# Robust Training under Label Noise by Over-parameterization

**Sheng Liu** [1] **Zhihui Zhu** [2] **Qing Qu** [3] **Chong You** [4]

## Abstract

Recently, over-parameterized deep networks, with increasingly more network parameters than training samples, have dominated the performances of modern machine learning. However, when the training data is corrupted, it has been well-known that over-parameterized networks tend to overfit and do not generalize. In this work, we propose a principled approach for robust training of over-parameterized deep networks in classification tasks where a proportion of training labels are corrupted. The main idea is yet very simple: label noise is sparse and incoherent with the network learned from clean data, so we model the noise and learn to separate it from the data. Specifically, we model the label noise via another *sparse over-parameterization* term, and exploit implicit algorithmic regularizations to recover and separate the underlying corruptions. Remarkably, when trained using such a simple method in practice, we demonstrate state-of-the-art test accuracy against label noise on a variety of real datasets. Furthermore, our experimental results are corroborated by theory on simplified linear models, showing that exact separation between sparse noise and low-rank data can be achieved under incoherent conditions. The work opens many interesting directions for improving over-parameterized models by using sparse over-parameterization and implicit regularization. Code is available at `https://github.com/shengliu66/SOP`.

## 1. Introduction

One of the most important factors for the success of deep models is their large model size and high expressive power, which enable them to learn complicated input-output relations. As such, over-parametrized deep networks or large models, with more parameters than the size of training data, have dominated the performance in computer vision, natural language processing, and so on. The adoption of large models is justified by the recent discovery that deep models exhibit a "double descent" (Belkin et al., 2019) and "unimodal variance" (Yang et al., 2020) generalization behavior, where their performance continues to improve beyond the interpolation point, extending the classical learning theory of bias-variance trade-off. While there are infinitely many global solutions that *overfit* to training data, the choice of optimization algorithm imposes certain *implicit* regularization (Neyshabur et al., 2014) so that over-parameterized models converge to those that are generalizable.

Nonetheless, the success of over-parameterization of deep networks critically depends on the availability of *clean* training data, while overfitting inevitably occurs when training data is corrupted. Consider the task of image classification with a training dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$, with $\boldsymbol{x}_i$ being an input image and $\boldsymbol{y}_i$ being the corresponding one-hot label. With an over-parameterized deep network $f(\cdot;\, \boldsymbol{\theta})$, model training is achieved by solving an optimization problem with respect to (w.r.t.) the network parameter $\boldsymbol{\theta}$ as follows:

$$\min_{\boldsymbol{\theta}} \; L(\boldsymbol{\theta}) \;=\; \frac{1}{N}\sum_{i=1}^{N} \ell\Big(f(\boldsymbol{x}_i;\, \boldsymbol{\theta}),\; \boldsymbol{y}_i\Big), \qquad (1)$$

where $\ell(\cdot,\cdot)$ is a loss function that measures the distance between network prediction $f(\boldsymbol{x}_i;\, \boldsymbol{\theta})$ and the label $\boldsymbol{y}_i$. If a proportion of the images in the training set is *mislabelled* (Song et al., 2020), it is well-known that the network will be optimized to zero training error hence produce $f(\boldsymbol{x}_i; \boldsymbol{\theta}) \approx \boldsymbol{y}_i$ for all $i \in \{1, \cdots, N\}$, even for $\boldsymbol{y}_i$'s that are incorrect (Zhang et al., 2021a). Overfitting to wrong labels inevitably leads to poor generalization performance (see Fig. 1).

In this paper, we introduce a principled method to address the challenges of overfitting over-parameterized deep networks in the presence of training data corruptions. We focus on the task of classification trained with noisy label, a ubiquitous problem in practice due to the extreme complexity of data annotation even for experienced domain experts (Frénay & Verleysen, 2013). Our idea leverages the property that the label noise is *sparse*, namely only a fraction of the labels are corrupted and the rest are intact. Principled methods for dealing with sparse corruption have a rich

[1]Center for Data Science, New York University [2]Electrical and Computer Engineering, University of Denver [3]Department of EECS, University of Michigan [4]Google Research, New York City. Correspondence to: Chong You <cyou@google.com>.

*Proceedings of the $39^{th}$ International Conference on Machine Learning*, Baltimore, Maryland, USA, PMLR 162, 2022. 

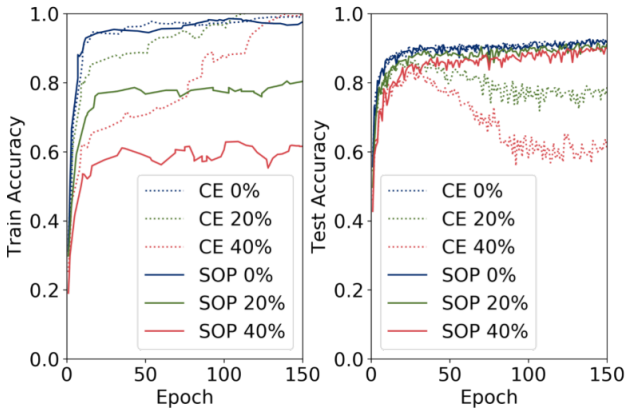



*Figure 1.* **Sparse over-parameterization prevents overfitting to label noise.** Training and test accuracy of a PreActResNet18 network trained with a standard cross entropy (CE) loss (dashed lines) and our Sparse Over-parameterization (SOP) (solid lines) for image classification on the CIFAR-10 dataset with 0%, 20%, and 40% of the labels flipped at random. SOP prevents overfitting to the wrong training labels, obtaining near 100%, 80%, 60% training accuracy respectively, therefore achieves better generalization on the test set without an accuracy drop at the end of training.

history, which can be retraced back to compressed sensing (Candes & Tao, 2005), robust subspace recovery (Candès et al., 2011; Wright et al., 2008), and even earlier (Claerbout & Muir, 1973). Such methods are based on using a robust loss function, such as the $\ell_1$ norm which is less sensitive to large outlying entries. While it is tempting to use sparse modeling for the label noise problem by setting the loss $\ell()$ in (1) as the $\ell_1$ loss, such an approach cannot solve the overfitting issue since all global solutions are still given by those that satisfy $f(\boldsymbol{x}_i;\boldsymbol{\theta}) \approx \boldsymbol{y}_i$ for all $i \in \{1,\cdots,N\}$. Hence, handling sparse corruptions with over-parameterized models requires the development of techniques beyond the classical $\ell_1$ loss for sparse modeling.

**Overview of our method and contribution.** To handle sparse corruption with over-parameterized models, our idea is simply to use an extra variable $\boldsymbol{s}_i$ to model the unknown label noise $\boldsymbol{s}_{\star i}$, which is the difference between the observed label $\boldsymbol{y}_i$ and the corresponding clean label. Hence, the goal is to minimize the discrepancy between $f(\boldsymbol{x}_i;\boldsymbol{\theta})+\boldsymbol{s}_i$ and $\boldsymbol{y}_i$. Inspired by a line of recent work (Vaskevicius et al., 2019; Zhao et al., 2019; You et al., 2020), we enforce sparsity of $\boldsymbol{s}_i$ by the over-parameterization $\boldsymbol{s}_i = \boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i$ and optimize the following training loss

$$\min_{\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{i=1}^N} L\left(\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{k=1}^N\right)), \tag{2}$$

where

$$L\left(\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\}\right)) = \tfrac{1}{N}\sum_{i=1}^N \ell\left(f(\boldsymbol{x}_i;\boldsymbol{\theta}) + \boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i, \boldsymbol{y}_i\right)),$$

with $\odot$ denoting an entry-wise Hadamard product. We term our method *Sparse Over-Parameterization* (SOP).

