DOUBLY ROBUST PRINCIPAL COMPONENT ANALYSIS THROUGH ROBUST LATENT SUBSPACE WITH BOUNDED NOISE

Anonymous authors
Paper under double-blind review

ABSTRACT

Principal component analysis (PCA) is a fundamental technique for dimensionality reduction and the recovery of underlying data structure, with broad applicability across real-world scenarios. To enhance its robustness against noise and outliers, numerous robust PCA (RPCA) methods have been proposed. However, most existing approaches focus predominantly on the input space, overlooking the potential presence of noise in the latent space. Moreover, their optimization typically relies on alternating updates across multiple matrix variables, such as the alternating direction method of multipliers (ADMM), which is prone to premature convergence and suboptimal solutions. To overcome these limitations, we introduce a novel doubly robust PCA approach, termed DRPCA, which jointly addresses robustness in both the input and latent spaces. By deriving upper bounds on the DRPCA objective, we further establish a theoretical connection between model robustness and sparsity-inducing regularization. To efficiently solve the resulting models, we propose a hybrid alternating optimization algorithm grounded in parameter-free exact solvers, which decomposes the original DRPCA problem into tractable subproblems, integrating rapid matrix-wise updates with element-wise updates that admit closed-form, globally optimal solutions. A prescreening-based acceleration strategy is additionally incorporated to expedite the optimization process without compromising solution quality. Extensive experiments on benchmark datasets of varying scales demonstrate the superior performance of DRPCA over state-of-the-art RPCA methods. Notably, the proposed hybrid optimization algorithm consistently surpasses its matrix-wise-only or element-wise-only counterparts, yielding better solutions and higher-quality reconstruction.

1 INTRODUCTION

High-dimensional structured data are ubiquitous in applications, yet noise often grows with dimensionality, rendering direct processing inefficient and unreliable. To address this challenge, dimensionality reduction techniques such as principal component analysis (PCA) (Jolliffe & Cadima, 2016) are widely employed to extract low-dimensional structures while preserving essential information. However, classical PCA, which relies on the ℓ_2- or Frobenius norm, is highly sensitive to outliers and corruptions, resulting in distorted subspace recovery. To overcome this limitation, robust PCA (RPCA) was introduced and has since attracted considerable attention across a wide range of domains, including face recognition (Meng et al., 2021; Kim et al., 2022), video background subtraction (Li et al., 2023; Alawode & Javed, 2025), feature identification (Xue et al., 2019; Liang et al., 2023), and audio processing (Huang et al., 2012; Ikemiya et al., 2016).

To enhance robustness, existing RPCA methods have been improved from various perspectives, including robust loss functions (Ke & Kanade, 2005; Ding et al., 2006; Zhang & Ding, 2013; Nie et al., 2014; 2021; Gao et al., 2023; Feng et al., 2024), structured sparsity modeling (Wright et al., 2009; Xu et al., 2010; Zhang & Tong, 2019; Han et al., 2024; Xu et al., 2024), deep unfolding techniques (Cai et al., 2021; Joukovsky et al., 2023), adaptive rank estimation (Xu et al., 2021; Fang et al., 2023) and data selection (Gao et al., 2024; Wang et al., 2025) strategies. While these approaches contribute to notable performance gains, they often overlook the need for more comprehensive modeling of corruption. In fact, corruption can arise not only in the input (observation) space but also in the latent

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.

(representation) space, yet most of these methods focus solely on the former, neglecting corruptions in the latter, which in turn limits the reconstruction quality of high-dimensional data. Separately, PCA-based models commonly rely on matrix-wise alternating optimization to solve the underlying multivariate problems, owing to its generality and flexibility (Lin et al., 2010; Candès et al., 2011; Zhang & Ding, 2013; Xu et al., 2021). Representative solvers include the widely adopted augmented Lagrangian method (ALM) and alternating direction method of multipliers (ADMM). However, these alternating optimization methods remain prone to premature convergence and suboptimal solutions.

To mitigate the impact of latent-space noise on reconstruction quality, we propose a novel *doubly robust PCA (DRPCA)* approach that jointly enhances robustness in both the input and latent spaces, where the latent noise is assumed to be bounded. By deriving the upper bounds of the DRPCA model under this assumption, we theoretically establish the connection between model robustness and sparsity-inducing regularization, leading to two variants (*i.e.*, *D_1RPCA* and *D_{21}RPCA*) that impose different structured constraints on latent-space noise. Moving beyond conventional matrix alternating update frameworks, we design a *hybrid alternating optimization algorithm* built upon *parameter-free exact solvers* that integrate matrix-wise and element-wise updates. This hybrid scheme not only captures the global low-rank structure but also guarantees the global optimum for each element of the principal directions and components, thereby eliminating the reliance on hyperparameter tuning and alleviating premature convergence. We further introduce an *prescreening-based acceleration strategy* that greatly boosts the computational efficiency of the proposed optimization algorithm.

Overall, the primary contributions of this work can be summarized as follows:

- We propose a doubly robust PCA (DRPCA) approach that jointly enhances robustness in both the input and latent spaces, an often overlooked source of corruption, leading to improved representation and reconstruction quality.
- We reveal a theoretical link between model robustness and sparsity-inducing regularization by rigorously deriving upper bounds for the DRPCA model, giving rise to two structured variants, *i.e.*, D_1RPCA and D_{21}RPCA.
- We develop a hybrid alternating optimization algorithm that integrates matrix-wise and element-wise updates to solve these two variants effectively, further incorporating a prescreening-based acceleration strategy to significantly reduce the computational cost.
- Extensive experiments show that DRPCA consistently outperforms state-of-the-art RPCA methods on a variety of benchmark datasets, while the proposed optimization algorithm attains superior solutions and reconstruction quality compared to standalone matrix-wise alternating approaches.

2 PRELIMINARIES

Notations and Definitions. Throughout this paper, scalars, vectors, and matrices are denoted by lowercase (or uppercase) letters, boldface lowercase letters, and boldface uppercase letters, respectively. Given a data matrix $\boldsymbol{X} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n) \in \mathbb{R}^{d\times n}$, where d denotes the data dimensionality and n the number of samples, let $\boldsymbol{x}_j \in \mathbb{R}^d$ denote the j-th column (*i.e.*, the j-th data sample). The Frobenius norm, ℓ_1-norm, and ℓ_{21}-norm of $\boldsymbol{X}$ are defined as follows: $\|\boldsymbol{X}\|_F = (\sum_{j=1}^n \sum_{i=1}^d X_{ij}^2)^{1/2}$, $\|\boldsymbol{X}\|_1 = \sum_{j=1}^n \|\boldsymbol{x}_j\|_1 = \sum_{j=1}^n \sum_{i=1}^d |X_{ij}|$, $\|\boldsymbol{X}\|_{21} = \sum_{j=1}^n \|\boldsymbol{x}_j\|_2 = \sum_{j=1}^n (\sum_{i=1}^d X_{ij}^2)^{1/2}$.

PCA and Robust PCA. PCA seeks a low-rank approximation $\boldsymbol{U}\boldsymbol{V}^T$ that minimizes the reconstruction error $\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_F^2$, equivalently maximizing the projected variance $\mathrm{Tr}(\boldsymbol{U}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{U})$ subject to $\boldsymbol{U}^T\boldsymbol{U} = I$, where $\boldsymbol{U} \in \mathbb{R}^{d\times k}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times k}$ denote the principal directions and components. Since PCA relies on the Frobenius norm, it is highly sensitive to noise and outliers, motivating robust PCA methods that model input-space corruption under explicit structural assumptions. For instance, under Laplacian noise, ℓ_1-PCA directly replaces the Frobenius norm with an ℓ_1-norm, *i.e.*, $\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_1$. Alternatively, a common strategy introduces an auxiliary error variable $\boldsymbol{E}$ to explicitly capture sparse deviations while factorizing the clean data as $\boldsymbol{U}\boldsymbol{V}^T$, yielding objectives of the form $\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T + \boldsymbol{E}\|_F^2 + \lambda\|\boldsymbol{E}\|_1$. In contrast to these factorization-based formulations, RPCA instead recovers a low-rank matrix $\boldsymbol{A}$ from a corrupted observation $\boldsymbol{X}$ containing unknown sparse errors $\boldsymbol{E}$, by relaxing the rank constraint via the nuclear norm, *i.e.*, minimizing $\|\boldsymbol{A}\|_* + \lambda\|\boldsymbol{E}\|_1$ subject to $\boldsymbol{X} = \boldsymbol{A} + \boldsymbol{E}$.

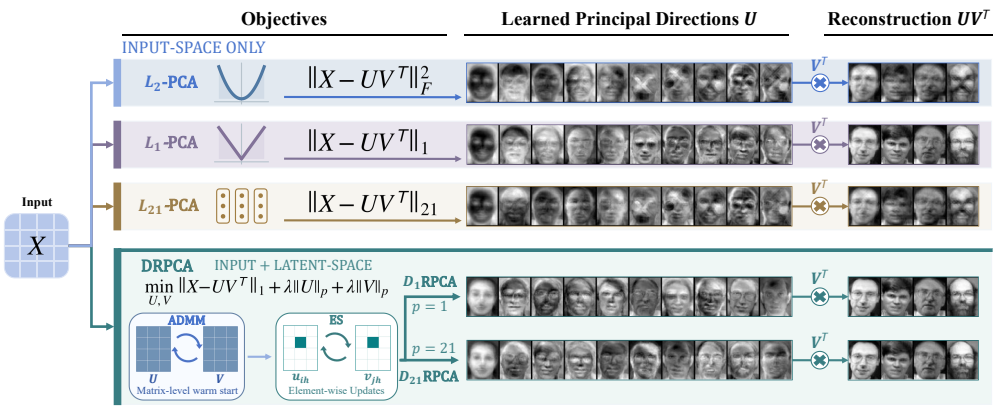


Figure 1: Comparison of learned principal directions and reconstruction results across L_2-PCA, L_1-PCA, L_{21}-PCA, D_1RPCA, and D_{21}RPCA (from top to bottom).

3 Methodology

3.1 Robust Latent Subspace with Bounded Noise

While improving robustness to a certain extent, existing robust PCA approaches (Ding et al., 2006; Wright et al., 2009; Xu et al., 2010; Zhang & Ding, 2013; Nie et al., 2014; 2021; Fang et al., 2023; Wang et al., 2025; Quan et al., 2025) remain constrained by strong assumptions about the data structure in the input space. For example, the underlying noise is typically modeled implicitly under a Laplacian prior, or explicitly via an auxiliary sparse error variable. Consequently, such an exclusive focus on input-space robustness neglects potential corruption in the latent space (*i.e.*, $\boldsymbol{U}$ and $\boldsymbol{V}$) and offers limited flexibility in the low-rank representation (*i.e.*, a fixed k), both of which may lead to degraded reconstruction performance of $\boldsymbol{U}\boldsymbol{V}^T$ in complex noisy environments.

Doubly Robust PCA (DRPCA). In contrast to ordinary least squares (LS), robust least squares (RLS) (Ghaoui & Lebret, 1997) assumes that the data matrix $(\boldsymbol{X}, \boldsymbol{b})$ is subject to deterministic (not necessarily small) perturbations $(\Delta\boldsymbol{X}, \Delta\boldsymbol{b})$, under which the worst-case residual is defined as

$$r = \max_{\|\Delta\boldsymbol{X}\ \Delta\boldsymbol{b}\|_F \le \rho} \|(\boldsymbol{X} + \Delta\boldsymbol{X})^T\boldsymbol{w} - (\boldsymbol{b} + \Delta\boldsymbol{b})\|_2^2, \tag{1}$$

where $\Delta = [\Delta\boldsymbol{X}\ \Delta\boldsymbol{b}]$ denotes an unknown-but-bounded perturbation matrix.

Motivated by the RLS framework, we consider a complex noisy scenario in which PCA must simultaneously address corruption in both the input and latent spaces. To this end, we introduce two auxiliary sparse-and-bounded matrix variables, $\Delta\boldsymbol{U}$ and $\Delta\boldsymbol{V}$, into the latent space to facilitate the construction of robust latent subspace with bounded noise. Building on this formulation, we propose the *doubly robust PCA (DRPCA)* model to minimize the worst-case reconstruction error induced by such latent noises, expressed as follows:

$$\begin{aligned} \min_{\boldsymbol{U},\boldsymbol{V}} \ \max_{\Delta\boldsymbol{U},\Delta\boldsymbol{V}} \ & \left\|\boldsymbol{X} - (\boldsymbol{U} + \Delta\boldsymbol{U})(\boldsymbol{V} + \Delta\boldsymbol{V})^T\right\|_1 \\ \text{s.t.} \quad & \|\Delta\boldsymbol{u}_i\|_1 \le \alpha_i, \qquad i = 1, \cdots, k \\ & \|\Delta\boldsymbol{v}_i\|_1 \le \beta_i, \qquad i = 1, \cdots, k \end{aligned} \tag{2}$$

where $\Delta\boldsymbol{u}_i$ and $\Delta\boldsymbol{v}_i$ denote the i-th columns of $\Delta\boldsymbol{U}$ and $\Delta\boldsymbol{V}$, respectively, and α_i and β_i are penalty parameters that control the magnitudes of $\|\Delta\boldsymbol{u}_i\|_1$ and $\|\Delta\boldsymbol{v}_i\|_1$, respectively. Since the latent noises are assumed to be sparse and bounded, the ℓ_1-norm is imposed on the additive error terms $\Delta\boldsymbol{u}_i$ and $\Delta\boldsymbol{v}_i$. As indicated by Eq. 2, the proposed DRPCA model not only employs an ℓ_1-norm based reconstruction loss (similar to L_1-PCA) to implicitly handle unknown noise in the d-dimensional input space, but also introduces an ℓ_1-norm based constraint to explicitly suppress unknown noise in the k-dimensional latent space, thereby ensuring "double" robustness across both spaces. Moreover, this latent-space constraint provides enhanced flexibility for low-rank representation learning by allowing the penalty parameters to vary adaptively across different types of noise.

Illustration the Double Robustness of DRPCA. In Fig. 1, we compare PCA and various robust PCA methods in terms of their reconstruction behavior on noisy inputs, illustrating how the pro-

posed double robustness translates into practical performance gains. The left subfigure of Fig. 1 illustrates the learned principal directions in the latent space, where the proposed DRPCA (rows 4–5) effectively suppresses noise in the learned latent representations, whereas other methods (rows 1–3) fail to do so. As shown in the right subfigure, benefiting from robust latent representations, DRPCA successfully reconstructs clean images, in sharp contrast to the noisy reconstructions produced by L_2-PCA, L_1-PCA, and L_{21}-PCA, thereby confirming that jointly enforcing robustness in both spaces yields superior reconstruction performance.

3.2 Deriving Upper Bounds for DRPCA

To address the limitations of conventional PCA and robust PCA in handling complex noise, our DRPCA model introduces robustness constraints in both the input and latent spaces. However, the resulting formulation involves non-smooth objectives and constraints, rendering direct optimization challenging. While alternating minimization is commonly used for multivariate optimization problems, it become less effective here due to the four interdependent variables (*i.e.*, $\boldsymbol{U}$, $\Delta\boldsymbol{U}$, $\boldsymbol{V}$, $\Delta\boldsymbol{V}$) coupled with ℓ_1-based sparsity constraints, resulting in a highly nontrivial optimization problem.

Deriving $\mathbf{D_1}$RPCA and $\mathbf{D_{21}}$RPCA. To make optimization tractable, we instead minimize theoretically grounded upper bounds of the DRPCA objective, yielding two variants: *"DRPCA with ℓ_1-Based Upper Bound (D_1RPCA)"* and *"DRPCA with ℓ_{21}-Based Upper Bound (D_{21}RPCA)"*. Without loss of generality, we assume that the additive noise level is uniform across all principal directions or components, and accordingly employ a single hyperparameter to control the ℓ_1-norm magnitude of the principal directions or components across each variant. This assumption leads to two distinct upper bounds for the DRPCA model, which we derive in the following.

Theorem 1. *By setting $\alpha_1 = \cdots = \alpha_k = \lambda$ and $\beta_1 = \cdots = \beta_k = \lambda$, the following ℓ_1-norm based upper bound is established for the DRPCA model (2):*

$$\max_{\Delta\boldsymbol{U},\Delta\boldsymbol{V}} \left\|\boldsymbol{X} - (\boldsymbol{U}+\Delta\boldsymbol{U})(\boldsymbol{V}+\Delta\boldsymbol{V})^T\right\|_1 \le \left\|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T\right\|_1 + \lambda\left\|\boldsymbol{U}\right\|_1 + \lambda\left\|\boldsymbol{V}\right\|_1 + k\lambda^2. \quad (3)$$

Theorem 2. *By setting $\alpha_1 = \cdots = \alpha_k = \lambda/\sqrt{n}$, and $\beta_1 = \cdots = \beta_k = \lambda/\sqrt{d}$, the following ℓ_{21}-norm based upper bound is established for the DRPCA model (2):*

$$\max_{\Delta\boldsymbol{U},\Delta\boldsymbol{V}} \left\|\boldsymbol{X} - (\boldsymbol{U}+\Delta\boldsymbol{U})(\boldsymbol{V}+\Delta\boldsymbol{V})^T\right\|_1 \le \left\|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T\right\|_1 + \lambda\left\|\boldsymbol{U}\right\|_{21} + \lambda\left\|\boldsymbol{V}\right\|_{21} + \frac{k\lambda^2}{\sqrt{nd}}. \quad (4)$$

As demonstrated in Theorems 1 and 2, the original DRPCA model (with variables $\boldsymbol{U}$, $\Delta\boldsymbol{U}$, $\boldsymbol{V}$, and $\Delta\boldsymbol{V}$) is simplified to the $\mathrm{D_1}$RPCA and $\mathrm{D_{21}}$RPCA models, which involve only the variables $\boldsymbol{U}$ and $\boldsymbol{V}$. Additionally, the latent noise components $\Delta\boldsymbol{U}$ and $\Delta\boldsymbol{V}$ are regulated by the hyperparameter λ through the ℓ_1- or ℓ_{21}-norm based regularization terms. Complete proofs of Theorems 1 and 2 are provided in Appendix B. Based on the established inequalities, minimization the upper bound loss in practice is guaranteed todecrease the original loss throughout the optimization process.

On the Connection Between Model Robustness and Sparsity-Inducing Regularization. By omitting the constant term on the right-hand side of Eqs. 3-4, the original DRPCA model (Eq. 2) simplifies to the $\mathrm{D_1}$RPCA (Eq. 5) and $\mathrm{D_{21}}$RPCA (Eq. 6) models below:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \left\|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T\right\|_1 + \lambda\left\|\boldsymbol{U}\right\|_1 + \lambda\left\|\boldsymbol{V}\right\|_1, \quad (5)$$

$$\min_{\boldsymbol{U},\boldsymbol{V}} \left\|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T\right\|_1 + \lambda\left\|\boldsymbol{U}\right\|_{21} + \lambda\left\|\boldsymbol{V}\right\|_{21}, \quad (6)$$

where λ is a hyperparameter that regulates the degree of sparsity in both the principal directions $\boldsymbol{U}$ and the principal components $\boldsymbol{V}$. The derivation of these upper bounds reveals that introducing additive noise variables $\Delta\boldsymbol{U}$ and $\Delta\boldsymbol{V}$ in latent space is mathematically equivalent to imposing sparsity-inducing regularization on the latent factors $\boldsymbol{U}$ and $\boldsymbol{V}$.

Our analysis from the perspective of robust latent representations provides a theoretical foundation for the widely accepted notion in machine learning that sparsity-inducing regularization enhances model robustness. As the hyperparameter λ approaches zero, the $\mathrm{D_1}$RPCA and $\mathrm{D_{21}}$RPCA models degenerate to L_1-PCA, emphasizing robustness solely in the input space. As λ increases from zero, these models progressively incorporate robustness in the latent space as well, ultimately achieving robustness in both spaces. Accordingly, in contrast to existing approaches, the proposed DRPCA framework provides a more generalized and robust solution for data reconstruction, with flexible control over the degree of latent-space robustness.

4 EFFICIENT HYBRID ALTERNATING OPTIMIZATION

Optimizing DRPCA (Eqs. 5-6) is challenging due to the bilinear coupling between $\boldsymbol{U}$ and $\boldsymbol{V}$ and the nonsmooth sparsity-inducing regularizers. *ADMM-based* methods Yuan & Yang (2013); Lin et al. (2010); Boyd et al. (2011); Malitsky & Mishchenko (2024); Oikonomidis et al. (2024) can efficiently produce an *approximate* solution through *matrix-wise updates* of the latent factors. Although such solutions are often suboptimal, they provide a good starting point for subsequent refinement. To further improve solution quality, we exploit the separable structure of the objective and design an *element-wise exact solver* that attains the *global optimum* of each element-wise subproblem in closed form. Accordingly, we develop a *hybrid alternating optimization* framework that first performs matrix-wise ADMM updates and then applies the exact solver for element-wise refinement. Building on this framework, we further introduce a *prescreening-based acceleration strategy* that reduces the refinement overhead without compromising solution quality. We elaborate on each algorithmic component in the following subsections.

4.1 MATRIX-WISE UPDATE VIA ADMM FOR $\mathrm{D_1}$RPCA AND $\mathrm{D_{21}}$RPCA

The alternating direction method of multipliers (ADMM) (Boyd et al., 2011) decomposes optimization problems with coupled variables and sparsity-inducing regularization into tractable subproblems. For the objectives in Eqs. 5-6, we introduce the auxiliary variables $\boldsymbol{E}$, $\boldsymbol{F}$, and $\boldsymbol{G}$, yielding the following equivalent formulation:

$$\begin{aligned} \min_{\boldsymbol{U},\boldsymbol{V},\boldsymbol{E},\boldsymbol{F},\boldsymbol{G}} \quad & \|\boldsymbol{E}\|_1 + \lambda\|\boldsymbol{F}\|_p + \lambda\|\boldsymbol{G}\|_p \\ \text{s.t.} \quad & \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{E} = 0, \quad \boldsymbol{U} - \boldsymbol{F} = 0, \quad \boldsymbol{V} - \boldsymbol{G} = 0, \end{aligned} \tag{7}$$

where $p \in \{1, 21\}$ corresponds to ℓ_1- or ℓ_{21}-norm regularization, respectively. The resulting subproblems are solved alternately, and the complete ADMM procedure is provided in Appendix D.1. Although this decomposition renders each subproblem tractable, ADMM updates five variables per iteration, and its convergence behavior depends on penalty parameters that often require problem-specific tuning. As a result, ADMM, which only approximately solves the original problem, tends to terminate at a suboptimal solution within a practical iteration budget. Nevertheless, this solution captures the global low-rank structure and thus provides a high-quality starting point for the subsequent element-wise refinement.

4.2 ELEMENT-WISE UPDATE VIA EXACT SOLVERS FOR $\mathrm{D_1}$RPCA AND $\mathrm{D_{21}}$RPCA

Simplified Optimization Subproblems. To avoid introducing additional variables and to address issues such as premature convergence, we adopt an element-wise alternating optimization strategy focused on individual matrix elements. Considering the (i_0, h_0)-th entry of $\boldsymbol{U}$, we define the corresponding residual matrix as $\widetilde{\boldsymbol{X}}$, where $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T + \boldsymbol{u}_{h_0}(\boldsymbol{v}_{h_0})^T$. During optimization, we minimize the loss function *w.r.t.* a single element $\boldsymbol{U}_{i_0h_0}$, while keeping all other elements in $\boldsymbol{U}$ fixed at each iteration. Accordingly, we derive the following simplified $\mathrm{D_1}$RPCA and $\mathrm{D_{21}}$RPCA subproblems.

