The algorithm is stated for real-valued matrices and assumes simple, well-separated singular values; clustered singular values are not handled.

Building on this formulation, Uchino et al. [24] reduce the per-iteration cost to four or five GEMMs and present the first GPU implementation. They establish Ogita–Aishima refinement as a practical mixed-precision strategy on GPUs, but explicitly leave the handling of clustered singular values for future work; their applicability test, based on consecutive singular-value pairs, rejects matrices that fall outside the basin of attraction of the original algorithm. Lifting this restriction and extending the SVD refinement to complex matrices are the main contributions of the present paper.

An alternative approach pursues mixed-precision through the Jacobi SVD algorithm, building on the fast and accurate one-sided Jacobi SVD of Drmač et al. [5]. Gao et al. [6] perform the bulk of the orthogonalization in low precision and design corrections that recover high-precision accuracy. Zhou et al. [27] use three precisions and a low-precision preconditioner to obtain a one-sided Jacobi SVD with forward-error bounds in the scaled condition number, extending an earlier three-precision framework for the symmetric eigenproblem of Higham et al. [8]. A different mixed-precision strategy, targeted at tall-and-skinny matrices, is taken by Carson et al. [3], who form the Gram matrix in a precision *higher* than the working precision and combine it with a Jacobi eigensolver to retain high relative accuracy. Jacobi-based methods do not rely on matrix multiplication and so do not directly benefit from tensor-core throughput; in return, they handle tiny and clustered singular values naturally and attain high relative accuracy on the smallest singular values.

3 Ogita–Aishima SVD Refinement Algorithm

In this section, we summarize the Ogita–Aishima algorithm for the singular value decomposition (SVD) [15]. We adopt the notation used in the original paper [15] and denote the true SVD of a matrix $A \in \mathbb{R}^{m\times n}$ with $m \geq n$ as $A = U\Sigma V^T$. Here $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal and correspond to the left and right singular vectors, respectively, and $\Sigma \in \mathbb{R}^{m\times n}$ holds the singular values $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$ on its diagonal. An approximation of a quantity w is denoted $\widetilde{w}$ and a computed result $\widehat{w}$.

Algorithm 1 RefSVD — refinement step for approximate singular vectors of a real matrix [15, Algorithm 1]. High precision is required for all steps.

Input: $A \in \mathbb{R}^{m\times n}$ with $m \geq n$; approximate $\widehat{U} \in \mathbb{R}^{m\times m}$, $\widehat{V} \in \mathbb{R}^{n\times n}$.
Output: refined $\widehat{U}', \widehat{V}'$ and $\widehat{\Sigma}' = \mathrm{diag}(\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_n) \in \mathbb{R}^{m\times n}$.

$R \leftarrow I_m - \widehat{U}^T\widehat{U}, \quad S \leftarrow I_n - \widehat{V}^T\widehat{V}, \quad T \leftarrow \widehat{U}^T A\widehat{V}$
$\widetilde{\sigma}_i \leftarrow t_{ii} \,/\, \big(1 - (r_{ii} + s_{ii})/2\big) \quad$ for $i = 1, \ldots, n$
$\widetilde{f}_{ii} \leftarrow r_{ii}/2, \quad \widetilde{g}_{ii} \leftarrow s_{ii}/2 \quad$ for $1 \leq i \leq n$
for $1 \leq i, j \leq n$ with $i \neq j$ **do**
    $\alpha_{ij} \leftarrow t_{ij} + \widetilde{\sigma}_j\, r_{ij}, \quad \beta_{ij} \leftarrow t_{ji} + \widetilde{\sigma}_j\, s_{ij}$
    $(\widetilde{f}_{ij}, \widetilde{g}_{ij}) \leftarrow \dfrac{1}{\widetilde{\sigma}_j^2 - \widetilde{\sigma}_i^2}\big(\alpha_{ij}\,\widetilde{\sigma}_j + \beta_{ij}\,\widetilde{\sigma}_i,\ \alpha_{ij}\,\widetilde{\sigma}_i + \beta_{ij}\,\widetilde{\sigma}_j\big)$
end for
$\widetilde{f}_{ij} \leftarrow -t_{ji}/\widetilde{\sigma}_i \quad$ for $1 \leq i \leq n,\ n+1 \leq j \leq m$
$\widetilde{f}_{ij} \leftarrow r_{ij} - \widetilde{f}_{ji} \quad$ for $n+1 \leq i \leq m,\ 1 \leq j \leq n$
$\widetilde{f}_{ij} \leftarrow r_{ij}/2 \quad$ for $n+1 \leq i, j \leq m$
$\widehat{U}' \leftarrow \widehat{U} + \widehat{U}\,\widetilde{F}, \quad \widehat{V}' \leftarrow \widehat{V} + \widehat{V}\,\widetilde{G}$

Algorithm 1 shows the original refinement algorithm presented by Ogita and Aishima [15]. It couples the orthogonality conditions $U^TU = I_m$ and $V^TV = I_n$ with the diagonality condition $U^TAV = \Sigma$ to

obtain a first-order Newton method. For the approximation $\widehat{U}$, the orthogonality requirement implies $(I_m + F)^T \widehat{U}^T \widehat{U}(I_m + F) = I_m$, where F is a correction matrix with $U = \widehat{U}(I_m + F)$. The matrix $\widehat{V}$ with its correction G is analogous. The diagonal entries of the correction matrices computed in line 3 of Algorithm 1 correspond to these orthogonality conditions. The off-diagonal entries computed in line 6 stem from solving a 2×2 linear system that couples the diagonality and orthogonality conditions. For a detailed derivation, we refer to Ogita and Aishima [15, Section 2]. They prove that the refinement algorithm converges quadratically if the initial correction matrices satisfy [15, Theorem 1]

$$\|F\|_2,\ \|G\|_2 \ <\ \min\Big(\frac{\delta_{\min}}{30\,m\,\|A\|_2},\ \frac{1}{60}\Big), \tag{1}$$

where $\delta_{\min} := \min_{1 \le i \le n}(\sigma_i - \sigma_{i+1})$ is the smallest singular-value gap of A, with $\sigma_{n+1} := 0$ for convenience. Under these conditions, the approximations of the singular vector matrices converge towards the true singular vectors.

Each iteration of Algorithm 1 requires six GEMMs. The remaining steps are $O(n^2)$ scalar operations.

4 Extension to Complex Arithmetic and Clusters

In this section, we extend the algorithm to complex-valued matrices and explain how we use Wedin's perturbation theory for two purposes: cluster detection and adaptation of the refinement algorithm to be cluster-safe.

4.1 Cluster Detection and Rayleigh–Ritz Refinement

Cluster Detection Ogita and Aishima's refinement algorithm for the symmetric eigenproblem [13, 14] detects clusters using the Davis–Kahan theorem. Clustered eigenvalues are refined in a postprocessing step that increases the relative gap by a diagonal shift, exploiting the shift invariance of eigenvectors. The SVD analogue of the Davis–Kahan theorem is Wedin's theorem [25]. Since singular vectors are not preserved under a diagonal shift, Ogita and Aishima's shifting strategy does not transfer to SVD. Instead, we modify the construction of the correction matrices (line 6 of Algorithm 1): If the singular values are well-separated according to a criterion based on Wedin's theorem, we use the original construction; otherwise, we drive the correction matrices to a safe state.

Wedin's theorem [25] provides a simultaneous bound on the perturbation in the subspaces spanned by the left and right singular vectors. It justifies treating clusters as blocks. The simultaneous nature of the bound allows us to devise a single cluster detection criterion and cluster refinement step rather than treating left and right subspaces separately.

Inspired by the cluster threshold for the symmetric eigenproblem [14, line 6 of Algorithm 1], we use

$$\omega = 2\,(\|\operatorname{offdiag}(T)\|_F + \|A\|_F \max(\|R\|_F, \|S\|_F)) \tag{2}$$

as the cluster threshold. That is, pairs of singular values whose gap is below ω are considered clustered and require refinement in a postprocessing step. Recall that R and S are the orthogonality residuals and T measures the diagonality (line 1 of Algorithm 1). We choose the Frobenius norm over the spectral norm used in [14] for ease of computation.

Remark. Uchino et al. [24, Section 2.4] consider the singular values clustered when

$$\max(\|F\|_2, \|G\|_2) \ge \frac{\min_{1 \le i \le n-1}(\sigma_i - \sigma_{i+1})}{30\,m\,\|A\|_2}.$$

In other words, their criterion is based on the basin of attraction, see (11). If any pair of singular values meets the criterion, the problem is considered to not be refinable with the basic Ogita–Aishima algorithm. Since our objective is not only detection but also refinement of clusters, our criterion identifies singular values that are below the achievable precision of the current iterate.

Algorithm 2 ClusterRR — Rayleigh–Ritz refinement of a single cluster.

Input: $A \in \mathbb{C}^{m\times n}$ with $m \geq n$; approximate $\widehat{U} \in \mathbb{C}^{m\times m}$, $\widehat{V} \in \mathbb{C}^{n\times n}$; cluster index set $\mathcal{J}_k \subset \{1,\dots,n\}$.
Output: Updated $\widehat{U}(:,\mathcal{J}_k)$, $\widehat{V}(:,\mathcal{J}_k)$, and $\widetilde{\sigma}_{\mathcal{J}_k}$.
1: $C_k \leftarrow \widehat{U}(:,\mathcal{J}_k)^H\, A\, \widehat{V}(:,\mathcal{J}_k)$
2: $[P_k, \Sigma_k, Q_k] \leftarrow \mathrm{SVD}(C_k)$ ▷ Working-precision SVD.
3: $\widehat{U}(:,\mathcal{J}_k) \leftarrow \widehat{U}(:,\mathcal{J}_k)\, P_k, \quad \widehat{V}(:,\mathcal{J}_k) \leftarrow \widehat{V}(:,\mathcal{J}_k)\, Q_k, \quad \widetilde{\sigma}_{\mathcal{J}_k} \leftarrow \mathrm{diag}(\Sigma_k)$

Rayleigh–Ritz Refinement For each detected cluster – identified by the index set $\mathcal{J}_k \subset \{1,\dots,n\}$ – we restore FP64-accurate bases by a Rayleigh–Ritz-style procedure applied to the cluster's left and right singular subspaces: we compute the SVD of the projected block $C_k := \widehat{U}(:,\mathcal{J}_k)^H\, A\, \widehat{V}(:,\mathcal{J}_k)$ and rotate $\widehat{U}(:,\mathcal{J}_k)$ and $\widehat{V}(:,\mathcal{J}_k)$ by its singular vectors. Recall that the cluster bases are only approximately orthonormal at this point. Algorithm 2 states the Rayleigh–Ritz step for a single cluster. Its accuracy is governed by Wedin's theorem applied to the cluster block.

