Stable and Interpretable Unrolled Dictionary Learning

Anonymous authors
Paper under double-blind review

Abstract

The dictionary learning problem, representing data as a combination of a few atoms, has long stood as a popular method for learning representations in statistics and signal processing. The most popular dictionary learning algorithm alternates between sparse coding and dictionary update steps, and a rich literature has studied its theoretical convergence. The success of dictionary learning relies on access to a "good" initial estimate of the dictionary and the ability of the sparse coding step to provide an unbiased estimate of the code. The growing popularity of unrolled sparse coding networks has led to the empirical finding that backpropagation through such networks performs dictionary learning. We offer the theoretical analysis of these empirical results through PUDLE, a Provable Unrolled Dictionary LEarning method. We provide conditions on the network initialization and data distribution sufficient to recover and preserve the support of the latent code. Additionally, we address two challenges; first, the vanilla unrolled sparse coding computes a biased code estimate, and second, gradients during backpropagated learning can become unstable. We show approaches to reduce the bias of the code estimate in the forward pass, and that of the dictionary estimate in the backward pass. We propose strategies to resolve the learning instability by tuning network parameters and modifying the loss function. Overall, we highlight the impact of loss, unrolling, and backpropagation on convergence. We complement our findings through synthetic and image denoising experiments. Finally, we demonstrate PUDLE's interpretability, a driving factor in designing deep networks based on iterative optimizations, by building a mathematical relation between network weights, its output, and the training set.

1 Introduction

This paper considers the dictionary learning problem, namely representing data $\boldsymbol{x} \in \mathcal{X} \subset \mathbb{R}^m$ as linear combinations of a few atoms from a dictionary $\boldsymbol{D} \in \mathcal{D} \subset \mathbb{R}^{m \times p}$. Given $\boldsymbol{x}$ and $\boldsymbol{D}$, the problem of recovering the sparse (few non-zero elements) coefficients $\boldsymbol{z} \in \mathbb{R}^p$ is referred to as sparse coding, and can be solved through the lasso (Tibshirani, 1996) (also known as basis pursuit (Chen et al., 2001)):

$$\ell_{\boldsymbol{x}}(\boldsymbol{D}) \coloneqq \min_{\boldsymbol{z} \in \mathbb{R}^p} \ \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) + h(\boldsymbol{z}) \tag{1}$$

where $\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}\|_2^2$, and $h(\boldsymbol{z}) = \lambda\|\boldsymbol{z}\|_1$. Specifically, the problem aims to recover a dictionary $\boldsymbol{D}^*$ that generates the data, i.e.,

$$\boldsymbol{x} = \boldsymbol{D}^* \boldsymbol{z}^* \tag{2}$$

where $\boldsymbol{z}^*$ is sparse. Olshausen and Field (Olshausen & Field, 1997) introduced (2) in computational neuroscience as a model for how early layers of the visual cortex process natural images. Sparse coding has been widely studied and utilized in the statistics (Hastie et al., 2015) and signal processing communities (Elad, 2010). A few practical examples are denoising (Elad & Aharon, 2006), super-resolution (Yang et al., 2010), text processing (Jenatton et al., 2011), and classification (Mairal et al., 2009b), where it enables the extraction of sparse high-dimensional features representing data. Moreover, sparse modelling is ubiquitous in many other fields such as seismic signal processing (Nose-Filho et al., 2018), radar sensing for target detections (Bajwa et al., 2011), and astrophysics for image reconstruction from interferometric data (Akiyama et al., 2017). Furthermore, Cleary et al. (2017; 2021) use this model to learn a dictionary consisting of gene modules for efficient imaging transcriptomics.

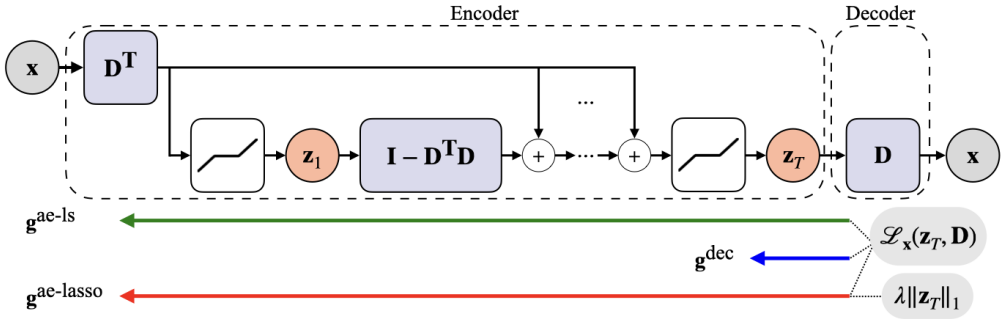


Figure 1: Provable unrolled dictionary learning (PUDLE): Unrolled network architecture with dictionary $\boldsymbol{D}$.

Sparse coding has been utilized to construct neural architectures through approaches such as sparse energy-based models (Ranzato et al., 2007; 2008) or recurrent sparsifying encoders (Gregor & LeCun, 2010). The latter has initiated a growing literature on constructing interpretable deep networks based on an approach referred to as algorithm unrolling (Hershey et al., 2014; Monga et al., 2019). Deep unrolled neural networks have gained popularity as inference maps in recent years due to their computational efficiency and their performance in various domains such as image denoising (Simon & Elad, 2019; Tolooshams et al., 2021a; 2020), super-resolution (Wang et al., 2015), medical imaging (Solomon et al., 2020), deblurring (Schuler et al., 2016; Li et al., 2020), radar sensing (Tolooshams et al., 2021b), and speech processing (Hershey et al., 2014).

Prior to the advent of unrolled networks, gradient-based dictionary learning relied on analytic gradients computed from the lasso given the sparse code. With unrolled networks, automatic differentiation (Baydin et al., 2018), referred to as backpropagation (LeCun et al., 2012) in the reverse-mode, gained attention for parameter estimation (Tolooshams et al., 2018). The automatic gradient is obtained by backpropagation through the algorithm used to estimate the code. Automatic differentiation in reverse and forward-mode (Franceschi et al., 2017) is used in other areas, e.g., hyperparameter selection (Feurer & Hutter, 2019), and in a more relevant context, in the seminal work of LISTA (Gregor & LeCun, 2010). Other works demonstrated empirically the convergence of ℓ_1-based dictionary learning by backpropagation through unrolled networks (Tolooshams et al., 2021a). Given finite computational power, Tolooshams et al. (2021a) convert sparse coding into an encoder by unrolling T iterations of ISTA (Daubechies et al., 2004; Blumensath & Davies, 2008), and attach to it a linear decoder for reconstructing. Unrolled networks obtained in this manner suffer from two important limitations.

First, the sparse coding step in the forward pass computes a biased estimate of the code. This results, in turn, in a biased estimate of the backward gradient and, hence, a degradation of dictionary recovery performance. Second, as studied recently (Malézieux et al., 2022), inaccuracies in the early iterations of the unrolled network make backpropagation unstable. We address both of these shortcomings in this paper. Moreover, while Malézieux et al. (2022) analyze the gradient computed by backpropagation through unrolled sparse-coding networks, there is no known theoretical analysis of how weight updates using this gradient impact the recovery of a ground-truth code $\boldsymbol{z}^*$, nor of their convergence to a ground-truth dictionary $\boldsymbol{D}^*$.

This paper proposes a Provable Unrolled Dictionary LEarning (PUDLE) (Figure 1). We aim to recover $\boldsymbol{D}^*$ by training the network using backpropagation with a learning rate of η. Three different choices affect the gradient: the number of unrolled iterations, the loss, and whether one backpropagates through the decoder only or through both the encoder and decoder. We highlight the impact of such choices on the convergence of the training algorithm. Backpropagation through the decoder results in the analytic gradient $\boldsymbol{g}_t^{\text{dec}}$ using the code estimate $\boldsymbol{z}_t$. The gradients $\boldsymbol{g}_t^{\text{ae-lasso}}$ and $\boldsymbol{g}_t^{\text{ae-ls}}$ are computed by backpropagation through the autoencoder using the lasso and least-squares objectives, respectively (Algorithm 2). We compare the gradients with the classical gradient-based alternating-minimization algorithm for dictionary learning (Chatterji & Bartlett, 2017) (i.e., cycling between sparse coding and dictionary update steps using the analytic gradient $\hat{\boldsymbol{g}}$ (Algorithm 1)), and provide a theoretical analysis of gradient-based recovery of the dictionary $\boldsymbol{D}^*$. We provide sufficient conditions under which the gradient computation, hence the learning, is stable. Additionally, we show how using the reconstruction loss with backpropagation not only does not suffer from backpropagated instability but also ameliorates the propagation of the forward pass bias into the gradient estimate from the backward pass. Finally, we demonstrate the interpretability of the unrolled network. Our contributions are:

- **Unrolled sparse coding** Unlike prior work (Malézieux et al., 2022) that studies the ability of sparse coding to recover the *solution of the lasso* (1) given the current estimate of the dictionary (we call this local estimation), we study unrolled sparse coding for recovery of the *true generating code in* (2) (we call this global estimation). We provide sufficient conditions on the network and data distributions such that the forward pass recovers (Theorem 4.1) and preserves (Theorem 4.2) the correct code support. Assuming support identification, we show the linear convergence of the code estimated through the unrolled iterations to the solution of the lasso (Theorem 4.3). We provide an explicit code expression at unrolled layer t and its error with respect the ground-truth code $\boldsymbol{z}^*$; we highlight the biased estimate of the code when the forward pass strictly solves lasso (Theorems 4.4 and 4.5). Moreover, in a more general scenario, we show that the error in the code estimate is upper bounded by two terms, i.e., one associated with the dictionary error and the other to the bias of the estimate of code amplitude, due to ℓ_1-based optimization (Theorem 4.6). The latter highlights that vanilla lasso (ℓ_1-based) sparse coding computes a biased estimate of codes, and below we discuss strategies to either alleviate this bias in the forward pass or mitigate its propagation into the backward pass for dictionary learning.
- **Mitigation of coding bias propagation into dictionary learning** We study gradient estimation for dictionary learning in PUDLE. We decompose the upper bound on the gradient errors compared to the gradient direction to recover $\boldsymbol{D}^*$ into terms involving the current dictionary error, the bias of the code estimate, and the lasso loss used to compute the gradient. We show that using only the reconstruction loss while backpropagating (i.e., $\boldsymbol{g}_t^{\text{ae-ls}}$) results in the vanishing of the upper bound due to the usage of lasso loss. This means that given fixed λ, $\boldsymbol{g}_t^{\text{ae-ls}}$ ameliorates the propagation of the forward pass bias into the backward pass. Specifically, we show that $\boldsymbol{g}_t^{\text{ae-ls}}$ is a better estimator of the direction to recover $\boldsymbol{D}^*$ than $\boldsymbol{g}_t^{\text{dec}}$ and $\boldsymbol{g}_t^{\text{ae-lasso}}$. Hence, weight updates using $\boldsymbol{g}_t^{\text{ae-ls}}$ converges to a closer neighbourhood of $\boldsymbol{D}^*$ (Theorem 4.10). In a supervised image denoising task, we show that the advantage of $\boldsymbol{g}_t^{\text{ae-ls}}$ goes beyond dictionary learning; $\boldsymbol{g}_t^{\text{ae-ls}}$ results in better image denoising compared to $\boldsymbol{g}_t^{\text{dec}}$. Furthermore, our network outperforms the sparse coding scheme in NOODL, a state-of-the-art online dictionary learning algorithm (Rambhatla et al., 2018) (Table 1). Moreover, we show that the bias in the estimate of $\boldsymbol{D}^*$ vanishes as $\lambda_t = \lambda\nu^t$ (with $0 < \nu < 1$) decays within the forward unrolled layers (Figure 16). This strategy, supported by Theorem 4.4, results in an unbiased estimate of the code $\boldsymbol{z}^*$ and recovery of $\boldsymbol{D}^*$ (Theorem 4.11).
- **Stability of unrolled learning** Our approach to resolve the instability issue of backpropagation in unrolled networks is two-fold. First, we show that under proper dictionary initialization, the instability of the gradient $\boldsymbol{g}_t^{\text{ae-lasso}}$ computation, studied by Malézieux et al. (2022), as T increases is resolved. We give a condition under which the code support is identified and recovered after one iteration and, hence, gradient computation stays stable. Second, in the absence of support identification in early iterations, we propose to use the gradient $\boldsymbol{g}_t^{\text{ae-ls}}$ which resolves the stability issue introduced by lasso loss in the backward pass. We highlight this stability through image denoising training without gradient explosion (Figure 6).
- **Interpretable sparse codes and dictionary** Prior work has discussed algorithm unrolling for designing interpretable deep architectures based on optimization models (Monga et al., 2019), or interpretability of sparse representations in dictionary learning models (Kim et al., 2010). However, there is no known work to mathematically characterize the interpretability of unrolled network architectures. In this regard, first, we construct a mathematical relation between learned weights (dictionary) at gradient convergence and the training data (Theorem 5.1). Second, we relate the inferred representation/reconstruction of test examples to the training data. We highlight several interpretable features of the unrolled dictionary learning network. Specifically, we perform analysis that provide insights into questions such as *why am I learning a particular feature in the dictionary?* or *from what part of the training set or an image I am learning that feature?* (Figure 8). Moreover, we provide an explanation of the relation between the new test image denoised/reconstructed through the network and the training dataset. The model provides insights on *how training images are used to reconstruct a new test image* (Figure 9) or *how the test image picks up training images that have a similar representation to itself to reconstruct* (Figure 10).

2 Related Works

There is vast literature on the theoretical convergence of dictionary learning. Spielman et al. (2012) proposed a factorization method to recover the dictionary in the undercomplete setting (i.e., $p \leq m$). Barak et al. (2015) proposed to solve dictionary learning via sum-of-squares semidefinite program. K-SVD (Aharon et al., 2006)

and MOD (Engan et al., 1999) are popular greedy approaches. Alternating-minimization-based methods have been used extensively in theory and practice (Jain et al., 2013; Agarwal et al., 2014; Arora et al., 2014).

Recent work has incorporated gradient-based updates into alternating minimization (Chatterji & Bartlett, 2017; Arora et al., 2015; Rambhatla et al., 2018). Chatterji & Bartlett (2017) provided a finite sample analysis and convergence guarantees when updating the dictionary using the analytic gradient. Arora et al. (2015) proposed neurally plausible sparse coding approaches with analytic gradients. Another work focused on online dictionary learning (Mairal et al., 2009a) with an unbiased gradient updates (Rambhatla et al., 2018). Arora et al. (2015) discussed methods to reduce the bias of dictionary estimate, and Rambhatla et al. (2018) showed how to reduce bias in code and dictionary estimates. A common feature in the above-mentioned work is the use of analytic gradients, i.e., explicitly designing gradient updates independent of the sparse coding step and not utilizing automatic gradients with deep learning optimizers. A theoretical analysis of backpropagation for dictionary learning exists only for shallow autoencoders (Rangamani et al., 2018; Nguyen et al., 2019).

The theoretical analysis of unrolled neural networks has mainly analyzed the convergence speed of variants of LISTA (Gregor & LeCun, 2010), where the focus is on sparse coding (i.e., the encoder) not dictionary learning (Sprechmann et al., 2012; Xin et al., 2016; Moreau & Bruna, 2017; Giryes et al., 2018; Chen et al., 2018; Liu & Chen, 2019; Ablin et al., 2019). Moreau & Bruna (2017) showed that upon successful factorization of the Gram matrix of the dictionary within layers, the network achieves accelerated convergence. Giryes et al. (2018) examined the tradeoffs between reconstruction accuracy and convergence speed of LISTA. Moreover, Chen et al. (2018) studied the learning dynamics of the weights and biases of unrolled-ISTA and proved that it achieves linear convergence. Follow-up works investigated the dynamics of step size in a recursive sparse coding encoder (Liu & Chen, 2019; Ablin et al., 2019). Ablin et al. (2019) minimized the lasso through backpropagation but still assumed the knowledge of the dictionary at the decoder.

Ablin et al. (2020) compared analytic and automatic gradient estimators of min-min optimizations with smooth and differentiable functions. Moreover, Malézieux et al. (2022) studied the stability of gradient approximation in the early regime of unrolling for dictionary learning. Unlike our work, where we evaluate the gradients for model recovery, Ablin et al. (2020) and Malézieux et al. (2022) studied the asymptotic gradient errors locally in each step of an alternating minimization and did not provide errors concerning $\boldsymbol{z}^*$ or $\boldsymbol{D}^*$.

3 Preliminaries

Given n independent samples, dictionary learning aims to minimize the empirical risk, i.e.,

$$\min_{\boldsymbol{D}\in\mathcal{D}} \ \mathcal{R}_n(\boldsymbol{D}) \quad \text{with} \quad \mathcal{R}_n(\boldsymbol{D}) \triangleq \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \ell_{\boldsymbol{x}^i}(\boldsymbol{D}) \tag{3}$$

where $\lim_{n\to\infty} \mathcal{R}_n(\boldsymbol{D}) = \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\ell_{\boldsymbol{x}}(\boldsymbol{D})\right]$ a.s. To prevent scaling ambiguity between the code $\boldsymbol{z}$ and dictionary $\boldsymbol{D}$, it is common to constrain the norm of the dictionary columns. Hence, we define the set of feasible solutions for the dictionary as $\mathcal{D} \triangleq \{\boldsymbol{D} \in \mathbb{R}^{m\times p} \text{ s.t. } \forall j \in \{1, 2, \ldots, p\},\ \|\boldsymbol{D}_j\|_2^2 \leq 1\}$. We can project estimates of $\boldsymbol{D}$ onto the feasible set by performing $\boldsymbol{D}_j \leftarrow {}^{1}\!/_{\max(\|\boldsymbol{D}_j\|_2,1)}\boldsymbol{D}_j$, either at every update or at the end of training. We assume certain properties on the data, specifically its domain (Assumption 3.1), energy (Assumption 3.2), code distribution (Assumption 3.3), and generating dictionary (Assumption 3.4).

Assumption 3.1 (Domain signals)**.** *$\mathcal{X}$ and $\mathcal{D}$ are both compact convex sets.*

Assumption 3.2 (Bounded signals)**.** *$\exists\ M > 0$ s.t. $\|\boldsymbol{x}\|_2 < M\ \forall \boldsymbol{x} \in \mathcal{X}$.*

Assumption 3.3 (Code distribution)**.** *The code $\boldsymbol{z}^*$ is at most s-sparse with the support $S^* = supp(\boldsymbol{z}^*)$. Each element in S^* is chosen from the set $[1, p]$, uniformly at random without replacement. $p_i = P(i \in S^*) = \Theta(s/p)$, and $p_{ij} = P(i, j \in S^*) = \Theta(s^2/p^2)$. Given the support, $\boldsymbol{z}^*_S$ is i.i.d, has symmetric probability distribution density function, $\mathbb{E}[\boldsymbol{z}^*_{(i)} \mid i \in S^*] = 0$ and $\mathbb{E}[\boldsymbol{z}^*_{(S)}\boldsymbol{z}^{*T}_{(S^*)} \mid S^*] = \boldsymbol{I}$. Moreover, the non-zero entries of the code are sub-Gaussian and lower bounded, i.e., for $i \in S^*$, $|\boldsymbol{z}^*_{(i)}| \geq C_{\min}$ where $C_{\min} \leq 1$.*

Assumption 3.4 (Generating dictionary)**.** *$\boldsymbol{D}^*$ is μ-incoherent (see Definition A.1) where $\mu = \mathcal{O}(\log(m))$. $\boldsymbol{D}^*$ is unit-norm columns matrix ($\|\boldsymbol{D}^*_i\|_2 = 1$), $\|\boldsymbol{D}^*\|_2 = \mathcal{O}(\sqrt{p/m})$, and $p = \mathcal{O}(m)$.*

To achieve model recovery using gradient descent, we assume an appropriate dictionary initialization, i.e.,

Assumption 3.5 (Dictionary closeness)**.** *The initial dictionary $\boldsymbol{D}^{(0)}$ is $(\delta_0, 2)$-close to $\boldsymbol{D}^*$ (see Definition A.2). The dictionary closeness at every update is denoted by $\|\boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*\|_2 \leq \delta_l \ \forall j$. Furthermore, $\delta_l = \mathcal{O}^*(1/\log p)$.*

Arora et al. (2015) proposed a dictionary initialization method offering $(\delta, 2)$-close to $\boldsymbol{D}^*$ for $\delta = O^*(1/\log m)$. The method is based on pairwise reweighting of samples $\{\boldsymbol{x}^i\}_{i=1}^n$ from the generative model (2), and does not require access to $\boldsymbol{D}^*$. In addition, Rambhatla et al. (2018) utilize dictionary closeness assumptions and such dictionary initialization for their theoretical analysis. Moreover, Agarwal et al. (2017) proposed a clustering approach to find a close initial estimate of the dictionary.

Given the μ-incoherence of $\boldsymbol{D}^*$ (Assumption 3.4) and δ_l-closeness of the dictionary, $\boldsymbol{D}^{(l)}$ is μ_l-incoherent, i.e.,

Lemma 3.1 (μ_l-incoherent)**.** *$\boldsymbol{D}^{(l)}$ is μ_l-incoherent where $\mu_l = \mu + 2\sqrt{m}\delta_l$.*

The recurrent encoder and decoder, which perform the computations shown in Algorithm 2, use the loss $\mathcal{L}$ and proximal operator $\mathcal{P}_b(v) \triangleq \text{sign}(v)\max(|v| - b, 0)$ for the ℓ_1 norm $h\colon \mathbb{R}^p \to \mathbb{R}$. The encoder implements ISTA (Daubechies et al., 2004; Blumensath & Davies, 2008) with step size α, assumed to be less than $1/\sigma_{\max}^2(\boldsymbol{D})$. With infinite encoder unrolling, the encoder's output is the solution to the lasso (1), following the optimality condition (Lemma A.3) where we denote $f_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) \triangleq \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) + h(\boldsymbol{z})$. One immediate observation is that $\lambda \geq \|\boldsymbol{D}^{\mathrm{T}}\boldsymbol{x}\|_\infty \Leftrightarrow \{\boldsymbol{0}\} \in \arg\min f_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$. We assume $\lambda < \|\boldsymbol{D}^{\mathrm{T}}\boldsymbol{x}\|_\infty$. We specify in Theorem 4.1 and Theorem 4.2 the conditions on λ at every encoder iteration to ensure support recovery and its preservation through the encoder. In case of a constant λ across encoder iterations while using $\boldsymbol{D}^*$ as the dictionary (i.e., sparse coding using ℓ_1 norm), the network recovers a biased code $\hat{\boldsymbol{z}}^*$. We denote this amplitude error in the code by $\hat{\delta}^* \triangleq \|\hat{\boldsymbol{z}}^* - \boldsymbol{z}^*\|_2$ which is small and goes to zero with λ decaying through the encoder.