Lemma 1. *Fixing $\boldsymbol{V}$ and all other elements of $\boldsymbol{U}$, the D_1RPCA model (5) simplifies to the following element-wise optimization subproblem w.r.t. u:*

$$\min_u \ \mathcal{L}_{D_1RPCA}(u) = \left\|\widetilde{\boldsymbol{x}} - u \cdot \boldsymbol{v}\right\|_1 + \lambda|u|, \tag{8}$$

where u is the (i_0, h_0)-th entry of the corresponding matrix $\boldsymbol{U}$, $\widetilde{\boldsymbol{x}}$ is the transpose of i_0-th row of the corresponding matrix $\widetilde{\boldsymbol{X}}$, and $\boldsymbol{v}$ is the h_0-th column of corresponding matrix $\boldsymbol{V}$.

Lemma 2. *Fixing $\boldsymbol{V}$ and all other elements of $\boldsymbol{U}$, the D_{21}RPCA model (6) simplifies to the following element-wise optimization subproblem w.r.t. u:*

$$\min_u \ \mathcal{L}_{D_{21}RPCA}(u) = \left\|\widetilde{\boldsymbol{x}} - u \cdot \boldsymbol{v}\right\|_1 + \lambda\sqrt{u^2 + c}, \tag{9}$$

where u, $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$ share the same definitions as in Eq. (8), and $c = \sum_{i \neq i_0} U_{ih_0}^2$.

As shown above, the loss functions in Eqs. 5–6 simplify to reconstruction and regularization terms *w.r.t.* u, with all remaining terms treated as constants. Complete proofs of Lemmas 1 and 2 are provided in Appendix C.1.

Element-Wise Exact Solvers. Both $\mathcal{L}_{\text{D}_1\text{RPCA}}$ and $\mathcal{L}_{\text{D}_{21}\text{RPCA}}$ involve the ℓ_1-norm-based reconstruction loss $\mathcal{L}_{\text{recon}}(u) = \|\widetilde{\boldsymbol{x}} - u \cdot \boldsymbol{v}\|_1$, which is piecewise linear in u. Exploiting this property, we partition the domain of u into consecutive intervals using the breakpoint set $\mathcal{S} = \{0, \widetilde{x}_1/v_1, \widetilde{x}_2/v_2, \cdots, \widetilde{x}_n/v_n\}$, within each of which $\mathcal{L}_{\text{recon}}(u)$ is linear. Accordingly, the optimal solution u^* can be identified by analyzing the derivative and convexity of the objective within each interval. In practice, u^* can be efficiently computed in linear time without requiring any parameter fine-tuning. To formally establish this result, we present the following theorems that provide closed-form solutions to Eqs. 8-9.

Theorem 3. *Given $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$, and the hyperparameter λ, the optimal solution u^* of the optimization problem (8) is given by*

$$u^* = \operatorname*{argmin}_{u \in \mathcal{S}} \mathcal{L}_{D_1RPCA}(u), \quad \nabla \mathcal{L}_{D_1RPCA}(u) = \sum_{i=0}^{n} \left|b_i\right| \cdot \operatorname{sign}(u - a_i), \tag{10}$$

where $\boldsymbol{a} = (0, \widetilde{x}_1/v_1, \cdots, \widetilde{x}_n/v_n)^T$ and $\boldsymbol{b} = (\lambda, v_1, \cdots, v_n)^T$.

Theorem 4. *Given $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$, and the hyperparameter λ, let $\widetilde{u} = \operatorname{argmin}_{u \in \mathcal{S}} \mathcal{L}_{D_{21}RPCA}(u)$, the optimal solution u^* of the optimization problem (9) is given by*

$$u^* = \begin{cases} \widetilde{u}, & \text{if } h(\widetilde{u} - \epsilon) < 0,\ h(\widetilde{u} + \epsilon) > 0, \\ \operatorname{sign}(-g(\widetilde{u} + \epsilon)) \dfrac{\sqrt{c}|g(\widetilde{u} + \epsilon)|}{\sqrt{\lambda^2 - g^2(\widetilde{u} + \epsilon)}}, & \text{if } h(\widetilde{u} \pm \epsilon) < 0, \\ \operatorname{sign}(-g(\widetilde{u} - \epsilon)) \dfrac{\sqrt{c}|g(\widetilde{u} - \epsilon)|}{\sqrt{\lambda^2 - g^2(\widetilde{u} - \epsilon)}}, & \text{if } h(\widetilde{u} \pm \epsilon) > 0, \end{cases} \tag{11}$$

where $h(u) = \nabla \mathcal{L}_{D_{21}RPCA}(u)$, $g(u) = \nabla \mathcal{L}_{recon}(u)$, and ϵ is a small positive number (e.g., 1×10^{-7}).

Additional details and complete proofs of Theorems 3 and 4 are provided in Appendices C.2-C.3.

4.3 Overall Optimization Procedure

We initialize $\boldsymbol{U}$ and $\boldsymbol{V}$ with the ADMM-based solution of Eq. 7, which already captures the global low-rank structure. We then refine both factor matrices using the proposed exact solvers in Eqs. 10 and 11. Specifically, since $\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_1 = \|\boldsymbol{Y} - \boldsymbol{V}\boldsymbol{U}^T\|_1$, where $\boldsymbol{Y} = \boldsymbol{X}^T$, the update of $\boldsymbol{V}$ follows from that of $\boldsymbol{U}$ by replacing u, $\boldsymbol{u}$, and $\widetilde{\boldsymbol{x}}$ with v, $\boldsymbol{v}$, and $\widetilde{\boldsymbol{y}}$, respectively. We therefore update $\boldsymbol{U}$ and $\boldsymbol{V}$ alternately using the proposed exact solvers until the loss reduction falls below a prescribed tolerance or the maximum number of iterations is reached. In practice, these alternating updates typically converge within five iterations. The complete procedure is presented in Algorithm 3. with detailed descriptions and convergence proofs provided in Appendices D.3 and D.4, respectively. Despite requiring only a few iterations, the Exact Solver (ES) algorithm can still incur substantial computational cost per iteration, since every entry of $\boldsymbol{U}$ and $\boldsymbol{V}$ must be updated. To further improve efficiency, we introduce a prescreening-based acceleration strategy in the following subsection.

4.4 Accelerated Hybrid Alternating Optimization

The hybrid framework combines matrix-wise ADMM updates with element-wise refinement via exact solvers. Each refinement iteration solves $k(d+n)$ element-wise subproblems and exhaustively evaluating all breakpoints costs $O(q^2)$ per subproblem, where q denotes the number of nonzero coefficients involved. To substantially improve the computational efficiency, we accelerate element-wise updates in two ways: a prescreening strategy that restricts refinement to poorly reconstructed samples, and accelerated element-wise exact solvers that avoid exhaustive breakpoint evaluation.

Residual-Driven Prescreening Strategy. Since reconstruction errors after the ADMM update tend to concentrate on a small subset of samples, we restrict the element-wise updates of each column of $\boldsymbol{V}$ to this subset, which is identified via component-specific residual scores:

$$\boldsymbol{s}_h = \left(\|\widetilde{\boldsymbol{R}}_h(1,:)\|_1, \cdots, \|\widetilde{\boldsymbol{R}}_h(n,:)\|_1\right)^T, \quad \widetilde{\boldsymbol{R}}_h = \boldsymbol{X}^T - \boldsymbol{V}\boldsymbol{U}^T + \boldsymbol{v}_h \boldsymbol{u}_h^T, \tag{12}$$

where $\boldsymbol{u}_h$ and $\boldsymbol{v}_h$ are the h-th columns of $\boldsymbol{U}$ and $\boldsymbol{V}$, $\widetilde{\boldsymbol{R}}_h$ excludes the contribution of the h-th rank-one component from the reconstruction, and $\widetilde{\boldsymbol{R}}_h(j,:)$ denotes the j-th row of $\widetilde{\boldsymbol{R}}_h$. We then use

these scores to select the entries to be refined, greedily prioritizing those with the largest residuals. Specifically, for each h, only the entries V_{jh} with $j \in \mathcal{I}_h(\gamma)$ are updated, where

$$\mathcal{I}_h(\gamma) = \mathrm{TopIdx}_{N_\gamma}(\boldsymbol{s}_h), \quad N_\gamma = \max\{\lfloor \gamma n \rfloor, 1\}, \tag{13}$$

and $\mathrm{TopIdx}_{N_\gamma}(\cdot)$ returns the indices of the N_γ largest residual scores. Here, $\gamma \in (0, 1]$ controls the fraction of samples refined for each component, while the remaining entries of $\boldsymbol{v}_h$ are left unchanged. In practice, a small γ suffices to retain most of the refinement benefit.

Accelerated Exact Solvers. For a single entry u with all other entries fixed, the reconstruction loss is $\mathcal{L}_{\text{recon}}(u) = \sum_j |\widetilde{x}_j - u\, v_j| = \sum_{j \in \mathcal{S}_{nz}} w_j |u - r_j| + \text{const}$, where $r_j = \widetilde{x}_j / v_j$, $w_j = |v_j|$, and $\mathcal{S}_{nz} = \{j : v_j \neq 0\}$ ($|\mathcal{S}_{nz}| = q$). Rather than exhaustively evaluating the objective at all q breakpoints, which requires $O(q^2)$ time, we exploit the local structure of the objective within each interval to locate the optimum more efficiently, leading to the following accelerated exact solvers.

(i) Accelerated ℓ_1-Based Exact Solver. The D_1RPCA loss defined in Eq. 8 becomes $\mathcal{L}_{\mathrm{D}_1\mathrm{RPCA}}(u) = \sum_{j \in \mathcal{S}_{nz}} w_j |u - r_j| + \lambda |u|$, where the regularizer can be viewed as an additional breakpoint at 0 with weight λ. Appending the pair $(0, \lambda)$ and sorting the breakpoint-weight pairs in ascending order of breakpoints yields $\{(\widehat{r}_{(i)}, \widehat{w}_{(i)})\}_{i=0}^{q}$. An exact minimizer is then given by the weighted median:

$$u^* = \widehat{r}_{\ell^*}, \quad \ell^* = \min\left\{ \ell : \sum_{i=0}^{\ell} \widehat{w}_{(i)} \geq \frac{1}{2} \sum_{i=0}^{q} \widehat{w}_{(i)} \right\}, \tag{14}$$

whose optimality is proven in Appendix E.2. Since the cumulative sums can be computed in a single pass after sorting, the per-subproblem cost is reduced from $O(q^2)$ to $O(q \log q)$.

(ii) Accelerated ℓ_{21}-Based Exact Solver. Up to an additive constant, the D_{21}RPCA loss defined in Eq. 9 becomes $\mathcal{L}_{\mathrm{D}_{21}\mathrm{RPCA}}(u) = \sum_{j \in \mathcal{S}_{nz}} w_j |u - r_j| + \lambda \sqrt{u^2 + c}$. When $c = 0$, the regularizer reduces to $\lambda |u|$, and the problem coincides with the ℓ_1 case solved by Eq. 14. When $c > 0$, letting $W = \sum_{j \in \mathcal{S}_{nz}} w_j$, the left and right derivatives of $\mathcal{L}_{\mathrm{D}_{21}\mathrm{RPCA}}$ are

$$g_-(u) = 2 \sum_{r_j < u} w_j - W + \lambda \frac{u}{\sqrt{u^2 + c}}, \qquad g_+(u) = 2 \sum_{r_j \leq u} w_j - W + \lambda \frac{u}{\sqrt{u^2 + c}}. \tag{15}$$

Since both terms of $g_\pm$ are non-decreasing in u, u is optimal if and only if $g_-(u) \leq 0 \leq g_+(u)$, as proven in Appendix E.3. With sorted breakpoints and precomputed prefix sums, each evaluation of $g_\pm(u)$ takes $O(\log q)$ time via binary search, so bisection yields a numerical solution in $O(q \log q + B \log q)$ time, where B is the number of bisection steps.

Overall, the proposed accelerations reduce the cost of ES refinement in two complementary aspects: prescreening decreases the number of element-wise subproblems, while the accelerated solvers lower the cost of each one. Together, they largely preserve the solution quality of exact refinement at a substantially lower computational cost, making DRPCA practical for large-scale data.

5 Experiments

5.1 Experimental Setup

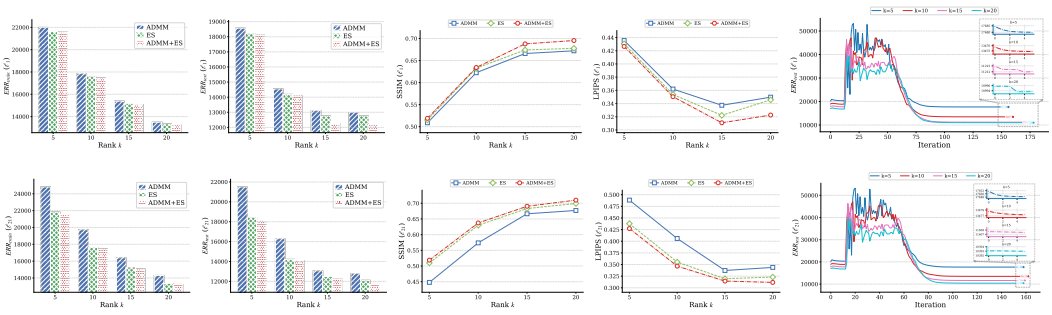
Benchmark Datasets. We evaluate eight face benchmark datasets, AR (Martínez & Kak, 2001), AT&T (Samaria & Harter, 1994), CMU-PIE (Sim et al., 2002), FaceGT (Nefian & Hayes, 1999), FERET (Phillips et al., 1998), JAFFE (Lyons et al., 1998), UMIST (Graham & Allinson, 1998), and YaleB (Georghiades et al., 2001), together with CIFAR-10 (Krizhevsky & Hinton, 2009) and ImageNet (Deng et al., 2009), under single-type rectangular and mixed-type rectangular, circular, and cross-shaped corruptions. Further details are provided in Appendices F.1-F.2.

Evaluation Metrics. Let $\boldsymbol{X}_0$, $\boldsymbol{X} = \boldsymbol{X}_0 + \boldsymbol{E}$, and $\hat{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{V}^T$ denote clean data, corrupted input, and reconstruction, respectively. We use the ℓ_1 training and testing errors, $ERR_{\text{train}} = \|\boldsymbol{X} - \hat{\boldsymbol{X}}\|_1$ and $ERR_{\text{test}} = \|\boldsymbol{X}_0 - \hat{\boldsymbol{X}}\|_1$. Additional ℓ_2 error, PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) results are reported in Appendix F.3.

Comparative Methods. We compare DRPCA with representative PCA and RPCA baselines: L_2-PCA, R_1-PCA (Huber), R_1-PCA (Cauchy) (Ding et al., 2006), L_1-PCA (ALM) (Zhang & Ding,

Table 1: Comparison of the average ℓ_1-based test reconstruction error (ERR_{test}) across four rank settings under single-type (S) and mixed-type (M) noise. Lower values are better. The best and second-best results are highlighted in red and blue, respectively.

Method	**AT&T** ($n = 100$)		**CMU** ($n = 100$)		**CIFAR-10** ($n = 1000$)		**ImageNet** ($n = 5000$)		**ImageNet** ($n = 10000$)	
	S	**M**	**S**	**M**	**S**	**M**	**S**	**M**	**S**	**M**
L_2-PCA	19049	20265	19357	17236	47702	51640	242387	258772	455798	516495
R_1-PCA (Huber)	18964	19940	18881	16520	47128	50975	240219	256703	451821	513547
R_1-PCA (Cauchy)	18865	19652	18651	15910	46404	50210	237771	254645	446818	510141
L_1-PCA (ALM)	14969	15759	13262	11431	38376	43123	196120	218490	375687	438970
RPCA	16815	18224	16203	14229	42115	45835	212749	229203	400692	459615
CRPCA	18864	19713	18669	15979	45812	49693	235890	253219	442756	507680
RPCA-OM	18548	19380	18509	15878	45759	49611	235586	252669	441459	506500
CRPCA-OM	18575	19445	18520	15826	45749	49637	235491	252814	442007	506825
L_{21}-PCA (ALM)	18861	19693	18676	16074	45927	49758	236332	253392	443203	507962
L_{21}-PCA (NG)	19050	20371	19220	17324	47863	51794	243485	260131	458843	519308
eRPCA	19102	20369	19184	18131	58197	61208	233297	248640	425651	480413
DW-RPCA	18668	18906	18666	15616	46854	50532	240402	256389	452472	514460
DCA	18856	19654	18663	15995	45834	49674	236008	253070	442196	507264
D_1RPCA(Ours)	**14157**	**14881**	**12348**	**10899**	**37689**	**42923**	**192674**	**217226**	**375506**	**436761**
D_{21}RPCA(Ours)	**14031**	**14558**	**11410**	**10862**	**37689**	**42849**	**193353**	**217251**	**375601**	**436736**



(a) ERR_{train} (b) ERR_{test} (c) SSIM (d) LPIPS (e) Convergence

Figure 2: Comparison of ADMM, ES, and hybrid ADMM+ES for solving D_1RPCA (top row) and D_{21}RPCA (bottom row) under single-noise corruption. Panels (a–d) report ERR_{train}, ERR_{test}, and SSIM across subspace ranks. Panel (e) shows the convergence of the hybrid strategy, with the inset highlighting ES iterations and their refinement of the ADMM solution.

2013), RPCA (Wright et al., 2009), CRPCA (Xu et al., 2010), RPCA-OM (Nie et al., 2014), CRPCA-OM (Nie et al., 2014), L_{21}-PCA (ALM), L_{21}-PCA (NG) (Nie et al., 2021), eRPCA (Fang et al., 2023), DW-RPCA (Wang et al., 2025), and DCA (Quan et al., 2025). Labels in parentheses denote the corresponding loss functions or optimizer variants.

Implementation Details. The convergence tolerance is set to $\tau = 10^{-6}$ for all methods. DRPCA uses ADMM initialization with at most $T_{\text{A}} = 5000$ iterations (typically < 200), followed by $T_{\text{E}} = 5$ exact-solver iterations; baselines retain their original initialization. We set $k \in \{5, 10, 15, 20\}$ for face datasets and $k \in \{40, 60, 80, 100\}$ for CIFAR-10 and ImageNet. For eRPCA, we use the larger of the predefined and adaptive ranks. The regularization parameter λ follows Section 5.3.

5.2 Comparative Performance and Hybrid Optimization

We compare DRPCA with baseline methods using the average ℓ_1 test reconstruction error (ERR_{test}) under single-type (S) and mixed-type (M) noise (Table 1). The face datasets use $k \in \{5, 10, 15, 20\}$, while CIFAR-10 and ImageNet use $k \in \{40, 60, 80, 100\}$. A DRPCA variant achieves the lowest error in all ten settings, with L_1-PCA (ALM) as the strongest non-DRPCA baseline. D_{21}RPCA performs better on both face datasets and CIFAR-10 under M noise; D_1RPCA performs better on ImageNet. The variants tie on CIFAR-10 under S noise. For DW-RPCA, we report the best result at a comparable estimated rank.

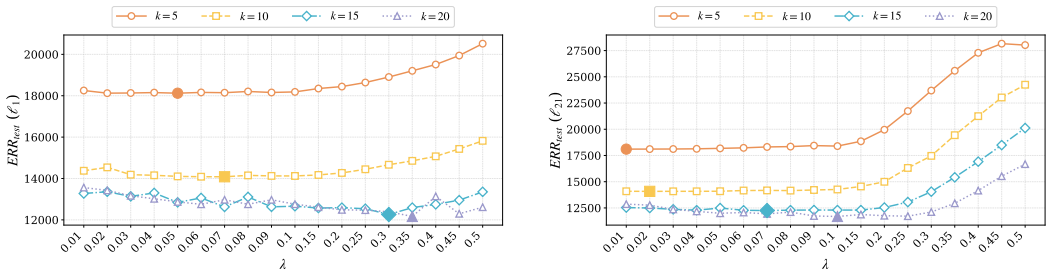


Figure 3: Sensitivity of the test reconstruction error to the regularization parameter λ for D_1RPCA (left) and D_{21}RPCA (right) on the AT&T dataset under single-noise corruption. Each curve corresponds to a subspace rank $k \in \{5, 10, 15, 20\}$, and the filled markers indicate the minimum error and corresponding optimal λ for each rank.

Table 2: Mean runtime (s) over $k \in \{5, 10, 15, 20\}$ for AT&T and CMU and $k \in \{40, 60, 80, 100\}$ for CIFAR-10 and ImageNet. Bold values indicate accelerated runtimes and their corresponding reductions. Runtime reduction is measured relative to the corresponding Hybrid w/o Acc. variant; higher percentages indicate greater savings.

Method	AT&T ($n = 100$)	CMU ($n = 100$)	CIFAR-10 ($n = 1000$)	ImageNet ($n = 5000$)
L_1-PCA (ALM)	1.04	1.93	23.45	282.41
RPCA	0.14	0.16	2.10	19.28
DCA	8.35	69.93	392.01	331.03
DW-RPCA	1.79	3.55	46.03	348.75
D_1RPCA (Matrix)	4.12	5.23	14.08	124.31
D_1RPCA (Hybrid w/o Acc.)	63.12	52.02	193.96	1670.60
D_1RPCA (Hybrid Acc.)	**14.23**	**16.53**	**51.23**	**219.81**
Runtime reduction (↓)	**77.5%**	**68.2%**	**73.6%**	**86.8%**
D_{21}RPCA (Matrix)	4.72	5.68	20.99	156.52
D_{21}RPCA (Hybrid w/o Acc.)	164.70	85.40	675.38	3933.57
D_{21}RPCA (Hybrid Acc.)	**124.60**	**52.03**	**240.60**	**725.31**
Runtime reduction (↓)	**24.3%**	**39.1%**	**64.4%**	**81.6%**

We further compare matrix-level ADMM, element-level ES, and their hybrid ADMM+ES strategy for the same D_1RPCA and D_{21}RPCA objectives (Eqs. 5–6). Under matched rank settings on AT&T, the hybrid generally yields lower training and test errors, higher SSIM, and lower LPIPS than either method alone (Fig. 2). The convergence curves are consistent with rapid initial progress from ADMM followed by further refinement through ES. Additional results appear in Appendix F.5.

5.3 Hyperparameter Sensitivity and Runtime

On AT&T under single-type noise, both DRPCA variants maintain relatively stable ERR_{test} at small-to-moderate λ across $k \in \{5, 10, 15, 20\}$ (Fig. 3). Larger λ degrades performance, while the best λ tends to increase with k; $k = 20$ gives the lowest errors within the tested range. These results motivate joint selection of k and λ. Additional results appear in Appendix F.6.

Table 2 reports rank-averaged runtime across four datasets. RPCA is the fastest method in every dataset. The matrix DRPCA variants are slower than L_1-PCA (ALM) on the face datasets but faster on CIFAR-10 and ImageNet. The hybrid variants require more time than their matrix counterparts, although acceleration reduces their runtime relative to the unaccelerated hybrid implementations.

