Learning inexact alternating minimization for convex problems with quadratic coupling

Paul Häusner[1] **Jevgenija Rudzusika**[2] **Jens Sjölund**[1] **Ozan Öktem**[2]

[1]Department of Information Technology, Uppsala University, Sweden
[2]Department of Mathematics, KTH Royal Institute of Technology, Stockholm, Sweden
paul.hausner@it.uu.se, jevaks@kth.se, jens.sjolund@it.uu.se, ozan@kth.se

Abstract

Alternating minimization is a standard tool for optimization problems with two coupled variable blocks. Its cost is dominated by the inner subproblem solvers, which rarely have a closed-form solution. In this paper, we propose to learn an inexact alternating scheme in which neural networks approximate the subproblem solution at each iteration. For convex problems with quadratic couplings between the variable blocks that satisfy a quadratic growth condition, we derive a worst-case bound on the optimality gap that exposes the per-iteration error that each network is trained to minimize in a greedy fashion. Combining the bound with a PAC-Bayes argument, we obtain a fixed-horizon, high-probability bound on the expected optimality gap over the distribution of problem instances. We apply the learned scheme to robust matrix completion and low-dose CT reconstruction with dictionary regularization where we achieve a $5\times$–$13\times$ speedup over the inexact first-order baseline at matched accuracy. A brief end-to-end fine-tuning stage extends this up to $30\times$ and outperforms other learned schemes. The code is available at https://github.com/paulhausner/learned-alternating-minimization.

1 Introduction

We consider large-scale convex optimization problems where one seeks to minimize a convex objective with two variable blocks $(x, z) \mapsto F(x, z)$ of the form

$$F(x,z) = f(x) + \frac{1}{2}\|Ax + Bz\|^2 + g(z) \quad \text{for } (x,z) \in \mathcal{X} \times \mathcal{Z}. \tag{1}$$

Here, $\mathcal{X}$ and $\mathcal{Z}$ are Hilbert spaces. Furthermore, both $f\colon \mathcal{X} \to \mathbb{R}$ and $g\colon \mathcal{Z} \to \mathbb{R}$ are proper, convex, lower semi-continuous functions. The two variable blocks x and z are coupled through a quadratic function that is defined in terms of bounded linear operators A and B that map $\mathcal{X}$ and $\mathcal{Z}$ to a common Hilbert space with inner product $\langle \cdot, \cdot \rangle$ and corresponding induced norm $\|\cdot\|$. Such problems appear in, e.g., inverse problems [Cheng et al., 2020, Chambolle and Pock, 2015], control [Khan et al., 2025], and naturally arise in half-quadratic splitting formulations [Geman and Yang, 1995].

Directly optimizing the objective function Eq. (1) is challenging due to the quadratic coupling between the two variable blocks that prevents efficient joint optimization. The following alternating minimization scheme [Tseng, 2001] instead decouples the problem into two subproblems each involving only a single variable block that can be solved more efficiently using first-order methods:

$$x^{k+1} := \operatorname*{argmin}_{x \in \mathcal{X}} F(x, z^k) \qquad z^{k+1} := \operatorname*{argmin}_{z \in \mathcal{Z}} F(x^{k+1}, z). \tag{2}$$

This alternating scheme is well studied and achieves a sublinear convergence with rate $\mathcal{O}(1/K)$ [Beck, 2017] or under additional assumptions on the strong convexity or regularity conditions [Liao et al., 2024], a linear convergence $\mathcal{O}(\rho^K)$ with rate $\rho < 1$ [Beck and Tetruashvili, 2013].

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

The convergence speed described in the previous paragraph refers to the number of *outer iterations* k. However, the two subproblems in an outer iteration Eq. (2) do not necessarily have a closed-form solution and are instead often solved with an iterative scheme themselves. We refer to these as *inner iterations*. Using an iterative solver for the inner subproblems has both practical and theoretical disadvantages: it is computationally costly (resulting in long wall-clock solving times) and limits the usability of many theoretical results, which often assume the subproblems are solved exactly. Our main goal is to replace the updates in Eq. (2) with an inexact solver that can be implemented, for example, using a neural network. This allows us to compute the next iterate significantly faster and avoid the computational bottleneck. In particular, our contributions can be summarized as follows:

- In Section 3, we derive a worst-case error bound of the inexact alternating scheme (Theorem 1) under the quadratic growth assumption and prove linear and sublinear convergence under suitable conditions on the decay of the error terms (Corollary 1).
- Based on the inexact alternating scheme, we take a probabilistic viewpoint and derive a novel learning-based approach to accelerate inexact alternating minimization by predicting the solution to the inner subproblems. We propose a greedy training strategy that optimizes the networks to solve each subproblem in isolation (Section 4.1).
- We derive a fixed-horizon PAC-Bayes bound on the expected optimality gap of the learned scheme under the posterior over its weights (Theorem 2). With a random-walk prior over the per-iteration weights, the complexity term reduces to the parameter drift across iterations.

In numerical experiments (Sections 5 and 6), we apply both inexact and learned methods to solve CT reconstruction with dictionary-based regularization [Rudzusika et al., 2022] and robust matrix completion [Tao and Yuan, 2011]. By predicting the solution to the subproblems with the greedily-trained networks we accelerate the inexact alternating scheme by an order of magnitude at matched accuracy.

2 Related work

2.1 Data-driven optimization

Data-driven optimization methods are justified in settings where structurally similar problems need to be solved repeatedly [Chen et al., 2022]. This allows the upfront cost of training to be amortized by solving many subsequent problems of the same type. Learned methods show strong empirical performance [Li and Malik, 2016] but obtaining guarantees can be challenging [Amos, 2023].

Learning initializations One natural approach is to learn the initial guess for an optimization algorithm. Sambharya et al. [2024] train neural networks end-to-end to warm start fixed point optimization algorithms, and Sjölund and Bånkestad [2022] predict the initialization of an alternating primal-dual scheme. Along similar lines, Schmidtobreick et al. [2026] learn to warm start dual active-set methods. These approaches inherit the convergence properties of existing methods. For alternating schemes a good starting guess is readily available as the solution to the subproblem from the previous iteration.

Learning iterative updates To derive convergence guarantees for learned updates, data-driven optimization methods restrict the allowed predictions. The safeguarding scheme proposed by Heaton et al. [2023] obtains convergence of the learned iterates using the theory of fixed-point operators, which results in asymptotic convergence. Similarly, Prémont-Schwarz et al. [2022] prove the asymptotic convergence of their proposed safeguarding mechanism based on choosing safe updates of the iterates. Liu et al. [2023] formalize and provide necessary conditions on the learned iterates of first-order methods to ensure asymptotic convergence that can be implemented for practical problems, for example, by learning preconditioners [Häusner et al., 2024, Doerks et al., 2026]. Along similar lines to our training strategy, Fahy et al. [2024] propose to learn a preconditioner in a greedy fashion.

Banert et al. [2020] learn the parameters of optimization algorithms, enforcing certain conditions on the learned parameters to retain the asymptotic convergence behavior. Sambharya and Stellato [2026] learn hyper-parameters of optimization algorithms but require reversion to a safe parameter region to inherit asymptotic convergence. Further, Banert et al. [2024] learn a deviation that restricts the updates to some a priori known set of provably convergent updates. Similar to our approach, Tan et al. [2023] provide a worst-case analysis of their learned mirror descent framework, but the obtained bound does not guarantee asymptotic convergence in general. For QP solvers, Gao et al. [2025] propose a sequence-based approach to learn the subproblem solution. We instead formulate a learning problem

for each iteration in isolation and analyze the combined convergence of separate networks. By considering the continuous-time dynamics of optimization algorithms, Xie et al. [2025] derive a linear convergence rate of learned updates. Further, Martin et al. [2025] characterize learned optimizers with a linear convergence rate. More recently, Sambharya and Stellato [2025] and Sucker et al. [2025] propose to use data-driven probabilistic guarantees both to estimate the performance of classical optimizers on parametric problems and to directly train learned optimizers on the PAC-Bayes guarantees.

Learned image reconstruction Instead of first deriving an optimization problem that is then solved, several approaches directly integrate the learning and optimization by incorporating the regularizer implicitly into the learned optimizer, which limits both interpretability and theoretical analysis [Mukherjee et al., 2023]. The seminal work by Gregor and LeCun [2010] learns sparse coding and Adler and Öktem [2018] perform reconstruction with a neural network obtained by unrolling the primal-dual hybrid gradient method for minimizing the ℓ_2 distance. For alternating schemes, Ding et al. [2023] learn the regularization term instead of a solution to an optimization problem similar to Xin et al. [2022] where the method is applied to half-quadratic splitting. Further, Fahes et al. [2022] unroll a linearized alternating scheme to learn the algorithm's hyperparameters and variables.

2.2 Inexact optimization

Inexact optimization methods have a long history of explicitly modeling suboptimal solutions to the subproblems while maintaining asymptotic convergence or proving worst-case bounds under the common assumption of error summability [Güler, 1992]. In the context of alternating minimization, Robini et al. [2016] solve related half-quadratic splitting problems inexactly under KL regularity conditions. The related primal-dual ADMM [Eckstein and Bertsekas, 1992] has also been studied extensively in the inexact setting [Ouyang et al., 2015, Xie et al., 2017]. However, we restrict ourselves to a class of alternating optimization schemes that naturally arise in inverse problems such as sparse signal recovery [Wang, 2024] and automatic control [Aravkin et al., 2014].

3 Inexact scheme

In this section, we propose an inexact scheme to solve alternating minimization problems with quadratic coupling terms between the variable blocks as in Eq. (1). Specifically, in each iteration k, we allow for inexact solutions $\tilde{x}^{k+1}$ and $\tilde{z}^{k+1}$ to the subproblems in the alternating scheme Eq. (2):

$$\tilde{x}^{k+1} \approx \operatorname*{argmin}_{x\in\mathcal{X}} F(x, \tilde{z}^k), \qquad \tilde{z}^{k+1} \approx \operatorname*{argmin}_{z\in\mathcal{Z}} F(\tilde{x}^{k+1}, z). \tag{3}$$

To measure the performance of the inexact scheme and derive convergence results, we denote the function gap introduced by the inexact scheme in each step respectively as:

$$\xi^{k+1} := F(\tilde{x}^{k+1}, \tilde{z}^k) - F(x^{k+1}, \tilde{z}^k) \qquad \zeta^{k+1} := F(\tilde{x}^{k+1}, \tilde{z}^{k+1}) - F(\tilde{x}^{k+1}, z^{k+1}), \tag{4}$$

where $x^{k+1} := \operatorname{argmin}_{x\in\mathcal{X}} F(x, \tilde{z}^k)$ and $z^{k+1} := \operatorname{argmin}_{z\in\mathcal{Z}} F(\tilde{x}^{k+1}, z)$ are the exact minimizers of the respective subproblems. Both quantities are non-negative by construction. To simplify the analysis of the inexact scheme, we also introduce the combined error $\Delta^{k+1} := \xi^{k+1} + \zeta^{k+1}$.

Next, we derive a worst-case bound of the optimality gap based on the iterates from the alternating scheme and the sequence of combined errors $\{\Delta^k\}_{k=1}^K$ introduced by the inexact solvers. The key ingredient in the proof of the worst-case bounds from Theorem 1 and convergence is the following lemma, which shows a one-step inequality for the iterates of the inexact alternating scheme.

Lemma 1. Assume that F is a convex objective of the form Eq. (1) and that the iterates $\{\tilde{x}^k, \tilde{z}^k\}_{k=0}^K \subseteq \mathcal{X}\times\mathcal{Z}$ are obtained using the inexact scheme Eq. (3). Then for any iteration $k \geq 0$ it holds for any $(x, z) \in \mathcal{X}\times\mathcal{Z}$ that the new function value $\tilde{F}^{k+1} := F(\tilde{x}^{k+1}, \tilde{z}^{k+1})$ satisfies

$$\tilde{F}^{k+1} + \frac{1}{2(1+\alpha)}\|B(z-\tilde{z}^{k+1})\|^2 \leq F(x,z) + \frac{1+\alpha}{2}\|B(z-\tilde{z}^k)\|^2 + \frac{1+\alpha}{\alpha}\Delta^{k+1} \tag{5}$$

where $\alpha > 0$ is an arbitrarily chosen constant.

To prove convergence of the inexact scheme we utilize the following assumption which states that the function is bounded from below by a quadratic form in the B-norm distance to the optimal set. Let $Z^* := \operatorname{argmin}_{(x,z)\in\mathcal{X}\times\mathcal{Z}} F(x,z)$ denote the set of minimizers and F^* the optimal value.

Assumption 1 (Quadratic growth)**.** We assume that the function gap between the inexact iterates $\{\tilde{x}^k, \tilde{z}^k\}$ and the optimal set Z^* can be bounded in B-norm distance with parameter $\mu > 0$ as:

$$\frac{\mu}{2} \inf_{(x',z')\in Z^*} \|B(z' - \tilde{z}^k)\|^2 \leq F(\tilde{x}^k, \tilde{z}^k) - F^*. \tag{6}$$

This assumption is implied if the function $F(x, z)$ is jointly strongly convex in both arguments but belongs to a weaker class of assumptions since the inequality only has to hold around the optimal set [Liao et al., 2024]. Now, we present the following theorem, which establishes a worst-case bound on the objective in terms of the initial iterates and the errors introduced by the inexact scheme.

Theorem 1 (Worst-case bound for the inexact scheme)**.** Let the objective function F in Eq. (1) be convex with F^* as its global minimum value. The iterates $\{\tilde{x}^k, \tilde{z}^k\}_{k=0}^K \subseteq \mathcal{X} \times \mathcal{Z}$ are obtained from the inexact alternating scheme in Eq. (3) with an arbitrary initial $(\tilde{x}^0, \tilde{z}^0) \in \mathcal{X} \times \mathcal{Z}$. Under Assumption 1 with some fixed μ and choosing the parameter α as in Lemma 1 from the interval

$$\alpha \in \left(0, \tfrac{\mu - 2 + \sqrt{\mu^2+4}}{2}\right), \tag{7}$$

and with the initial Lyapunov energy function defined as

$$\Phi_0 := \tilde{F}^0 - F^* + \tfrac{1}{2(1+\alpha)} \inf_{(x',z')\in Z^*} \|B(z' - \tilde{z}^0)\|^2, \tag{8}$$

we can bound the function gap for any iteration $K > 0$ as:

$$\tilde{F}^K - F^* \leq \rho^K \Phi_0 + \frac{1+\alpha}{\alpha} \sum_{k=1}^{K} \rho^{K-k} \Delta^k, \tag{9}$$

where $\rho = (1+\alpha)^2/(1+\mu(1+\alpha)) < 1$ is the contraction rate of the bound.

Remark 1. The parameter α in Theorem 1 is a hyper-parameter of the bound that can be used to trade-off between the solver inaccuracy and the convergence rate of the error bound. In particular, if the errors Δ^k are zero, we can choose $\alpha \to 0$ and recover the exact linear convergence rate $\rho \to 1/(1+\mu)$. Conversely, increasing α reduces the amplification of the error term $(1+\alpha)/\alpha$ but worsens the contraction rate ρ towards 1 slowing down the outer convergence.

The first term in the bound decays geometrically with rate ρ, so to establish convergence of the scheme, we are required to analyze how the error Δ^{k+1} decays. Taking into account the structure of the accumulated errors, we can make the following key observation before deriving the convergence.

Remark 2. In the bound from Theorem 1, early error terms are discounted exponentially. Thus solvers that become more accurate over time can compensate for large errors made early on.

Corollary 1. Under the same assumptions as in Theorem 1, the convergence of the function value is determined by the decay of the combined error Δ^k. We distinguish three separate regimes:

i. *Geometric decay.* If $\Delta^k \leq Cq^k$ for some constants $C > 0$ and $q \in (0, 1)$, we retain linear convergence of the inexact scheme with rate $\mathcal{O}(\max(\rho, q)^K)$.

ii. *Polynomial decay.* If $\Delta^k \to 0$, we achieve convergence to the optimum where the sublinear rate is determined by the speed of the decay. In particular, for polynomial decay $\Delta^k \leq C/k^q$ with $q > 0$, we obtain sublinear convergence with a rate of $\mathcal{O}(1/K^q)$.

iii. *Eventually bounded errors.* If there exist $\bar{k} \in \mathbb{N}$ and $\Delta^{\text{tail}} > 0$ such that $\Delta^k \leq \Delta^{\text{tail}}$ for all $k > \bar{k}$, the function gap converges to a neighborhood of the optimum: $\limsup_{K\to\infty} \tilde{F}^K - F^* \leq \Delta^{\text{tail}}(1+\alpha)/[\alpha(1-\rho)]$.

The final case in Corollary 1 is particularly relevant for the learned scheme in the next section. While we cannot guarantee that the error terms vanish for the neural networks, we provide conditions under which the objective gap for a fixed horizon K is bounded with high probability over the training data.

4 Learned scheme

In this section, we approximate the solution to each subproblem in Eq. (3) with a learned mapping. We implement each inexact solver using a neural network that takes the current iterate together with the gradient of the corresponding partial objective and outputs the next iterate of the inexact scheme:

$$\tilde{x}^{k+1} = \phi_{\theta^{k+1}}\big(\tilde{x}^k,\, \nabla_x F(\tilde{x}^k, \tilde{z}^k)\big) \qquad \tilde{z}^{k+1} = \psi_{\vartheta^{k+1}}\big(\tilde{z}^k,\, \nabla_z F(\tilde{x}^{k+1}, \tilde{z}^k)\big)\,. \tag{10}$$

Each network thus realizes a learned first-order map for the corresponding subproblem. For non-smooth problems, the gradient input refers to the gradient of the smooth part while the proximal behavior of the non-smooth part is absorbed into the network architecture. To ensure the learned iterates are bounded, the last layer of the network projects the data onto a box of predefined dimensions derived from the problem domain by clipping the network outputs $\Pi_{[l,u]}$. Here, θ^{k+1} and ϑ^{k+1} are the neural network parameters of each network at iteration $k+1$. We collect both into the joint parameter set $\Theta^{k+1} := (\theta^{k+1}, \vartheta^{k+1})$ and write $\Theta^{1:k} := (\Theta^1, \dots, \Theta^k)$ for the parameters up to iteration k.

Convergence results in Section 3 apply to a fixed objective. To learn the solution map, we consider a family of objectives $\{F_y\}_{y\in\mathcal{Y}}$ indexed by a random variable $y \sim p_y$ with bounded support [Amos, 2023]. Each y yields an objective as in Eq. (1) so iterates $\tilde{x}_y^k, \tilde{z}_y^k$ and errors Δ_y^k are random variables. For fixed weights $\Theta^{1:K}$, these are functions of y alone and expectations $\mathbb{E}[\,\cdot\,]$ are with respect to p_y.

For each $y \in \mathcal{Y}$, the bound in Theorem 1 applies to the corresponding instance F_y. To characterize the alternating scheme's behavior over a distribution of instances on which a model can be trained, we take the expectation with respect to p_y. The projection $\Pi_{[l,u]}$ confines the learned iterates to a bounded set for every parameter value. We assume Assumption 1 holds on this set with a common constant μ for all y in the support of p_y, so that α and ρ are independent of the instance and network weights. Together with the bounded support of p_y, the following result holds as an analogous version in expectation:

$$\mathbb{E}[\tilde{F}_y^K - F_y^*] \;\le\; \rho^K\, \mathbb{E}\,[\Phi_0] + \frac{1+\alpha}{\alpha} \sum_{k=1}^{K} \rho^{K-k}\, \mathbb{E}[\Delta_y^k]. \tag{11}$$

If the conditions of Corollary 1 hold for the expected errors $\mathbb{E}[\Delta_y^k]$, the same argument and linearity of expectation extend its three convergence regimes to convergence in expectation.

4.1 Greedy training

We now use the bound Eq. (11) to motivate a greedy training objective. On the right-hand side, the sum of weighted per-iteration errors is the only term that depends on the network. Minimizing jointly over all $\Theta^{1:K}$ is computationally expensive. Each error Δ_y^k depends on the whole trajectory through the previous iterates so the gradient computation requires backpropagation through k unrolled iterations. Memory therefore grows linearly in K and gradients through long unrolls can be unstable [Metz et al., 2019]. Instead, we only minimize the errors Δ_y^k greedily one at a time in their order in the scheme.

For each outer iteration k, with $\Theta^{1:k-1}$ held fixed, we first train the x-network ϕ_{θ^k} to minimize $\mathbb{E}[\xi_y^k]$. With θ^k also held fixed, we then train the z-network ψ_{ϑ^k} to minimize $\mathbb{E}[\zeta_y^k]$. Since ζ_y^k also depends on θ^k through $\tilde{x}_y^k$, the two phases do not in general minimize $\mathbb{E}[\Delta_y^k]$ jointly. We refer to this strategy as *greedy training*: each network is trained on its own subproblem, with all previously trained networks held fixed and no anticipation of downstream iterations, so it does not minimize the bound in general.

Minimizing the population risk requires access to the unknown distribution p_y. We replace the expectation by an empirical mean over i.i.d. samples $\mathcal{D} = \{y_i\}_{i=1}^N$ drawn from the distribution p_y. Since $\Delta_y^k = \xi_y^k + \zeta_y^k$, the training phases minimize, one after the other, the two terms of the empirical risk

$$\hat{\Delta}^k(\Theta^{1:k}; \mathcal{D}) := \tfrac{1}{N} \sum_{i=1}^{N} \Delta_{y_i}^k(\Theta^{1:k}) = \tfrac{1}{N} \sum_{i=1}^{N} \xi_{y_i}^k(\Theta^{1:k}) + \tfrac{1}{N} \sum_{i=1}^{N} \zeta_{y_i}^k(\Theta^{1:k}). \tag{12}$$

The error terms we minimize are defined in Eq. (4) and also depend on their respective reference values, which are, however, independent of the parameters updated in that phase. The training loss is therefore equivalent to the objective evaluated at the predicted iterate itself. This yields an unsupervised training scheme that does not require ground-truth subproblem minimizers as targets.

Algorithm 1 Greedy training algorithm for learned inexact alternating minimization

input: Training data $\mathcal{D} = \{y_i\}_{i=1}^N \subseteq \mathcal{Y}$ of i.i.d. samples, initial parameters (θ^0, ϑ^0).
for $k = 0, 1 \dots K-1$ **do**
  Warm-start the learning task $(\theta^{k+1}, \vartheta^{k+1}) \leftarrow (\theta^k, \vartheta^k)$
  // Phase 1: train the x-network with $\Theta^{1:k}$ held fixed
  Update θ^{k+1} by SGD to minimize the empirical risk $\frac{1}{N}\sum_{i=1}^N F_{y_i}(\tilde{x}_{y_i}^{k+1}(\theta^{k+1}),\, \tilde{z}_{y_i}^k)$
  // Phase 2: train the z-network with $\Theta^{1:k}$ and θ^{k+1} held fixed
  Update ϑ^{k+1} by SGD to minimize the empirical risk $\frac{1}{N}\sum_{i=1}^N F_{y_i}(\tilde{x}_{y_i}^{k+1},\, \tilde{z}_{y_i}^{k+1}(\vartheta^{k+1}))$
output: Learned optimizer for the subproblems of each iteration $\{\phi_{\theta^k}, \psi_{\vartheta^k}\}_{k=1}^K$

Algorithm 1 formalizes the greedy training procedure described above. We warm-start each new pair of networks from the weights trained at the previous iteration. As the iterates approach the minimizer, consecutive subproblems become increasingly similar, so the weights from the previous iteration provide a strong initialization for iteration k. The warm-start lets us cut the training budget at later iterations, where each pair of networks only requires fine-tuning rather than training from scratch.