In addition, we assume the solution to (1) is unique; sufficient conditions for uniqueness in the overcomplete case (i.e., $p > m$) are extensively studied in the literature (Wainwright, 2009; Candès & Plan, 2009; Tibshirani, 2013). Tibshirani (2013) discussed that the solution is unique with probability one if entries of $\boldsymbol{D}$ are drawn from a continuous probability distribution (Tibshirani, 2013) (Assumption 3.6). This assumption implies that $\boldsymbol{D}_S^{\mathrm{T}}\boldsymbol{D}_S$ is full-rank. We argue that as long as the data $\boldsymbol{x} \in \mathcal{X}$ are sampled from a continuous distribution, this assumption holds for the entire learning process. The preservation of this property is guaranteed at all iterations of the alternating minimization proposed in (Agarwal et al., 2014). Moreover, this assumption has been previously considered in analyses of unrolled sparse coding networks (Ablin et al., 2019; Malézieux et al., 2022) and can be extended to ℓ_1-based optimization problems (Tibshirani, 2013; Rosset et al., 2004).

Assumption 3.6 (Lasso uniqueness)**.** *The entries of the dictionary $\boldsymbol{D}$ are continuously distributed. Hence, the minimizer of (1) is unique, i.e., $\hat{\boldsymbol{z}} = \arg\min f_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ with probability one.*

Lemma 3.2 states the fixed-point property of the encoder recursion (Parikh & Boyd, 2014). Given the definitions for *Lipschitz* and *Lipschitz differentiable* functions, (Definitions A.3 and A.4), the loss $\mathcal{L}$ and function h satisfy following *Lipschitz* properties.

Lemma 3.2 (Fixed-point property of lasso)**.** *Given Assumption 3.6, we have $\boldsymbol{0} \in \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \partial h(\hat{\boldsymbol{z}})$. The minimizer is a fixed-point of the mapping, i.e., $\hat{\boldsymbol{z}} = \mathcal{P}_{\alpha\lambda}(\hat{\boldsymbol{z}} - \alpha\nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})) = \Phi(\hat{\boldsymbol{z}})$ (Parikh & Boyd, 2014).*

Lemma 3.3 (Lipschitz differentiable least squares)**.** *Given $\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}\|_2^2$, $\mathcal{D}$, and Assumption 3.2, the loss is Lipschitz differentiable. Let L_1 and L_2 denote the Lipschitz constants of the first derivatives $\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ and $\nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$, L_{11} and L_{21} the Lipschitz constants of the second derivatives $\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ and $\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$, all w.r.t $\boldsymbol{z}$. Let $\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ be L_{1D}-Lipschitz w.r.t $\boldsymbol{D}$, and we denote the Lipschitz constant of $\nabla_{11}\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ and $\nabla_{21}\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ w.r.t to $\boldsymbol{D}$ by L_{11D} and L_{21D}, respectively.*

Lemma 3.4 (Lipschitz proximal)**.** *Given $h(\boldsymbol{z}) = \lambda\|\boldsymbol{z}\|_1$, its proximal operator has bounded sub-derivative, i.e., $\|\partial\mathcal{P}_\lambda(\boldsymbol{z})\|_2 \leq c_{prox}$.*

4 Unrolled Dictionary Learning

The gradients defined in PUDLE (Algorithm 2) can be compared against the local direction at each update of classical alternating-minimization (Algorithm 1). Assuming there are infinite samples, i.e.,

$$\text{Best local direction}: \quad \hat{\boldsymbol{g}} \triangleq \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^n \nabla_2\mathcal{L}_{\boldsymbol{x}^i}(\hat{\boldsymbol{z}}^i, \boldsymbol{D}) = \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\nabla_2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})\right] \tag{4}$$

Algorithm 1: Classical alternating-minimization-based dictionary learning using lasso (1).

Initialize: Samples $\{\boldsymbol{x}^i\}_{i=1}^n \in \mathcal{X}$, initial dictionary $\boldsymbol{D}^{(0)}$
Repeat: $l = 0, 1, \ldots,$ number of epochs
Sparse coding step: $\boldsymbol{z}^{i(l)} = \arg\min_{\boldsymbol{z}} \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}, \boldsymbol{D}^{(l)}) + h(\boldsymbol{z}), \quad (\text{for } i \in [1, n])$
Dictionary update: $\boldsymbol{D}^{(l+1)} = \boldsymbol{D}^{(l)} - \eta \hat{\boldsymbol{g}}^{(l)}$ where $\hat{\boldsymbol{g}}^{(l)} \triangleq \frac{1}{n}\sum_{i=1}^n \nabla_2 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}^{i(l)}, \boldsymbol{D}^{(l)})$

Algorithm 2: PUDLE: Provable unrolled dictionary learning framework.

Initialize: Samples $\{\boldsymbol{x}^i\}_{i=1}^n \in \mathcal{X}$, initial dictionary $\boldsymbol{D}^{(0)}$, and $\boldsymbol{z}_0 = \boldsymbol{0}$.
Repeat: $l = 0, 1, \ldots,$ number of epochs
Forward pass: (for $i \in [1, n]$)

$$\begin{aligned} &\text{Encoder:} \quad \boldsymbol{z}_{t+1}^{i(l)} = \Phi(\boldsymbol{z}_t^{i(l)}, \boldsymbol{D}^{(l)}) = \mathcal{P}_{\alpha\lambda}(\boldsymbol{z}_t^{i(l)} - \alpha \nabla_1 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_t^{i(l)}, \boldsymbol{D}^{(l)})) \text{ (repeat for } T) \\ &\text{Decoder:} \quad \hat{\boldsymbol{x}}^{i(l)} = \boldsymbol{D}^{(l)} \boldsymbol{z}_T^{i(l)} \end{aligned} \tag{6}$$

Backward pass: $\boldsymbol{D}^{(l+1)} = \boldsymbol{D}^{(l)} - \eta \boldsymbol{g}_T^{(l)}$ where $\boldsymbol{g}_T^{(l)}$ is either of

$$\begin{aligned} \boldsymbol{g}_T^{(l)\ \text{dec}} &\triangleq \tfrac{1}{n}\textstyle\sum_{i=1}^n \nabla_2 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_T^{i(l)}, \boldsymbol{D}^{(l)}) \\ \boldsymbol{g}_T^{(l)\ \text{ae-lasso}} &\triangleq \tfrac{1}{n}\textstyle\sum_{i=1}^n \nabla_2 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_T^{i(l)}, \boldsymbol{D}^{(l)}) + \boldsymbol{J}_T^{i(l)+} \left(\nabla_1 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_T^{i(l)}, \boldsymbol{D}^{(l)}) + \partial h(\boldsymbol{z}_T^{i(l)})\right) \\ \boldsymbol{g}_T^{(l)\ \text{ae-ls}} &\triangleq \tfrac{1}{n}\textstyle\sum_{i=1}^n \nabla_2 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_T^{i(l)}, \boldsymbol{D}^{(l)}) + \boldsymbol{J}_T^{i(l)+} \nabla_1 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_T^{i(l)}, \boldsymbol{D}^{(l)}) \end{aligned} \tag{7}$$

See Definition 4.1 for $\boldsymbol{J}_T^+$.

where $\hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{z} \in \mathbb{R}^p} \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) + h(\boldsymbol{z})$. Additionally, to assess the estimators for model recovery, hence dictionary learning, we compare them against gradient pointing towards $\boldsymbol{D}^*$, namely

$$\text{Desired global gradient for } \boldsymbol{D}^*: \quad \boldsymbol{g}^* \triangleq \lim_{n\to\infty} \tfrac{1}{n}\textstyle\sum_{i=1}^n \nabla_2 \mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}^{i*}, \boldsymbol{D}) = \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\nabla_2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})\right]. \tag{5}$$

To see why the above is the desired direction, $(\boldsymbol{z}^*, \boldsymbol{D}^*)$ is a critical point of the loss $\mathcal{L}$ which reaches zero for data following the model (2). Hence, to reach $\boldsymbol{D}^* \in \arg\min_{\boldsymbol{D}\in\mathcal{D}} \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}[\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})]$, we move towards the direction minimizing the loss in expectation. Specifically, using the gradient $\nabla_2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) = -(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}^*)\boldsymbol{z}^{*\mathrm{T}} = (\boldsymbol{D} - \boldsymbol{D}^*)\boldsymbol{z}^*\boldsymbol{z}^{*\mathrm{T}}$ as a descent direction, we move from $\boldsymbol{D}$ toward $\boldsymbol{D}^*$ modulo the code presence matrix $\boldsymbol{z}^*\boldsymbol{z}^{*\mathrm{T}}$. Given these directions, we analyze the error of the gradients $\boldsymbol{g}_t^{\text{dec}}$, $\boldsymbol{g}_t^{\text{ae-lasso}}$, and $\boldsymbol{g}_t^{\text{ae-ls}}$ assuming infinite samples. In local analysis, we compare the code and gradient estimates to the lasso optimization in each update of the alternating minimization. In global analysis, we evaluate the performance in recovery of the ground-truth code $\boldsymbol{z}^*$ and the dictionary $\boldsymbol{D}^*$. In this regard, we first study the forward pass.

4.1 Forward pass

We show convergence results in the forward pass for $\boldsymbol{z}$ and the Jacobian, i.e.,

Definition 4.1 (Code Jacobian)**.** *Given $\boldsymbol{D}$, the Jacobian of $\boldsymbol{z}_t$ is defined as $\boldsymbol{J}_t \triangleq \frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{D}}$ with adjoint $\boldsymbol{J}_t^+$.*

The forward pass analyses give upper bounds on the error between $\boldsymbol{z}_t$ and $\hat{\boldsymbol{z}}$ and the error between $\boldsymbol{J}_t$ and $\hat{\boldsymbol{J}}$ as a function of unrolled iterations t. We define $\hat{\boldsymbol{J}}$ as following: considering the function $\boldsymbol{z} \to \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) + h(\boldsymbol{z})$, $\hat{\boldsymbol{z}}(\boldsymbol{D})$ is its minimizer and $\hat{\boldsymbol{J}} = \frac{\partial \hat{\boldsymbol{z}}(\boldsymbol{D})}{\partial \boldsymbol{D}}$. We will require these errors in Section 4.2, where we analyze the gradient estimation errors. Similar to (Chatterji & Bartlett, 2017), the error associated with $\boldsymbol{g}_t^{\text{dec}}$ depends on the code convergence. Unlike $\boldsymbol{g}_t^{\text{dec}}$, the convergence of backpropagation with gradient estimates $\boldsymbol{g}_t^{\text{ae-lasso}}$ and $\boldsymbol{g}_t^{\text{ae-ls}}$ relies on the convergence properties of the code *and* the Jacobian (Ablin et al., 2020). Forward-pass theories are based on studies by Gilbert (1992) on the convergence of variables and their derivatives in an iterative process governed by a smooth operator (Gilbert, 1992). Moreover, Hale et al. (2007) studied the convergence analysis of fixed point iterations for ℓ_1 regularized optimization problems (Hale et al., 2007).

Support recovery and preservation We re-state a result from (Hale et al., 2007) on support selection.

Proposition 4.1 (Finite-iteration support selection)**.** *Given Assumption 3.6, let* $\hat{\boldsymbol{z}} = \arg\min f_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ *with* $S \triangleq supp(\hat{\boldsymbol{z}})$*. There exists a* $B > 0$ *such that* $supp(\boldsymbol{z}_t) = S, \forall t > B$*.*

This means the unrolled encoder identifies the support in finite iterations. Support recovery in finite iterations has been studied in the literature for LISTA (Chen et al., 2018), Step-LISTA (Ablin et al., 2019), and shallow autoencoders (Arora et al., 2015; Rangamani et al., 2018; Nguyen et al., 2019; Tolooshams et al., 2020). We show that under proper initialization of the dictionary, the encoder achieves linear convergence. Arora et al. (2015) discussed some appropriate initialization which is used by Rambhatla et al. (2018). Given initial closeness δ_0, the encoder selects and recovers the correct signed support of the code with high probability in one iteration $B = 1$ (Theorem 4.1), and the iterations preserve the correct support (Theorem 4.2). In spite of slow convergence of ISTA Liang et al. (2014), support recovery after one iteration in unrolled networks is studied in the literature (Arora et al., 2015; Rambhatla et al., 2018; Chen et al., 2018; Nguyen et al., 2019).

Theorem 4.1 (Forward pass support recovery)**.** *Given Assumptions 3.3 and 3.4, suppose* $\boldsymbol{D}^{(l)}$ *is* $\delta_l = \mathcal{O}^*(1/\sqrt{\log p})$ *close to* $\boldsymbol{D}^*$*. If* $s = \mathcal{O}^*(\sqrt{m}/\mu \log m)$*, and* $\mu = \mathcal{O}(\log m)$*, then with high probablity of at least* $1 - \epsilon^{(l)}_{supp\text{-}rec}$*, the choice of* $\lambda_0 = C_{\min}/4$ *recovers the support of the code* $\boldsymbol{z}^*$ *in one encoder iteration, i.e.,* $sign(\mathcal{S}_{\alpha\lambda_0}(\alpha \boldsymbol{D}^{(l)T}\boldsymbol{x}) = sign(\boldsymbol{z}^*)$*, where* $\epsilon^{(l)}_{supp\text{-}rec} = 2p\exp\left(-\frac{C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}\right)$*.*

Theorem 4.2 (Forward pass support preservation)**.** *Given Assumptions 3.3 and 3.4, suppose* $\boldsymbol{D}^{(l)}$ *is* $\delta_l = \mathcal{O}^*(1/\log p)$ *close to* $\boldsymbol{D}^*$*. If* $s = \mathcal{O}^*(\sqrt{m}/\mu \log m)$*,* $\mu = \mathcal{O}(\log m)$*, and the regularizer and step size are chosen such that* $\lambda^{(l)}_t = \frac{\mu_l}{\sqrt{m}}\|\boldsymbol{z}^* - \boldsymbol{z}_t\|_1 + a_\gamma = \boldsymbol{\Omega}(\frac{s \log m}{\sqrt{m}})$ *and* $\alpha^{(l)} \leq 1 - \frac{2\lambda^{(l)}_t - (1 - \frac{\delta_l^2}{2})C_{\min}}{\lambda^{(l)}_{t-1}}$*, then with high probability of at least* $1 - \epsilon^{(l)}_{supp\text{-}pres}$*, the support, recovered at the first iteration, is preserved through the encoder iterations. We have* $a_\gamma = \mathcal{O}(\sqrt{s\delta_l})$ *and* $\epsilon^{(l)}_{supp\text{-}pres} := \epsilon^{(l)}_{supp\text{-}rec} + \epsilon^{(l)}_\gamma = 2p\exp\left(\frac{-C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}\right) + 2s\exp\left(\frac{-1}{\mathcal{O}(\delta_l)}\right)$*.*

The support preservation conditions on λ_t and α introduce two insights. First, with an increase of t, the code error decrease, hence the lower bound on λ_t. Second, the decay of λ_t as the encoder unrolls increases the upper bound on α. Hence, we suggest a decaying strategy in values of λ_t as t increases.

The utilization of knowledge of the code error, as we do in Theorem 4.2, to set the proper thresholding/bias/regularization parameters (λ) constitutes a fairly standard practice. Below we discuss similar results in the literature. For the preservation of correct signed-support in a sparse coding network, Rambhatla et al. (2018) provided a proper thresholding value at every iteration as a function of the ℓ_1-norm of the code error with respect to a ground-truth code; they additionally demonstrated an upper bound on the estimate of the code coefficients as a function of dictionary closeness. Moreover, Nguyen et al. (2019) used information on the range of ground-truth code to choose proper biases in their neural network to guarantee support recovery. Chen et al. (2018) similarly provided an upper bound on the bias of their unrolled sparse coding network at every layer as a function of ℓ_2-norm error between the code estimate at the layer and the ground-truth code. Overall, the error between a code estimate and the ground-truth code appearing in the lower bound on $\lambda^{(l)}_t$ can further simplified into terms related to terms such as the dictionary closeness δ_l, code sparsity. For example, Chatterji & Bartlett (2017), for their particular sparse coding algorithm, provided ℓ_∞-norm upper bound as a function of terms such as code sparsity, data dimensionality, code range, and dictionary error.

Code convergence and error Given the support recovery and its preservation, the encoder convergence studied in (Malézieux et al., 2022) can achieve linear convergences after its first iteration. We re-state this result on the rate of convergence of the encoder in Theorem 4.3. We drop the superscript (l) to simplify the notation.

Theorem 4.3 (Local forward pass code convergence)**.** *Given the encoder* $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$*, Assumption 3.6, Lemmas 3.2, A.1 and A.2, then* $\exists\ \rho < 1, B > 0$ *s.t.* $\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2 \leq \mathcal{O}(\rho^t)\ \forall t > B$*, where* $\hat{\boldsymbol{z}}$ *is the unique minimizer of lasso (1). Furthermore, given Theorem 4.1 and Theorem 4.2,* $B = 1$*.*

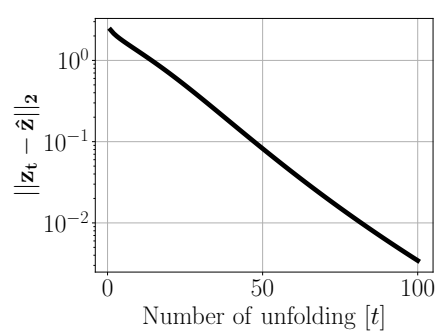


Figure 2: Code convergence (Theorem 4.3). As the network unrolls, $\boldsymbol{z}_t$ converges to $\hat{\boldsymbol{z}}$, the solution of lasso.

Theorem 4.3 shows that in PUDLE, $\boldsymbol{z}_t$ converges to $\hat{\boldsymbol{z}}$ at a linear rate eventually after a certain number of unrolling (Figure 2). The local linear convergence of ISTA and FISTA (Beck & Teboulle, 2009) (with global rates of $\mathcal{O}(1/t)$ and $\mathcal{O}(1/t^2)$) in the neighbourhood of a fixed-point is studied in (Tao et al., 2016). The speed of convergence depends on when support selection happens (Proposition 4.1) (Bredies & Lorenz, 2008; Zhang et al., 2017b; Liang et al., 2014). We showed in Theorem 4.1 and Theorem 4.2 that under mild assumptions, the support is selected and recovered after one encoder iteration. In addition to local convergence, we focus on recovery of $\boldsymbol{z}^*$ and show error on the unrolled code coefficients $\boldsymbol{z}^{(l)}_{t,(j)}$ with respect to ground-truth $\boldsymbol{z}^*_{(j)}$ as t increases. In Theorem 4.4, we consider the case where λ_t at layer t is set to according to Theorem 4.2; the bias decreases as the code error decreases among the layers and dictionary updates. We provide an upper bound on the coefficients errors as a function of code sparsity, dictionary error, and an unrolling error $e^{(l)\text{unroll}}_{t,j}$. The unrolling error goes to zero for appropriately large t. Moreover, Theorem 4.5 studies the case where the bias is fixed across the layers. In this scenario, we observe an additional term of λ^{fixed} in the upper bounds on the code coefficients error; this term shows that the code error when we strictly perform ℓ_1-norm based sparse coding does not go to zero. We refer to this error as an amplitude bias estimate error.

Theorem 4.4 (Global forward pass code error with variable λ_t)**.** *Given Assumptions 3.3 and 3.4, suppose $\boldsymbol{D}^{(l)}$ is μ_l-incoherent and $\delta_l = \mathcal{O}^*(1/\log p)$ close to $\boldsymbol{D}^*$. If $s = \mathcal{O}^*(\sqrt{m}/\mu \log m)$, $\mu = \mathcal{O}(\log m)$, and the regularizer and step size are chosen such that $\lambda^{(l)}_t = \frac{\mu_l}{\sqrt{m}}\|\boldsymbol{z}^* - \boldsymbol{z}_t\|_1 + a_\gamma = \boldsymbol{\Omega}(\frac{s \log m}{\sqrt{m}})$ and $\alpha^{(l)} \leq 1 - \frac{2\lambda^{(l)}_t - (1-\frac{\delta_l^2}{2})C_{\min}}{\lambda^{(l)}_{t-1}}$, then with high probability of at least $1 - \epsilon^{(l)}_{supp\text{-}pres}$, for $j \in supp(\boldsymbol{z}^*)$, the code coefficient error is*

$$|\boldsymbol{z}^{(l)}_{t,(j)} - \boldsymbol{z}^*_{(j)}| \leq \mathcal{O}(\sqrt{s\|\boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j\|_2} + e^{(l)unroll}_{t,j}) \tag{8}$$

and

$$\boldsymbol{z}_{T,(j)} = \boldsymbol{z}^*_{(j)}(1 - \beta^{(l)}_j) + \zeta^{(l)}_{T,j} \tag{9}$$

where $e^{(l)unroll}_{t,j} := 2(s-1)t\alpha\frac{\mu_l}{\sqrt{m}}\max_i |\boldsymbol{z}^{(l)}_{0,(i)} - \boldsymbol{z}^*_{(i)}|\delta_{\alpha,t-1} + |\boldsymbol{z}^{(l)}_{0,(j)} - \boldsymbol{z}^*_{(j)}|\delta_{\alpha,t}$, $\delta_{\alpha,t} := (1 - \alpha + 2\alpha\frac{\mu_l}{\sqrt{m}})^t$, $|\zeta^{(l)}_{T,j}| = \mathcal{O}(a_\gamma)$ with $a_\gamma = \mathcal{O}(\sqrt{s\delta_l})$, $\beta^{(l)}_j = \langle \boldsymbol{D}^*_j - \boldsymbol{D}^{(l)}_j, \boldsymbol{D}^*_j\rangle \leq \frac{\delta_l^2}{2}$ and $\epsilon^{(l)}_{supp\text{-}pres} := \epsilon^{(l)}_{supp\text{-}rec} + \epsilon^{(l)}_\gamma = 2p\exp(\frac{-C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}) + 2s\exp(\frac{-1}{\mathcal{O}(\delta_l)})$. With appropriately large t, we have $|\boldsymbol{z}^{(l)}_{t,(j)} - \boldsymbol{z}^*_{(j)}| = \mathcal{O}(\sqrt{s\|\boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j\|_2})$.