6 Conclusion

In this paper, we propose a novel doubly robust principal component analysis (DRPCA) approach, which enhances robustness in both the input and latent spaces. Additionally, we introduce a hybrid alternating optimization algorithm based on exact solvers, which effectively solves the multivariate optimization problem and mitigates the premature convergence commonly observed in purely matrix-wise alternating updates. Experimental results demonstrate that the proposed DRPCA method outperforms state-of-the-art robust PCA approaches, and the proposed hybrid optimization algorithm consistently yields better solutions than either matrix-wise or element-wise updates alone.

REFERENCES

Basit Alawode and Sajid Javed. Learning spatial-temporal regularized tensor sparse rpca for background subtraction. *IEEE Transactions on Neural Networks and Learning Systems*, 36(6):11034–11048, 2025.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3:1–122, 2011.

HanQin Cai, Keaton Hamm, Longxiu Huang, Jiaqi Li, and Tao Wang. Rapid robust principal component analysis: Cur accelerated inexact low rank estimation. *IEEE Signal Processing Letters*, 28:116–120, 2020.

HanQin Cai, Jialin Liu, and Wotao Yin. Learned robust PCA: A scalable deep unfolding approach for high-dimensional outlier detection. In *Advances in Neural Information Processing Systems*, volume 34, pp. 16977–16989, 2021.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Chris Ding, Ding Zhou, Xiaofeng He, and Hongyuan Zha. R1-pca: Rotational invariant ℓ_1-norm principal component analysis for robust subspace factorization. In *Proceedings of the 23rd International Conference on Machine Learning*, pp. 281–288, 2006.

Shun Fang, Zhengqin Xu, Shiqian Wu, and Shoulie Xie. Efficient robust principal component analysis via block krylov iteration and cur decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1348–1357, 2023.

Lanlan Feng, Yipeng Liu, Ziming Liu, and Ce Zhu. Online nonconvex robust tensor principal component analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 36(8):14384–14398, 2024.

Yunlong Gao, Yuzhe Feng, Youwei Xie, Jinyan Pan, and Feiping Nie. Normalized robust pca with adaptive reconstruction error minimization. *IEEE Transactions on Knowledge and Data Engineering*, 36(6):2587–2599, 2023.

Yunlong Gao, Qinting Wu, Zhenghong Xu, Chao Cao, Jinyan Pan, Guifang Shao, Feiping Nie, and Qingyuan Zhu. Principal component analysis with fuzzy elastic net for feature selection. *IEEE Transactions on Fuzzy Systems*, 32:6878–6890, 2024.

A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:643–660, 2001.

Laurent El Ghaoui and Hervé Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM Journal on Matrix Analysis and Applications*, 18:1035–1064, 1997.

Daniel B Graham and Nigel M Allinson. Characterising virtual eigensignatures for general purpose face recognition. In *Face recognition: from theory to applications*, pp. 446–456. Springer, 1998.

Zhi Han, Yanmei Wang, Shaojie Zhang, Huijie Fan, Yandong Tang, and Yao Wang. Online video sparse noise removing via nonlocal robust pca. *IEEE Transactions on Multimedia*, 26:7130–7145, 2024.

Po-Sen Huang, Scott Deeann Chen, Paris Smaragdis, and Mark Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 57–60. IEEE, 2012.

Yukara Ikemiya, Katsutoshi Itoyama, and Kazuyoshi Yoshii. Singing voice separation and vocal f0 estimation based on mutual combination of robust principal component analysis and subharmonic summation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):2084–2095, 2016.

Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

Boris Joukovsky, Yonina C Eldar, and Nikos Deligiannis. Interpretable neural networks for video separation: Deep unfolding rpca with foreground masking. *IEEE Transactions on Image Processing*, 33:108–122, 2023.

Qifa Ke and Takeo Kanade. Robust l1 norm factorization in the presence of outliers and missing data by alternative convex programming. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pp. 739–746, 2005.

Minchul Kim, Anil K Jain, and Xiaoming Liu. Adaface: Quality adaptive margin for face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18729–18738, 2022.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Zina Li, Yao Wang, Qian Zhao, Shijun Zhang, and Deyu Meng. A tensor-based online rpca model for compressive background subtraction. *IEEE Transactions on Neural Networks and Learning Systems*, 34:10668–10682, 2023.

Cheng Liang, Lianzhi Wang, Li Liu, Huaxiang Zhang, and Fei Guo. Multi-view unsupervised feature selection with tensor robust principal component analysis and consensus graph learning. *Pattern Recognition*, 141:109632, 2023.

Zhouchen Lin, Minming Chen, and Yi Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.

Michael J Lyons, Shigeru Akamatsu, Miyuki Kamachi, Jiro Gyoba, and Julien Budynek. The japanese female facial expression (jaffe) database. In *Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition*, pp. 200–205, 1998.

Yura Malitsky and Konstantin Mishchenko. Adaptive proximal gradient method for convex optimization. In *Advances in Neural Information Processing Systems*, volume 37, pp. 100670–100697, 2024.

Aleix M. Martínez and Avinash C. Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:228–233, 2001.

Qiang Meng, Shichao Zhao, Zhida Huang, and Feng Zhou. Magface: A universal representation for face recognition and quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14220–14229, 2021.

A.V. Nefian and M.H. Hayes. An embedded hmm-based approach for face detection and recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pp. 3553–3556, 1999.

Feiping Nie, Jianjun Yuan, and Heng Huang. Optimal mean robust principal component analysis. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pp. 1062–1070, 2014.

Feiping Nie, Lai Tian, Heng Huang, and Chris Ding. Non-greedy l21-norm maximization for principal component analysis. *IEEE Transactions on Image Processing*, 30:5277–5286, 2021.

Konstantinos Oikonomidis, Emanuel Laude, Puya Latafat, Andreas Themelis, and Panagiotis Patrinos. Adaptive proximal gradient methods are universal without approximation. In *International Conference on Machine Learning*, pp. 38663–38682, 2024.

P.Jonathon Phillips, Harry Wechsler, Jeffery Huang, and Patrick J. Rauss. The feret database and evaluation procedure for face-recognition algorithms. *Image and Vision Computing*, 16:295–306, 1998.

Jan Quan, Johan Suykens, and Panagiotis Patrinos. Rethinking pca through duality. In *Advances in Neural Information Processing Systems*, volume 38, pp. 52381–52415, 2025.

F.S. Samaria and A.C. Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of IEEE Workshop on Applications of Computer Vision*, pp. 138–142, 1994.

Terence Sim, Simon Baker, and Maan Bsat. The cmu pose, illumination, and expression (pie) database. In *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, pp. 53–58, 2002.

Sisi Wang, Feiping Nie, Zheng Wang, Rong Wang, and Xuelong Li. Data subdivision based dual-weighted robust principal component analysis. *IEEE Transactions on Image Processing*, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13:600–612, 2004.

John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2080–2088. 2009.

Honghui Xu, Chuangjie Fang, Renfang Wang, Shengyong Chen, and Jianwei Zheng. Dual-enhanced high-order self-learning tensor singular value decomposition for robust principal component analysis. *IEEE Transactions on Artificial Intelligence*, 5(7):3564–3578, 2024.

Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust PCA via outlier pursuit. In *Advances in Neural Information Processing Systems*, volume 23, pp. 2496–2504, 2010.

Zhengqin Xu, Rui He, Shoulie Xie, and Shiqian Wu. Adaptive rank estimate in robust principal component analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6573–6582, 2021.

Niannan Xue, Jiankang Deng, Shiyang Cheng, Yannis Panagakis, and Stefanos Zafeiriou. Side information for face completion: A robust pca approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:2349–2364, 2019.

Xiaoming Yuan and Junfeng Yang. Sparse and low-rank matrix decomposition via alternating direction method. *Pacific Journal of Optimization*, 9:167, 2013.

Miao Zhang and Chris Ding. Robust tucker tensor decomposition for effective image representation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2448–2455, 2013.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

Rui Zhang and Hanghang Tong. Robust principal component analysis with adaptive neighbors. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

APPENDIX

This appendix provides additional theoretical proofs, algorithmic details, and experimental results that further substantiate the findings presented in the main paper, and is organized as follows:

- **Section A**: A comprehensive review of related work on PCA and its robust variants.
- **Section B**: Detailed derivations and proofs of the upper bounds on the DRPCA objective, leading to the ℓ_1- and $\ell_{2,1}$-regularized formulations.
- **Section C**: Simplification of the DRPCA problem into element-wise subproblems, proofs of the exact solvers, and visualizations of the resulting solutions.
- **Section D**: The hybrid alternating optimization algorithm, which combines matrix-wise ADMM updates with element-wise exact-solver refinement, together with its convergence analysis.
- **Section E**: Accelerated exact solvers for the ℓ_1 and $\ell_{2,1}$ subproblems, including optimality proofs, complete algorithms, and complexity analysis.
- **Section F**: Additional experimental details and results, including dataset and corruption settings, extended comparisons, analysis of the hybrid optimization, and hyperparameter sensitivity.

Use of Large Language Models. Large language models (LLMs) were used solely to assist with editing and polishing the writing of this paper. All substantive content, including the theoretical results, empirical findings, and experimental design, was developed entirely by the authors without the assistance of LLMs.

A RELATED WORK

A.1 PCA

Principal component analysis (PCA) (Jolliffe & Cadima, 2016) is a fundamental technique for dimensionality reduction. From the matrix decomposition viewpoint, PCA seeks a low-rank approximation of the original data by minimizing the Frobenius norm of reconstruction errors, *i.e.*, $\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_F^2$, where $\boldsymbol{U} = (\boldsymbol{u}_1, \cdots, \boldsymbol{u}_k) \in \mathbb{R}^{d\times k}$ and $\boldsymbol{V} = (\boldsymbol{v}_1, \cdots, \boldsymbol{v}_k) \in \mathbb{R}^{n\times k}$ represent principal directions and components. Complementarily, from the statistical perspective, PCA aims to identify the top k directions that capture the maximum variance in the data. This is equivalent to maximizing the projected variance defined by the sample covariance matrix, *i.e.*, $\mathrm{Tr}(\boldsymbol{U}^T\boldsymbol{X}\boldsymbol{X}^T\boldsymbol{U})$ subject to $\boldsymbol{U}^T\boldsymbol{U} = I$.

A.2 ROBUST PCA

We begin by reviewing PCA variants developed with robust loss functions. The ℓ_1-norm, being less sensitive to outliers than the ℓ_2-norm, has been widely adopted in robust formulations. To enhance robustness, L_1-PCA (Ke & Kanade, 2005; Zhang & Ding, 2013) reformulates the matrix factorization problem as an ℓ_1-norm minimization $\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_1$. Building upon this, the rotational invariant R_1-PCA (Ding et al., 2006) introduces a mixed-norm objective that applies the ℓ_2-norm across features and the ℓ_1-norm across data samples, thereby further reducing the influence of outliers. Later, L_{21}-PCA (NG) (Nie et al., 2021) improves robustness by adopting an ℓ_{21}-norm based objective along with a non-greedy optimization strategy. In addition, Nie *et al.* (Nie et al., 2014) propose robust PCA with optimal mean (RPCA-OM), which improves the reconstruction quality by adaptively removing the optimal sample mean. In contrast to the aforementioned formulations, RPCA (Wright et al., 2009; Lin et al., 2010; Candès et al., 2011) aims to recover a low-rank matrix $\boldsymbol{A}$ from a corrupted observation $\boldsymbol{X}$ that contains unknown sparse errors $\boldsymbol{E}$. Accordingly, RPCA minimizes the following objective function $\|\boldsymbol{A}\|_* + \lambda\|\boldsymbol{E}\|_1$ subject to $\boldsymbol{X} = \boldsymbol{A} + \boldsymbol{E}$. Moreover, Xu et al. (2010) propose a convex RPCA (CRPCA) which replaces the ℓ_1-norm with ℓ_{21}-norm to handle structured outliers.

A.3 EFFICIENT AND ADAPTIVE ROBUST PCA

Adaptive rank estimation-based RPCA (ARE-RPCA) (Xu et al., 2021) adaptively assigns weights to different singular values based on estimated rank information, thereby enhancing recovery accuracy. Cai et al. (2020) propose the iterated robust CUR algorithm, which accelerates low-rank recovery

by leveraging CUR decomposition on small submatrices. Similarly, Fang et al. (2023) introduce an efficient RPCA method that combines block Krylov iteration for adaptive rank estimation with CUR decomposition for low-rank matrix reconstruction. Learned robust PCA (LRPCA) (Cai et al., 2021) parameterizes the classical RPCA model and unfolds it into a feedforward neural network, allowing its parameters to be optimized via backpropagation for improved performance. In addition, a general framework named RWL-AN (Zhang & Tong, 2019) is introduced to learn an adaptive weight vector that simultaneously promotes robustness and preserves sparse neighborhood structure. More recently, Wang et al. (2025) introduce data-subdivision-based dual-weighted RPCA (DW-RPCA), which adaptively assigns weights to outliers, positive samples, and hard samples while refining the data center. Quan et al. (2025) study PCA within a difference-of-convex framework, show that simultaneous iteration can be interpreted as an instance of the difference-of-convex algorithm (DCA), and derive a kernelizable dual formulation for robust ℓ_1-PCA.

A.4 Data Selection Based Robust PCA

Gao et al. (2024) propose PCA with fuzzy elastic net, which introduces a regularization term combining the ℓ_{22} and ℓ_{2p} norms. By employing an adaptive weighting scheme, the method effectively balances sparsity and feature correlation, thereby improving the generalization ability and robustness of feature selection. In another line of work, normal samples are further categorized into positive samples and hard samples, where the former offer more informative guidance for learning the projection matrix. To exploit this distinction, the data subdivision-based dual-weighted robust PCA (DW-RPCA) method (Wang et al., 2025) introduces a mark vector to differentiate positive samples from outliers and applies self-constrained weights along with the optimal mean to enhance both robustness and model accuracy.

B Further Explanations and Proofs of Upper Bounds for DRPCA

To simplify the constrained min-max optimization process, we first apply the triangle inequality property of the ℓ_1-norm to derive an upper bound for the original doubly robust PCA (DRPCA) objective defined in Eq. 2:

$$\begin{aligned} & \left\| \boldsymbol{X} - (\boldsymbol{U} + \Delta\boldsymbol{U})(\boldsymbol{V} + \Delta\boldsymbol{V})^T \right\|_1 \\ = \; & \left\| (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T) - \Delta\boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{U}\Delta\boldsymbol{V}^T - \Delta\boldsymbol{U}\Delta\boldsymbol{V}^T \right\|_1 \\ \leq \; & \left\| \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T \right\|_1 + \left\| \Delta\boldsymbol{U}\boldsymbol{V}^T \right\|_1 + \left\| \boldsymbol{U}\Delta\boldsymbol{V}^T \right\|_1 + \left\| \Delta\boldsymbol{U}\Delta\boldsymbol{V}^T \right\|_1. \end{aligned} \tag{16}$$

Excluding the first term (*i.e.*, the ℓ_1-norm based reconstruction loss), which does not involve the latent perturbations $\Delta\boldsymbol{U}$ and $\Delta\boldsymbol{V}$. we proceed to derive upper bounds for the remaining three terms as follows.

Lemma 3. *By setting $\alpha_1 = \cdots = \alpha_k = \lambda$, the following upper bound holds:*

$$\max_{\Delta\boldsymbol{U}} \left\| \Delta\boldsymbol{U}\boldsymbol{V}^T \right\|_1 \leq \lambda \left\| \boldsymbol{V} \right\|_1. \tag{17}$$

Proof of Lemma 3. The term $\|\Delta\boldsymbol{U}\boldsymbol{V}^T\|_1$ can be equivalently reformulated as follows:

$$\begin{aligned} \left\| \Delta\boldsymbol{U}\boldsymbol{V}^T \right\|_1 &= \sum_{ij} \left| \left(\Delta\boldsymbol{U}\boldsymbol{V}^T \right)_{ij} \right| = \sum_{ij} \left| \sum_h \Delta U_{ih} V_{jh} \right| \\ &\leq \sum_{ij} \sum_h \left| \Delta U_{ih} \right| \left| V_{jh} \right| = \sum_h \Big(\sum_i \left| \Delta U_{ih} \right| \Big) \Big(\sum_j \left| V_{jh} \right| \Big) \\ &= \sum_h \left\| {}_\Delta\boldsymbol{u}_h \right\|_1 \left\| \boldsymbol{v}_h \right\|_1 \leq \lambda \Big(\sum_h \left\| \boldsymbol{v}_h \right\|_1 \Big) = \lambda \left\| \boldsymbol{V} \right\|_1, \end{aligned} \tag{18}$$

where ${}_\Delta\boldsymbol{u}_h$ and $\boldsymbol{v}_h$ represent the h-th column of the corresponding matrices $\Delta\boldsymbol{U}$ and $\boldsymbol{V}$ respectively. The first inequality in Eq. 18 is obtained according to the triangular inequality of ℓ_1-norm, *i.e.*, $\left\| \boldsymbol{a}^T\boldsymbol{b} \right\|_1 = \left| \sum_i a_i b_i \right| \leq \sum_i |a_i b_i| = \sum_i |a_i||b_i|$. The second inequality in Eq. 18 is based on the following constraints $\left\| {}_\Delta\boldsymbol{u}_h \right\|_1 \leq \lambda$ (for $h = 1, \cdots, k$). Thus, this proof is complete. □

Lemma 4. *By setting $\alpha_1 = \cdots = \alpha_k = \lambda/\sqrt{n}$, the following upper bound holds:*

$$\max_{\Delta\boldsymbol{U}} \left\| \Delta\boldsymbol{U}\boldsymbol{V}^T \right\|_1 \leq \lambda \left\| \boldsymbol{V} \right\|_{21}. \tag{19}$$

Proof of Lemma 4. The term $\|\Delta \boldsymbol{U}\boldsymbol{V}^T\|_1$ can be equivalently reformulated as follows:

$$\begin{aligned}\big\|\Delta \boldsymbol{U}\boldsymbol{V}^T\big\|_1 &= \big\|\sum_h \Delta \boldsymbol{u}_h \boldsymbol{v}_h^T\big\|_1 \leq \sum_h \big\|\Delta \boldsymbol{u}_h \boldsymbol{v}_h^T\big\|_1 \\ &= \sum_h \sum_{j=1}^{d} \big\|\big\langle \Delta u_h^j \cdot \mathbf{1}, \boldsymbol{v}_h\big\rangle\big\|_1 \leq \sum_h \sum_{j=1}^{d} \big(\sqrt{n}\big|\Delta u_h^j\big|\big)\big(\big\|\boldsymbol{v}_h\big\|_2\big) \\ &= \sum_h \sqrt{n}\,\big\|\Delta \boldsymbol{u}_h\big\|_1 \big\|\boldsymbol{v}_h\big\|_2 \leq \big(\sqrt{n}\tfrac{\lambda}{\sqrt{n}}\big)\big(\sum_h \big\|\boldsymbol{v}_h\big\|_2\big) = \lambda \big\|\boldsymbol{V}\big\|_{21},\end{aligned} \tag{20}$$

where $\Delta \boldsymbol{u}_h$ and $\boldsymbol{v}_h$ represent the h-th column of corresponding matrices $\Delta \boldsymbol{U}$ and $\boldsymbol{V}$ respectively, Δu_h^j is the j-th element in $\Delta \boldsymbol{u}_h$, and $\mathbf{1}$ is the n-dimensional vector with all ones. The first inequality in Eq. 20 is obtained according to the triangular inequality of ℓ_1-norm, *i.e.*, $\|\boldsymbol{a}_1^T\boldsymbol{b}_1 + \cdots + \boldsymbol{a}_n^T\boldsymbol{b}_n\|_1 \leq \|\boldsymbol{a}_1^T\boldsymbol{b}_1\|_1 + \cdots + \|\boldsymbol{a}_n^T\boldsymbol{b}_n\|_1$. The second inequality in Eq. 20 is obtained using the Cauchy Schwarz inequality, *i.e.*, given $\boldsymbol{x}$ and $\boldsymbol{y}$, we have the following inequality in dot-product space $|\langle \boldsymbol{x}, \boldsymbol{y}\rangle| \leq \|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2$, where $\boldsymbol{x} = u_h^j \cdot \mathbf{1}$ and $\boldsymbol{y} = \boldsymbol{v}_h$. The third inequality in Eq. 20 is based on the following constraints $\|\Delta \boldsymbol{u}_h\|_1 \leq \lambda/\sqrt{n}$ (for $h = 1, \cdots, k$). Thus, this proof is complete. □

With Lemmas 3 and 4 bounding the perturbation terms in Eq. 16 under the ℓ_1- and ℓ_{21}-norm constraints, respectively, we are now ready to prove Theorems 1 and 2. Specifically, substituting these bounds into Eq. 16 yields upper bounds on the DRPCA objective that depend only on $\boldsymbol{U}$ and $\boldsymbol{V}$, as detailed below.

Proof of Theorem 1. According to Lemma 3, we can obtain the following upper bounds for the third and fourth terms on the right-hand side of Eq. (16) by setting $\alpha_1 = \cdots = \alpha_k = \lambda$ and $\beta_1 = \cdots = \beta_k = \lambda$:

$$\max_{\Delta \boldsymbol{V}} \big\|\boldsymbol{U}\Delta \boldsymbol{V}^T\big\|_1 \leq \lambda \big\|\boldsymbol{U}\big\|_1, \quad \max_{\Delta \boldsymbol{U},\, \Delta \boldsymbol{V}} \big\|\Delta \boldsymbol{U}\Delta \boldsymbol{V}^T\big\|_1 \leq k\lambda^2, \tag{21}$$

where the inequalities are based on the constraints that $\|\Delta \boldsymbol{v}_h\|_1 \leq \lambda$, $\|\Delta \boldsymbol{u}_h\|_1 \leq \lambda$ (for $h = 1, \cdots, k$). By combining the results from Eqs. 16, 17, and 21, we can obtain the following inequality:

$$\max_{\Delta \boldsymbol{U}, \Delta \boldsymbol{V}} \big\|\boldsymbol{X} - (\boldsymbol{U} + \Delta \boldsymbol{U})(\boldsymbol{V} + \Delta \boldsymbol{V})^T\big\|_1 \leq \big\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\big\|_1 + \lambda\big\|\boldsymbol{U}\big\|_1 + \lambda\big\|\boldsymbol{V}\big\|_1 + k\lambda^2, \tag{22}$$

which completes the proof. □

Proof of Theorem 2. According to Lemma 4, we can obtain the following upper bounds for the third and fourth terms on the right-hand side of Eq. (16) by setting $\alpha_1 = \cdots = \alpha_k = \lambda$ and $\beta_1 = \cdots = \beta_k = \lambda$:

$$\max_{\Delta \boldsymbol{V}} \big\|\boldsymbol{U}\Delta \boldsymbol{V}^T\big\|_1 \leq \lambda \big\|\boldsymbol{U}\big\|_{21}, \quad \max_{\Delta \boldsymbol{U}, \Delta \boldsymbol{V}} \big\|\Delta \boldsymbol{U}\Delta \boldsymbol{V}^T\big\|_1 \leq \frac{k\lambda^2}{\sqrt{nd}}, \tag{23}$$

where the inequalities are based on the assumption that $\|\Delta \boldsymbol{u}_h\|_1 \leq \lambda/\sqrt{n}$, $\|\Delta \boldsymbol{v}_h\|_1 \leq \lambda/\sqrt{d}$ (for $h = 1, \cdots, k$). By combining the results from Eqs. 16, 19, and 23, we can obtain the following inequality:

$$\max_{\Delta \boldsymbol{U}, \Delta \boldsymbol{V}} \big\|\boldsymbol{X} - (\boldsymbol{U} + \Delta \boldsymbol{U})(\boldsymbol{V} + \Delta \boldsymbol{V})^T\big\|_1 \leq \big\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\big\|_1 + \lambda\big\|\boldsymbol{U}\big\|_{21} + \lambda\big\|\boldsymbol{V}\big\|_{21} + \frac{k\lambda^2}{\sqrt{nd}}, \tag{24}$$

which completes the proof. □

C Further Explanations and Proofs of Exact Solver for DRPCA

C.1 Derivation of the Simplified Optimization Subproblem

To effectively optimize the DRPCA problem, we update one element of the principal directions or components at a time while keeping all others fixed. By doing so, the proposed exact solvers efficiently achieve the global minimum of the overall loss function *w.r.t.* each individual element. Here, we simplify D_1RPCA and D_{21}RPCA models *w.r.t.* a single entry (i_0, h_0) of the matrix $\boldsymbol{U}$ ($i_0 = 1, \cdots, d$, $h_0 = 1, \cdots, k$), while assuming the rest of the entries ($i \neq i_0$, $h \neq h_0$) are fixed, *i.e.*, the constant values.