A natural alternative to the proposed greedy approach is end-to-end training, in which all K networks are optimized jointly through the full alternating scheme. With end-to-end training, however, the networks are no longer trained to solve the per-iteration subproblems, so the training objective no longer matches the per-iteration errors in the bound and the bound no longer motivates the training. In addition, greedy training avoids backpropagation through the full K-iteration scheme, which reduces memory consumption and accelerates training [Metz et al., 2019].

4.2 Generalization bound

We now bound the expected optimality gap of the learned scheme on unseen instances at a fixed horizon K. Greedy training fits the networks to the empirical risk $\hat{\Delta}^k$, while the bound in expectation in Eq. (11) depends on the population risk $\mathbb{E}[\Delta_y^k]$. A PAC-Bayes bound [Alquier, 2024, Catoni, 2007, McAllester, 1999] closes this gap for a randomized predictor drawn from a posterior $\mathbb{Q}$, at the cost of the divergence $\mathrm{KL}(\mathbb{Q}\|\mathbb{P})$ to a prior $\mathbb{P}$ fixed before seeing the data [Parrado-Hernández et al., 2012].

Training on the N training samples $\mathcal{D}$ produces a trajectory of weights $\hat{\Theta}^1, \dots, \hat{\Theta}^K$, one network pair per iteration. The bound concerns a randomized version of this trajectory: Gaussian noise of variance ς^2 is added to each increment $\hat{\Theta}^k - \hat{\Theta}^{k-1}$, so that a random walk runs alongside the trained weights. The posterior $\mathbb{Q}$ is the distribution of this walk given $\mathcal{D}$. The prior $\mathbb{P}$ is the same walk with zero-mean increments from the data-independent initialization $\Theta^0 =: \hat{\Theta}^0$, which extends the initialization-centered Gaussian prior [Dziugaite and Roy, 2017] to a sequence of networks. Since the two walks have the same noise and differ only in their mean increments, their divergence is the cumulative parameter drift $\mathrm{KL} := \mathrm{KL}(\mathbb{Q}\|\mathbb{P}) = \frac{1}{2\varsigma^2}\sum_{k=1}^K \|\hat{\Theta}^k - \hat{\Theta}^{k-1}\|^2$.

Theorem 2 (Fixed-horizon generalization bound)**.** Let $\mathbb{P}$ and $\mathbb{Q}$ be the prior and posterior defined above with $\varsigma > 0$ fixed before training. Pick a fixed α that satisfies Eq. (7) with the common constant μ, and let ρ be as in Theorem 1. Suppose that $0 \le \Delta_y^k(\Theta^{1:k}) \le M$ for all y in the support of p_y, all $\Theta^{1:k}$, and $k = 1, \dots, K$. Fix $\beta \in (0,1)$ and an integer $J \ge 0$, and let $C := \log((2J+1)/\beta)$ and $\lambda^\star := \frac{2(1-\rho)}{M}\sqrt{2N(\mathrm{KL}+C)}$. Then, with probability at least $1-\beta$ over the N i.i.d. training samples, the following holds: if $2^{-J} \le \lambda^\star \le 2^J$, then

$$\mathbb{E}_{\mathbb{Q}}\mathbb{E}_y[\tilde{F}_y^K - F_y^*] \le \rho^K\, \mathbb{E}_y[\Phi_0] + \tfrac{1+\alpha}{\alpha}\left(\textstyle\sum_{k=1}^K \rho^{K-k}\mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k] + \frac{3M}{4(1-\rho)}\sqrt{(\mathrm{KL}+C)/N}\right). \quad (13)$$

The three terms of Eq. (13) align with the learned scheme. The initial Lyapunov energy is independent of the networks and decays at the linear rate ρ. Each term in the sum of discounted training risks is the empirical risk $\hat{\Delta}^k$ of Eq. (12) averaged over $\mathbb{Q}$. Early iterations are discounted as discussed in Remark 2. The last term on the right-hand side of Eq. (13) depends on the trained weights only through the parameter drift in KL, which warm-starting in Algorithm 1 helps to keep small.

To run the scheme beyond a trained depth K_0 fixed before training, we reuse the last trained networks, $\Theta^k = \Theta^{K_0}$ for additional iterations $k > K_0$, so the divergence stays at its value KL_{K_0}. Splitting the discounted sum in Eq. (13) at K_0 then gives, for any fixed $K \geq K_0$ (see Corollary 2),

$$\mathbb{E}_{\mathbb{Q}}\mathbb{E}_y[\tilde{F}_y^K - F_y^*] \leq \frac{1+\alpha}{\alpha(1-\rho)}\left(\hat{\Delta}^{>K_0} + \tfrac{3M}{4}\sqrt{(\mathrm{KL}_{K_0} + C)/N}\right) + \rho^{K-K_0} B_{K_0}, \tag{14}$$

where $\hat{\Delta}^{>K_0} := \max_{K_0 < k \leq K} \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k]$ is the largest training risk of the reused networks and B_{K_0} depends only on the first K_0 iterations. The second term decays at the rate ρ in K, so for large horizons the first term dominates. This term matches the neighborhood in case iii. of Corollary 1, but the unknown error bound Δ^{tail} is now replaced by quantities computed from the training data. It stays small as long as the reused networks remain accurate on the subproblems beyond the training horizon K_0.

5 Dictionary regularized CT image reconstruction

The motivating application of our learned optimization approach is the reconstruction of computed tomography (CT) images using dictionary-based regularization in low-dose settings [Rudzusika et al., 2022]. For this particular use-case, the objective function has the form of Eq. (1) and is given by[1]

$$F(x, z; y) = \|Tx - y\|_{\mathcal{Y}}^2 + \lambda_1 \|Dz - x\|_{\mathcal{X}}^2 + \lambda_2 \|z\|_1 \tag{15}$$

where the linear operator $T : \mathcal{X} \to \mathcal{Y}$ models how noise-free tomographic data is generated (forward operator), and $D : \mathcal{Z} \to \mathcal{X}$ is the linear synthesis operator for the fixed dictionary that reconstructs the image from the dictionary coefficients, which is implemented as a convolution operator. We refer to the optimization variable x as the reconstructed image and z as the dictionary coefficients. The optimization problem is parameterized by the problem data y given as noisy indirect observations.

Proposition 1. For every bounded set of data $\mathcal{Y}_0$ and every bounded set $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{Z}$, there is a constant $\mu > 0$ such that, for every $y \in \mathcal{Y}_0$, the dictionary-regularized CT objective from Eq. (15) satisfies the quadratic growth condition (Assumption 1) at every point of $\mathcal{S}$.

Assuming p_y has bounded support and the learned iterates lie in a bounded set (Section 4), Proposition 1 gives one constant μ for all instances, as required in Section 4.2. The constant depends on a Hoffman constant that cannot be computed at this scale, so we estimate μ numerically along the learned iterates in Fig. 5 (Appendix D.7). We now compare the learned scheme to the inexact baseline that solves each subproblem with first-order methods with varying budgets for the number of inner iterations.

5.1 Learned vs. inexact solvers

We evaluate two variants of the learned scheme. The first uses the greedy training of Section 4.1, which directly targets the per-iteration risk that Theorem 1 bounds. The second approach applies additional end-to-end fine-tuning to the greedily trained networks. Figure 1 shows the trade-off between the wall-clock runtime and the final relative objective gap for 20 outer iterations, traced out as a Pareto front by sweeping the inner-iteration budget of the inexact baseline from 1 to 100 inner iterations. Both learned variants lie well below the frontier traced by the inexact baseline. Over 5 independent training runs, the greedy variant reaches a relative objective gap of 0.043 ± 0.001 (mean $\pm$ standard deviation) in 1.1 seconds. The inexact baseline needs roughly 60 inner iterations and 14 seconds to match this gap, a $13\times$ runtime reduction. The fine-tuned variant lowers the gap further to 0.033 ± 0.001 in 0.54 seconds. Matching this performance requires roughly 75 inner iterations and 16 seconds, leading to an overall $30\times$ reduction.

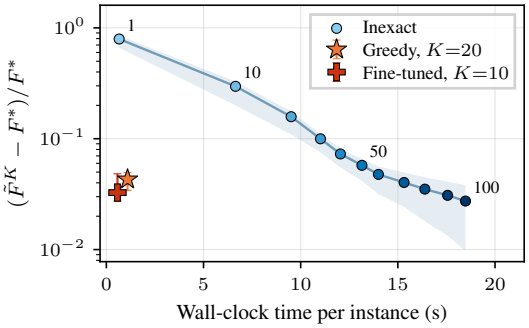


Figure 1: Runtime vs. final relative objective gap of learned and baseline methods: medians with 5th–95th percentiles and numbers give the inner iterations of the inexact baseline.

[1] To obtain the form of Eq. (1), we rescale the objective by $1/(2\lambda_1)$ which does not change the minimizer.

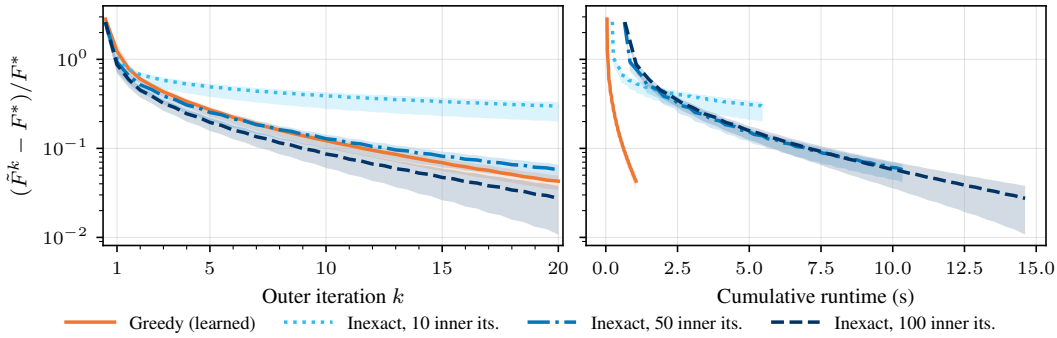


Figure 2: Relative objective gap vs. outer iteration k (left) and cumulative runtime (right) over 20 iterations: medians with 5th–95th percentiles over test instances.

5.2 Convergence of alternating scheme

In Fig. 2 we show the per-iteration relative objective gap for the greedy learned scheme and three inexact baselines as a function of the outer iteration k (left) and the cumulative runtime (right). After the first few outer iterations, all four trajectories decay roughly linearly on the log scale in line with the bound derived in Theorem 1. Plotted against wall-clock time, the trajectories of the inexact baselines span an order of magnitude more time, since they are dominated by the inner solver times.

In the first few outer iterations, the learned scheme is slightly worse in terms of objective value than the inexact baseline with 50 inner iterations and only catches up around $k \approx 8$. This is consistent with Remark 2: errors in early iterations enter the bound with a discount factor and therefore tolerate larger magnitudes. The convergence of the learned scheme is therefore governed by the approximation error of the neural network, which we evaluate empirically in the following section.

5.3 Solving of subproblems

Here, we analyze how well the greedily-trained networks ϕ and ψ solve their respective subproblems from the inexact scheme Eq. (3) in isolation for each iteration k. In other words, we are examining the performance independently of error accumulation across iterations. Given the iterates from Eq. (10), we evaluate the function gaps ξ_y^k and ζ_y^k defined in Eq. (4) by comparing the network predictions to an approximate ground truth solution obtained from a first-order method starting at the current iterate.

The empirical performance of the greedily trained network on the test data is shown in Fig. 3. On the left plot, the x-subproblem in the first iteration has a larger error which rapidly decreases. The error for the z-subproblem decreases as well on the right plot but the decay is slower and the magnitude of the error is larger in later iterations. This is unsurprising: the z-subproblem in Eq. (15) involves a non-smooth ℓ_1 term and contains significantly more variables, making it an inherently harder optimization problem. However, both networks are able to solve the subproblem to a high accuracy and improve the subproblem solution quality with additional outer iterations.

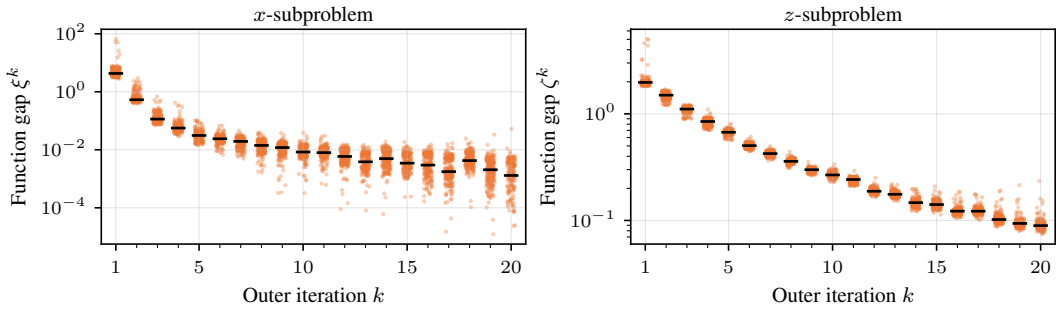


Figure 3: Network approximation error of both subproblems across iterations k; dots are test instances and black bars medians.

Table 1: Comparison of different learning based approaches to solve Eq. (15) showing the mean objective and standard deviation across 210 test instances.

Method	Training time ↓	Inference steps ↓	Inference time ↓	Test objective ↓
Fully-amortized	17h28m	**1**	**0.35**s	45.547 ± 1.996
Semi-amortized	34h11m	10	0.51s	24.425 ± 0.508
Greedy (ours)	**11h21m**	10	0.53s	25.824 ± 0.362
+ fine tuning	+5h55m	10	0.54s	$\mathbf{23.840 \pm 0.468}$
Greedy (ours)	16h27m	20	1.05s	24.043 ± 0.328

5.4 Comparison to other learning approaches

Additionally, we compare both of our variants to two established learned optimization methods without convergence guarantees. The first approach uses a *fully-amortized* network to predict the solution. In other words, the network is trained to directly predict the optimal solution to the joint optimization problem in Eq. (1) in a single forward pass [Amos, 2023]. The second approach uses a *semi-amortized* approach based on algorithm unrolling [Monga et al., 2021]. The network uses the same architecture as our proposed method but we apply a forward pass through all layers combined and train the model end-to-end. More details about the model implementations can be found in Appendix D.4.

The fine-tuned greedy variant achieves the lowest test objective in Table 1. It reaches 23.84 at $K = 10$ while greedy alone reaches 24.04 at $K = 20$, both below the 24.43 of the best baseline. The fully-amortized baseline performs substantially worse, confirming that one-shot prediction is the wrong approximation class for this problem. The end-to-end unrolling approach cannot be extended to more iterations because backpropagation through more unrolled iterates exceeds GPU memory. In contrast, greedy training optimizes one iteration at a time and so scales freely with K. At its maximum supported depth ($K = 10$), end-to-end unrolling reaches a lower objective than greedy alone (24.43 vs. 25.82), but greedy trains three times faster (11h21m vs. 34h11m). Adding a few steps of end-to-end fine-tuning to the greedy networks closes this gap and produces the lowest objective overall (23.84), at roughly half the training cost of the unrolled baseline.

6 Robust matrix completion

A second, synthetic, example is the robust matrix completion problem [Tao and Yuan, 2011], also known as low-rank matrix completion with sparse corruptions. Given partial observations of an $n \times m$ matrix y that is the sum of a low-rank and a sparse matrix, as for example in video sequences, we aim to recover both components by minimizing [Klopp et al., 2017]

$$F(x, z; y) = \lambda_1 \|x\|_* + \frac{1}{2}\|P_\Omega(y - x - z)\|_F^2 + \lambda_2 \|z\|_1, \tag{16}$$

where the optimization variables x and z are $n \times m$ matrices. The sampling operator P_Ω keeps the entries in the observed index set Ω and sets all other entries to zero. The nuclear norm favors a low-rank component x, the ℓ_1 norm promotes a sparse z, and the masked Frobenius norm couples the two variables. The regularization parameters λ_1 and λ_2 are chosen to minimize the reconstruction error of the minimizer with respect to the ground truth. We consider 100×100 matrices of rank 5 with 40% of the entries observed and 5% of the entries corrupted by sparse noise (Appendix E.4).

Proposition 2. If a minimizer of Eq. (16) satisfies the standard robust PCA assumptions of strict complementarity and restricted injectivity, then it is the unique minimizer and the robust completion objective satisfies the quadratic growth condition (Assumption 1) on every sublevel set.

The constant in this proposition depends on the problem instance. In Appendix E.3 we obtain a common constant on an event of instances and combine the probability of its complement with Eq. (13).

Methods The z-subproblem has a closed-form solution, since it is separable in the entries of z. The alternating scheme therefore computes the z-update exactly, so the z-error ζ_y^k in Eq. (4) vanishes. The x-subproblem has no closed-form solution, and each proximal gradient step requires singular value thresholding (SVT) with a computationally expensive singular value decomposition (SVD).

In our learned method, the network unrolls a single SVT step in the style of LISTA [Gregor and LeCun, 2010] and learns a step size and one threshold per singular value, that is, 101 parameters per iteration k. We compare against truncated FISTA, a tuned proximal gradient method, and ADMM, whose reformulation gives closed-form updates with a single SVD per iteration (Appendix E.6).

Results Figure 4 shows the relative optimality gap on 128 unseen test instances at an equal budget of one SVD per outer iteration. Up to the trained depth $k = 8$, the learned scheme attains the lowest gap, by a factor of 1.4 to 15 over the best baseline (Table 6 in Appendix E.8). At $k = 8$, FISTA needs $5\times$ more wall-clock time to reach the objective of the learned scheme (Table 5). Beyond the training horizon, we reuse the last trained networks as in Corollary 2. The gap further decreases until $k \approx 14$ and then plateaus, which is consistent with the error term of the late training risk in Corollary 2. The tuned proximal gradient method and ADMM keep decreasing and overtake the learned scheme at $k = 20$ and $k = 29$, respectively.

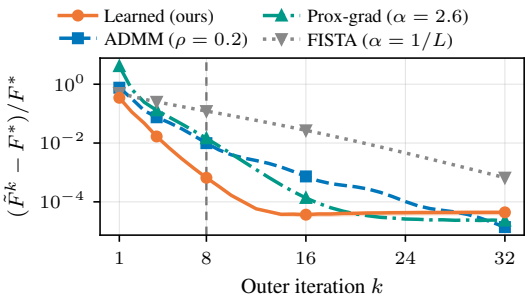


Figure 4: Relative objective gap for robust matrix completion against the outer iteration k with equal computational budget per iteration. The learned curve is the mean over 5 training seeds. The dashed vertical line marks the trained depth $k = 8$.

7 Discussion & Conclusion

Limitations and future work The convergence analysis in Section 3 relies on joint convexity with quadratic coupling as in Eq. (1) and the quadratic growth condition (Assumption 1) which excludes bi-convex objectives common in alternating minimization. A natural next step is to relax these assumptions. Further, the empirical evaluation covers one imaging modality with a single held-out patient at a fixed noise level and fixed regularization weights λ_1, λ_2, and synthetic robust matrix completion instances, so variability across patients, problem parameters, and real data is not evaluated. Further scaling the imaging from 2D to 3D CT reconstruction can lead to even larger speedups.

Theorem 2 assumes that training and evaluation instances are drawn from the same distribution p_y, and thus does not cover distribution shift, a standard limitation of learning-to-optimize methods. Moreover, the bound in Theorem 2 concerns the posterior $\mathbb{Q}$ at a horizon fixed in advance, while the deployed point estimate is covered only by the per-instance bound of Theorem 1. Evaluated with an empirical estimate of μ and other plug-in constants, the PAC-Bayes bound on the discounted error sum is informative for robust matrix completion but not for CT reconstruction, and the resulting bound on the optimality gap is vacuous for both (Appendices D.10 and E.9).

The largest speedup in the experiments comes from end-to-end fine-tuning, where the training objective no longer coincides with the per-iteration risk in the bound, so the analysis covers these networks but does not motivate their training directly. Our greedy scheme uses separate weights Θ^k for every trained iteration, which scales the parameter count and inference-time memory linearly. Reusing the last trained networks caps this growth at the cost of an error term in the bound that does not vanish.

Conclusion In this paper, we accelerate alternating minimization schemes of convex objectives with quadratic coupling by replacing each subproblem with an inexact solver that can be implemented by a neural network. For the inexact scheme, Theorem 1 gives a worst-case bound on the optimality gap, and Corollary 1 characterizes three convergence regimes dependent on the decay of the per-iteration error. Building on this bound, we train a sequence of neural networks greedily by iteration and combine the inexact analysis with a PAC-Bayes argument to obtain a fixed-horizon high-probability guarantee on the expected optimality gap under the posterior of the trained network (Theorem 2).

For the dictionary-regularized CT reconstruction, the learned scheme traces out a runtime/accuracy frontier strictly below that of the inexact baseline and improves on both fully-amortized and unrolled learned alternatives. In the synthetic robust matrix completion problem, our method outperforms classical solvers at the trained depth under the same computational budget and remains ahead of them up to twice the trained depth.

Acknowledgments and Disclosure of Funding

We would like to thank the anonymous reviewers for their helpful feedback on earlier versions of this paper. This work was supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

Jonas Adler and Ozan Öktem. Learned primal-dual reconstruction. *IEEE transactions on medical imaging*, 37(6):1322–1332, 2018. doi: 10.1109/TMI.2018.2799231.

Jonas Adler, Holger Kohr, and Ozan Öktem. Operator discretization library (ODL), 2017. URL `https://github.com/odlgroup/odl`. MPL-2.0 License.

Pierre Alquier. User-friendly introduction to PAC-Bayes bounds. *Foundations and Trends in Machine Learning*, 17(2):174–303, 2024. doi: 10.1561/2200000100.

Brandon Amos. Tutorial on amortized optimization. *Foundations and Trends in Machine Learning*, 16(5):592–732, 2023. doi: 10.1561/2200000102.

Aleksandr Y. Aravkin, James V. Burke, and Gianluigi Pillonetto. *Optimization Viewpoint on Kalman Smoothing with Applications to Robust and Sparse Estimation*, pages 237–280. Springer Berlin Heidelberg, 2014. doi: 10.1007/978-3-642-38398-4_8.

Sebastian Banert, Axel Ringh, Jonas Adler, Johan Karlsson, and Ozan Öktem. Data-driven nonsmooth optimization. *SIAM Journal on Optimization*, 30(1):102–131, 2020. doi: 10.1137/18M1207685.