At the first glance, our SOP approach is seemingly problematic, because adding more learnable parameters $\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{i=1}^N$ to an over-parameterized network $f(\cdot,\boldsymbol{\theta})$ would aggravate rather than alleviate the overfitting issue. Indeed, a global solution to (2) is given by $\boldsymbol{u}_i \equiv \boldsymbol{v}_i \equiv \boldsymbol{0}$ and $f(\boldsymbol{x}_i,\boldsymbol{\theta}) \equiv \boldsymbol{y}_i$ for all $i \in \{1,\cdots,N\}$ where the network overfits to noisy labels. Here, we leverage the choice of a particular training algorithm to enforce an *implicit bias* towards producing the desired solutions. Technically, we run gradient descent on the objective in (2) starting from a small initialization for $\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{i=1}^N$:

$$\begin{aligned}
\boldsymbol{\theta} &\leftarrow \boldsymbol{\theta} - \tau \cdot \frac{\partial L(\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\})}{\partial \boldsymbol{\theta}}, \\
\boldsymbol{u}_i &\leftarrow \boldsymbol{u}_i - \alpha\tau \cdot \frac{\partial L(\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\})}{\partial \boldsymbol{u}_i}, \quad i = 1,\ldots,N, \\
\boldsymbol{v}_i &\leftarrow \boldsymbol{v}_i - \alpha\tau \cdot \frac{\partial L(\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\})}{\partial \boldsymbol{v}_i}, \quad i = 1,\ldots,N,
\end{aligned} \tag{3}$$

where $\alpha > 0$ is the ratio of learning rates for different training variables. Such a simple algorithm enables our method of SOP to train a deep image classification networks without overfitting to wrong labels and obtain better generalization performance (see Fig. 1). A more comprehensive empirical study with a variety of datasets is presented in Section 2.

To rigorously justify our method, we theoretically investigate our method based upon a simplified over-parameterized linear model with sparse corruptions. As justified by a line of recent work (Jacot et al., 2018; Chizat et al., 2019), over-parameterized linear models capture similar phenomena because they well approximate over-parameterized deep networks in a linearized regime around the initial points. Under sparse corruption and certain low-rank assumptions on the data, we show that the gradient descent (3) with an $\alpha$ below a certain threshold recovers the underlying model parameters with sparse corruptions. Our result is obtained by explicitly characterizing the implicit regularization for the term $\boldsymbol{u}_i \odot \boldsymbol{u}_i - \boldsymbol{v}_i \odot \boldsymbol{v}_i$. In particular, we explicitly show that it leads to an $\ell_1$-norm regularization on the sparse corruption, hence connecting our method to classical $\ell_1$ loss approaches for model robustness. For more details, we refer readers to Section 3.

In summary, our contributions are two-folds:

- *Method.* We proposed a simple yet practical SOP method that can effectively prevent overfitting for learning over-parameterized deep networks from corrupted training data, demonstrated on a variety of datasets.
- *Theory.* Under a simplified over-parameterized linear model, we rigorously justify our approach for exactly separating sparse corruption from the data.

Moreover, we believe the methodology we developed here could be far beyond the label noise setting, with the potential for dealing with more challenging scenarios of preventing overfitting in learning modern over-parametrized models of an ever-increasing size.

## 2. Robust Classification with Label Noise

In this section, we show how our SOP method plays out on image classification problems with the noisy label. In particular, we discuss extra implementation details of our method, followed by experimental demonstrations on a variety of datasets with synthetic and real label noise.

### 2.1. Implementation Details of SOP

We train an over-parameterized deep neural network $f(\cdot;\boldsymbol{\theta})$ from the noisy training data $\{(\boldsymbol{x}_i,\boldsymbol{y}_i)\}_{i=1}^N$ using the method briefly discussed in Section 1. Specifically, we train the network $f(\cdot;\boldsymbol{\theta})$ using the objective (2) with stochastic gradient descent (SGD) (i.e. a batch version of (3)).

Notice that there is additional prior information on label noise $\boldsymbol{s}_{\star i}$ associated with a sample $\{\boldsymbol{x}_i,\boldsymbol{y}_i\}$, namely, the positive and negative entries of $\boldsymbol{s}_{\star i}$ must correspond to nonzero entry and zero entries of $\boldsymbol{y}_i$, respectively. Moreover, all entries of $\boldsymbol{s}_{\star i}$ must lie in the range of $[-1,1]$. To leverage such information, we optimize a variant of (2) given by

$$\min_{\boldsymbol{\theta},\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{i=1}^N} \frac{1}{N}\sum_{i=1}^N \ell\Big(f(\boldsymbol{x}_i;\,\boldsymbol{\theta})+\boldsymbol{s}_i,\,\boldsymbol{y}_i\Big), \tag{4}$$

$$\text{s.t. } \boldsymbol{s}_i \doteq \boldsymbol{u}_i\odot\boldsymbol{u}_i\odot\boldsymbol{y}_i-\boldsymbol{v}_i\odot\boldsymbol{v}_i\odot(1-\boldsymbol{y}_i),\ \text{ and} \tag{5}$$

$$\boldsymbol{u}_i\in[-1,1]^K,\ \ \boldsymbol{v}_i\in[-1,1]^K, \tag{6}$$

where $K$ is the number of classes. In above, constraints on $\boldsymbol{u}_i,\boldsymbol{v}_i$ are realized by performing a projection step after each gradient descent update.

**Choice of the loss function $\ell(\cdot,\cdot)$ in (4).** The most commonly used loss function for classification tasks is the cross-entropy loss $\ell_{\rm CE}(\cdot,\cdot)$ (Krizhevsky et al., 2012). Because the $\ell_{\rm CE}(\cdot,\cdot)$ loss requires a probability distribution as an input, we define a mapping

$$\phi(\boldsymbol{w}) \doteq \frac{\max\{\boldsymbol{w},\epsilon\mathbf{1}\}}{\|\max\{\boldsymbol{w},\epsilon\mathbf{1}\}\|_1}, \tag{7}$$

and set the loss $\ell(\cdot,\cdot)$ in (4) to be

$$L_{\rm CE}\big(\boldsymbol{\theta},\boldsymbol{u}_i,\boldsymbol{v}_i;\boldsymbol{x}_i,\boldsymbol{y}_i\big) \doteq \ell_{\rm CE}\Big(\phi\big(f(\boldsymbol{x}_i;\,\boldsymbol{\theta})+\boldsymbol{s}_i\big),\,\boldsymbol{y}_i\Big). \tag{8}$$

On the other hand, the cross-entropy loss cannot be used to optimize the variables $\{\boldsymbol{v}_i\}$ (see Section A.1 for an explanation). Hence, we use the mean squared error loss $\ell_{\rm MSE}$ and set the loss in (4) to be

$$L_{\rm MSE}\big(\boldsymbol{\theta},\boldsymbol{u}_i,\boldsymbol{v}_i;\boldsymbol{x}_i,\boldsymbol{y}_i\big) \doteq \ell_{\rm MSE}\Big(f(\boldsymbol{x}_i;\,\boldsymbol{\theta})+\boldsymbol{s}_i,\,\boldsymbol{y}_i\Big), \tag{9}$$

when optimizing $\{\boldsymbol{v}_i\}$ [1]. We summarize our training method in Algorithm 1.

[1] We also project $f(\boldsymbol{x}_i;\,\boldsymbol{\theta})$ to a one-hot vector when using MSE loss which is empirically found to accelerate convergence of $\{\boldsymbol{v}_i\}$.