We first decompose the ℓ_1-norm based reconstruction loss term, *i.e.*, $\|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T\|_1$, as follows:

$$\begin{aligned}\mathcal{L}_{\text{recon}}(\boldsymbol{U}) &= \left\|\boldsymbol{X}-\boldsymbol{U}\boldsymbol{V}^T\right\|_1 \\ &= \sum_{i\neq i_0}\sum_j\left|X_{ij}-\sum_h U_{ih}V_{jh}\right| + \sum_j\left|X_{i_0j}-\sum_{h\neq h_0}U_{i_0h}V_{jh}-U_{i_0h_0}V_{jh_0}\right| \\ &= \sum_j\left|\widetilde{X}_{i_0j}-U_{i_0h_0}V_{jh_0}\right| + \text{const},\end{aligned} \tag{25}$$

where $\widetilde{X}_{i_0j}=X_{i_0j}-\sum_{h\neq h_0}U_{i_0h}V_{jh}$, and const denotes the constant values that are independent of the entry (i_0,h_0).

Subsequently, we derive the simplified optimization subproblem by explicitly decomposing the ℓ_1 and ℓ_{21} regularization terms as follows.

Proof of Lemma 1. The ℓ_1-norm based regularization term $\|\boldsymbol{U}\|_1$ is decomposed *w.r.t.* the entry $U_{i_0h_0}$ as follows:

$$\mathcal{L}_{\text{reg1}}(\boldsymbol{U}) = \left\|\boldsymbol{U}\right\|_1 = \left|U_{i_0h_0}\right| + \sum_{i\neq i_0}\sum_{h\neq h_0}\left|U_{ih}\right| = \left|U_{i_0h_0}\right| + \text{const}, \tag{26}$$

where const denotes the constant values that are independent of the entry (i_0,h_0). By combining the results from Eqs. 25 and 26, we can obtain the simplified subproblem via discarding the constant terms:

$$\begin{aligned}\mathcal{L}_{\text{recon}}(\boldsymbol{U})+\lambda\cdot\mathcal{L}_{\text{reg1}}(\boldsymbol{U}) &= \sum_j\left|\widetilde{X}_{i_0j}-U_{i_0h_0}V_{jh_0}\right| + \lambda\cdot\left|U_{i_0h_0}\right| \\ &= \left\|\widetilde{\boldsymbol{x}}-u\cdot\boldsymbol{v}\right\|_1+\lambda|u|,\end{aligned} \tag{27}$$

where u is the (i_0,h_0)-th entry of the corresponding matrix $\boldsymbol{U}$, $\widetilde{\boldsymbol{x}}$ is the transpose of i_0-th row of the corresponding matrix $\widetilde{\boldsymbol{X}}$, and $\boldsymbol{v}$ is the h_0-th column of corresponding matrix $\boldsymbol{V}$. Thus, we obtain the simplified loss function $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ defined in Eq. 8, which completes the proof. □

Proof of Lemma 2. The ℓ_{21}-norm based regularization term $\|\boldsymbol{U}\|_{21}$ is decomposed *w.r.t.* the entry $U_{i_0h_0}$ as follows:

$$\mathcal{L}_{\text{reg21}}(\boldsymbol{U}) = \left\|\boldsymbol{U}\right\|_{21} = \sqrt{U_{i_0h_0}^2+\sum_{i\neq i_0}U_{ih_0}^2} + \sum_{h\neq h_0}\sqrt{\sum_i U_{ih}^2} = \sqrt{U_{i_0h_0}^2+c}+\text{const}, \tag{28}$$

where $c=\sum_{i\neq i_0}U_{ih_0}^2$, and const denotes the constant values that are independent of the entry (i_0,h_0). By combining the results from Eqs. 25 and 28, we can obtain the simplified subproblem via discarding the constant terms:

$$\begin{aligned}\mathcal{L}_{\text{recon}}(\boldsymbol{U})+\lambda\cdot\mathcal{L}_{\text{reg21}}(\boldsymbol{U}) &= \sum_j\left|\widetilde{X}_{i_0j}-U_{i_0h_0}V_{jh_0}\right| + \lambda\cdot\sqrt{U_{i_0h_0}^2+c} \\ &= \left\|\widetilde{\boldsymbol{x}}-u\cdot\boldsymbol{v}\right\|_1+\lambda\sqrt{u^2+c},\end{aligned} \tag{29}$$

where u is the (i_0,h_0)-th entry of the corresponding matrix $\boldsymbol{U}$, $\widetilde{\boldsymbol{x}}$ is the transpose of i_0-th row of the corresponding matrix $\widetilde{\boldsymbol{X}}$, and $\boldsymbol{v}$ is the h_0-th column of corresponding matrix $\boldsymbol{V}$. Thus, we obtain the simplified loss function $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ defined in Eq. 9, which completes the proof.

C.2 Optimality Analysis of Exact Solvers for the D_1RPCA Model

Analysis of the Gradient and Convexity of the Loss Function. First, we rewrite $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ defined in Eq. 8 equivalently as

$$\begin{aligned}\mathcal{L}_{\text{D}_1\text{RPCA}}(u) &= \left\|\widetilde{\boldsymbol{x}}-u\cdot\boldsymbol{v}\right\|_1+\lambda|u| \\ &= \sum_{i=1}^n\left|v_i\right|\left|\tfrac{\widetilde{x}_i}{v_i}-u\right|+\lambda|0-u| \\ &= \sum_{i=0}^n\left|b_i\right|\left|a_i-u\right|\end{aligned} \tag{30}$$

where $\boldsymbol{a} = (0, \widetilde{x}_1/v_1, \cdots, \widetilde{x}_n/v_n)^T$, $\boldsymbol{b} = (\lambda, v_1, \cdots, v_n)^T$. Next, we compute the gradient of $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ *w.r.t.* u as

$$\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u) = \sum_{i=0}^{n} \big|b_i\big| \cdot \text{sign}(u - a_i). \tag{31}$$

It is obvious that the gradient changes when u is crossing any a_i for $i = 0, \cdots, n$, since $\text{sign}(u - a_i)$ turns from -1 to $+1$ at the breakpoint a_i.

For simplicity of the gradient analysis of $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$, we sort $\boldsymbol{a} = (a_0, a_1, \cdots, a_n)^T$ in an ascending order $\mathcal{I}$, where $a_{\mathcal{I}_0} \leq a_{\mathcal{I}_1} \leq a_{\mathcal{I}_2} \leq \cdots \leq a_{\mathcal{I}_n}$. Then, the entire coordinate axis of u can be divided into $n + 2$ intervals, according to these $n + 1$ end points of $\boldsymbol{a}$.

Thus, when u is in each interval, the gradient can be analyzed as follows: (i) If $u \in (-\infty, a_{\mathcal{I}_0})$, $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u) = -\sum_{i=0}^{n}\big|b_i\big|$. (ii) If $u \in (a_{\mathcal{I}_n}, +\infty)$, $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u) = \sum_{i=0}^{n}\big|b_i\big|$. (iii) If $u \in (a_{\mathcal{I}_k}, a_{\mathcal{I}_{k+1}})$ and $0 < k < n$, $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u) = \sum_{i=0}^{k}\big|b_{\mathcal{I}_i}\big| - \sum_{j=k+1}^{n}\big|b_{\mathcal{I}_j}\big|$, which is in-between the result of (i) and (ii). (iv) Given $u_1 \in (a_{\mathcal{I}_k}, a_{\mathcal{I}_{k+1}})$ and $u_2 \in (a_{\mathcal{I}_k+1}, a_{\mathcal{I}_{k+2}})$, the following inequality can be obtained: $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u_2) - \nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u_1) = 2\big|b_{\mathcal{I}_{k+1}}\big| > 0$.

To summarize, when u passes through all the intervals $(-\infty, a_{\mathcal{I}_0})$, $(a_{\mathcal{I}_0}, a_{\mathcal{I}_1})$, $\cdots$, $(a_{\mathcal{I}_{n-1}}, a_{\mathcal{I}_n})$, $(a_{\mathcal{I}_n}, +\infty)$ sequentially, $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ is increasing monotonously from negative to positive according to (i)-(iv). This implies that (i)-(iv) proves the convexity of optimization problem (8). As a result, the global minimum of $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ can be found within the interval that has the smallest absolute gradient value.

Building on the above analysis, we can formally prove the following Theorem.

Proof of Theorem 3. First, all the breakpoints in the set $\mathcal{S}$ can be sorted in an ascending order $\mathcal{I}$. Given any interval $(\mathcal{S}_{\mathcal{I}_k}, \mathcal{S}_{\mathcal{I}_{k+1}})$, we have $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u) = \sum_{i=0}^{k}\big|b_{\mathcal{I}_i}\big| - \sum_{j=k+1}^{n}\big|b_{\mathcal{I}_j}\big|$. In this interval, $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ is a linear function, since $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ is a constant. Thus, the local minimum of $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ can be obtained by either $\mathcal{S}_{\mathcal{I}_k}$ or $\mathcal{S}_{\mathcal{I}_{k+1}}$, and the solution (*i.e.*, one of them) is decided by the sign of $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$.

Obviously, $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ is a piecewise-linear function, when u passes through all the intervals along the entire coordinate axis of u, *i.e.*, $(-\infty, \mathcal{S}_{\mathcal{I}_0})$, $(\mathcal{S}_{\mathcal{I}_0}, \mathcal{S}_{\mathcal{I}_1})$, $\cdots$, $\big(\mathcal{S}_{\mathcal{I}_{n-1}}, \mathcal{S}_{\mathcal{I}_n}\big)$, $(\mathcal{S}_{\mathcal{I}_n}, +\infty)$. When u approaches $\pm\infty$, $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ increases to the infinity. Thus, the global minimum of the objective function $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ is obtained from one of breakpoints in the set $\mathcal{S}$, *i.e.*, $u^* = \text{argmin}_{u\in\mathcal{S}}\,\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$, which completes the proof. □

C.3 Optimality Analysis of Exact Solvers for the $\text{D}_{\boldsymbol{21}}$RPCA Model

Analysis of the Gradient and Convexity of the Loss Function. First, the loss function $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ defined in Eq. 9 is rewritten equivalently as

$$\begin{aligned}\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u) &= \big\|\widetilde{\boldsymbol{x}} - u \cdot \boldsymbol{v}\big\|_1 + \lambda\sqrt{u^2 + c} \\ &= \sum_{i=1}^{n}\big|v_i\big|\big|\tfrac{\widetilde{x}_i}{v_i} - u\big| + \lambda\sqrt{u^2 + c} \\ &= \sum_{i=0}^{n}\big|b_i\big|\big|a_i - u\big| + \lambda\sqrt{u^2 + c},\end{aligned} \tag{32}$$

where $\boldsymbol{a} = (0, \widetilde{x}_1/v_1, \cdots, \widetilde{x}_n/v_n)^T$, $\boldsymbol{b} = (\lambda, v_1, \cdots, v_n)^T$. Thus, the gradient of $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ *w.r.t.* u is computed as

$$\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u) = \sum_{i=0}^{n}\big|b_i\big| \cdot \text{sign}\,(u - a_i) + \lambda\frac{u}{\sqrt{u^2 + c}}. \tag{33}$$

As compared to $\nabla\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$, it becomes more difficult to analyze the variation tendency of $\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$, since $\nabla\sqrt{u^2 + c}$ varies in any given interval.

Following the same procedure of solving $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$, we still divide the entire coordinate axis of u into $n + 2$ intervals, using these $n + 1$ end points from the ascending order $\mathcal{I}$ of $\boldsymbol{a}$. Since $\nabla\sqrt{u^2 + c}$

changes sign from negative to positive as u crosses 0, the point $u = 0$ should be treated as one of the breakpoints.

Thus, when u is in each interval, the gradient can be analyzed as follows. (i) If $u \in (-\infty, a_{\mathcal{I}_0})$, $\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ is negative and can go to negative infinity when $u \to -\infty$. (ii) If $u \in (a_{\mathcal{I}_n}, +\infty)$, $\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ is positive and goes to positive infinity when $u \to +\infty$. (iii) however when $u \in (a_{\mathcal{I}_k}, a_{\mathcal{I}_{k+1}})$, the variation tendency of $\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ is not very obvious. As a result, we turn to make a comparison between the gradients in any two adjacent intervals. Given $u_1 \in (a_{\mathcal{I}_k}, a_{\mathcal{I}_{k+1}})$ and $u_2 \in (a_{\mathcal{I}_k+1}, a_{\mathcal{I}_{k+2}})$, we have the following inequality for the first term of $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$

$$\nabla\|\widetilde{\boldsymbol{x}} - u_1 \cdot \boldsymbol{v}\|_1 \leq \nabla\|\widetilde{\boldsymbol{x}} - u_2 \cdot \boldsymbol{v}\|_1, \tag{34}$$

according to the gradient analysis of $\mathcal{L}_{\text{D}_1\text{RPCA}}(u)$ when $u_1 \leq u_2$. On the other hand, we can establish the following inequality for the second term of $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$

$$\nabla\sqrt{(u_1)^2 + c} \leq \nabla\sqrt{(u_2)^2 + c}, \tag{35}$$

since $\nabla\sqrt{(u)^2 + c}$ is a monotonously increasing function, which is based on the fact that $\nabla^2\sqrt{(u)^2 + c} = c/\left(u^2 + c\right)^{3/2} > 0$. By combining the results from Eqs. 34-35, we reach to the following equality for the gradient of $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$:

$$\nabla\left\|\widetilde{\boldsymbol{x}} - u_1 \cdot \boldsymbol{v}\right\|_1 + \lambda \cdot \nabla\sqrt{(u_1)^2 + c} \leq \nabla\left\|\widetilde{\boldsymbol{x}} - u_2 \cdot \boldsymbol{v}\right\|_1 + \lambda \cdot \nabla\sqrt{(u_2)^2 + c}. \tag{36}$$

That implies that $\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u_1) \leq \nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u_2)$.

To summarize, when u passes through all the intervals, $\nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ is increasing monotonously from negative to positive. Thus, the global minimum of the objective function $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ lies in the interval where the gradient has the smallest absolute value, which also confirms the convexity of $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$.

Based on the above analysis, we can formally prove the following Theorem.

Proof of Theorem 4. Let $h(u) = \nabla\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ and $g(u) = \nabla\mathcal{L}_{\text{recon}}(u)$. Since $\widetilde{u} = \operatorname{argmin}_{u\in\mathcal{S}}\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$, $\widetilde{u}$ achieves the minimum of $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ among these breakpoints in the set $\mathcal{S}$, and $\widetilde{u}$ should be either the left breakpoint or the right breakpoint in certain interval, where $|h(u)|$ is the smallest. Suppose the optimal interval is $\left(\mathcal{S}_{\mathcal{I}_k}, \mathcal{S}_{\mathcal{I}_{k+1}}\right)$, if $\nabla h(\widetilde{u} \pm \epsilon) < 0$, $\widetilde{u}$ should be the left breakpoint $\mathcal{S}_{\mathcal{I}_k}$; if $\nabla h(\widetilde{u} \pm \epsilon) > 0$, $\widetilde{u}$ should be the right breakpoint $\mathcal{S}_{\mathcal{I}_{k+1}}$.

However, if $h(u)$ could not approach zero, we will not have any optimal interval. This implies that the optimal solution u^* should be the breakpoint $\widetilde{u}$, where $h(\widetilde{u} - \epsilon) < 0$ and $h(\widetilde{u} + \epsilon) > 0$. In this case, we will have $|g(\widetilde{u})| > \lambda$, since $h(\widetilde{u}) \neq 0$ and $-1 < \nabla\sqrt{\widetilde{u}^2 + c} < 1$.

On the other hand, when $|g(u)| < \lambda$, we have $h(u) = 0$ in the optimal interval $\left(\mathcal{S}_{\mathcal{I}_k}, \mathcal{S}_{\mathcal{I}_{k+1}}\right)$. If $h(\widetilde{u} \pm \epsilon) < 0$, we have $\widetilde{u} = \mathcal{S}_{\mathcal{I}_k}$ and $g(\widetilde{u} + \epsilon) = g(u^*)$, since $g(\cdot)$ is a constant in the corresponding interval. By setting $h(u^*) = 0$, we have the following equality

$$g(\widetilde{u}+\epsilon) + \lambda\frac{u^*}{\sqrt{(u^*)^2 + c}} = 0, \tag{37}$$

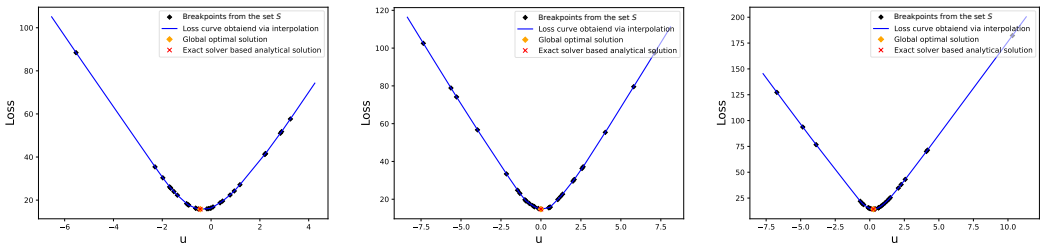
which gives the optimal solution u^* as

$$u^* = \operatorname{sign}\left(-g(\widetilde{u} + \epsilon)\right)\frac{\sqrt{c}|g(\widetilde{u} + \epsilon)|}{\sqrt{\lambda^2 - g^2(\widetilde{u} + \epsilon)}}. \tag{38}$$

Moreover, if $h(\widetilde{u} \pm \epsilon) > 0$, we have $\widetilde{u} = \mathcal{S}_{\mathcal{I}_{k+1}}$ and $g(\widetilde{u} - \epsilon) = g(u^*)$. Then, the optimal solution u^* is obtained in the similar way as

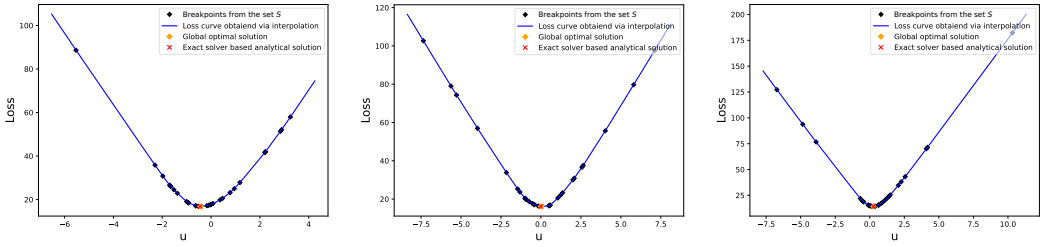
$$u^* = \operatorname{sign}\left(-g(\widetilde{u} - \epsilon)\right)\frac{\sqrt{c}|g(\widetilde{u} - \epsilon)|}{\sqrt{\lambda^2 - g^2(\widetilde{u} - \epsilon)}}. \tag{39}$$

Based on the above analysis of the three cases, we obtain the optimal solution u^* for $\mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$ as defined in Eq. 6, which completes the proof. □



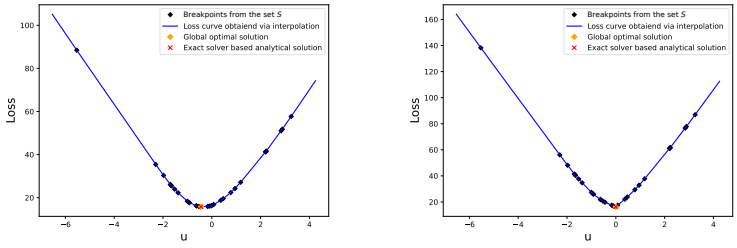
(a) $u^* < 0$ (b) $u^* = 0$ (c) $u^* > 0$

Figure 4: Comparison between the exact solver and linear interpolation when solving D_1RPCA.



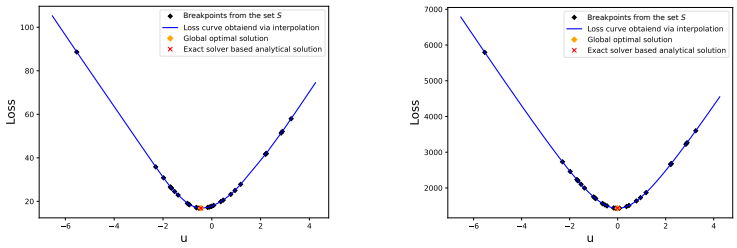
(a) $u^* < 0$ (b) $u^* = 0$ (c) $u^* > 0$

Figure 5: Comparison between the exact solver and linear interpolation when solving D_{21}RPCA.



(a) $u^* \neq 0$ (b) $u^* = 0$

Figure 6: Comparison between the exact solver and linear interpolation when solving D_1RPCA as the hyperparameter λ increases.



(a) $u^* \neq 0$ (b) $u^* = 0$

Figure 7: Comparison between the exact solver and linear interpolation when solving D_{21}RPCA as the hyperparameter λ increases.

C.4 Visualization and Analysis of Exact Solver for D_1RPCA and D_{21}RPCA Models

To validate the effectiveness of the proposed exact solvers (Eq. 10) and (Eq. 11), we compare their analytical solutions (produced by the exact solvers) with the global optimum under various settings.

Over the entire domain of the variable u, we search for the optimal solution u^* of $\mathcal{L}_{\mathrm{D}_1\mathrm{RPCA}}(u)$ and $\mathcal{L}_{\mathrm{D}_{21}\mathrm{RPCA}}(u)$ within $n+2$ intervals, which are predefined by the breakpoint set $\mathcal{S} = \{0, \widetilde{x}_1/v_1, \widetilde{x}_2/v_2, \cdots, \widetilde{x}_n/v_n\}$. As illustrated in Figs. 4-7, all the breakpoints in the set $\mathcal{S}$ are marked with the symbol "♦". A loss curve is then constructed by performing linear interpolation over the set $\mathcal{S}$, from which the reference global minimum (marked with the symbol "♦") is subsequently

identified. In parallel, the analytical solution (marked with the symbol "✗") is computed using the proposed exact solvers.