Theorem 1 (Rayleigh–Ritz refinement accuracy)**.** *Let $\mathcal{J}_k$ be a cluster index set with inter-cluster separation $\mathrm{sep}(\mathcal{J}_k) := \min_{j\in\mathcal{J}_k,\, j'\notin\mathcal{J}_k} |\widetilde{\sigma}_j - \widetilde{\sigma}_{j'}| > 0$, let $T := \widehat{U}^H A \widehat{V}$ and $E := \mathrm{offdiag}(T)$ at the entry to Algorithm 2, and let $[P_k, \Sigma_k, Q_k] = \mathrm{SVD}(C_k)$. Then there exist unitary matrices W_U, W_V such that*

$$\|\widehat{U}(:,\mathcal{J}_k)\, P_k - U(:,\mathcal{J}_k)\, W_U\|_F \;\leq\; \|E\|_F/\,\mathrm{sep}(\mathcal{J}_k), \qquad \|\widehat{V}(:,\mathcal{J}_k)\, Q_k - V(:,\mathcal{J}_k)\, W_V\|_F \;\leq\; \|E\|_F/\,\mathrm{sep}(\mathcal{J}_k). \tag{3}$$

Proof. Partition $T = \widehat{U}^H A \widehat{V}$ along the cluster index sets as $T = \mathrm{blockdiag}_k(C_k) + E_{\mathrm{off}}$, where E_{off} collects the inter-cluster blocks and is bounded by $\|E_{\mathrm{off}}\|_F \leq \|E\|_F$. The diagonal blocks C_k are the inputs to Algorithm 2 on line 1. By definition of $\mathrm{sep}(\mathcal{J}_k)$, the spectrum of C_k is separated from the spectra of all other diagonal blocks of T by at least $\mathrm{sep}(\mathcal{J}_k)$. Wedin's theorem applied to the cluster block with separation $\mathrm{sep}(\mathcal{J}_k)$ and perturbation E_{off} then bounds the canonical angles between the singular subspaces of C_k and those of $U(:,\mathcal{J}_k)$, $V(:,\mathcal{J}_k)$ by $\|E_{\mathrm{off}}\|_F/\,\mathrm{sep}(\mathcal{J}_k) \leq \|E\|_F/\,\mathrm{sep}(\mathcal{J}_k)$. Choosing W_U, W_V as the unitaries that align the bases converts the canonical-angle bound to the bound (3). □

Theorem 1 certifies that Algorithm 2 produces within-cluster bases at high-precision accuracy as long as the inter-cluster separation $\mathrm{sep}(\mathcal{J}_k)$ is large compared to the residual $\|E\|_F$. This bound transfers to floating-point arithmetic.

4.2 Cluster-safe Refinement Step

This section devises a cluster-safe fallback and the condition for taking it. Together they extend Algorithm 1 so that the loop refines all well-separated singular values to the target precision while keeping clustered singular values in a safe state for postprocessing.

We focus on the correction matrix $\widetilde{F}$; $\widetilde{G}$ is analogous. Recall that $R = I - \widehat{U}^H\widehat{U}$ and $S = I - \widehat{V}^H\widehat{V}$ are Hermitian, so $\overline{r_{ji}} = r_{ij}$, $\overline{s_{ji}} = s_{ij}$, and their diagonals r_{ii}, s_{ii} are real. The standard update (line 6 of Algorithm 1) computes $\widetilde{f}_{ij} = (\alpha_{ij}\widetilde{\sigma}_j + \beta_{ij}\widetilde{\sigma}_i)/(\widetilde{\sigma}_j^2 - \widetilde{\sigma}_i^2)$, where $\alpha_{ij} \leftarrow t_{ij} + \widetilde{\sigma}_j r_{ij}$ and $\beta_{ij} \leftarrow \overline{t_{ji}} + \widetilde{\sigma}_j s_{ij}$.

We will split $\widetilde{f}_{ij}$ into benign and "unstable" terms where $\widetilde{\sigma}_j \approx \widetilde{\sigma}_i$. Specifically, we split

$$\widetilde{f}_{ij} = \frac{t_{ij}\widetilde{\sigma}_j + \overline{t_{ji}}\widetilde{\sigma}_i + \widetilde{\sigma}_j^2 r_{ij} + \widetilde{\sigma}_i\widetilde{\sigma}_j s_{ij}}{\widetilde{\sigma}_j^2 - \widetilde{\sigma}_i^2} = \frac{t_{ij} + \overline{t_{ji}}}{2(\widetilde{\sigma}_j - \widetilde{\sigma}_i)} + \frac{t_{ij} - \overline{t_{ji}}}{2(\widetilde{\sigma}_j + \widetilde{\sigma}_i)} + \frac{\widetilde{\sigma}_j(r_{ij} + \overline{s_{ji}})}{2(\widetilde{\sigma}_j - \widetilde{\sigma}_i)} + \frac{\widetilde{\sigma}_j(r_{ij} - \overline{s_{ji}})}{2(\widetilde{\sigma}_j + \widetilde{\sigma}_i)}.$$

If we substitute $\widetilde{\sigma}_j = \frac{1}{2}(\widetilde{\sigma}_j - \widetilde{\sigma}_i) + \frac{1}{2}(\widetilde{\sigma}_j + \widetilde{\sigma}_i)$ in the last two terms, we obtain

$$\widetilde{f}_{ij} = \frac{r_{ij}}{2} + \underbrace{\frac{t_{ij} + \overline{t_{ji}}}{2(\widetilde{\sigma}_j - \widetilde{\sigma}_i)}}_{=:P_1} + \underbrace{\frac{t_{ij} - \overline{t_{ji}}}{2(\widetilde{\sigma}_j + \widetilde{\sigma}_i)}}_{=:P_2} + \underbrace{\frac{r_{ij} + \overline{s_{ji}}}{4}\frac{\widetilde{\sigma}_j + \widetilde{\sigma}_i}{\widetilde{\sigma}_j - \widetilde{\sigma}_i}}_{=:P_3} + \underbrace{\frac{r_{ij} - \overline{s_{ji}}}{4}\frac{\widetilde{\sigma}_j - \widetilde{\sigma}_i}{\widetilde{\sigma}_j + \widetilde{\sigma}_i}}_{=:P_4}.$$

Analogously, we can write $\widetilde{g}_{ij} = \overline{s_{ji}}/2 + P_1 - P_2 + P_3 - P_4$. For any pair of close singular values, the terms with $\widetilde{\sigma}_j - \widetilde{\sigma}_i$ in the denominator cannot be stably computed. The terms with the denominator $\widetilde{\sigma}_j + \widetilde{\sigma}_i$ can

only be safely computed if the singular values are sufficiently far away from zero. This determines which terms we use in the two Wedin fallback cases. First, if the singular values are sufficiently far away from zero, the terms P_2 and P_4 can be safely evaluated and P_4 vanishes when $\widetilde{\sigma}_j - \widetilde{\sigma}_i \approx 0$. Using $\overline{s_{ji}} = s_{ij}$, we obtain the corrections

$$(\widetilde{f}_{ij}, \widetilde{g}_{ij}) \leftarrow \left(\tfrac{1}{2} r_{ij} + P_2,\ \ \tfrac{1}{2} s_{ij} - P_2\right). \tag{4}$$

Second, if the singular values are near zero, any denominator $\widetilde{\sigma}_j \pm \widetilde{\sigma}_i$ cannot safely be evaluated. Then the correction matrices are set to

$$(\widetilde{f}_{ij}, \widetilde{g}_{ij}) \leftarrow \left(\tfrac{1}{2} r_{ij},\ \ \tfrac{1}{2} s_{ij}\right). \tag{5}$$

Both fallback cases are the natural extension of the original update, preserving all components that can be stably evaluated. What remains is a justification why these fallbacks are valid.

Ogita and Aishima [15, eqs. (17)–(20)] set up the correction as a system of equations that, for complex matrices, reads

$$\widetilde{f}_{ij} + \overline{\widetilde{f}_{ji}} = r_{ij}, \qquad \widetilde{g}_{ij} + \overline{\widetilde{g}_{ji}} = s_{ij}, \tag{6}$$

$$\widetilde{\sigma}_i\, \widetilde{f}_{ij} + \widetilde{\sigma}_j\, \overline{\widetilde{g}_{ji}} = -\overline{t_{ji}}, \qquad \widetilde{\sigma}_j\, \overline{\widetilde{f}_{ji}} + \widetilde{\sigma}_i\, \widetilde{g}_{ij} = -t_{ij}. \tag{7}$$

We first focus on $\widetilde{\sigma}_j = \widetilde{\sigma}_i = 0$. The four equations reduce to the orthogonality conditions (6), two equations with four unknowns, and the constraint $t_{ij} = \overline{t_{ji}} = 0$. Analogously to Ogita and Aishima's artificial closure for the symmetric eigenproblem, we choose $\widetilde{f}_{ij} = \overline{\widetilde{f}_{ji}}$ and $\widetilde{g}_{ij} = \overline{\widetilde{g}_{ji}}$ as the two artificial conditions, enforcing that $\widetilde{F}$ and $\widetilde{G}$ are Hermitian. This yields the second fallback update (5).