**Algorithm 1** Image classification under label noise by the method of Sparse Over-Parameterization (SOP).

**Input:** Training data $\{(\boldsymbol{x}_i,\boldsymbol{y}_i)\}_{i=1}^N$, network backbone $f(\cdot,\boldsymbol{\theta})$, variables $\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{i=1}^N$, number of epochs $T$, learning rate $\tau$, learning rate ratio $\alpha_u,\alpha_v$, batch size $\beta$
**Initialization:** Draw entries of $\boldsymbol{u}_i,\boldsymbol{v}_i$ from i.i.d. Gaussian distribution with zero-mean and s.t.d. $1e-8$
**for** each $t\in\{1,\cdots,T\}$ **do**
  `# Train network` $f(\cdot,\boldsymbol{\theta})$ `with SGD`
  **for** each $b\in\{1,\cdots,N/\beta\}$ **do**
    Sample a batch $\mathcal{B}\subseteq\{1,\ldots,N\}$ with $|\mathcal{B}|=\beta$
    Set $\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}-\tau\cdot\sum_{i\in\mathcal{B}}\frac{\partial L_{\rm CE}(\boldsymbol{\theta},\boldsymbol{u}_i,\boldsymbol{v}_i;\boldsymbol{x}_i,\boldsymbol{y}_i)}{\partial\boldsymbol{\theta}}$
  **end for**
  `# Update` $\{\boldsymbol{u}_i,\boldsymbol{v}_i\}$
  **for** each $i\in\{1,\cdots,N\}$ **do**
    Set $\boldsymbol{u}_i\leftarrow\mathcal{P}_{[-1,1]}\Big(\boldsymbol{u}_i-\alpha_u\tau\frac{\partial L_{\rm CE}(\boldsymbol{\theta},\boldsymbol{u}_i,\boldsymbol{v}_i;\boldsymbol{x}_i,\boldsymbol{y}_i)}{\partial\boldsymbol{u}_i}\Big)$
    Set $\boldsymbol{v}_i\leftarrow\mathcal{P}_{[-1,1]}\Big(\boldsymbol{v}_i-\alpha_v\tau\frac{\partial L_{\rm MSE}(\boldsymbol{\theta},\boldsymbol{u}_i,\boldsymbol{v}_i;\boldsymbol{x}_i,\boldsymbol{y}_i)}{\partial\boldsymbol{v}_i}\Big)$
  **end for**
**end for**
**Output:** Network parameters $\boldsymbol{\theta}$ and $\{\boldsymbol{u}_i,\boldsymbol{v}_i\}_{i=1}^N$

### 2.2. Experiments

We experimentally demonstrate the effectiveness of our proposed SOP method on datasets with both synthetic (i.e., CIFAR-10 and CIFAR-100) and realistic (i.e., CIFAR-N, Clothing-1M, and WebVision) label noise. In addition to the SOP described in Algorithm 1, we also implement an improved version, termed SOP+, which incorporates two commonly used regularization techniques in the literature of label noise, namely the consistency regularization and the class-balance regularization. We explain SOP+ in more detail in Appendix A.3.

**Dataset descriptions.** We use datasets with synthetic label noise generated from CIFAR-10 and CIFAR-100 datasets (Krizhevsky et al., 2009). Each dataset contains 50k training images and 10k test images, all with clean labels, where each image is of size $32\times32$. Following previous works (Han et al., 2018; Liu et al., 2020; Xia et al., 2020), we generate symmetric label noise by uniformly flipping labels for a percentage of the training set for all classes, as well as asymmetric label noise by flipping labels for particular pairs of classes. For datasets with realistic label noise, we test on CIFAR-10N/CIFAR-100N (Wei et al., 2021b) which contains a re-annotation of CIFAR-10/CIFAR-100 with human workers. Specifically, each image in CIFAR-10N contains three submitted labels (i.e., **Random 1, 2, 3**) which are further combined to have an **Aggregate** and a **Worst** label. Each image in CIFAR-100N contains a single submitted label for the fine classes. We also test on Clothing-1M (Xiao et al., 2015) which is a large-scale dataset with images clawed from online shopping websites and labels

*Table 1.* **Test accuracy with synthetic label noise** on CIFAR-10 and CIFAR-100 with $\{20\%, 40\%, 60\%, 80\%\}$ percent of labels for training data randomly flipped uniformly to another class. All methods use ResNet34 as the architecture. Mean and standard deviation over 5 independent runs are reported.

| METHODS | CIFAR-10 | | | | CIFAR-100 | | | |
|---|---|---|---|---|---|---|---|---|
| | 20% | 40% | 60% | 80% | 20% | 40% | 60% | 80% |
| CE | $86.32_{\pm 0.18}$ | $82.65_{\pm 0.16}$ | $76.15_{\pm 0.32}$ | $59.28_{\pm 0.97}$ | $51.43_{\pm 0.58}$ | $45.23_{\pm 0.53}$ | $36.31_{\pm 0.39}$ | $20.23_{\pm 0.82}$ |
| FORWARD | $87.99_{\pm 0.36}$ | $83.25_{\pm 0.38}$ | $74.96_{\pm 0.65}$ | $54.64_{\pm 0.44}$ | $39.19_{\pm 2.61}$ | $31.05_{\pm 1.44}$ | $19.12_{\pm 1.95}$ | $8.99_{\pm 0.58}$ |
| GCE | $89.83_{\pm 0.20}$ | $87.13_{\pm 0.22}$ | $82.54_{\pm 0.23}$ | $64.07_{\pm 1.38}$ | $66.81_{\pm 0.42}$ | $61.77_{\pm 0.24}$ | $53.16_{\pm 0.78}$ | $29.16_{\pm 0.74}$ |
| SL | $89.83_{\pm 0.32}$ | $87.13_{\pm 0.26}$ | $82.81_{\pm 0.61}$ | $68.12_{\pm 0.81}$ | $70.38_{\pm 0.13}$ | $62.27_{\pm 0.22}$ | $54.82_{\pm 0.57}$ | $25.91_{\pm 0.44}$ |
| ELR | $91.16_{\pm 0.08}$ | $89.15_{\pm 0.17}$ | $86.12_{\pm 0.49}$ | $\mathbf{73.86}_{\pm 0.61}$ | $74.21_{\pm 0.22}$ | $68.28_{\pm 0.31}$ | $59.28_{\pm 0.67}$ | $29.78_{\pm 0.56}$ |
| SOP (OURS) | $\mathbf{93.18}_{\pm 0.57}$ | $\mathbf{90.09}_{\pm 0.27}$ | $\mathbf{86.76}_{\pm 0.22}$ | $68.32_{\pm 0.77}$ | $\mathbf{74.67}_{\pm 0.30}$ | $\mathbf{70.12}_{\pm 0.57}$ | $\mathbf{60.26}_{\pm 0.41}$ | $\mathbf{30.20}_{\pm 0.63}$ |

*Table 2.* **Comparison with the state-of-the-art methods** that use two network ensembles and semi-supervised learning on CIFAR-10 and CIFAR-100 under symmetric (with $20\%, 50\%, 80\%$) and asymmetric (with $40\%$) label noise. All methods use ResNet34 as the architecture.

| METHODS | CIFAR-10 | | | | CIFAR-100 | | | |
|---|---|---|---|---|---|---|---|---|
| | SYMMETRIC | | | ASYM | SYMMETRIC | | | ASYM |
| | 20% | 50% | 80% | 40% | 20% | 50% | 80% | 40% |
| CE | 87.2 | 80.7 | 65.8 | 82.2 | 58.1 | 47.1 | 23.8 | 43.3 |
| MIXUP | 93.5 | 87.9 | 72.3 | - | 69.9 | 57.3 | 33.6 | - |
| DIVIDEMIX | 96.1 | 94.6 | 93.2 | 93.4 | 77.1 | 74.6 | 60.2 | 72.1 |
| ELR+ | 95.8 | 94.8 | 93.3 | 93.0 | 77.7 | 73.8 | 60.8 | 77.5 |
| SOP+ (OURS) | **96.3** | **95.5** | **94.0** | **93.8** | **78.8** | **75.9** | **63.3** | **78.0** |

*Table 3.* **Test accuracy with realistic label noise on Clothing1M and WebVision.** We use a pre-trained ResNet50 for Clothing1M and an InceptionResNetV2 for WebVision dataset. The results of the comparing methods are taken from their respective papers.

| METHODS | CLOTHING1M | WEBVISION | ILSVRC12 |
|---|---|---|---|
| CE | 69.1 | - | - |
| FORWARD | 69.8 | 61.1 | 57.3 |
| CO-TEACHING | 69.2 | 63.6 | 61.5 |
| ELR | 72.9 | 76.2 | 68.7 |
| CORES$^2$ | 73.2 | - | - |
| SOP (OURS) | **73.5** | **76.6** | **69.1** |

generated based on surrounding texts. Clothing-1M contains 1 million training images, 15k validation images, and 10k test images with clean labels. Finally, we also test on the mini WebVision dataset (Li et al., 2017) which contains the top 50 classes from the Google image subset of WebVision (approximately 66 thousand images). Models trained on mini WebVision are evaluated on both WebVision and ImageNet ILSVRC12 validation set. Details on the label noise for these datasets is provided in Section A.2.