Here, we generate the breakpoint set $\mathcal{S} = \{0, \widetilde{x}_1/v_1, \widetilde{x}_2/v_2, \cdots, \widetilde{x}_n/v_n\}$ with $n = 30$ and fix $c = 2$, following the variable settings of $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$, and c in the simplified D_1RPCA and D_{21}RPCA subproblems (8) and (9). As shown in Figs. 4-5, the proposed exact solver consistently identifies the global optimum (marked with the symbol "✗") for both D_1RPCA and D_{21}RPCA models across various scenarios, such as $u^* < 0$, $u^* = 0$, and $u^* > 0$. Moreover, Figs. 6–7 demonstrate that when the original global optimum is nonzero, increasing the regularization parameter λ effectively guides the solution toward zero, reflecting the sparsity-inducing effect of the regularizer. Consistent with previous cases, the proposed exact solver successfully identifies the global optimum.

D Hybrid Alternating Optimization Algorithm

We develop a hybrid alternating optimization algorithm that combines matrix-wise ADMM updates with element-wise exact solver (ES) refinement. Both stages optimize the following regularized objective:

$$\mathcal{F}_p(\boldsymbol{U}, \boldsymbol{V}) = \|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\|_1 + \lambda\|\boldsymbol{U}\|_p + \lambda\|\boldsymbol{V}\|_p, \tag{40}$$

where $p = 1$ and $p = 21$ correspond to D_1RPCA and D_{21}RPCA, respectively. The ADMM stage updates each factor matrix as a whole to obtain a favorable starting point that recovers the global low-rank structure, and the ES stage then refines this solution through successive exact element-wise updates.

In the following, we first describe these two stages, then present their integration, and finally establish the convergence of the ES refinement stage.

D.1 Matrix-Wise Update via ADMM

In this subsection, we present the optimization procedures of the ADMM-based algorithm for solving the DRPCA problem.

D.1.1 Lagrangian Function

First, we introduce three auxiliary variables $\boldsymbol{E}$, $\boldsymbol{F}$, and $\boldsymbol{G}$ in Eq. 7 to make the optimization separable between loss and penalty terms. Thus, the Lagrangian function can be defined as

$$\begin{aligned}\mathcal{L}(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{E}, \boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Omega}, \boldsymbol{\Sigma}, \boldsymbol{\Lambda}, \mu) = \big\|\boldsymbol{E}\big\|_1 + \lambda\big\|\boldsymbol{F}\big\|_p + \lambda\big\|\boldsymbol{G}\big\|_p \\ + \big\langle\boldsymbol{\Omega}, \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{E}\big\rangle + \tfrac{\mu}{2}\big\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{E}\big\|_F^2 \\ + \big\langle\boldsymbol{\Sigma}, \boldsymbol{U} - \boldsymbol{F}\big\rangle + \tfrac{\mu}{2}\big\|\boldsymbol{U} - \boldsymbol{F}\big\|_F^2 \\ + \big\langle\boldsymbol{\Lambda}, \boldsymbol{V} - \boldsymbol{G}\big\rangle + \tfrac{\mu}{2}\big\|\boldsymbol{V} - \boldsymbol{G}\big\|_F^2,\end{aligned} \tag{41}$$

where $p \in \{1, 21\}$, $\langle\cdot,\cdot\rangle$ denotes the inner product, *i.e.*, $\langle\boldsymbol{A}, \boldsymbol{B}\rangle = \mathrm{tr}(\boldsymbol{A}^T\boldsymbol{B})$, μ denotes the penalty parameter, and $\boldsymbol{\Omega}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Lambda}$ denotes the Lagrangian multipliers.

Then, Eq. (41) can be further rewritten to the following equivalent formulation:

$$\begin{aligned}\min_{\boldsymbol{U},\boldsymbol{V},\boldsymbol{E},\boldsymbol{F},\boldsymbol{G}} \quad & \big\|\boldsymbol{E}\big\|_1 + \lambda\big\|\boldsymbol{F}\big\|_p + \lambda\big\|\boldsymbol{G}\big\|_p + \tfrac{\mu}{2}\big\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{E} + \boldsymbol{\Omega}/\mu\big\|_F^2 \\ & + \tfrac{\mu}{2}\big\|\boldsymbol{U} - \boldsymbol{F} + \boldsymbol{\Sigma}/\mu\big\|_F^2 + \tfrac{\mu}{2}\big\|\boldsymbol{V} - \boldsymbol{G} + \boldsymbol{\Lambda}/\mu\big\|_F^2.\end{aligned} \tag{42}$$

As compared to the original optimization problem (7), three sparsity-inducing loss terms are decoupled in Eq. 42. Then, we can employ the alternating minimization method to solve problem (42), which can be decomposed to several subproblems *w.r.t.* to one unknown variable while fixing the rest of unknown variables. Details of the optimization *w.r.t.* each subproblem are explained in the following.

D.1.2 SUBPROBLEM OPTIMIZATION

Solving for $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{G}$. The optimization *w.r.t.* $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{G}$ is decomposed as follows. (i) Fixing $\boldsymbol{F}$, $\boldsymbol{G}$, $\boldsymbol{U}$, $\boldsymbol{V}$, the optimization *w.r.t.* $\boldsymbol{E}$ becomes to:

$$\min_{\boldsymbol{E}} \ \left\|\boldsymbol{E}\right\|_1 + \frac{\mu}{2}\left\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{E} + \boldsymbol{\Omega}/\mu\right\|_F^2. \tag{43}$$

The optimal solutions of $\boldsymbol{E}^*$ is obtained as follows:

$$\boldsymbol{E}^* = \text{sign}\left(\widetilde{\boldsymbol{X}}\right) \odot \max\left(\left|\widetilde{\boldsymbol{X}}\right| - 1/\mu, 0\right), \tag{44}$$

where $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T + \boldsymbol{\Omega}/\mu$, and $\odot$ is the Hardmard (element-wise) multiplication. (ii) Fixing $\boldsymbol{E}$, $\boldsymbol{G}$, $\boldsymbol{U}$, $\boldsymbol{V}$, the optimization *w.r.t.* $\boldsymbol{F}$ becomes to

$$\min_{\boldsymbol{F}} \ \lambda\left\|\boldsymbol{F}\right\|_p + \frac{\mu}{2}\left\|\boldsymbol{U} - \boldsymbol{F} + \boldsymbol{\Sigma}/\mu\right\|_F^2. \tag{45}$$

The optimal solutions of $\boldsymbol{F}^*$ is obtained as follows:

$$\boldsymbol{F}^* = \text{sign}\left(\widetilde{\boldsymbol{U}}\right) \odot \max\left(\left|\widetilde{\boldsymbol{U}}\right| - \lambda/\mu, 0\right), \quad \text{for } p = 1, \tag{46}$$

$$\boldsymbol{F}^* = \left(\boldsymbol{1}\boldsymbol{a}^T\right) \odot \widetilde{\boldsymbol{U}}, \quad \text{for } p = 21, \tag{47}$$

where $\widetilde{\boldsymbol{U}} = \boldsymbol{U} + \boldsymbol{\Sigma}/\mu$, $\boldsymbol{1}$ is the column vector with all ones, and $\boldsymbol{a}$ is the column vector with $a_i = \max(1 - (\lambda/\mu)/\|\widetilde{\boldsymbol{u}}_i\|_2, 0)$. (iii) Fixing $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{U}$, $\boldsymbol{V}$, the optimization *w.r.t.* $\boldsymbol{G}$ becomes to

$$\min_{\boldsymbol{G}} \ \lambda\left\|\boldsymbol{G}\right\|_p + \frac{\mu}{2}\left\|\boldsymbol{V} - \boldsymbol{G} + \boldsymbol{\Lambda}/\mu\right\|_F^2. \tag{48}$$

The optimal solutions of $\boldsymbol{E}^*$ is obtained as follows:

$$\boldsymbol{G}^* = \text{sign}\left(\widetilde{\boldsymbol{V}}\right) \odot \max\left(\left|\widetilde{\boldsymbol{V}}\right| - \lambda/\mu, 0\right), \quad \text{for } p = 1, \tag{49}$$

$$\boldsymbol{G}^* = \left(\boldsymbol{1}\boldsymbol{b}^T\right) \odot \widetilde{\boldsymbol{V}}, \quad \text{for } p = 21, \tag{50}$$

where $\widetilde{\boldsymbol{V}} = \boldsymbol{V} + \boldsymbol{\Lambda}/\mu$, $\boldsymbol{1}$ is the column vector with all ones, and $\boldsymbol{b}$ is the column vector with $b_i = \max(1 - (\lambda/\mu)/\|\widetilde{\boldsymbol{v}}_i\|_2, 0)$.

Solving for $\boldsymbol{U}$, $\boldsymbol{V}$. The optimization *w.r.t.* $\boldsymbol{U}$, $\boldsymbol{V}$ is decomposed as follows. (i) Fixing $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{G}$, $\boldsymbol{V}$, the optimization *w.r.t.* $\boldsymbol{U}$ becomes to:

$$\min_{\boldsymbol{U}} \ \frac{\mu}{2}\left\|\widetilde{\boldsymbol{X}} - \boldsymbol{U}\boldsymbol{V}^T\right\|_F^2 + \frac{\mu}{2}\left\|\boldsymbol{U} - \boldsymbol{C}\right\|_F^2, \tag{51}$$

where $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{E} + \boldsymbol{\Omega}/\mu$, $\boldsymbol{C} = \boldsymbol{F} - \boldsymbol{\Sigma}/\mu$. By setting derivatives of Eq. 51 *w.r.t.* $\boldsymbol{U}$ to zero, the optimal solutions of $\boldsymbol{U}^*$ is obtained as follows:

$$\boldsymbol{U}^* = \left(\boldsymbol{C} + \widetilde{\boldsymbol{X}}\boldsymbol{V}\right)\left(\boldsymbol{V}^T\boldsymbol{V} + \boldsymbol{I}\right)^{-1}, \tag{52}$$

where $\boldsymbol{I}$ is the k-by-k identity matrix. (ii) Fixing $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{G}$, $\boldsymbol{U}$, the optimization *w.r.t.* $\boldsymbol{V}$ becomes to

$$\min_{\boldsymbol{V}} \ \frac{\mu}{2}\left\|\widetilde{\boldsymbol{X}} - \boldsymbol{U}\boldsymbol{V}^T\right\|_F^2 + \frac{\mu}{2}\left\|\boldsymbol{V} - \boldsymbol{D}\right\|_F^2, \tag{53}$$

where $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{E} + \boldsymbol{\Omega}/\mu$, $\boldsymbol{D} = \boldsymbol{G} - \boldsymbol{\Lambda}/\mu$. By setting derivatives of Eq. 53 *w.r.t.* $\boldsymbol{V}$ to zero, the optimal solutions of $\boldsymbol{V}^*$ is obtained as follows:

$$\boldsymbol{V}^* = \left(\boldsymbol{D} + \widetilde{\boldsymbol{X}}^T\boldsymbol{U}\right)\left(\boldsymbol{U}^T\boldsymbol{U} + \boldsymbol{I}\right)^{-1}, \tag{54}$$

where $\boldsymbol{I}$ is the k-by-k identity matrix.

Updating Multipliers and Parameters $\boldsymbol{\Omega}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Lambda}$, μ. Finally, we update multipliers and parameters $\boldsymbol{\Omega}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Lambda}$, ν at the end of each iteration as the following:

$$\boldsymbol{\Omega} = \boldsymbol{\Omega} + \mu \cdot \left(\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{E}\right), \tag{55}$$

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma} + \mu \cdot \left(\boldsymbol{U} - \boldsymbol{F}\right), \tag{56}$$

$$\boldsymbol{\Lambda} = \boldsymbol{\Lambda} + \mu \cdot \left(\boldsymbol{V} - \boldsymbol{G}\right), \tag{57}$$

$$\mu = \rho \cdot \mu. \tag{58}$$

Algorithm 1 ADMM-Based Optimization Algorithm

Input: data matrix $\boldsymbol{X} \in \mathbb{R}^{d\times n}$, rank k, hyperparameter λ, model type p, convergence threshold ϵ, maximum number of iterations T
Output: principal directions $\boldsymbol{U} \in \mathbb{R}^{d\times k}$, principal components $\boldsymbol{V} \in \mathbb{R}^{n\times k}$

Parameter settings: $\boldsymbol{E} = 0$, $\boldsymbol{F} = 0$, $\boldsymbol{G} = 0$, $\boldsymbol{\Omega} = 0$, $\boldsymbol{\Sigma} = 0$, $\boldsymbol{\Lambda} = 0$, $t = 0$, $\rho = 1.1$, $\mu = 1/\sum_{ij} \boldsymbol{X}_{ij}^2$
Initialization: $(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}) = \text{svd}(\boldsymbol{X}, k)$, $\mathcal{L}^{(t)} = \mathcal{F}_p(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)})$
repeat
  Computing $\boldsymbol{E}^{(t+1)}$ using Eq. 44
  Computing $\boldsymbol{F}^{(t+1)}$ based on p using Eq. 46 or Eq. 47.
  Computing $\boldsymbol{G}^{(t+1)}$ based on p using Eq. 49 or Eq. 50.
  Computing $\boldsymbol{U}^{(t+1)}$ using Eq. 52.
  Computing $\boldsymbol{V}^{(t+1)}$ using Eq. 54.
  Updating $\boldsymbol{\Omega}^{(t+1)}$ using Eq. 55.
  Updating $\boldsymbol{\Sigma}^{(t+1)}$ using Eq. 56.
  Updating $\boldsymbol{\Lambda}^{(t+1)}$ using Eq. 57.
  Updating $\mu^{(t+1)}$ using Eq. 58.
  Computing $\mathcal{L}^{(t+1)} = \mathcal{F}_p(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)})$
  $t = t + 1$.
until $t > T$ or $|\mathcal{L}^{(t)} - \mathcal{L}^{(t-1)}|/\mathcal{L}^{(t-1)} < \epsilon$
return Optimal solutions: $\boldsymbol{U}^{(t)}$, $\boldsymbol{V}^{(t)}$

D.1.3 The Summary of ADMM-Based Optimization Algorithm

The complete ADMM-based optimization algorithm for solving the D_1RPCA and D_{21}RPCA problems is summarized in Algorithm 1. By employing the alternating optimization technique, the optimal solutions for $\boldsymbol{U}$ and $\boldsymbol{V}$ are obtained through an iterative process.

First, SVD is applied to initialize the low-rank variables $\boldsymbol{U}$ and $\boldsymbol{V}$. After computing the optimal solutions for $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{G}$, $\boldsymbol{U}$, and $\boldsymbol{V}$, the multipliers and parameters $\boldsymbol{\Omega}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Lambda}$, and μ are updated at the end of each iteration. The proposed optimization algorithm terminates when the reduction in the overall loss function $\mathcal{L}$ becomes sufficiently small, or when the maximum number of iterations is reached.

D.2 Element-Wise Update via Exact Solvers

In this subsection, we detail how the proposed exact solvers are employed to minimize the D_1RPCA and D_{21}RPCA models. The proposed alternating optimization algorithm, based on the exact solver, is summarized in Algorithms 2 and 3.

Algorithm 2 presents the main training loop. At the initialization stage, various strategies (*e.g.*, random, SVD-based, or ADMM-based) can be utilized to initialize the low-rank components $\boldsymbol{U}$ and $\boldsymbol{V}$. In practice, we employ ADMM-based initialization to improve computational efficiency and provide a good starting point.

Next, $\boldsymbol{U}$ and $\boldsymbol{V}$ are alternately minimized using the "update()" subroutine described in Algorithm 3, where exact solvers *w.r.t.* a single entry are repeatedly applied to all entries of the corresponding matrix. Based on the specified model type p (*i.e.*, "1" or "21"), Algorithm 3 applies the exact solvers defined in Eqs. 10-11 to compute the optimal solutions of the simplified subproblems for the D_1RPCA and D_{21}RPCA models, respectively.

Finally, in Algorithm 2, the optimization process terminates once the convergence condition is met, which is determined by either the change in the overall loss or reaching the maximum number of iterations. In each iteration, the overall loss of D_1RPCA and D_{21}RPCA is computed based on the specified model type p.

Algorithm 2 Exact Solver-Based Alternating Optimization Algorithm

Input: data matrix $\boldsymbol{X} \in \mathbb{R}^{d\times n}$, rank k, hyperparameter λ, model type p, convergence threshold ϵ, maximum number of iterations T
Output: principal directions $\boldsymbol{U} \in \mathbb{R}^{d\times k}$, principal components $\boldsymbol{V} \in \mathbb{R}^{n\times k}$

Initialization: $t = 0$, $\boldsymbol{Y} = \boldsymbol{X}^T$, $\boldsymbol{U}^{(0)}$, $\boldsymbol{V}^{(0)}$, $\mathcal{L}^{(0)} = \mathcal{F}_p(\boldsymbol{U}^{(0)}, \boldsymbol{V}^{(0)})$
repeat
  $\boldsymbol{U}^{(t+1)} = \text{update}(\boldsymbol{X}, \boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}, d, k, \lambda, p)$
  $\boldsymbol{V}^{(t+1)} = \text{update}(\boldsymbol{Y}, \boldsymbol{V}^{(t)}, \boldsymbol{U}^{(t+1)}, n, k, \lambda, p)$
  $\mathcal{L}^{(t+1)} = \mathcal{F}_p(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)})$
  $t = t + 1$
until $t > T$ or $|\mathcal{L}^{(t)} - \mathcal{L}^{(t-1)}|/\mathcal{L}^{(t-1)} < \epsilon$
return $\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}$

Algorithm 3 Exact Solver: update$(\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{V}, d, k, \lambda, p)$

Input: $\boldsymbol{X}$, $\boldsymbol{U}$, $\boldsymbol{V}$, d, k, λ, p
Output: $\boldsymbol{U}$

for $i_0 = 1$ to d **do**
  for $h_0 = 1$ to k **do**
    $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T + \boldsymbol{u}_{h_0}(\boldsymbol{v}_{h_0})^T$
    $\widetilde{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}^{i_0})^T$
    $\boldsymbol{v} = \boldsymbol{v}_{h_0}$
    $c = \|\boldsymbol{u}_{h_0}\|_2^2 - U_{i_0 h_0}^2$
    Update $U_{i_0 h_0}$ based on p using Eq. 10 or Eq. 11.
  end for
end for
return $\boldsymbol{U}$

Algorithm 4 Hybrid Alternating Optimization with ADMM Initialization and ES Refinement

Input: $\boldsymbol{X} \in \mathbb{R}^{d\times n}$, k, $\lambda > 0$, $p \in \{1, 21\}$, tolerances $\epsilon_{\mathrm{A}}, \epsilon_{\mathrm{E}} > 0$, maximum iterations $T_{\mathrm{A}}, T_{\mathrm{E}} \geq 1$
Output: refined factors $\boldsymbol{U}$ and $\boldsymbol{V}$

// Matrix-Wise Update via ADMM
Obtain $(\boldsymbol{U}, \boldsymbol{V}) = (\boldsymbol{U}_{\mathrm{A}}, \boldsymbol{V}_{\mathrm{A}})$ using Algorithm 1 with ϵ_{A} and T_{A}
// Element-Wise Update via ES
for $t = 1$ to T_{E} **do**
  Set $f_{\mathrm{old}} = \mathcal{F}_p(\boldsymbol{U}, \boldsymbol{V})$
  $\boldsymbol{U} \leftarrow \mathrm{update}(\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{V}, d, k, \lambda, p)$ using Algorithm 3
  $\boldsymbol{V} \leftarrow \mathrm{update}(\boldsymbol{X}^T, \boldsymbol{V}, \boldsymbol{U}, n, k, \lambda, p)$ using Algorithm 3
  Set $f_{\mathrm{new}} = \mathcal{F}_p(\boldsymbol{U}, \boldsymbol{V})$
  if $(f_{\mathrm{old}} - f_{\mathrm{new}})/\max\{1, f_{\mathrm{old}}\} < \epsilon_{\mathrm{E}}$ **then**
    break
  end if
end for
return $(\boldsymbol{U}, \boldsymbol{V})$

D.3 Hybrid Optimization Procedure

Algorithm 4 combines the two stages in sequence. In the first stage, we first run Algorithm 1 to obtain $(\boldsymbol{U}_{\mathrm{A}}, \boldsymbol{V}_{\mathrm{A}})$ which serves as the starting point for ES refinement. In the second stage, Algorithm 3 is applied in each iteration first to $\boldsymbol{U}$ and then to $\boldsymbol{V}$, with the newly updated $\boldsymbol{U}$ held fixed.

Thus, ADMM is executed only once for initialization, whereas the two factor matrices are updated alternately during the ES stage. Refinement terminates when the normalized objective reduction falls below a prescribed tolerance or the maximum number of iterations is reached. In practice, the ES stage typically converges within five alternating iterations.

D.4 Convergence Analysis of Exact Solver-Based Alternating Optimization Algorithm

Next, we analyze the convergence of the ES refinement stage, showing that it monotonically decreases an upper bound on the worst-case DRPCA loss. For notional simplicity, we define the original loss J_{orig} and the auxiliary loss J_{aux} as follows:

$$J_{orig}(\boldsymbol{U}, \Delta\boldsymbol{U}, \boldsymbol{V}, \Delta\boldsymbol{V}) = \left\|\boldsymbol{X} - (\boldsymbol{U} + \Delta\boldsymbol{U})(\boldsymbol{V} + \Delta\boldsymbol{V})^T\right\|_1, \tag{59}$$

$$J_{aux}(\boldsymbol{U}, \boldsymbol{V}) = \left\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\right\|_1 + \lambda\left\|\boldsymbol{U}\right\|_1 + \lambda\left\|\boldsymbol{V}\right\|_1 + k\lambda^2 \quad (\ell_1\text{-based upper bound}), \tag{60}$$

$$J_{aux}(\boldsymbol{U}, \boldsymbol{V}) = \left\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T\right\|_1 + \lambda\left\|\boldsymbol{U}\right\|_{21} + \lambda\left\|\boldsymbol{V}\right\|_{21} + \frac{k\lambda^2}{\sqrt{nd}} \quad (\ell_{21}\text{-based upper bound}). \tag{61}$$

Theorem 5. *Let $\{(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)})\}_{t\geq 0}$ be the iterates generated by the ES refinement stage. Then $J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)}) \leq J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)})$, $\forall\, t \geq 0$, and the sequence $\{J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)})\}_{t\geq 0}$ converges. Consequently, the worst-case DRPCA loss, i.e., $\max_{\Delta\boldsymbol{U}, \Delta\boldsymbol{V}} J_{orig}(\boldsymbol{U}^{(t)}, \Delta\boldsymbol{U}, \boldsymbol{V}^{(t)}, \Delta\boldsymbol{V})$, is controlled by a monotonically non-increasing upper bound.*

Proof of Theorem 5. Recall that DRPCA aims to minimize the worst-case reconstruction loss $\max_{\Delta\boldsymbol{U}, \Delta\boldsymbol{V}} J_{orig}(\boldsymbol{U}, \Delta\boldsymbol{U}, \boldsymbol{V}, \Delta\boldsymbol{V})$ over feasible latent perturbations, which is intractable to optimize directly. For any $(\boldsymbol{U}, \boldsymbol{V})$, the loss under any feasible perturbations $(\Delta\boldsymbol{U}, \Delta\boldsymbol{V})$ cannot exceed the worst-case loss, which is further upper-bounded by J_{aux} according to Theorems 1 and 2, *i.e.*,

$$J_{orig}(\boldsymbol{U}, \Delta\boldsymbol{U}, \boldsymbol{V}, \Delta\boldsymbol{V}) \leq \max_{\Delta\boldsymbol{U}, \Delta\boldsymbol{V}} J_{orig}(\boldsymbol{U}, \Delta\boldsymbol{U}, \boldsymbol{V}, \Delta\boldsymbol{V}) \leq J_{aux}(\boldsymbol{U}, \boldsymbol{V}), \tag{62}$$

where the first inequality follows from the definition of the maximum, and the second shows that J_{aux} upper-bounds the worst-case loss. Thus, J_{aux} serves as a tractable surrogate for the DRPCA objective, and we next show that the ES stage monotonically decreases this surrogate.