Theorem 4.5 (Global forward pass code error with fixed λ_t)**.** *Given Assumptions 3.3 and 3.4, suppose $\boldsymbol{D}^{(l)}$ is μ_l-incoherent and $\delta_l = \mathcal{O}^*(1/\log p)$ close to $\boldsymbol{D}^*$. If $s = \mathcal{O}^*(\sqrt{m}/\mu \log m)$, $\mu = \mathcal{O}(\log m)$, and the regularizer and step size are chosen such that $\lambda^{(l)}_t = \lambda^{fixed} = \frac{\mu_l}{\sqrt{m}}\|\boldsymbol{z}^* - \boldsymbol{z}_0\|_1 + a_\gamma = \boldsymbol{\Omega}(\frac{s \log m}{\sqrt{m}})$ and $\alpha^{(l)} \leq 1 - \frac{2\lambda^{(l)}_t - (1-\frac{\delta_l^2}{2})C_{\min}}{\lambda^{(l)}_{t-1}}$, then with high probability of at least $1 - \epsilon^{(l)}_{supp\text{-}pres}$, for $j \in supp(\boldsymbol{z}^*)$, the code coefficient error is*

$$|\boldsymbol{z}^{(l)}_{t,(j)} - \boldsymbol{z}^*_{(j)}| \leq \mathcal{O}(\sqrt{s\|\boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j\|_2} + e^{(l)unroll,fixed}_{t,j} + \lambda^{fixed}) \tag{10}$$

andTHe

$$\boldsymbol{z}_{T,(j)} = \boldsymbol{z}^*_{(j)}(1 - \beta^{(l)}_j) + \zeta^{(l)}_{T,j} \tag{11}$$

where $e^{(l)unroll,\ fixed}_{t,j} := (s-1)t\alpha\frac{\mu_l}{\sqrt{m}}\max_i |\boldsymbol{z}^{(l)}_{0,(i)} - \boldsymbol{z}^*_{(i)}|\delta^{fixed}_{\alpha,t-1} + |\boldsymbol{z}^{(l)}_{0,(j)} - \boldsymbol{z}^*_{(j)}|\delta^{fixed}_{\alpha,t}$, $\delta^{fixed}_{\alpha,t} := (1 - \alpha + \alpha\frac{\mu_l}{\sqrt{m}})^t$, $|\zeta^{(l)}_{T,j}| = \mathcal{O}(a_\gamma + \lambda^{fixed})$ with $a_\gamma = \mathcal{O}(\sqrt{s\delta_l})$, $\beta^{(l)}_j = \langle \boldsymbol{D}^*_j - \boldsymbol{D}^{(l)}_j, \boldsymbol{D}^*_j\rangle \leq \frac{\delta_l^2}{2}$, and $\epsilon^{(l)}_{supp\text{-}pres} := \epsilon^{(l)}_{supp\text{-}rec} + \epsilon^{(l)}_\gamma = 2p\exp(\frac{-C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}) + 2s\exp(\frac{-1}{\mathcal{O}(\delta_l)})$. With appropriately large t, we have $|\boldsymbol{z}^{(l)}_{t,(j)} - \boldsymbol{z}^*_{(j)}| = \mathcal{O}(\sqrt{s\|\boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j\|_2} + \lambda^{fixed})$.

Aside from code estimation where the network parameters (e.g., regularization and step size) are finely tuned according to support recovery and preservation conditions (Theorems 4.1 and 4.2), we provide a general upper bound on the error between the converged code and $\boldsymbol{z}^*$; the bound can be decomposed into two terms of the dictionary error and the biased amplitude estimate of the code.

Theorem 4.6 (Global forward pass code error)**.** *Let $\hat{\boldsymbol{z}}$ be the fixed-point of the encoder with iterations $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$. Given Assumption 3.6, Lemmas 3.2, A.1 and A.2, we have $\|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2 \leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^*)$, where $\hat{\delta}^* = \|\hat{\boldsymbol{z}}^* - \boldsymbol{z}^*\|_2$, $\hat{\boldsymbol{z}}$ is the unique minimizer of lasso (1) given the dictionary $\boldsymbol{D}$, $\hat{\boldsymbol{z}}^*$ is the unique minimizer of lasso (1) given the dictionary $\boldsymbol{D}^*$, and $\boldsymbol{z}^*$ is the ground-truth code.*

This general decomposition is to emphasize that aside from the current estimate of the dictionary, the code error is a function of the forward pass algorithm used to solve the sparse coding problem. Specifically, the upper bound states that at the best scenario where there is access to the generating dictionary $\boldsymbol{D}^*$, the forward pass solving lasso with fixed λ still gives a biased amplitude estimate of $\boldsymbol{z}^*$. Overall, the assumptions to get this bound are mild; the bound is valid independent of successful support recovery or data distribution. With incorporation of data distribution and conditions stated in Theorem 4.1 and Theorem 4.2, the upper bound $\hat{\delta}^*$ can be replaced with terms involving λ, and reaches at zero as λ decays across forward iterations.

Jacobian convergence and error Following properties similar to those used in Theorem 4.3, and assuming $\boldsymbol{J}_t$ is bounded (Assumption 4.1), we show in Theorem 4.7 that, as the PUDLE unrolls, the code Jacobian $\boldsymbol{J}_t$ converges to $\hat{\boldsymbol{J}}$, the Jacobian of the solution of the lasso. The convergence of the Jacobian of proximal gradient descent is also studied in (Bertrand et al., 2021) for hyperparameter selection through implicit differentiation (Bengio, 2000), where the Jacobian is taken w.r.t to the hyperparameter λ as opposed to $\boldsymbol{D}$.

Assumption 4.1 (Bounded Jacobian)**.** *The Jacobian is bounded, i.e., $\exists\ M_J > 0,$ s.t. $\|\boldsymbol{J}_t\|_2 \leq M_J\ \forall t$.*

Theorem 4.7 (Local forward pass Jacobian convergence)**.** *Given the recursion $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$, and $\hat{\boldsymbol{z}}$ the unique minimizer of lasso with Jacobian $\hat{\boldsymbol{J}}$, then $\exists\ \rho < 1, B > 0$ s.t. $\|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 \leq \mathcal{O}(t\rho^t)\ \forall t > B$. Furthermore, given Theorem 4.1 and Theorem 4.2, $B = 1$.*

The forward pass code and Jacobian convergences *after* support selection is similar to the results from (Malézieux et al., 2022). The highlights of our finding are that the order of upper bound convergences can be achieved from the first iteration of the encoder. In other words, we specify, in Theorem 4.1 and Theorem 4.2, the dictionary and data conditions such that the support can be recovered with $B = 1$. This resolves the instability issue discussed by Malézieux et al. (2022) in computation of the gradient $\boldsymbol{g}_t^{\text{ae-lasso}}$ outside of the support. Finally, we show that the global Jacobian error is in the order of dictionary error.

Theorem 4.8 (Global forward pass Jacobian error)**.** *Let $\hat{\boldsymbol{z}}$ be the fixed-point of the encoder with iterations $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$. Given Assumption 3.6, Lemmas 3.2, A.1 and A.2, we have $\|\hat{\boldsymbol{J}} - \boldsymbol{J}^*\|_2 \leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}_J^*)$, where $\hat{\delta}_J^* := \|\hat{\boldsymbol{J}}^* - \boldsymbol{J}^*\|_2$, and $\hat{\boldsymbol{J}}$, $\hat{\boldsymbol{J}}^*$ and $\boldsymbol{J}^*$ are Jacobians corresponding to $\hat{\boldsymbol{z}}$, $\hat{\boldsymbol{z}}^*$ and $\boldsymbol{z}^*$.*

4.2 Backward pass

We show two results for local gradient $\hat{\boldsymbol{g}}$ and global gradient $\boldsymbol{g}^*$ convergence. The goal is not to provide a finite sample analysis but to emphasize the relative differences between the gradients in Algorithm 2. The impact of gradient error for parameter estimation in the convex setting has been studied by Devolder et al. (2013) indicating that the convergence to the parameter's neighbourhood is dictated by the gradient error (Devolder et al., 2013; 2014). As dictionary learning is a bi-convex problem, findings of Devolder et al. (2013) hold as well for better estimation of the local dictionary at every step of alternating minimization. Moreover, Arora et al. (2015), provided a detailed analysis of sparse coding and various gradient estimations for dictionary learning, showing that by computing a more accurate gradient at every step of the alternating minimization scheme, the dictionary estimates converge to a closer neighbourhood of $\boldsymbol{D}^*$. Overall, the intuition is that the size of the gradient error dictates the size of the neighbourhood of the dictionary within which one can guarantee convergence. We argue that the method with lower gradient error recovers the dictionary better.

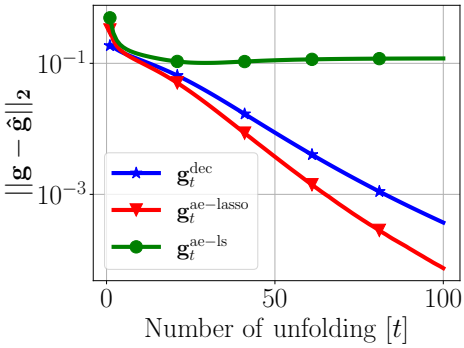


Figure 3: Convergence rate of gradients (Theorem 4.9).

Local gradient estimations We highlight the effect of finite unrolling on the gradient for parameter estimation (Ablin et al., 2020). Theorem 4.9 shows the convergence rate of gradients to $\hat{\boldsymbol{g}}$, determining the similarity of PUDLE and Algorithm 1.

Theorem 4.9 (Local convergence of gradients)**.** *Given the convergence results from the forward pass (Theorems 4.3 and 4.7), $\exists\ \rho < 1, B > 0$ such that $\forall t > B$, the errors of gradients defined in Algorithm 2 w.r.t $\hat{\boldsymbol{g}}$ (4) satisfy*

$$\begin{aligned} \|\boldsymbol{g}_t^{dec} - \hat{\boldsymbol{g}}\|_2 &\leq \mathcal{O}(\rho^t) \\ \|\boldsymbol{g}_t^{ae\text{-}lasso} - \hat{\boldsymbol{g}}\|_2 &\leq \mathcal{O}(t\rho^{2t}) \\ \|\boldsymbol{g}_t^{ae\text{-}ls} - \hat{\boldsymbol{g}}\|_2 &\leq \mathcal{O}(t\rho^{2t} + M_J\lambda\sqrt{s}). \end{aligned} \tag{12}$$

Moreover, the order of upper bounds is tight (see Lemma A.4).

First, upper bounds on the errors related to $\boldsymbol{g}_t^{\text{dec}}$ and $\boldsymbol{g}_t^{\text{ae-lasso}}$ go to zero as t increases. Hence, both gradients converge to $\hat{\boldsymbol{g}}$. This means that asymptotically as t increases, training PUDLE with $\boldsymbol{g}_t^{\text{dec}}$ and $\boldsymbol{g}_t^{\text{ae-lasso}}$ is equivalent to classical alternating-minimization (Algorithm 1). Second, as t increases, $\boldsymbol{g}_t^{\text{ae-lasso}}$ has faster convergence than $\boldsymbol{g}_t^{\text{dec}}$. Lastly, $\boldsymbol{g}_t^{\text{ae-ls}}$ is a biased estimator of $\hat{\boldsymbol{g}}$ (Figure 3). The convergence results on the error $\|\boldsymbol{g}_t^{\text{ae-lasso}} - \hat{\boldsymbol{g}}\|_2$ is previously studied by Malézieux et al. (2022).

Given the above convergence results, one may conclude that $\boldsymbol{g}_t^{\text{ae-lasso}}$ should be used for dictionary recovery. However, we show next that for dictionary recovery, the gradient $\boldsymbol{g}_t^{\text{ae-lasso}}$, used by Malézieux et al. (2022), is indeed a biased estimator of the global gradient $\boldsymbol{g}^*$ for recovery of $\boldsymbol{D}^*$. We decrease this bias by replacing $\boldsymbol{g}_t^{\text{ae-lasso}}$ with $\boldsymbol{g}_t^{\text{ae-ls}}$ and show that $\boldsymbol{g}_t^{\text{ae-ls}}$ results in a better recovery of $\boldsymbol{D}^*$ than $\boldsymbol{g}_t^{\text{ae-lasso}}$.

Global gradient estimations Theorem 4.10 shows the global gradient errors w.r.t $\boldsymbol{g}^*$ from (5). We omit the gradient $\boldsymbol{g}_t^{\text{dec}}$, as it is asymptotically equivalent to $\boldsymbol{g}_t^{\text{ae-lasso}}$. We study the errors in the limit to unrolling, i.e., as $t \to \infty$. This determines which PUDLE gradients recover $\boldsymbol{D}^*$ better (Devolder et al., 2013; 2014).

Theorem 4.10 (Global error of gradients)**.** *Given the convergence results from the forward pass, (Theorems 4.6 and 4.8), the errors of gradients defined in Algorithm 2 w.r.t global direction $\boldsymbol{g}^*$(defined in (5)) satisfy*

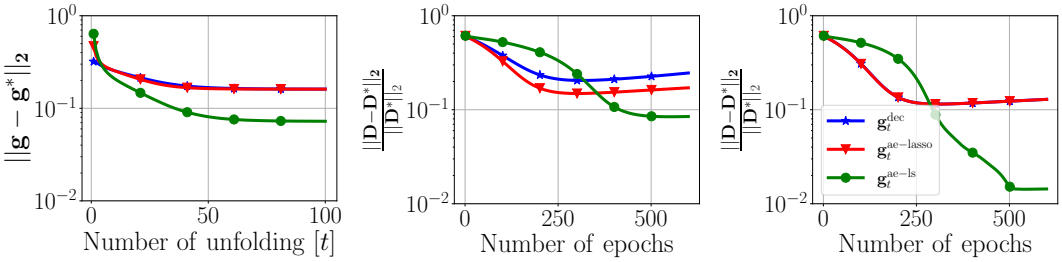
$$\begin{aligned} \|\boldsymbol{g}_\infty^{ae\text{-}lasso} - \boldsymbol{g}^*\|_2 &\leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2^2 + \|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^* + \hat{\delta}_J^* + M_J\lambda\sqrt{s}) \\ \|\boldsymbol{g}_\infty^{ae\text{-}ls} - \boldsymbol{g}^*\|_2 &\leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2^2 + \|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^* + \hat{\delta}_J^*). \end{aligned} \tag{13}$$

Several factors affect the order of upper bounds: the current estimate of the dictionary, code amplitude-bias error due to ℓ_1 norm, and the usage of ℓ_1 norm in the loss used for backpropagation. To study the bias in the gradient computation, let consider the scenario where $\boldsymbol{D} = \boldsymbol{D}^*$. We denote those gradients by superscript $\boldsymbol{D}^*$. If the gradients are not biased, then the upper bounds should goes to zero. The gradient errors are

$$\|\boldsymbol{g}_\infty^{\text{ae-lasso},\boldsymbol{D}^*} - \boldsymbol{g}^*\|_2 \leq \mathcal{O}(\hat{\delta}^* + \hat{\delta}_J^* + M_J\lambda\sqrt{s}) \qquad \text{and} \qquad \|\boldsymbol{g}_\infty^{\text{ae-ls},\boldsymbol{D}^*} - \boldsymbol{g}^*\|_2 \leq \mathcal{O}(\hat{\delta}^* + \hat{\delta}_J^*). \tag{14}$$

For $\boldsymbol{g}_\infty^{\text{ae-ls},\boldsymbol{D}^*}$, the radius of the error ball is only a function of the amplitude error of the code estimated through lasso compare to the ground-truth code $\boldsymbol{z}^*$. However, the error ball for the gradient $\boldsymbol{g}_\infty^{\text{ae-lasso},\boldsymbol{D}^*}$ includes an additional term concerning the usage of lasso loss containing the regularization term λ. This implies that the $\boldsymbol{D}^*$ neighbourhood at which the gradient $\boldsymbol{g}_\infty^{\text{ae-ls},\boldsymbol{D}^*}$ is guaranteed to converge to is smaller than of the $\boldsymbol{g}_\infty^{\text{ae-lasso},\boldsymbol{D}^*}$ (Figure 4a). Implications of such gradient estimation are seen in dictionary learning where $\boldsymbol{g}_\infty^{\text{ae-ls}}$ recovers $\boldsymbol{D}^*$ better (Figures 4b and 4c). In Figure 4b, the encoder unrolls for $T = 25$, hence the phenomenon of implicit acceleration is seen in faster and better dictionary learning performance of $\boldsymbol{g}_\infty^{\text{ae-lasso}}$ than $\boldsymbol{g}_\infty^{\text{dec}}$. In Figure 4c where $T = 100$, similar performance of $\boldsymbol{g}_\infty^{\text{dec}}$ and $\boldsymbol{g}_\infty^{\text{ae-lasso}}$ illustrates their asymptotic equivalence as $t \to \infty$ (See Appendix for additional noisy dictionary learning experiments where the measurements $\boldsymbol{x}$ are corrupted with zero-mean Gaussian noise such that the Signal-to-Noise-Ratio is approximately 12 SNR; in this setting, the aforementioned comparative analysis still holds.)

Towards unbiased estimation As long as λ is fixed within PUDLE, all defined gradients remain biased estimators of $\boldsymbol{g}^*$, due to the biased estimate of the code $\boldsymbol{z}^*$ through ℓ_1 norm. This bias exists while dictionary learning is performed strictly using lasso through Algorithm 1. Given the conditions on the regularizer



(a) Convergence for $\boldsymbol{g}^*$. (b) Learning ($T = 25$). (c) Learning ($T = 100$).

Figure 4: Results for PUDLE's global convergence (Theorem 4.10) and dictionary learning.

in Theorem 4.2 which we discussed in Section 4.1 and the derived upper bounds in Theorem 4.10, we suggest the decaying of λ across the encoder to reduce the gradient biases and improve dictionary learning. Next, we prove in Theorem 4.11 that PUDLE converges to $\boldsymbol{D}^*$ if λ decays across the layers t according to Theorem 4.4. Moreover, Theorem 4.12 proves that if λ stays fixed across the layers according to Theorem 4.5, then PUDLE only guarantees to converge to a close neighbourhood of the dictionary. In these analyses, we focus on $\boldsymbol{g}_T^{\text{dec}}$. Furthermore, we show in Section 4.3 that by decaying λ at each unrolled layer, the gradient bias vanishes, and we recover $\boldsymbol{D}^*$.

Dictionary learning Given the network parameters set by Theorem 4.4, Theorem 4.11 proves that using $\boldsymbol{g}_T^{\text{dec}}$, PUDLE recovers the dictionary; the dictionary error contracts at every update. Moreover, Theorem 4.12 proves that as long as λ stays fixed across the unrolled layers, PUDLE guarantees to converge to only $\boldsymbol{D}^*$ neighbourhood characterized by the regularization parameter λ. These analyses requires for $\boldsymbol{D}^{(l)}$ to maintain a closeness to $\boldsymbol{D}^*$ which we provide a proof for in Lemma A.7. Hence, the dictionary closeness assumption (Assumption 3.5) stays valid.

Theorem 4.11 (Dictionary learning with variable λ_t)**.** *Given Assumptions 3.3 and 3.4, suppose $\boldsymbol{D}^{(l)}$ is μ_l-incoherent and $(\delta_l, 2)$-close to $\boldsymbol{D}^*$ with $\delta_l = \mathcal{O}^*(1/\log p)$. If $s = \mathcal{O}(\sqrt{m})$, $\mu = \mathcal{O}(\log m)$, learning rate is $\eta = \mathcal{O}(\frac{p}{s(1-\delta_l^2/2)})$, and the regularizer and step size are set according to Theorem 4.4, then for any dictionary update l using $\boldsymbol{g}_T^{dec}$, with high probability of at least $1 - \epsilon_{supp\text{-}pres}^{(l)}$,*

$$\|\boldsymbol{D}_j^{(l+1)} - \boldsymbol{D}_j^*\|_2^2 \leq (1-\psi)\|\boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*\|_2^2 \tag{15}$$

where $\epsilon_{supp\text{-}pres}^{(l)} := \epsilon_{supp\text{-}rec}^{(l)} + \epsilon_\gamma^{(l)} = 2p\exp\left(\frac{-C_{\min}^2}{\mathcal{O}^*(\delta_l^2)}\right) + 2s\exp\left(\frac{-1}{\mathcal{O}(\delta_l)}\right)$.

Theorem 4.12 (Dictionary learning with fixed λ_t)**.** *Given Assumptions 3.3 and 3.4, suppose $\boldsymbol{D}^{(l)}$ is μ_l-incoherent and $(\delta_l, 2)$-close to $\boldsymbol{D}^*$ with $\delta_l = \mathcal{O}^*(1/\log p)$. If $s = \mathcal{O}(\sqrt{m})$, $\mu = \mathcal{O}(\log m)$, learning rate is $\eta = \mathcal{O}(\frac{p}{s(1-\delta_l^2/2)})$, and the regularizer λ^{fixed} and step size are set according to Theorem 4.5, then for any dictionary update l using $\boldsymbol{g}_T^{dec}$, with high probability of at least $1 - \epsilon_{supp\text{-}pres}^{(l)}$,*

$$\|\boldsymbol{D}_j^{(l+1)} - \boldsymbol{D}_j^*\|_2^2 \leq (1-\psi)\|\boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*\|_2^2 + \epsilon_\lambda \tag{16}$$

where $\epsilon_\lambda := \eta\frac{2p_j}{(1-\beta_j^{(l)})}\lambda^{fixed2}$, $\epsilon_{supp\text{-}pres}^{(l)} := \epsilon_{supp\text{-}rec}^{(l)} + \epsilon_\gamma^{(l)} = 2p\exp\left(\frac{-C_{\min}^2}{\mathcal{O}^*(\delta_l^2)}\right) + 2s\exp\left(\frac{-1}{\mathcal{O}(\delta_l)}\right)$.

4.3 Experiments

Dictionary learning We focus on the performance of the best-performing gradient estimator $\boldsymbol{g}_t^{\text{ae-ls}}$, and compare it with NOODL (Rambhatla et al., 2018), a state-of-the-art online dictionary learning algorithm, and SPORCO (Wohlberg, 2017), an alternating-minimization dictionary learning algorithm that uses lasso. NOODL, which uses iterative hard-thresholding (HT) for sparse coding and a gradient update employing the code's sign, has linear convergence upon proper initialization (Rambhatla et al., 2018). We note that the results from $\boldsymbol{g}_t^{\text{ae-lasso}}$ are not shown, as the gradient computation was unstable (Malézieux et al., 2022). We emphasize that our proposed gradient $\boldsymbol{g}_t^{\text{ae-ls}}$ does not suffer such instability. We train:

- $\boldsymbol{g}_t^{\text{ae-ls}}$: λ is fixed across iterations.
- $\boldsymbol{g}_t^{\text{ae-ls, decay}}$: λ decays (i.e., $\lambda_t = \lambda\nu^t$, with $0 < \nu < 1$) where ν decreases as training progresses.
- $\boldsymbol{g}_t^{\text{ae-ls, HT}}$: $\mathcal{P}_{\alpha\lambda}(v)$ is replaced with $\text{HT}_b(v) \triangleq v\mathbf{1}_{|v|\geq b}$.

With HT, the sparse coding step reduces to that from NOODL. In this case, we highlight the difference between the gradient update of our method (backpropagation) with NOODL. We focus on convergence, as η across methods is not comparable.