**Network structures & hyperparameters.** We implement our method with PyTorch v1.7. For each dataset, the choices of network architectures and hyperparameters for SOP are as follows. Additional details, as well as hyper-parameters for both SOP and SOP+, can be found in Appendix A.4.

- *CIFAR-10/100 and CIFAR-10N/100N*. We follow (Liu et al., 2020) to use ResNet-34 and PreActResNet18 architectures trained with SGD using a 0.9 momentum. The initial learning rate is $0.02$ decayed with a factor of 10 at the 40th and 80th epochs for CIFAR-10/CIFAR-10N and at the 80th and 120th epochs for CIFAR-100/CIFAR-100N, respectively. Weight decay for network parameters $\boldsymbol{\theta}$ is set to $5 \times 10^{-4}$. No weight decay is used for parameters $\{\boldsymbol{u}_i, \boldsymbol{v}_i\}_{i=1}^N$.
- *Clothing-1M*. We follow the previous work (Liu et al., 2020) to use a ResNet-50 (He et al., 2016) pre-trained on ImageNet (Krizhevsky et al., 2012). The network is trained with batch size 64 and an initial learning rate 0.001, which is reduced by a factor of 10 after 5th epoch (10 training epochs in total). Optimization is performed using SGD with a momentum 0.9. Weight decay is 0.001 for parameters $\boldsymbol{\theta}$ and is zero for parameters $\{\boldsymbol{u}_i, \boldsymbol{v}_i\}_{i=1}^N$.
- *Mini Webvision*. We use InceptionResNetV2 as the backbone architecture. All other optimization details are the same as for CIFAR-10, except that we use weight decay $0.0005$ and batch size 32.

**Experimental results.** We compare with methods based on estimation of the transition matrix (Forward (Patrini et al., 2017)), design of loss functions (GCE (Zhang & Sabuncu, 2018) and SL (Wang et al., 2019)), training two networks (Co-teaching (Han et al., 2018) and DivideMix (Li et al., 2020a)), and label noise correction (ELR (Liu et al., 2020) and CORES$^2$ (Cheng et al., 2021)).

Table 1 reports the performance of our method on synthetically generated symmetric label noise using CIFAR-10 and CIFAR-100. To compare with state-of-the-art methods, we also report the performance of SOP+ which contains additional regularization on both symmetric and asymmetric label noise and report the results in Table 2. It can be observed that our method is robust to a fairly large amount of label noise, and compares favorably to existing techniques.

We further demonstrate that our method can effectively handle datasets with realistic label noise by reporting its performance on Clothing1M & WebVision (see Table 3) and CIFAR-N (see Table 4) datasets. We can observe a performance gain over all comparing methods.

Finally, we compare the training time (on a single Nvidia V100 GPU) of our method to the baseline methods in Table 5. We observe that our algorithm SOP/SOP+ achieves the fastest speed across all baselines.

*Table 4.* **Test accuracy with realistic label noise on CIFAR-N.** Mean and standard deviation over 5 independent runs are reported. The results of the baseline methods are taken from (Wei et al., 2021b) which all use ResNet34 as the architecture. For SOP+, we use PreActResNet18.

| METHODS | CIFAR-10N | | | | | | CIFAR-100N | |
|---|---|---|---|---|---|---|---|---|
| | CLEAN | RANDOM 1 | RANDOM 2 | RANDOM 3 | AGGREGATE | WORST | CLEAN | NOISY |
| CE | $92.92_{\pm 0.11}$ | $85.02_{\pm 0.65}$ | $86.46_{\pm 1.79}$ | $85.16_{\pm 0.61}$ | $87.77_{\pm 0.38}$ | $77.69_{\pm 1.55}$ | $76.70_{\pm 0.74}$ | $55.50_{\pm 0.66}$ |
| FORWARD | $93.02_{\pm 0.12}$ | $86.88_{\pm 0.50}$ | $86.14_{\pm 0.24}$ | $87.04_{\pm 0.35}$ | $88.24_{\pm 0.22}$ | $79.79_{\pm 0.46}$ | $76.18_{\pm 0.37}$ | $57.01_{\pm 1.03}$ |
| CO-TEACHING | $93.35_{\pm 0.14}$ | $90.33_{\pm 0.13}$ | $90.30_{\pm 0.17}$ | $90.15_{\pm 0.18}$ | $91.20_{\pm 0.13}$ | $83.83_{\pm 0.13}$ | $73.46_{\pm 0.09}$ | $60.37_{\pm 0.27}$ |
| ELR+ | $95.39_{\pm 0.05}$ | $94.43_{\pm 0.41}$ | $94.20_{\pm 0.24}$ | $94.34_{\pm 0.22}$ | $94.83_{\pm 0.10}$ | $91.09_{\pm 1.60}$ | $78.57_{\pm 0.12}$ | $66.72_{\pm 0.07}$ |
| CORES* | $94.16_{\pm 0.11}$ | $94.45_{\pm 0.14}$ | $94.88_{\pm 0.31}$ | $94.74_{\pm 0.03}$ | $95.25_{\pm 0.09}$ | $91.66_{\pm 0.09}$ | $73.87_{\pm 0.16}$ | $55.72_{\pm 0.42}$ |
| SOP+(OURS) | $\mathbf{96.38}_{\pm 0.31}$ | $\mathbf{95.28}_{\pm 0.13}$ | $\mathbf{95.31}_{\pm 0.10}$ | $\mathbf{95.39}_{\pm 0.11}$ | $\mathbf{95.61}_{\pm 0.13}$ | $\mathbf{93.24}_{\pm 0.21}$ | $\mathbf{78.91}_{\pm 0.43}$ | $\mathbf{67.81}_{\pm 0.23}$ |

*Table 5.* **Comparison of total training time** in hours on CIFAR-10 with 50% symmetric label noise.

| CE | CO-TEACHING+ | DIVIDEMIX | ELR+ | SOP | SOP+ |
|---|---|---|---|---|---|
| 0.9H | 4.4H | 5.4H | 2.3H | 1.0H | 2.1H |

## 3. Theoretical Insights with Simplified Models

This section provides theoretical insights into our SOP method by studying structured data recovery with sparse corruption in the context of over-parameterized *linear* models. We will start with model simplification, followed by our main theoretical results and experimental verification.

### 3.1. Problem Setup & Main Result

Given a highly *overparameterized* network $f(\cdot;\boldsymbol{\theta})$, recent work (Jacot et al., 2018; Kalimeris et al., 2019) suggests that the parameter $\boldsymbol{\theta} \in \mathbb{R}^p$ may not change much from its initialization $\boldsymbol{\theta}_0$ before obtaining zero training error. Hence, a nonlinear network $f(\cdot;\boldsymbol{\theta}) : \mathbb{R}^n \mapsto \mathbb{R}$ can be well approximated by its first-order Taylor expansion:

$$f(\boldsymbol{x};\boldsymbol{\theta}) \approx f(\boldsymbol{x};\boldsymbol{\theta}_0) + \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}_0),\ \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle, \quad (10)$$

where we consider $f(\cdot;\boldsymbol{\theta})$ as a scalar function for simplicity. Since the bias term $f(\boldsymbol{x};\boldsymbol{\theta}_0) - \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}_0),\ \boldsymbol{\theta}_0 \rangle$ is constant w.r.t. $\boldsymbol{\theta}$, for simplicity we may further assume that

$$f(\boldsymbol{x};\boldsymbol{\theta}) \approx \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x};\boldsymbol{\theta}_0),\ \boldsymbol{\theta} \rangle. \quad (11)$$

Thus, for a dataset $\{\boldsymbol{x}_i\}_{i=1}^N$ of $N$ points, collectively

$$\begin{bmatrix} f(\boldsymbol{x}_1;\boldsymbol{\theta}) \\ \vdots \\ f(\boldsymbol{x}_N;\boldsymbol{\theta}) \end{bmatrix} \approx \begin{bmatrix} \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}_1;\boldsymbol{\theta}_0) \\ \vdots \\ \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}_N;\boldsymbol{\theta}_0) \end{bmatrix} \cdot \boldsymbol{\theta} \ = \ \boldsymbol{J} \cdot \boldsymbol{\theta}, \quad (12)$$

where $\boldsymbol{J} \in I\!R^{N \times p}$ is a Jacobian matrix. This observation motivates us to consider the following problem setup.