Next, we address $\widetilde{\sigma}_j = \widetilde{\sigma}_i$ when the singular values are sufficiently far away from zero. Substituting the orthogonality conditions (6) into the diagonality equations (7) and setting $\sigma := \widetilde{\sigma}_j = \widetilde{\sigma}_i$ yields

$$\sigma\, \widetilde{f}_{ij} - \sigma\, \widetilde{g}_{ij} = t_{ij} + \sigma\, r_{ij}, \qquad \sigma\, \widetilde{f}_{ij} - \sigma\, \widetilde{g}_{ij} = -\overline{t_{ji}} - \sigma\, s_{ij}. \tag{8}$$

The system is underdetermined and we again have to choose one artificial closure condition to have a unique solution. Since we have only one degree of freedom, we set $\widetilde{f}_{ij} + \widetilde{g}_{ij} = \overline{\widetilde{f}_{ji}} + \overline{\widetilde{g}_{ji}}$ as the artificial condition. This is the closest choice we can make to mimic the $\widetilde{\sigma}_j = \widetilde{\sigma}_i = 0$ case, forcing $\widetilde{F} + \widetilde{G}$ to be Hermitian. Equating the right-hand sides of (8) gives the consistency condition $t_{ij} + \overline{t_{ji}} = -\sigma(r_{ij} + s_{ij})$. Plugged into (8), we obtain

$$\widetilde{f}_{ij} - \widetilde{g}_{ij} = \frac{t_{ij}}{\sigma} + r_{ij} = \underbrace{\frac{t_{ij} + \overline{t_{ji}}}{2\sigma}}_{-(r_{ij}+s_{ij})/2} + \underbrace{\frac{t_{ij} - \overline{t_{ji}}}{2\sigma}}_{2P_2} + r_{ij} = \frac{r_{ij} - s_{ij}}{2} + 2P_2. \tag{9}$$

Adding the orthogonality conditions in (6) yields $(\widetilde{f}_{ij} + \widetilde{g}_{ij}) + (\overline{\widetilde{f}_{ji}} + \overline{\widetilde{g}_{ji}}) = r_{ij} + s_{ij}$, which with the artificial closure gives

$$\widetilde{f}_{ij} + \widetilde{g}_{ij} = \frac{r_{ij} + s_{ij}}{2}. \tag{10}$$

The solution of (9) and (10) is the first fallback update (4).

This artificial closure mirrors the symmetric eigenproblem, where a multiplicity likewise leaves a single degree of freedom: Ogita and Aishima [13, Sections 2 and 3.2] impose the orthogonality condition plus an artificial symmetry condition for uniqueness. Unique to the SVD is the antisymmetric contribution P_2, which has no eigenproblem analogue. As in [14], which extends the closure to clusters by treating them as perturbed multiplicities, our SVD closure does the same.

The diagonal entries ($i = j$) are the limiting case of this antisymmetric correction. As the gap vanishes, P_1 and P_3 are dropped and $P_4 = 0$, leaving $\widetilde{f}_{ii} = r_{ii}/2 + P_2$ and $\widetilde{g}_{ii} = s_{ii}/2 - P_2$ with $P_2|_{i=j} = (t_{ii} - \overline{t_{ii}})/(4\widetilde{\sigma}_i) = \mathrm{i}\,\mathrm{Im}(t_{ii})/(2\widetilde{\sigma}_i) =: \delta_i$, gated like the antisymmetric term by $2\widetilde{\sigma}_i > \sqrt{\varepsilon_{\mathrm{lp}}}\,\widetilde{\sigma}_{\max}$ (line 4 of Algorithm 3). The real parts are the usual orthogonality correction; the increment δ_i is the per-vector phase, a true U(1) gauge of the SVD, fixed here by the same closure $\widetilde{f}_{ii} + \widetilde{g}_{ii} = (r_{ii} + s_{ii})/2$ underlying (4). It rotates the complex diagonal of $T = \widehat{U}^H A \widehat{V}$ onto the real axis *during* refinement, so the singular values emerge real and non-negative with no separate phase-correction pass; for real A, $\mathrm{Im}(t_{ii}) = 0$ and δ_i vanishes.

Algorithm 3 MxpSVDStep — cluster-safe extension of Algorithm 1. High precision is required for all steps.

Input: $A \in \mathbb{C}^{m\times n}$ with $m \geq n$; approximate $\widehat{U} \in \mathbb{C}^{m\times m}$, $\widehat{V} \in \mathbb{C}^{n\times n}$.
Output: refined $\widehat{U}', \widehat{V}'$ and $\widehat{\Sigma}' = \mathrm{diag}(\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_n) \in \mathbb{R}^{m\times n}$; $\omega \in \mathbb{R}$.

$R \leftarrow I - \widehat{U}^H\widehat{U}, \quad S \leftarrow I - \widehat{V}^H\widehat{V}, \quad T \leftarrow \widehat{U}^H A\widehat{V}$
$\widetilde{\sigma}_i \leftarrow |t_{ii}| \,/\, \big(1 - \frac{1}{2}(\mathrm{Re}(r_{ii}) + \mathrm{Re}(s_{ii}))\big) \quad$ for $i = 1, \ldots, n$
$\widetilde{f}_{ii} \leftarrow r_{ii}/2, \quad \widetilde{g}_{ii} \leftarrow s_{ii}/2, \quad \delta_i \leftarrow \mathrm{i}\,\mathrm{Im}(t_{ii})/(2\widetilde{\sigma}_i) \quad$ for $1 \leq i \leq n$
$(\widetilde{f}_{ii}, \widetilde{g}_{ii}) \leftarrow (\widetilde{f}_{ii} + \delta_i,\ \widetilde{g}_{ii} - \delta_i) \quad$ if $2\widetilde{\sigma}_i > \sqrt{\varepsilon_{\mathrm{lp}}}\,\widetilde{\sigma}_{\max}$ ▷ diagonal phase: P_2 at $i = j$
for $1 \leq i, j \leq n$ with $i \neq j$ **do**
  if $|\widetilde{\sigma}_j - \widetilde{\sigma}_i| > \sqrt{\varepsilon_{\mathrm{lp}}}\,\widetilde{\sigma}_{\max}$ **then**
    $\alpha_{ij} \leftarrow t_{ij} + \widetilde{\sigma}_j\, r_{ij}, \quad \beta_{ij} \leftarrow \overline{t_{ji}} + \widetilde{\sigma}_j\, s_{ij}$
    $(\widetilde{f}_{ij}, \widetilde{g}_{ij}) \leftarrow ((\widetilde{\sigma}_j - \widetilde{\sigma}_i)(\widetilde{\sigma}_j + \widetilde{\sigma}_i))^{-1}\big(\alpha_{ij}\,\widetilde{\sigma}_j + \beta_{ij}\,\widetilde{\sigma}_i,\ \alpha_{ij}\,\widetilde{\sigma}_i + \beta_{ij}\,\widetilde{\sigma}_j\big)$ ▷ original update
  else if $\widetilde{\sigma}_i + \widetilde{\sigma}_j > \sqrt{\varepsilon_{\mathrm{lp}}}\,\widetilde{\sigma}_{\max}$ **then**
    $a_{ij} \leftarrow (t_{ij} - \overline{t_{ji}})/\big(2(\widetilde{\sigma}_i + \widetilde{\sigma}_j)\big)$
    $(\widetilde{f}_{ij}, \widetilde{g}_{ij}) \leftarrow \big(\frac{1}{2}r_{ij} + a_{ij},\ \frac{1}{2}s_{ij} - a_{ij}\big)$ ▷ orthogonality and antisymmetric correction
  else
    $(\widetilde{f}_{ij}, \widetilde{g}_{ij}) \leftarrow \big(\frac{1}{2}r_{ij},\ \frac{1}{2}s_{ij}\big)$ ▷ orthogonality correction only
  end if
end for
$\widetilde{f}_{ij} \leftarrow -\overline{t_{ji}}/\widetilde{\sigma}_i \quad$ for $1 \leq i \leq n$, $n+1 \leq j \leq m$
$\widetilde{f}_{ij} \leftarrow r_{ij} - \overline{\widetilde{f}_{ji}} \quad$ for $n+1 \leq i \leq m$, $1 \leq j \leq n$
$\widetilde{f}_{ij} \leftarrow r_{ij}/2 \quad$ for $n+1 \leq i, j \leq m$
$\omega \leftarrow 2\big(\|\mathrm{offdiag}(T)\|_F + \|A\|_F \max(\|R\|_F, \|S\|_F)\big)$ ▷ threshold for cluster identification
$\widehat{U}' \leftarrow \widehat{U} + \widehat{U}\,\widetilde{F}, \quad \widehat{V}' \leftarrow \widehat{V} + \widehat{V}\,\widetilde{G}$

Numerical details. Two floating-point considerations arise in MxpSVDStep. The denominator $\widetilde{\sigma}_j^2 - \widetilde{\sigma}_i^2$ is evaluated in the factored form $(\widetilde{\sigma}_j - \widetilde{\sigma}_i)(\widetilde{\sigma}_j + \widetilde{\sigma}_i)$, which is more accurate when $\widetilde{\sigma}_j \approx \widetilde{\sigma}_i$: the singular values are non-negative, so Sterbenz's lemma then guarantees that the difference $\widetilde{\sigma}_j - \widetilde{\sigma}_i$ is computed exactly, and the full denominator is accurate to 2 units in the last place. And although the diagonals of R and S are real, their *computed* values carry a small imaginary rounding error, so the singular-value update takes $\mathrm{Re}(r_{ii})$ and $\mathrm{Re}(s_{ii})$.

4.3 Wedin-Union Cluster Criterion

We use a two-stage pipeline for our cluster-handling iterative SVD refinement. The first stage is a refinement loop that leaves clusters untouched. This is realized by repeating MxpSVDStep (Algorithm 3) until all well-separated singular values have converged. The second stage performs cluster refinement, which handles the singular values that could not be refined in the first stage by solving sub-SVDs. We prefer this pipeline over alternating the two stages for performance reasons. The first stage is dominated by GEMMs and runs efficiently on GPUs, while the potentially costly sub-SVDs of the second stage are invoked only once.

Our single-pass design, however, requires careful cluster detection. As we repeatedly execute MxpSVDStep, the cluster threshold ω defined in expression (2) shrinks. The orthogonality residuals R and S reach high-precision accuracy; the contribution of $\mathrm{offdiag}(T)$ depends on the spectrum because the fallback paths only drive the antisymmetric component to zero. Hence, the ω criterion captures singular value pairs whose gap is below the accuracy reached by the first stage. We highlight that the ω criterion also allows safe switching to the second stage before the first stage is fully converged.