Sebastian Banert, Jevgenija Rudzusika, Ozan Öktem, and Jonas Adler. Accelerated forward-backward optimization using deep learning. *SIAM Journal on Optimization*, 34(2):1236–1263, 2024. doi: 10.1137/22M1532548.

Heinz H Bauschke and Patrick L Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408 of *CMS Books in Mathematics*. Springer, 2 edition, 2017. doi: 10.1007/978-3-319-48311-5.

Amir Beck. *First-order methods in optimization*. MOS-SIAM Series on Optimization. SIAM, 2017. doi: 10.1137/1.9781611974997.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. doi: 10.1137/080716542.

Amir Beck and Luba Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013. doi: 10.1137/120887679.

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):1–122, 2011. doi: 10.1561/2200000016.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011. doi: 10.1145/1970392.197039.

Olivier Catoni. PAC-Bayesian supervised classification: The thermodynamics of statistical learning. *Lecture Notes-Monograph Series*, 56:i–163, 2007. doi: 10.1214/074921707000000391.

Antonin Chambolle and Thomas Pock. A remark on accelerated block coordinate descent for computing the proximity operators of a sum of convex functions. *The SMAI Journal of computational mathematics*, 1:29–54, 2015. doi: 10.5802/smai-jcm.3.

Venkat Chandrasekaran, Sujay Sanghavi, Pablo A. Parrilo, and Alan S. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011. doi: 10.1137/090761793.

Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. Learning to optimize: A primer and a benchmark. *Journal of Machine Learning Research*, 23(189):1–59, 2022. URL http://jmlr.org/papers/v23/21-0308.html.

Kuanhong Cheng, Juan Du, Huixin Zhou, Dong Zhao, and Hanlin Qin. Image super-resolution based on half quadratic splitting. *Infrared Physics & Technology*, 105:103193, 2020. doi: 10.1016/j.infrared.2020.103193.

Ying Cui, Chao Ding, and Xinyuan Zhao. Quadratic growth conditions for convex matrix optimization problems associated with spectral functions. *SIAM Journal on Optimization*, 27(4):2332–2355, 2017. doi: 10.1137/17M1116325.

Chi Ding, Qingchao Zhang, Ge Wang, Xiaojing Ye, and Yunmei Chen. Learned alternating minimization algorithm for dual-domain sparse-view CT reconstruction. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 173–183. Springer, 2023. doi: 10.1007/978-3-031-43999-5_17.

Henri Doerks, Paul Häusner, Daniel Hernández Escobar, and Jens Sjölund. Learning to accelerate distributed ADMM using graph neural networks. In *Proceedings of The 8th Annual Learning for Dynamics and Control Conference*, volume 331, pages 1995–2020. PMLR, 2026. URL https://proceedings.mlr.press/v331/doerks26a.html.

Dmitriy Drusvyatskiy and Adrian S Lewis. Error bounds, quadratic growth, and linear convergence of proximal methods. *Mathematics of operations research*, 43(3):919–948, 2018. doi: 10.1287/moor.2017.0889.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the 33rd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017. URL https://www.auai.org/uai2017/proceedings/papers/173.pdf.

Jonathan Eckstein and Dimitri P Bertsekas. On the douglas—rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical programming*, 55(1): 293–318, 1992. doi: 10.1007/BF01581204.

Mohammad Fahes, Christophe Kervazo, Jérôme Bobin, and Florence Tupin. Unrolling PALM for sparse semi-blind source separation. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=aBVxf5NaaRt.

Patrick Fahy, Mohammad Golbabaee, and Matthias J Ehrhardt. Greedy learning to optimize with convergence guarantees. *ArXiv e-print*, math.OC(2406.00260), 2024. doi: 10.48550/arXiv.2406.00260.

Xi Gao, Jinxin Xiong, Linxin Yang, Akang Wang, Weiwei Xu, and Jiang Xue. A learning-based inexact ADMM for solving quadratic programs. *ArXiv e-print*, math.OC(2505.09391), 2025. doi: 10.48550/arXiv.2505.09391.

Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE transactions on Image Processing*, 4(7):932–946, 1995. doi: 10.1109/83.392335.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th international conference on machine learning (ICML)*, pages 399–406, 2010. URL https://dl.acm.org/doi/abs/10.5555/3104322.3104374.

Osman Güler. New proximal point algorithms for convex minimization. *SIAM Journal on Optimization*, 2(4):649–664, 1992. doi: 10.1137/0802032.

Paul Häusner, Ozan Öktem, and Jens Sjölund. Neural incomplete factorization: learning preconditioners for the conjugate gradient method. *Transactions on Machine Learning Research*, 2024. doi: 10.48550/arXiv.2305.16368.

Howard Heaton, Xiaohan Chen, Zhangyang Wang, and Wotao Yin. Safeguarded learned convex optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7848–7855, 2023. doi: 10.1609/aaai.v37i6.25950.

Danish Khan, Mohammed Qais, Irfan Sami, and Pengfei Hu. Half-quadratic criterion-based continuous-time adaptive control for robust lcl-filtered grid-tied inverter. *Electrical Engineering*, 107(5):5341–5363, 2025. doi: 10.1007/s00202-024-02664-6.

Olga Klopp, Karim Lounici, and Alexandre B Tsybakov. Robust matrix completion. *Probability Theory and Related Fields*, 169(1):523–564, 2017. doi: 10.1007/s00440-016-0736-y.

Ke Li and Jitendra Malik. Learning to optimize. *ArXiv e-print*, cs.LG(1606.01885), 2016. doi: 10.48550/arXiv.1606.01885.

Feng-Yi Liao, Lijun Ding, and Yang Zheng. Error bounds, PL condition, and quadratic growth for weakly convex functions, and linear convergences of proximal point methods. In *Proceedings of the 6th Annual Learning for Dynamics & Control Conference*, volume 242, pages 993–1005. PMLR, 2024. URL https://proceedings.mlr.press/v242/liao24a.html.

Jialin Liu, Xiaohan Chen, Zhangyang Wang, Wotao Yin, and HanQin Cai. Towards constituting mathematical structures for learning to optimize. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, volume 202, pages 21426–21449. PMLR, 2023. URL https://proceedings.mlr.press/v202/liu23e.html.

Andrea Martin, Ian R Manchester, and Luca Furieri. Learning to optimize with guarantees: a complete characterization of linearly convergent algorithms. *ArXiv e-print*, eess.SY(2508.00775), 2025. doi: 10.48550/arXiv.2508.00775.

David A McAllester. PAC-Bayesian model averaging. In *Proceedings of the 12th annual conference on Computational Learning Theory*, pages 164–170, 1999. doi: 10.1145/307400.307435.

Luke Metz, Niru Maheswaranathan, Jeremy Nixon, Daniel Freeman, and Jascha Sohl-Dickstein. Understanding and correcting pathologies in the training of learned optimizers. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97, pages 4556–4565. PMLR, 2019. URL https://proceedings.mlr.press/v97/metz19a.html.

Taylor R. Moen, Baiyu Chen, David R. Holmes, Xinhui Duan, Zhicong Yu, Lifeng Yu, Shuai Leng, Joel G. Fletcher, and Cynthia H. McCollough. Low-dose CT image and projection dataset. *Medical Physics*, 48:902–911, 2021. doi: 10.1002/mp.14594.

Vishal Monga, Yuelong Li, and Yonina C Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021. doi: 10.1109/msp.2020.3016905.

Subhadip Mukherjee, Andreas Hauptmann, Ozan Öktem, Marcelo Pereyra, and Carola-Bibiane Schönlieb. Learned reconstruction methods with convergence guarantees: A survey of concepts and applications. *IEEE Signal Processing Magazine*, 40(1):164–182, 2023. doi: 10.1109/MSP.2022.3207451.

Ion Necoara, Yurii Nesterov, and François Glineur. Linear convergence of first order methods for non-strongly convex optimization. *Mathematical Programming*, 175:69–107, 2019. doi: 10.1007/s10107-018-1232-1.

Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018. doi: 10.1007/978-3-319-91578-4.

Yuyuan Ouyang, Yunmei Chen, Guanghui Lan, and Eduardo Pasiliao Jr. An accelerated linearized alternating direction method of multipliers. *SIAM Journal on Imaging Sciences*, 8(1):644–681, 2015. doi: 10.1137/14095697X.

Emilio Parrado-Hernández, Amiran Ambroladze, John Shawe-Taylor, and Shiliang Sun. PAC-Bayes bounds with data dependent priors. *Journal of Machine Learning Research*, 13(112):3507–3531, 2012. URL http://jmlr.org/papers/v13/parrado12a.html.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*, pages 8026–8037, 2019. URL https://dl.acm.org/doi/10.5555/3454287.3455008.

Isabeau Prémont-Schwarz, Jaroslav Vítků, and Jan Feyereisl. A simple guard for learned optimizers. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162, pages 17910–17925. PMLR, 2022. URL https://proceedings.mlr.press/v162/premont-schwarz22a.html.

Marc Robini, Yuemin Zhu, Xudong Lv, and Wanyu Liu. Inexact half-quadratic optimization for image reconstruction. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3513–3517. IEEE, 2016. doi: 10.1109/ICIP.2016.7533013.

Jevgenija Rudzusika, Thomas Koehler, and Ozan Öktem. Deep learning–based dictionary learning and tomographic image reconstruction. *SIAM Journal on Imaging Sciences*, 15(4):1729–1764, 2022. doi: 10.1137/21M1445697.

Rajiv Sambharya and Bartolomeo Stellato. Data-driven performance guarantees for classical and learned optimizers. *Journal of Machine Learning Research*, 26(171):1–49, 2025. URL http://jmlr.org/papers/v26/24-0755.html.

Rajiv Sambharya and Bartolomeo Stellato. Learning algorithm hyperparameters for fast parametric convex optimization. *SIAM Journal on Mathematics of Data Science*, 8(3):649–676, 2026. doi: 10.1137/24M1712242.

Rajiv Sambharya, Georgina Hall, Brandon Amos, and Bartolomeo Stellato. Learning to warm-start fixed-point optimization algorithms. *Journal of Machine Learning Research*, 25(166):1–46, 2024. URL https://jmlr.org/papers/v25/23-1174.html.

Ella J. Schmidtobreick, Daniel Arnström, Paul Häusner, and Jens Sjölund. Warm-starting active-set solvers using graph neural networks. In *Proceedings of The 8th Annual Learning for Dynamics and Control Conference*, volume 331, pages 662–677. PMLR, 2026. URL https://proceedings.mlr.press/v331/schmidtobreick26a.html.

Jens Sjölund and Maria Bånkestad. Graph-based neural acceleration for nonnegative matrix factorization. *ArXiv e-print*, cs.LG(2202.00264), 2022. doi: 10.48550/arXiv.2202.00264.

Michael Sucker, Jalal Fadili, and Peter Ochs. Learning-to-optimize with PAC-Bayesian guarantees: Theoretical considerations and practical implementation. *Journal of Machine Learning Research*, 26(211):1–53, 2025. URL https://jmlr.org/papers/v26/24-0486.html.

Hong Ye Tan, Subhadip Mukherjee, Junqi Tang, and Carola-Bibiane Schönlieb. Data-driven mirror descent with input-convex neural networks. *SIAM Journal on Mathematics of Data Science*, 5(2):558–587, 2023. doi: 10.1137/22M1508613.

Min Tao and Xiaoming Yuan. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM Journal on Optimization*, 21(1):57–81, 2011. doi: 10.1137/100781894.

Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001. doi: 10.1023/A:1017501703105.

Wim Van Aarle, Willem Jan Palenstijn, Jan De Beenhouwer, Thomas Altantzis, Sara Bals, K Joost Batenburg, and Jan Sijbers. The ASTRA toolbox: A platform for advanced algorithm development in electron tomography. *Ultramicroscopy*, 157:35–47, 2015. doi: 10.1016/j.ultramic.2015.05.002.

Wim Van Aarle, Willem Jan Palenstijn, Jeroen Cant, Eline Janssens, Folkert Bleichrodt, Andrei Dabravolski, Jan De Beenhouwer, K Joost Batenburg, and Jan Sijbers. Fast and flexible x-ray tomography using the ASTRA toolbox. *Optics express*, 24(22):25129–25147, 2016. doi: 10.1364/OE.24.025129.

Xueqian Wang. *Study on Signal Detection and Recovery Methods with Joint Sparsity*. Springer Theses. Springer Verlag, 2024. doi: 10.1007/978-981-99-4117-9.

Jiaxin Xie, Anping Liao, and Xiaobo Yang. An inexact alternating direction method of multipliers with relative error criteria. *Optimization Letters*, 11(3):583–596, 2017. doi: 10.1007/s11590-016-1021-9.

Zhonglin Xie, Wotao Yin, and Zaiwen Wen. ODE-based learning to optimize. *Mathematical Programming*, pages 1–70, 2025. doi: 10.1007/s10107-025-02303-3.

Bingyu Xin, Timothy Phan, Leon Axel, and Dimitris Metaxas. Learned half-quadratic splitting network for MR image reconstruction. In *International Conference on Medical Imaging with Deep Learning (MIDL)*, volume 172, pages 1403–1412. PMLR, 2022. URL https://proceedings.mlr.press/v172/xin22a.html.

Zirui Zhou and Anthony Man-Cho So. A unified approach to error bounds for structured convex optimization problems. *Mathematical Programming*, 165(2):689–728, 2017. doi: 10.1007/s10107-016-1100-9.

A Preliminaries

A.1 Basic notions and facts from convex analysis

We review some key results from optimization and convex analysis used in the proofs. The main reference for this part is [Nesterov, 2018] but some results can also be found in [Bauschke and Combettes, 2017]. In the following, let $\mathcal{X}$ be a Hilbert space with inner product $\langle \cdot, \cdot \rangle$ and induced norm $\| \cdot \|$. Throughout this paper we typically use $\mathcal{X} = \mathbb{R}^n$.

We next list some basic notions and facts from convex analysis that are used throughout the paper. When stating these, $f \colon \mathcal{X} \to \mathbb{R}$ will be a functional that is, unless stated otherwise, assumed to be differentiable. Finally, $x, y \in \mathcal{X}$ are two arbitrary points in the domain of f.

- Assume f is L-smooth, i.e., its gradient ∇f is Lipschitz continuous with constant L. Then we have the following inequality (*descent lemma*):

$$f(y) \leq f(x) + \big\langle \nabla f(x), y - x \big\rangle + \frac{L}{2}\|y - x\|^2. \tag{17}$$

- A differentiable function f is convex if and only if

$$f(x) \geq f(y) + \big\langle \nabla f(y), x - y \big\rangle. \tag{18}$$

- Furthermore, f is strongly convex with modulus $\mu > 0$ if and only if

$$f(x) \geq f(y) + \langle \nabla f(y), x - y \rangle + \frac{\mu}{2}\|y - x\|^2. \tag{19}$$

  Equivalently, f is μ-strongly convex iff $x \mapsto f(x) - \frac{\mu}{2}\|x\|^2$ is convex.

- If f is non-smooth but convex, then the set of subgradients is defined as follows; g is a sub-gradient of a convex function, denoted by $g \in \partial f(y)$, if

$$f(x) \geq f(y) + \langle g, x - y \rangle \quad \text{for any} \quad x. \tag{20}$$

  In other words, the local approximation of f at the point y through g is a global lower bound of f. If f is Fréchet differentiable, then $g = \nabla f(y)$.

- A proper lower semi-continuous function is closed.

- If f and g are closed convex functions then $\partial(f + g) = \partial f + \partial g$.

- If $W \colon \mathcal{X} \to \mathcal{X}$ is a bounded, self-adjoint, positive definite linear operator, then

$$\|x\|_W^2 := \langle Wx, x \rangle = \big\langle W^{\frac{1}{2}}x, W^{\frac{1}{2}}x \big\rangle \tag{21}$$

  defines a norm on $\mathcal{X}$. If W is only positive semi-definite, then the above is a semi-norm as positivity is not guaranteed since the elements can be mapped to the nullspace of the operator W that parametrizes the norm.

- The arithmetic mean - geometric mean (AM-GM) inequality (also referred to as Young's inequality) states that for any $a, b \in \mathbb{R}$ and any $\lambda > 0$:

$$ab \leq \frac{\lambda}{2}a^2 + \frac{1}{2\lambda}b^2. \tag{22}$$

- A simple but useful fact about norms is the following quadratic expansion of squared norms:

$$\frac{1}{2}\|a + b\|^2 - \frac{1}{2}\|a\|^2 = \frac{1}{2}\|b\|^2 + \langle a, b \rangle. \tag{23}$$

  Further, the parallelogram law which is satisfied by norms that are induced by an inner product:

$$\|x + y\|^2 + \|x - y\|^2 = 2\|x\|^2 + 2\|y\|^2. \tag{24}$$

A.2 Auxiliary lemmas for the alternating scheme

In this section, we provide some background on the convergence of alternating schemes and provide some key technical lemmas.

The key step in proving the convergence of the alternating scheme relies on a generalized notion of strong convexity that we state below, which allows us to couple the updates across the two variable blocks through the quadratic term in Eq. (1).

Lemma 2 (Generalized strong convexity)**.** Assume $x \mapsto f(x) - \frac{1}{2}\|Ax\|^2$ for an arbitrary bounded linear operator A and norm $\|\cdot\|$ on the Hilbert space of the co-domain is a convex function and $g \in \partial f(y)$. Then, for any x

$$f(y) \leq f(x) + \langle g, y - x\rangle - \frac{1}{2}\|A(y-x)\|^2. \tag{25}$$

If y is additionally a minimizer of f, then

$$f(x) \geq f(y) + \frac{1}{2}\|A(y-x)\|^2. \tag{26}$$

The proof mimics the strongly convex case using instead the semi-norm induced by the linear operator A to weight the norm.

Proof. Assume that $g \in \partial f(y)$ is a subgradient of f, then by applying subdifferential calculus $g - A^T Ay \in \partial(f(y) - \frac{1}{2}\|Ay\|^2)$ and by using the subgradient property Eq. (20), we obtain

$$\begin{aligned} f(x) - \frac{1}{2}\|Ax\|^2 &\geq f(y) - \frac{1}{2}\|Ay\|^2 + \langle g - A^T Ay, x - y\rangle \\ &= f(y) - \frac{1}{2}\|Ay\|^2 + \langle g, x - y\rangle - \langle Ay, Ax\rangle + \|Ay\|^2 \\ &= f(y) + \frac{1}{2}\|Ay\|^2 - \langle Ay, Ax\rangle + \langle g, x - y\rangle. \end{aligned} \tag{27}$$

By rewriting and exploiting the linearity of the inner product, we obtain the following inequality

$$\begin{aligned} f(x) &\geq f(y) + \frac{1}{2}\|Ax\|^2 + \frac{1}{2}\|Ay\|^2 - \langle Ay, Ax\rangle + \langle g, x - y\rangle \\ &= f(y) + \frac{1}{2}\|A(x-y)\|^2 + \langle g, x - y\rangle. \end{aligned} \tag{28}$$

If y is the optimum, then $g = 0 \in \partial f(y)$. If f is differentiable $\partial f(y) = \{\nabla f(y)\}$. ■

We can immediately observe that the objective function Eq. (1) satisfies the assumption of Lemma 2 in the following remark.

Remark 3. Let F be a function of the form given in Eq. (1). Then the functions $x \mapsto F(x,z) - \frac{1}{2}\|Ax\|^2$ and $z \mapsto F(x,z) - \frac{1}{2}\|Bz\|^2$ are convex.

Proof sketch. This can be easily verified as the following is convex in x for a fixed z:

$$\begin{aligned} F(x,z) - \frac{1}{2}\|Ax\|^2 &= f(x) + \frac{1}{2}\|Ax + Bz\|^2 - \frac{1}{2}\|Ax\|^2 + g(z) \\ &= f(x) + \langle Ax, Bz\rangle + \frac{1}{2}\|Bz\|^2 + g(z), \end{aligned} \tag{29}$$

since f is assumed to be convex and the inner-product term is linear in x. The convexity in z follows by the same argument. ■

In other words, the subproblems are generalized strongly convex. Now, we have all ingredients to prove the following lemma that allows us to derive the convergence of the alternating scheme where the subproblems are solved exactly:

Lemma 3. Let F be a convex objective that is block separable with quadratic coupling term as in Eq. (1). Then, it holds for the iterates obtained from the alternating scheme Eq. (2) and any $(x, z) \in \mathcal{X} \times \mathcal{Z}$ that

$$F(x^{k+1}, z^{k+1}) + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(x, z) + \frac{1}{2}\|B(z - z^k)\|^2 \tag{30}$$

Proof. Using the observation from Remark 3 and applying Lemma 2 yields the following two inequalities for the updates of the alternating scheme Eq. (2). Here, we are using the fact that the next iterates themselves are computed as the solution to an optimization problem:

$$F(x^{k+1}, z^k) + \frac{1}{2}\|A(x - x^{k+1})\|^2 \leq F(x, z^k), \tag{31a}$$

$$F(x^{k+1}, z^{k+1}) + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(x^{k+1}, z), \tag{31b}$$

which hold for any $x \in \mathcal{X}$ and $z \in \mathcal{Z}$. Summing these inequalities and using an explicit form of the objective function Eq. (1) results in

$$F(x^{k+1}, z^{k+1}) + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(x, z) + \frac{1}{2}\|B(z - z^k)\|^2 + \mathcal{R}^{k+1}(x, z) \tag{32}$$

where the residual can be simplified and bounded in the following way

$$\begin{aligned}\mathcal{R}^{k+1}(x, z) = &\frac{1}{2}\|Ax + Bz^k\|^2 + \frac{1}{2}\|Ax^{k+1} + Bz\|^2 - \frac{1}{2}\|Ax^{k+1} + Bz^k\|^2 \\ &- \frac{1}{2}\|Ax + Bz\|^2 - \frac{1}{2}\|A(x - x^{k+1})\|^2 - \frac{1}{2}\|B(z - z^k)\|^2 \\ = &-\frac{1}{2}\|Ax^{k+1} + Bz^k - Ax - Bz\|^2 \leq 0.\end{aligned} \tag{33}$$

Plugging this back into Eq. (32) finalizes the proof. ■

In Section 3 we extend the alternating scheme to inexact solves, making use of Lemma 2 and Remark 3. In particular, we extend the inequality in Eq. (30) to account for the inexactness error introduced by solving each problem inexactly.

B Analysis of the inexact scheme

This section contains the proofs of the results for the inexact scheme stated in Section 3. We provide the proofs for Lemma 1, Theorem 1, and Corollary 1 in this appendix.

B.1 Proof of Lemma 1

The proof closely follows the proof of the exact case in Lemma 3, adds error terms from the inexact solvers, and bridges the exact z-minimizer on the left-hand side to the inexact iterate $\tilde{z}^{k+1}$. The key difference of Eq. (5) compared to the exact case is the additional combined error term and the inflated coefficients on the norm terms.

Proof. The proof proceeds in three steps: (step 1) we derive an inequality for each subproblem, containing an individual error term; (step 2) we sum the two inequalities, exploit a non-positive residual, and bound the resulting inner product that remains from the error terms; (step 3) we replace the exact minimizer z^{k+1} on the left-hand side with the inexact iterate $\tilde{z}^{k+1}$.