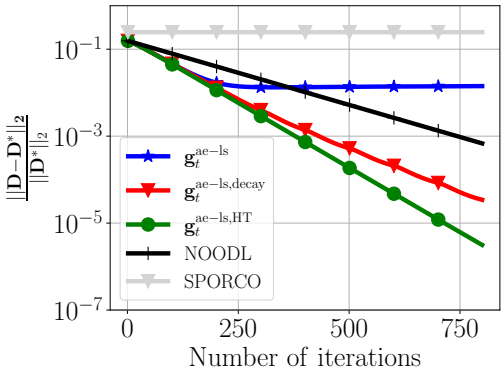
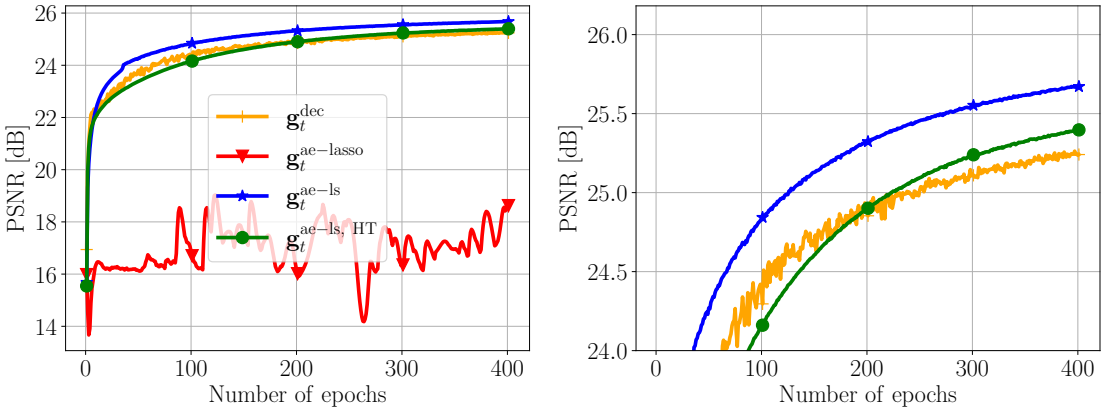


Figure 5: Dictionary convergences.

Figure 5 shows the convergence of $\boldsymbol{D} \in \mathbb{R}^{1000\times1500}$ to $\boldsymbol{D}^*$ when the code is 20-sparse (for other sparsity levels and details see Appendix C). A biased estimate of the code amplitudes results in convergence only to a neighbourhood of the dictionary (Rambhatla et al., 2018). This is observed in the convergence of $\boldsymbol{g}_t^{\text{ae-ls}}$ and SPORCO (final error is shown). The convergence of $\boldsymbol{g}_t^{\text{ae-ls}}$ to a closer neighbourhood than SPORCO supports Theorem 4.10. Moreover, with decaying λ, the code bias vanishes, hence $\boldsymbol{g}_t^{\text{ae-ls, decay}}$ and $\boldsymbol{g}_t^{\text{ae-ls, HT}}$ converges to $\boldsymbol{D}^*$ similar to NOODL.



(a) $\boldsymbol{g}_t^{\text{ae-lasso}}$ training is not stable.

(b) $\boldsymbol{g}_t^{\text{ae-ls}}$ improvement is smoother than $\boldsymbol{g}_t^{\text{dec}}$.

Figure 6: Networks behaviour (test PSNR) during training as a function of epochs.

Image denoising To further highlight the advantage of $\boldsymbol{g}_t^{\text{ae-ls}}$ over the other gradients, we compare them in a supervised task of image denoising. In addition to $\boldsymbol{g}_t^{\text{ae-ls}}$, $\boldsymbol{g}_t^{\text{ae-lasso}}$, and $\boldsymbol{g}_t^{\text{dec}}$, we consider $\boldsymbol{g}_t^{\text{ae-ls, HT}}$ where the proximal operator is replaced with HT. This is to compare with sparse coding scheme of NOODL. We do not compare against NOODL's dictionary update, as this computation for two-dimensional convolutions is not straightforward. Prior works have shown that variants of PUDLE either rival or outperform state-of-the-art architectures (Simon & Elad, 2019; Tolooshams et al., 2020). Thus, we focus on a comparative analysis of the gradients. We trained on 432 and tested on 68 images from BSD (Martin et al., 2001). BSD dataset is a popular training dataset for denoising (Zhang et al., 2017a; Simon & Elad, 2019; Mohan et al., 2019). We used a convolutional dictionary and corrupted images with zero-mean Gaussian noise of standard deviation of 25 (see Appendix C for details). We initialized the dictionary filters by standard Normal distribution; this is to follow the norm in the deep learning literature and to demonstrate the practicality and usefulness of PUDLE in the absence of an initialization method. We evaluate the denoising performance of soft-thresholding using λ and HT with b in peak signal-to-noise-ratio (PSNR).

First, we highlight the stability of $\boldsymbol{g}_t^{\text{ae-ls}}$ against $\boldsymbol{g}_t^{\text{ae-lasso}}$; Figure 6a shows the network dynamics in terms of test PSNR as a function of epochs when $\lambda = 0.16$ for $\boldsymbol{g}_t^{\text{ae-ls}}$, $\boldsymbol{g}_t^{\text{ae-lasso}}$, $\boldsymbol{g}_t^{\text{dec}}$ and $b = 0.05$ for $\boldsymbol{g}_t^{\text{ae-ls, HT}}$. We observed that $\boldsymbol{g}_t^{\text{ae-ls}}$ uses full backpropagation and stays stable. However, the training with $\boldsymbol{g}_t^{\text{ae-lasso}}$ is not stable and unstable to perform denoising where the noisy PSNR is approximately 20 dB (Malézieux et al.,

2022). Second, Figure 6b shows that compared to $\boldsymbol{g}_t^{\text{dec}}$, the backpropagated gradients result in a smoother improvement during training. Moreover, Table 1 shows that the advantage of $\boldsymbol{g}_t^{\text{ae-ls}}$ over $\boldsymbol{g}_t^{\text{dec}}$ is not limited to dictionary learning and is seen in denoising. We have excluded the results for $\boldsymbol{g}_t^{\text{ae-lasso}}$ from Table 1 as the network failed to denoise (see Figure 6a). Additionally, the superior performance of $\boldsymbol{g}_t^{\text{ae-ls}}$ compared to $\boldsymbol{g}_t^{\text{ae-ls, HT}}$ highlights the benefits of PUDLE (i.e., ℓ_1-based unrolling) against HT used in NOODL.

Table 1: Denoising of BSD68. Reported numbers are mean (std) PSNR given three independent trials.

METHOD		**PSNR [dB]**			
	λ	0.08	0.12	0.16	0.2
$\boldsymbol{g}_t^{\text{dec}}$		24.21 (0.12)	24.93 (0.14)	25.25 (0.06)	24.88 (0.00)
$\boldsymbol{g}_t^{\text{ae-ls}}$		24.79 (0.03)	25.43 (0.03)	**25.63** (0.04)	25.46 (0.05)
	b	0.02	0.05	0.08	0.1
$\boldsymbol{g}_t^{\text{ae-ls, HT}}$		22.92 (0.07)	25.26 (0.1)	24.76 (0.06)	23.94 (0.13)

5 Interpretable Sparse Codes and Dictionary

One motivation behind using algorithm unrolling to design deep architectures is interpretability (Monga et al., 2019); they argue that the designed networks are interpretable as they capture domain knowledge via an optimization model. For example, Tolooshams et al. (2021a) takes advantage of the interpretability of learned weights in an unrolled dictionary learning network to solve spike sorting, an unsupervised source separation problem in computational neuroscience. Moreover, Kim et al. (2010) uses sparse coding to learn interpretable representations of human motions. However, none of the existing methods in the literature provide interpretability results that open the black-box network through building a mathematical relation between the learned dictionary, training data, and test representation/reconstruction. This section analyzes the interpretability of the unrolled sparse coding method in this context. We note that such mathematical relation and interpretability results is missing in the dictionary-learning literature, irrespective of whether one use an unrolling network. We provide the following theorem.

Theorem 5.1 (Interpretable unrolled network)**.** *Consider the dictionary learning optimization of the form* $\min_{\boldsymbol{Z},\boldsymbol{D}} \frac{1}{2}\|\boldsymbol{X}-\boldsymbol{D}\boldsymbol{Z}\|_2^2 + \lambda\|\boldsymbol{Z}\|_1 + \omega/2\|\boldsymbol{D}\|_F^2$*, where* $\boldsymbol{X} = [\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^n] \in \mathbb{R}^{m\times n}$ *and* $\boldsymbol{Z} = [\boldsymbol{z}^1, \boldsymbol{z}^2, \ldots, \boldsymbol{z}^n] \in \mathbb{R}^{p\times n}$*. Let* $\tilde{\boldsymbol{Z}}$ *be the given converged sparse codes, then stationary points of the problem w.r.t the network weights (dictionary) follows* $\tilde{\boldsymbol{D}} = \boldsymbol{X}\boldsymbol{G}^{-1}\tilde{\boldsymbol{Z}}^T$*, where we denote* $\boldsymbol{G} \coloneqq (\tilde{\boldsymbol{Z}}^T\tilde{\boldsymbol{Z}} + \omega\boldsymbol{I})$*.*

The dictionary interpolates the training data Given Theorem 5.1, each learned atom interpolates the training data, i.e.,

$$\tilde{\boldsymbol{D}}_j = \boldsymbol{X}(\boldsymbol{G}^{-1}\boldsymbol{w}_j) = \sum_{k=1}^{n}(\boldsymbol{G}^{-1}\boldsymbol{w}_j)_k\boldsymbol{x}^k \tag{17}$$

where $\boldsymbol{w}_j = [\tilde{\boldsymbol{z}}_j^1, \tilde{\boldsymbol{z}}_j^2, \ldots, \tilde{\boldsymbol{z}}_j^n]^{\mathrm{T}} \in \mathbb{R}^n$ is a vector containing the training code activity for dictionary atom j. Specifically, the importance of training image $\boldsymbol{x}^k$ in learning dictionary atom j is captured by the term $(\boldsymbol{G}^{-1}\boldsymbol{w}_j)_k$. This proves the dictionary lives in the spans of the training set. Given the small number of atoms compared to the size of training set, (17) shows that the dictionary summarizes the training examples. We trained the network on digits of $\{0, 1, 2, 3, 4\}$ MNIST (Figure 7 shows a fraction of the most used learned atoms). Figure 8 visualizes dictionary atoms along with training images with the highest contribution (green color) and the lowest contribution (red color).

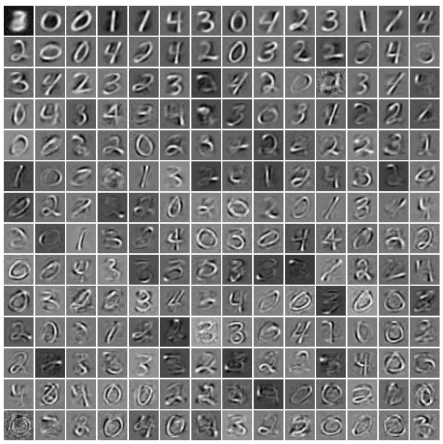
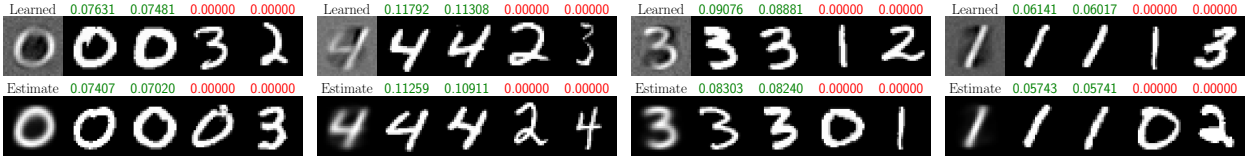


Figure 7: Fraction of dictionary atoms learned from $\{0, 1, 2, 3, 4\}$ MNIST.

In addition, we used (17) on the partial training data to reconstruct the learned training image (shown as Estimate). Next, we interpret the relation between the new test data to the training data using representer point selection, similar to (Yeh et al., 2018).



(a) 0 looking like atom. (b) 4 looking like atom. (c) 3 looking like atom. (d) 1 looking like atom.

Figure 8: Training image contributions to learning the dictionary.

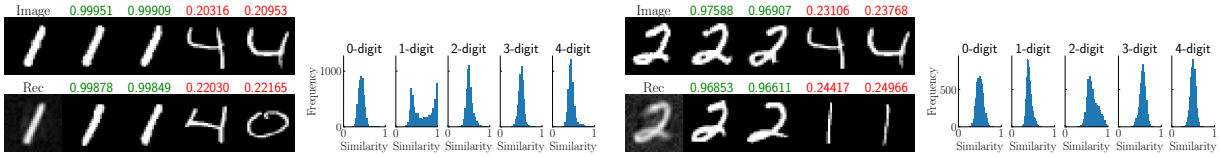
Relation between new test image and training data For representation of a new data, we observe that the reconstruction of a new example $\boldsymbol{x}^j$ is a linear combination of all the training examples, i.e.,

$$\hat{\boldsymbol{x}}^j = \tilde{\boldsymbol{D}}\hat{\boldsymbol{z}}^j = \boldsymbol{X}\boldsymbol{\beta}^j = \sum_{k=1}^{n} \boldsymbol{\beta}_k^j \boldsymbol{x}^k \tag{18}$$

where $\hat{\boldsymbol{x}}^j$ denotes reconstruction, $\hat{\boldsymbol{z}}^j$ is the code estimate, $\boldsymbol{\beta}^j = \boldsymbol{G}^{-1}\tilde{\boldsymbol{Z}}^{\mathrm{T}}\hat{\boldsymbol{z}}^j \in \mathbb{R}^n$, and $\boldsymbol{\beta}_k^j = \sum_{a=1}^{n} \boldsymbol{G}_{ka}^{-1}\langle \tilde{\boldsymbol{z}}^a, \hat{\boldsymbol{z}}^j\rangle$. We observe that the contribution of image k into the reconstruction of the test image is a function of $\boldsymbol{\beta}_k^j$, and the energy of $\boldsymbol{\beta}_k^j$ itself depends on the whole training set, and $\boldsymbol{G}^{-1}$. (18) shows how each image is reconstructed as interpolation of the training images. Figure 9 shows this results, where images with high (green color) $\boldsymbol{\beta}_k^j$ contribution are similar to the test image and those with low (red color) $\boldsymbol{\beta}_k^j$ contribution are different. In addition, we can evaluate the overall quality of the reconstruction by looking into $\boldsymbol{\beta}_k^j$ in (18). For example, we observed that for test MNIST, unnormalized $\boldsymbol{\beta}_k^j$ corresponding to high contributing training images is above 1. However, for resized-CIFAR, unnormalized $\boldsymbol{\beta}_k^j$ of high contributing training images are often half or an order of magnitude lower than the MNIST case. This informs us of a bad representation/reconstruction of CIFAR image by the trained network. From another perspective, we can write the new image as

(a) 0 test image. (b) 1 test image. (c) 3 test image. (d) 4 test image.

Figure 9: Interpolation of training data to reconstruct a new image. Contribution of training images are shown in green color (high contribution) and red color (low contribution). $\boldsymbol{\beta}^j$ is normalized over the used training examples.



(a) Digit 1 test image. (b) Digit 2 test image.

Figure 10: Contribution of images with code similarity into reconstruction of a new test image along with the histograms of the similarity of the test code to training codes from each class.

$$\boldsymbol{x}^j = \tilde{\boldsymbol{D}}\hat{\boldsymbol{z}}^j = \sum_{k=1}^{n} (\boldsymbol{X}\boldsymbol{G}^{-1})_k \langle \tilde{\boldsymbol{z}}^k, \hat{\boldsymbol{z}}^j\rangle \tag{19}$$

i.e., the contribution of each training image for reconstruction is a function of their code similarity to the new image and properties of the Gram matrix of training set code similarities. Specifically, the relation rules the contribution of transformed image k (i.e., $(\boldsymbol{X}\boldsymbol{G}^{-1})_k$) into reconstruction of the test image as a function of its code similarity $\langle \tilde{\boldsymbol{z}}^k, \hat{\boldsymbol{z}}^j \rangle$. In other words, (19) shows that training images with the highest code similarity to the representation of the new image have the highest contribution to its reconstruction. This interpretation is demonstrated in Figure 10. The training images with the highest code similarity (green color) and the lowest similarity (red) are shown. In addition, the figure demonstrates the histogram of the code similarity between the test image and the training set, grouped by their class digit. For example, for digit 1 test image, its code similarity to train images from class 1 are bimodal. This corresponds to 1 digits that are tilted to the left (low similarity) and right (high similarity). Moreover, for digit 2 test image, we observe that the histogram of images corresponding to digit 2 are shifted the most to the right (highest similarity) than the other classes.

6 Conclusions

This paper studied dictionary learning and analyzed the dynamics of unrolled sparse coding networks through a provable unrolled dictionary learning (PUDLE) framework. First, we provided a theoretical analysis of the forward pass for code recovery. We discussed the bias introduced by ℓ_1-based sparse coding in the forward pass, and how this affects the dictionary estimate in the backward pass. Second, we showed strategies to mitigate the propagation of this code bias into the backward pass; this is achieved by modification of the loss function used in the backward pass. We demonstrated that this bias could be further reduced and eliminated by decaying the regularization parameter within the unrolled layers. Additionally, we provided sufficient conditions on the data distribution and network initialization to guarantee stability of backpropagated gradient computations. In the absence of such conditions, we proposed a modification to the loss function that resolves the gradient explosion and allows stable learning. In an image denoising task, we showed PUDLE outperforms the NOODL sparse coding scheme (Rambhatla et al., 2018). Motivated by interpretability as a popular feature unrolled deep neural networks, we derived a mathematical relation between the network weights (dictionary) and the training dataset. We proved that the network weights live in the span of the training dataset, and constructed a relation between predictions of new test examples and the training set. The latter allows the user to extract images from the training set that are similar/dissimilar to the test image in representation/reconstruction.

References

Pierre Ablin, Thomas Moreau, Mathurin Massias, and Alexandre Gramfort. Learning step sizes for unfolded sparse coding. In *Proceedings of Advances in Neural Information Processing Systems*, volume 32, pp. 1–11, 2019.

Pierre Ablin, Gabriel Peyré, and Thomas Moreau. Super-efficiency of automatic differentiation for functions defined as a minimum. In *Proceedings of International Conference on Machine Learning*, pp. 32–41. PMLR, 2020.

Alekh Agarwal, Animashree Anandkumar, Prateek Jain, Praneeth Netrapalli, and Rashish Tandon. Learning sparsely used overcomplete dictionaries. In Maria Florina Balcan, Vitaly Feldman, and Csaba Szepesvári (eds.), *Proc the 27th Conference on Learning Theory*, volume 35 of *Proceedings of Machine Learning Research*, pp. 123–137, Barcelona, Spain, 13–15 Jun 2014. PMLR.

Alekh Agarwal, Animashree Anandkumar, and Praneeth Netrapalli. A clustering approach to learning sparsely used overcomplete dictionaries. *IEEE Transactions on Information Theory*, 63(1):575–592, 2017. doi: 10.1109/TIT.2016.2614684.

M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

Kazunori Akiyama, Kazuki Kuramochi, Shiro Ikeda, Vincent L Fish, Fumie Tazaki, Mareki Honma, Sheperd S Doeleman, Avery E Broderick, Jason Dexter, Monika Mościbrodzka, et al. Imaging the schwarzschild-

radius-scale structure of m87 with the event horizon telescope using sparse modeling. *The Astrophysical Journal*, 838(1):1, 2017.

Sanjeev Arora, Rong Ge, and Ankur Moitra. New algorithms for learning incoherent and overcomplete dictionaries. In Maria Florina Balcan, Vitaly Feldman, and Csaba Szepesvári (eds.), *Proceedings of the 27th Conference on Learning Theory*, volume 35 of *Proceedings of Machine Learning Research*, pp. 779–806, Barcelona, Spain, 13–15 Jun 2014. PMLR.

Sanjeev Arora, Rong Ge, Tengyu Ma, and Ankur Moitra. Simple, efficient, and neural algorithms for sparse coding. In Peter Grünwald, Elad Hazan, and Satyen Kale (eds.), *Proceedings of Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 113–149, Paris, France, 03–06 Jul 2015. PMLR.

Hedy Attouch and Jérôme Bolte. On the convergence of the proximal algorithm for nonsmooth functions involving analytic features. *Mathematical Programming*, 116(1):5–16, 2009.

Waheed U. Bajwa, Kfir Gedalyahu, and Yonina C. Eldar. Identification of parametric underspread linear systems and super-resolution radar. *IEEE Transactions on Signal Processing*, 59(6):2548–2561, 2011.

Boaz Barak, Jonathan A. Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of Annual ACM Symposium on Theory of Computing*, STOC '15, pp. 143–151, 2015.

Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18, 2018.

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.

Yoshua Bengio. Gradient-based optimization of hyperparameters. *Neural computation*, 12(8):1889–1900, 2000.

Quentin Bertrand, Quentin Klopfenstein, Mathurin Massias, Mathieu Blondel, Samuel Vaiter, Alexandre Gramfort, and Joseph Salmon. Implicit differentiation for fast hyperparameter selection in non-smooth convex learning. *arXiv:2105.01637*, 2021.

Thomas Blumensath and Mike E Davies. Iterative thresholding for sparse approximations. *Journal of Fourier analysis and Applications*, 14(5-6):629–654, 2008.

Kristian Bredies and Dirk A Lorenz. Linear convergence of iterative soft-thresholding. *Journal of Fourier Analysis and Applications*, 14(5-6):813–837, 2008.

Emmanuel J. Candès and Yaniv Plan. Near-ideal model selection by ℓ_1 minimization. *The Annals of Statistics*, 37(5A):2145 – 2177, 2009.

Niladri S Chatterji and Peter L Bartlett. Alternating minimization for dictionary learning: Local convergence guarantees. *arXiv:1711.03634*, pp. 1–26, 2017.

Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

Xiaohan Chen, Jialin Liu, Zhangyang Wang, and Wotao Yin. Theoretical linear convergence of unfolded ista and its practical weights and thresholds. In *Proceedings of Advances in Neural Information Processing Systems*, volume 31, pp. 1–11, 2018.

Brian Cleary, Le Cong, Anthea Cheung, Eric S. Lander, and Aviv Regev. Efficient generation of transcriptomic profiles by random composite measurements. *Cell*, 171(6):1424–1436.e18, 2017. ISSN 0092-8674.

Brian Cleary, Brooke Simonton, Jon Bezney, Evan Murray, Shahul Alam, Anubhav Sinha, Ehsan Habibi, Jamie Marshall, Eric S Lander, Fei Chen, et al. Compressed sensing for highly efficient imaging transcriptomics. *Nature Biotechnology*, pp. 1–7, 2021.

I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.

Olivier Devolder, François Glineur, Yurii Nesterov, et al. First-order methods with inexact oracle: the strongly convex case. Technical report, UniversitÃ© catholique de Louvain, Center for Operations Research and . . . , 2013.

Olivier Devolder, François Glineur, and Yurii Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146(1):37–75, 2014.

Michael Elad. *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer Science & Business Media, 2010.

Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

K. Engan, S.O. Aase, and J. Hakon Husoy. Method of optimal directions for frame design. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 5, pp. 2443–2446 vol.5, 1999.

Matthias Feurer and Frank Hutter. Hyperparameter optimization. In *Automated Machine Learning*, pp. 3–33. Springer, Cham, 2019.

Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In *Proceedings of International Conference on Machine Learning*, pp. 1165–1173, 2017.

Jean Charles Gilbert. Automatic differentiation and iterative processes. *Optimization methods and software*, 1(1):13–21, 1992.

Raja Giryes, Yonina C. Eldar, Alex M. Bronstein, and Guillermo Sapiro. Tradeoffs between convergence speed and reconstruction accuracy in inverse problems. *IEEE Transactions on Signal Processing*, 66(7): 1676–1690, 2018.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of international conference on international conference on machine learning*, pp. 399–406, 2010.