The first stage's resolution is bounded by the low-precision SVD's resolution, as well as the per-pair convergence of the refinement step, which we establish rigorously in Section 4.4. This in particular affects singular values whose gap lies between $\varepsilon_{\mathrm{hp}}\sigma_{\max}$ and $\varepsilon_{\mathrm{lp}}\sigma_{\max}$. Hence, we need a second criterion that marks these singular values for refinement. We base this criterion on Wedin's theorem. In summary, we use the following two complementary criteria:

- $|\sigma_i - \sigma_j| \le \omega$. Pairs of singular values whose gap is below the accuracy level reached by the current iterate of MxpSVDStep have not been accurately resolved.
- $|\sigma_j - \sigma_i| \le \sqrt{\varepsilon_{\mathrm{lp}}}\,\sigma_{\max}$. Pairs whose gap falls below the per-pair convergence criterion will not have been refined by MxpSVDStep.

If a pair of singular values meets either criterion, it is handled in the cluster refinement stage. We refer to these two criteria as the Wedin-union cluster criterion.

4.4 Per-Pair Convergence

Theorem 2 ([15, Theorem 1]). *Let $A \in \mathbb{R}^{m\times n}$, $\widehat{U} \in \mathbb{R}^{m\times m}$, and $\widehat{V} \in \mathbb{R}^{n\times n}$ with $m \ge n$ and $m \ge 2$. Define $\sigma_{n+1} := 0$ for the sake of convenience. Define $\epsilon := \max(\|F\|, \|G\|)$ with F, G satisfying $U = \widehat{U}(I_m + F)$, $V = \widehat{V}(I_n + G)$. Similarly, define $\epsilon' := \max(\|F'\|, \|G'\|)$ with F', G' satisfying $U' = \widehat{U}'(I_m + F')$, $V' = \widehat{V}'(I_n + G')$, where $\widehat{U}', \widehat{V}'$ are obtained in Algorithm 1. If*

$$\epsilon < \min_{1\le i\le n} (\sigma_i - \sigma_{i+1})/(30m\|A\|) \tag{11}$$

is satisfied, then

$$\epsilon' < \tfrac{7}{10}\epsilon, \qquad \limsup_{\epsilon\to 0} \epsilon'/\epsilon^2 \le 18m\|A\|/\min_{1\le i\le n}(\sigma_i - \sigma_{i+1}).$$

Theorem 2 ties the convergence rate of Algorithm 1 to the global smallest gap $\delta_{\min}$ through inequality (11) that also scales with m — far too pessimistic once small-gap pairs are handled separately by the Rayleigh–Ritz step (Algorithm 2). The following *per-pair* analysis instead classifies each index pair by the branch test of Algorithm 3: well-separated pairs (Group A) are corrected to second order at a rate set by their *own* gap, with no dependence on m or $\delta_{\min}$, while the rest (Groups B and C) are deferred to the cluster refinement stage. We assume real arithmetic to align with the original analysis.

Theorem 3 (Per-pair convergence of cluster-safe MxpSVDStep). *Let $\varepsilon_{\mathrm{lp}}$ be the low-precision unit roundoff so that $c := \sqrt{\varepsilon_{\mathrm{lp}}}$ is the branch-test coefficient of Algorithm 3. Let further $\epsilon_k := \max(\|F^{(k)}\|_2, \|G^{(k)}\|_2)$. Assume*

$$\epsilon_k \;\le\; \sqrt{c/\big((c+6)\eta_{\max}\big)} \tag{12}$$

where $\eta_{\max} \le 6.573$ is given by inequality (36) of [15]. For each index pair (i, j) with $i \ne j$:

***(i)** – **Group A**: If (i, j) satisfies $|\widetilde{\sigma}_i - \widetilde{\sigma}_j| > c\,\widetilde{\sigma}_{\max}$, then there is a constant $C_1 \le 13$, such that*

$$|\widetilde{f}_{ij}^{(k)} - f_{ij}^{(k)}| \;\le\; C_1\,\|A\|_2\,\epsilon_k^2/|\sigma_i - \sigma_j| \;\le\; 2C_1\,\epsilon_k^2/c. \tag{13}$$

That is, the computed correction $\widetilde{f}_{ij}^{(k)}$ reproduces the true residual entry $f_{ij}^{(k)}$ to second order at a rate set by the pair's own gap; for an isolated *component, Theorem 4 carries this to quadratic convergence of the residual column.*

***(ii)** – **Group B**: If (i, j) is not in Group A, but satisfies $\widetilde{\sigma}_i + \widetilde{\sigma}_j > c\,\widetilde{\sigma}_{\max}$, then there is a constant $\chi_{\max} \le 3.069$, such that*

$$|\widetilde{f}_{ij}^{(k)}| \;\le\; \tfrac{1}{2}|r_{ij}^{(k)}| + |a_{ij}^{(k)}|, \qquad |a_{ij}^{(k)}| \;<\; \epsilon_k + \chi_{\max}\epsilon_k^2/\big(c(1 - \eta_{\max}\epsilon_k^2)\big). \tag{14}$$

***(iii)** – **Group C**: If (i, j) is neither in Group A nor Group B:*

$$|\widetilde{f}_{ij}^{(k)}| \;\le\; \tfrac{1}{2}|r_{ij}^{(k)}|. \tag{15}$$

Proof. **Step 1 — Group A rate.** Combining inequalities (36) and (42) in [15] yields $|\widetilde{\sigma}_i - \sigma_i| \le \eta_{\max}\|A\|_2\epsilon_k^2$, so

$$|\sigma_i - \sigma_j| \ge |\widetilde{\sigma}_i - \widetilde{\sigma}_j| - 2\eta_{\max}\|A\|_2\epsilon_k^2 \ge c\widetilde{\sigma}_{\max} - 2\eta_{\max}\|A\|_2\epsilon_k^2. \tag{16}$$

The same inequality (42) applied to $i = 1$ and using $\sigma_1 = \|A\|_2$ yields

$$\widetilde{\sigma}_1 \geq \sigma_1 - \eta_{\max}\|A\|_2\epsilon_k^2 = (1 - \eta_{\max}\epsilon_k^2)\|A\|_2. \tag{17}$$

Hence, we have $\widetilde{\sigma}_{\max} \geq \widetilde{\sigma}_1 \geq (1 - \eta_{\max}\epsilon_k^2)\|A\|_2$. Substituting this into (16)

$$|\sigma_i - \sigma_j| \geq c(1 - \eta_{\max}\epsilon_k^2)\|A\|_2 - 2\eta_{\max}\|A\|_2\epsilon_k^2 = \|A\|_2\big(c - (c+2)\eta_{\max}\epsilon_k^2\big). \tag{18}$$

By the theorem assumption (12) we have

$$(c+2)\eta_{\max}\epsilon_k^2 \leq (c+2)c/(c+6) \Leftrightarrow c - (c+2)\eta_{\max}\epsilon_k^2 \;\geq\; c - \frac{(c+2)c}{c+6} \;=\; \frac{4c}{c+6}. \tag{19}$$

Substituting this into (18) and using $c \leq 2$, we bound

$$|\sigma_i - \sigma_j| \geq 4c\|A\|_2/(c+6) \geq c\|A\|_2/2. \tag{20}$$

By rearranging (18) and substituting in (19), we obtain the upper bound on $\|A\|_2$ as

$$\|A\|_2 < |\sigma_i - \sigma_j|/\big(c - (c+2)\eta_{\max}\epsilon_k^2\big) \leq (c+6)|\sigma_i - \sigma_j|/(4c) \tag{21}$$

By multiplying the previous bound by $2\eta_{\max}\epsilon_k^2$ and using the theorem assumption (12), we obtain

$$2\eta_{\max}\|A\|_2\epsilon_k^2 \;\leq\; 2\eta_{\max}\epsilon_k^2 \cdot (c+6)|\sigma_i - \sigma_j|/(4c) \;\leq\; |\sigma_i - \sigma_j|/2. \tag{22}$$

For $c = \sqrt{\varepsilon_{\text{lp}}}$, the theorem assumption (12) implies $\epsilon_k < \frac{1}{60}$, so the constants $\chi_{\max} \leq 3.069$ and $\eta_{\max} \leq 6.573$ from [15, eq. (36)] apply. Bounding the function values $\chi(\epsilon_k) \leq \chi_{\max}$ and $\eta(\epsilon_k) \leq \eta_{\max}$ in Ogita and Aishima's per-entry bound [15, eq. (53)] gives

$$|\widetilde{f}_{ij}^{(k)} - f_{ij}^{(k)}| \;\leq\; \frac{(2\chi_{\max} + 2\eta_{\max}\epsilon_k + \chi_{\max}\eta_{\max}\epsilon_k^2)\,\|A\|_2\,\epsilon_k^2}{|\sigma_i - \sigma_j| - 2\eta_{\max}\|A\|_2\epsilon_k^2}$$

The numerator is bounded from above by $2\chi_{\max} + 2\eta_{\max}\epsilon_k + \chi_{\max}\eta_{\max}\epsilon_k^2 \;\leq\; 6.36$; the denominator is bounded from below by $|\sigma_i - \sigma_j|/2$, which is a consequence of inequality (22). Combining, we obtain $|\widetilde{f}_{ij}^{(k)} - f_{ij}^{(k)}| \;\leq\; C_1\,\|A\|_2\,\epsilon_k^2/|\sigma_i - \sigma_j|$ with $C_1 \;\leq\; 6.36 \cdot 2 \;=\; 12.72 \;\leq\; 13$. Substituting the lower bound $|\sigma_i - \sigma_j| \geq c\|A\|_2/2$, yields the desired bound (13).