Step 1: Base inequalities. Let $z^{k+1} := \arg\min_{z \in \mathcal{Z}} F(\tilde{x}^{k+1}, z)$ be the exact minimizer of the z-subproblem given $\tilde{x}^{k+1}$. By generalized strong convexity with respect to z (see Remark 3) and applying Lemma 2 we obtain for any $z \in \mathcal{Z}$:

$$F(\tilde{x}^{k+1}, z^{k+1}) + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(\tilde{x}^{k+1}, z) \tag{34}$$

Adding ζ^{k+1} to both sides and using its definition to simplify the left-hand side gives

$$\underbrace{F(\tilde{x}^{k+1}, z^{k+1})}_{=:\tilde{F}^{k+1}} + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(\tilde{x}^{k+1}, z) + \zeta^{k+1}. \tag{35}$$

For the x-update using the same argument as in the previous case, in the exact case we have

$$F(x^{k+1}, \tilde{z}^k) + \frac{1}{2}\|A(x - x^{k+1})\|^2 \leq F(x, \tilde{z}^k) \tag{36}$$

from the generalized strong convexity. Replacing the exact minimizer x^{k+1} with the approximate iterate $\tilde{x}^{k+1}$ introduces an error $\mathcal{E}^{k+1}(x)$ so that for any $x \in \mathcal{X}$

$$F(\tilde{x}^{k+1}, \tilde{z}^k) + \frac{1}{2}\|A(x - \tilde{x}^{k+1})\|^2 \leq F(x, \tilde{z}^k) + \mathcal{E}^{k+1}(x), \tag{37}$$

where the error is defined as

$$\mathcal{E}^{k+1}(x) := \underbrace{F(\tilde{x}^{k+1}, \tilde{z}^k) - F(x^{k+1}, \tilde{z}^k)}_{=\xi^{k+1}} + \frac{1}{2}\|A(x - \tilde{x}^{k+1})\|^2 - \frac{1}{2}\|A(x - x^{k+1})\|^2. \tag{38}$$

Step 2: Sum, simplify, and bound the inner product. Summing Eq. (35) and Eq. (37) and using the explicit form of the objective Eq. (1) gives a non-positive residual term and the error term on the right-hand side:

$$\tilde{F}^{k+1} + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(x, z) + \frac{1}{2}\|B(z - \tilde{z}^k)\|^2 + \tilde{\mathcal{R}}^{k+1}(x, z) + \mathcal{E}^{k+1}(x) + \zeta^{k+1}, \tag{39}$$

where the residual term simplifies using the explicit form of the objective Eq. (1):

$$\begin{aligned}
\tilde{\mathcal{R}}^{k+1}(x, z) &= \frac{1}{2}\|Ax + B\tilde{z}^k\|^2 + \frac{1}{2}\|A\tilde{x}^{k+1} + Bz\|^2 - \frac{1}{2}\|A\tilde{x}^{k+1} + B\tilde{z}^k\|^2 \\
&\quad - \frac{1}{2}\|Ax + Bz\|^2 - \frac{1}{2}\|A(x - \tilde{x}^{k+1})\|^2 - \frac{1}{2}\|B(z - \tilde{z}^k)\|^2 \\
&= -\frac{1}{2}\|A(\tilde{x}^{k+1} - x) + B(\tilde{z}^k - z)\|^2 \\
&= -\frac{1}{2}\|A(\tilde{x}^{k+1} - x)\|^2 + \langle A(x - \tilde{x}^{k+1}),\, B(\tilde{z}^k - z)\rangle - \frac{1}{2}\|B(z - \tilde{z}^k)\|^2.
\end{aligned} \tag{40}$$

We now use $\tilde{\mathcal{R}}^{k+1}(x, z)$ to simplify the right-hand side of Eq. (39). Substituting the definition Eq. (38) of $\mathcal{E}^{k+1}(x)$, two cancellations occur:

- first, the terms $\pm\frac{1}{2}\|A(x - \tilde{x}^{k+1})\|^2$ from $\tilde{\mathcal{R}}^{k+1}$ and $\mathcal{E}^{k+1}$ cancel
- second, the $-\frac{1}{2}\|B(z - \tilde{z}^k)\|^2$ from $\tilde{\mathcal{R}}^{k+1}$ cancels with the explicit $+\frac{1}{2}\|B(z - \tilde{z}^k)\|^2$ on the right-hand side of Eq. (39).

This gives a simplified bound where the errors of the two subproblems can be added up to each other:

$$\tilde{F}^{k+1} + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(x, z) + \langle A(x - \tilde{x}^{k+1}),\, B(\tilde{z}^k - z)\rangle - \frac{1}{2}\|A(x - x^{k+1})\|^2 + \Delta^{k+1} \tag{41}$$

We bound the inner product by decomposing $x - \tilde{x}^{k+1} = (x - x^{k+1}) + (x^{k+1} - \tilde{x}^{k+1})$. Using the linearity of the inner product we proceed to bound each part individually as follows.

For the first part, the identity

$$\langle a, b\rangle - \frac{1}{2}\|a\|^2 = \frac{1}{2}\|b\|^2 - \frac{1}{2}\|a - b\|^2 \tag{42}$$

with $a = A(x - x^{k+1})$ and $b = B(\tilde{z}^k - z)$ gives

$$\langle A(x - x^{k+1}),\, B(\tilde{z}^k - z)\rangle - \frac{1}{2}\|A(x - x^{k+1})\|^2 \leq \frac{1}{2}\|B(z - \tilde{z}^k)\|^2. \tag{43}$$

For the second part, Young's inequality with parameter $\alpha > 0$ and the bound

$$\frac{1}{2}\|A(x^{k+1} - \tilde{x}^{k+1})\|^2 \leq \xi^{k+1} \tag{44}$$

which follows from Eq. (26) in Lemma 2 applied to the minimizer x^{k+1} of $x \mapsto F(x, \tilde{z}^k)$, using Remark 3 give

$$\begin{aligned}\langle A(x^{k+1} - \tilde{x}^{k+1}),\, B(\tilde{z}^k - z)\rangle &\leq \frac{\alpha}{2}\|B(z - \tilde{z}^k)\|^2 + \frac{1}{2\alpha}\|A(x^{k+1} - \tilde{x}^{k+1})\|^2 \\ &\leq \frac{\alpha}{2}\|B(z - \tilde{z}^k)\|^2 + \frac{1}{\alpha}\xi^{k+1}.\end{aligned} \tag{45}$$

Combining the steps Eqs. (41), (43) and (45) yields

$$F(\tilde{x}^{k+1}, \tilde{z}^{k+1}) + \frac{1}{2}\|B(z - z^{k+1})\|^2 \leq F(x, z) + \frac{1+\alpha}{2}\|B(z - \tilde{z}^k)\|^2 + \frac{1+\alpha}{\alpha}\xi^{k+1} + \zeta^{k+1}. \tag{46}$$

Step 3: Bridge exact to inexact z-iterate. It remains to replace the exact z^{k+1} on the left-hand side with the inexact iterate $\tilde{z}^{k+1}$. We decompose $B(z - z^{k+1}) = B(z - \tilde{z}^{k+1}) + B(\tilde{z}^{k+1} - z^{k+1})$.

Using Young's inequality on the quadratic expansion gives

$$\|u + v\|^2 \geq \frac{1}{1+\varepsilon}\|u\|^2 - \frac{1}{\varepsilon}\|v\|^2 \tag{47}$$

for any $\varepsilon > 0$. Applying this result with $u = B(z - \tilde{z}^{k+1})$ and $v = B(\tilde{z}^{k+1} - z^{k+1})$ yields

$$\frac{1}{2}\|B(z - z^{k+1})\|^2 \geq \frac{1}{2(1+\varepsilon)}\|B(z - \tilde{z}^{k+1})\|^2 - \frac{1}{2\varepsilon}\|B(\tilde{z}^{k+1} - z^{k+1})\|^2. \tag{48}$$

Since z^{k+1} minimizes $z \mapsto F(\tilde{x}^{k+1}, z)$, generalized strong convexity (Lemma 2 and Remark 3) gives

$$\frac{1}{2}\|B(\tilde{z}^{k+1} - z^{k+1})\|^2 \leq \zeta^{k+1}. \tag{49}$$

Substituting into Eq. (46) and combining the ζ^{k+1} terms on the right-hand side gives:

$$\tilde{F}^{k+1} + \frac{1}{2(1+\varepsilon)}\|B(z - \tilde{z}^{k+1})\|^2 \leq F(x, z) + \frac{1+\alpha}{2}\|B(z - \tilde{z}^k)\|^2 + \frac{1+\alpha}{\alpha}\xi^{k+1} + \left(1 + \frac{1}{\varepsilon}\right)\zeta^{k+1}. \tag{50}$$

In order to combine the bounds, we are setting $\varepsilon = \alpha$ which gives $1/(1+\varepsilon) = 1/(1+\alpha)$ and $1 + 1/\varepsilon = (1+\alpha)/\alpha$, so the error terms combine into

$$\frac{1+\alpha}{\alpha}(\xi^{k+1} + \zeta^{k+1}) = \frac{1+\alpha}{\alpha}\Delta^{k+1}. \tag{51}$$

This concludes the proof. ■

B.2 Proof of Theorem 1

The proof of the linear convergence result relies on the inexact descent inequality shown in Lemma 1 and the quadratic growth assumption formulated in Assumption 1. The final key step is determining the bounds for α such that the rate $\rho < 1$ to obtain convergence of the scheme.

Proof. Since Lemma 1 holds for any $(x, z) \in \mathcal{X} \times \mathcal{Z}$, we evaluate Eq. (5) at an arbitrary $(x^*, z^*) \in Z^*$, for which $F(x^*, z^*) = F^*$:

$$F(\tilde{x}^{k+1}, \tilde{z}^{k+1}) + \frac{1}{2(1+\alpha)}\|B(z^* - \tilde{z}^{k+1})\|^2 \leq F^* + \frac{1+\alpha}{2}\|B(z^* - \tilde{z}^k)\|^2 + \frac{1+\alpha}{\alpha}\,\Delta^{k+1}. \tag{52}$$

This inequality holds for every $(x^*, z^*) \in Z^*$. On the left-hand side we lower-bound by the infimum,

$$\|B(z^* - \tilde{z}^{k+1})\|^2 \geq \inf_{(x', z') \in Z^*} \|B(z' - \tilde{z}^{k+1})\|^2,$$

which is independent of the chosen (x^*, z^*). Taking the infimum over $(x^*, z^*) \in Z^*$ on the right-hand side then yields

$$\tilde{F}^{k+1} + \frac{1}{2(1+\alpha)} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^{k+1})\|^2 \leq F^* + \frac{1+\alpha}{2} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^k)\|^2 + \frac{1+\alpha}{\alpha} \Delta^{k+1}. \tag{53}$$

Defining the Lyapunov energy

$$\Phi_k := \tilde{F}^k - F^* + \frac{1}{2(1+\alpha)} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^k)\|^2, \tag{54}$$

we now bound the B-norm term on the right-hand side of the inequality using Assumption 1. Using Assumption 1 gives the bound on the function value gap

$$\tilde{F}^k - F^* \geq \frac{\mu}{2} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^k)\|^2, \tag{55}$$

so that by applying the inequality and combining the two B-norm terms we obtain

$$\Phi_k \geq \frac{\mu(1+\alpha)+1}{2(1+\alpha)} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^k)\|^2, \tag{56}$$

and in particular $\inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^k)\|^2 \leq \frac{2(1+\alpha)}{\mu(1+\alpha)+1} \Phi_k$. Substituting this into Eq. (53) where we bring F^* to the left-hand side gives the one-step contraction:

$$\Phi_{k+1} \leq \frac{(1+\alpha)^2}{1+\mu(1+\alpha)} \Phi_k + \frac{1+\alpha}{\alpha} \Delta^{k+1}. \tag{57}$$

Now, we recursively unroll these iterations to obtain for $\rho = (1+\alpha)^2/(1+\mu(1+\alpha)) < 1$:

$$\Phi_K \leq \rho^K \Phi_0 + \frac{1+\alpha}{\alpha} \sum_{k=1}^{K} \rho^{K-k} \Delta^k, \tag{58}$$

where $\Phi_0 = \tilde{F}^0 - F^* + \frac{1}{2(1+\alpha)} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^0)\|^2$.

It remains to show that $\rho < 1$ for the choice of α in Eq. (7). This is equivalent to the condition

$$(1+\alpha)^2 < 1 + \mu(1+\alpha) \tag{59}$$

for the specific contraction rate ρ. Now setting $t = 1 + \alpha$, we can see this is a quadratic in t. We find the roots of this quadratic form which yields that the condition is satisfied for

$$\alpha < \frac{\mu - 2 + \sqrt{\mu^2 + 4}}{2} \tag{60}$$

since the quadratic $t^2 - \mu t - 1$ is negative between its two roots and the lower root is negative, this is satisfied for any α in the interval provided in Eq. (7). This concludes the proof of the bound. ■

B.3 Proof of Corollary 1

Since the first term in Eq. (9) decays geometrically and the weights ρ^{K-k} discount early errors exponentially, the behavior of the bound for large K is dictated by how fast the late errors Δ_k decay, which leads to the three regimes listed in Corollary 1.

Proof. In all three cases, the first term of the bound in Theorem 1 given by

$$\rho^K \left(F(\tilde{x}^0, \tilde{z}^0) - F^* + \frac{1}{2(1+\alpha)} \inf_{(x',z') \in Z^*} \|B(z' - \tilde{z}^0)\|^2 \right) \tag{61}$$

decays geometrically at rate ρ, so it suffices to bound the error sum $S_K = \frac{1+\alpha}{\alpha} \sum_{k=1}^{K} \rho^{K-k} \Delta^k$ for the different cases.

Case i. In the case that $q < \rho$, we substitute $\Delta^k \leq C q^k$ and factor out ρ^K to obtain:

$$S_K \leq \frac{C(1+\alpha)}{\alpha} \rho^K \sum_{k=1}^{K} \left(\frac{q}{\rho} \right)^k. \tag{62}$$

The ratio $q/\rho < 1$ and the infinite sum for $K \to \infty$ converges as a geometric sum with limit $(q/\rho)/(1-(q/\rho))$ giving $S_K = \mathcal{O}(\rho^K)$.

If instead $q > \rho$, we mirror the argument from above and factor out q^K:

$$S_K \leq \frac{C(1+\alpha)}{\alpha} q^K \sum_{k=1}^{K} \rho^{K-k} q^{k-K} \leq \frac{C(1+\alpha)}{\alpha} q^K \sum_{j=0}^{K-1} \left(\frac{\rho}{q}\right)^j \tag{63}$$

where $\rho/q < 1$ and we set $j = K - k$ to obtain a converging sum with the same argument as above yielding a $\mathcal{O}(q^K)$ convergence.

If $q = \rho$, the sum simplifies to $S_K = \frac{C(1+\alpha)}{\alpha} K\rho^K$. The linear factor K reflects a resonance between the contraction and the error decay: the contraction damps past errors at exactly the rate at which new ones are injected, so no part of the history dominates and all K contributions accumulate equally. However, the resulting rate $\mathcal{O}(K\rho^K)$ is still dominated by $(\rho')^K$ for any $\rho' \in (\rho, 1)$, so the conclusion below holds in this asymptotic sense at the boundary.

Thus, in all cases $S_K = \mathcal{O}(\max(\rho, q)^K)$ yielding in both cases a linear convergence of the function gap $\tilde{F}^K - F^* \leq \mathcal{O}(\max(\rho, q)^K)$.

Case ii.a. (Convergence without a rate)
Suppose only that $\Delta^k \to 0$. We now show that $\sum_{k=1}^{K} \rho^{K-k}\Delta^k \to 0$ as $K \to \infty$.

Choosing a fixed $\epsilon > 0$. Since $\Delta^k \to 0$, there exists $\bar{k} \in \mathbb{N}$ such that $\Delta^k < \epsilon$ for all $k > \bar{k}$. Splitting the sum at $\bar{k}$:

$$\sum_{k=1}^{K} \rho^{K-k}\Delta^k = \sum_{k=1}^{\bar{k}} \rho^{K-k}\Delta^k + \sum_{k=\bar{k}+1}^{K} \rho^{K-k}\Delta^k. \tag{64}$$

For the first sum, $\rho^{K-k} \leq \rho^{K-\bar{k}}$ for all $k \leq \bar{k}$, so

$$\sum_{k=1}^{\bar{k}} \rho^{K-k}\Delta^k \leq \rho^{K-\bar{k}} \sum_{k=1}^{\bar{k}} \Delta^k, \tag{65}$$

where $\sum_{k=1}^{\bar{k}} \Delta^k$ is a fixed finite constant and $\rho^{K-\bar{k}} \to 0$ as $K \to \infty$.

For the second sum, bounding $\Delta^k < \epsilon$ and extending the geometric series gives

$$\sum_{k=\bar{k}+1}^{K} \rho^{K-k}\Delta^k < \epsilon \sum_{k=\bar{k}+1}^{K} \rho^{K-k} \leq \frac{\epsilon}{1-\rho}. \tag{66}$$

Hence $\limsup_{K\to\infty} \sum_{k=1}^{K} \rho^{K-k}\Delta^k \leq \epsilon/(1-\rho)$, and since ϵ was arbitrary, the sum vanishes.

Case ii.b. (Polynomial decay)
We substitute the polynomial decay assumption into the bound and split the resulting error term into two separate sums similar to the previous step at the threshold $m = \lfloor K/2 \rfloor$.

For $k \leq m$, we have $K - k \geq K/2$, so we can bound the first sum as

$$\Sigma_1 = \sum_{k=1}^{m} \rho^{K-k} \frac{C}{k^q} \leq C\rho^{K/2} \sum_{k=1}^{m} \frac{1}{k^q}. \tag{67}$$

The harmonic sum is bounded by $\mathcal{O}(K^{\max(1-q,\,0)})$ for $q \neq 1$ and $\mathcal{O}(\log K)$ for $q = 1$. In either case, the exponential factor $\rho^{K/2}$ dominates the convergence as $K \to \infty$.

For $k > m$, we have $k \geq K/2$ and thus we have $1/k^q \leq 2^q/K^q$, giving the following inequality by bounding the geometric sum:

$$\Sigma_2 = \sum_{k=m+1}^{K} \rho^{K-k} \frac{C}{k^q} \leq C \frac{2^q}{K^q} \sum_{j=0}^{K-m-1} \rho^j \leq C \frac{2^q}{(1-\rho)K^q}. \tag{68}$$

Combining the two previous terms and plugging them back into the expression for S_K yields the following polynomial decay since the convergence is determined by the second part of the sum:

$$S_K = \frac{1+\alpha}{\alpha} \sum_{k=1}^{K} \rho^{K-k} \Delta^k \leq \frac{C(1+\alpha)}{\alpha} \left(\frac{2^q}{(1-\rho)\, K^q} + \rho^{K/2} \sum_{k=1}^{m} \frac{1}{k^q} \right) = \mathcal{O}(1/K^q), \tag{69}$$

Adding the geometrically decaying term to the exponentially decaying Φ_0 from the result in Theorem 1 preserves the rate giving a sublinear convergence with a rate determined by the exponent q.

Case iii. To start, we split the error sum at the threshold $\bar{k}$ from the assumption:

$$\sum_{k=1}^{K} \rho^{K-k} \Delta^k = \sum_{k=1}^{\bar{k}} \rho^{K-k} \Delta^k + \sum_{k=\bar{k}+1}^{K} \rho^{K-k} \Delta^k. \tag{70}$$

For the first sum, $\rho^{K-k} \leq \rho^{K-\bar{k}}$ for all $k \leq \bar{k}$, so

$$\sum_{k=1}^{\bar{k}} \rho^{K-k} \Delta^k \leq \rho^{K-\bar{k}} \sum_{k=1}^{\bar{k}} \Delta^k, \tag{71}$$

where $\sum_{k=1}^{\bar{k}} \Delta^k$ is a finite constant and $\rho^{K-\bar{k}} \to 0$ as $K \to \infty$. For the second sum, the tail bound $\Delta^k \leq \Delta^{\text{tail}}$ and the geometric series give

$$\sum_{k=\bar{k}+1}^{K} \rho^{K-k} \Delta^k \leq \Delta^{\text{tail}} \sum_{k=\bar{k}+1}^{K} \rho^{K-k} \leq \frac{\Delta^{\text{tail}}}{1-\rho}. \tag{72}$$

Combining the two bounds yields $\sum_{k=1}^{K} \rho^{K-k} \Delta^k \leq \rho^{K-\bar{k}} \sum_{k=1}^{\bar{k}} \Delta^k + \Delta^{\text{tail}}/(1-\rho)$ for all $K > \bar{k}$, and the first term vanishes as $K \to \infty$. Taking $\limsup$ on both sides gives

$$\limsup_{K \to \infty} S_K \leq \frac{1+\alpha}{\alpha} \cdot \frac{\Delta^{\text{tail}}}{1-\rho}. \tag{73}$$

This finalizes the proof. ■

C Analysis of the learned scheme

This section contains the proof of Theorem 2 and provides the formal statement for the weight reuse that is mentioned in the main text (Appendix C.2). Then, we describe how we evaluate the bound numerically (Appendix C.3).

C.1 Proof of Theorem 2

Proof. We define a joint prior $\mathbb{P}$ over the parameters $\Theta^{1:K}$ as the Gaussian random walk

$$\Theta^k \mid \Theta^{k-1} \sim \mathcal{N}(\Theta^{k-1}, \varsigma^2 I) \quad \text{for } k = 1, \ldots, K, \tag{74}$$

where Θ^0 is a fixed initialization chosen independently of the training data and $\varsigma^2 > 0$ is the prior variance. Any data-independent prior yields a valid PAC-Bayes bound. Here, we adopt the random walk above because its KL divergence to a suitably chosen posterior reduces to the parameter drift across iterations, giving the bound a direct interpretation in terms of warm-starting.

We choose a corresponding posterior $\mathbb{Q}$ that follows the same random-walk structure of $\mathbb{P}$ but keeps track of the trained trajectory $\hat{\Theta}^{1:K}$ rather than staying around the initialization Θ^0. Under $\mathbb{Q}$, each step adds the trained parameter change to the previous parameter and the same Gaussian noise as in the prior:

$$\Theta^k = \Theta^{k-1} + (\hat{\Theta}^k - \hat{\Theta}^{k-1}) + \epsilon^k, \quad \epsilon^k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \varsigma^2 I), \quad k = 1, \ldots, K, \tag{75}$$

with $\hat{\Theta}^k$ the parameters returned by greedy training and $\hat{\Theta}^0 := \Theta^0$ the shared initialization. The KL divergence between posterior and prior then reduces to the total parameter drift across the trained iterations:

$$\mathrm{KL}(\mathbb{Q}\|\mathbb{P}) = \frac{1}{2\varsigma^2} \sum_{k=1}^{K} \|\hat{\Theta}^k - \hat{\Theta}^{k-1}\|^2. \tag{76}$$

We view the discounted error sum as a single loss function we aim to bound with the PAC-Bayes argument in the parameters $(\Theta^{1:K}, y)$,

$$L_y(\Theta^{1:K}) := \sum_{k=1}^{K} \rho^{K-k}\, \Delta_y^k(\Theta^{1:k}), \tag{77}$$

with empirical counterpart on the training samples $\mathcal{D} = \{y_i\}_{i=1}^N$

$$\hat{L}(\Theta^{1:K}) := \frac{1}{N} \sum_{i=1}^{N} L_{y_i}(\Theta^{1:K}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \rho^{K-k}\, \Delta_{y_i}^k(\Theta^{1:k}) = \sum_{k=1}^{K} \rho^{K-k}\, \hat{\Delta}^k(\Theta^{1:k}). \tag{78}$$

where the population risk is replaced by its empirical counterpart in Eq. (12).