In the t-th iteration, the entries of $\boldsymbol{U}$ are updated sequentially in the order $U_{1,1}, U_{1,2}, \cdots, U_{d,k}$ with $\boldsymbol{V}^{(t)}$ fixed. By Lemmas 1 and 2, each update solves the element-wise subproblem of J_{aux} exactly. Since the previous value of the updated entry is itself a feasible candidate, each single-entry update does not increase J_{aux}, which leads to the following chain:

$$\begin{aligned}
J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t)}) &= J_{aux}\big(\{U_{1,1}^{(t+1)}, U_{1,2}^{(t+1)}, \cdots, U_{d,k-1}^{(t+1)}, U_{d,k}^{(t+1)}\}, \boldsymbol{V}^{(t)}\big) \\
&\leq J_{aux}\big(\{U_{1,1}^{(t+1)}, U_{1,2}^{(t+1)}, \cdots, U_{d,k-1}^{(t+1)}, U_{d,k}^{(t)}\}, \boldsymbol{V}^{(t)}\big) \\
&\quad\vdots \\
&\leq J_{aux}\big(\{U_{1,1}^{(t+1)}, U_{1,2}^{(t+1)}, U_{1,3}^{(t)}, \cdots, U_{d,k}^{(t)}\}, \boldsymbol{V}^{(t)}\big) \\
&\leq J_{aux}\big(\{U_{1,1}^{(t+1)}, U_{1,2}^{(t)}, U_{1,3}^{(t)}, \cdots, U_{d,k}^{(t)}\}, \boldsymbol{V}^{(t)}\big) \\
&\leq J_{aux}\big(\{U_{1,1}^{(t)}, U_{1,2}^{(t)}, U_{1,3}^{(t)}, \cdots, U_{d,k}^{(t)}\}, \boldsymbol{V}^{(t)}\big) = J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}).
\end{aligned} \tag{63}$$

That is,

$$J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t)}) \leq J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}). \tag{64}$$

Next, the entries of $\boldsymbol{V}$ are visited sequentially in the order $V_{1,1}, V_{1,2}, ; V_{n,k}$ with $\boldsymbol{U}^{(t+1)}$ fixed. For each entry selected by prescreening, the exact solver likewise does not increase J_{aux}. For each unselected entry, $V_{j,h}^{(t+1)} = V_{j,h}^{(t)}$, and the corresponding relation holds with equality. This yields

$$\begin{aligned}
J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)}) &= J_{aux}\big(\boldsymbol{U}^{(t+1)}, \{V_{1,1}^{(t+1)}, V_{1,2}^{(t+1)}, \cdots, V_{n,k-1}^{(t+1)}, V_{n,k}^{(t+1)}\}\big) \\
&\leq J_{aux}\big(\boldsymbol{U}^{(t+1)}, \{V_{1,1}^{(t+1)}, V_{1,2}^{(t+1)}, \cdots, V_{n,k-1}^{(t+1)}, V_{n,k}^{(t)}\}\big) \\
&\quad\vdots \\
&\leq J_{aux}\big(\boldsymbol{U}^{(t+1)}, \{V_{1,1}^{(t+1)}, V_{1,2}^{(t+1)}, V_{1,3}^{(t)}, \cdots, V_{n,k}^{(t)}\}\big) \\
&\leq J_{aux}\big(\boldsymbol{U}^{(t+1)}, \{V_{1,1}^{(t+1)}, V_{1,2}^{(t)}, V_{1,3}^{(t)}, \cdots, V_{n,k}^{(t)}\}\big) \\
&\leq J_{aux}\big(\boldsymbol{U}^{(t+1)}, \{V_{1,1}^{(t)}, V_{1,2}^{(t)}, V_{1,3}^{(t)}, \cdots, V_{n,k}^{(t)}\}\big) = J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t)}).
\end{aligned} \tag{65}$$

That is,

$$J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)}) \leq J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t)}). \tag{66}$$

Combining Eqs. 64 and 66, one full alternating iteration satisfies

$$J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)}) \leq J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t)}) \leq J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}). \tag{67}$$

Hence, $\{J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)})\}_{t\geq 0}$ is monotonically non-increasing. Since J_{aux} is bounded below by a nonnegative constant, this sequence converges. Finally, combining Eqs. 62 and 67, we have

$$\max_{\Delta \boldsymbol{U}, \Delta \boldsymbol{V}} J_{orig}(\boldsymbol{U}^{(t+1)}, \Delta \boldsymbol{U}, \boldsymbol{V}^{(t+1)}, \Delta \boldsymbol{V}) \leq J_{aux}(\boldsymbol{U}^{(t+1)}, \boldsymbol{V}^{(t+1)}) \leq J_{aux}(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)}), \tag{68}$$

indicating that the worst-case DRPCA loss is controlled by a monotonically non-increasing upper bound throughout the ES stage. This completes the proof. □

E Accelerated Exact Solvers

This section provides the proofs and implementation details for Section 4.4. We first summarize the baseline that exhaustively evaluates all breakpoints, and then establish the optimality of the weighted median for the ℓ_1 subproblem and the validity of the bisection procedure for the $\ell_{2,1}$ subproblem. The latter returns an exact minimizer if the subgradient optimality test is satisfied, and otherwise a numerical approximation with an explicit error bound. We further discuss boundary cases, the hybrid ADMM-ES schedule, and the associated computational costs. Throughout this section, $\lambda \geq 0$, $c \geq 0$, and$0 < \gamma \leq 1$.

E.1 Exhaustive Breakpoint Baseline

For a single entry u, recall from Section 4.4 that, up to an additive constant, the reconstruction loss can be written as $\sum_{j\in\mathcal{S}_{nz}} w_j|u - r_j|$, where $r_j = \widetilde{x}_j/v_j$, $w_j = |v_j|$, and $\mathcal{S}_{nz} = \{j : v_j \neq 0\}$ with $|\mathcal{S}_{nz}| = q$. Let $\mathcal{S} = \{a_1, \ldots, a_m\}$ denote the set of distinct breakpoints $\{r_j\}_{j\in\mathcal{S}_{nz}} \cup \{0\}$, where $m \leq q + 1$. For the ℓ_1 subproblem, the exhaustive solver evaluates

$$\mathcal{L}_{\text{D}_1\text{RPCA}}(a_i) = \sum_{j\in\mathcal{S}_{nz}} w_j|a_i - r_j| + \lambda|a_i|, \qquad i = 1, \ldots, m, \tag{69}$$

and returns $u^* = \operatorname{argmin}_{u\in\mathcal{S}} \mathcal{L}_{\text{D}_1\text{RPCA}}(u)$, which is optimal by Theorem 3. For the ℓ_{21} subproblem, the solver similarly evaluates

$$\mathcal{L}_{\text{D}_{21}\text{RPCA}}(a_i) = \sum_{j\in\mathcal{S}_{nz}} w_j|a_i - r_j| + \lambda\sqrt{a_i^2 + c}, \qquad i = 1, \ldots, m, \tag{70}$$

to obtain $\widetilde{u} = \operatorname{argmin}_{u\in\mathcal{S}} \mathcal{L}_{\text{D}_{21}\text{RPCA}}(u)$. However, since the regularizer $\lambda\sqrt{u^2 + c}$ is nonlinear within each interval, the minimizer may lie strictly between two breakpoints. In this case, u^* is determined from $\widetilde{u}$ via the interval search in Theorem 4, as detailed in Appendix C.3.

Since each of the $m = O(q)$ candidates requires $O(q)$ time to evaluate, the exhaustive baseline costs $O(q^2)$ time per subproblem. Moreover, a vectorized implementation that evaluates all candidates simultaneously materializes an $m \times q$ residual matrix and thus requires $O(q^2)$ temporary memory.

E.2 Weighted-Median Solver for the ℓ_1 Subproblem

As shown in Section 4.4, up to an additive constant, the D_1RPCA subproblem in Eq. 8 can be written as

$$\mathcal{L}_{\text{D}_1\text{RPCA}}(u) = \sum_{j\in\mathcal{S}_{nz}} w_j|u - r_j| + \lambda|u|, \qquad r_j = \frac{\widetilde{x}_j}{v_j}, \quad w_j = |v_j|, \tag{71}$$

where $\mathcal{S}_{nz} = \{j : v_j \neq 0\}$ with $|\mathcal{S}_{nz}| = q$ indexes the residual terms that depend on u. Each ratio r_j is a breakpoint at which the sign of $\widetilde{x}_j - u\, v_j$ changes, and w_j measures the contribution of this breakpoint to the objective. Therefore, Eq. 71 converts the element-wise residual minimization into a weighted absolute-deviation problem.

Moreover, the regularizer can be represented by an additional breakpoint-weight pair $(r_0, w_0) = (0, \lambda)$, which yields

$$\mathcal{L}_{\mathrm{D_1RPCA}}(u) = \sum_{j \in \mathcal{S}_{nz} \cup \{0\}} w_j |u - r_j|. \tag{72}$$

This augmented pair has a direct interpretation: the regularizer $\lambda|u|$ pulls u toward the breakpoint at zero with weight λ. Consequently, the reconstruction term and the regularizer can be handled jointly by a single weighted-median operation.

Lemma 5 (Weighted-Median Optimality)**.** *Let $\{(\widehat{r}_{(i)}, \widehat{w}_{(i)})\}_{i=0}^{q}$ denote the augmented breakpoint-weight pairs sorted in ascending order of the breakpoints, and let $\widehat{W} = \sum_{i=0}^{q} \widehat{w}_{(i)}$. Define*

$$\ell^* = \min \left\{ \ell : \sum_{i=0}^{\ell} \widehat{w}_{(i)} \geq \frac{\widehat{W}}{2} \right\}. \tag{73}$$

Then $u^* = \widehat{r}_{(\ell^*)}$ is a global minimizer of Eq. 71.

Proof of Lemma 5. For any u, let

$$W_<(u) = \sum_{r_j < u} w_j, \qquad W_>(u) = \sum_{r_j > u} w_j, \qquad W_=(u) = \widehat{W} - W_<(u) - W_>(u) \tag{74}$$

denote the total weights strictly to the left of, strictly to the right of, and located at u, respectively, where the sums run over $j \in \mathcal{S}_{nz} \cup \{0\}$. Since Eq. 72 is convex, its subdifferential is

$$\partial \mathcal{L}_{\mathrm{D_1RPCA}}(u) = \left[W_<(u) - W_>(u) - W_=(u),\ W_<(u) - W_>(u) + W_=(u) \right], \tag{75}$$

and the optimality condition $0 \in \partial \mathcal{L}_{\mathrm{D_1RPCA}}(u)$ is equivalent to

$$W_<(u) \leq \frac{\widehat{W}}{2} \quad \text{and} \quad W_>(u) \leq \frac{\widehat{W}}{2}. \tag{76}$$

Now consider $u^* = \widehat{r}_{(\ell^*)}$. By the minimality of ℓ^* in Eq. 73, we have $W_<(u^*) \leq \sum_{i=0}^{\ell^*-1} \widehat{w}_{(i)} < \widehat{W}/2$. Moreover, since $\sum_{i=0}^{\ell^*} \widehat{w}_{(i)} \geq \widehat{W}/2$, we have $W_>(u^*) \leq \widehat{W} - \sum_{i=0}^{\ell^*} \widehat{w}_{(i)} \leq \widehat{W}/2$. Hence, Eq. 76 holds at u^*, *i.e.*, $0 \in \partial \mathcal{L}_{\mathrm{D_1RPCA}}(u^*)$, which proves that u^* is a global minimizer. □

Lemma 5 shows that the exhaustive evaluation in Eq. 69 is unnecessary. Sorting the augmented breakpoints and accumulating their weights in a single pass returns an exact minimizer in $O(q \log q)$ time and $O(q)$ memory. Therefore, the acceleration changes only how the optimum is located, without modifying the element-wise objective or introducing any approximation.

E.3 Bisection Solver for the $\ell_{2,1}$ Subproblem

As shown in Section 4.4, up to an additive constant, the $\mathrm{D_{21}RPCA}$ subproblem in Eq. 9 can be written as

$$\mathcal{L}_{\mathrm{D_{21}RPCA}}(u) = \sum_{j \in \mathcal{S}_{nz}} w_j |u - r_j| + \lambda \sqrt{u^2 + c}, \qquad c \geq 0, \tag{77}$$

where c is the sum of squares of the other entries in the same column of $\boldsymbol{U}$ (or $\boldsymbol{V}$). Since these entries are fixed during a single element-wise update, c is a nonnegative constant and u is the only optimization variable. The first term is the same piecewise-linear reconstruction loss as in the ℓ_1 case, whereas the second term couples u with the remaining entries in its column. When $c = 0$, the regularizer reduces to $\lambda|u|$, and the problem is solved exactly by Lemma 5; we therefore focus on $c > 0$ below.

One-Sided Derivatives. Let $W = \sum_{j \in \mathcal{S}_{nz}} w_j$, and for any u, let $P_<(u) = \sum_{r_j < u} w_j$ and $P_\leq(u) = \sum_{r_j \leq u} w_j$ denote the cumulative weights strictly to the left of u and up to u, respectively. Since the regularizer is differentiable for $c > 0$, the left and right derivatives of Eq. 77 are

$$\begin{aligned} g_-(u) &= 2P_<(u) - W + \lambda \frac{u}{\sqrt{u^2 + c}}, \\ g_+(u) &= 2P_\leq(u) - W + \lambda \frac{u}{\sqrt{u^2 + c}}, \end{aligned} \tag{78}$$

which coincide with Eq. 15 in the main text. Here, $2P_<(u) - W$ and $2P_\leq(u) - W$ are the left and right slopes of the reconstruction term, and the last term is the derivative of the regularizer. The subdifferential is

$$\partial\mathcal{L}_{\mathrm{D}_{21}\mathrm{RPCA}}(u) = [g_-(u), g_+(u)], \tag{79}$$

so that u is optimal if and only if $g_-(u) \leq 0 \leq g_+(u)$. At a point that is not a breakpoint, $g_-(u) = g_+(u)$, and we denote their common value by $g(u)$.

Lemma 6 (Monotonicity of the One-Sided Derivatives)**.** *For $c > 0$ and $\lambda \geq 0$, $g_-(u)$ and $g_+(u)$ are nondecreasing on $\mathbb{R}$, and $g(u)$ is strictly increasing between consecutive breakpoints when $\lambda > 0$.*

Proof of Lemma 6. Between two consecutive breakpoints, the cumulative weights are constant, and

$$g'(u) = \frac{\lambda c}{(u^2 + c)^{3/2}} \geq 0, \tag{80}$$

where the inequality is strict when $\lambda > 0$. At a breakpoint r, let $W_=(r) = \sum_{r_j = r} w_j$. As u crosses r, the cumulative weight increases by $W_=(r)$, so the one-sided derivatives have a nonnegative jump of $2W_=(r)$. Therefore, $g_-(u)$ and $g_+(u)$ are nondecreasing both within and across the intervals between breakpoints. □

Bisection procedure. Lemma 6 allows the search interval to be reduced based on the signs of the one-sided derivatives, without evaluating the objective at every breakpoint. For $q \geq 1$, we initialize the bracket as

$$L_0 = \min\{0, r_j : j \in \mathcal{S}_{nz}\}, \qquad R_0 = \max\{0, r_j : j \in \mathcal{S}_{nz}\}. \tag{81}$$

Including zero is necessary, since the regularizer may move the minimizer outside the range of the ratios. For $u < L_0$, all breakpoints lie to the right of u and $u < 0$, so $g(u) = -W + \lambda u/\sqrt{u^2 + c} < 0$; for $u > R_0$, symmetrically, $g(u) > 0$. Hence, a minimizer lies in $[L_0, R_0]$. If $L_0 = R_0$, this point is returned directly; otherwise, the optimality condition is first checked at both endpoints before bisection starts.

At the b-th step, let $u_b = (L_b + R_b)/2$. If $g_-(u_b) \leq 0 \leq g_+(u_b)$, u_b is returned; otherwise, the bracket is updated as

$$(L_{b+1}, R_{b+1}) = \begin{cases} (L_b, u_b), & \text{if } g_-(u_b) > 0, \\ (u_b, R_b), & \text{if } g_+(u_b) < 0. \end{cases} \tag{82}$$

After B steps, the midpoint of the retained bracket is returned.

Theorem 6 (Correctness of the Bisection Solver)**.** *Suppose $q \geq 1$, $c > 0$, and $\lambda > 0$. Then Eq. 77 has a unique global minimizer u^*, which lies in the bracket defined by Eq. 81, and every point satisfying $g_-(u) \leq 0 \leq g_+(u)$ equals u^*. Moreover, if no such point is returned during bisection, the returned midpoint $\widehat{u}$ after B steps satisfies*

$$|\widehat{u} - u^*| \leq \frac{R_0 - L_0}{2^{B+1}}. \tag{83}$$

Proof of Theorem 6. The reconstruction term is convex, and since

$$\frac{\mathrm{d}^2}{\mathrm{d}u^2}\sqrt{u^2 + c} = \frac{c}{(u^2 + c)^{3/2}} > 0, \tag{84}$$

the regularizer is strictly convex. Hence, $\mathcal{L}_{\mathrm{D}_{21}\mathrm{RPCA}}$ is strictly convex for $\lambda > 0$. It is also coercive, and thus attains a unique global minimizer u^*. The derivative signs outside $[L_0, R_0]$ established above guarantee $u^* \in [L_0, R_0]$, and the subgradient optimality condition in Eq. 79 together with uniqueness implies that any point satisfying $g_-(u) \leq 0 \leq g_+(u)$ equals u^*. During bisection, if $g_-(u_b) > 0$, Lemma 6 implies that u^* lies to the left of u_b; if $g_+(u_b) < 0$, it lies to the right. Thus, each update preserves a bracket containing u^* and halves its width. After B steps, the bracket width is $(R_0 - L_0)/2^B$, and the distance from its midpoint to u^* is at most half of this width, which proves Eq. 83. □

These guarantees hold in exact arithmetic. In practice, finite-precision evaluation introduces additional rounding error. With sorted breakpoints and precomputed prefix sums, each evaluation of $g_\pm(u)$ takes $O(\log q)$ time, so the overall cost is $O(q \log q + B \log q)$, consistent with Section 4.4. In all experiments, we set $B = 35$, which reduces the bracket width to $(R_0 - L_0)2^{-35}$ if no early return occurs. Since B is fixed, this yields an absolute error bound that scales with $R_0 - L_0$, rather than an exact solution in all cases.

Algorithm 5 Accelerated ℓ_1-Based Exact Solver (Weighted Median)

Require: $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$, and hyperparameter λ
Ensure: Minimizer u^* of $\mathcal{L}_{\mathrm{D_1RPCA}}(u)$
$\mathcal{S}_{nz} \leftarrow \{j : v_j \neq 0\}$, $q \leftarrow |\mathcal{S}_{nz}|$
if $q = 0$ **then**
  return 0
end if
Compute $r_j = \widetilde{x}_j / v_j$ and $w_j = |v_j|$ for $j \in \mathcal{S}_{nz}$
Append the pair $(0, \lambda)$
Sort the pairs in ascending order of breakpoints and merge duplicates to obtain $\{(\widehat{r}_{(i)}, \widehat{w}_{(i)})\}_{i=0}^{m}$
$\widehat{W} \leftarrow \sum_{i=0}^{m} \widehat{w}_{(i)}$, $S \leftarrow 0$
for $i = 0$ **to** m **do**
  $S \leftarrow S + \widehat{w}_{(i)}$
  if $S \geq \widehat{W}/2$ **then**
    return $\widehat{r}_{(i)}$
  end if
end for

E.4 Boundary and Degenerate Cases

The following cases do not alter the underlying optimization principle, but must be handled explicitly to avoid undefined ratios, ambiguous minimizers, and numerical instability.

Zero Coefficients. If $v_j = 0$, then $|\widetilde{x}_j - u\, v_j| = |\widetilde{x}_j|$ is independent of u. Such indices are excluded from $\mathcal{S}_{nz}$ before the ratios are constructed, which avoids division by zero without changing the minimizer of the element-wise subproblem.

Empty Index Set. If $q = 0$, no reconstruction term depends on u. When $\lambda > 0$, both regularizers $\lambda|u|$ and $\lambda\sqrt{u^2 + c}$ are minimized at $u^* = 0$. When $\lambda = 0$, every u is optimal, and $u^* = 0$ is returned as a deterministic choice.

Repeated Breakpoints. Breakpoints with identical values are merged by summing their weights. This leaves both subproblem objectives unchanged and ensures that the prefix sums and one-sided derivatives are evaluated unambiguously.

Nonunique ℓ_1 Solutions. If the cumulative weight equals exactly $\widehat{W}/2$ at $\widehat{r}_{(\ell^*)}$, every point in the interval $[\widehat{r}_{(\ell^*)}, \widehat{r}_{(\ell^*+1)}]$ is optimal. In this case, the weighted-median rule in Lemma 5 returns the left endpoint $\widehat{r}_{(\ell^*)}$, which is a valid and deterministic minimizer.

Vanishing Column Constant. If $c = 0$, then $\sqrt{u^2 + c} = |u|$, and the ℓ_{21} subproblem reduces to the weighted-median problem in Eq. 72. The ℓ_1 solver is therefore applied directly, avoiding the evaluation of the one-sided derivatives in Eq. 78, whose regularizer term $u/\sqrt{u^2 + c}$ is undefined at $u = 0$.

Zero Regularization. If $\lambda = 0$, both subproblems reduce to minimizing $\sum_{j \in \mathcal{S}_{nz}} w_j |u - r_j|$, and a weighted median of the ratios $\{r_j\}_{j \in \mathcal{S}_{nz}}$ is returned. This case is handled separately because Theorem 6 assumes $\lambda > 0$.

E.5 Complete Algorithms

Algorithm 5 gives one accelerated coordinate update for $\mathrm{D_1}$RPCA. Algorithm 6 presents the corresponding update for $\mathrm{D_{21}}$RPCA, including one-sided derivative tests and bisection on a valid bracket. Both solvers are repeatedly invoked inside Algorithm 7, which summarizes the complete ADMM–ES optimization schedule. All entries of $\boldsymbol{U}$ are refined, whereas only the high-residual rows of $\boldsymbol{V}$ selected by $\mathcal{I}_h(\gamma)$ are updated.