**Problem setup.** Based upon the above linearization, we assume that our corrupted observation $\boldsymbol{y} \in \mathbb{R}^N$ (e.g., noisy labels) is generated by

$$\boldsymbol{y} = \boldsymbol{J} \cdot \boldsymbol{\theta}_\star + \boldsymbol{s}_\star, \quad (13)$$

where $\boldsymbol{\theta}_\star \in \mathbb{R}^p$ is the underlying groundtruth parameter, and the noise $\boldsymbol{s}_\star \in \mathbb{R}^N$ is *sparse* so that only a subset of observation (e.g., labels) is corrupted. Given $\boldsymbol{J}$ and $\boldsymbol{y}$ generated from (13), our goal is to recover both $\boldsymbol{\theta}_\star$ and $\boldsymbol{s}_\star$.

However, as we are considering the problem in an over-parameterized regime with $p > N$, the underdetermined system (13) implies that there are *infinite* solutions for $\boldsymbol{\theta}_\star$ even if $\boldsymbol{s}_\star$ is given. Nonetheless, recent work showed that the implicit bias of gradient descent for overparameterized linear models and deep networks tend to find minimum $\ell_2$-norm solutions (Zhang et al., 2021a). To make our problem more well-posed, motivated by these results, we would like to find an $\boldsymbol{\theta}_\star$ with minimum $\ell_2$-norm, namely,

$$\boldsymbol{\theta}_\star = \arg\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|_2^2 \quad \text{s.t.} \quad \boldsymbol{y} = \boldsymbol{J}\boldsymbol{\theta} + \boldsymbol{s}_\star. \quad (14)$$

Analogous to (2), we will show that $\boldsymbol{\theta}_\star$ and $\boldsymbol{s}_\star$ can be provably recovered by solving the problem

$$\min_{\boldsymbol{\theta},\boldsymbol{u},\boldsymbol{v}} h(\boldsymbol{\theta},\boldsymbol{u},\boldsymbol{v}) \doteq \frac{1}{2}\|\boldsymbol{J}\boldsymbol{\theta} + \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v} - \boldsymbol{y}\|_2^2, \quad (15)$$

using the gradient descent algorithm with learning rates $\tau$ and $\alpha\tau$ on $\boldsymbol{\theta}$ and $\{\boldsymbol{u},\boldsymbol{v}\}$, respectively:

$$\begin{aligned} \boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k - \tau \cdot \boldsymbol{J}^\top \boldsymbol{r}_k, \\ \boldsymbol{u}_{k+1} &= \boldsymbol{u}_k - 2\alpha\tau \cdot \boldsymbol{u}_k \odot \boldsymbol{r}_k, \\ \boldsymbol{v}_{k+1} &= \boldsymbol{v}_k + 2\alpha\tau \cdot \boldsymbol{v}_k \odot \boldsymbol{r}_k, \end{aligned} \quad (16)$$

where $\boldsymbol{r}_k \doteq \boldsymbol{J}\boldsymbol{\theta}_k + \boldsymbol{u}_k \odot \boldsymbol{u}_k - \boldsymbol{v}_k \odot \boldsymbol{v}_k - \boldsymbol{y}$. Based on these, our result can be summarized as follows.

**Theorem 3.1** (Main result, informal)**.** *Suppose $\boldsymbol{J}$ is rank-$r$ and $\mu$-incoherent defined in Section 3.3, and $\boldsymbol{s}_\star$ is $k$-sparse. If $k^2 r < N/(4\mu)$, with $\tau \to 0$ and a proper choice of $\alpha$ depending on $(\boldsymbol{J}, \boldsymbol{\theta}_\star, k)$, the gradient dynamics of (16) converges to the ground truth solution $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ in (13) starting from a small initialization of $(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v})$.*

We state our result at a high level with more technical details in Section 3.2 and Section 3.3. The overall idea of the proof can be sketched through the following two steps.

- First, although the problem (15) is *nonconvex*, in Section 3.2 we show that it has benign global landscape, and that the gradient descent (16) converges to particular global solutions that are the same as solutions to a *convex* problem with explicit regularizations on $\boldsymbol{\theta}$ and $\boldsymbol{s}$.
- Building upon above results, in Section 3.3 we complete our analysis by showing that $\boldsymbol{\theta}_\star$ and $\boldsymbol{s}_\star$ can be exactly

recovered by the convex problem with a small enough value for $\alpha$.

Throughout the analysis, we corroborate our findings with numerical simulations.

### 3.2. Landscapes & Implicit Sparse Regularization

**Benign global landscape.** We start by characterizing the nonconvex landscape of (15), showing the following result.

**Proposition 3.2.** *Any critical point of* (15) *is either a global minimizer, or it is a strict saddle (Ge et al., 2015) with its Hessian having at least one negative eigenvalue.*

For a strict saddle function, recent work (Lee et al., 2016) showed that gradient descent with random initialization almost surely escapes saddle points and converges to a local minimizer. Thus, Proposition 3.2 ensures that the algorithm in (16) almost surely converges to a global solution of (15).

However, because there are infinite many global solutions for the overparameterized model (15) and not all global solutions are of equal quality, convergence to a global solution alone is not sufficient for us to establish the correctness of our method. Nonetheless, as we will show in the following, the particular choice of the algorithm in (16) enables it to converge to a particular *regularized* global solution.

**Implicit sparse regularization.** To understand which solution the algorithm (16) converges to, we study its *gradient flow* counterpart by taking the stepsize $\tau \to 0$ in (16). Thus, the dynamics of such a gradient flow is governed by the following differential equations

$$\begin{aligned} \dot{\boldsymbol{\theta}}_t(\gamma,\alpha) &= -\boldsymbol{J}^\top \boldsymbol{r}_t(\gamma,\alpha), \\ \dot{\boldsymbol{u}}_t(\gamma,\alpha) &= -2\alpha \cdot \boldsymbol{u}_t(\gamma,\alpha) \odot \boldsymbol{r}_t(\gamma,\alpha), \\ \dot{\boldsymbol{v}}_t(\gamma,\alpha) &= 2\alpha \cdot \boldsymbol{v}_t(\gamma,\alpha) \odot \boldsymbol{r}_t(\gamma,\alpha), \end{aligned} \tag{17}$$

where we define

$$\begin{aligned} \boldsymbol{r}_t(\gamma,\alpha) = \boldsymbol{J}\boldsymbol{\theta}_t(\gamma,\alpha) + \boldsymbol{u}_t(\gamma,\alpha) \odot \boldsymbol{u}_t(\gamma,\alpha) \\ - \boldsymbol{v}_t(\gamma,\alpha) \odot \boldsymbol{v}_t(\gamma,\alpha) - \boldsymbol{y}. \end{aligned} \tag{18}$$

Here, we assume that $\boldsymbol{\theta}, \boldsymbol{u}$, and $\boldsymbol{v}$ are initialized at

$$\boldsymbol{\theta}_0(\gamma,\alpha) = \boldsymbol{0},\ \boldsymbol{u}_0(\gamma,\alpha) = \gamma\boldsymbol{1},\ \boldsymbol{v}_0(\gamma,\alpha) = \gamma\boldsymbol{1}, \tag{19}$$

with some small $\gamma > 0$. Solving the differential equations in (17) gives the gradient flow

$$\begin{aligned} \boldsymbol{\theta}_t(\gamma,\alpha) &= \boldsymbol{J}^\top \boldsymbol{\nu}_t(\gamma,\alpha), \\ \boldsymbol{u}_t(\gamma,\alpha) &= \gamma \exp(2\alpha \boldsymbol{\nu}_t(\gamma,\alpha)), \\ \boldsymbol{v}_t(\gamma,\alpha) &= \gamma \exp(-2\alpha \boldsymbol{\nu}_t(\gamma,\alpha)), \end{aligned} \tag{20}$$

where we define

$$\boldsymbol{\nu}_t(\gamma,\alpha) \doteq -\int_0^t \boldsymbol{r}_\tau(\gamma,\alpha) d\tau. \tag{21}$$

The following result shows that the solution that the gradient flow $\big(\boldsymbol{\theta}_t(\gamma,\alpha), \boldsymbol{u}_t(\gamma,\alpha), \boldsymbol{v}_t(\gamma,\alpha)\big)$ in (20) converges to at $t \to \infty$ is a global solution to (15) that is *regularized* with a particular of $(\gamma,\alpha)$.