The bounded-error assumption must hold for every parameter value in the support of $\mathbb{Q}$, not only at $\hat{\Theta}^{1:K}$, since the bound averages over $\Theta^{1:K} \sim \mathbb{Q}$. In our learned schemes, the output projection $\Pi_{[l,u]}$ is part of each network, so the iterates remain in a fixed box for every parameter value, including those drawn from $\mathbb{Q}$. Together with the bounded support of y, the iterates and the data lie in a compact set on which the continuous objective F_y and its subproblem minima are bounded, which bounds each Δ_y^k uniformly. Since $0 \le \Delta_y^k \le M$ for every parameter trajectory and $\sum_{k=1}^{K} \rho^{K-k} \le 1/(1-\rho)$, the combined loss across the trajectory of iterations for a single instance L_y takes values in $[0, M/(1-\rho)]$.

Applying the PAC-Bayes inequality for bounded losses with a finite grid [Alquier, 2024, Theorem 2.4] to $L_{y_1}, \ldots, L_{y_N}$ yields, with probability at least $1-\beta$ over the sample, simultaneously for all $\lambda \in \Lambda$,

$$\mathbb{E}_{\mathbb{Q}}[\mathbb{E}_y[L_y]] \;\le\; \mathbb{E}_{\mathbb{Q}}[\hat{L}] + \frac{\lambda M^2}{8N(1-\rho)^2} + \frac{\mathrm{KL}(\mathbb{Q}\|\mathbb{P}) + C}{\lambda}, \tag{79}$$

where $C = \log((2J+1)/\beta)$. The parameter λ must be chosen independently of the sample, so we use the grid $\Lambda := \{2^j : j = -J, \ldots, J\}$ with $|\Lambda| = 2J+1$ points, fixed before training.

The right-hand side in Eq. (79) has the form $a\lambda + b/\lambda$ with $a := M^2/(8N(1-\rho)^2)$ and $b := \mathrm{KL}(\mathbb{Q}\|\mathbb{P}) + C$. Its unconstrained minimizer $\lambda^\star = \sqrt{b/a} = \frac{2(1-\rho)}{M}\sqrt{2N(\mathrm{KL}(\mathbb{Q}\|\mathbb{P}) + C)}$ depends on the data through the KL divergence and therefore cannot be used directly. We call J large enough if $\lambda^\star \in [2^{-J}, 2^J]$, which can be checked after training. In that case, the grid contains a point $\lambda = t\lambda^\star$ with $t \in [2^{-1/2}, 2^{1/2}]$, at which $a\lambda + b/\lambda = \sqrt{ab}\,(t + 1/t) \le \frac{3}{\sqrt{2}}\sqrt{ab}$. Inserting a and b gives

$$\mathbb{E}_{\mathbb{Q}}[\mathbb{E}_y[L_y]] \;\le\; \mathbb{E}_{\mathbb{Q}}[\hat{L}] + \frac{3M}{4(1-\rho)} \sqrt{\frac{\mathrm{KL}(\mathbb{Q}\|\mathbb{P}) + C}{N}}. \tag{80}$$

Expanding L_y and $\hat{L}$ and exchanging the finite sum with $\mathbb{E}_{\mathbb{Q}}$ and $\mathbb{E}_y$ due to the linearity of the expectation operator yields

$$\sum_{k=1}^{K} \rho^{K-k}\, \mathbb{E}_{\mathbb{Q}}\big[\mathbb{E}_y[\Delta_y^k]\big] \;\le\; \sum_{k=1}^{K} \rho^{K-k}\, \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k] + \frac{3M}{4(1-\rho)} \sqrt{\frac{\mathrm{KL}(\mathbb{Q}\|\mathbb{P}) + C}{N}}. \tag{81}$$

Finally, Eq. (11) holds for every fixed parameter trajectory. All terms are bounded, so taking the expectation over $\Theta^{1:K} \sim \mathbb{Q}$ and bounding the error sum by Eq. (81) yields Eq. (13).

■

The prior variance ς^2 enters the prior, and the constants α, μ (through the contraction rate ρ), M, and the horizon K enter the loss L_y, so together with β and J they must be fixed before training.

If ς and α are selected from a finite grid of G values, applying the argument above at each grid point with confidence β/G and taking a union bound shows that Eq. (13) holds simultaneously for all

grid points where the constant C needs to be replaced by $C + \log G$. The inequality Eq. (79) holds simultaneously for all posteriors $\mathbb{Q}$. Choosing among several trained models afterwards, all with the same initialization Θ^0 and the same grid, is therefore covered by the same event and requires no further union bound.

C.2 Weight reuse beyond the trained depth

This section contains the bound for weight reuse from Section 4.2. The scheme runs for K outer iterations beyond a trained depth K_0 which is fixed before training, by reusing the last trained networks, $\Theta^k = \Theta^{K_0}$ for $k > K_0$. Corollary 2 splits the bound of Theorem 2 at an index m into an asymptotic term, which depends on both the training risk after iteration m and the divergence KL_{K_0}, and a transient term, which decays linearly with the rate ρ^{K-m}. The split holds simultaneously for all $1 \le m \le K$, and the main text uses $m = K_0$.

Corollary 2. Let the assumptions of Theorem 2 hold. We fix a horizon K_0 before training such that $K_0 \le K$ and reuse the last trained networks for $k > K_0$.

For a split index $1 \le m \le K$, define the late training risk $\hat{\Delta}^{>m} := \max_{m<k\le K} \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k]$, with $\hat{\Delta}^{>m} := 0$ if $m = K$, and the transient constant of the early iterations

$$B_m := \rho^m \, \mathbb{E}_y[\Phi_0] + \frac{1+\alpha}{\alpha} \sum_{k=1}^{m} \rho^{m-k} \, \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k].$$

Then, for any $\beta \in (0,1)$, with probability at least $1-\beta$, it holds for all $1 \le m \le K$ that

$$\mathbb{E}_{\mathbb{Q}}\mathbb{E}_y[\tilde{F}_y^K - F_y^*] \le \frac{1+\alpha}{\alpha(1-\rho)} \Big(\hat{\Delta}^{>m} + \tfrac{3M}{4} \sqrt{(\mathrm{KL}_{K_0} + C)/N} \Big) + \rho^{K-m} B_m, \tag{82}$$

where KL_{K_0} is the divergence of the random walks on $\Theta^{1:K_0}$.

Proof. Fix a number of outer iterations at least as large as the number of trained steps $K \ge K_0$. Under the weight reuse after the training horizon, the prior and posterior on $\Theta^{1:K}$ are the random walks Eq. (74) and Eq. (75) on $\Theta^{1:K_0}$, extended by the deterministic map that fixes $\Theta^k = \Theta^{K_0}$ for $k > K_0$. This prior remains independent of the data, and the proof of Theorem 2 holds for any data-independent prior.

Both distributions are images of their marginals on $\Theta^{1:K_0}$ under the same injective map that copies the weights, so $\mathrm{KL}(\mathbb{Q}\|\mathbb{P}) = \mathrm{KL}_{K_0}$. The bounded-error assumption holds for the reused networks as well, so Eq. (13) holds at horizon K with probability at least $1-\beta$.

On this event, we split the discounted training risk at an index $1 \le m \le K$. The early block satisfies

$$\sum_{k=1}^{m} \rho^{K-k} \, \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k] = \rho^{K-m} \sum_{k=1}^{m} \rho^{m-k} \, \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k],$$

and the late block satisfies

$$\sum_{k=m+1}^{K} \rho^{K-k} \, \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k] \le \hat{\Delta}^{>m} \sum_{j=0}^{K-m-1} \rho^j \le \frac{\hat{\Delta}^{>m}}{1-\rho}.$$

Writing $\rho^K \mathbb{E}_y[\Phi_0] = \rho^{K-m}\rho^m \, \mathbb{E}_y[\Phi_0]$ and collecting the terms yields Eq. (82). The split is applied to the inequality after the event is fixed, so the bound holds for every m on the same event. ∎

Corollary 2 holds at a single horizon K fixed in advance. For finitely many horizons $K \le K_{\max}$ fixed in advance a simple union bound argument already gives a simultaneous statement: applying Corollary 2 at each horizon with confidence $\beta/K_{\max}$ replaces C by $C + \log K_{\max}$. Since C enters only through $\sqrt{\mathrm{KL} + C}$, this penalty is small whenever KL dominates.

C.3 Numerical evaluation of the bound

This section describes how we evaluate Theorem 2 numerically for both problems. The problem-specific settings and results are given in Appendix D.10 for CT reconstruction and in Appendix E.9 for robust matrix completion.

We evaluate the inequality Eq. (81) from the proof of Theorem 2, which bounds the discounted posterior risk $\sum_k \rho^{K-k}\mathbb{E}_{\mathbb{Q}}[\mathbb{E}_y[\Delta_y^k]]$ directly. The full theorem statement Eq. (13) multiplies this quantity by $(1+\alpha)/\alpha$ and adds the decaying initial term $\rho^K\mathbb{E}_y[\Phi_0]$. Since α enters Eq. (81) only through ρ, the selected α lies at the lower end of the grid where $(1+\alpha)/\alpha$ is large. We therefore report the evaluation as a bound on the discounted error sum itself.

Plug-in constants Two constants of the theorem are not available in certified form, and we replace them by plug-in estimates we obtain from data.

- *The bound M.* The theorem requires $\Delta_y^k \le M$ for every instance and every parameter value. The output projection guarantees that such an M exists, but the resulting a priori value is several orders of magnitude larger than the errors that occur and makes the bound vacuous on both problems. We therefore use $\hat{M}(\varsigma)$, the largest Δ_y^k observed over the posterior samples at the variance ς. We report a running maximum over increasing ς, so that $\hat{M}$ is nondecreasing in ς, and bound it from below by the largest Δ_y^k at the trained weights, computed over the full training set for CT and over the training and test sets for robust matrix completion.
- *The constant μ.* The contraction rate ρ depends on the quadratic growth constant. The theorem requires one constant μ for all instances and for every parameter value in the support of $\mathbb{Q}$. We use a common empirical constant $\hat{\mu}$, the smallest quadratic growth ratio observed along the iterates. For CT, we compute it along the deployed iterates on the test set (Appendix D.7). For robust matrix completion, we additionally include the iterates under $\mathbb{Q}$ and the iterates of the classical solvers (Appendix E.9).

The resulting numbers are a numerical evaluation of the bound, not a certificate.

Monte Carlo estimates and grid The posterior risks $\mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k]$ are Monte Carlo estimates over sampled trajectories $\Theta^{1:K} \sim \mathbb{Q}$. We do not include the Monte Carlo error in the bound. We select α and ς from a grid of G fixed values, which by a union bound adds $\log G$ to the constant C in Theorem 2. We use $J = 40$ for the grid over the free parameter λ, which covers every posterior width we evaluate, and $\beta = 0.05$ for both problems. We compare the numerical evaluation of the PAC-Bayes bound to the trivial bound $\hat{M}(1-\rho^K)/(1-\rho)$, which follows from $\Delta_y^k \le \hat{M}$ alone.

D Dictionary regularized CT reconstruction

This section contains the proof of Proposition 1, the data generation, the baseline and learned methods, and additional results for the CT reconstruction experiments, including the evaluation of the PAC-Bayes bound.

D.1 Proof of Proposition 1

Proof. Fix a bounded set of data $\mathcal{Y}_0$ and a bounded set $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{Z}$. Here, $\mathcal{X}$ and $\mathcal{Z}$ are finite-dimensional and we work with the objective Eq. (15) without the rescaling by $1/(2\lambda_1)$ and convert the constant itself at the end by the same factor. The proof has four steps: we eliminate x by partial minimization, write the reduced problem in a form where the data only enters a strongly convex function, apply a quadratic growth result based on Hoffman's error bound, and return to the original variables.

Step 1: reduction to z. For a fixed z, the x-subproblem is strongly convex since its Hessian is $2(T^*T + \lambda_1 I) \succ 0$. Its unique minimizer follows from the normal equations as[2]

$$x^*(z) = (T^*T + \lambda_1 I)^{-1}(T^*y + \lambda_1 Dz). \tag{83}$$

[2] In practice it is infeasible to compute the closed form minimizer due to the high dimensionality of the problem.

Substituting $x^*(z)$ into the objective and simplifying with $(T^*T + \lambda_1 I)^{-1}T^* = T^*(TT^* + \lambda_1 I)^{-1}$ gives the reduced problem

$$\min_z \ \varphi_y(z) := \|\bar{A}z - Wy\|^2 + \lambda_2\|z\|_1, \qquad W := (I + TT^*/\lambda_1)^{-1/2}, \quad \bar{A} := WTD.$$

The operator $\bar{A}$ does not depend on y, and $\|W\| \le 1$. Let $Z^*_{\mathrm{red}} := \operatorname{argmin}_z \varphi_y(z)$. Every optimum $(x^*, z^*) \in Z^*$ satisfies $x^* = x^*(z^*)$, so $Z^* = \{(x^*(z), z) : z \in Z^*_{\mathrm{red}}\}$, and both problems have the same optimal value F^*_y. Since $\varphi_y(z) \ge \lambda_2\|z\|_1$, the function φ_y is coercive and Z^*_{red} is nonempty.

Step 2: polyhedral reformulation. We split $z = u - v$ with $u, v \ge 0$ and consider

$$\min_{u,v\ge 0} \ f_y(u, v) := g_y\big(\bar{A}(u - v)\big) + \lambda_2 \mathbf{1}^T(u + v), \qquad g_y(t) := \|t - Wy\|^2.$$

The function g_y is 2-strongly convex with a 2-Lipschitz gradient for every y. Taking u and v as the positive and negative parts of z shows that the optimal value is F^*_y. Since $\|u - v\|_1 \le \mathbf{1}^T(u + v)$, every minimizer $(\bar{u}, \bar{v})$ satisfies $\varphi_y(\bar{u} - \bar{v}) \le f_y(\bar{u}, \bar{v}) = F^*_y$, so $\bar{u} - \bar{v} \in Z^*_{\mathrm{red}}$. The problem has the form $g(Aw) + c^T w$ over the polyhedron $\{w : Cw \le d\}$ with $A = \bar{A}[I, -I]$, $c = \lambda_2\mathbf{1}$, $C = -I$, and $d = 0$. For this form, Necoara et al. [2019, Section 4.3] show that for every $M > 0$, every feasible w with $f_y(w) - F^*_y \le M$ satisfies

$$f_y(w) - F^*_y \ge \frac{\kappa_y}{2}\operatorname{dist}(w, W^*_y)^2, \qquad \kappa_y = \frac{2}{\theta^2(1 + 2M + 2c_y^2)},$$

where W^*_y is the set of minimizers, $c_y := \|\nabla g_y(\bar{A}z^*)\|$ for any $z^* \in Z^*_{\mathrm{red}}$, and θ is the Hoffman constant of the linear system with matrices A, c^T, and C. The Hoffman constant depends only on these matrices, which do not depend on y.

Step 3: bounds that hold for all $y \in \mathcal{Y}_0$. Let $\bar{y} := \sup_{y\in\mathcal{Y}_0}\|y\|$. For $z^* \in Z^*_{\mathrm{red}}$, we have $\|\bar{A}z^* - Wy\|^2 \le \varphi_y(z^*) \le \varphi_y(0) = \|Wy\|^2 \le \bar{y}^2$, so $c_y = 2\|\bar{A}z^* - Wy\| \le 2\bar{y}$. For $(\tilde{x}, \tilde{z}) \in \mathcal{S}$, the value $\varphi_y(\tilde{z}) - F^*_y \le \varphi_y(\tilde{z})$ is bounded by a constant $\bar{M}$ that depends only on $\mathcal{S}$, $\bar{y}$, $\|\bar{A}\|$, and λ_2. With $M = \bar{M}$, we obtain the constant $\bar{\kappa} := 2/(\theta^2(1 + 2\bar{M} + 8\bar{y}^2))$, which bounds κ_y from below for every $y \in \mathcal{Y}_0$.

Now take $(\tilde{x}, \tilde{z}) \in \mathcal{S}$, let $\tilde{w}$ consist of the positive and negative parts of $\tilde{z}$, so that $f_y(\tilde{w}) = \varphi_y(\tilde{z})$, and let $(\bar{u}, \bar{v})$ be the projection of $\tilde{w}$ onto W^*_y. Since $\bar{u} - \bar{v} \in Z^*_{\mathrm{red}}$ and the map $(u, v) \mapsto u - v$ has norm $\sqrt{2}$,

$$\operatorname{dist}(\tilde{z}, Z^*_{\mathrm{red}})^2 \le \|\tilde{z} - (\bar{u} - \bar{v})\|^2 \le 2\operatorname{dist}(\tilde{w}, W^*_y)^2 \le \frac{4}{\bar{\kappa}}\big(\varphi_y(\tilde{z}) - F^*_y\big).$$

Step 4: return to (x, z). Since $x^*(\tilde{z})$ minimizes $F(\cdot, \tilde{z})$, we have $F(\tilde{x}, \tilde{z}) \ge F(x^*(\tilde{z}), \tilde{z}) = \varphi_y(\tilde{z})$. With $B = D$ and the description of Z^* from Step 1,

$$\inf_{(x',z')\in Z^*} \|B(z' - \tilde{z})\|^2 = \inf_{z'\in Z^*_{\mathrm{red}}} \|B(z' - \tilde{z})\|^2 \le \|B\|^2 \operatorname{dist}(\tilde{z}, Z^*_{\mathrm{red}})^2.$$

Combining the inequalities yields the quadratic growth inequality at every point of $\mathcal{S}$ with constant $\bar{\kappa}/(2\|D\|^2)$ for the unscaled objective and every $y \in \mathcal{Y}_0$. For the objective rescaled by $1/(2\lambda_1)$, whose coupling term has the form of Eq. (1) with $B = D$, this gives Assumption 1 with $\mu = \bar{\kappa}/(4\lambda_1\|D\|^2)$. ■

D.2 Problem parametrization

Imaging data We are using the computed tomography (CT) imaging data of the human abdomen from the 2016 Low Dose CT Grand Challenge as ground truth [Moen et al., 2021]. The dataset contains imaging data from a total of 10 patients out of which we are using the data from 9 patients for training and validation while the final patient is reserved for testing.

Data simulation In this paper, we are only focusing on 2D image reconstruction by splitting the 3D volumetric data into axial slices. We are using the Operator Discretisation Library (ODL) [Adler et al., 2017] with the astra-toolbox CT backend [Van Aarle et al., 2015, 2016] to simulate 2D projection data from the extracted 512×512 image slices. The acquisition geometry is modeled as a

fan beam geometry with 1 000 uniformly partitioned angles and 1 000 detector elements. The noise free tomographic data modeled by the forward operator is given by $y = Tx$, where T is the linear Radon transform operator also referred to as Ray Transform.[3]

We model the observed noisy data $y_{\text{noisy}} \approx y$ as low-dosage clinical CT data. Therefore, we re-scale the images given in Hounsfield scale by $\mu_0/1\,000$ for the water attenuation constant $\mu_0 = 0.0192\text{mm}^{-1}$. For the single energy X-Ray noise model we assume a Poisson distribution with mean corresponding to the number of photons absorbed by the detector. This number of absorbed photons is modeled by the Beer-Lambert law as

$$N_d = N_0 \cdot \exp\left(-Tx\right). \tag{84}$$

The noisy observation is given as a linearization of the data:

$$y_{\text{noisy}} = -\ln\left(\frac{1}{N_0}\,\text{Poisson}(N_d)\right). \tag{85}$$

To avoid taking the logarithm of zero, the photon counts are clipped from below at 1, so that $y_{\text{noisy}} \leq \ln N_0$. Since Poisson counts have no upper limit, the data is not bounded from below, although large deviations have negligible probability. In our experiments we choose $N_0 = 50\,000$ to model the clinical low-dose setting.

From this simulation, we obtain pairs of noisy observations and ground truth images $\{(x, y_{\text{noisy}})\}_{i=1}^{N}$ with a total of 2 147 training samples, 21 validation samples, and 210 samples from a different individual used for testing purposes. Training the data-driven optimizer, however, is done in an unsupervised fashion where only the observations y_{noisy} are required that parameterize the objective. Further, note that our primary goal is not to recover the ground truth x as well as possible but rather solve the optimization problem Eq. (15) as fast as possible. How to design a good regularizer such that the solution to the optimization problem is close to the true solution x is out of scope for this paper.

Dictionary regularization In our experiments, the optimization objective is given by the data fidelity term, a term corresponding to dictionary representation of the signal as well as a sparsifying ℓ_1 term on the dictionary coefficients. The dictionary consists of 128 dictionary elements of size 16×16 and are pre-trained using the method described by Rudzusika et al. [2022]. The synthesis operator D is a linear operator reconstructing an image from the dictionary coefficients using the learned dictionary elements. As regularization parameters, we choose $\lambda_1 = 50$ and $\lambda_2 = 0.0016$ to minimize the validation error in terms of image reconstruction performance. In order to reduce the number of dictionary coefficients for the reconstruction, we apply a stride 2 to the elements of the dictionary.

Optimization variables The total number of optimization variables in the CT reconstruction objective from Eq. (15) is given by the sum of image variables (for the x-subproblem) which is of dimension $512 \times 512 = 262\,144$ and the dictionary coefficients (for the z-subproblem) which is $128 \times 256 \times 256 = 8\,388\,608$. Thus, we have a total of 8 650 752 variables.

D.3 Baseline methods

Here, we discuss how all methods apply the pre- and post-processing. Further, we specify the details for the solving procedure for the alternating and inexact scheme and how the iterates are numerically computed. Further, we provide details how the ground truth solution (x^*, z^*) is approximated.

Pre- and post-processing In order to apply the dictionary regularized variational formulation, we are removing the low frequency components from the data with a low-pass filter in the pre-processing step. Therefore, in a first step we apply the filtered back projection algorithm (FBP) with a Hanning filter and relative frequency cut-off set to 0.1. This step is necessary to allow good regularization performance via the dictionary elements that only model the high frequency signal components [Rudzusika et al., 2022]. To get the final reconstructed image, we add the previously extracted low frequency structures of the image to the result obtained from the learned alternating minimization algorithm.

In order to initialize our alternating minimization scheme, we are using the analytical filtered back projection (FBP) method with Hanning filter and relative frequency cut-off 0.75 to obtain an initial guess x^0 that is provided to the algorithm.