Accelerated ℓ_1-Based Exact Solver. Algorithm 5 first removes constant residual terms, transforms the remaining terms into breakpoint–weight pairs, and incorporates the regularizer as the pair $(0, \lambda)$.

Algorithm 6 Accelerated ℓ_{21}-Based Exact Solver (Bisection)

Require: $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$, hyperparameter $\lambda \geq 0$, constant $c \geq 0$, and number of bisection steps B
Ensure: Minimizer u^* of $\mathcal{L}_{\mathrm{D}_{21}\mathrm{RPCA}}(u)$

$\mathcal{S}_{nz} \leftarrow \{j : v_j \neq 0\}$, $q \leftarrow |\mathcal{S}_{nz}|$
if $q = 0$ **then**
  return 0
end if
if $c = 0$ **or** $\lambda = 0$ **then**
  return the output of Algorithm 5 with $\widetilde{\boldsymbol{x}}$, $\boldsymbol{v}$, and λ
end if
Compute $r_j = \widetilde{x}_j / v_j$ and $w_j = |v_j|$ for $j \in \mathcal{S}_{nz}$
Sort the pairs in ascending order of breakpoints, merge duplicates, and compute W and the prefix sums of the weights
$L_0 \leftarrow \min\{0, r_j : j \in \mathcal{S}_{nz}\}$, $R_0 \leftarrow \max\{0, r_j : j \in \mathcal{S}_{nz}\}$
if $L_0 = R_0$ **then**
  return L_0
end if
for each $z \in \{L_0, R_0\}$ **do**
  Evaluate $g_-(z)$ and $g_+(z)$ via Eq. 78
  if $g_-(z) \leq 0 \leq g_+(z)$ **then**
    return z
  end if
end for
for $b = 0$ **to** $B - 1$ **do**
  $u_b \leftarrow (L_b + R_b)/2$
  Evaluate $g_-(u_b)$ and $g_+(u_b)$ via binary search and prefix sums
  if $g_-(u_b) \leq 0 \leq g_+(u_b)$ **then**
    return u_b
  else if $g_-(u_b) > 0$ **then**
    $(L_{b+1}, R_{b+1}) \leftarrow (L_b, u_b)$
  else
    $(L_{b+1}, R_{b+1}) \leftarrow (u_b, R_b)$
  end if
end for
return $(L_B + R_B)/2$

The first sorted breakpoint at which the cumulative weight reaches half of the total weight is the weighted median and hence the exact coordinate minimizer.

Accelerated ℓ_{21}-Based Exact Solver. Algorithm 6 checks the bracket breakpoints and then reduces an interval containing the minimizer using one-sided derivatives. It handles both smooth stationary points and nonsmooth breakpoint optima without assuming that two original ratios bracket the solution. Prefix sums avoid repeatedly summing all residual terms.

Overall Accelerated Hybrid Alternating Optimization. Starting from random or truncated-SVD initialization, we perform at most T_{A} ADMM iterations, followed by at most T_{E} alternating ES iterations. The ES stage terminates when $|\mathcal{L}^{(t+1)} - \mathcal{L}^{(t)}|/(|\mathcal{L}^{(t)}| + \epsilon) \leq \tau$, where $\mathcal{L}^{(t)} = \mathcal{F}_p(\boldsymbol{U}^{(t)}, \boldsymbol{V}^{(t)})$ denotes the objective at the t-th iteration , τ is a prescribed tolerance, and $\epsilon > 0$. The same hyperparameter λ is used for both $\boldsymbol{U}$ and $\boldsymbol{V}$, and for D_{21}RPCA, the constant c is recomputed from the current values of the other entries in the same column before each element-wise update. In all experiments, we set $\gamma = 0.1$ and $B = 35$. Algorithm 7 uses the ADMM solution as the starting point and then refines the factor matrices via element-wise updates. To avoid recomputing the full product $\boldsymbol{U}\boldsymbol{V}^T$ after each update, we maintain the residual matrix $\boldsymbol{R} = \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T$ and update it incrementally: after a change $\Delta U_{i,h}$ in $U_{i,h}$, $\boldsymbol{R}(i,:) \leftarrow \boldsymbol{R}(i,:) - \Delta U_{i,h}\, \boldsymbol{v}_h^T$; after a change $\Delta V_{j,h}$ in $V_{j,h}$, $\boldsymbol{R}(:,j) \leftarrow \boldsymbol{R}(:,j) - \Delta V_{j,h}\, \boldsymbol{u}_h$. The residual scores for prescreening are computed from $\widetilde{\boldsymbol{R}}_h = \boldsymbol{R}^T + \boldsymbol{v}_h \boldsymbol{u}_h^T$ at the beginning of the update of $\boldsymbol{v}_h$, which coincides with Eq. 12. Since this score excludes the h-th component, it measures the residual before that component is re-

Algorithm 7 Accelerated Hybrid ADMM-ES Alternating Optimization

Require: Data $\boldsymbol{X}$, rank k, parameters λ, ADMM iterations T_{A}, ES iterations T_{E}, screening ratio γ, tolerance τ, denominator offset $\epsilon > 0$, bisection steps B, and solver type $\nu \in \{1, 21\}$
Ensure: Factors $\boldsymbol{U}, \boldsymbol{V}$
Initialize $\boldsymbol{U}$ and $\boldsymbol{V}$ randomly or by truncated SVD
// Matrix-Wise Update via ADMM
Run T_{A} matrix-level ADMM iterations to obtain a warm start
Cache $\boldsymbol{R} \leftarrow \boldsymbol{X} - \boldsymbol{U}\boldsymbol{V}^T$
// Element-Wise Update via ES
for $t = 0, \ldots, T_{\mathrm{E}} - 1$ **do**
  Store the previous objective value $\mathcal{L}^{(t)}$
  // Element-Wise Updates of $\boldsymbol{U}$ with $\boldsymbol{V}$ Fixed via Exact Solvers
  for $h = 1, \ldots, k$ **do**
    Form residuals from $\boldsymbol{R} + \boldsymbol{u}_h \boldsymbol{v}_h^T$ as needed
    for $i = 1, \ldots, p$ **do**
      Update u_{ih} using Algorithm 5 or Algorithm 6, according to ν
      Update the cached residual incrementally
    end for
  end for
  // Element-Wise Updates of $\boldsymbol{V}$ with $\boldsymbol{U}$ Fixed via Exact Solvers
  for $h = 1, \ldots, k$ **do**
    Compute s_{jh} from Eq. 12
    Select $\mathcal{I}_h(\gamma)$ using Eq. 13
    for each $j \in \mathcal{I}_h(\gamma)$ **do**
      Update v_{jh} using Algorithm 5 or Algorithm 6, according to ν
      Update the cached residual incrementally
    end for
  end for
  Evaluate $\mathcal{L}^{(t+1)}$
  if $|\mathcal{L}^{(t+1)} - \mathcal{L}^{(t)}|/(|\mathcal{L}^{(t)}| + \epsilon) \leq \tau$ **then**
    break
  end if
end for
return $\boldsymbol{U}, \boldsymbol{V}$

fined, and thus serves as a heuristic for prioritization rather than a guarantee of the largest objective decrease. Only the selected entries of $\boldsymbol{v}_h$ are updated, while all others retain their current values.

We note two properties of the overall accelerated procedure. First, since unselected entries remain unchanged, prescreening preserves the monotonic decrease of J_{aux} established in Theorem 5; however, it does not guarantee convergence to the same point as the full ES iteration. Second, since finite-step bisection may return an approximate solution for D_{21}RPCA, an update is accepted only if it does not increase the objective, so that the monotonicity guarantee continues to hold.

E.6 Complexity Analysis

Let q_h^V and q_h^U bound the numbers of nonzero coefficients in $\boldsymbol{v}_h$ and $\boldsymbol{u}_h$, respectively, over the relevant refinement sweep. For fixed supports, these equal $\mathrm{nnz}(\boldsymbol{v}_h)$ and $\mathrm{nnz}(\boldsymbol{u}_h)$. Here, q_h^V bounds the number of active residual terms involved when an entry of $\boldsymbol{U}$ is updated, and q_h^U plays the analogous role for an entry of $\boldsymbol{V}$. These quantities make the following expressions applicable to both dense and sparse factors. A full update of $\boldsymbol{U}$ solves p coordinate subproblems for every latent component, whereas residual screening updates only $N_\gamma = \max\{\lfloor \gamma n \rfloor, 1\}$ rows of $\boldsymbol{V}$ per component.

For the original exhaustive solver, each coordinate evaluates $O(q)$ breakpoint candidates, with $O(q)$ residual operations per candidate. Therefore, the breakpoint evaluation requires

$$O\left(p \sum_{h=1}^{k} (q_h^V)^2 + n \sum_{h=1}^{k} (q_h^U)^2\right) \tag{85}$$

time per refinement iteration for nonempty coordinate subproblems. The first term corresponds to the complete update of $\boldsymbol{U}$, and the second term corresponds to the complete update of $\boldsymbol{V}$.

To include empty and singleton retained sets, define $A(q) = 1 + q\log(1+q)$ and $H(q) = A(q) + B\log(1+q)$. The accelerated ℓ_1 exact solves require

$$O\Big(p\sum_{h=1}^{k} A(q_h^V) + N_\gamma \sum_{h=1}^{k} A(q_h^U)\Big), \tag{86}$$

where the logarithmic factors arise from sorting the breakpoint-weight pairs. The accelerated $\ell_{2,1}$ exact solver additionally performs B bisection steps with binary-search evaluation of the prefix weights, giving

$$O\Big(p\sum_{h=1}^{k} H(q_h^V) + N_\gamma \sum_{h=1}^{k} H(q_h^U)\Big). \tag{87}$$

Computing the residual scores, maintaining the cached residual, and evaluating the objective add $O(pnk)$ arithmetic operations per sweep. Group squared norms can also be maintained incrementally. Selecting the largest N_γ scores costs $O(n)$ per component with a linear-time partial-selection routine and is covered by this bound; fully sorting the scores instead adds $O(kn\log n)$ per sweep. Because B is fixed to 35 and $N_\gamma \simeq \gamma n$, the improvements are the replacement of quadratic breakpoint evaluation by sorting and the reduction in the number of $\boldsymbol{V}$ coordinate solves by approximately the factor γ. Residual scoring still processes all samples, so this factor does not describe the full iteration cost.

In the dense case, $q_h^V = n$ and $q_h^U = p$. The following simplified expressions assume $p, n \geq 2$ and $N_\gamma \simeq \gamma n$; the preceding formulas retain the exact N_γ and cover smaller dimensions. The unaccelerated breakpoint evaluation in Eq. 85 reduces to

$$O(pkn^2 + nkp^2), \tag{88}$$

whereas the accelerated ℓ_1 refinement requires

$$O(pkn\log n + \gamma nkp\log p + pnk). \tag{89}$$

The corresponding $\ell_{2,1}$ complexity contains the additional terms $O(pkB\log n + \gamma nkB\log p)$. Thus, sorting replaces the quadratic dependence on the active vector length, while screening reduces the number of $\boldsymbol{V}$ coordinate subproblems from nk to approximately γnk.

The original vectorized breakpoint evaluation uses $O(q^2)$ temporary memory per coordinate because it materializes the residuals of all candidates. For $q \geq 1$, both accelerated solvers require only $O(q)$ temporary memory for the ratios, weights, sorted indices, and prefix sums. Including the data, factors, and cached residual, the total working memory is

$$O\big(pn + pk + nk + \max_h\{q_h^U, q_h^V\}\big). \tag{90}$$

Finally, let $C_{\mathrm{A}}(p,n,k)$ denote the cost of one matrix-level ADMM iteration and let $C_{\mathrm{E}}(p,n,k,\gamma)$ denote the appropriate refinement cost in Eq. 86 or Eq. 87, including residual scoring and incremental updates. The complete hybrid procedure has total time complexity

$$O(T_{\mathrm{A}}\cdot C_{\mathrm{A}}(p,n,k) + T_{\mathrm{E}}\cdot C_{\mathrm{E}}(p,n,k,\gamma))\,. \tag{91}$$

The first term corresponds to the matrix-level ADMM stage, while the second term corresponds to element-level refinement. Here T_{E} is an upper bound, and the stopping criterion may terminate refinement earlier. These expressions quantify arithmetic costs, and actual runtime and the effect of screening on solution quality require empirical evaluation.

F Additional Experiments to Validate the Effectiveness of DRPCA

F.1 Dataset Description

Table 3 summarizes eight facial image datasets and two natural image datasets. Each dataset consists of one or more MAT subfiles, with every subfile represented by a data matrix in $\mathbb{R}^{d\times n}$, where d is

Table 3: Statistics of the benchmark datasets. Each MAT subfile contains a data matrix of size $d \times n$.

Dataset	**Dimension** d	**Samples** n	**#MAT Files**
Facial image datasets			
AT&T	2,576	100	4
AR	2,200	140	4
FaceGT	2,400	100	1
YaleB	4,096	165	1
FERET	6,400	70	1
CMU-PIE	4,096	100	1
UMIST	10,304	100	1
JAFFE	4,096	140	1
Natural image datasets			
CIFAR-10	1,024	1,000	4
ImageNet	1,024	5,000 / 10,000	4

the feature dimension and n is the number of samples. For example, AT&T consists of four subfiles, each containing a $2{,}576 \times 400$ matrix. Similarly, CIFAR-10 contains four $1{,}024 \times 1{,}000$ subfiles. For ImageNet, two data scales are considered, with each setting comprising four subfiles of size $1{,}024 \times 5{,}000$ or $1{,}024 \times 10{,}000$.

F.2 Details of Corrupted Data Generation

As described in Section 5.1, two corruption settings are considered: single-type noise and mixed-type noise. Different corruption strengths are used for facial and natural images because these datasets exhibit substantially different visual characteristics.

Corruption of Facial Images. Under the single-type setting, every facial image is corrupted by a rectangular occlusion whose height and width are both one-fifth of the image dimensions. The resulting occlusion covers approximately 4% of the image. Its position is sampled independently for each image to avoid identical corruption patterns across samples.

Under the mixed-type setting, five images are randomly selected from each class. Each selected image is corrupted using one cross-shaped, two rectangular, and two circular occlusions, with different regions placed without overlap. The class-level corruption ratio is defined as

$$r_{\text{noise}} = \frac{\text{total number of corrupted pixels in a class}}{\text{total number of pixels in the class}}, \tag{92}$$

and is controlled between 3% and 4%. The dimensions of the corruption regions are adjusted according to the image resolution and class size. For example, the cross-shaped occlusion is 25×7 pixels for AT&T and 33×7 pixels for AR, while the rectangular and circular regions use approximately 25% and 22% of the corresponding image dimensions, respectively.

Corruption of Natural Images. CIFAR-10 and ImageNet require a different setting because their foreground objects and backgrounds are substantially more diverse than the highly aligned and structurally similar facial images. A small occlusion ratio can be difficult to distinguish from their intrinsic background variation. Therefore, the combined corruption mask for each corrupted natural image is enlarged to cover approximately 30% of its pixels. For mixed-type corruption, the cross-shaped, rectangular, and circular regions are scaled jointly and placed without overlap so that their total area remains close to $0.3HW$, where H and W denote the image height and width.

Construction of the CIFAR-10 Subsets. The CIFAR-10 subsets are constructed from the test split. A pretrained ResNet-18 without its classification head is used to extract ℓ_2-normalized features, and the images closest to each class centroid are selected as representative prototypes. For each class, 40 prototypes are selected, randomly shuffled with seed 42, and divided into four disjoint groups of 10 prototypes. Each prototype is retained and augmented nine times using random horizontal flipping with probability 0.5 and color jitter with brightness, contrast, and saturation factors of 0.2 and a hue factor of 0.05. The resulting images are converted to grayscale and flattened into $1{,}024$-dimensional

vectors. Consequently, each MAT subfile contains

$$10 \text{ classes} \times 10 \text{ prototypes} \times (1+9) \text{ instances} = 1{,}000 \tag{93}$$

samples, producing a data matrix of size $1{,}024 \times 1{,}000$.

Construction of the ImageNet Subsets. For ImageNet, the validation classes are randomly shuffled with seed 42 and partitioned into four disjoint groups of 50 classes. Within each class, one or two images closest to the class centroid in the normalized ResNet-18 feature space are selected as prototypes. Each prototype is retained and augmented 99 times using the same horizontal-flipping and color-jitter operations. All images are converted to grayscale, resized to 32×32, and flattened into 1,024-dimensional vectors. Selecting one prototype per class yields

$$50 \text{ classes} \times 1 \text{ prototype} \times (1+99) \text{ instances} = 5{,}000 \tag{94}$$

samples per MAT subfile, whereas selecting two prototypes yields 10,000 samples. The corresponding data matrices therefore have sizes $1{,}024 \times 5{,}000$ and $1{,}024 \times 10{,}000$, respectively.

F.3 Evaluation on Additional Benchmark Datasets Against State-of-the-Art Methods

To further assess the generalization and robustness of the proposed DRPCA approach, we conduct complementary evaluations against state-of-the-art RPCA methods. We first examine reconstruction errors on six additional face datasets under single-type and mixed-type noise, and then evaluate reconstruction quality on AT&T, CMU, CIFAR-10, and ImageNet-5000 using ℓ_2, SSIM, PSNR, and LPIPS. All results are averaged over four settings of the subspace rank to reduce sensitivity to the choice of rank.

Table 4 reports the average ℓ_1-based test reconstruction errors on the six additional face datasets. The two proposed DRPCA variants (D_1RPCA and D_{21}RPCA) consistently achieve the best or second-best performance across all 12 dataset-noise settings, with D_{21}RPCA obtaining the lowest error in 11 settings and D_1RPCA performing best on UMIST under single-type noise. These results support the benefit of latent-space regularization for robust PCA.

Table 5 presents results on AT&T, CMU, CIFAR-10, and ImageNet-5000. Across all eight dataset-noise settings, D_1RPCA and D_{21}RPCA achieve the best and second-best ℓ_2 errors, respectively. They also rank among the top two methods across all four metrics on AT&T and CMU. On CIFAR-10 and ImageNet-5000, metric-wise rankings are more heterogeneous, while DRPCA remains strongest in ℓ_2 reconstruction and competitive on structural and perceptual metrics. Overall, D_{21}RPCA generally yields lower reconstruction errors than D_1RPCA, whereas their relative performance on structural and perceptual metrics varies across datasets and noise types.

Table 4: Comparison of the average ℓ_1-based test reconstruction error (ERR_{test}) on additional datasets across four rank settings under single-type (S) and mixed-type (M) noise. Lower values are better. The best and second-best results are highlighted in red and blue, respectively.

Method	AR ($n=140$)		FaceGT ($n=100$)		FERET ($n=70$)		JAFFE ($n=140$)		UMIST ($n=100$)		YaleB ($n=165$)	
	S	M	S	M	S	M	S	M	S	M	S	M
L_2-PCA	24184	23000	21589	22713	27369	26764	39438	33786	73736	73938	55231	51749
R_1-PCA (Huber)	23912	22233	21509	22500	27138	26487	38980	32840	73359	72734	55055	51407
R_1-PCA (Cauchy)	23660	21820	21442	22381	26887	26054	38514	32212	72933	71738	54918	51180
L_1-PCA (ALM)	17421	17748	18613	19531	21889	19582	30677	25487	57166	56199	47151	44358
RPCA	20241	20287	20200	21177	23877	23470	34803	29556	68112	68695	51640	48472
CRPCA	23690	21910	21466	22432	26934	25895	38560	32212	72974	71718	54956	51206
RPCA-OM	23243	21590	21312	22177	26473	25232	37860	31509	71033	69539	54630	50824
CRPCA-OM	23280	21542	18575	19445	26677	25462	37931	31526	71392	69563	54682	50878
L_{21}-PCA (ALM)	23623	21934	21430	22339	26860	25690	38479	32252	72815	71602	54768	51187
L_{21}-PCA (NG)	24127	23152	21612	22763	27341	26807	39283	33886	73596	74017	55159	51827
eRPCA	22179	22485	21992	22931	26005	25995	37243	33068	71063	71747	53768	51201
DW-RPCA	23292	20560	21600	22028	26664	25088	37965	31095	75393	70670	56235	51490
DCA	23639	21969	21418	22369	26803	25625	38483	32223	72869	71672	54912	51121
D_1RPCA(Ours)	**16355**	**17101**	**18562**	**18822**	**20052**	**18326**	**28297**	**25249**	**55224**	**55745**	**46137**	**44299**
D_{21}RPCA(Ours)	**16107**	**16845**	**18181**	**18460**	**19238**	**17838**	**28055**	**25201**	**55256**	**55708**	**45829**	**44259**

Table 5: Comparison of reconstruction results averaged over all available rank settings on AT&T, CMU, CIFAR-10, and ImageNet (n=5000) under single-type (S) and mixed-type (M) noise. The evaluated metrics are ℓ_2, SSIM, PSNR, and LPIPS. Lower values are better for ℓ_2 and LPIPS, whereas higher values are better for SSIM and PSNR. Red and blue indicate the best and second-best results, respectively.