Elaine T Hale, Wotao Yin, and Yin Zhang. A fixed-point continuation method for l1-regularized minimization with applications to compressed sensing. *CAAM TR07-07, Rice University*, 43:44, 2007.

Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

John R. Hershey, Jonathan Le Roux, and Felix Weninger. Deep unfolding: Model-based inspiration of novel deep architectures. *arXiv:1409.2574*, pp. 1–27, 2014.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of Annual ACM Symposium on Theory of Computing*, pp. 665–674, 2013. ISBN 9781450320290.

Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, and Francis Bach. Proximal methods for hierarchical sparse coding. *The Journal of Machine Learning Research*, 12:2297–2334, 2011.

Taehwan Kim, Gregory Shakhnarovich, and Raquel Urtasun. Sparse coding for learning interpretable spatio-temporal primitives. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Proceedings of Advances in Neural Information Processing Systems*, volume 23, 2010.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pp. 9–48. Springer, 2012.

Yuelong Li, Mohammad Tofighi, Junyi Geng, Vishal Monga, and Yonina C. Eldar. Efficient and interpretable deep blind image deblurring via algorithm unrolling. *IEEE Transactions on Computational Imaging*, 6: 666–681, 2020.

Jingwei Liang, Jalal Fadili, and Gabriel Peyré. Local linear convergence of forward–backward under partial smoothness. *Proceedings of Advances in neural information processing systems*, 27, 2014.

Jialin Liu and Xiaohan Chen. Alista: Analytic weights are as good as learned weights in lista. In *Proceedings of International Conference on Learning Representations*, 2019.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of Annual International Conference on Machine Learning*, pp. 689–696, 2009a.

Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis Bach. Supervised dictionary learning. In *Proceedings of Advances in Neural Information Processing Systems*, volume 21, pp. 1–8, 2009b.

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Understanding approximate and unrolled dictionary learning for pattern recovery. In *Proceedings of International Conference on Learning Representations*, 2022.

D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of IEEE International Conference on Computer Vision*, volume 2, pp. 416–423, 2001.

Sreyas Mohan, Zahra Kadkhodaie, Eero P Simoncelli, and Carlos Fernandez-Granda. Robust and interpretable blind image denoising via bias-free convolutional neural networks. In *Proceedings of International Conference on Learning Representations*, 2019.

Vishal Monga, Yuelong Li, and Yonina C Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *arXiv:1912.10557*, pp. 1–27, 2019.

Thomas Moreau and Joan Bruna. Understanding trainable sparse coding via matrix factorization. In *Proceedings of 5th International Conference on Learning Representations*, pp. 1–13, 2017.

Thanh V Nguyen, Raymond KW Wong, and Chinmay Hegde. On the dynamics of gradient descent for autoencoders. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pp. 2858–2867. PMLR, 2019.

Kenji Nose-Filho, Andre Kazuo Takahata, Renato Lopes, and Joao Marcos Travassos Romano. Improving sparse multichannel blind deconvolution with correlated seismic data: Foundations and further results. *IEEE Signal Processing Magazine*, 35(2):41–50, 2018.

Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Sirisha Rambhatla, Xingguo Li, and Jarvis Haupt. Noodl: Provable online dictionary learning and sparse coding. In *Proceedings of International Conference on Learning Representations*, pp. 1–11, 2018.

Akshay Rangamani, Anirbit Mukherjee, Amitabh Basu, Ashish Arora, Tejaswini Ganapathi, Sang Chin, and Trac D. Tran. Sparse coding and autoencoders. In *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pp. 36–40, 2018.

Marc aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann Cun. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2007.

Marc aurelio Ranzato, Y-lan Boureau, and Yann Cun. Sparse feature learning for deep belief networks. In J. Platt, D. Koller, Y. Singer, and S. Roweis (eds.), *Proceedings of Advances in Neural Information Processing Systems*, volume 20, 2008.

Saharon Rosset, Ji Zhu, and Trevor Hastie. Boosting as a regularized path to a maximum margin classifier. *The Journal of Machine Learning Research*, 5:941–973, 2004.

Christian J. Schuler, Michael Hirsch, Stefan Harmeling, and Bernhard Schölkopf. Learning to deblur. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(7):1439–1451, 2016.

Dror Simon and Michael Elad. Rethinking the csc model for natural images. In *Proceedings of Advances in Neural Information Processing Systems*, volume 32, pp. 1–11, 2019.

Oren Solomon, Regev Cohen, Yi Zhang, Yi Yang, Qiong He, Jianwen Luo, Ruud J. G. van Sloun, and Yonina C. Eldar. Deep unfolded robust pca with application to clutter suppression in ultrasound. *IEEE Transactions on Medical Imaging*, 39(4):1051–1063, 2020.

Daniel A. Spielman, Huan Wang, and John Wright. Exact recovery of sparsely-used dictionaries. In *Proceedings of Annual Conference on Learning Theory*, volume 23 of *PMRL*, pp. 37.1–37.18, 2012.

Pablo Sprechmann, Alex Bronstein, and Guillermo Sapiro. Learning efficient structured sparse models. In *Proceedings of International Coference on International Conference on Machine Learning*, pp. 219–226, 2012.

Shaozhe Tao, Daniel Boley, and Shuzhong Zhang. Local linear convergence of ista and fista on the lasso problem. *SIAM Journal on Optimization*, 26(1):313–336, 2016.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. ISSN 00359246.

Ryan J. Tibshirani. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7(none):1456 –1490, 2013.

Bahareh Tolooshams, Sourav Dey, and Demba Ba. Scalable convolutional dictionary learning with constrained recurrent sparse auto-encoders. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2018.

Bahareh Tolooshams, Andrew Song, Simona Temereanca, and Demba Ba. Convolutional dictionary learning based auto-encoders for natural exponential-family distributions. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9493–9503. PMLR, 13–18 Jul 2020.

Bahareh Tolooshams, Sourav Dey, and Demba Ba. Deep residual autoencoders for expectation maximization-inspired dictionary learning. *IEEE Transactions on Neural Networks and Learning Systems*, 32(6):2415–2429, 2021a.

Bahareh Tolooshams, Satish Mulleti, Demba Ba, and Yonina C Eldar. Unfolding neural networks for compressive multichannel blind deconvolution. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2890–2894. IEEE, 2021b.

Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso). *IEEE transactions on information theory*, 55(5):2183–2202, 2009.

Zhaowen Wang, Ding Liu, Jianchao Yang, Wei Han, and Thomas Huang. Deep networks for image super-resolution with sparse prior. In *Proceedings of IEEE International Conference on Computer Vision*, pp. 370–378, 2015.

Brendt Wohlberg. Sporco: A python package for standard and convolutional sparse representations. In *Proceedings of the 15th Python in Science Conference, Austin, TX, USA*, pp. 1–8, 2017.

Bo Xin, Yizhou Wang, Wen Gao, David Wipf, and Baoyuan Wang. Maximal sparsity with deep networks? In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 1–9, 2016.

Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

Chih-Kuan Yeh, Joon Sik Kim, Ian EH Yen, and Pradeep Ravikumar. Representer point selection for explaining deep neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 9311–9321, 2018.

Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017a.

Lufang Zhang, Yaohua Hu, Chong Li, and Jen-Chih Yao. A new linear convergence result for the iterative soft thresholding algorithm. *Optimization*, 66(7):1177–1189, 2017b.

A Appendix - proofs

A.1 Notation

Bold-lower-case and upper-case letters refer to vectors $\boldsymbol{d}$ and matrices $\boldsymbol{D}$. We use $\boldsymbol{d}_{(j)}$ to denote the jth element of the vector $\boldsymbol{d}$, and $\boldsymbol{D}_j$ is the jth column of the matrix $\boldsymbol{D}$. We denote the code estimate at unrolled layer t by $\boldsymbol{z}_t$. $\lambda > 0$ is the regularization (sparsity-enforcing) parameter. $\sigma_{\max}(\boldsymbol{D})$ is the maximum singular value of $\boldsymbol{D}$. When taking the derivatives or norms w.r.t the matrix $\boldsymbol{D}$, we assume that $\boldsymbol{D}$ is vectorized. $\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ and $\nabla_2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ are the first derivatives of the loss w.r.t $\boldsymbol{z}$ and $\boldsymbol{D}$, respectively. $\nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ is the second derivative of the loss w.r.t $\boldsymbol{z}$. $\nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ is the derivative of $\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ w.r.t $\boldsymbol{D}$. The support of $\boldsymbol{z}$ is $\text{supp}(\boldsymbol{z}) \triangleq \{j : \boldsymbol{z}_{(j)} \neq 0\}$.

A.2 Basic definitions and Lemmas

We list four definitions used throughout the paper below.

Definition A.1 (μ-incoherence)**.** *$\boldsymbol{D}$ is μ-incoherent, i.e., for every pair (i, j) of columns, $|\langle \boldsymbol{D}_i, \boldsymbol{D}_j \rangle| \leq \mu/\sqrt{m}$.*

Definition A.2 ((δ, κ)-closeness)**.** *Dictionary $\boldsymbol{D}$ is δ-close to $\boldsymbol{D}^*$, i.e., there is a permutation π and sign flip operator u such that $\forall i$ $\|u(i)\boldsymbol{D}_{\pi(i)} - \boldsymbol{D}_i^*\|_2 \leq \delta$. Additionally, $\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \leq \kappa \|\boldsymbol{D}^*\|_2$.*

Definition A.3 (Lipschitz function)**.** *A function $f : \mathbb{R}^m \to \mathbb{R}^p$ is L-Lipschitz w.r.t a norm $\|\cdot\|$ if $\exists$ $L > 0$ s.t. $\|f(a) - f(b)\| \leq L\|a - b\|$ $\forall a, b \in \mathbb{R}^m$.*

Definition A.4 (Lipschitz differentiable function)**.** *A twice differentiable function $f : \mathbb{R}^m \to \mathbb{R}^p$ is L-Lipschitz differentiable w.r.t a norm $\|\cdot\|$ iff $\exists$ $L > 0$ s.t. $\|\nabla^2 f(a)\| \leq L$ $\forall a \in \mathbb{R}^m$.*

Definition A.5 (Strong convexity)**.** *A twice differentiable function $f : \mathbb{R}^m \to \mathbb{R}^p$ is strongly convex if $\exists$ $\mu > 0$ s.t. $\nabla^2 f(a) \succeq \mu \boldsymbol{I}$.*

Definition A.6 (Norm of subgradient)**.** *For norms involving subgradients, we define $\|\partial h(\boldsymbol{z})\| := \max_{\boldsymbol{v} \in \partial h(\boldsymbol{z})} \|\boldsymbol{v}\|$.*

In the proof of the theorems, we use the strong convexity of the reconstruction loss after support selection and the bounded property of the Lipschitz mapping stated below.

Lemma A.1 (Strong convexity of reconstruction loss)**.** *Given the support selection (Proposition 4.1), $\boldsymbol{D}_S^T \boldsymbol{D}_S$ is full-rank. Thus, $\forall t > B, \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) = \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_{t,S}, \boldsymbol{D}_S)$ is strongly convex (Definition A.5) in $\boldsymbol{z}$.*

Lemma A.2 (Lipschitz mapping)**.** *Given the recursion $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t) = \mathcal{P}_{\alpha\lambda}(\boldsymbol{z}_t - \alpha \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}))$, from Lemma A.1, there exist $B > 0$ such that loss $\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D})$ is μ-strongly convex $\forall t > B$. Hence, using Lemma 3.4,*

$$\|\nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D})\|_2 = \|(\boldsymbol{I} - \alpha \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D})) \partial \mathcal{P}_{\alpha\lambda}(\boldsymbol{z}_t)\|_2 \leq \rho \tag{20}$$

where $\rho \triangleq c_{prox}(1 - \alpha\mu) < 1$.

One key term, used in the proofs, is that $\mathbf{0} \in \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \partial h(\hat{\boldsymbol{z}})$ which is followed by the lasso optimality, i.e.,

Lemma A.3 (Lasso optimality)**.** *Lasso Karush-Kuhn-Tucker (KKT) optimality conditions are*

$$\hat{\boldsymbol{z}} \in \underset{\boldsymbol{z}\in\mathbb{R}^p}{\arg\min}\, f_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D}) \Leftrightarrow \boldsymbol{D}^T(\boldsymbol{x} - \boldsymbol{D}\hat{\boldsymbol{z}}) \in \lambda\partial\|\hat{\boldsymbol{z}}\|_1, \textit{and } \partial|\hat{\boldsymbol{z}}_{(j)}| = \begin{cases} \{sign(\hat{\boldsymbol{z}}_{(j)}\} & \textit{if } \hat{\boldsymbol{z}}_{(j)} \neq 0 \\ [-1, 1] & \textit{if } \hat{\boldsymbol{z}}_{(j)} = 0 \end{cases}, \forall j \in \{1, 2, \ldots, p\}. \tag{21}$$

A.3 Forward pass proof details

Given the μ-incoherence of $\boldsymbol{D}^*$, and current dictionary closeness of δ_l, we re-state Lemma 3.1 and proof it below. It shows that the current dictionary is μ_l-close to $\boldsymbol{D}^*$.

Lemma 3.1 (μ_l-incoherent)**.** *$\boldsymbol{D}^{(l)}$ is μ_l-incoherent where $\mu_l = \mu + 2\sqrt{m}\delta_l$.*

Proof.

$$\begin{aligned} \langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^{(l)} \rangle &= \langle \boldsymbol{D}_i^*, \boldsymbol{D}_j^* \rangle - \langle \boldsymbol{D}_i^* - \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^* \rangle - \langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^* - \boldsymbol{D}_j^{(l)} \rangle \\ |\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^{(l)} \rangle| &\leq \mu/\sqrt{m} + \|\boldsymbol{D}_i^* + \boldsymbol{D}_i^{(l)}\|_2 \|\boldsymbol{D}_j^*\|_2 + \|\boldsymbol{D}_i^{(l)}\|_2 \|\boldsymbol{D}_j^* - \boldsymbol{D}_j^{(l)}\|_2 \leq \mu/\sqrt{m} + 2\delta_l \end{aligned} \tag{22}$$

■

We re-state and proof the forward pass support recovery (Theorem 4.1). This shows that given proper initialization and under mild conditions, the support of the true code $\boldsymbol{z}^*$ is recovered with high probability in one iteration of the encoder.

Theorem 4.1 (Forward pass support recovery)**.** *Given Assumptions 3.3 and 3.4, suppose $\boldsymbol{D}^{(l)}$ is $\delta_l = \mathcal{O}^*(1/\sqrt{\log p})$ close to $\boldsymbol{D}^*$. If $s = \mathcal{O}^*(\sqrt{m}/\mu \log m)$, and $\mu = \mathcal{O}(\log m)$, then with high probablity of at least $1 - \epsilon^{(l)}_{supp\text{-}rec}$, the choice of $\lambda_0 = C_{\min}/4$ recovers the support of the code $\boldsymbol{z}^*$ in one encoder iteration, i.e., $sign(\mathcal{S}_{\alpha\lambda_0}(\alpha \boldsymbol{D}^{(l)T}\boldsymbol{x}) = sign(\boldsymbol{z}^*)$, where $\epsilon^{(l)}_{supp\text{-}rec} = 2p\exp(-\frac{C_{\min}^2}{\mathcal{O}^*(\delta_l^2)})$.*

Proof. The code estimate after one iteration is $\boldsymbol{z}_1 = \mathcal{P}_{\alpha\lambda}(\alpha \boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{x})\mathrm{ReLU}(\alpha(|\boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{D}^*\boldsymbol{z}^*| - \lambda_0))$. We focus on the positive entries. The analysis for negative entries is similar. Writting the relation for i-th entry,

$$\begin{aligned} \boldsymbol{z}_{1,(i)} &= \mathrm{sign}(\boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{x})\mathrm{ReLU}(\alpha(\sum_{j\in S^*} \langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^* \rangle \boldsymbol{z}^*_{(j)} - \lambda_0)) \\ &= \mathrm{ReLU}(\alpha(\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle \boldsymbol{z}^*_{(i)} + \sum_{j\in S^*\setminus\{i\}} \langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^* \rangle \boldsymbol{z}^*_{(j)} - \lambda_0)) \end{aligned} \tag{23}$$

We focus on the term inside ReLU and discard α, shared by all terms. We shows that under proper choice of λ_0, $\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle \boldsymbol{z}^*_{(i)}$ is greater than λ_0 and $\boldsymbol{v}_i = \sum_{j\in S^*\setminus\{i\}} \langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_j^* \rangle \boldsymbol{z}^*_{(j)}$ is small with respect to λ_0, hence getting cancelled by ReLU. The small value of $\boldsymbol{v}_i$, compared to $\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle \boldsymbol{z}^*_{(i)}$, results in $\mathrm{sign}(\boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{x})$ be equal to the $\mathrm{sign}(\boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{D}^*\boldsymbol{z}^*)$ which is equal to the sign of $\boldsymbol{z}^*$.

Given the current dictionary distance $\|\boldsymbol{D}_i^{(l)} - \boldsymbol{D}_i^*\|_2 \leq \delta_l$, we can find a lower bound on $\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle \boldsymbol{z}^*_{(i)}$ as follows

$$\begin{aligned} \langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle &= \frac{1}{2}(\|\boldsymbol{D}_i^*\|_2^2 + \|\boldsymbol{D}_i^{(l)}\|_2^2 - \|\boldsymbol{D}_i^{(l)} - \boldsymbol{D}_i^*\|_2^2) = 1 - \frac{1}{2}\|\boldsymbol{D}_i^{(l)} - \boldsymbol{D}_i^*\|_2^2 \\ |\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle| &\geq 1 - \delta_l^2/2 \end{aligned} \tag{24}$$

Hence, for $i \in S^*$

$$|\langle \boldsymbol{D}_i^{(l)}, \boldsymbol{D}_i^* \rangle \boldsymbol{z}^*_{(i)}| \geq (1 - \delta_l^2/2) C_{\min} \tag{25}$$

otherwise, it is 0. Given, $var(\boldsymbol{z}^*_{(i)}) = 1$ for $i \in S^*$, we find an upper bound on the variance $\boldsymbol{v}_i$ of as follows

$$\begin{aligned} var(\boldsymbol{v}_i) &= \sum_{j \in S^* \setminus \{i\}} \langle \boldsymbol{D}^{(l)}_i, \boldsymbol{D}^*_j \rangle^2 = \sum_{j \in S^* \setminus \{i\}} (\langle \boldsymbol{D}^*_i, \boldsymbol{D}^*_j \rangle + \langle \boldsymbol{D}^{(l)}_i - \boldsymbol{D}^*_i, \boldsymbol{D}^*_j \rangle)^2 \\ &\leq \sum_{j \in S^* \setminus \{i\}} 2(\langle \boldsymbol{D}^*_i, \boldsymbol{D}^*_j \rangle^2 + \langle \boldsymbol{D}^{(l)}_i - \boldsymbol{D}^*_i, \boldsymbol{D}^*_j \rangle^2) \leq \sum_{j \in S^* \setminus \{i\}} (2\mu^2/m) + 2\|(\boldsymbol{D}^{(l)}_i - \boldsymbol{D}^*_i)^{\mathrm{T}} \boldsymbol{D}^*_{S^* \setminus \{i\}}\|_2^2 \\ &\leq (2s\mu^2/m) + 2\|(\boldsymbol{D}^{(l)}_i - \boldsymbol{D}^*_i)\|_2^2 \|\boldsymbol{D}^*_{S^* \setminus \{i\}}\|_2^2 \leq 2(s\mu^2/m + 4\delta_l^2) = \mathcal{O}^*(\delta_l^2) \end{aligned} \tag{26}$$

where we used the Gershgorin Circle Theorem for the bound $\|\boldsymbol{D}^*_{S \setminus \{i\}}\|_2 \leq 2$. With the sub-Gaussian assumption on the coefficients $\boldsymbol{z}^*$, we get the following using Chernoff bound concerning $\boldsymbol{v}_i$.

$$P(|\boldsymbol{v}_i| \geq \frac{C_{\min}}{4}) \leq 2\exp(-\frac{C^2_{\min}}{4s\mu^2/m + 16\delta_l^2}) = 2\exp(-\frac{C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}) \tag{27}$$

Taking a union bound over all indices $i \in [1, p]$ will result in

$$P(\max_i |\boldsymbol{v}_i| \geq \frac{C_{\min}}{4}) \leq 2p\exp(-\frac{C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}) \coloneqq \epsilon^{(l)}_{\text{supp-rec}} \tag{28}$$

Hence, we can set $\lambda_0 = C_{\min}/2$. ■

We re-state and prove the forward pass support preservation (Theorem 4.2).