[3] Note that T is implemented as a matrix-free operator. It is not possible to form the matrix T directly since the size of the matrix is prohibitively large containing over 262 billion elements. The matrix-free version directly computes the matrix-vector product by solving line integrals.

The baseline methods utilize the Lipschitz constant of the data fidelity term and the quadratic coupling between the variable blocks to set the step-size. To compute the Lipschitz constant, we use the power method with 10 iterations which estimates the largest singular values of the linear operators T (ray transform) and D (dictionary synthesis operator) as part of the pre-processing.

Solving the subproblems In each iteration of the alternating minimization scheme, the two subproblems shown in Eq. (2) have to be solved to compute the next iterates. For both subproblems the solution obtained in the previous outer iteration can be used to warm-start the solver leading to a significant reduction in solving time for the iterative schemes.

The first sub-problem in Eq. (2), which corresponds to the update of the x-variable, is both convex and smooth. Therefore, we solve the problem using the first-order gradient method with Nesterov momentum for the baseline method. Since this subproblem involves the forward operator in the CT reconstruction problem, it is still computationally demanding.

The second subproblem is non-smooth due to the ℓ_1 norm used to obtain sparse dictionary coefficients. Since the proximal operator is available as a closed form solution by the soft shrinkage operator [Beck, 2017] we apply the forward-backward splitting method. Similar to the first subproblem, we use the corresponding accelerated version of the proximal gradient method (FISTA) [Beck and Teboulle, 2009].

In practice, we stop the iterative scheme to solve the inner subproblems when the relative change in objective value is less than a predefined threshold (in practice we choose 10^{-5}). The maximum number of inner iterations is variable giving rise to the different baseline methods. We run the outer scheme using a fixed number of iterations ($K = 20$) and train a neural network to approximate the solution to the subproblem of each iteration as outlined in Section 4 and as we discuss in detail in the next section.

Ground truth solution To evaluate the optimality gap, we approximate a minimizer (x^*, z^*) by running the baseline alternating scheme for up to 200 outer iterations with up to 50 inner iterations for each subproblem, stopping each loop once the relative decrease of the objective falls below 10^{-5}. Since the subproblems are solved only approximately, the reference value $F(x^*, z^*)$ upper-bounds the optimal value F^*, so the reported relative gaps underestimate the true gaps. All methods are evaluated against the same reference ensuring a fair comparison. Note that the problem is not strongly convex so the found minimizer may be non-unique.

D.4 Learned methods

In each outer iteration k we parameterize the learned solvers ϕ_{θ^k} and ψ_{ϑ^k} as small convolutional networks tailored to each subproblem. Both networks use the same underlying backbone consisting of 3×3 convolutions with 32 hidden channels which are interleaved with GroupNorm layers and ReLU activations. The two approaches only differ in terms of input and output projection. Further, the z-network has additional learned soft-thresholding to incorporate the non-smooth ℓ_1 penalty into the architecture. We do not share weights across outer iterations, so each of the $K = 20$ iterations has its own pair of networks.

The x-networks: The x-network takes as input the current iterate $\tilde{x}^k$ stacked with the two terms of the partial gradient $\nabla_x F(\tilde{x}^k, \tilde{z}^k)$ separated into the data-fidelity contribution $2T^T(T\tilde{x}^k - y)$ and the coupling contribution $2\lambda_1(\tilde{x}^k - D\tilde{z}^k)$, giving 3 input channels. The network produces a scalar correction d_x of the same spatial dimension as the image, and the next iterate is formed by augmenting a single preconditioned gradient step with this correction

$$\tilde{x}^{k+1} = \tilde{x}^k - \frac{1}{L_x}\big(\nabla_x F(\tilde{x}^k, \tilde{z}^k) + d_x\big), \tag{86}$$

where L_x is the Lipschitz constant of $\nabla_x F(\,\cdot\,, \tilde{z}^k)$ estimated by power iteration during pre-processing. The backbone uses 4 convolutional layers with 32 hidden channels which results in 29 127 trainable parameters per outer iteration.

The z-networks: The z-network is inspired by the LISTA architecture [Gregor and LeCun, 2010] which unrolls steps of a proximal gradient method into a learned shrinkage map. Unlike the original LISTA formulation, the network is trained in a fully unsupervised fashion against the per-iteration subproblem objective and we use only a single shrinkage step. The network takes as input the current sparse code $\tilde{z}^k \in \mathbb{R}^{128\times256\times256}$ with 128 dictionary atoms living on a coarser spatial grid than the image due to the stride-2 dictionary D, stacked with the smooth-part gradient $2\lambda_1 D^T(D\tilde{z}^k - \tilde{x}^{k+1})$,

giving 256 input channels. The network produces a correction d_z, and the next iterate is obtained from an additional proximal-gradient step augmented by d_z followed by atom-wise soft-thresholding

$$u = \tilde{z}^k - \frac{1}{L_z}\big(g_z + d_z\big), \qquad \tilde{z}^{k+1} = \operatorname{sign}(u) \odot \max\big(|u| - \theta,\, 0\big), \tag{87}$$

where $L_z = 2\lambda_1 \|D\|^2$ and $\theta \in \mathbb{R}^{128}_{>0}$ is a learned per-atom threshold parameterized as $\theta = \operatorname{softplus}(\rho)$ to enforce positivity. We zero-initialize the final convolution and initialize ρ such that $\theta = \lambda_2 / L_z$ at the start of training, so the network reduces to one classical ISTA step at initialization. The z-network has 139 328 trainable parameters per outer iteration.

Output projection: To keep the learned iterates bounded for every parameter value of the posterior as discussed in Section 4, the output of both networks is projected onto an a priori box,

$$\tilde{x}^{k+1} \leftarrow \Pi_{[-0.1,\,0.1]}(\tilde{x}^{k+1}) \qquad \tilde{z}^{k+1} \leftarrow \Pi_{[-4,\,4]}(\tilde{z}^{k+1})$$

applied elementwise. The box for x acts on the image after the low-frequency component has been removed (Appendix D.3) and not on the physical attenuation values. Since the projection is part of the learned map, it is also applied to every parameter sample drawn from the posterior in Appendix D.10. On the parameters trained from the greedy scheme, the projection is inactive across all test instances.

Training settings At each outer iteration k we train the x- and z-networks sequentially: we first fit ϕ_{θ^k} against the per-iteration x-subproblem objective with θ^{k-1} as a warm start, and then, with θ^k held fixed, fit ψ_{ϑ^k} against the per-iteration z-subproblem objective with ϑ^{k-1} as a warm start (see Algorithm 1). Initializing from the previous iteration (line 3 of Algorithm 1) lets us reduce the training budget at later iterations to a fine-tuning regime. The number of training epochs for each network decays with the outer iteration index k according to the schedule

$$(\mathrm{E}_x^k, \mathrm{E}_z^k) = \begin{cases} (10, 50) & \text{for } k = 1, \\ (5,\ 25) & \text{for } 2 \le k \le 6, \\ (2,\ 10) & \text{for } k \ge 7, \end{cases}$$

where E_x^k and E_z^k are the number of epochs used to train the x- and z-network at iteration k, respectively. We use a batch size of 4 for the x-subproblem and 8 for the z-subproblem. As the outer scheme converges, the data distribution presented to subsequent networks stabilizes, making the warm start increasingly effective.

Additional hyper-parameters and training settings are shown in Table 2.

Learned baselines We consider two learned baselines that target the canonical objective Eq. (15) but differ in how the predictor is parameterized. The *fully-amortized baseline* maps the warm-started variables to a prediction of (x, z) in a single forward pass. The *semi-amortized baseline* retains the iterative structure of the proposed scheme but trains all per-iteration networks jointly via algorithm unrolling [Monga et al., 2021]. The hyperparameters of the two methods are summarized in Table 2.

Fully-amortized baseline: The fully-amortized network predicts (x, z) in a single forward pass from the initial input variables. We use a Y-shaped architecture that consists of a shared image-domain residual trunk followed by two thin heads, one for each variable. The trunk consists of an input projection and 6 residual blocks, each with two 3×3 convolutions of 96 filters interleaved with GroupNorm and ReLU activations and a skip connection across the block. The x-head is a 2-layer convolutional network that produces an additive correction to the input. The z-head is a 3-layer network that consumes the trunk features projected into atom space concatenated with $\tilde{z}^0$, $D^T\tilde{x}$ and $D^T D\tilde{z}^0$, produces a LISTA-style correction, and applies the same atom-wise soft-thresholding step as in the proposed z-network.

Semi-amortized baseline: The semi-amortized baseline uses the same per-iteration architecture as the proposed greedy method but differs in the training strategy. Rather than training each pair of networks in isolation against the per-iteration subproblem objective, all networks are trained jointly end-to-end with a single loss on the final iterate $(\tilde{x}^K, \tilde{z}^K)$ propagated through the unrolled scheme. We use $K = 10$ outer iterations for this baseline.

All baselines use the same validation frequency and early stopping based on the validation performance to avoid overfitting.

Table 2: Details about the training settings for the different learned methods

Hyperparameter	Greedy	Greedy + FT	Semi-amortized	Fully-amortized
Outer iterations (K)	20	10	10	1
Total parameters	3 369 100	1 684 550	1 684 550	2 213 957
Training epochs	Appendix D.4	greedy epochs + 10	50	50
Optimizer	AdamW	AdamW	AdamW	AdamW
Batch size	4 (x) / 8 (z)	2	2	2
LR Scheduler	cosine	cosine	cosine	cosine

D.5 Computational resources

All reported experiments are executed on a single NVIDIA TITAN Xp GPU with 12 GB of memory on an in-house server with 12 CPU cores and 62 GB of RAM.

Training compute A complete layer-wise training run of the learned alternating model takes approximately 17 hours of wall-clock time on the hardware above. Additional end-to-end fine-tuning takes around 6 more hours for the model. The fully-amortized baseline takes about 17 hours, and the semi-amortized baseline with $K = 10$ about 34 hours. We repeated full model training 5 times for the paper. The training times are also reported in Table 1.

Inference compute For all wall-clock timings we repeat each evaluation 3 times after a warm-up pass and report the mean. On a single test instance, the exact baseline with 100 inner FISTA iterations per outer step requires about 18 s, while the learned model solves the same problem in about 1.1 s, a speedup of roughly 16×. The fully and semi-amortized baselines as well as the end-to-end fine-tuned model run in 0.3–0.6 s per instance. For learned methods, every test run additionally solves each inner subproblem to high accuracy with 100 FISTA iterations in order to report the per-subproblem suboptimality used in Fig. 3. This dominates the inference cost and brings a full evaluation pass over the 210 test instances to roughly one day on the same hardware. The high-accuracy reference solution (x^*, z^*) used as the denominator of the relative objective gap was computed once and required an additional day of compute.

PAC-Bayes evaluation The posterior Monte Carlo sampling in Appendix D.10 runs the learned scheme with $K = 20$ for 15 values of ς and 5 trajectories each on 10 training slices, including the reference solves for the per-iteration errors, and takes 4.4 hours.

Previous experiments The full research project required several times the compute of the reported runs. During development we trained roughly 20 additional models that are not used in the paper, mostly with reduced numbers of outer iterations between 5 and 10 or with shorter training schedules.

D.6 Licenses and assets

Table 3 summarizes the third-party datasets and software used in this work. All assets are used in accordance with their respective licenses and within the terms of use.

Table 3: Third-party assets used in this work.

Asset	Type	License	Reference
Low Dose CT Grand Challenge	Dataset	CC BY-NC 4.0 / NIH Controlled Data Access Policy	[Moen et al., 2021]
ODL	Software	MPL-2.0	[Adler et al., 2017]
ASTRA Toolbox	Software	GPL-3.0	[Van Aarle et al., 2015, 2016]
PyTorch	Software	BSD-3-Clause	[Paszke et al., 2019]

D.7 Verification of assumptions

Here, we empirically investigate the ability of the trained neural network to fulfill the assumptions based on which we prove the convergence of the inexact scheme.

Quadratic growth assumption To validate Assumption 1 we collect the iterates $\{(\tilde{x}^k, \tilde{z}^k)\}_{k=1}^K$ produced by the learned alternating scheme on each of the 210 test instances, together with the

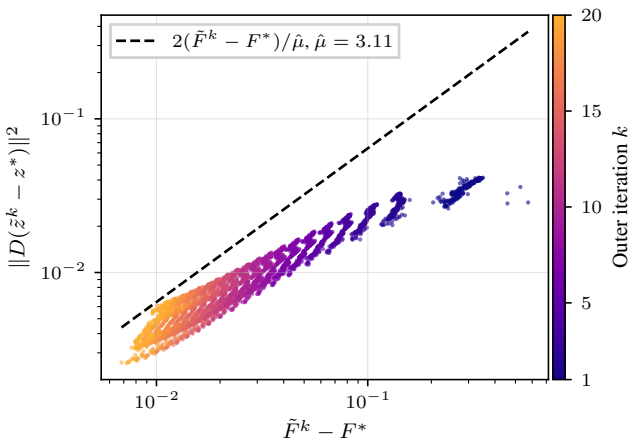


Figure 5: Numerical verification of the assumption of the quadratic growth in a neighborhood of the optimal set as stated in Assumption 1.

estimated optimum (x^*, z^*). The reference value $F(x^*, z^*)$ and the logged function values $F(\tilde{x}^k, \tilde{z}^k)$ are evaluated on the canonical objective Eq. (1), that is, after rescaling the objective by $1/(2\lambda_1)$ as discussed in the footnote of Eq. (15).

We discard the initial iterate $k = 0$ since it precedes any optimization step and is uninformative for the assumption. For each remaining iterate we plot $\|D(\tilde{z}^k - z^*)\|^2$ against $F(\tilde{x}^k, \tilde{z}^k) - F(x^*, z^*)$ in Fig. 5, where the color encodes the iteration index k. The empirical quadratic growth constant

$$\hat{\mu} \;=\; \min_{k, y_i} \frac{2\left(F(\tilde{x}^k_{y_i}, \tilde{z}^k_{y_i}; y_i) - F(x^*_{y_i}, z^*_{y_i}; y_i)\right)}{\|D(\tilde{z}^k_{y_i} - z^*_{y_i})\|^2} \;\approx\; 3.11 \tag{88}$$

is the largest constant for which Assumption 1 holds across every retained iterate; the dashed line in Fig. 5 traces the corresponding upper bound $2(F - F^*)/\hat{\mu}$ on $\|D(z - z^*)\|^2$, and every iterate sits on or below it. Since $\hat{\mu}$ is a minimum over finitely many iterates and uses one computed minimizer z^*_y, it is an empirical estimate and not a certified bound on the constant in Assumption 1. The true constant can be larger leading to faster convergence.

Bounded parameter drift The complexity term in Theorem 2 depends on the cumulative parameter drift $\sum_k \|\hat{\Theta}^k - \hat{\Theta}^{k-1}\|^2$ through $\mathrm{KL}(\mathbb{Q}\|\mathbb{P})$, where $\hat{\Theta}^k$ stacks the trained weights of both subproblem networks at iteration k. Figure 6 reports the relative drift $\|\hat{\Theta}^k - \hat{\Theta}^{k-1}\|^2 / \|\hat{\Theta}^{k-1}\|^2$ produced by greedy training with warm-starting over 20 outer iterations. The drift decays by roughly an order of magnitude over the first six iterations, drops sharply once the per-iteration training budget is reduced, and continues to shrink towards 10^{-3} thereafter.

We observe this decay empirically. A plausible explanation is that consecutive subproblems become similar as the scheme converges, so warm-started fine-tuning needs only small parameter adjustments. Since the late increments are small, additional trained iterations add little to the divergence, so the complexity term grows only slowly with the trained depth.

Efficient warm-starting of the training Greedy training (Algorithm 1) initializes the networks at iteration k from the parameters trained at iteration $k - 1$. Figure 7 reports the per-outer-iteration training curves of the x- and z-networks, each normalized to its loss at the start of the iteration, where the panel headers show the epoch budget allocated to each network. The first iteration is trained from cold-initialized weights and consumes the largest budget, with both networks reducing their loss substantially. From the third iteration the x-network is essentially at a fixed point, while the z-network continues to exhibit only mild residual fitting. The flattening of both curves justifies the budget cuts visible in the panel headers and supports the claim of Section 4.1 that warm-starting reduces the per-iteration training cost at later iterations.

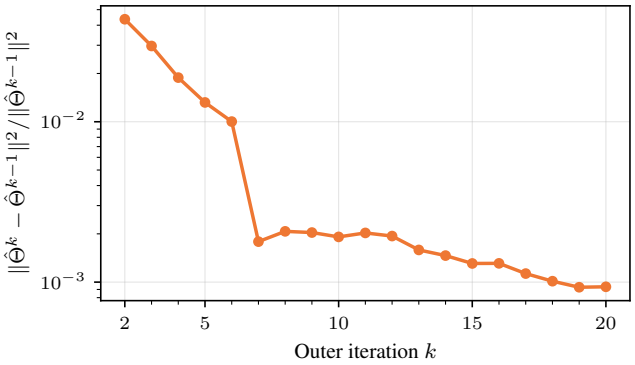


Figure 6: Relative parameter drift of the learned model across outer iterations k. We can see a decrease of the relative drift across the training epochs even though the training steps remain constant for later iterations.

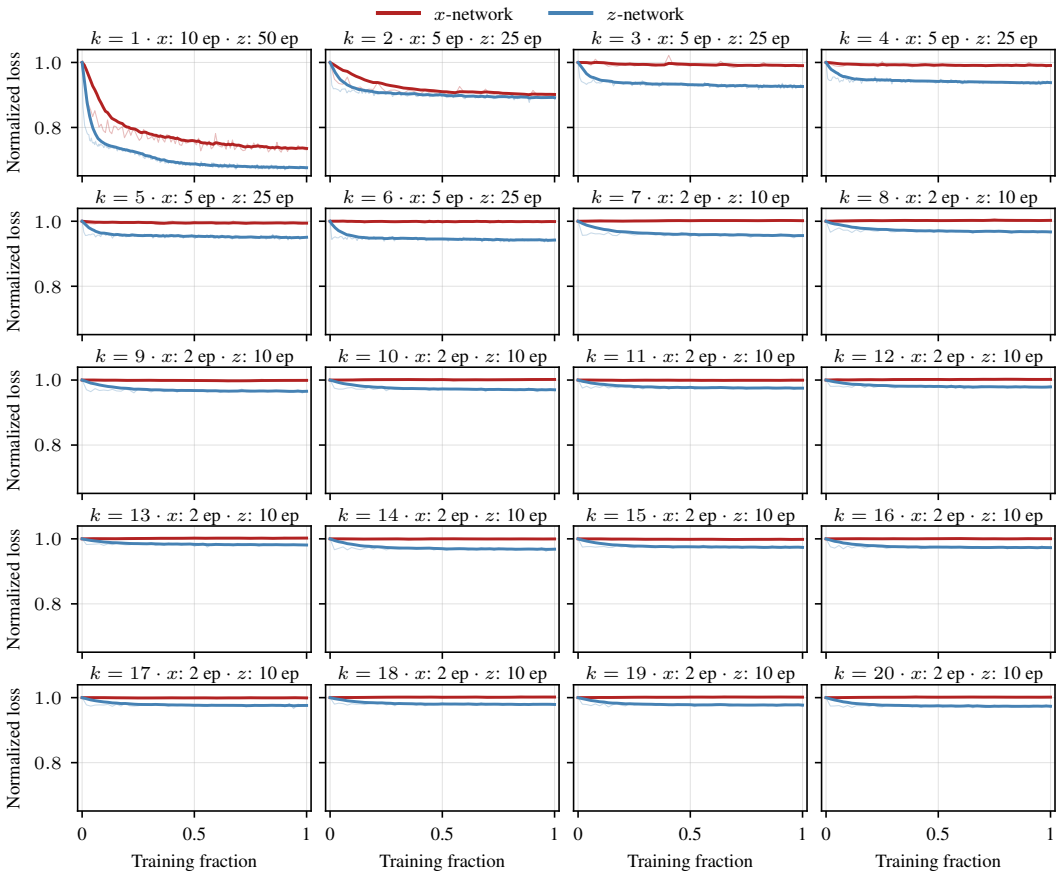


Figure 7: Per-iteration training loss of the x- and z-networks under greedy training with warm-starting, normalized to the loss at the start of each outer iteration. Panel headers list the epoch budget used for each network at iteration k.

D.8 Learned method comparison

In Fig. 8 we compare three training strategies that share the architecture introduced in Appendix D.4 but differ in how the per-iteration networks are fitted: (i) *greedy training* via our proposed Algorithm 1; (ii) *greedy training with end-to-end fine-tuning*, where the K greedily trained network pairs are subsequently fine-tuned jointly through the full alternating scheme; and (iii) a *semi-amortized* variant that retains the per-iteration architecture of the proposed scheme but trains all K network pairs jointly through the unrolled alternating scheme.

Greedy training reaches a slightly higher final-iteration gap than the unrolled baseline, consistent with the objective values reported in Section 5.4. Adding end-to-end fine-tuning to the greedy networks closes this gap and produces the lowest final-iteration error of the three variants. Interestingly, the fine-tuned variant is able to reduce the functional value further than the greedy variant across most iterations. In contrast, the unrolled variant starts at a markedly higher relative gap and decreases monotonically as k approaches K, since end-to-end training penalizes only the final iterate and does not train on the intermediate iterates of the alternating scheme.

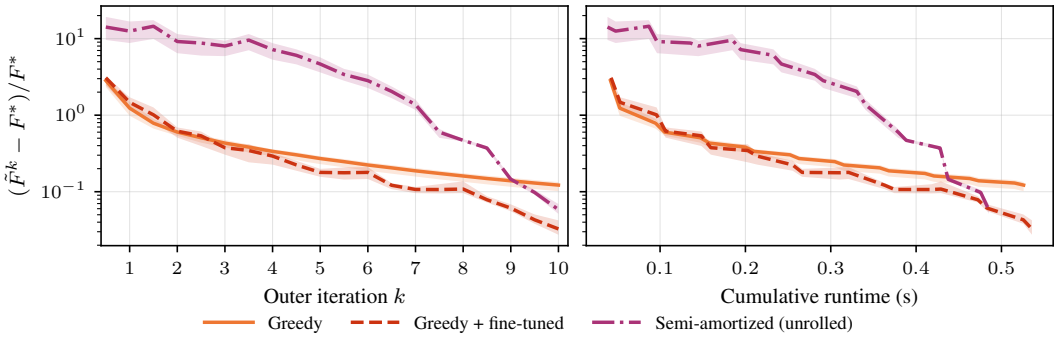


Figure 8: Comparison of three training strategies for the learned alternating scheme at $K = 10$ outer iterations: greedy training, greedy training with end-to-end fine-tuning, and the semi-amortized variant. Both panels show the relative objective gap $(\tilde{F}^K - F^*)/F^*$ on a log scale, against the outer iteration k on the left and against cumulative runtime on the right. The shaded bands cover the 5th–95th percentile range over the test instances.

D.9 Comparison to learned warm-starting

Using learned warm-starting methods [Sambharya et al., 2024] is an orthogonal idea to our acceleration of the individual subproblems. The original method is warm-started using the FBP solution as we describe in Appendix D.2. In the experiment here, we instead reuse the solution obtained from the fully-amortized model as a warm start to the alternating minimization scheme.