**Proposition 3.3.** *Consider the gradient flow in* (20) *with the initialization in* (19)*.*

- ***(Global convergence)*** *For any $(\gamma,\alpha)$, if the limit*

$$\begin{aligned} &\Big(\boldsymbol{\theta}_\infty(\gamma,\alpha),\ \boldsymbol{u}_\infty(\gamma,\alpha),\ \boldsymbol{v}_\infty(\gamma,\alpha)\Big) \\ &\qquad \doteq \lim_{t\to\infty} \Big(\boldsymbol{\theta}_t(\gamma,\alpha),\ \boldsymbol{u}_t(\gamma,\alpha),\ \boldsymbol{v}_t(\gamma,\alpha)\Big) \end{aligned} \tag{22}$$

  *exists, then it is a global solution to* (15)*.*

- ***(Implicit regularization)*** *Fix any $\lambda > 0$ and let $\alpha$ be a function of $\gamma$ as*

$$\alpha(\gamma) = -\frac{\log\gamma}{2\lambda}. \tag{23}$$

  *If the limit*

$$\begin{aligned} \big(\widehat{\boldsymbol{\theta}},\ \widehat{\boldsymbol{u}},\ \widehat{\boldsymbol{v}}\big) \doteq \lim_{\gamma\to 0} \Big(&\boldsymbol{\theta}_\infty(\gamma,\alpha(\gamma)), \\ &\boldsymbol{u}_\infty(\gamma,\alpha(\gamma)),\ \boldsymbol{v}_\infty(\gamma,\alpha(\gamma))\Big) \end{aligned} \tag{24}$$

  *exists, then $\big(\widehat{\boldsymbol{\theta}},\ \widehat{\boldsymbol{u}},\ \widehat{\boldsymbol{v}}\big)$ is a global solution to* (15)*. In particular, let*

$$\widehat{\boldsymbol{s}} \doteq \widehat{\boldsymbol{u}} \odot \widehat{\boldsymbol{u}} - \widehat{\boldsymbol{v}} \odot \widehat{\boldsymbol{v}}, \tag{25}$$

  *then $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}})$ is an optimal solution to the convex program*

$$\min_{\boldsymbol{\theta},\,\boldsymbol{s}} \frac{1}{2}\|\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{s}\|_1, \quad \text{s.t.} \quad \boldsymbol{y} = \boldsymbol{J}\boldsymbol{\theta} + \boldsymbol{s}. \tag{26}$$

As we observe from the above result, because $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}})$ that the gradient flow (17) converges to is also an optimal solution of (26), it implies that $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}})$ is regularized. In particular, the $\ell_1$-norm regularization on $\boldsymbol{s}$ comes as a result of implicit regularization on overparameterization $\boldsymbol{s} = \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}$, leading to a sparse solution on $\boldsymbol{s}$ as we desired. On the other hand, the $\ell_2$ regularization on $\boldsymbol{\theta}$ leads to the desired minimum $\ell_2$-norm solution as we discussed in (14). Thus, the only question remains is whether the ground truth $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ in (13) can be identified through solving the convex problem (26), which we will discuss in the following Section 3.3.

**Numerical verification.** While Proposition 3.3 is proved for gradient flow with both learning rate $\tau \to 0$ and initialization scale $\gamma \to 0$, we numerically show that such a result also holds non-asymptotically with finitely small $\tau$ and $\gamma$.

Given a tuple $(N, p, r, k)$ of model parameters, we generate simulation data $(\boldsymbol{J}, \boldsymbol{\theta}_\star, \boldsymbol{s}_\star, \boldsymbol{y})$ as follows. The matrix $\boldsymbol{J} \in \mathbb{R}^{N\times p}$ is generated by multiplying two randomly generated matrices of shape $N \times r$ and $r \times p$, respectively with entries drawn i.i.d. from a standard Gaussian distribution. The sparse vector $\boldsymbol{s}_\star \in \mathbb{R}^N$ is generated by randomly choosing $k$ entries to be i.i.d. standard Gaussian, with the rest of entries zero. Then, we generate a vector $\boldsymbol{\theta} \in \mathbb{R}^p$ with all entries drawn i.i.d. from a standard Gaussian distribution, and let $\boldsymbol{y} = \boldsymbol{J}\boldsymbol{\theta} + \boldsymbol{s}_\star$. Finally, we set $\boldsymbol{\theta}_\star$ as the minimum $\ell_2$-norm solution according to (14).

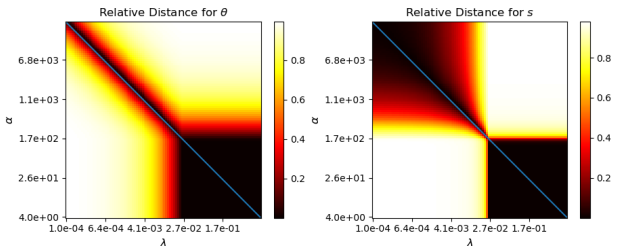



*Figure 2.* The gradient descent in (16) and the convex problem in (26) produce the same solutions with $\alpha = -\frac{\log\gamma}{2\lambda}$. For fixed data $(\boldsymbol{J}, \boldsymbol{y})$, left figure shows the relative difference $\frac{\|\boldsymbol{\theta}_\alpha - \boldsymbol{\theta}_\lambda\|_2}{\max\{\|\boldsymbol{\theta}_\alpha\|_2, \|\boldsymbol{\theta}_\lambda\|_2\}}$ between the solution $\boldsymbol{\theta}_\alpha$ to (16) with varying values of $\alpha$ (in y-axis) and the solution $\boldsymbol{\theta}_\lambda$ computed from (26) with varying values of $\lambda$ (in x-axis). Likewise, right figure shows the relative difference for $\boldsymbol{s}$. Blue line shows the curve $\alpha = -\frac{\log\gamma}{2\lambda}$ where $\gamma$ is fixed to $\exp(-8)$ in all experiments.