Theorem 4.2 (Forward pass support preservation)**.** *Given Assumptions 3.3 and 3.4, suppose $\boldsymbol{D}^{(l)}$ is $\delta_l = \mathcal{O}^*(1/\log p)$ close to $\boldsymbol{D}^*$. If $s = \mathcal{O}^*(\sqrt{m}/\mu \log m)$, $\mu = \mathcal{O}(\log m)$, and the regularizer and step size are chosen such that $\lambda^{(l)}_t = \frac{\mu_l}{\sqrt{m}}\|\boldsymbol{z}^* - \boldsymbol{z}_t\|_1 + a_\gamma = \boldsymbol{\Omega}(\frac{s \log m}{\sqrt{m}})$ and $\alpha^{(l)} \leq 1 - \frac{2\lambda^{(l)}_t - (1 - \frac{\delta_l^2}{2})C_{\min}}{\lambda^{(l)}_{t-1}}$, then with high probability of at least $1 - \epsilon^{(l)}_{\text{supp-pres}}$, the support, recovered at the first iteration, is preserved through the encoder iterations. We have $a_\gamma = \mathcal{O}(\sqrt{s\delta_l})$ and $\epsilon^{(l)}_{\text{supp-pres}} \coloneqq \epsilon^{(l)}_{\text{supp-rec}} + \epsilon^{(l)}_\gamma = 2p\exp(\frac{-C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}) + 2s\exp(\frac{-1}{\mathcal{O}(\delta_l)})$.*

Proof. Given current dictionary $\boldsymbol{D}^{(l)}$, in each iteration of the forward pass, we have $\boldsymbol{z}_{t+1} = \mathcal{P}_{\alpha\boldsymbol{\lambda}}(\boldsymbol{z}_t + \alpha \boldsymbol{D}^{\mathrm{T}}(\boldsymbol{D}^*\boldsymbol{z}^* - \boldsymbol{D}\boldsymbol{z}_t)$. We focus on the entires that are non-negative. Then procedure for negative code entries is similar. We follow similar steps as in (Rambhatla et al., 2018). We get

$$\begin{aligned} \boldsymbol{z}_{t+1,(j)} &= \mathrm{ReLU}((\boldsymbol{I} - \alpha \boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{D}^{(l)})_{(j,:)}\boldsymbol{z}_t + \alpha(\boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{D}^*)_{(j,:)}\boldsymbol{z}^* - \alpha\lambda^{(l)}_{t,j}) \\ &= \mathrm{ReLU}((\boldsymbol{I} - \alpha \boldsymbol{D}^{(l)\mathrm{T}}\boldsymbol{D}^{(l)})_{(j,:)}\boldsymbol{z}_t + \alpha((\boldsymbol{D}^{(l)} - \boldsymbol{D}^*)^{\mathrm{T}}\boldsymbol{D}^*)_{(j,:)}\boldsymbol{z}^* + \alpha(\boldsymbol{D}^{*\mathrm{T}}\boldsymbol{D}^*)_{(j,:)}\boldsymbol{z}^* - \alpha\lambda^{(l)}_{t,j}) \\ &= \mathrm{ReLU}((1-\alpha)\boldsymbol{z}_{t,(j)} - \alpha\sum_{i \neq j}\langle \boldsymbol{D}^{(l)}_j, \boldsymbol{D}^{(l)}_i \rangle \boldsymbol{z}_{t,(i)} + \alpha\langle (\boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j), \boldsymbol{D}^*_j \rangle \boldsymbol{z}^*_{(j)} \\ &+ \alpha\sum_{i \neq j}\langle \boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j, \boldsymbol{D}^*_i \rangle \boldsymbol{z}^*_{(i)} + \alpha\boldsymbol{z}^*_{(j)} + \alpha\sum_{i \neq j}\langle \boldsymbol{D}^*_j, \boldsymbol{D}^*_i \rangle \boldsymbol{z}^*_{(i)} - \alpha\lambda^{(l)}_{t,j}) \\ &= \mathrm{ReLU}((1-\alpha)\boldsymbol{z}_{t,(j)} + \alpha(1 - \beta^{(l)}_j)\boldsymbol{z}^*_{(j)} + \alpha\eta^{(l)}_{t,j} - \alpha\lambda^{(l)}_{t,j}) \end{aligned} \tag{29}$$

where $\beta^{(l)}_j = \langle \boldsymbol{D}^*_j - \boldsymbol{D}^{(l)}_j, \boldsymbol{D}^*_j \rangle$, and $\eta^{(l)}_{t,j} = -\sum_{i \neq j}\langle \boldsymbol{D}^{(l)}_j, \boldsymbol{D}^{(l)}_i \rangle \boldsymbol{z}_{t,(i)} + (\langle \boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j, \boldsymbol{D}^*_i \rangle + \langle \boldsymbol{D}^*_j, \boldsymbol{D}^*_i \rangle)\boldsymbol{z}^*_{(i)}$. With $\|\boldsymbol{D}^{(l)}_j - \boldsymbol{D}^*_j\|_2 \leq \delta_l$, $\beta^{(l)}_j$ can be bounded as follows

$$\beta^{(l)}_j = \langle \boldsymbol{D}^*_j - \boldsymbol{D}^{(l)}_j, \boldsymbol{D}^*_j \rangle \leq \delta_l^2/2 \tag{30}$$

where we used the relation $\|\boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*\|_2^2 = 2(1 - \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_j^* \rangle)$. We re-write $\eta_{t,j}^{(l)}$ below

$$\begin{aligned}
\eta_{t,j}^{(l)} &= -\sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle \boldsymbol{z}_{t,(i)} + (\langle \boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*, \boldsymbol{D}_i^* \rangle + \langle \boldsymbol{D}_j^*, \boldsymbol{D}_i^* \rangle) \boldsymbol{z}_{(i)}^* \\
&= -\sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle \boldsymbol{z}_{t,(i)} + \sum_{i\neq j} (\langle \boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*, \boldsymbol{D}_i^* \rangle + \langle \boldsymbol{D}_j^*, \boldsymbol{D}_i^* \rangle) \boldsymbol{z}_{(i)}^* + \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle \boldsymbol{z}_{(i)}^* - \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle \boldsymbol{z}_{(i)}^* \\
&= \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle (\boldsymbol{z}_{(i)}^* - \boldsymbol{z}_{t,(i)}) + \sum_{i\neq j} (\langle \boldsymbol{D}_j^{(l)} - \boldsymbol{D}_j^*, \boldsymbol{D}_i^* \rangle + \langle \boldsymbol{D}_j^*, \boldsymbol{D}_i^* \rangle - \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle) \boldsymbol{z}_{(i)}^* \\
&= \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle (\boldsymbol{z}_{(i)}^* - \boldsymbol{z}_{t,(i)}) + \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^* - \boldsymbol{D}_i^{(l)} \rangle \boldsymbol{z}_{(i)}^* \\
&= \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle (\boldsymbol{z}_{(i)}^* - \boldsymbol{z}_{t,(i)}) + \gamma_j^{(l)}
\end{aligned} \tag{31}$$

where $\gamma_j^{(l)} = \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^* - \boldsymbol{D}_i^{(l)} \rangle \boldsymbol{z}_{(i)}^*$. Given the sub-Gaussian entries of the code $\boldsymbol{z}^*$, we provide a bound on the variance of $\gamma_j^{(l)}$ below:

$$var(\gamma_j^{(l)}) = \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^* - \boldsymbol{D}_i^{(l)} \rangle^2 \leq s\delta_l^2 \tag{32}$$

Now, using Chernoff bound on the sub-Gaussian code entries, we get

$$P(|\gamma_j^{(l)}| > a) \leq 2\exp\left(\frac{-a^2}{2s\delta_l^2}\right) \tag{33}$$

To bound all the terms in the support, for $j \in S^*$, we have

$$P(\max |\gamma_j^{(l)}| > a_\gamma) \leq \epsilon_\gamma^{(l)} \tag{34}$$

where $\epsilon_\gamma^{(l)} = 2s\exp\left(\frac{-a_\gamma^2}{2s\delta_l^2}\right)$. Let $a_\gamma = \mathcal{O}(\sqrt{s\delta_l})$, then $\epsilon_\gamma^{(l)} = 2s\exp\left(\frac{-1}{\mathcal{O}(\delta_l)}\right)$. The above analysis states that with high probability of at least $1 - \epsilon_\gamma^{(l)}$, $|\gamma_j^{(l)}| \leq a_\gamma = \mathcal{O}(\sqrt{s\delta_l})$. Next, we write the recursion for when the support is identified (see Theorem 4.1). For the code at iteration T, we have

$$\begin{aligned}
\boldsymbol{z}_{T,(j)} &= (1-\alpha)^T \boldsymbol{z}_{0,(j)} + \boldsymbol{z}_{(j)}^* \sum_{t=1}^{T} \alpha(1-\beta_j^{(l)})(1-\alpha)^{T-t} + \sum_{t=1}^{T} \alpha(\eta_{t-1,j}^{(l)} - \lambda_{t-1,j}^{(l)})(1-\alpha)^{T-t} \\
&= (1-\alpha)^T \boldsymbol{z}_{0,(j)} + \boldsymbol{z}_{(j)}^* (1-\beta_j^{(l)})(1-(1-\alpha)^T) + \sum_{t=1}^{T} \alpha(\eta_{t-1,j}^{(l)} - \lambda_{t-1,j}^{(l)})(1-\alpha)^{T-t} \\
&= \boldsymbol{z}_{(j)}^* (1-\beta_j^{(l)}) + (1-\alpha)^T (\boldsymbol{z}_{0,(j)} - \boldsymbol{z}_{(j)}^* (1-\beta_j^{(l)})) + \sum_{t=1}^{T} \alpha(\eta_{t-1,j}^{(l)} - \lambda_{t-1,j}^{(l)})(1-\alpha)^{T-t} \\
&= \boldsymbol{z}_{(j)}^* (1-\beta_j^{(l)}) + \zeta_{T,j}^{(l)}
\end{aligned} \tag{35}$$

where $\zeta_{T,j}^{(l)} = (1-\alpha)^T (\boldsymbol{z}_{0,(j)} - \boldsymbol{z}_{(j)}^* (1-\beta_j^{(l)})) + \sum_{t=1}^{T} \alpha(\eta_{t-1,j}^{(l)} - \lambda_{t-1,j}^{(l)})(1-\alpha)^{T-t}$. With the support correctly identified at iteration $t-1$, we show that the support is preserved at iteration t. With $\|\boldsymbol{z}^* - \boldsymbol{z}_t\|_1 = \mathcal{O}(s)$, for each $j \in S^*$, we have

$$\eta_{t,j}^{(l)} = \sum_{i\neq j} \langle \boldsymbol{D}_j^{(l)}, \boldsymbol{D}_i^{(l)} \rangle (\boldsymbol{z}_{(i)}^* - \boldsymbol{z}_{t,(i)}) + \gamma_j^{(l)} \leq \frac{\mu_l}{\sqrt{m}} \|\boldsymbol{z}^* - \boldsymbol{z}_t\|_1 + a_\gamma = \mathcal{O}\left(\frac{s\log m}{\sqrt{m}}\right) \tag{36}$$

We make sure the regularizer is chosen such that

$$\lambda_t \geq \frac{\mu_l}{\sqrt{m}} \|\boldsymbol{z}^* - \boldsymbol{z}_t\|_1 + a_\gamma \tag{37}$$

We see that the larger the code error and coherence between the columns of the current dictionary, the larger λ_t should be. This is to suppress the noise component in the code recursion and make sure no false support is introduced. Furthermore, we want λ_t to be lower than half of the signal component, i.e.,

$$\begin{aligned} \alpha\lambda_t &\leq \frac{1-\alpha}{2}\boldsymbol{z}_{t,(j)} + \frac{\alpha}{2}(1-\beta_j^{(l)})\boldsymbol{z}^*_{(j)}, \forall j \in S^* \\ \alpha\lambda_t &\leq \frac{1-\alpha}{2}\boldsymbol{z}_t^{\min} + \frac{\alpha}{2}(1-\frac{\delta_l^2}{2})C_{\min} \end{aligned} \tag{38}$$

where $\boldsymbol{z}_t^{\min} = \min_j \boldsymbol{z}_{t,(j)}$. We further shrink the upper bound, given the code from previous iteration $t-1$ (i.e., $\alpha\lambda_{t-1} \leq \boldsymbol{z}_t^{\min}$). Hence, we want the regularizer to follow

$$\begin{aligned} \alpha\lambda_t &\leq \frac{1-\alpha}{2}\alpha\lambda_{t-1} + \frac{\alpha}{2}(1-\frac{\delta_l^2}{2})C_{\min} \\ \lambda_t &\leq \frac{1-\alpha}{2}\lambda_{t-1} + \frac{1}{2}(1-\frac{\delta_l^2}{2})C_{\min} \end{aligned} \tag{39}$$

This condition is to make sure the identified supports are not killed in the recursion. We use the condition to set the step size α. We get

$$\alpha \leq 1 - \frac{2\lambda_t - (1-\frac{\delta_l^2}{2})C_{\min}}{\lambda_{t-1}} \tag{40}$$

Hence, $\lambda_t = \boldsymbol{\Omega}(\frac{s\log m}{\sqrt{m}})$ and α should be chosen sufficiently small such that the condition above is met. We denote $\epsilon^{(l)}_{\text{supp-pres}} := \epsilon^{(l)}_{\text{supp-rec}} + \epsilon^{(l)}_{\gamma} = 2p\exp\left(\frac{-C^2_{\min}}{\mathcal{O}^*(\delta_l^2)}\right) + 2s\exp\left(\frac{-1}{\mathcal{O}(\delta_l)}\right)$. Hence, with high probability of at least $1-\epsilon^{(l)}_{\text{supp-pres}}$, the support, recovered at the first iteration, is preserved through the encoder iterations. ■

Theorem 4.1 and Theorem 4.2 allow to achieve linear convergence in the forward pass right after the first encoder iteration, i.e., $B = 1$. With support recovery at first iteration and its preservation, we now re-state the forward pass code convergence (Theorem 4.3).

Theorem 4.3 (Local forward pass code convergence)**.** *Given the encoder $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$, Assumption 3.6, Lemmas 3.2, A.1 and A.2, then $\exists\ \rho < 1, B > 0$ s.t. $\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2 \leq \mathcal{O}(\rho^t)\ \forall t > B$, where $\hat{\boldsymbol{z}}$ is the unique minimizer of lasso (1). Furthermore, given Theorem 4.1 and Theorem 4.2, $B = 1$.*

Proof. Given the support selection at iteration B, from Lemma A.1, we have $\nabla^2_{11}\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) \succeq \mu\boldsymbol{I}$ restricted to the support for $t > B$. Then, from Lemma A.2, we get

$$\|\nabla_1\Phi(\boldsymbol{z}_t, \boldsymbol{D})\|_2 = \|(\boldsymbol{I} - \alpha\nabla^2_{11}\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}))\partial\mathcal{P}_{\alpha\lambda}(\boldsymbol{z}_t)\|_2 \leq \rho$$

where $\rho \triangleq c_{\text{prox}}(1-\alpha\mu) < 1$. Hence, using fixed-point property (Lemma 3.2)

$$\exists\ B > 0,\ \text{s.t.}\ \|\boldsymbol{z}_{t+1} - \hat{\boldsymbol{z}}\|_2 = \|\Phi(\boldsymbol{z}_t) - \Phi(\hat{\boldsymbol{z}})\|_2 \leq \rho\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2\ \forall t > B$$

where $\hat{\boldsymbol{z}} = \arg\min f_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$. Unrolling the recursion,

$$\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2 \leq \rho^{t-B}\|\boldsymbol{z}_B - \hat{\boldsymbol{z}}\|_2.$$

■

We now re-state the forward pass Jacobian (Theorem 4.7) convergence.

Theorem 4.7 (Local forward pass Jacobian convergence)**.** *Given the recursion $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$, and $\hat{\boldsymbol{z}}$ the unique minimizer of lasso with Jacobian $\hat{\boldsymbol{J}}$, then $\exists\ \rho < 1, B > 0$ s.t. $\|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 \leq \mathcal{O}(t\rho^t)\ \forall t > B$. Furthermore, given Theorem 4.1 and Theorem 4.2, $B = 1$.*

Proof. Differentiating the recursion,

$$\boldsymbol{J}_{t+1} = \nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D})^{\mathrm{T}} \boldsymbol{J}_t + \nabla_2 \Phi(\boldsymbol{z}_t, \boldsymbol{D})^{\mathrm{T}}.$$

Similarly,

$$\hat{\boldsymbol{J}} = \nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}} \hat{\boldsymbol{J}} + \nabla_2 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}}$$

where $\hat{\boldsymbol{z}}$ is a minimizer of lasso and fixed-point of the mapping (see Lemma 3.2). Subtract the terms

$$\boldsymbol{J}_{t+1} - \hat{\boldsymbol{J}} = \nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D})^{\mathrm{T}} (\boldsymbol{J}_t - \hat{\boldsymbol{J}}) + (\nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D}))^{\mathrm{T}} \hat{\boldsymbol{J}} + (\nabla_2 \Phi(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_2 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D}))^{\mathrm{T}}$$

Given the Lipschitz properties of $\mathcal{L}$ and h, we can further get the upper bounds on $\|\nabla_1 \Phi(\boldsymbol{a}, \boldsymbol{D}) - \nabla_1 \Phi(\boldsymbol{b}, \boldsymbol{D})\|_2 \leq L_{\Phi_1} \|\boldsymbol{b} - \boldsymbol{a}\|_2$ and $\|\nabla_2 \Phi(\boldsymbol{a}, \boldsymbol{D}) - \nabla_2 \Phi(\boldsymbol{b}, \boldsymbol{D})\|_2 \leq L_{\Phi_2} \|\boldsymbol{b} - \boldsymbol{a}\|_2$. Hence, with upper bound on the norm of Jacobian (Assumption 4.1), there exists $B > 0$ such that $\forall t > B$

$$\begin{aligned}
\|\boldsymbol{J}_{t+1} - \hat{\boldsymbol{J}}\|_2 &\leq \|\nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D})\|_2 \|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 + \|\nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 \|\hat{\boldsymbol{J}}\|_2 \\
&\quad + \|\nabla_2 \Phi(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_2 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 \\
&\leq \rho \|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 + c \|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2
\end{aligned}$$

where $c \triangleq M_J L_{\Phi_1} + L_{\Phi_2}$. Hence,

$$\|\boldsymbol{J}_{t+1} - \hat{\boldsymbol{J}}\|_2 \leq \rho \|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 + \mathcal{O}(\rho^t).$$

Unrolling the recursion,

$$\|\boldsymbol{J}_{t+1} - \hat{\boldsymbol{J}}\|_2 \leq \mathcal{O}((t+1)\rho^t).$$

■

Theorem 4.6 (Global forward pass code error)**.** *Let $\hat{\boldsymbol{z}}$ be the fixed-point of the encoder with iterations $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$. Given Assumption 3.6, Lemmas 3.2, A.1 and A.2, we have $\|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2 \leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^*)$, where $\hat{\delta}^* = \|\hat{\boldsymbol{z}}^* - \boldsymbol{z}^*\|_2$, $\hat{\boldsymbol{z}}$ is the unique minimizer of lasso (1) given the dictionary $\boldsymbol{D}$, $\hat{\boldsymbol{z}}^*$ is the unique minimizer of lasso (1) given the dictionary $\boldsymbol{D}^*$, and $\boldsymbol{z}^*$ is the ground-truth code.*

Proof. We first find the error between $\hat{\boldsymbol{z}}$ and $\hat{\boldsymbol{z}}^*$ which is the unique minimizer of lasso (1) given the true dictionary $\boldsymbol{D}^*$. Using fixed-point property (Lemma 3.2), we get

$$\|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 = \|\Phi(\hat{\boldsymbol{z}}, \boldsymbol{D}) - \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}^*)\|_2 \leq \|\Phi(\hat{\boldsymbol{z}}, \boldsymbol{D}) - \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D})\|_2 + \|\Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}) - \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}^*)\|_2 \tag{41}$$

Using the μ-strongly convexity of $\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D})$ on the support, and L_{21} Lipschitz constants of $\nabla^2_{21} \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$, we upper bound the term as follows:

$$\|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 \leq \rho \|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 + \alpha L_{21} c_{\text{prox}} \|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \tag{42}$$

Where $\rho \triangleq c_{\text{prox}}(1 - \alpha\mu) < 1$. Denote $q \triangleq \frac{\alpha c_{\text{prox}} L_{21}}{1-\rho}$ which can be made to be small with proper choice of step size α.

$$\|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 \leq q \|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \tag{43}$$

Hence, we get the following code error

$$\|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2 \leq \|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 + \|\hat{\boldsymbol{z}}^* - \boldsymbol{z}^*\|_2 \leq q \|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^* \leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^*) \tag{44}$$

■

Theorem 4.8 (Global forward pass Jacobian error)**.** *Let $\hat{\boldsymbol{z}}$ be the fixed-point of the encoder with iterations $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$. Given Assumption 3.6, Lemmas 3.2, A.1 and A.2, we have $\|\hat{\boldsymbol{J}} - \boldsymbol{J}^*\|_2 \leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}^*_J)$, where $\hat{\delta}^*_J \coloneqq \|\hat{\boldsymbol{J}}^* - \boldsymbol{J}^*\|_2$, and $\hat{\boldsymbol{J}}$, $\hat{\boldsymbol{J}}^*$ and $\boldsymbol{J}^*$ are Jacobians corresponding to $\hat{\boldsymbol{z}}$, $\hat{\boldsymbol{z}}^*$ and $\boldsymbol{z}^*$.*

Proof. First, we define $\boldsymbol{J}^*$. For $\boldsymbol{z}^*$, we define the mapping function $\boldsymbol{z} \to \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$, where $\boldsymbol{z}^*(\boldsymbol{D})$ is its minimizer evaluated at $\boldsymbol{D}^*$, i.e., $\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) = \boldsymbol{D}^{*\mathrm{T}}(\boldsymbol{D}^*\boldsymbol{z}^* - \boldsymbol{x}) = 0$ given the generative model ($\boldsymbol{x} = \boldsymbol{D}^*\boldsymbol{z}^*$). Hence, we define the Jacobian $\boldsymbol{J}^* = \frac{\partial \boldsymbol{z}^*(\boldsymbol{D})}{\partial \boldsymbol{D}}|_{\boldsymbol{D}=\boldsymbol{D}^*}$. From implicit function theorem, we get

$$\boldsymbol{J}^{*+}\nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) + \nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) = \boldsymbol{0}$$

which is later used in the global backward pass analysis. Alternatively, if $\nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)$ is invertible, then we can compute $\boldsymbol{J}^{*+}$ as follows:

$$\boldsymbol{J}^{*+} = -\nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)^{-1}$$

The Jacobian w.r.t row i of the dictionary is

$$\boldsymbol{J}^*_{(i,:)} = -(\boldsymbol{D}_{S^*}^{*\mathrm{T}}\boldsymbol{D}_{S^*}^*)^{-1}(\boldsymbol{D}_{i,:}^*\boldsymbol{z}^{*\mathrm{T}} + (\boldsymbol{D}_{i,:}^{*\mathrm{T}}\boldsymbol{z}^* - \boldsymbol{x}_i)\boldsymbol{I}_p)_{S^*}$$

on the support S^* of $\boldsymbol{z}^*$. Outside of the support, it is zero. Now, given the recursion $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$, we differentiate the recursion,

$$\boldsymbol{J}_{t+1} = \nabla_1 \Phi(\boldsymbol{z}_t, \boldsymbol{D})^{\mathrm{T}}\boldsymbol{J}_t + \nabla_2 \Phi(\boldsymbol{z}_t, \boldsymbol{D})^{\mathrm{T}}.$$

Hence, we have

$$\hat{\boldsymbol{J}} = \nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}}\hat{\boldsymbol{J}} + \nabla_2 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}}$$

$$\hat{\boldsymbol{J}}^* = \nabla_1 \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}^*)^{\mathrm{T}}\hat{\boldsymbol{J}}^* + \nabla_2 \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}^*)^{\mathrm{T}}$$

where $\hat{\boldsymbol{J}}^*$ is the Jacobian of $\hat{\boldsymbol{z}}^*$. Then, following similar step to Theorem 4.7, we can write

$$\hat{\boldsymbol{J}} - \hat{\boldsymbol{J}}^* = \nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}}(\hat{\boldsymbol{J}} - \hat{\boldsymbol{J}}^*) + (\nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D}) - \nabla_1 \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}^*))^{\mathrm{T}}\hat{\boldsymbol{J}}^* + (\nabla_2 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D}) - \nabla_2 \Phi(\hat{\boldsymbol{z}}^*, \boldsymbol{D}^*))^{\mathrm{T}}$$

With respect to $\boldsymbol{D}$, we denote the Lipschitz constants of $\nabla_1 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})$ and $\nabla_2 \Phi(\hat{\boldsymbol{z}}, \boldsymbol{D})$ with $L_{\Phi_{1D}}$ and $L_{\Phi_{2D}}$, respectively. Then,

$$\|\hat{\boldsymbol{J}} - \hat{\boldsymbol{J}}^*\|_2 \leq \rho\|\hat{\boldsymbol{J}} - \hat{\boldsymbol{J}}^*\|_2 + c\|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 + c_D\|\hat{\boldsymbol{D}} - \boldsymbol{D}^*\|_2$$

where $c \triangleq M_J L_{\Phi_1} + L_{\Phi_2}$ and $c_D \triangleq M_J L_{\Phi_{1D}} + L_{\Phi_{2D}}$. Given the global forward pass code error, we get

$$\|\hat{\boldsymbol{J}} - \hat{\boldsymbol{J}}^*\|_2 \leq q_z\|\hat{\boldsymbol{z}} - \hat{\boldsymbol{z}}^*\|_2 + q_D\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \leq (q_D + q_z q)\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \tag{45}$$

where $q_z \triangleq \frac{c}{1-\rho}$, $q_D \triangleq \frac{c_D}{1-\rho}$. Hence, we get

$$\|\hat{\boldsymbol{J}} - \boldsymbol{J}^*\|_2 \leq \|\hat{\boldsymbol{J}} - \hat{\boldsymbol{J}}^*\|_2 + \|\hat{\boldsymbol{J}}^* - \boldsymbol{J}^*\|_2 \leq \mathcal{O}(\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \hat{\delta}_J^*) \tag{46}$$

where we denote $\hat{\delta}_J^* := \|\hat{\boldsymbol{J}}^* - \boldsymbol{J}^*\|_2$ ■

A.4 Local backward pass proof details

In each update of the dictionary, we bound the gradient approximations as function of unrolling t (Theorem 4.9). This shows that $\boldsymbol{g}_t^{\text{ae-lasso}}$ converges faster than $\boldsymbol{g}_t^{\text{dec}}$ and $\boldsymbol{g}_t^{\text{ae-ls}}$, and the latter is a biased estimator of $\hat{\boldsymbol{g}}$. This is followed by Theorem 4.9 showing the order magnitude of the bounds is indeed tight.