The results are shown in Fig. 9 where we can see that the model performs very similar to the inexact method and only when restricting the methods to a single inner iteration there is a benefit to the learned warm starting. Both methods are dominated by the learned approach which leads to significantly better performance. For the single iteration problem the learned method is slightly slower overall due to the additional computation required to predict the warm-start. When using additional iterations for the inner solvers some of this benefit vanishes since the x-subproblems are faster to converge for the warm-started version leading to a slight overall benefit in the subproblem computations. However, overall both methods perform nearly on par when using sufficiently many inner iterations.

This is consistent with our theory: in the bound of Theorem 1, the warm start enters explicitly only through the initial Lyapunov energy, which is discounted exponentially in later iterations, although it also changes the later errors. Thus, this does not effect the overall convergence speed as much as the later error terms. Instead, the convergence is dominated by the second term in the bound which is the suboptimality of the inner iterations that can lead to a large gap to the true function value.

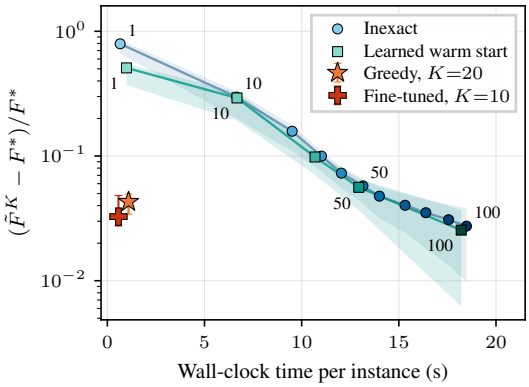


Figure 9: Pareto-front comparison of the learned warm start with the inexact and learned methods. Both iterative methods are run for 20 outer iterations with different number of inner iterations. The inner solver can be stopped early if the subproblem has converged. Markers show the median over test instances and bands the 5th–95th percentiles and numbers give the inner-iteration budget.

D.10 Evaluation of the PAC-Bayes bound

We evaluate the bound with the procedure of Appendix C.3. The CT data used for the training does not satisfy the assumptions in Theorem 2. The training instances are axial slices from 9 patients, so neighboring slices of the same patient are correlated and the N samples are not i.i.d. draws from a population distribution. The bound is therefore not a guarantee even with certified constants, which does not change the conclusion below since it is uninformative for every chosen variance ς.

We sample 5 posterior trajectories per value of ς on 10 training images. The grid consists of 200 equally spaced values of α in $[10^{-4}, 0.999\,\alpha_{\max}]$, where $\alpha_{\max}$ is the largest α allowed by Eq. (7), and the 15 sampled values of ς between 10^{-6} and 10, so that $G = 3\,000$ and $C = 15.4$.

Results We evaluate the greedy model with $K = 20$ on a fixed subset of $N = 2\,000$ training slices (Appendix D.2), with $\hat{\mu} = 3.11$. Table 4 shows that the bound is uninformative for every ς: it exceeds the trivial bound for $\varsigma \leq 0.32$, and for $\varsigma \geq 1$ it lies below the trivial bound only because both exceed the observed discounted gap by more than six orders of magnitude. The Monte Carlo standard error of the risk term over the 5 posterior trajectories is at most 14% of its value for $\varsigma \leq 0.1$ and 20% at $\varsigma = 0.32$. At $\varsigma = 1$ it grows to 69%, so within two standard errors the ratio of the bound to the trivial bound lies between 0.69 and 1.0. This error does not account for evaluating the posterior on 10 of the N training slices, and neither source changes the conclusion.

The main cause for the loose bound is the large parameter drift $D = \sum_k \|\hat{\Theta}^k - \hat{\Theta}^{k-1}\|^2 = 3\,819$ of the CT networks. The ratio between the complexity term and the trivial bound is approximately $\frac{3}{4}\sqrt{(\mathrm{KL} + C)/N}$, which does not depend on M. It falls below one only if $\mathrm{KL} < 16N/9 - C$, that is, only if $\varsigma \gtrsim 0.73$. Gaussian perturbations of this size destroy the networks: at $\varsigma = 1$, the posterior risk

Table 4: Evaluation of Eq. (81) for CT reconstruction ($K = 20$, $N = 2\,000$, $\rho = 0.243$) over the prior standard deviation ς. The risk term is $\sum_k \rho^{K-k}\mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k]$, and the trivial bound grows with ς because the plug-in estimate $\hat{M}$ does.

variance ς	KL	risk term	complexity	RHS	trivial	RHS/trivial
0.001	1.9×10^{9}	2.2×10^{-3}	608	608	0.83	733
0.0032	1.9×10^{8}	0.080	192	192	0.83	232
0.01	1.9×10^{7}	1.57	214	216	2.92	74
0.032	1.9×10^{6}	4.11	139	143	5.99	24
0.1	1.9×10^{5}	6.45	109	116	14.9	7.8
0.32	1.9×10^{4}	38.7	237	276	102	2.7
1	1.9×10^{3}	1.1×10^{3}	7.1×10^{3}	8.2×10^{3}	9.7×10^{3}	0.85

exceeds 10^3. Conversely, for $\varsigma \leq 3 \times 10^{-4}$ the posterior risk matches that of the deployed networks, but then the divergence exceeds 10^{10}.

The risk side of the bound, in contrast, is small and generalizes. At the deployed weights, the discounted training risk is 1.13×10^{-3}. The per-iteration training risk $\hat{\Delta}^k$ decreases from 6.5×10^{-2} at $k = 1$ to 8.2×10^{-4} at $k = 20$, and the test risk follows the same profile. The obstruction is therefore the size of the networks relative to N, measured by the drift, and not the generalization of the learned solver itself. For a smaller learned update, the bound becomes informative (Appendix E.9).

E Robust matrix completion

E.1 Proof of Proposition 2

Here, we show the proof for the quadratic growth condition for the robust matrix completion problem. Our approach follows the standard dual certificate analysis of robust PCA [Candès et al., 2011, Chandrasekaran et al., 2011] and the quadratic growth literature for structured convex problems [Cui et al., 2017, Drusvyatskiy and Lewis, 2018, Zhou and So, 2017].

We first make the two conditions of Proposition 2 precise. Let (x^*, z^*) be a minimizer of the robust completion objective, $R = P_\Omega(y - x^* - z^*)$ the residual at the optimum, T the tangent space of the rank-r matrix manifold at $x^* = U\Sigma V^T$, and S the support of z^*. First-order optimality states that R is a subgradient of $\lambda_1\|\cdot\|_*$ at x^* and of $\lambda_2\|\cdot\|_1$ at z^*, that is,

$$R = \lambda_1(UV^T + W), \quad P_T W = 0, \quad \|W\|_2 \leq 1,$$
$$R_{ij} = \lambda_2 \mathrm{sign}(z^*_{ij}) \text{ for } (i,j) \in S, \quad |R_{ij}| \leq \lambda_2 \text{ for } (i,j) \notin S,$$

where $\|\cdot\|_2$ denotes the spectral norm.

- *Strict complementarity* holds if the two inequalities hold with a margin: there are $\delta_1, \delta_2 \in (0, 1]$ with $\|W\|_2 \leq 1 - \delta_1$ and $|R_{ij}| \leq \lambda_2(1 - \delta_2)$ for all $(i, j) \notin S$.
- *Restricted injectivity* holds if there is $\sigma > 0$ with $\|P_\Omega(u + v)\|_F^2 \geq \sigma^2(\|u\|_F^2 + \|v\|_F^2)$ for all $u \in T$ and all v supported on S.

The first condition controls moving away from the minimizer by leaving the low-rank and sparse structure, and the second controls moving along it. The proof shows that Proposition 2 holds with an explicit constant μ that depends only on δ_1, δ_2, σ, λ_1, λ_2, as well as the chosen sublevel value.

Proof. We write $\mathcal{A}(x, z) := P_\Omega(x + z)$, so that the objective is $F(x, z) = \frac{1}{2}\|P_\Omega y - \mathcal{A}(x, z)\|_F^2 + \lambda_1\|x\|_* + \lambda_2\|z\|_1$. Fix a sublevel value $\bar{\varepsilon} > 0$ and a point $(\tilde{x}, \tilde{z})$ with $F(\tilde{x}, \tilde{z}) \leq F^* + \bar{\varepsilon}$. Let $d = (d_x, d_z) := (\tilde{x} - x^*, \tilde{z} - z^*)$ and $\varepsilon := F(\tilde{x}, \tilde{z}) - F^* \in [0, \bar{\varepsilon}]$.

The proof has four steps: we split the gap ε into a curvature term and a convexity gap, then we bound the component of d that leaves the structure $T \times S$ by the convexity gap and the component along $T \times S$ by the curvature term, and finally combine the two bounds.

Step 1: split of the gap. The coupling term is quadratic, so its expansion around (x^*, z^*) is exact. Since R is supported on Ω, we have $\langle R, \mathcal{A}d\rangle = \langle R, d_x\rangle + \langle R, d_z\rangle$, and therefore

$$\varepsilon = \frac{1}{2}\|\mathcal{A}d\|_F^2 + D_x + D_z, \tag{89}$$

where

$$D_x := \lambda_1\|\tilde{x}\|_* - \lambda_1\|x^*\|_* - \langle R, d_x\rangle, \qquad D_z := \lambda_2\|\tilde{z}\|_1 - \lambda_2\|z^*\|_1 - \langle R, d_z\rangle.$$

Both D_x and D_z are nonnegative because R is a subgradient of the respective norm.

Step 2: strict complementarity bounds the component leaving the structure. For the ℓ_1 part, each entry $(i, j) \in S$ contributes a nonnegative term to D_z, and each entry $(i, j) \notin S$ contributes $\lambda_2|(d_z)_{ij}| - R_{ij}(d_z)_{ij} \geq \lambda_2\delta_2|(d_z)_{ij}|$. Hence $D_z \geq \lambda_2\delta_2\|(d_z)_{S^c}\|_1$, where $(d_z)_{S^c}$ is the restriction of d_z to the complement of S.

For the nuclear-norm part, let $U_2\Sigma_2 V_2^T$ be a compact SVD of $P_T^\perp d_x$ and set $W' := U_2 V_2^T$. Since $P_T^\perp d_x \in T^\perp$, the matrix W' lies in $T^\perp$ with $\|W'\|_2 \leq 1$, so $UV^T + W'$ is a subgradient of $\|\cdot\|_*$ at

x^*, and $\langle W', d_x\rangle = \|P_T^\perp d_x\|_*$. The subgradient inequality with $UV^T + W'$ and the duality between the spectral and nuclear norms give

$$D_x \geq \lambda_1\langle W' - W, d_x\rangle = \lambda_1\left(\|P_T^\perp d_x\|_* - \langle W, P_T^\perp d_x\rangle\right) \geq \lambda_1\delta_1\|P_T^\perp d_x\|_*.$$

Let $\eta := (P_T^\perp d_x, (d_z)_{S^c})$ be the component of d orthogonal to $T \times S$ and $\omega := d - \eta \in T \times S$ the component along it. With $\delta := \min(\lambda_1\delta_1, \lambda_2\delta_2)$, the bounds $\|M\|_F \leq \|M\|_*$ and $\|M\|_F \leq \|M\|_1$, and Eq. (89), we get

$$\|\eta\|_F \leq \|P_T^\perp d_x\|_F + \|(d_z)_{S^c}\|_F \leq \frac{D_x + D_z}{\delta} \leq \frac{\varepsilon}{\delta}.$$

Step 3: restricted injectivity bounds the component along the structure. By Eq. (89), $\|\mathcal{A}d\|_F \leq \sqrt{2\varepsilon}$. The operator $\mathcal{A}$ has norm at most $\sqrt{2}$, since $\|P_\Omega(u+v)\|_F \leq \|u\|_F + \|v\|_F$. Restricted injectivity applied to $\omega \in T \times S$ then gives the bound

$$\sigma\|\omega\|_F \leq \|\mathcal{A}\omega\|_F \leq \|\mathcal{A}d\|_F + \|\mathcal{A}\eta\|_F \leq \sqrt{2\varepsilon} + \sqrt{2}\frac{\varepsilon}{\delta}.$$

Step 4: combination of the bounds. The components ω and η are orthogonal. Using $(a+b)^2 \leq 2a^2 + 2b^2$ and $\varepsilon \leq \bar{\varepsilon}$, we obtain

$$\|d\|_F^2 = \|\omega\|_F^2 + \|\eta\|_F^2 \leq \frac{4\varepsilon + 4\varepsilon^2/\delta^2}{\sigma^2} + \frac{\varepsilon^2}{\delta^2} \leq \left[\frac{4}{\sigma^2} + \left(\frac{4}{\sigma^2} + 1\right)\frac{\bar{\varepsilon}}{\delta^2}\right]\varepsilon.$$

This is quadratic growth on the sublevel set $\{F \leq F^* + \bar{\varepsilon}\}$,

$$\frac{\mu}{2}\|(\tilde{x}, \tilde{z}) - (x^*, z^*)\|_F^2 \leq F(\tilde{x}, \tilde{z}) - F^*, \qquad \mu := 2\left[\frac{4}{\sigma^2} + \left(\frac{4}{\sigma^2} + 1\right)\frac{\bar{\varepsilon}}{\delta^2}\right]^{-1},$$

and μ depends only on δ_1, δ_2, σ, λ_1, λ_2, and $\bar{\varepsilon}$.

Applying this inequality to another minimizer gives $\varepsilon = 0$ and hence $d = 0$, so the minimizer is unique. Finally, substituting $\bar{z} := z - y$ writes the objective in the form Eq. (1) with $A = B = P_\Omega$, $f = \lambda_1\|\cdot\|_*$, and $g(\bar{z}) = \lambda_2\|\bar{z} + y\|_1$. The substitution is a translation and leaves all distances unchanged. Since $Z^* = \{(x^*, z^*)\}$ and $\|P_\Omega\| = 1$, we have $\|P_\Omega(\tilde{z} - z^*)\|_F \leq \|d\|_F$, so Assumption 1 holds with the same μ. ■

E.2 Verification of the quadratic growth condition

The constant μ from the proof certifies existence but is conservative in practice. We solve all 128 test instances to a KKT residual of at most 1.4×10^{-12} and compute the margins at the resulting optimum. Both regularity conditions hold in practice on every instance, with $\delta_1 \in [8.4 \times 10^{-4}, 0.11]$, $\delta_2 \in [3.3 \times 10^{-5}, 0.026]$, and $\sigma \in [0.050, 0.27]$. By Proposition 2, the minimizer is therefore unique on every test instance.

On every instance, the smallest margin exceeds the KKT residual of that instance by more than three orders of magnitude, so the conditions are resolved numerically. The small margin δ_2, which comes from residual entries close to the threshold λ_2, makes the formula give a value of μ many orders of magnitude below the empirical constant along the iterates. Along the iterates of proximal gradient and ADMM, the smallest ratio $2(F_y - F_y^*)/\|P_\Omega(z - z_y^*)\|^2$ of each instance lies in $[0.56, 0.77]$. Along the deployed iterates of the learned models of Appendices E.5 and E.9, the smallest ratio over all 128 instances lies in $[0.81, 1.90]$, depending on the model.

E.3 A common constant over instances

The constant in Proposition 2 depends on the instance through the margins δ_1, δ_2, and σ, while Theorem 2 uses one common constant μ for all instances. The two conditions are those under which robust PCA theory establishes exact recovery with high probability for random low-rank and sparse components [Candès et al., 2011, Chandrasekaran et al., 2011]. We therefore obtain a common constant on an event of instances and bound the probability of its complement.

Fix margins $\bar{\delta}_1, \bar{\delta}_2, \bar{\sigma} > 0$ before sampling, and let G be the set of data y for which a minimizer of F_y satisfies strict complementarity with $\delta_1 \geq \bar{\delta}_1$ and $\delta_2 \geq \bar{\delta}_2$ and restricted injectivity with $\sigma \geq \bar{\sigma}$. Since y has bounded support, the iterates are bounded for every parameter value: the learned x-update is projected onto a box, and the exact z-update satisfies $\|\tilde{z}\|_\infty \leq \|y\|_\infty + \|\tilde{x}\|_\infty$ (Appendix E.5). Hence there is a constant $\bar{r}$ with $F_y(\tilde{x}_y^k, \tilde{z}_y^k) - F_y^* \leq \bar{r}$ for all y, all parameters, and all k.

For $y \in G$, the proof of Proposition 2 with sublevel value $\bar{r}$ gives quadratic growth along all iterates with a constant $\mu_G > 0$ that depends only on $\bar{\delta}_1$, $\bar{\delta}_2$, $\bar{\sigma}$, λ_1, λ_2, and $\bar{r}$. Hence Theorem 1 holds for every $y \in G$ with the contraction rate ρ of μ_G. For $y \notin G$, the resulting gap is at most $\bar{r}$. Since the right-hand side of Theorem 1 is nonnegative, we obtain for every y

$$\tilde{F}_y^K - F_y^* \leq \rho^K \Phi_0 + \frac{1+\alpha}{\alpha} \sum_{k=1}^{K} \rho^{K-k} \Delta_y^k + \bar{r}\, \mathbb{1}[y \notin G].$$

The proof of Theorem 2 uses Theorem 1 only through this pointwise bound, so Eq. (13) holds with $\mu = \mu_G$ and the additional $\bar{r}\ \Pr(y \notin G)$ term.

Membership in G depends only on y and not on the network weights. If n independent instances all lie in G, then $\Pr(y \notin G) \leq \log(1/\beta')/n$ with probability at least $1 - \beta'$, since otherwise all n instances lie in G with probability at most $(1 - \log(1/\beta')/n)^n \leq \beta'$. A union bound with Theorem 2 gives a total confidence of $1 - \beta - \beta'$.

The margins in Appendix E.2 are computed after solving the test instances, so they illustrate the construction rather than certify G for fixed margins. As discussed, the explicit constant μ_G is many orders of magnitude below the empirical constant, and we use the empirical constant for the numerical evaluation in Appendix E.9.

E.4 Problem parametrization

Data generation Each instance of the dataset is a 100×100 matrix $y = P_\Omega(L+S)$ with a low-rank component L and a sparse corruption S. The low-rank component is generated as $L = UV^T$ with $U, V \in \mathbb{R}^{100 \times r}$ with rank $r = 5$, and i.i.d. entries drawn from $\mathcal{N}(0, 1/100)$. Each entry of S is nonzero independently with probability 0.05. Its nonzero entries are drawn uniformly from the interval $[-3s, 3s]$, where s is the empirical entrywise standard deviation of the low-rank components in the sampled batch, so that the corruption-to-signal ratio does not depend on n, m, or r. The observation set Ω contains each entry independently with probability 0.4, and the observations contain no additional noise.

The low-rank factors are sampled from a Gaussian, so the data y is not bounded by construction. We therefore define p_y as this generator with the additional constraint that it is conditioned on $\|y\|_\infty \leq c$ with $c = 20\sqrt{r/(nm)} \approx 0.45$, which is fixed from the entry-wise scale of the low-rank component before sampling. Sampling from the conditioned distribution amounts to rejecting draws with $\|y\|_\infty > c$. No instance in the training pools or in the test set is rejected in practice and the largest observed value is $\|y\|_\infty = 0.20$, so these samples are exact draws from the conditioned distribution.

Training and test data Since instances are cheap to generate, the networks for Table 6 are trained on a new dataset of instances at every optimization step instead of a fixed training set. The models used to evaluate the PAC-Bayes bound instead use a fixed training pool of N instances, so that the sample size in Theorem 2 is well defined (Appendix E.9). All methods are evaluated on the same 128 test instances, generated once with a fixed random seed that is independent of the training seeds.

Regularization parameters We use $\lambda_1 = 0.05$ and $\lambda_2 = 0.01$ for all instances chosen based on cross-validation to minimize the reconstruction error of the minimizer using the ground truth data without corruption.

Initialization All methods in our experiments are initialized with zero vectors $x^0 = z^0 = 0$.

E.5 Learned method

Architecture The learned scheme keeps the exact closed-form z-update and replaces only the x-update which is computationally expensive. At each outer iteration k, the network ϕ_{θ^k} performs

one proximal gradient step on the x-subproblem with a learned step size $\alpha_k > 0$ and a learned threshold vector $\tau_k \in \mathbb{R}_{>0}^{\min(n,m)}$, in the style of LISTA [Gregor and LeCun, 2010]:

$$\tilde{x}^{k+1} = \mathrm{SVT}_{\tau_k}\left(\tilde{x}^k - \alpha_k P_\Omega(\tilde{x}^k + \tilde{z}^k - y)\right), \qquad \tilde{z}^{k+1} = P_\Omega\, \mathcal{S}_{\lambda_2}(y - \tilde{x}^{k+1}).$$

Here, $\mathrm{SVT}_{\tau_k}(u) = U\,\mathrm{diag}\left(\max(\sigma_i - (\tau_k)_i, 0)\right) V^T$ is a spectral shrinkage map, which is the proximal operator of the nuclear norm when all thresholds are equal, requiring the singular value decomposition (SVD) of the input $u = U\,\mathrm{diag}(\sigma)V^T$ with singular values sorted in decreasing order. The proximal operator of the ℓ_1 norm $\mathcal{S}_{\lambda_2}$ is given by the entrywise soft-thresholding and is efficient to compute.

Unlike the classical SVT, in the learned method the threshold may differ across singular values, so that a layer has more flexibility and can shrink the leading singular values less than the tail. Entries outside the observed set Ω do not appear in the objective, and the z-update sets them to zero. Since the z-update is the exact minimizer, the error ζ_y^k in Eq. (4) is zero for every k.

Both α_k and τ_k are parameterized through a softplus function to ensure they are positive. Each outer iteration therefore has 101 parameters, 808 in total for $K = 8$, and costs a single SVD, the same as one iteration of each baseline. Beyond the trained depth, every iteration $k > 8$ reuses the parameters of the last trained iteration.

To keep the iterates bounded for every parameter, the x-update is followed by the entrywise projection onto the box $\{x \mid |x_{ij}| \le c\}$ with the same a priori constant $c = 20\sqrt{r/(nm)} \approx 0.45$ derived based on the problem generation from Appendix E.4. The z-update needs no projection, since the exact minimizer already satisfies $\|\tilde{z}^{k+1}\|_\infty \le \|y\|_\infty + c$, and projecting it would make the z-step inexact.

Training settings We train the networks greedily in a layer-to-layer fashion as in Algorithm 1. The training loss at iteration k is exactly the x-subproblem objective $\lambda_1\|\tilde{x}^{k+1}\|_* + \frac{1}{2}\|P_\Omega(y - \tilde{z}^k - \tilde{x}^{k+1})\|_F^2$, averaged over the batch, so training never uses the ground truth L and remains fully unsupervised.

The first iteration is initialized with $\alpha_1 = 1$ and $\tau_1 = \lambda_1 \mathbf{1}$, which reproduces one classical proximal gradient step with step size $1/L_x = 1$. Every later iteration is initialized from the parameters of the previous network using the greedily trained parameters as a warm start. Each iteration is trained for 400 Adam steps with learning rate 10^{-2} and batch size 16, so every iteration sees 6 400 fresh instances. We repeat the training with 5 random seeds and report the mean over the seeds.