Dataset	Method	Single-type noise				Mixed-type noise			
		$\ell_2 \downarrow$	SSIM↑	PSNR↑	LPIPS↓	$\ell_2 \downarrow$	SSIM↑	PSNR↑	LPIPS↓
AT&T	L_2-PCA	52.7628	0.5608	16.6550	0.4323	57.8120	0.5387	16.5625	0.4231
	R_1-PCA (Huber)	52.8291	0.5618	16.5775	0.4293	56.9096	0.5493	16.7050	0.4140
	R_1-PCA (Cauchy)	52.8643	0.5625	16.5075	0.4254	56.1608	0.5579	16.8600	0.4036
	L_1-PCA (ALM)	47.9206	0.6131	17.5450	0.3745	51.2116	0.6015	17.5675	0.3758
	RPCA	47.9115	0.5861	17.2075	0.4029	52.8863	0.5617	16.6925	0.4069
	CRPCA	52.9149	0.5623	16.5275	0.4250	56.5372	0.5547	16.8200	0.4056
	RPCA-OM	52.3582	0.5748	16.6825	0.4171	55.9339	0.5671	16.9825	0.3970
	CRPCA-OM	52.4461	0.5741	16.6975	0.4178	56.2154	0.5650	16.9575	0.4003
	L_{21}-PCA (ALM)	52.8860	0.5624	16.5125	0.4250	56.3547	0.5568	16.8375	0.4042
	L_{21}-PCA (NG)	52.7170	0.5620	16.6650	0.4307	58.0909	0.5361	16.5075	0.4242
	eRPCA	49.9618	0.5392	15.4175	0.4294	54.0710	0.5181	15.0000	0.4308
	DW-RPCA	52.7216	0.5734	16.6200	0.4091	53.9391	0.5760	17.0925	0.3827
	DCA	52.8642	0.5624	16.5100	0.4250	56.2832	0.5573	16.8650	0.4039
	D_1RPCA(Ours)	**43.7733**	**0.6341**	**18.3825**	**0.3526**	**46.2392**	**0.6183**	**18.1025**	**0.3607**
	D_{21}RPCA(Ours)	**43.6144**	**0.6391**	**18.5300**	**0.3499**	**44.2146**	**0.6270**	**18.2900**	**0.3572**
CMU	L_2-PCA	46.5639	0.6997	19.3750	0.2741	39.9755	0.7464	21.2500	0.2205
	R_1-PCA (Huber)	46.2379	0.7070	19.3550	0.2648	38.4196	0.7648	21.6400	0.2035
	R_1-PCA (Cauchy)	46.1376	0.7101	19.3675	0.2605	37.0218	0.7776	21.8675	0.1912
	L_1-PCA (ALM)	37.1600	0.7688	20.5575	0.1988	29.6402	0.8161	22.9725	0.1625
	RPCA	39.9880	0.7424	21.2925	0.2328	33.8848	0.7890	22.8700	0.1897
	CRPCA	46.1724	0.7093	19.3675	0.2609	37.1747	0.7770	21.8750	0.1925
	RPCA-OM	45.9718	0.7135	19.4200	0.2585	37.2242	0.7743	21.9000	0.1953
	CRPCA-OM	45.9839	0.7137	19.4175	0.2585	37.0632	0.7763	21.8925	0.1943
	L_{21}-PCA (ALM)	46.0169	0.7095	19.3550	0.2604	37.4128	0.7745	21.8650	0.1941
	L_{21}-PCA (NG)	46.2292	0.7077	19.5075	0.2686	39.9501	0.7498	21.2900	0.2194
	eRPCA	41.9049	0.7144	20.2025	0.2445	38.9890	0.7425	20.7125	0.2179
	DW-RPCA	46.3704	0.7161	19.3775	0.2552	36.6071	0.7786	22.0050	0.1910
	DCA	46.1632	0.7092	19.3650	0.2611	37.2156	0.7758	21.8700	0.1929
	D_1RPCA(Ours)	**33.7073**	**0.7933**	**21.7400**	**0.1803**	**26.9205**	**0.8294**	**23.8975**	**0.1542**
	D_{21}RPCA(Ours)	**28.0633**	**0.8212**	**23.6875**	**0.1627**	**26.8283**	**0.8297**	**24.0375**	**0.1537**
CIFAR-10	L_2-PCA	72.0721	0.7530	19.8075	0.3384	84.4165	0.7295	19.1425	0.3533
	R_1-PCA (Huber)	71.9992	0.7552	20.0950	0.3314	84.0092	0.7327	19.4100	0.3456
	R_1-PCA (Cauchy)	72.6748	0.7532	20.7750	0.3221	84.2428	0.7318	19.9950	0.3359
	L_1-PCA (ALM)	70.6176	0.7544	20.7725	0.3173	83.6527	0.7289	19.5175	0.3331
	RPCA	71.7221	0.7543	20.3575	0.3264	82.5744	**0.7336**	19.6150	0.3393
	CRPCA	73.0761	0.7462	**21.9000**	0.3203	84.7390	0.7254	**21.0675**	0.3342
	RPCA-OM	73.0898	0.7480	21.8525	0.3206	84.6925	0.7274	20.9950	0.3344
	CRPCA-OM	72.9308	0.7476	21.8475	0.3197	84.5687	0.7262	21.0375	0.3330
	L_{21}-PCA (ALM)	72.9608	0.7482	21.6500	0.3225	84.5593	0.7271	20.8850	0.3356
	L_{21}-PCA (NG)	72.3645	0.7545	19.8625	0.3392	84.5637	0.7305	19.1625	0.3538
	eRPCA	83.7574	0.6596	17.1350	0.3838	91.4454	0.6466	17.0375	0.3936
	DW-RPCA	76.8894	0.7398	**22.5850**	**0.3166**	87.3305	0.7203	**21.6425**	**0.3300**
	DCA	73.2498	0.7469	21.8950	0.3214	84.8240	0.7260	21.0475	0.3350
	D_1RPCA(Ours)	**68.4743**	**0.7580**	21.5925	**0.3133**	**82.3571**	0.7305	19.9675	0.3315
	D_{21}RPCA(Ours)	**68.5112**	**0.7610**	21.7400	0.3196	**81.7535**	**0.7344**	20.2175	**0.3304**
ImageNet (n=5000)	L_2-PCA	183.4768	0.7434	19.7950	0.3301	223.6447	0.7164	18.8425	0.3460
	R_1-PCA (Huber)	182.7319	0.7445	19.9675	0.3268	222.8054	0.7174	18.9800	0.3430
	R_1-PCA (Cauchy)	184.2947	0.7420	20.4175	0.3226	223.4581	0.7156	19.2750	0.3397
	L_1-PCA (ALM)	185.1143	0.7367	19.8700	0.3264	221.7012	0.7126	18.7900	0.3397
	RPCA	185.3589	0.7441	20.0400	0.3255	220.1005	**0.7203**	19.1075	0.3395
	CRPCA	208.9828	0.7412	22.1625	0.3198	262.9221	0.7109	19.8900	0.3417
	RPCA-OM	186.2001	0.7376	21.4500	0.3232	224.4386	0.7125	19.9700	0.3398
	CRPCA-OM	208.2952	0.7428	22.0675	**0.3191**	262.3958	0.7119	19.8550	0.3406
	L_{21}-PCA (ALM)	207.6835	**0.7459**	21.9200	0.3199	262.5472	0.7132	19.7725	0.3417
	L_{21}-PCA (NG)	205.7669	**0.7518**	20.2300	0.3258	262.8374	0.7154	18.8500	0.3470
	eRPCA	212.5870	0.7429	19.9425	0.3264	257.8493	0.7158	18.9525	0.3431
	DW-RPCA	223.1119	0.7314	**22.9500**	**0.3191**	271.0134	0.7035	**20.4650**	0.3402
	DCA	208.9391	0.7440	**22.2300**	0.3200	263.1375	0.7118	**19.9775**	0.3417
	D_1RPCA(Ours)	**153.2497**	0.7374	20.4450	0.3245	**219.9654**	**0.7262**	19.1700	**0.3387**
	D_{21}RPCA(Ours)	**171.1959**	0.7411	20.3425	0.3358	**209.4858**	0.7189	19.3425	**0.3382**

F.4 Qualitative Reconstruction Comparison

Figs 8–9 compare the rank-60 reconstructions of the same occluded images across methods. On both CIFAR-10 and ImageNet-5000, DRPCA generally recovers more recognizable structures than other PCA and RPCA methods. Nevertheless, compared with facial images, natural images contain more complex textures and diverse object structures, making the recovery of fine details more challenging for all methods.

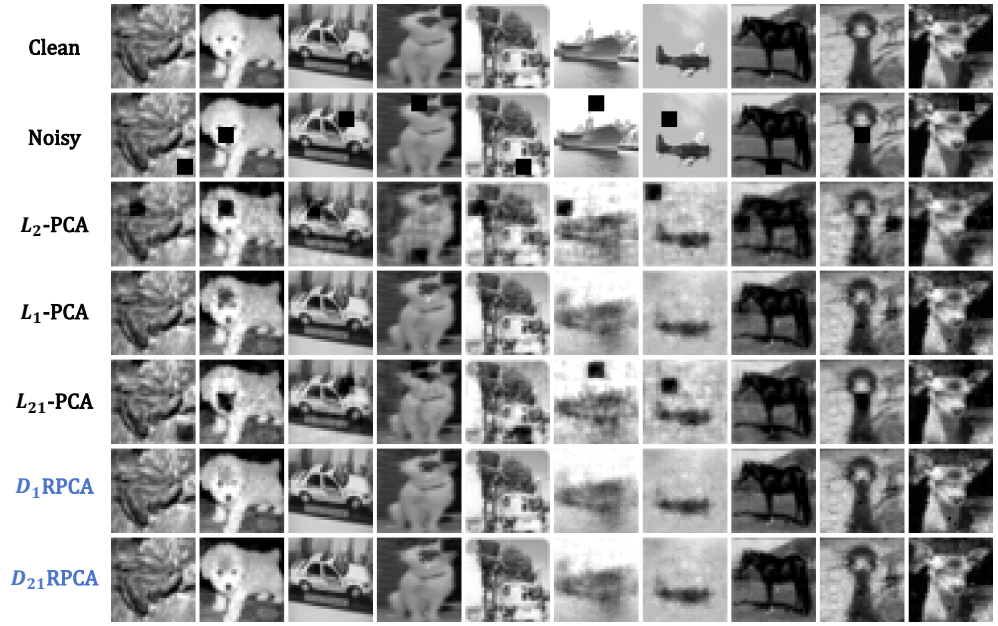


Figure 8: Comparison of CIFAR-10 image reconstructions under occlusion noise. Columns show ten example images; rows show clean images, noisy inputs, and reconstructions obtained by L_2-PCA, L_1-PCA, L_{21}-PCA, D_1RPCA, and D_{21}RPCA (top to bottom).

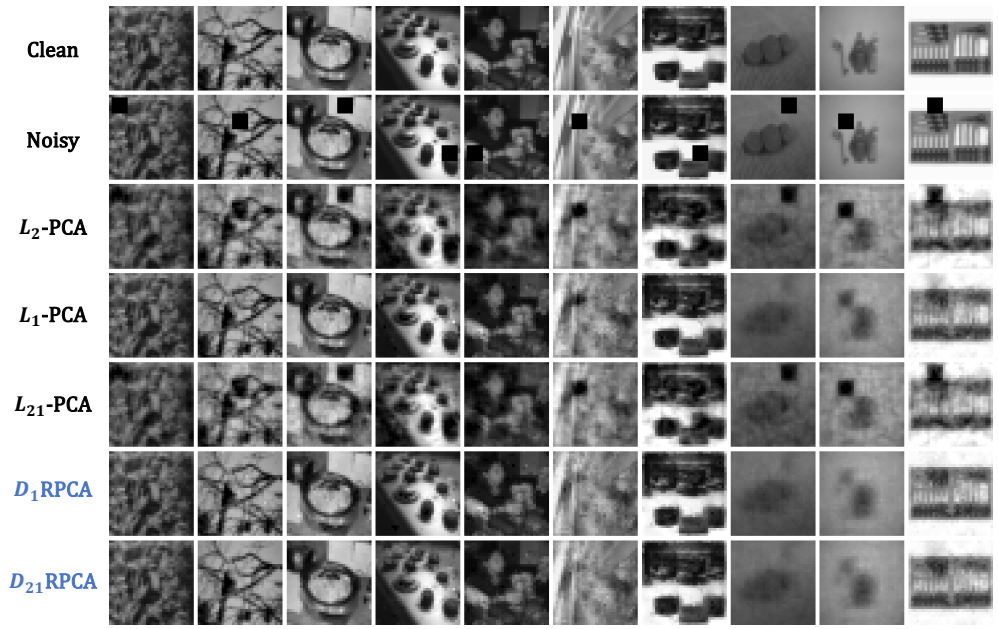
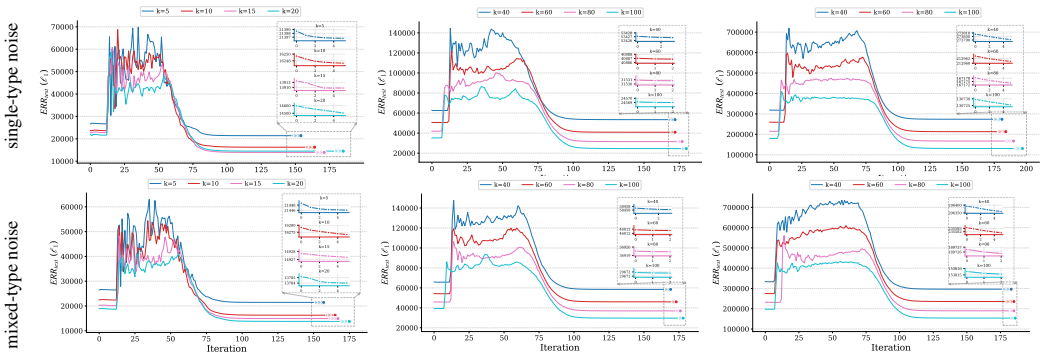


Figure 9: Comparison of ImageNet image reconstructions under occlusion noise. Columns show ten example images; rows show clean images, noisy inputs, and reconstructions obtained by L_2-PCA, L_1-PCA, L_{21}-PCA, D_1RPCA, and D_{21}RPCA (top to bottom).

F.5 Detailed Comparison Between ADMM and ES

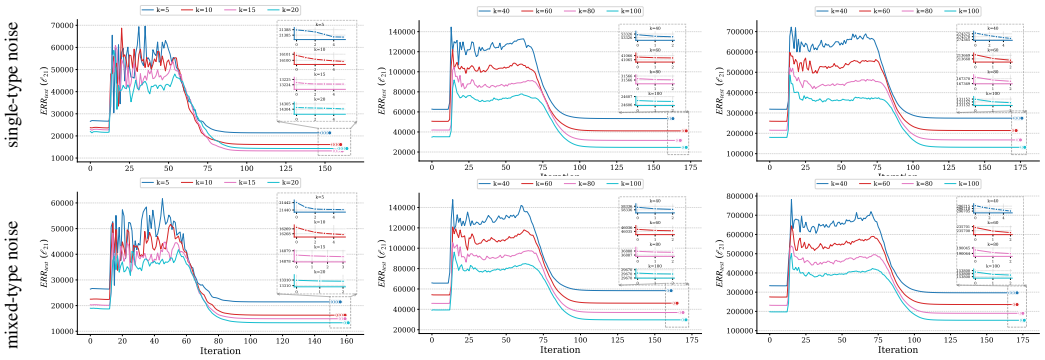
To ensure a fair comparison, we evaluated ADMM and ES under the same low-rank subspace and sparsity constraints, *i.e.*, $rank(\hat{\boldsymbol{X}}_{ADMM}) = rank(\hat{\boldsymbol{X}}_{ES}) = k$, and $\lambda_{ADMM} = \lambda_{ES} = \lambda$. As described in Appendix D.3, we first initialize $\boldsymbol{U}$ and $\boldsymbol{V}$ by solving D_1RPCA and D_{21}RPCA problems using the ADMM algorithm with hyperparameter λ_{init}. We then apply the ES method using $\lambda_{ES} = \lambda$ to obtain $\hat{\boldsymbol{X}}_{ES}$, ensuring that both ES and ADMM solve the same optimization problem under identical sparsity constraints. Due to the different type of sparsity-inducing regularizations, we solve the D_1RPCA problem with $\lambda_{init} = \lambda$, while the D_{21}RPCA problem is solved with $\lambda_{init} \neq \lambda$ to provide a better initialization for the ES algorithm.

To more clearly demonstrate the efficiency and effectiveness of ES compared with ADMM, Figs. 10–11 illustrate the total loss curves for solving the D_1RPCA and D_{21}RPCA problems under two types of noise across three datasets. Specifically, AR ($n = 140$), CIFAR-10 ($n = 1000$), and ImageNet ($n = 5000$) represent small-, medium-, and large-scale datasets, respectively, providing comparisons across different dataset scales. As shown in the figures, the loss curves exhibit noticeable fluctuations during the ADMM stage, while the loss decreases rapidly after switching to ES and gradually converges to stable values. In most cases, the loss converges within five ES iterations, indicating that only a small number of ES iterations are required to further refine the solutions obtained by ADMM.



AR (n=140) CIFAR-10 (n=1000) ImageNet (n=5000)

Figure 10: Loss Curves of ADMM and ES for solving D_1RPCA.



AR (n=140) CIFAR-10 (n=1000) ImageNet (n=5000)

Figure 11: Loss Curves of ADMM and ES for solving D_{21}RPCA.

F.6 Hyperparameter Analysis

To assess sensitivity to the regularization parameter λ, the test reconstruction errors of D_1RPCA and D_{21}RPCA are reported across different values of λ and subspace ranks k. The experiments cover AR, CIFAR-10, and ImageNet under both single-type noise (Fig. 12) and mixed-type noise (Fig. 13).

Across most settings, the reconstruction error remains stable over a broad range of λ, particularly on CIFAR-10 and ImageNet at moderate-to-large ranks. On AR, however, large values of λ increase the error, especially for D_{21}RPCA at smaller ranks, indicating that excessive regularization can impair reconstruction. Increasing k generally reduces the error more substantially than fine-tuning λ. Similar patterns are observed under both noise settings, although the error-minimizing λ varies across datasets and ranks. Overall, both methods are relatively insensitive to moderate variations in λ, while dataset- and rank-specific tuning remains beneficial.

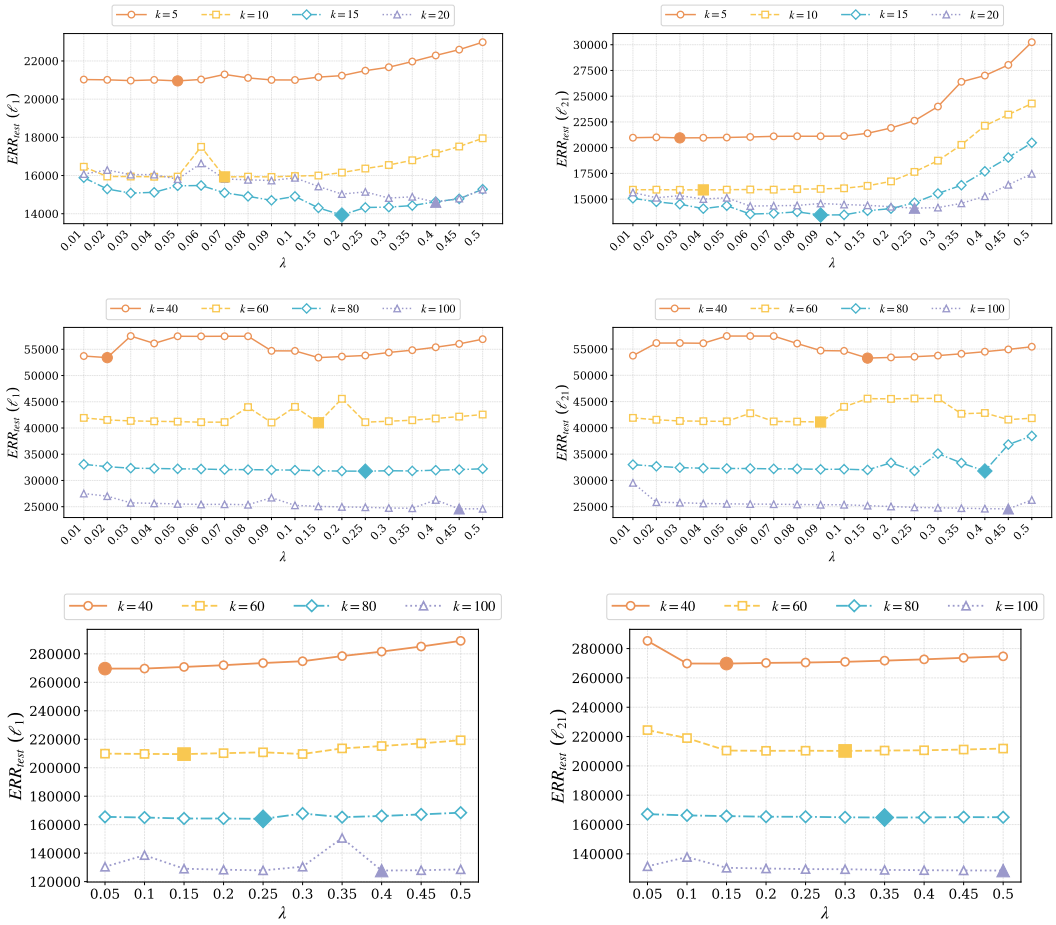


Figure 12: Sensitivity of the test reconstruction error to the regularization parameter λ under "single-noise corruption" for D_1RPCA (left) and D_{21}RPCA (right). From top to bottom, the results are shown for AR, CIFAR-10, and ImageNet ($N = 5000$). Each curve corresponds to a subspace rank k, and the filled markers indicate the minimum error and the corresponding optimal λ for each rank.

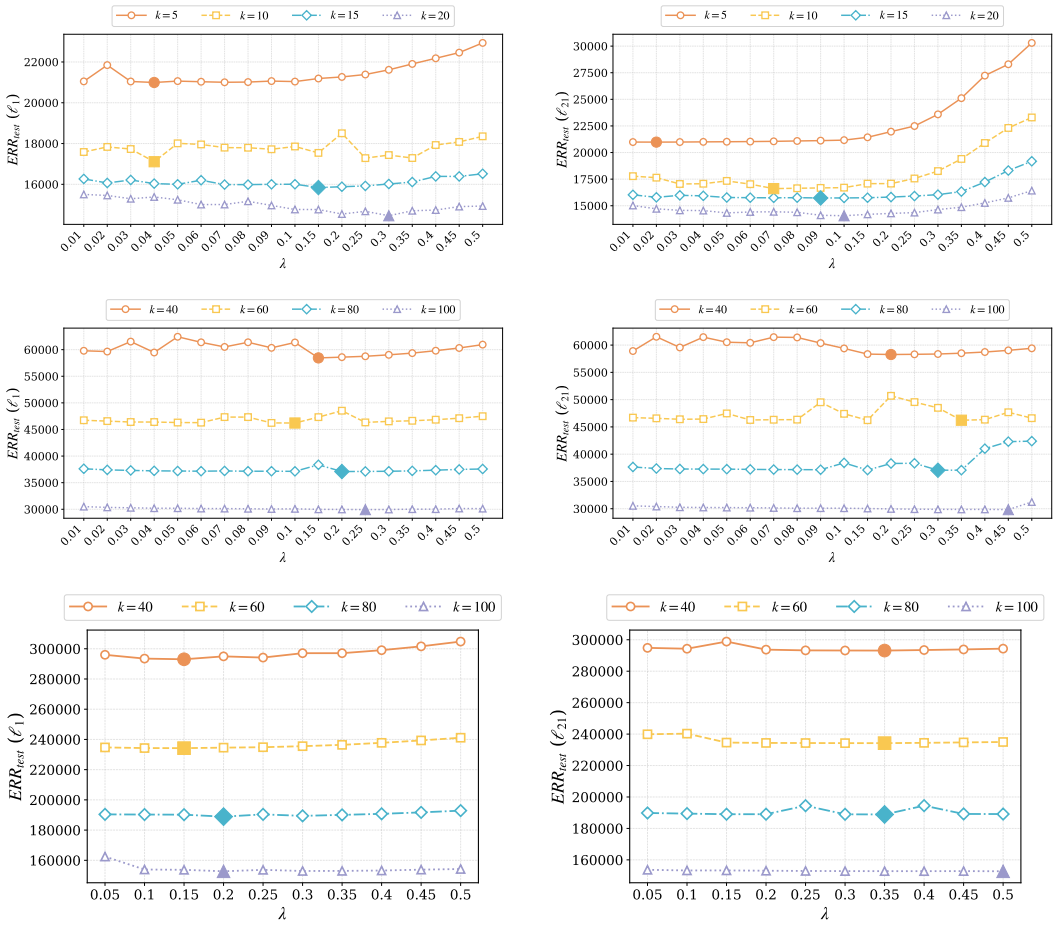


Figure 13: Sensitivity of the test reconstruction error to the regularization parameter λ under "mixed-noise corruption" for D_1RPCA (left) and D_{21}RPCA (right). From top to bottom, the results are shown for AR, CIFAR-10, and ImageNet ($N = 5000$). Each curve corresponds to a subspace rank k, and the filled markers indicate the minimum error and the corresponding optimal λ for each rank.