Step 2 — Group B bound on a_{ij}. The relation [15, eq. (25)],

$$\Sigma - F^T\Sigma - \Sigma G \;=\; T + \Delta_3, \qquad \|\Delta_3\|_2 \leq \chi_{\max}\,\|A\|_2\,\epsilon_k^2, \tag{23}$$

gives the off-diagonal entries $t_{ij} \;=\; -f_{ji}\,\sigma_j - \sigma_i\,g_{ij} - \Delta_{3,ij}$, $i \neq j$. Taking magnitudes and using the triangle inequality as well as $|f_{ji}|, |g_{ij}| \leq \epsilon_k$ yields $|t_{ij} - t_{ji}| \;\leq\; 2\epsilon_k(\sigma_i + \sigma_j) + 2\chi_{\max}\,\|A\|_2\,\epsilon_k^2$. By using in order, definition of $a_{ij}^{(k)}$, the above bound on $|t_{ij} - t_{ji}|$, and [15, inequality (43)], we obtain

$$|a_{ij}^{(k)}| \;=\; \frac{|t_{ij} - t_{ji}|}{2(\widetilde{\sigma}_i + \widetilde{\sigma}_j)} \;\leq\; \frac{\epsilon_k(\sigma_i + \sigma_j) + \chi_{\max}\,\|A\|_2\,\epsilon_k^2}{\widetilde{\sigma}_i + \widetilde{\sigma}_j} \;\leq\; \epsilon_k + \frac{\chi_{\max}\,\|A\|_2\,\epsilon_k^2}{\widetilde{\sigma}_i + \widetilde{\sigma}_j}. \tag{24}$$

We bound the branch condition using the theorem assumption (12) and inequality (17)

$$\widetilde{\sigma}_i + \widetilde{\sigma}_j > c\,\widetilde{\sigma}_{\max} \geq c\widetilde{\sigma}_1 \geq c(1 - \eta_{\max}\epsilon_k^2)\|A\|_2 \Leftrightarrow \|A\|_2/(\widetilde{\sigma}_i + \widetilde{\sigma}_j) < 1/\big(c(1 - \eta_{\max}\epsilon_k^2)\big).$$

Inserting this into (24), we get our desired bound (14) on $|a_{ij}^{(k)}|$.

Step 3 — Group C bound. The algorithm sets $\widetilde{f}_{ij}^{(k)} = r_{ij}^{(k)}/2$, which gives (15) immediately. □

Theorem 4 (Quadratic convergence of isolated components)**.** *Collecting the computed corrections into $\widetilde{F}^{(k)}$, the multiplicative update reads $\widehat{U}^{(k+1)} = \widehat{U}^{(k)}(I + \widetilde{F}^{(k)})$, and likewise for $\widehat{V}$. By definition, $U = \widehat{U}^{(k)}(I + F^{(k)}) = \widehat{U}^{(k+1)}(I + F^{(k+1)})$, so*

$$F^{(k+1)} = (I + \widetilde{F}^{(k)})^{-1}\big(F^{(k)} - \widetilde{F}^{(k)}\big), \tag{25}$$

Algorithm 4 MxpSVD — mixed-precision SVD via iterative refinement.

Input: $A \in \mathbb{C}^{m\times n}$ with $m \geq n$.
Output: $\widehat{U} \in \mathbb{C}^{m\times m}$, $\widehat{V} \in \mathbb{C}^{n\times n}$, $\widehat{\Sigma} = \mathrm{diag}(\widetilde{\sigma}_1, \ldots, \widetilde{\sigma}_n) \in \mathbb{R}^{m\times n}$ such that $A \approx \widehat{U}\,\widehat{\Sigma}\,\widehat{V}^H$.

1: $A_{\mathrm{lp}} \leftarrow \mathrm{downconvert}(A)$ ▷ Convert A to low precision.
2: $[U_{\mathrm{lp}}, \Sigma_{\mathrm{lp}}, V_{\mathrm{lp}}] \leftarrow \mathrm{SVD}(A_{\mathrm{lp}})$ ▷ Low-precision SVD.
3: $\widehat{U} \leftarrow \mathrm{upconvert}(U_{\mathrm{lp}}), \quad \widehat{V} \leftarrow \mathrm{upconvert}(V_{\mathrm{lp}})$ ▷ Convert U_{lp} and V_{lp} to high precision.
4: **repeat**
5: $\quad [\widehat{U}, \widehat{V}, \widehat{\Sigma}, \omega] \leftarrow \textsc{MxpSVDStep}(A, \widehat{U}, \widehat{V})$ ▷ Cluster-safe refinement.
6: **until** $\omega \leq 16n\varepsilon_{\mathrm{hp}}\widetilde{\sigma}_{\max}$ or ω stagnated.
7: Determine cluster index sets $\mathcal{J}_1, \ldots, \mathcal{J}_{n_{\mathcal{J}}}$ such that
$\quad \widetilde{\sigma}_i, \widetilde{\sigma}_j$ are in the same cluster $\iff |\widetilde{\sigma}_i - \widetilde{\sigma}_j| \leq \omega$ or $|\widetilde{\sigma}_j - \widetilde{\sigma}_i| \leq \sqrt{\varepsilon_{\mathrm{lp}}}\,\widetilde{\sigma}_{\max}$.
8: $[\widehat{U}, \widehat{V}, \widehat{\Sigma}] \leftarrow \textsc{ClusterRR}(A, \widehat{U}, \widehat{V}, \mathcal{J}_k) \quad$ for $k = 1, \ldots, n_{\mathcal{J}}$
9: $\pi \leftarrow \mathrm{argsort}_{\mathrm{desc}}(\widetilde{\sigma}), \quad \widetilde{\sigma} \leftarrow \widetilde{\sigma}_\pi, \quad \widehat{U} \leftarrow \widehat{U}(:, \pi), \quad \widehat{V} \leftarrow \widehat{V}(:, \pi)$ ▷ Restore $\widetilde{\sigma}_1 \geq \cdots \geq \widetilde{\sigma}_n$.

the multiplicative-update relation underlying Theorem 2. Call a component j isolated *if every pair (l, j), $l \neq j$, is well separated (Group A), and assume the basin condition $\|\widetilde{F}^{(k)}\|_2 \leq \rho < 1$. Then for an isolated component j the residual column converges quadratically,*

$$\|F^{(k+1)}_{:,j}\|_2 \;\leq\; \frac{C_1}{1-\rho}\,\|A\|_2\Big(\textstyle\sum_{l\neq j} |\sigma_l - \sigma_j|^{-2}\Big)^{1/2}\,\epsilon_k^2 \;=\; O(\epsilon_k^2), \tag{26}$$

at a rate set by the inverse-square gaps from σ_j alone — with no reference to the global smallest gap $\delta_{\min}$, and $O(\sqrt{n}/c)$ in the worst case (dimension-free when the spectrum is spread). This holds regardless of clusters elsewhere, which inflate $\|\widetilde{F}^{(k)}\|_2$ only within the $O(1)$ factor $(1-\rho)^{-1}$. A component lying in *a cluster is not isolated: its within-cluster column entries are first order, so its column is only $O(\epsilon_k)$, and it is reoriented by the Rayleigh–Ritz step of Algorithm 2 instead.*

Proof. Solving $(I + \widetilde{F}^{(k)})(I + F^{(k+1)}) = I + F^{(k)}$ for $F^{(k+1)}$ gives the identity (25). Its j-th column is $F^{(k+1)}_{:,j} = (I+\widetilde{F}^{(k)})^{-1}(F^{(k)} - \widetilde{F}^{(k)})_{:,j}$, so $\|F^{(k+1)}_{:,j}\|_2 \leq \|(F^{(k)} - \widetilde{F}^{(k)})_{:,j}\|_2/(1-\|\widetilde{F}^{(k)}\|_2)$ via $\|(I+\widetilde{F}^{(k)})^{-1}\|_2 \leq (1-\|\widetilde{F}^{(k)}\|_2)^{-1}$. For isolated j every off-diagonal entry obeys the Group A bound $|f^{(k)}_{lj} - \widetilde{f}^{(k)}_{lj}| \leq C_1\|A\|_2\epsilon_k^2/|\sigma_l - \sigma_j|$ of (13), while the diagonal entry is $O(\epsilon_k^2)$ in real arithmetic; squaring and summing the column gives $\|(F^{(k)} - \widetilde{F}^{(k)})_{:,j}\|_2 \leq C_1\|A\|_2\epsilon_k^2(\sum_{l\neq j}|\sigma_l - \sigma_j|^{-2})^{1/2}$, and $\|\widetilde{F}^{(k)}\|_2 \leq \rho$ yields (26). The worst-case constant follows from $|\sigma_l - \sigma_j| \geq c\|A\|_2/2$ (Group A, (20)) in each of the $n-1$ terms. □

Together, Theorem 3 and Theorem 4 replace the global admissibility threshold (11) of Theorem 2 — whose rate $18m\|A\|_2/\delta_{\min}$ couples the dimension to the global smallest gap and so collapses for any clustered spectrum ($\delta_{\min} \to 0$) — by a criterion referencing neither m nor any global gap, only $c = \sqrt{\varepsilon_{\mathrm{lp}}}$ and the universal constant $\eta_{\max}$. The hypothesis can therefore hold where (11) is unsatisfiable: once a cluster is present the global theorem certifies no pair, whereas every isolated component is refined quadratically at its own gap and the clustered pairs are cleaned up by the Rayleigh–Ritz step. This is the behavior seen across all twelve spectra of Section 5, including the multiplicity-bearing and rank-deficient patterns with $\delta_{\min} = 0$.

4.5 Mixed-Precision SVD Algorithm

Algorithm 4 assembles the full mixed-precision SVD. The algorithm has four phases. First, we compute a low-precision SVD approximation (line 2). Second, the cluster-safe refinement loop is run until the well-separated singular values have converged (line 5). Third, the cluster refinement phase processes the remaining singular values. At this point the diagonal of T is already real and non-negative: the in-step diagonal phase correction (line 4 of Algorithm 3) renders it real for the resolved singular values, and the Rayleigh–Ritz sub-SVD does so for the clustered ones, so no separate phase-correction pass is required. Fourth, the per-element $\widetilde{\sigma}$ updates in the refinement loop and in cluster refinement preserve the column order of the low-precision SVD only up to $O(\varepsilon_{\mathrm{lp}})$, and cluster refinement sorts only within cluster blocks. We therefore sort the singular values into non-increasing order and permute the singular vector columns accordingly (line 9).