Theorem 4.9 (Local convergence of gradients)**.** *Given the convergence results from the forward pass (Theorems 4.3 and 4.7), $\exists\ \rho < 1, B > 0$ such that $\forall t > B$, the errors of gradients defined in Algorithm 2 w.r.t $\hat{\boldsymbol{g}}$ (4) satisfy*

$$\begin{aligned} \|\boldsymbol{g}_t^{dec} - \hat{\boldsymbol{g}}\|_2 &\leq \mathcal{O}(\rho^t) \\ \|\boldsymbol{g}_t^{ae\text{-}lasso} - \hat{\boldsymbol{g}}\|_2 &\leq \mathcal{O}(t\rho^{2t}) \\ \|\boldsymbol{g}_t^{ae\text{-}ls} - \hat{\boldsymbol{g}}\|_2 &\leq \mathcal{O}(t\rho^{2t} + M_J\lambda\sqrt{s}). \end{aligned} \tag{12}$$

Moreover, the order of upper bounds is tight (see Lemma A.4).

Proof. For $\boldsymbol{g}_t^{\text{dec}}$, with the infinite fresh samples, we have $\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\nabla_2\mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_t^i,\boldsymbol{D}) = \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t,\boldsymbol{D})\right]$ a.s. Based on Lemma 3.3, we get

$$\begin{aligned}\|\boldsymbol{g}_t^{\text{dec}}-\hat{\boldsymbol{g}}\|_2 &= \|\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t,\boldsymbol{D})\right]-\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\nabla_2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}},\boldsymbol{D})\right]\|_2\\ &\le \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\|\nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t,\boldsymbol{D})-\nabla_2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}},\boldsymbol{D})\|_2\right]\le\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[L_2\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2\right]\le\mathcal{O}(\rho^t).\end{aligned}\tag{47}$$

Similarly, for $\boldsymbol{g}_t^{\text{ae-lasso}}$ and $\boldsymbol{g}_t^{\text{ae-ls}}$, we replace the sample mean for gradient computations with expectation in their limit. We re-write the gradient estimation error as following

$$\begin{aligned}\boldsymbol{g}_t^{\text{ae-lasso}}-\hat{\boldsymbol{g}} &= \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[Q(\hat{\boldsymbol{z}},\boldsymbol{J}_t)(\boldsymbol{z}_t-\hat{\boldsymbol{z}})\right]+\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[Q_t^{21}(\hat{\boldsymbol{z}})\right]+\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\boldsymbol{J}_tQ_t^{\text{lasso-11}}(\hat{\boldsymbol{z}})\right]\\ \boldsymbol{g}_t^{\text{ae-ls}}-\hat{\boldsymbol{g}} &= \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[Q(\hat{\boldsymbol{z}},\boldsymbol{J}_t)(\boldsymbol{z}_t-\hat{\boldsymbol{z}})\right]+\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[Q_t^{21}(\hat{\boldsymbol{z}})\right]+\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[\boldsymbol{J}_tQ_t^{\text{ls-11}}(\hat{\boldsymbol{z}})\right]\end{aligned}\tag{48}$$

where

$$\begin{aligned}Q_t^{21}(\boldsymbol{z}) &\triangleq \nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t,\boldsymbol{D})-\nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z},\boldsymbol{D})-\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z},\boldsymbol{D})(\boldsymbol{z}_t-\boldsymbol{z})\\ Q_t^{\text{lasso-11}}(\boldsymbol{z}) &\triangleq \nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t,\boldsymbol{D})+\partial h(\boldsymbol{z}_t)-\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z},\boldsymbol{D})(\boldsymbol{z}_t-\boldsymbol{z})\\ Q_t^{\text{ls-11}}(\boldsymbol{z}) &\triangleq \nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t,\boldsymbol{D})-\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z},\boldsymbol{D})(\boldsymbol{z}_t-\boldsymbol{z})\\ Q(\boldsymbol{z},\boldsymbol{J}) &\triangleq \boldsymbol{J}^+\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z},\boldsymbol{D})+\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z},\boldsymbol{D}).\end{aligned}\tag{49}$$

We provide bounds on the above in Lemma A.5. Hence, it suffices to bound the terms on the *r.h.s* as follows:

$$\|\boldsymbol{g}_t^{\text{ae-lasso}}-\hat{\boldsymbol{g}}\|_2\le\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[L_1\|\boldsymbol{J}_t-\hat{\boldsymbol{J}}\|_2\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2+({}^{L_{21}}/_2)\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2^2+M_J({}^{L_{11}}/_2)\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2^2\right].\tag{50}$$

Using the convergence errors from the forward pass (Theorems 4.3 and 4.7),

$$\|\boldsymbol{g}_t^{\text{ae-lasso}}-\hat{\boldsymbol{g}}\|_2\le L_1\mathcal{O}(t\rho^{2t})+({}^{L_{21}}/_2+M_J({}^{L_{11}}/_2))\,\mathcal{O}(\rho^{2t})=\mathcal{O}(t\rho^{2t}).\tag{51}$$

Similarly,

$$\|\boldsymbol{g}_t^{\text{ae-ls}}-\hat{\boldsymbol{g}}\|_2\le\mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\left[L_1\|\boldsymbol{J}_t-\hat{\boldsymbol{J}}\|_2\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2+({}^{L_{21}}/_2)\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2^2+M_J(({}^{L_{11}}/_2)\|\boldsymbol{z}_t-\hat{\boldsymbol{z}}\|_2^2+\|\partial h(\hat{\boldsymbol{z}})\|_2)\right].\tag{52}$$

Using the convergence errors from the forward pass (Theorems 4.3 and 4.7),

$$\|\boldsymbol{g}_t^{\text{ae-ls}}-\hat{\boldsymbol{g}}\|_2\le L_1\mathcal{O}(t\rho^{2t})+({}^{L_{21}}/_2+{}^{M_JL_{11}}/_2)\,\mathcal{O}(\rho^{2t})+M_J\|\partial h(\hat{\boldsymbol{z}})\|_2=\mathcal{O}(t\rho^{2t}+M_J\lambda\sqrt{s}).\tag{53}$$

■

Lemma A.4 (Tight local bound)**.** *The order magnitude of the upper bounds in Theorem 4.9 is tight.*

Proof. It is sufficient to show that there exist an example such that its forward pass code and Jacobian convergences are $\mathcal{O}(\rho^t)$ and $\mathcal{O}(t\rho^t)$, respectively. The following example confirms this. Without loss of generality, let $\boldsymbol{z}^*$ be 1-sparse and non-negative, $\boldsymbol{D}=\boldsymbol{D}^*$ and $\boldsymbol{D}_j=\boldsymbol{0}$ for $j\neq i$. The loss function is $\frac{1}{2}\|\boldsymbol{D}_i^*\boldsymbol{z}_{(i)}^*-\boldsymbol{D}_i\boldsymbol{z}_{(i)}\|_2^2+\lambda|\boldsymbol{z}_{(i)}|$. Given the support recovery after first iteration, the encoder forward pass implements $\boldsymbol{z}_{t+1,(i)}=\boldsymbol{z}_{t,(i)}-\alpha(\boldsymbol{D}_i^{\mathrm{T}}(\boldsymbol{D}_i\boldsymbol{z}_{t,(i)}-\boldsymbol{D}_i^*\boldsymbol{z}_{(i)}^*)+\lambda)=(1-\alpha)\boldsymbol{z}_{t,(i)}+\alpha(\boldsymbol{z}_{(i)}^*-\lambda)$. Hence, the forward pass convergences are

$$\begin{aligned}\boldsymbol{z}_{t,(i)} &= (1-\alpha)^t\boldsymbol{z}_0+\sum_{k=1}^{t}\alpha(1-\alpha)^{t-k}(\boldsymbol{z}_{(i)}^*-\lambda)=(1-\alpha)^t\boldsymbol{z}_0+(1-(1-\alpha)^t)(\boldsymbol{z}_{(i)}^*-\lambda)\\ \boldsymbol{z}_{t,(i)}-\hat{\boldsymbol{z}}_{(i)} &= \rho^t(\boldsymbol{z}_0-\boldsymbol{z}_{(i)}^*+\lambda)=\mathcal{O}(\rho^t)\end{aligned}\tag{54}$$

and

$$\begin{aligned}\boldsymbol{J}_{t,(i)} &= \boldsymbol{J}_{t-1,(i)}-\alpha(\boldsymbol{J}_{t-1,(i)}+2\boldsymbol{D}_i\boldsymbol{z}_{t,(i)}-\boldsymbol{D}_i^*\boldsymbol{z}_{(i)}^*)=\rho\boldsymbol{J}_{t-1,(i)}+\mathcal{O}(\rho^t)+\hat{\boldsymbol{J}}_{(i)}\\ \boldsymbol{J}_{t,(i)}-\hat{\boldsymbol{J}}_{(i)} &= \rho^t\boldsymbol{J}_{0,(i)}+\sum_{k=1}^{t}\mathcal{O}(\rho^t)=\mathcal{O}(t\rho^t)\end{aligned}\tag{55}$$

where $\rho=1-\alpha$, $\hat{\boldsymbol{z}}_{(i)}=\boldsymbol{z}_{(i)}^*-\lambda$, and $\hat{\boldsymbol{J}}_{(i)}=\alpha(2\boldsymbol{D}_i\hat{\boldsymbol{z}}_{(i)}-\boldsymbol{D}_i^*\boldsymbol{z}_{(i)}^*)$ ■

Lemma A.5 (Local bounds)**.** *From local gradient errors in Theorem 4.9, the following are satisfied*

$$\begin{aligned} \|Q_t^{21}(\hat{\boldsymbol{z}})\|_2 &\leq ({}^{L_{21}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2, & \|Q_t^{lasso\text{-}11}(\hat{\boldsymbol{z}})\|_2 &\leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 \\ \|Q(\hat{\boldsymbol{z}}, \boldsymbol{J}_t)\|_2 &\leq L_1\|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2, & \|Q_t^{ls\text{-}11}(\hat{\boldsymbol{z}})\|_2 &\leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\partial h(\hat{\boldsymbol{z}})\|_2. \end{aligned} \tag{56}$$

Proof. For $Q_t^{21}(\hat{\boldsymbol{z}})$, given convexity of $\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ and its domain (Assumption 3.1) and Lemma 3.3, we achieve the quadratic upper bound. For $Q_t^{\text{lasso-11}}(\hat{\boldsymbol{z}})$, we add and subtract $\nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})$, and then use quadratic upper bound. At line four, given Lemma A.3, we use $\boldsymbol{0} \in \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \partial h(\hat{\boldsymbol{z}})$ and assume that $\boldsymbol{z}_t$ recovers the sign entries of $\hat{\boldsymbol{z}}$.

$$\begin{aligned} \|Q_t^{\text{lasso-11}}\|_2 &= \|\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) + \partial h(\boldsymbol{z}_t) - \nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})(\boldsymbol{z}_t - \hat{\boldsymbol{z}})\| \\ &= \|\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \partial h(\boldsymbol{z}_t) - \nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})(\boldsymbol{z}_t - \hat{\boldsymbol{z}})\| \\ &\leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\partial h(\boldsymbol{z}_t) + \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 \\ &\leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\partial h(\boldsymbol{z}_t) - \partial h(\hat{\boldsymbol{z}})\|_2 \leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2. \end{aligned} \tag{57}$$

Similarly,

$$\begin{aligned} \|Q_t^{\text{ls-11}}\|_2 &= \|\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})(\boldsymbol{z}_t - \hat{\boldsymbol{z}})\| \\ &= \|\nabla_1\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) - \nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})(\boldsymbol{z}_t - \hat{\boldsymbol{z}})\| \\ &\leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\nabla_1\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 \leq ({}^{L_{11}}/_{2})\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\partial h(\hat{\boldsymbol{z}})\|_2. \end{aligned} \tag{58}$$

For $Q(\hat{\boldsymbol{z}}, \boldsymbol{J}_t)$, from implicit function theorem, $Q(\hat{\boldsymbol{z}}, \hat{\boldsymbol{J}}) = \boldsymbol{0}$ under the support S of $\hat{\boldsymbol{z}}$ that is identified by $\boldsymbol{z}_t$. To prove this, consider the minimizer $\hat{\boldsymbol{z}}(\boldsymbol{D})$. We have $\boldsymbol{0} \in \nabla_1 f(\hat{\boldsymbol{z}}, \boldsymbol{D})$, hence, we get $\boldsymbol{0} \in \hat{\boldsymbol{J}}(\boldsymbol{D})\nabla_{11}^2 f(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \nabla_{21}^2 f(\hat{\boldsymbol{z}}, \boldsymbol{D})$. Given the support recovery, the relation $\hat{\boldsymbol{J}}(\boldsymbol{D})(\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) \odot \boldsymbol{1}_{S^*}) + \nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) \odot \boldsymbol{1}_{S^*} = \boldsymbol{0}$ also holds which is equivalent to $Q(\hat{\boldsymbol{z}}, \hat{\boldsymbol{J}})$ under the support. To show this, given the recursion $\boldsymbol{z}_{t+1} = \Phi(\boldsymbol{z}_t, \boldsymbol{D})$, we differentiate it and get $\boldsymbol{J}_{t+1} = \nabla_1\Phi(\boldsymbol{z}_t, \boldsymbol{D})\boldsymbol{J}_t + \nabla_2\Phi(\boldsymbol{z}_t, \boldsymbol{D})$. Given the support recovery and fixed-point property, we can write

$$\begin{aligned} \hat{\boldsymbol{J}} &= \boldsymbol{1}_S \odot (\hat{\boldsymbol{J}} - \alpha\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}}\hat{\boldsymbol{J}}) + \boldsymbol{1}_S \odot (-\alpha\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}}) \\ \hat{\boldsymbol{J}} - \boldsymbol{1}_S \odot \hat{\boldsymbol{J}} &= -\hat{\boldsymbol{J}}\alpha\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}} \odot \boldsymbol{1}_S - \alpha\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})^{\mathrm{T}} \odot \boldsymbol{1}_S \\ \boldsymbol{0} &= \hat{\boldsymbol{J}}^+(\nabla_{11}^2\mathcal{L}(\hat{\boldsymbol{z}}, \boldsymbol{D}) \odot \boldsymbol{1}_S) + \nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) \odot \boldsymbol{1}_S \end{aligned} \tag{59}$$

If the term $(\nabla_{11}^2\mathcal{L}(\boldsymbol{z}_t, \boldsymbol{D}) \odot \boldsymbol{1}_S)$ is invertible, then we can write

$$\hat{\boldsymbol{J}}^+ = -\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) \odot \boldsymbol{1}_S(\nabla_{11}^2\mathcal{L}(\hat{\boldsymbol{z}}, \boldsymbol{D}) \odot \boldsymbol{1}_S)^{-1} \tag{60}$$

For the Jacobian corresponding to row i of the dictionary, we get

$$\hat{\boldsymbol{J}}_{(i,:)} = -(\boldsymbol{D}_S^{\mathrm{T}}\boldsymbol{D}_S)^{-1}(\boldsymbol{D}_{i,:}\hat{\boldsymbol{z}}^{\mathrm{T}} + (\boldsymbol{D}_{i,:}^{\mathrm{T}}\hat{\boldsymbol{z}} - \boldsymbol{x}_i)\boldsymbol{I}_p)_S \tag{61}$$

on the support. Outside of the support S, the Jacobian is zero. This proof is similarly provided by Malézieux et al. (2022). Hence, we can use $\nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) = -\hat{\boldsymbol{J}}^+\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})$ under the support S in the following

$$\begin{aligned} \|Q(\hat{\boldsymbol{z}}, \boldsymbol{J}_t)\|_2 &= \|\boldsymbol{J}_t^+\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) + \nabla_{21}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 = \|\boldsymbol{J}_t^+\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D}) - \hat{\boldsymbol{J}}^+\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 \\ &\leq \|(\boldsymbol{J}_t - \hat{\boldsymbol{J}})^+\nabla_{11}^2\mathcal{L}_{\boldsymbol{x}}(\hat{\boldsymbol{z}}, \boldsymbol{D})\|_2 \leq L_1\|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2. \end{aligned} \tag{62}$$

■

A.5 Global backward pass proof details

We re-state and proof Theorem 4.10 as follows:

Theorem 4.10 (Global error of gradients)**.** *Given the convergence results from the forward pass, (Theorems 4.6 and 4.8), the errors of gradients defined in Algorithm 2 w.r.t global direction $\boldsymbol{g}^*$ (defined in (5)) satisfy*

$$\begin{aligned}\|\boldsymbol{g}_\infty^{ae\text{-}lasso} - \boldsymbol{g}^*\|_2 &\leq \mathcal{O}(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2^2 + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^* + \hat{\delta}_J^* + M_J\lambda\sqrt{s})\\ \|\boldsymbol{g}_\infty^{ae\text{-}ls} - \boldsymbol{g}^*\|_2 &\leq \mathcal{O}(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2^2 + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^* + \hat{\delta}_J^*).\end{aligned} \tag{13}$$

Proof. For $\boldsymbol{g}_t^{\text{dec}}$, we compute the gradient in their limit assuming infinite fresh samples $\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^n \nabla_2\mathcal{L}_{\boldsymbol{x}^i}(\boldsymbol{z}_t^i, \boldsymbol{D}) = \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [\nabla_2\mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D})]$ a.s.. Similar to Theorem 4.9, we re-write the errors of gradients $\boldsymbol{g}_t^{\text{ae-lasso}}$ and $\boldsymbol{g}_t^{\text{ae-ls}}$ as following

$$\begin{aligned}\boldsymbol{g}_t^{\text{ae-lasso}} - \boldsymbol{g}^* &= \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [Q(\boldsymbol{z}^*, \boldsymbol{J}_t)(\boldsymbol{z}_t - \boldsymbol{z}^*)] + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [Q_t^{21}(\boldsymbol{z}^*)] + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [\boldsymbol{J}_t Q_t^{\text{lasso-}11}(\boldsymbol{z}^*)]\\ \boldsymbol{g}_t^{\text{ae-ls}} - \boldsymbol{g}^* &= \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [Q(\boldsymbol{z}^*, \boldsymbol{J}_t)(\boldsymbol{z}_t - \boldsymbol{z}^*)] + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [Q_t^{21}(\boldsymbol{z}^*)] + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [\boldsymbol{J}_t Q_t^{\text{ls-}11}(\boldsymbol{z}^*)].\end{aligned} \tag{63}$$

where $Q_t^{21}(\boldsymbol{z})$, $Q_t^{\text{lasso-}11}(\boldsymbol{z})$, $Q_t^{\text{ls-}11}(\boldsymbol{z})$, and $Q(\boldsymbol{z}, \boldsymbol{J})$ are defined as in Theorem 4.9. Given Assumption 4.1 and Lemma A.6, we find an upper bound on the *r.h.s* of the gradient errors as follows:

$$\begin{aligned}\|\boldsymbol{g}_t^{\text{ae-lasso}} - \boldsymbol{g}^*\|_2 &\leq \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [(L_1\|\boldsymbol{J}_t - \boldsymbol{J}^*\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2 + ({}^{L_{21}}/_2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2]\\ &\quad + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [M_J({}^{L_{11}}/_2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + M_J\|\partial h(\boldsymbol{z}_t)\|_2 + L_{1D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2]\\ &\leq \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [L_1(\|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 + \|\hat{\boldsymbol{J}} - \boldsymbol{J}^*\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2)(\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2 + \|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2)]\\ &\quad + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [({}^{L_{21}}/_2)(\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2^2) + L_{1D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2]\\ &\quad + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [M_J({}^{L_{11}}/_2)(\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2^2) + M_J\|\partial h(\boldsymbol{z}_t)\|_2]\end{aligned} \tag{64}$$

Similarly,

$$\begin{aligned}\|\boldsymbol{g}_t^{\text{ae-ls}} - \boldsymbol{g}^*\|_2 &\leq \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [(L_1\|\boldsymbol{J}_t - \boldsymbol{J}^*\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2 + ({}^{L_{21}}/_2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2]\\ &\quad + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [M_J({}^{L_{11}}/_2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + L_{1D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2]\\ &\leq \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [L_1(\|\boldsymbol{J}_t - \hat{\boldsymbol{J}}\|_2 + \|\hat{\boldsymbol{J}} - \boldsymbol{J}^*\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2)(\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2 + \|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2)]\\ &\quad + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [({}^{L_{21}}/_2)(\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2^2)]\\ &\quad + \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}\ [M_J({}^{L_{11}}/_2)(\|\boldsymbol{z}_t - \hat{\boldsymbol{z}}\|_2^2 + \|\hat{\boldsymbol{z}} - \boldsymbol{z}^*\|_2^2) + L_{1D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2].\end{aligned} \tag{65}$$

Using the convergence errors from the forward pass (Theorems 4.3 and 4.7),

$$\begin{aligned}\|\boldsymbol{g}_t^{\text{ae-lasso}} - \boldsymbol{g}^*\|_2 &\leq L_1\mathcal{O}(t\rho^{2t} + (\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^*)t\rho^t + \rho^t(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}_J^*)\\ &\quad + L_1\mathcal{O}(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^*)(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}_J^*))\\ &\quad + M_J L_{11D} L_{21D}\|\boldsymbol{D}-\boldsymbol{D}^*\|_2(\rho^t + (\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^*))\\ &\quad + ({}^{L_{21}}/_2 + {}^{M_J L_{11}}/_2)\,\mathcal{O}(\rho^t + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^*) + \mathcal{O}(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2) + M_J\|\partial h(\boldsymbol{z}_t)\|_2)\end{aligned} \tag{66}$$