E.6 Baseline methods

All baselines run in the same implementation and compute exactly one SVD per outer iteration, so that the outer iteration k is a common budget for all methods in Table 6.

FISTA The classical alternating scheme solves the x-subproblem with FISTA [Beck and Teboulle, 2009] with step size $1/L_x = 1$, since $\|P_\Omega\| = 1$. We apply warm-starting at the previous iterate, and computes the z-update in closed form. In Table 6 we truncate the FISTA iteration to use one inner iteration per outer iteration in order to match the budget of the other methods. With a single inner iteration there is no momentum step, so this column coincides with the proximal gradient method discussed below at the fixed step size $\alpha = 1$. For the time-to-match comparison in Table 5, we increase the number of inner iterations over the grid $\{1, 2, 3, 4, 5, 6, 8, 10, 15, 20, 30, 50\}$ and report the smallest budget at which FISTA reaches the objective of the learned scheme.

Tuned proximal gradient This proximal-gradient baseline applies the same update as the learned network, but with a fixed step size α and the uniform threshold $\alpha\lambda_1$ in every iteration. Comparing with a tuned step-size separates the gain from learning the update from the gain of a larger step size itself. We select α from the grid $\{1, 1.5, 1.8, 2, 2.2, 2.4, 2.5, 2.6, 2.8, 3\}$ and obtain the best performance when choosing $\alpha = 2.6$. For $\alpha > 2/L_x$, the method has no convergence guarantee, so this is an empirical ablation.

ADMM The problem (16) can be reformulated such that each iteration can be efficiently computed using ADMM [Boyd et al., 2011] which introduces an auxiliary copy w of x and moves the nuclear norm on it while adding an equality constraint such that this reformulation leads to the equivalent problem

$$\begin{array}{ll} \text{minimize} & \lambda_1\|w\|_* + \frac{1}{2}\|P_\Omega(y - x - z)\|_F^2 + \lambda_2\|z\|_1 \\ \text{subject to} & x = w, \end{array}$$

where x, z, and w are the optimization variables. With a chosen penalty parameter $\rho > 0$ and scaled dual variable u, the augmented Lagrangian is given by

$$\mathcal{L}_\rho(x, z, w, u) = \lambda_1 \|w\|_* + \frac{1}{2}\|P_\Omega(y - x - z)\|_F^2 + \lambda_2 \|z\|_1 + \frac{\rho}{2}\|x - w + u\|_F^2 - \frac{\rho}{2}\|u\|_F^2.$$

One ADMM iteration minimizes $\mathcal{L}_\rho$ over x, z, and w in turn which can all be solved in closed form and then updates the dual variable:

$$\begin{aligned}
x^{k+1} &= \big(P_\Omega(y - z^k) + \rho(w^k - u^k)\big) \oslash (P_\Omega \mathbf{1} + \rho \mathbf{1}), \\
z^{k+1} &= P_\Omega \, \mathcal{S}_{\lambda_2}(y - x^{k+1}), \\
w^{k+1} &= \mathrm{SVT}_{\lambda_1/\rho}(x^{k+1} + u^k), \\
u^{k+1} &= u^k + x^{k+1} - w^{k+1},
\end{aligned}$$

where $\oslash$ denotes entry-wise division and $\mathbf{1}$ is the all-ones matrix. The x-update is the closed-form minimizer of a separable quadratic form, the z-update is the same soft-thresholding step as in the baseline alternating scheme, and the w-update is the proximal operator of $(\lambda_1/\rho)\|\cdot\|_*$. Only the update over the auxiliary variable w requires an SVD, so one ADMM iteration costs one SVD, the same as one iteration of the learned scheme and the other baseline methods in the comparison.

All variables are initialized at zero. Since the variable z does not enter the constraint, the x- and z-updates form one pass of block minimization rather than a joint minimization over (x, z). We therefore use ADMM as an empirical baseline and do not rely on its convergence theory. We select the penalty parameter ρ from $\{0.02, 0.05, 0.1, 0.2, 0.3, 0.5, 1, 2, 5\}$ and obtain the best performance for $\rho = 0.2$. The two copies x and w agree only in the limit, and intermediate iterates of x are generically full rank. We therefore report the objective value at whichever of (x, z) and $(w, \mathcal{S}(w))$ attains the lower value, where $\mathcal{S}(w)$ is the exact z-update at w.

Hyperparameter selection Both α and ρ in the proximal gradient and ADMM method respectively are selected by the lowest objective after 8 outer iterations which matches the trained depth of the learned scheme. We then use the fixed found parameters for all steps in the outer scheme also beyond the tuning horizon. Both selected values lie in the interior of their grids.

Reference solution The relative optimality gap in Table 6 is computed as $(F - F^*)/F^*$, where F and F^* are the objective values averaged over the 128 test instances. We compute F^* with the classical alternating scheme with 30 warm-started FISTA iterations per subproblem and 60 outer iterations, that is, 1 800 SVDs per instance. The averaged objective agrees to all reported digits after 40, 50, and 60 outer iterations. Since the minimizer is unique on every test instance (Appendix E.2), the minimum F^* does not depend on the choice of solver.

As in the CT case, the reference solution does not match the ground truth: the relative error $\|x^* - L\|_F / \|L\|_F$ is 0.195, which is the recovery error of the convex formulation itself.

E.7 Computational resources

All robust matrix completion experiments run on an 11th Gen Intel Core i7-11850H CPU (8 cores, 16 threads, 2.50 GHz) and 32 GB of RAM, without GPU acceleration. We use PyTorch with its default CPU backend for the SVD. Training one model takes about 3 minutes, so the 5 seeds take about 15 minutes in total. Each PAC-Bayes evaluation in Appendix E.9, including training on the fixed pool, the reference solves, and the posterior Monte Carlo sampling, takes about 30 to 40 minutes on the same machine, and we run it for 5 different configurations.

E.8 Additional results

Here, we present additional results on timing, detailed optimality gaps and analyze the learned parameters of the scheme.

Timing results For the wall-clock times in Table 5, each method processes all 128 test instances as one batch. We perform one SVD before timing so that the initial library initialization does not fall into the first timed method. The objective itself is evaluated outside the timed region, since evaluating the nuclear norm costs another SVD. For the learned scheme, we report the mean and standard deviation over the 5 training seeds.

Table 5: Wall-clock time in seconds for the 128 test instances of robust matrix completion. Equal SVDs: FISTA at the same outer iteration k. Time to match: FISTA with the smallest number of SVDs (in parentheses) that reaches the objective of the learned scheme at k.

k	**Learned (ours)**	FISTA (equal SVDs)	FISTA (time to match)
1 *(trained)*	0.112 ± 0.002 s	0.116 s	0.335 s (3 SVDs)
4 *(trained)*	0.439 ± 0.003 s	0.442 s	2.141 s (20 SVDs)
8 *(trained)*	0.873 ± 0.003 s	0.878 s	4.355 s (40 SVDs)
16 *(extrapolated)*	1.746 ± 0.009 s	1.764 s	5.114 s (48 SVDs)
32 *(extrapolated)*	3.495 ± 0.012 s	3.460 s	6.935 s (64 SVDs)

The SVD dominates the cost of every method. One outer iteration takes about 0.11 s for the 128 instances, both for the learned scheme and for FISTA with one inner iteration. The learned update adds only a scalar multiplication with the learned step size and a thresholding step with a vector of 100 entries, so its overhead is negligible which is confirmed by the empirical results above. At equal computational budget, the wall-clock time is therefore proportional to the number of SVDs, and the gains in the objective decrease from Table 6 carry over directly to runtime.

The final time-to-match column in Table 5 shows how much the learned acceleration is worth for this problem. At the trained depth $k = 8$, FISTA needs 40 SVDs and 4.4 s to reach the objective that the learned scheme attains with 8 SVDs in 0.87 s, a $5\times$ overall reduction. The reduction is $3\times$ at $k = 1$ and $5\times$ at $k = 4$. For comparison, ADMM with the tuned $\rho = 0.2$ needs 17 iterations and 1.9 s to match the learned scheme at $k = 8$, which is still a $2.2\times$ reduction. Beyond the trained depth, the reduction relative to FISTA shrinks to $2.9\times$ at $k = 16$ and $2.0\times$ at $k = 32$. This is consistent with Table 6: once the learned scheme reuses the last trained parameters, its gap plateaus, and the classical methods catch up as also predicted by theory.

Relative optimality gap Table 6 lists the relative objective gap values of Fig. 4 at five different budgets. The FISTA method is truncated and uses one inner iteration per outer iteration and therefore coincides with the proximal gradient method at $\alpha = 1/L$ to match the computational budget. Beyond the trained depth, the gap of the learned scheme plateaus around 4×10^{-5} from $k \approx 14$ onward. In this regime the bound is dominated by the error term of the learned optimizer.

Table 6: Relative optimality gap $(F - F^\star)/F^\star$ for robust matrix completion problem at equal SVD budget k (one SVD per outer iteration for all four methods). Learned results are the mean $\pm$ standard deviation over 5 independent training runs.

Outer iterations k	**Learned (ours)**	ADMM ($\rho = 0.2$)	Prox-grad ($\alpha = 2.6$)	FISTA ($\alpha = 1/L$)
1 *(trained)*	$\mathbf{(3.46 \pm 0.01) \times 10^{-1}}$	7.6×10^{-1}	4.1×10^{0}	$\underline{5.0 \times 10^{-1}}$
4 *(trained)*	$\mathbf{(1.67 \pm 0.01) \times 10^{-2}}$	$\underline{7.5 \times 10^{-2}}$	1.3×10^{-1}	2.5×10^{-1}
8 *(trained)*	$\mathbf{(6.59 \pm 0.11) \times 10^{-4}}$	$\underline{9.9 \times 10^{-3}}$	1.4×10^{-2}	1.2×10^{-1}
16 *(extrapolated)*	$\mathbf{(3.66 \pm 0.53) \times 10^{-5}}$	7.3×10^{-4}	$\underline{1.4 \times 10^{-4}}$	2.7×10^{-2}
32 *(extrapolated)*	$(4.38 \pm 0.64) \times 10^{-5}$	$\mathbf{1.4 \times 10^{-5}}$	$\underline{2.3 \times 10^{-5}}$	6.6×10^{-4}

Learned parameters Figure 10 shows what the 808 learned parameters encode. The step size (left) grows from 1.28 in the first iteration to about 2.6 and then settles between 2.35 and 2.5, well above the classical $1/L$ and the $2/L$ theoretical limit for the convergence of the scheme. This roughly matches the chosen step size of the proximal gradient method where the tuned step size $\alpha = 2.6$ is selected for the best performance. However, the step size alone does not explain the gain in performance, since the proximal gradient method is substantially slower than our learned method as discussed in Table 6.

The thresholds explain the rest and give the network sufficiently large expressiveness to accelerate the convergence. In the first iterations, the network shrinks the first 5 singular values far less than the classical SVT which uses a fixed threshold parameter across all singular values and the remaining ones more. This is a learned soft rank cutoff at exactly the rank $r = 5$ of the ground truth, learned from the objective alone rather than hand-engineered. In later iterations, the thresholds approach

the classical SVT, as the iterates are already close to the minimizer of the objective. All learned parameters are nearly identical across the 5 seeds.

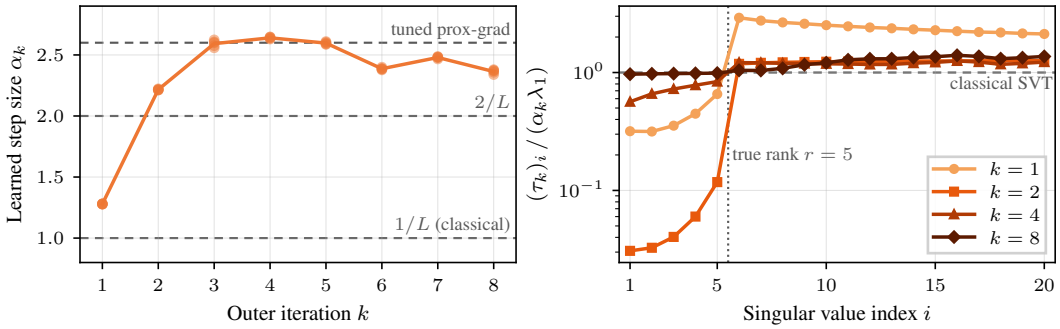


Figure 10: Learned parameters for robust matrix completion solver. Left: learned step size α_k for each of the 5 seeds (markers) and their mean (line). Right: learned threshold $(\tau_k)_i$ for the i-th singular value relative to the classical threshold $\alpha_k \lambda_1$ at the same step size, averaged over the seeds. We show the first 20 indices only. The thresholds of smaller singular values receive no gradient during training, since these singular values lie below the threshold, and, therefore, remain at their initial value.

E.9 Evaluation of the PAC-Bayes bound

We follow the procedure of Appendix C.3. The bound requires a fixed training sample, so we retrain the learned scheme with $K = 8$ on a fixed pool of $N = 2\,048$ instances with the settings of Appendix E.5, drawing minibatches from the pool. These models include the output projection onto the box $\{x \mid |x_{ij}| \leq 0.45\}$ (Appendix E.4). The projection is inactive on all iterates at the selected ς and becomes active only for $\varsigma \geq 2.3$. Since the z-update is exact, $\Delta_y^k = \xi_y^k$. We compute the reference minimizer of each x-subproblem with 40 FISTA iterations warm-started at the learned iterate and keep the best objective value, so that $\xi_y^k \geq 0$. This still remains an approximation of the true error since the reference is not exact but ensures non-negativity of the function gap.

The posterior risks use 4 sampled trajectories per value of ς on 48 training instances, and the evaluation of the left-hand side uses 4 trajectories on 48 held-out instances. We sample at 12 values of ς between 10^{-3} and 5. We further train variants with an explicit drift penalty $\lambda_{\mathrm{D}} \|\Theta^k - \Theta^{k-1}\|^2$ added to the greedy loss of iteration k, which directly targets the divergence in Theorem 2. Since Eq. (79) holds for all posteriors simultaneously, comparing these models afterwards requires no further union bound.

We use one constant $\mu = 0.564$ for all models. It is the smallest quadratic growth ratio on the 128 test instances over three sets of iterates: the deployed iterates of all ten learned models (Appendix E.5 and the models of this section), the iterates under $\mathbb{Q}$ for every sampled $\varsigma \leq 0.5$ with 8 trajectories, and the iterates of proximal gradient and ADMM. The minimum is attained along the classical iterates. Along the deployed iterates the ratio is at least 0.81, and under $\mathbb{Q}$ it is at least 0.66. For $\varsigma \geq 1$, the ratio under $\mathbb{Q}$ drops to 0.30, so we restrict the grid to $\varsigma \leq 0.5$. Since the same μ applies to all models, the rows of Table 8 share the same contraction rate ρ.

We use 60 equally spaced values of α in $[0.02, 0.98\,\alpha_{\max}]$ with $\alpha_{\max} = 0.32$ the largest value allowed by Eq. (7) for $\mu = 0.564$, and 300 log-spaced values of ς in $[10^{-3}, 0.5]$. Thus $G = 18\,000$ and $C = 17.2$.

Results Table 7 lists all quantities for the model with $\lambda_{\mathrm{D}} = 10^{-4}$. The right-hand side of Eq. (81) is 0.218, a factor 1.7 below the trivial bound of 0.375. Unlike for CT, the evaluated bound is therefore informative. It is still about 13 times larger than the estimated left-hand side of 0.017, and the complexity term accounts for 92% of it. The Monte Carlo standard errors of the posterior training risk and of the left-hand side are 2.0×10^{-3} and 2.3×10^{-3}, small compared with the complexity term.

The training and test risks at the trained weights agree to within 1% at every iteration, from 8.05×10^{-2} and 8.01×10^{-2} at $k = 1$ to 6.70×10^{-5} and 6.67×10^{-5} at $k = 8$. At the selected ς, the posterior risk is larger than the risk at the trained weights in late iterations, 2.9×10^{-3} instead of 6.7×10^{-5} at $k = 8$, but the discounted sum grows only by a factor of 2.0.

Table 7: Quantities entering Eq. (81) for robust matrix completion at the selected (α, ς). Model trained on a fixed pool of $N = 2\,048$ instances with drift penalty $\lambda_{\mathrm{D}} = 10^{-4}$. The constants μ and $\hat{M}$ are plug-in estimates, not certified constants.

Symbol	Meaning	Value
K	trained depth	8
N	training sample size	2 048
β	confidence level	0.05
C	$\log((2J+1)/\beta) + \log G$ with $J = 40$, $G = 18\,000$	17.2
μ	common quadratic growth constant (*estimated*)	0.564
ρ	contraction rate at the selected $\alpha = 0.02$	0.66
$\hat{M}$	largest observed Δ_y^k under $\mathbb{Q}$ (*estimated*)	0.13
D	drift $\sum_k \lVert\hat{\Theta}^k - \hat{\Theta}^{k-1}\rVert^2$	20.2
ς	selected prior standard deviation	0.103
KL	$D/(2\varsigma^2)$	951
$\sum_k \rho^{K-k} \hat{\Delta}^k$	training risk at $\hat{\Theta}$	8.9×10^{-3}
$\sum_k \rho^{K-k} \mathbb{E}_y[\Delta_y^k]$	test risk at $\hat{\Theta}$	8.9×10^{-3}
$\sum_k \rho^{K-k} \mathbb{E}_{\mathbb{Q}}[\hat{\Delta}^k]$	training risk under $\mathbb{Q}$	1.8×10^{-2}
$\frac{3\hat{M}}{4(1-\rho)}\sqrt{(\mathrm{KL}+C)/N}$	complexity term	0.200
	right-hand side of Eq. (81)	**0.218**
	trivial bound $\hat{M}(1-\rho^K)/(1-\rho)$	0.375
	test risk under $\mathbb{Q}$ (left-hand side, estimated)	1.7×10^{-2}

The full statement Eq. (13) is vacuous. For the model with $\lambda_{\mathrm{D}} = 10^{-4}$ and α selected to minimize Eq. (13) (yielding $\alpha = 0.14$, $\rho = 0.79$, $(1+\alpha)/\alpha = 8.2$), its right-hand side is 3.13, while the trivial bound is 4.50, the initial energy is $\mathbb{E}_y[\Phi_0] = 0.93$, and the measured gap is $\mathbb{E}_{\mathbb{Q}}\mathbb{E}_y[\tilde{F}_y^8 - F_y^*] = 5.1 \times 10^{-3}$. The bound thus exceeds the initial energy. Most of its right-hand side is the bound on the error sum, 0.37 at this α, multiplied by the factor $(1+\alpha)/\alpha$.

Table 8 shows how the drift penalty and the sample size change the bound. The penalty $\lambda_{\mathrm{D}} = 10^{-4}$ reduces the drift by a factor 2.7 and tightens the bound from 0.341 to 0.218, while the trained model also attains a lower optimality gap. Larger penalties tighten the bound further, down to 0.095, but at the cost of a less accurate model. With only $N = 128$ training instances, the bound is vacuous at the same drift. A union bound over all horizons $K \le 32$ (see Appendix C.2) adds $\log 32 \approx 3.5$ to $\mathrm{KL} + C \approx 968$ and leaves these values unchanged at the reported precision.

Table 8: Effect of the drift penalty λ_{D} and the sample size N on the bound for robust matrix completion. RHS is the right-hand side of Eq. (81) at the selected (α, ς), and LHS is its left-hand side estimated on held-out instances. The last column is the relative optimality gap of the trained model at $k = 8$, averaged per instance. All rows use the common constant $\mu = 0.564$ and $\rho = 0.66$.

N	λ_{D}	D	KL	RHS	trivial	LHS	gap at $k = 8$
2 048	0	53.6	2 524	0.341	0.375	1.5×10^{-2}	6.2×10^{-4}
2 048	10^{-4}	20.2	951	0.218	0.375	1.7×10^{-2}	4.3×10^{-4}
2 048	3×10^{-4}	8.5	418	0.135	0.335	1.3×10^{-2}	1.0×10^{-3}
2 048	10^{-3}	3.5	174	0.095	0.335	1.4×10^{-2}	3.1×10^{-3}
128	10^{-4}	19.4	915	0.954	0.446	1.6×10^{-2}	4.3×10^{-4}

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims of our paper are clearly listed out in the end of the introduction as a bullet point list with links to the exact sections that contain the corresponding theoretical and experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: There is a designated paragraph for limitations of our approach with a corresponding heading in Section 7. The limited empirical scope is discussed in the future work paragraph in the same section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Yes, the proofs of the claims can be found in Appendices B and C and the corresponding sections in the appendix for the CT reconstruction and matrix completion problem families.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the code to reproduce all experiments and figures in the supplement material and provide a link to a publicly accessible github repository in the abstract. Further, Appendix D.4 contains a description of all hyperparameters and design decisions for the experimental settings. The problem setup and baseline methods are described in Appendix D.2 and Appendix D.3. Accordingly, the methods for the matrix completion experiments are described in the sections Appendix E.4, Appendix E.5, and Appendix E.6.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The Mayo Low Dose CT Grand Challenge dataset is publicly accessible after registration but is distributed under CC BY-NC 4.0 / NIH Controlled Data Access, so we cannot redistribute it. We release all code, the pre-trained dictionary, and model checkpoints in the supplement and a public github repository linked in the abstract, together with instructions for reproducing the experiments once the dataset has been obtained.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The hyperparameters are described in Appendix D.4. Further details can be found in the code.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: Shaded bands in our figures show the $5-95$ percentile across the test instances. Tables report the mean and standard deviations over the test data slices. Further, we retrain the networks across 5 independent seeds and report the standard deviation of the objective gap in the result section.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware, per-epoch training time, and the total cost of the reported training run are reported in Appendix D.5 for CT reconstruction and in Appendix E.7 for robust matrix completion. The full research project, including preliminary experiments at smaller depths and architecture searches, required several times the compute of the reported runs, as noted in the "Previous experiments" paragraph.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our research conforms with the guidelines in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The contribution is theoretical and algorithmic: a worst-case bound for inexact alternating minimization, a greedy training scheme, and a PAC-Bayes generalization argument. The CT reconstruction experiments serve only as a numerical instance of the abstract objective Eq. (1). The paper makes no clinical claims, releases no diagnostic model, and shows no patient images.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our models are trained to solve large-scale optimization problems and do not have a high risk of misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have summarized the third-party assets used with corresponding license and reference in Table 3. The data for the second experiment on robust matrix completion is entirely synthetic.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The supplement contains the source code, the pre-trained dictionary, and a `README` that documents the directory layout, dependencies, and the commands needed to reproduce each table and figure. The code is also replicated on a publicly accessible github repository that is linked in th abstract.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our research does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No crowdsourcing or research with human subjects was part of this research so there is no requirement for an approval by an ethics board.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: In the paper we used LLMs only for standard components and editing/writing as well as a verifier for the proofs and theorem statements that are included in the paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.