5 Numerical Experiments

We present numerical results to demonstrate the correctness and effectiveness of our mixed-precision SVD algorithm. All numerical experiments and benchmarks for RefSVD, MxpSVD (Algorithm 4), and `cusolverDnXgesvdp` – which is the fastest general-purpose SVD in the `cuSolverDn` library – were run on the NVIDIA RTX PRO 6000 Blackwell Workstation Edition (WSE) GPU with CUDA 13.2, with one exception: The implementation of `cusolverDnXgesvdp` used both inside MxpSVD and as our reference state-of-the-art FP64 SVD is a pre-release version to appear in CUDA 13.3 Update 1, which includes accuracy fixes for ill-conditioned and rank-deficient matrices. Our measurements are conducted in two modes: 1) using native FP64 arithmetic and 2) using tensor-core-emulated FP64 GEMM using the Automatic Dynamic Precision framework in cuBLAS based on the Ozaki-I algorithm [18].

Test matrices We test our algorithm on a total of twelve types of test matrices. The matrices with a prescribed singular value distribution synthesize $A = U\Sigma V^H$ via two matrix-matrix multiplications with random unitary matrices U and V. Due to rounding errors during the GEMMs, multiplicities can be expected to become clusters.

(1) diagonally dominant: The matrix A is filled such that $\mathrm{Re}(a_{ij}) \sim \mathcal{N}(0,1)$ and $\mathrm{Im}(a_{ij}) \sim \mathcal{N}(0,1)$. The diagonal is set $a_{ii} \leftarrow 5 + \sum_{j \neq i} |a_{ij}|$, rendering A diagonally dominant.

(2) polynomial decay: $\sigma_i = i^{-2}$, $i = 1, \ldots, n$.

(3) two plateaus: $\sigma_i = 10^{-4} + (1 + e^{i-10})^{-1}$, $i = 1, \ldots, n$.

(4) slow decay: $\sigma_i = \big(1 - (i-1)/(n-1)\big)^{1/10}$, $i = 1, \ldots, n$.

(5) uniformly spaced: $\sigma_i = n - i + 1$, $i = 1, \ldots, n$.

(6) column-normalized: pattern (1) with each column scaled to unit norm.

(7) pattern (6) with $\max(1, \lceil n/10 \rceil)$ columns overwritten by zeros at randomly chosen positions.

(8) geometrically distributed: $\sigma_i = \varepsilon_{\mathrm{hp}}^{(i-1)/(n-1)}$, $i = 1, \ldots, n$.

(9) rank-deficient: $A = LR$, $L \in \mathbb{C}^{n \times r}$, $R \in \mathbb{C}^{r \times n}$ random, $r = \lfloor n/2 \rfloor$.

(10) multi-cluster: pattern (5) modified by three multiplicity blocks — a head cluster $\sigma_1 = \sigma_2 = \sigma_3 = n$, a center cluster of multiplicity 5 around the median index with value $\lfloor n/2 \rfloor$, and a tail cluster $\sigma_{n-2} = \sigma_{n-1} = \sigma_n = 1$.

(11) one large: $\sigma_1 = 1$ and $\sigma_i = \varepsilon_{\mathrm{hp}}$ for $i = 2, \ldots, n$.

(12) one small: $\sigma_i = 1$ for $i = 1, \ldots, n-1$ and $\sigma_n = \varepsilon_{\mathrm{hp}}$.

We validate convergence of our mixed-precision algorithm on the twelve test matrices described above; Table 1 shows the results. Since the accuracy of well-separated singular values depends on the number of iterations, we report the iteration count that yields a relative error $(\|A - \hat{U}\hat{\Sigma}\hat{V}^H\|_F)/(N\,\|A\|_F)$ of less than 10^{-16}. Our algorithm attains an accuracy comparable to a high-precision SVD in all test cases. For comparison, we also report whether the original Ogita–Aishima RefSVD algorithms can solve the case. Despite not having any clusters, patterns (2) and (8) cannot be refined by RefSVD because some singular values are below the resolution of the low-precision SVD preconditioner.

Next we study the convergence behavior of our cluster-safe refinement loop. First, we look at the test cases (11) and (12). Ogita and Aishima [15, Fig. 1] use these tests as examples where the original algorithm RefSVD (Algorithm 1) does not converge. Figure 1 shows the evolution of the corrections and the diagonality measure for RefSVD and the cluster-safe MxpSVDStep. Since both algorithms start from the same single-precision SVD, $\|\mathrm{offdiag}(T)\|_F/\|A\|_F$ coincides at the first iteration. The behavior of MxpSVDStep is consistent with quadratic convergence, reaching working precision after three iterations. RefSVD, by contrast, exits the basin of attraction at the first iteration.

Table 1: Accuracy of MxpSVD (Algorithm 4) on twelve test matrices with $m = n = 2048$ and real arithmetic, computed with native FP64 GEMM and with tensor-core-emulated FP64 GEMM (Ozaki-I, via the cuBLAS Automatic Dynamic Precision framework). The two modes attain the same accuracy up to floating-point rounding. Columns report $\eta = \|A - \hat{U}\hat{\Sigma}\hat{V}^H\|_F/(n\,\|A\|_F)$, $\rho_U = \|I - \hat{U}^H\hat{U}\|_F/m$, and $\rho_V = \|I - \hat{V}^H\hat{V}\|_F/n$.

			native FP64			emulated FP64		
test matrix	RefSVD	iter.	η	ρ_U	ρ_V	η	ρ_U	ρ_V
(1) diagonally dominant	converges	4	2.44×10^{-17}	2.28×10^{-17}	2.28×10^{-17}	1.13×10^{-17}	1.67×10^{-17}	1.62×10^{-17}
(2) polynomial decay	diverges	5	1.78×10^{-18}	2.41×10^{-16}	2.44×10^{-16}	6.17×10^{-19}	2.33×10^{-16}	2.31×10^{-16}
(3) two plateaus	diverges	3	1.29×10^{-18}	9.98×10^{-17}	8.79×10^{-17}	5.83×10^{-19}	9.69×10^{-17}	7.84×10^{-17}
(4) slow decay	converges	3	1.39×10^{-18}	2.30×10^{-17}	2.32×10^{-17}	5.34×10^{-19}	1.65×10^{-17}	1.68×10^{-17}
(5) uniformly spaced	converges	3	1.39×10^{-18}	2.29×10^{-17}	2.30×10^{-17}	5.54×10^{-19}	1.65×10^{-17}	1.65×10^{-17}
(6) column-normalized diag. dom.	diverges	4	4.14×10^{-17}	4.95×10^{-17}	4.88×10^{-17}	3.81×10^{-17}	4.89×10^{-17}	4.68×10^{-17}
(7) pattern (6) with zero columns	diverges	5	3.49×10^{-17}	3.49×10^{-17}	3.46×10^{-17}	3.25×10^{-17}	3.40×10^{-17}	3.31×10^{-17}
(8) geometrically distributed	diverges	4	1.18×10^{-17}	1.11×10^{-16}	9.23×10^{-17}	5.25×10^{-18}	1.07×10^{-16}	8.84×10^{-17}
(9) rank-deficient	diverges	5	2.98×10^{-18}	1.03×10^{-16}	7.42×10^{-17}	5.58×10^{-18}	9.86×10^{-17}	6.94×10^{-17}
(10) multi-cluster	diverges	2	1.39×10^{-18}	2.29×10^{-17}	2.26×10^{-17}	5.62×10^{-19}	1.66×10^{-17}	1.68×10^{-17}
(11) one large	diverges	2	1.26×10^{-18}	1.34×10^{-16}	1.12×10^{-16}	5.22×10^{-19}	1.18×10^{-16}	1.03×10^{-16}
(12) one small	diverges	2	2.34×10^{-18}	8.29×10^{-17}	8.07×10^{-17}	1.78×10^{-18}	6.65×10^{-17}	6.66×10^{-17}

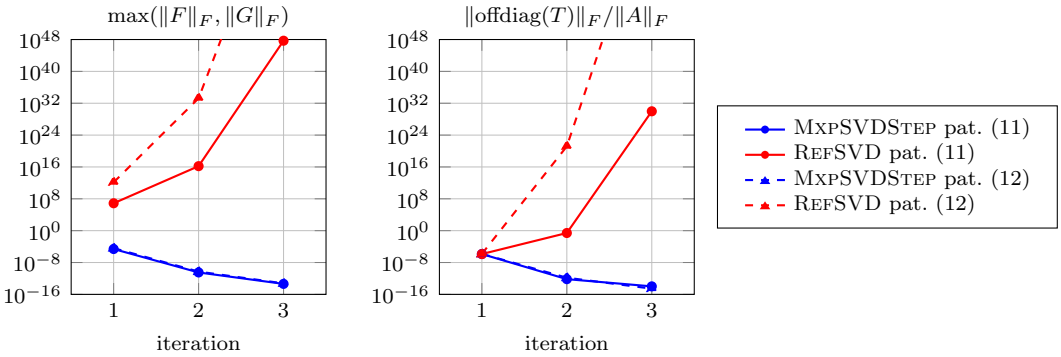

Figure 1: Comparison of three iterations of MxpSVDStep (Algorithm 3) and RefSVD (Algorithm 1) on patterns (11) and (12). The experiment uses real arithmetic and $n = 2000$.

Next we study the rank-deficient case (9). Figure 2 shows the results. The iterations of MxpSVDStep drive the correction matrices $\|F\|_F$ and $\|G\|_F$ linearly to around 10^{-11}, where they stagnate. The offdiagonal of $T = \hat{U}^H A\hat{V}$ does not improve. While its antisymmetric contribution improves slightly, its symmetric contribution does not. For this test case, MxpSVDStep is dominated by the orthogonality-correction-only fallback (Algorithm 3, line 13), which explains the stagnation: Both fallbacks of Algorithm 3 leave the symmetric contribution untouched. The correctness of the algorithm is due to the cluster postprocessing, as shown in the plot as the 'final' diagonality measure.