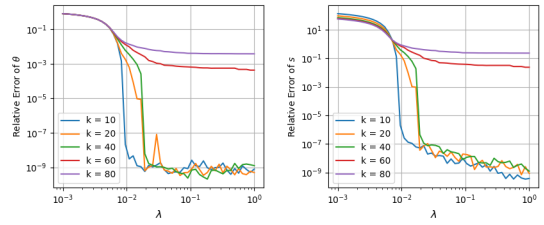
In this experiment, we choose and fix $(N, p, r, k) = (20, 40, 3, 3)$ for the data generation described above. With a varying learning rate $\alpha \in [4, 4000]$, we compute $(\boldsymbol{\theta}_\alpha, \boldsymbol{s}_\alpha)$ as the solution provided by gradient descent in (16) with an initialization by (19) with $\gamma = e^{-8}$. With a varying regularization $\lambda \in [0.0001, 1]$ in (26), we compute $(\boldsymbol{\theta}_\lambda, \boldsymbol{s}_\lambda)$ as the solution provided by the convex problem in (26) with weight parameter $\lambda$, using the ECOS solver (Domahidi et al., 2013) provided in CVXPY (Diamond & Boyd, 2016). Figure 2 provides a visualization of the relative difference $\rho = \frac{\|\boldsymbol{\theta}_\alpha - \boldsymbol{\theta}_\lambda\|_2}{\max\{\|\boldsymbol{\theta}_\alpha\|_2, \|\boldsymbol{\theta}_\lambda\|_2\}}$ between $\boldsymbol{\theta}_\alpha$ and $\boldsymbol{\theta}_\lambda$ (and likewise for $\boldsymbol{s}$), across all pairs of $(\alpha, \lambda)$. We can observe that as long as $(\alpha, \lambda)$ satisfies the relationship in (23), the relative difference $\rho$ is small for $\boldsymbol{\theta}$, which is also true for $\boldsymbol{s}$. On the other hand, the relative differences can be large if (23) is not satisfied, corroborating Proposition 3.3.

### 3.3. Exact Recovery under Incoherence Conditions

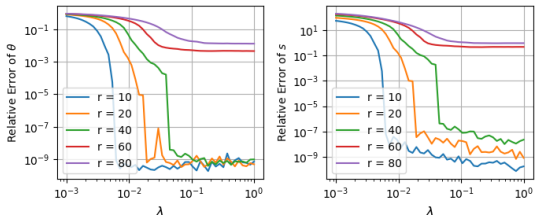
Given the overparameterized model (13) with $\boldsymbol{y} \in \mathbb{R}^N$, $\boldsymbol{\theta} \in \mathbb{R}^p$, and $p \gg N$, there is no enough information from $\boldsymbol{y}$ to recover $\boldsymbol{\theta}_\star$ and $\boldsymbol{s}_\star$ even with the prior information that $\boldsymbol{s}_\star$ is sparse – any given vector $\boldsymbol{y} \in \mathbb{R}^p$ can be decomposed as a summation of an arbitrary sparse vector $\boldsymbol{s}$ and a vector $\boldsymbol{\theta}$ cooked up from the column space of $\boldsymbol{J}$ *as long as* $\boldsymbol{J}$ *has full row rank*.

For the solution $\boldsymbol{\theta}_\star$ and $\boldsymbol{s}_\star$ to be identified, first, we assume that $\boldsymbol{J}$ is *low-rank*, motivated by empirically observation in practical deep neural network $f_{\boldsymbol{\theta}}$ that the Jacobian matrix $\boldsymbol{J}$ of $f_{\boldsymbol{\theta}}$ is approximately low-rank (Oymak et al., 2019).[2] However, the low-rank condition of $\boldsymbol{J}$ alone does *not* guarantee identifiability, because it cannot address the separability

[2]The low-rank assumption simplifies our analysis but at the cost that our model is not able to overfit to any corrupted labels as a deep neural network. We leave the study under a realistic *approximate* low-rank assumption to future work.




(a) Varying $k$ with fixed $r = 20$.




(b) Varying $r$ with fixed $k = 20$.

*Figure 3.* Effect of model parameter $\lambda$ for exact recovery by (26). The $y$-axis is the relative error of $\boldsymbol{\theta}$ (left) and $\boldsymbol{s}$ (right) defined as $\frac{\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_\star\|_2}{\|\boldsymbol{\theta}_\star\|_2}$ and $\frac{\|\boldsymbol{s} - \boldsymbol{s}_\star\|_2}{\|\boldsymbol{s}_\star\|_2}$, respectively, where $(\boldsymbol{\theta}, \boldsymbol{s})$ is the solution to (26). The curves are averages over 10 independent trials.

between $\boldsymbol{J}\boldsymbol{\theta}_\star$ and $\boldsymbol{s}_\star$ – following a similar argument as that in (Candès et al., 2011), if any column of $\boldsymbol{J}$ has a single nonzero entry, then any $\boldsymbol{s}_\star$ that is supported on the same entry cannot be recovered without ambiguities. Hence, we further assume that the column space of $\boldsymbol{J}$ and the standard basis $[\boldsymbol{e}_1, \ldots, \boldsymbol{e}_N] \doteq \mathrm{diag}\{1, \ldots, 1\} \in I\!R^{N\times N}$ are *incoherent*, defined as follows.

**Definition 3.4** (Candès et al. (2011))**.** Let $\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \in I\!R^{N\times p}$ be the compact SVD of $\boldsymbol{J}$ and $r$ be the rank of $\boldsymbol{J}$. The coherence of $\boldsymbol{J}$ (w.r.t. the standard basis) is defined as

$$\mu(\boldsymbol{J}) = \frac{N}{r} \max_{1\le i\le N} \|\boldsymbol{U}^\top \boldsymbol{e}_i\|_2^2. \tag{27}$$

It should be noted that the low-rank and incoherence assumptions are common for matrix recovery (Davenport & Romberg, 2016; Chi et al., 2019). Based upon the above assumptions on $\boldsymbol{J}$ and $\boldsymbol{s}_\star$, we show the following.

**Proposition 3.5.** *Let $r$ be the rank of $\boldsymbol{J}$ and $k$ be the number of nonzero entries of $\boldsymbol{s}_\star$. If we have*

$$k^2 r < \frac{N}{4\mu(\boldsymbol{J})}, \tag{28}$$

*then the solution to (26) is $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ for any $\lambda > \lambda_0$, where $\lambda_0 > 0$ is a scalar depending on $(\boldsymbol{J}, \boldsymbol{\theta}_\star, k)$.*

Thus, combining this result with Proposition 3.3, the gradient flow in (17) with initialization (19) converges to $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ when the choice of learning rate ratio $\alpha$ in (17) is smaller than a certain threshold, justifying our claim in Theorem 3.1.

**Numerical verification.** To corroborate Proposition 3.5, we numerically solve (26) under varying conditions of $\lambda$, $r$, and $k$. The simulated data $(\boldsymbol{J}, \boldsymbol{\theta}_\star, \boldsymbol{s}_\star, \boldsymbol{y})$ is generated

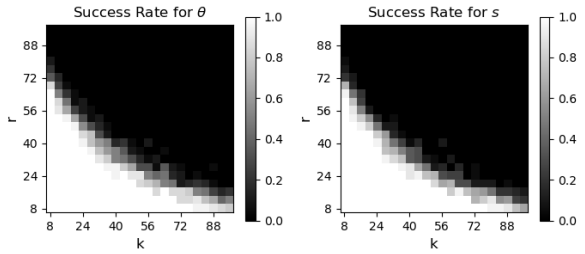



*Figure 4.* Phase transition for solving (26) over 20 trials, with fixed $\lambda = 0.1$ and varying $k$, $r$. Recovery is declared success if $\frac{\|\boldsymbol{\theta}-\boldsymbol{\theta}_\star\|_2}{\|\boldsymbol{\theta}_\star\|_2} < 0.001$ (left) and $\frac{\|\boldsymbol{s}-\boldsymbol{s}_\star\|_2}{\|\boldsymbol{s}_\star\|_2} < 0.001$ (right).

the same way as the experimental part in Section 3.2 with $N = 100$ and $p = 150$, and for an obtained solution $(\boldsymbol{\theta}, \boldsymbol{s})$ via solving (26), we measure the relative recovery error $\epsilon_{\boldsymbol{\theta}} = \frac{\|\boldsymbol{\theta}-\boldsymbol{\theta}_\star\|_2}{\|\boldsymbol{\theta}_\star\|_2}$ and $\epsilon_{\boldsymbol{s}} = \frac{\|\boldsymbol{s}-\boldsymbol{s}_\star\|_2}{\|\boldsymbol{s}_\star\|_2}$.