Hence,

$$\begin{aligned}\|\boldsymbol{g}_\infty^{\text{ae-lasso}} - \boldsymbol{g}^*\|_2 &\leq \mathcal{O}((\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}_J^*)(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \hat{\delta}^* + 1) + M_J\lambda\sqrt{s})\\ &= \mathcal{O}(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2^2 + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2(\hat{\delta}^* + \hat{\delta}_J^*) + \hat{\delta}^* + \hat{\delta}_J^* + M_J\lambda\sqrt{s})\end{aligned} \tag{67}$$

Similarly,

$$\|\boldsymbol{g}_\infty^{\text{ae-ls}} - \boldsymbol{g}^*\|_2 \leq \mathcal{O}(\|\boldsymbol{D}-\boldsymbol{D}^*\|_2^2 + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2 + \|\boldsymbol{D}-\boldsymbol{D}^*\|_2(\hat{\delta}^* + \hat{\delta}_J^*) + \hat{\delta}^* + \hat{\delta}_J^*) \tag{68}$$

■

Lemma A.6 (Global bounds)**.** *From global gradient errors in Theorem 4.10, the following are satisfied*

$$\begin{aligned}
\|Q_t^{21}(\boldsymbol{z}^*)\|_2 &\leq (L_{21}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 \\
\|Q_t^{lasso\text{-}11}(\boldsymbol{z}^*)\|_2 &\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + L_{1D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \|\partial h(\boldsymbol{z}_t)\|_2 \\
\|Q_t^{ls\text{-}11}(\boldsymbol{z}^*)\|_2 &\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + L_{1D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \\
\|Q(\boldsymbol{z}^*, \boldsymbol{J}_t)\|_2 &\leq L_1\|\boldsymbol{J}_t - \boldsymbol{J}^*\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2.
\end{aligned} \tag{69}$$

Proof. For $Q_t^{21}(\boldsymbol{z}^*)$, we achieve the quadratic bound using convexity of $\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{D})$ and its domain (Assumption 3.1) and Lemma 3.3. For $Q_t^{\text{lasso-11}}(\boldsymbol{z}^*)$, we add and subtract $\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})$, and use quadratic upper bound similar to Lemma A.5. At line four, we use $\boldsymbol{0} \in \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)$ (Lemma A.3) and assume that $\boldsymbol{z}_t$ recovers the sign entries of $\boldsymbol{z}^*$ (see Theorem 4.1 and Theorem 4.2).

$$\begin{aligned}
&\|Q_t^{\text{lasso-11}}(\boldsymbol{z}^*)\|_2 = \|\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) + \partial h(\boldsymbol{z}_t) - \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})(\boldsymbol{z}_t - \boldsymbol{z}^*)\|_2 \\
&= \|\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) + \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) + \partial h(\boldsymbol{z}_t) - \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})(\boldsymbol{z}_t - \boldsymbol{z}^*)\|_2 \\
&\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + \|\partial h(\boldsymbol{z}_t) + \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})\|_2 \\
&\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + \|\partial h(\boldsymbol{z}_t) + \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 \\
&\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + L_{1D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \|\partial h(\boldsymbol{z}_t)\|_2.
\end{aligned} \tag{70}$$

Similarly,

$$\begin{aligned}
\|Q_t^{\text{ls-11}}(\boldsymbol{z}^*)\|_2 &= \|\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})(\boldsymbol{z}_t - \boldsymbol{z}^*)\|_2 \\
&= \|\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}_t, \boldsymbol{D}) - \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) + \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})(\boldsymbol{z}_t - \boldsymbol{z}^*)\|_2 \\
&\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + \|\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})\|_2 \leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + \|\nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \nabla_1 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 \\
&\leq (L_{11}/2)\|\boldsymbol{z}_t - \boldsymbol{z}^*\|_2^2 + L_{1D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2.
\end{aligned} \tag{71}$$

For $Q(\boldsymbol{z}^*, \boldsymbol{J}_t)$, from implicit function theorem, $Q(\boldsymbol{z}^*, \boldsymbol{J}^*) = 0$ for $\boldsymbol{D}$ evaluated at $\boldsymbol{D}^*$. Hence, we can use $\nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) = -\boldsymbol{J}^{*+} \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)$ in the following

$$\begin{aligned}
\|Q(\boldsymbol{z}^*, \boldsymbol{J}_t)\|_2 &= \|\boldsymbol{J}_t^+ \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) + \nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) + \nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 \\
&= \|\boldsymbol{J}_t^+ \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \boldsymbol{J}^{*+} \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 + \|\nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \nabla_{21}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 \\
&= \|\boldsymbol{J}_t^+ \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}) - \boldsymbol{J}_t^+ \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 \\
&+ \|\boldsymbol{J}_t^+ \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) - \boldsymbol{J}^{*+} \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 + L_{21D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \\
&= M_J L_{11D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 + \|\boldsymbol{J}_t^+ \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*) - \boldsymbol{J}^{*+} \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D}^*)\|_2 + L_{21D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \\
&\leq \|(\boldsymbol{J}_t^+ - \boldsymbol{J}^{*+}) \nabla_{11}^2 \mathcal{L}_{\boldsymbol{x}}(\boldsymbol{z}^*, \boldsymbol{D})\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2 \\
&\leq L_1\|\boldsymbol{J}_t - \boldsymbol{J}^*\|_2 + M_J L_{11D} L_{21D}\|\boldsymbol{D} - \boldsymbol{D}^*\|_2.
\end{aligned} \tag{72}$$

■

Lemma A.7 (Dictionary maintains closeness)**.** *Suppose $\boldsymbol{D}^{(l)}$ has $(\delta_l, 2)$-closeness to $\boldsymbol{D}^*$ where $\delta_l = \mathcal{O}^*(1/\log m)$, then with probability of $1 - \epsilon_{supp\text{-}pres}^{(l)}$, we have $\|\boldsymbol{D}^{(l+1)} - \boldsymbol{D}^*\|_2 \leq 2\|\boldsymbol{D}^*\|_2$ when using $\boldsymbol{g}_T^{dec}$ and the network parameters set by Theorem 4.4.*

A.6 Interpretability

Theorem 5.1 (Interpretable unrolled network)**.** *Consider the dictionary learning optimization of the form $\min_{\boldsymbol{Z},\boldsymbol{D}} \ \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{D}\boldsymbol{Z}\|_2^2 + \lambda\|\boldsymbol{Z}\|_1 + \omega/2\|\boldsymbol{D}\|_F^2$, where $\boldsymbol{X} = [\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^n] \in \mathbb{R}^{m \times n}$ and $\boldsymbol{Z} = [\boldsymbol{z}^1, \boldsymbol{z}^2, \ldots, \boldsymbol{z}^n] \in \mathbb{R}^{p \times n}$. Let $\tilde{\boldsymbol{Z}}$ be the given converged sparse codes, then stationary points of the problem w.r.t the network weights (dictionary) follows $\tilde{\boldsymbol{D}} = \boldsymbol{X}\boldsymbol{G}^{-1}\tilde{\boldsymbol{Z}}^T$, where we denote $\boldsymbol{G} := (\tilde{\boldsymbol{Z}}^T\tilde{\boldsymbol{Z}} + \omega\boldsymbol{I})$.*

Proof. For all stationary points, the objective gradient is $\mathbf{0}$ with respect to the dictionary, i.e.,

$$\mathbf{0} = (\boldsymbol{X} - \tilde{\boldsymbol{D}}\tilde{\boldsymbol{Z}})\tilde{\boldsymbol{Z}}^{\mathrm{T}} + \omega\tilde{\boldsymbol{D}} \tag{73}$$

where $\tilde{\boldsymbol{D}}$ is the learned dictionary at convergence. Re-aranging the terms, we get

$$\tilde{\boldsymbol{D}} = \boldsymbol{X}\tilde{\boldsymbol{Z}}^{\mathrm{T}}(\tilde{\boldsymbol{Z}}\tilde{\boldsymbol{Z}}^{\mathrm{T}} + \omega\boldsymbol{I})^{-1} \tag{74}$$

Using the relation $\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}} + \omega\boldsymbol{I})^{-1} = (\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} + \omega\boldsymbol{I})^{-1}\boldsymbol{A}^{\mathrm{T}}$, we can re-write the solution as

$$\tilde{\boldsymbol{D}} = \boldsymbol{X}\boldsymbol{G}^{-1}\tilde{\boldsymbol{Z}}^{\mathrm{T}} \tag{75}$$

where we denote $\boldsymbol{G} \coloneqq (\tilde{\boldsymbol{Z}}^{\mathrm{T}}\tilde{\boldsymbol{Z}} + \omega\boldsymbol{I})$. ■

B Appendix - future works and limitations

Beyond dictionary learning Our results are founded on three main properties: Lipschitz differentiability of the loss relating to the parameter of interest, proximal gradient descent, and strong convexity in finite-iteration. The findings can be applied to other min-min optimization problems, e.g., ridge regression and logistic regression, following such properties. For example, our analysis generalizes to the unrolled network in (Tolooshams et al., 2020) for learning dictionaries using data from the natural exponential family. In this case, the least-squares loss is replaced with negative log-likelihood, and the dictionary models the expectation of the data.

Limitations Finite-iteration support selection (Proposition 4.1) (Hale et al., 2007) and strong convexity may seem stringent going beyond dictionary learning. Ablin et al. (2020) discuss generalization of local gradient convergence by relaxing strong convexity to the p-Łojasiewicz property (Ablin et al., 2020; Attouch & Bolte, 2009). We considered the noiseless setting and conjecture that the relative comparison of the gradients in the presence of noise still holds, where the upper bounds will involve an additional noise term. We focused on infinite sample convergence to highlight the relative differences between the gradients. We leave for future work the derivation of finite-sample upper bounds, a step similar to (Chatterji & Bartlett, 2017; Arora et al., 2015).

C Appendix - details of experiments

PUDLE is developed using PyTorch (Paszke et al., 2017) on Python. We used one GeForce GTX 1080 Ti GPU.

C.1 Numerical experiments for theories

Dataset We generated $n = 10{,}000$ samples following the model (2). We sampled $\boldsymbol{D}^* \in \mathbb{R}^{50\times 100}$ from zero-mean Gaussian distribution, and then normalized the columns. The codes are 5-sparse with their support uniformly chosen at random and their amplitudes are sampled from Uniform$(1, 2)$.

Training We let $T = 200$, $\lambda = 0.2$, and $\alpha = 0.2$. The dictionary is initialized to $\boldsymbol{D} = \boldsymbol{D}^* + \tau_B\boldsymbol{B}$ with $\boldsymbol{B} \sim \mathcal{N}(\mathbf{0}, \frac{1}{m}\boldsymbol{I})$. For Figures 2, 3 and 4a, we set $\tau_B \approx {}^{0.55}/_{\log m}$. Figure 11 shows the sparse code estimates from one example given this initialized dictionary; this is to highlight that a) the initial dictionary is not very close to the ground-truth dictionary, and b) our algorithm is able to successfully perform dictionary learning and recover the support by the end of training in spite of a failed exact recovery of the support.

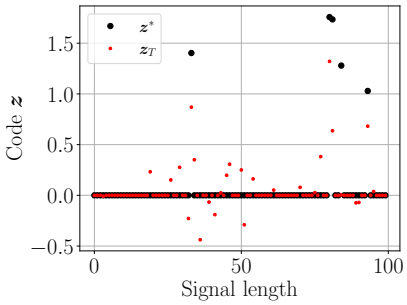
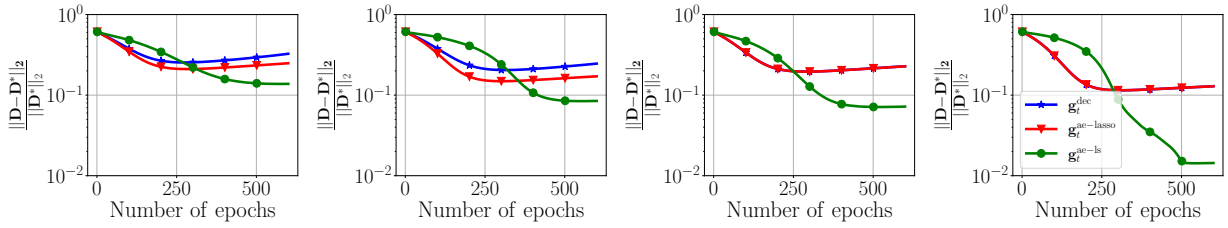


Figure 11: Example of code estimates with the initialized dictionary.

For Figures 4b and 4c, we chose much larger noise level, $\tau_B \approx {}^{2.8}/_{\log m}$. The network is trained for 600 epochs with full-batch gradient descent using Adam optimizer (Kingma & Ba, 2014) with learning rate of 10^{-3} and $\epsilon = 10^{-8}$. The learned dictionary is evaluated based on the error $\|\boldsymbol{D}-\boldsymbol{D}^*\|_2/\|\boldsymbol{D}^*\|_2$. The results and conclusion were consistent across various realizations of the dataset and across various optimizers. Hence, in the main paper, the figures visualize results of one realization.

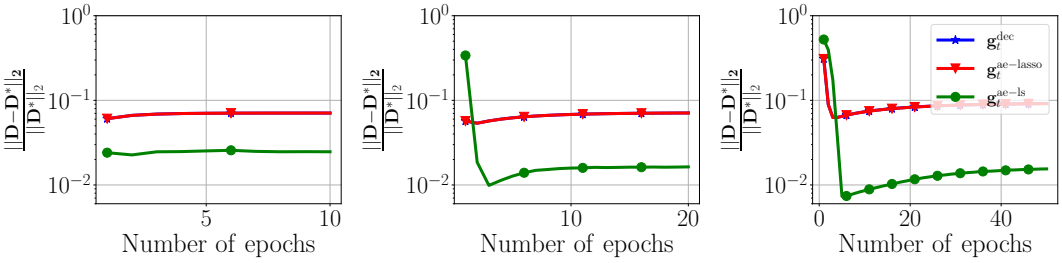
Noisy measurements We repeated the dictionary learning experiments shown in Figures 4b and 4c where the measurements $\boldsymbol{x}$ are corrupted by zero-mean Gaussian noise such that the SNR is approximately 12 dB. Accordingly, we increased the regularization value to $\lambda = 0.3$. Figure 12 shows the results for both noisy and noiseless scenarios.



(a) Noisy ($T = 25$). (b) Noiseless ($T = 25$). (c) Noisy ($T = 100$). (d) Noiseless ($T = 100$).

Figure 12: Dictionary learning in noisy and noiseless scenarios.

Stochastic Dictionary Learning In addition to the full-batch gradient descent results in the main paper, we repeated the experiments in Figure 4c using batch size of $4, 16$ and 64. We observed (Figure 13) that in all scenarios PUDLE is able to learn a good estimate of the ground-truth dictionary, and $\boldsymbol{g}_t^{\text{ae-ls}}$ is superior to the other two. We note that for lower batch-size, there will be more gradient updates in one epoch, hence, the algorithm converges in lower number of epochs.

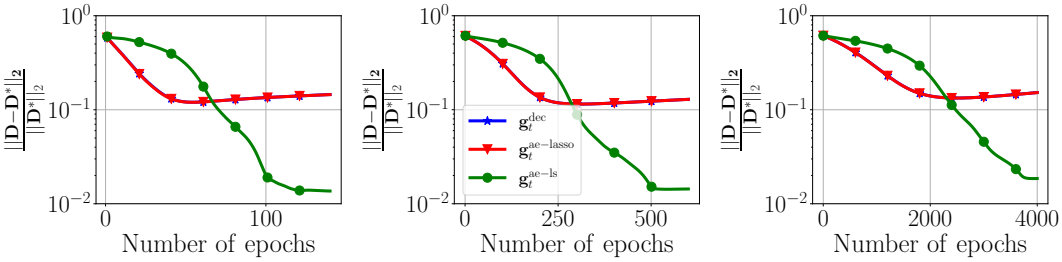


(a) Batch size 4 ($T = 25$). (b) Batch size 16 ($T = 100$). (c) Batch size 64 ($T = 100$).

Figure 13: Dictionary learning using various batch sizes.

Effect of learning rate We performed the experiments in Figure 4c for various learning rate of $10^{-4}, 10^{-3}$, and 10^{-2}. This shows the robustness of the gradient-based dictionary learning against learning rate. Figure 14 demonstrates such results where PUDLE successfully converges to the neighbourhood of the ground-truth dictionary; Regardless of the learning rate, $\boldsymbol{g}_t^{\text{ae-ls}}$ converges to a closer neighbourhood than the other two gradients. Overall, smaller the learning rate, more epochs is needed to reach convergence.

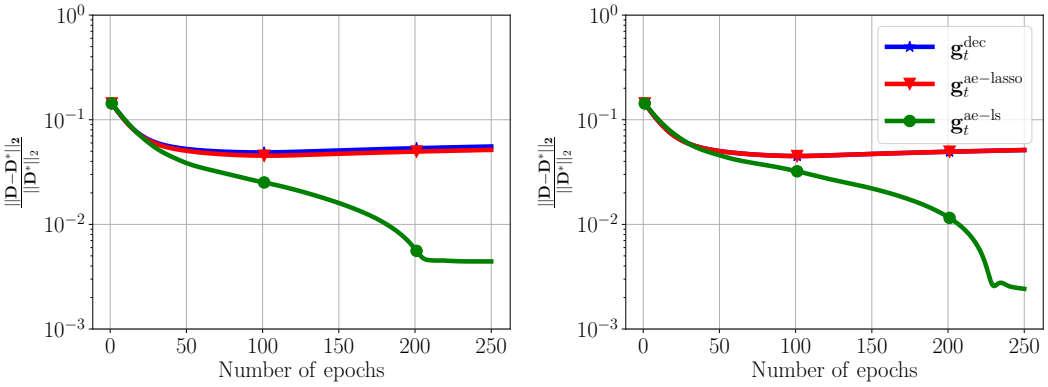
Dictionary initialization We conducted similar experiments to Figures 4b and 4c. We let $n = 10{,}000$, $m = 100$, and $p = 100$. We generated an orthogonal $\boldsymbol{D}^*$. The sparse codes $\boldsymbol{z}^*$ are 5-sparse and their amplitudes are drawn from sub-Gaussian $\mathcal{N}(0, 1)$. We set $\lambda = 0.05$, and $\alpha = 0.2$. We used the pairwise method proposed by Arora et al. (2015) to initialize the dictionary. This close initialization resulted in a dictionary that provides support recovery prior training. Figure 15 shows successful dictionary learning where $\boldsymbol{g}_t^{\text{ae-ls}}$ converges to a closer neighbourhood of $\boldsymbol{D}^*$ than the other two gradients. We used linear



(a) Learning rate 10^{-2}. (b) Learning rate 10^{-3}. (c) Learning rate 10^{-4}.

Figure 14: Dictionary learning for various learning rates when $T = 100$.

sum assignment optimization (i.e., `scipy.optimize.linear_sum_assignment`) to find the correct column permutations before computing the dictionary distance error.



(a) $T = 25$. (b) $T = 100$.

Figure 15: Dictionary learning when $\boldsymbol{D}$ is initialized using the pairwise method proposed by Arora et al. (2015).

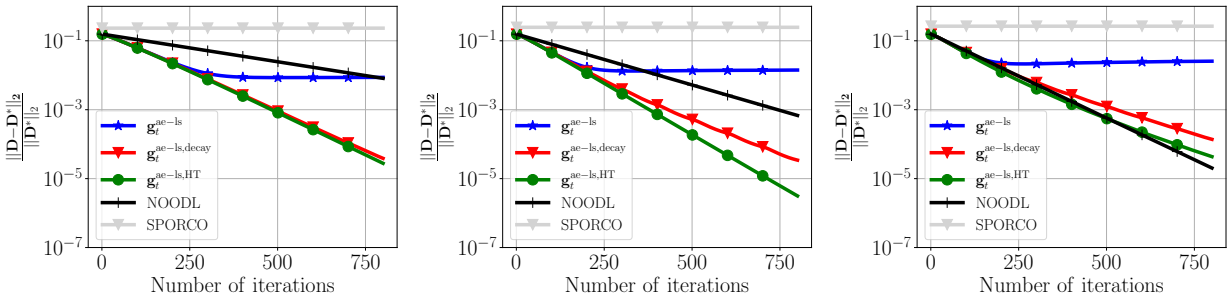
C.2 Dictionary learning

Dataset We generated $n = 50{,}000$ samples following (2). We let $m = 1000$ and $p = 1500$, and sample $\boldsymbol{D}^*$ from zero-mean Gaussian distribution, and then normalized the columns. The sparse codes $\boldsymbol{z}^i$ are 10, 20, 40-sparse, where their supports are chosen uniformly at random and their amplitudes are sampled from Uniform$(1, 2)$.

Training The dictionary is initialized to $\boldsymbol{D} = \boldsymbol{D}^* + \tau_B \boldsymbol{B}$ with $\boldsymbol{B} \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{m}\boldsymbol{I})$ where $\tau_B \approx 1/\log m$. We let $\lambda = 0.2$, and $\alpha = 0.2$, and $T = 100$. The network is trained for 1,000 iterative updates with batch-size of 50 using Adam (Kingma & Ba, 2014) with learning rate of 10^{-3} and $\epsilon = 10^{-3}$. For decay method, ν is decreased in value by 0.005 every 100 update iterations. Each filter is normalized after every update. The learned dictionary is evaluated based on the relative error $\|\boldsymbol{D}-\boldsymbol{D}^*\|_2 / \|\boldsymbol{D}^*\|_2$.

C.3 Image denoising

Training We trained PUDLE where the dictionary is convolutional with 64 filters of size 9×9 and strides of 4. The encoder unrolls for $T = 15$, and the step size is set to $\alpha = 0.1$. Unlike the theoretical analysis where full-batch gradient descent is studied, the network is trained stochastically with Adam optimizer (Kingma & Ba, 2014) with a learning rate of 10^{-4} and $\epsilon = 10^{-3}$ for 250 epochs. At every training iteration, a random 129×129 patch is cropped and a zero-mean Gaussian noise with a standard deviation of 25 is added.



(a) 10-sparse code. (b) 20-sparse code. (c) 40-sparse code.

Figure 16: Dictionary learning convergence using $\boldsymbol{g}_t^{\text{ae-ls}}$ compared to NOODL (Rambhatla et al., 2018) and SPORCO (Wohlberg, 2017).

We utilize random horizontal and vertical flip for augmentation. We report results in terms of the peak signal-to-noise ratio (PSNR). The standard deviation of the test PSNR across multiple noise realizations was lower than 0.02 dB for all the methods. Hence, we only reported the mean PSNR of the test set.

C.4 Interpretable sparse coding and dictionary learning

We focused on digits of $\{0, 1, 2, 3, 4\}$ of MNIST. We set $T = 15$, $\lambda = 0.7$, and $\alpha = 1$. The dictionary dimensions are $m = 784$ and $p = 500$. We trained the network for 200 epochs using Adam optimizer with a learning rate of 10^{-4} and batch size of 32. For construction of $\boldsymbol{G}$, ω is set to 0.001. For Figure 8, we computed the image contributions using 6,000 randomly chosen training images. The Gram matrix used in Figure 9, is constructed by 6,000 training examples, and the reconstruction is from the 200 most contributed training images.