Performance Experiments We benchmark MxpSVD (Algorithm 4) against FP64 `cusolverDnXgesvdp` on the NVIDIA RTX PRO 6000 Blackwell WSE GPU, which has a native peak performance of 125 TFlops (FP32) / 1.95 TFlops (FP64). The mixed-precision pipeline uses `cusolverDnXgesvdp` in single precision as the low-precision preconditioner and a hard cap of $\nu = 5$ refinement iterations, with the actual loop exiting early when the cluster threshold ω from (2) has stopped contracting (*cf.* lines 4–6 of Algorithm 4). Matrices with well-separated singular values such as pattern (4), (5) typically converge in 3–4 inner steps.

For reporting benchmark results, we select two patterns that bracket the cluster behavior of MxpSVD. Pattern (4) has $\sigma_i = (1 - (i-1)/(n-1))^{1/10}$, whose top half is nearly flat and produces a single cluster of size $\approx n/2$ (e.g. 4399 of 8192 singular values at $n = 8192$), so Phase 2 runs `cusolverDnXgesvdp` on a sub-matrix of nearly half the original dimension; this exercises the full cluster refinement pass. Pattern (5) has $\sigma_i = n - i + 1$, so every adjacent gap clears the per-pair Wedin threshold of Algorithm 3, line 6; the cluster scan returns no clusters and Phase 2's sub-SVD does not execute, isolating the cost of Phase 1. Table 2 reports the results, and Figure 3 shows the speedup of MxpSVD relative to `cusolverDnXgesvdp` on these

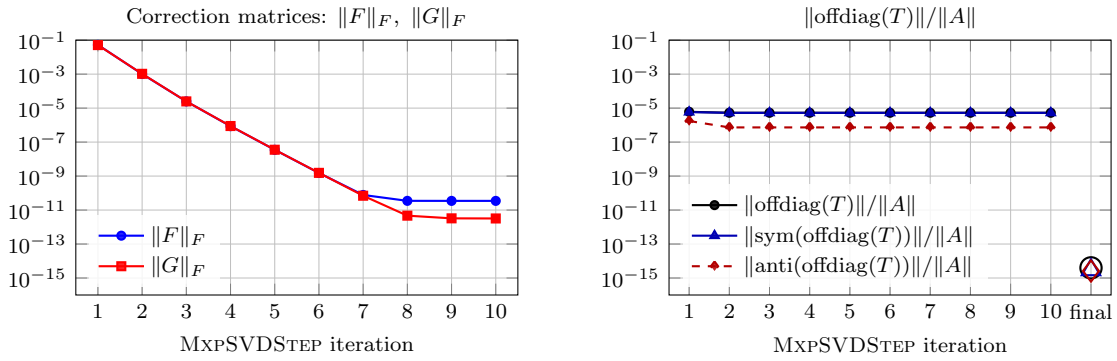

Figure 2: MxpSVD on the rank-deficient test case (9) with $n = 2000$, complex precision, and `cusolverDnXgesvdp` as SVD solver. The cluster refinement phase recovers the expected residual, in the plot marked as 'final'.

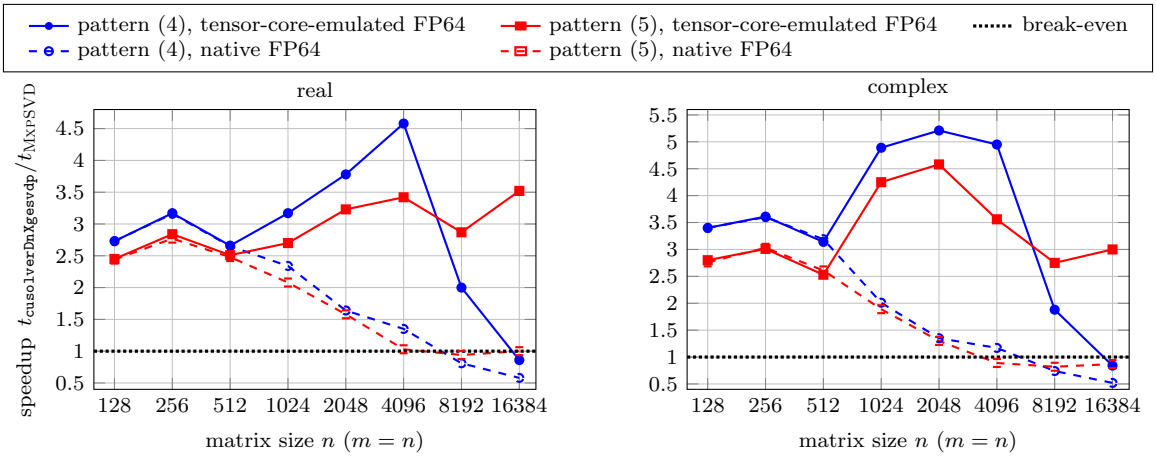

Figure 3: Speedup of MxpSVD (Algorithm 4) over `cusolverDnXgesvdp` on test matrices (4) and (5) on an NVIDIA RTX PRO 6000 Blackwell WSE. Solid lines show the speedup when both algorithms use tensor-core-emulated FP64; dashed lines show the speedup when both use native FP64.

two patterns.

MxpSVD replaces a single native-FP64 `cusolverDnXgesvdp` call with a single-precision preconditioner, the GEMM-dominated Phase 1 refinement loop, and—for pattern (4)—the Phase 2 cluster sub-SVD. Native FP64 runs at a small fraction of the single-precision and tensor-core throughput on this consumer-class GPU, so at small n the cheap preconditioner dominates and MxpSVD is roughly 2.5–3× faster. The refinement GEMMs and the Phase 2 sub-SVD are themselves $O(n^3)$ and, in native FP64, run at the same low throughput as `cusolverDnXgesvdp`; as n grows they dominate the runtime and this head start is amortized away. The native-FP64 speedup therefore decreases with n, crossing break-even near $n = 8192$ for pattern (4) and settling around unity for pattern (5).

Emulation reverses this trend because it accelerates exactly the GEMM-bound work that dominates MxpSVD. Phase 1 is almost entirely GEMM, which Ozaki-I emulation maps onto the FP64 tensor cores at several times the native rate (Figure 4); `cusolverDnXgesvdp` spends more of its time in non-GEMM kernels, so MxpSVD captures more of the emulation speedup. For pattern (5)—whose cluster scan finds nothing, so Phase 2 is empty and the runtime is essentially all Phase 1—this applies to the dominant $O(n^3)$ term and is sustained across moderate-to-large n: the real speedup stays in the 2.5–3.5× range and reaches 3.5× at $n = 16384$, while complex arithmetic is larger, peaking near 4.6× around $n \approx 2048$ and holding 3.0× at $n = 16384$. Pattern (4) attains the largest emulated speedups of all in the mid-range—up to 4.6× in real and 5.2× in complex arithmetic around $n \approx 2048$–4096—but its Phase 2 sub-SVD is a native-FP64

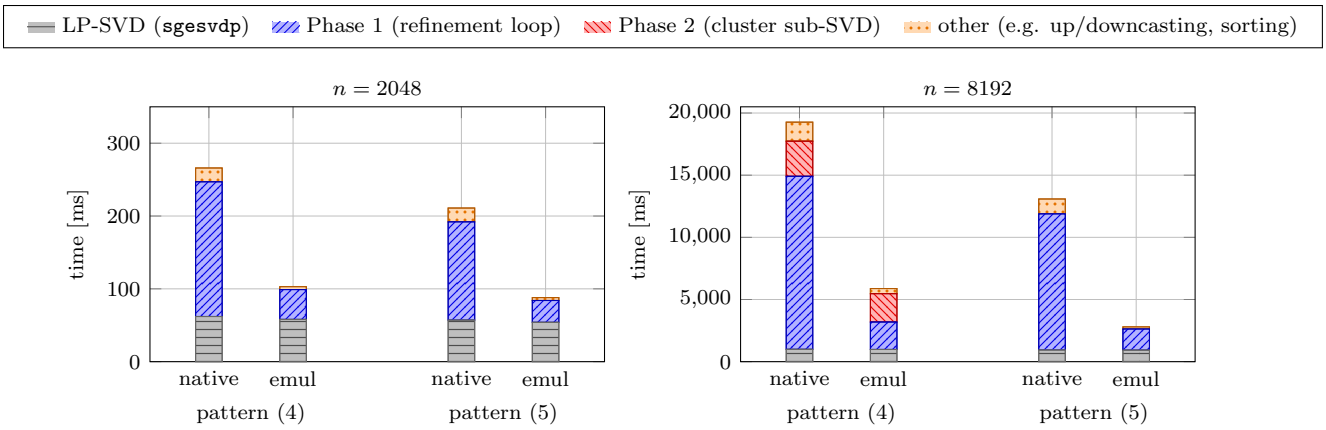

Figure 4: Time breakdown of MxpSVD (Algorithm 4), real arithmetic, with native and emulated FP64 on NVIDIA RTX PRO 6000 Blackwell WSE.

Table 2: Timings in seconds of `cusolverDnXgesvdp` (column g) and MxpSVD (column r, Algorithm 4) on an NVIDIA RTX PRO 6000 Blackwell WSE.

	real								complex							
	native FP64				emulated FP64				native FP64				emulated FP64			
	pattern (4)		pattern (5)		pattern (4)		pattern (5)		pattern (4)		pattern (5)		pattern (4)		pattern (5)	
n	g	r	g	r	g	r	g	r	g	r	g	r	g	r	g	r
128	0.010	0.004	0.008	0.003	0.010	0.004	0.008	0.003	0.021	0.006	0.015	0.006	0.021	0.006	0.015	0.006
256	0.019	0.006	0.014	0.005	0.019	0.006	0.014	0.005	0.035	0.010	0.027	0.009	0.035	0.010	0.027	0.009
512	0.042	0.016	0.031	0.013	0.042	0.016	0.031	0.012	0.092	0.029	0.069	0.026	0.090	0.029	0.067	0.026
1024	0.118	0.051	0.087	0.042	0.117	0.037	0.087	0.032	0.294	0.146	0.222	0.118	0.282	0.058	0.207	0.049
2048	0.422	0.258	0.320	0.204	0.378	0.100	0.274	0.084	1.24	0.914	0.935	0.721	1.17	0.225	0.861	0.188
4096	2.27	1.68	1.74	1.68	1.89	0.413	1.38	0.402	8.17	6.99	6.24	6.99	6.74	1.36	4.80	1.35
8192	15.6	19.3	12.2	13.0	11.7	5.84	7.74	2.70	57.5	78.1	44.4	54.3	41.9	22.3	29.0	10.5
16384	115	197	99.3	99.8	80.9	93.7	72.4	20.6	433	830	376	430	304	361	270	90.1

`cusolverDnXgesvdp` on an $\approx n/2$ submatrix whose non-GEMM kernels stay in native precision and are not emulated; as n grows this unaccelerated phase becomes the bottleneck, so the pattern (4) speedup turns over above $n \approx 4096$ and falls to break-even at $n = 16384$. Throughout, complex arithmetic yields the larger emulated speedups, owing to the higher arithmetic intensity of complex GEMM.

Figure 4 shows a cost decomposition of MxpSVD (Algorithm 4). The refinement loop (Phase 1) is dominated with over 99% by GEMMs. Consequently, Phase 1 can benefit substantially from emulation. Phase 2 is empty for pattern (5) because the cluster scan finds no clusters; Phase 2 on pattern (4) is dominated by `cusolverDnXgesvdp` acting on the approximately $n/2$ cluster in the top half of the spectrum, and benefits only mildly from emulation because its non-GEMM kernels stay more in native FP64.

These numbers are specific to a GPU without dedicated fast FP64. On data-center parts like NVIDIA B200 – peak native performance of 75 TFlops (FP32) / 37 TFlops (FP64) – the two are essentially on par: MxpSVD is marginally faster than `cusolverDnXgesvdp` in real arithmetic (1.0×) and marginally slower in complex (0.9×), as shown in Table 3. The pronounced advantage seen on the consumer-class WSE (3.5× real, 3.0× complex) thus narrows toward break-even where fast native FP64 is available. This balance is expected to shift in favor of MxpSVD as emulation improves with the Ozaki-II scheme.

6 Conclusion

This work extends the Ogita–Aishima refinement for SVD to cluster handling. The resulting algorithm has two stages: The first is a cluster-safe refinement loop that detects clusters using a Wedin-union criterion

Table 3: Timings in seconds of `cusolverDnXgesvdp` (column g) and MxpSVD (column r) on NVIDIA RTX PRO 6000 Blackwell WSE and B200 for $n = m = 16384$, pattern (5), emulated FP64. Column g/r is the MxpSVD speedup over `cusolverDnXgesvdp` on the same GPU.

	real			complex		
GPU	g	r	g/r	g	r	g/r
NVIDIA RTX PRO 6000 Blackwell WSE	72.4 s	20.6 s	3.5 ×	269.9 s	90.1 s	3.0 ×
NVIDIA B200	11.1 s	10.7 s	1.0 ×	36.1 s	39.5 s	0.9 ×

and lets clustered singular values converge to a safe state; the second refines those clusters. This pipeline maximizes the time spent in the first stage, which is dominated by GEMMs and runs efficiently on GPUs. We have demonstrated the correctness and efficiency of our GPU implementation across a wide range of matrices. On an NVIDIA RTX PRO 6000 Blackwell WSE, we observe speedups between 2× and 5× over FP64 `cusolverDnXgesvdp`. The result is a library-grade SVD implementation that is robust regardless of matrix size and singular value distribution, and achieves full FP64 accuracy while taking advantage of Ozaki-I FP64 emulation—an advantage that will continue to grow as Ozaki-II becomes available on current and future GPUs.

References

[1] Ahmad Abdelfattah, Hartwig Anzt, Erik G. Boman, Erin Carson, Terry Cojean, Jack Dongarra, Alyson Fox, Mark Gates, Nicholas J. Higham, Xiaoye S. Li, Jennifer Loe, Piotr Luszczek, Srikara Pranesh, Sivasankaran Rajamanickam, Tobias Ribizel, Barry F. Smith, Kasia Swirydowicz, Stephen Thomas, Stanimire Tomov, Yaohung M. Tsai, and Ulrike Meier Yang. A survey of numerical linear algebra methods utilizing mixed-precision arithmetic. *The International Journal of High Performance Computing Applications*, 35(4):344–369, 2021.

[2] Zvonimir Bujanović, Daniel Kressner, and Christian Schröder. Iterative refinement of schur decompositions. *Numerical algorithms*, 92(1):247–267, 2023.

[3] Erin Carson, Yuxin Ma, and Meiyue Shao. Mixed precision thin svd algorithms based on the gram matrix. *arXiv preprint arXiv:2603.11953*, 2026.

[4] Jack Dongarra, John Gunnels, Harun Bayraktar, Azzam Haidar, and Dan Ernst. Hardware trends impacting floating-point computations in scientific applications, Nov. 2024.

[5] Zlatko Drmač and Krešimir Veselić. New fast and accurate jacobi svd algorithm. ii. *SIAM Journal on matrix analysis and applications*, 29(4):1343–1362, 2008.

[6] Weiguo Gao, Yuxin Ma, and Meiyue Shao. A mixed precision Jacobi SVD algorithm. *ACM Transactions on Mathematical Software*, 51(1), Apr. 2025.

[7] Azzam Haidar, Stanimire Tomov, Jack Dongarra, and Nicholas J. Higham. Harnessing gpu tensor cores for fast fp16 arithmetic to speed up mixed-precision iterative refinement solvers. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 603–613, Dallas, TX, USA, Nov. 2018. IEEE.

[8] Nicholas Higham, Françoise Tisseur, Marcus Webb, and Zhengbo Zhou. Computing accurate eigenvalues using a mixed-precision jacobi algorithm. *SIAM Journal on Matrix Analysis and Applications*, 46:2423–2448, 10 2025.

[9] Nicholas J. Higham and Theo Mary. Mixed precision algorithms in numerical linear algebra. *Acta Numerica*, 31:347–414, 2022.

[10] Toshiyuki Imamura, Susumu Yamada, and Masahiko Machida. Development of a high performance eigensolver on the petascale next generation supercomputer system. *Progress in Nuclear Science and Technology*, 2:643–650, 2011.

[11] Aditya Kashi, Hao Lu, Wesley Brewer, David Rogers, Michael Matheson, Mallikarjun Shankar, and Feiyi Wang. Mixed-precision numerics in scientific applications: survey and perspectives. *The Journal of Supercomputing*, 82(5):287, 2026.

[12] Daichi Mukunoki, Katsuhisa Ozaki, Takeshi Ogita, and Toshiyuki Imamura. Dgemm using tensor cores, and its accurate and reproducible versions. In *High Performance Computing (ISC High Performance 2020)*, pages 230–248. Springer International Publishing, 2020.

[13] Takeshi Ogita and Kensuke Aishima. Iterative refinement for symmetric eigenvalue decomposition. *Japan Journal of Industrial and Applied Mathematics*, 35(3):1007–1035, May 2018.

[14] Takeshi Ogita and Kensuke Aishima. Iterative refinement for symmetric eigenvalue decomposition II: clustered eigenvalues. *Japan Journal of Industrial and Applied Mathematics*, 36:435–459, Feb. 2019.

[15] Takeshi Ogita and Kensuke Aishima. Iterative refinement for singular value decomposition based on matrix multiplication. *Journal of Computational and Applied Mathematics*, 369:112512, 2020.

[16] Hiroyuki Ootomo, Katsuhisa Ozaki, and Rio Yokota. Dgemm on integer matrix multiplication unit. *The International Journal of High Performance Computing Applications*, 38(4):297–313, Mar. 2024.

[17] Katsuhisa Ozaki, Yuki Uchino, and Toshiyuki Imamura. Ozaki scheme ii: A gemm-oriented emulation of floating-point matrix multiplication using an integer modular technique. *arXiv preprint*, 2025.

[18] Angelika Schwarz, Anton Anders, Cole Brower, Harun Bayraktar, John Gunnels, Kate Clark, RuQing Xu, Samuel Rodriguez, Sebastien Cayrols, Paweł Tabaszewski, and Victor Podlozhnyuk. Guaranteed dgemm accuracy while using reduced precision tensor cores through extensions of the ozaki scheme, 11 2025.

[19] Takeshi Terao. Iterative refinement for diagonalizable non-Hermitian eigendecompositions, 2026.

[20] Takeshi Terao and Katsuhisa Ozaki. Forward error-oriented iterative refinement for eigenvectors of a real symmetric matrix. *arXiv preprint arXiv:2602.19090*, 2026.

[21] Yaohung M. Tsai, Piotr Luszczek, and Jack J. Dongarra. Mixed-precision algorithm for finding selected eigenvalues and eigenvectors of symmetric and hermitian matrices. In *2022 IEEE/ACM Workshop on Latest Advances in Scalable Algorithms for Large-Scale Heterogeneous Systems (ScalAH)*, pages 43–50, Dallas, TX, USA, 2022. IEEE.

[22] Yuki Uchino and Toshiyuki Imamura. High-performance eigensolver combining eigenexa and iterative refinement. In *SC24-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1703–1712, Atlanta, GA, USA, 2024. IEEE.

[23] Yuki Uchino, Katsuhisa Ozaki, and Toshiyuki Imamura. Performance enhancement of the ozaki scheme on integer matrix multiplication unit. *International Journal of High Performance Computing Applications*, 39(3):462–476, Jan. 2025.

[24] Yuki Uchino, Takeshi Terao, and Katsuhisa Ozaki. Acceleration of iterative refinement for singular value decomposition. *Numerical Algorithms*, 95(2):979–1009, 2024.

[25] Per-Åke Wedin. Perturbation bounds in connection with singular value decomposition. *BIT Numerical Mathematics*, 12(1):99–111, 1972.

[26] Zhiyuan Zhang and Zheng-Jian Bai. A mixed precision preconditioned jacobi method for the symmetric eigenvalue problem. *arXiv preprint*, 2025.

[27] Zhengbo Zhou, Françoise Tisseur, and Marcus Webb. Computing accurate singular values using a mixed-precision one-sided jacobi algorithm. *arXiv preprint arXiv:2602.18134*, 2026.