- *Effects of the parameter* $\lambda$. Here, we consider the recovery with varying $\lambda \in [0.0001, 1]$. First, we fix $r = 20$ and vary $k \in \{10, 20, 40, 60, 80\}$, showing the relative recovery errors $\epsilon_{\boldsymbol{\theta}}$ and $\epsilon_{\boldsymbol{s}}$ in Figure 3(a). Second, we fix $k = 20$ and vary $r \in \{10, 20, 40, 60, 80\}$, showing the results in Figure 3(b). The results show a clear phase transition that correct recovery is obtained only when $\lambda$ is greater than a particular threshold $\lambda_0$. Moreover, $\lambda_0$ varies depending on $k$ and $r$, consistent with Proposition 3.5.
- *Relationships between the rank* $r$ *and sparsity* $k$. Here, we fix $\lambda = 0.1$ and plot the phase transition with respect to $r$ and $k$. For each $(r, k)$, the simulation is repeated for 20 random instances, and for each instance we declare the recovery to be successful if $\epsilon_{\boldsymbol{\theta}} < 0.001$ and $\epsilon_{\boldsymbol{s}} < 0.001$. As shown in Figure 4, the phase transition is consistent with Proposition 3.5 that successful recovery is achieved only when both $k$ and $r$ are small.

# 4. Related Work and Discussion

## 4.1. Prior Arts on Implicit Regularization

Since overparameterized deep neural networks do not overfit (in the absence of data corruption) even without any explicit regularization (Zhang et al., 2021a), it is argued that there are implicit regularizations pf learning algorithms that enable the models to converge to desired solutions. Under the assumption of linear or deep linear models, many work characterized the mathematics of such implicit bias via explicit regularizations (Soudry et al., 2018; Gunasekar et al., 2018; Li et al., 2018; Oymak & Soltanolkotabi, 2019; Arora et al., 2019; Razin & Cohen, 2020; Li et al., 2020c; Ji et al., 2020; Stöger & Soltanolkotabi, 2021; Jacot et al., 2021). Among those, the closest related to ours include (Vaskevicius et al., 2019; Zhao et al., 2019; Woodworth et al., 2020; Li et al., 2021a; Chou et al., 2021), which studied the implicit *sparse* regularization induced by a term of the form $\boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}$. While all the above works aim to understand implicit regularization by studying linear models, the practical benefits of such studies are unclear. Our work provides an inspiring result showing that principled design with implicit regularization leads to robust learning of over-parameterized models. In particular, our model in (2) is motivated by existing studies on the implicit sparse regularization, but adds such a regularization to an (already) implicitly regularized model for handling sparse corruptions. In other words, two forms of implicit regularization are involved in our model which poses new problems in the design of the optimization algorithm and in mathematical analysis. To the best of our knowledge, the only prior works that use implicit sparse regularization for robust learning are (You et al., 2020; Ma & Fattahi, 2021; Ding et al., 2021) which studied the robust recovery of low-rank matrices and images. Among them, our work extends (You et al., 2020) to the problem of image classification with label noise, demonstrates its effectiveness, and provides dedicated theoretical analyses. Additionally, methods in (Ma & Fattahi, 2021; Ding et al., 2021) require a particular learning rate schedule that may not be compatible with commonly used schedules such as cosine annealing (Loshchilov & Hutter, 2016) in image classification.

## 4.2. Relationship to Existing Work on Label Noise

Deep neural networks are over-parameterized hence prone to *overfitting* to the label noises. While many popular regularization techniques for alleviating overfitting, such as label smoothing (Szegedy et al., 2016; Lukasik et al., 2020; Wei et al., 2021a) and *mixup* (Zhang et al., 2018), are useful for mitigating the impact of label noise, they do not completely solve the problem due to a lack of precise noise modeling. In the following, we discuss three of the most popular line of work dedicated to the label noise problem; we refer the reader to the survey papers (Algan & Ulusoy, 2021; Song et al., 2020; Wei et al., 2021b) for a comprehensive review.

**Loss design.** Robust loss function, such as the $\ell_1$ loss (Ghosh et al., 2017), is one of the most popular approaches to the label noise problem which has many recent extensions (Zhang & Sabuncu, 2018; Wang et al., 2019; Amid et al., 2019; Ma et al., 2020; Yu et al., 2020; Wei & Liu, 2021). The method is based on reducing the loss associated with large outlying entries, hence the impact of label noise. A similar idea is also explored in gradient clipping (Menon et al., 2019) and loss reweighting (Liu & Tao, 2015; Wang et al., 2017; Chang et al., 2017; Zhang et al., 2021b; Zetterqvist et al., 2021) methods. While robust loss enables the model to learn faster from correct labels, *the global solution still overfits to corrupted labels with over-parameterized models*.

**Label transition probability.** Another popular line of work for label noise is based on the assumption that the noisy label is drawn from a probability distribution conditioned on

the true label. Here, the main task is to estimate the underlying transition probabilities. The early work (Chen & Gupta, 2015; Goldberger & Ben-Reuven, 2017) encapsulates the transition probabilities as a noise adaptation layer that is stacked on top of a classification network and trained jointly in an end-to-end fashion. Recent work (Patrini et al., 2017) uses separated procedures to estimate the transition probabilities, the success of which requires either the availability of a clean validation data (Hendrycks et al., 2018) or additional data assumptions (Xia et al., 2019; Zhu et al., 2021; Li et al., 2021b; Zhang et al., 2021c). Even if the underlying transition probabilities can be correctly recovered, *overfitting is only prevented asymptotically, requiring sufficiently many samples of corrupted labels for each input (Patrini et al., 2017), which is not practical.*

**Label correction.** In contrast to the above methods, our method completely avoids overfitting even with finite training samples. This is achieved by the over-parameterization term $\boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v}$ in (2) which recovers the clean labels. Hence, our method is related to techniques based on noisy label detection and refurbishment. Nonetheless, existing techniques are based on heuristic argument about different behaviors of clean and corrupted samples in the training process, such as properties of learned representations (Kim et al., 2021; Ma et al., 2018; Jiang et al., 2020), prediction consistency (Reed et al., 2014; Song et al., 2019a), learning speed (Li et al., 2020b; Liu et al., 2020; 2021a), margin (Lin & Bradic, 2021), confidence (Cheng et al., 2021). They often need to be combined with engineering tricks such as moving average (Huang et al., 2020; Liu et al., 2020) and burning-in (Zheng et al., 2020) to make them work well. Finally, the work (Hu et al., 2019) introduces a variable to estimate the label noise in a way similar to (2). However, the variable is not over-parameterized to induce sparsity, and their method does not have competitive performance.

### 4.3. Sparsity in Deep Learning

Our method is broadly related to existing efforts on introducing sparsity into deep learning (Hoefler et al., 2021), but is notably different in both the objective of introducing sparsity, the origin of sparsity, and how sparsity is enforced. First, previous exploration of sparsity primarily aims to improve training and inference efficiency with large-scale models, while our paper focuses on robust training under label noise. Second, previous introduction of sparsity is often motivated by its presence in biological brains, but there is still a lack of clean understanding of how sparsity helps with learning. In contrast, sparsity in our method has the clear mathematically meaning that the percentage of corrupted labels is small. Finally, while pruning (Liu et al., 2021b; Chen et al., 2021) is a dominant approach for obtaining sparsity, our method leverages the implicit bias of gradient descent associated with a particular sparse over-parameterization.

### 4.4. Limitations and Future Directions

**Choice of optimization algorithms.** Our SOP method is based on introducing more parameters to an already over-parameterized model, hence relies critically on the choice of the optimization algorithm to induce the desired implicit regularization. For *vanilla* gradient descent, our analysis in Section 3 shows that it has the desired implicit regularization by design. In practical deep network training, it is more common to use the *stochastic* gradient descent *with momentum*. While not theoretically justified, experiments in Section 2 show that our method works with such practical variants. This may not come as a surprise, because existing studies already show that stochastic gradient descent (Nacson et al., 2019) and momentum acceleration (Wang et al., 2021) have the same implicit bias as the vanilla gradient decent under certain models. We leave the extension of such results to our method as future works.

**Modeling of label noise.** Our method is based on the assumption that the label noise matrix $\boldsymbol{S}_\star = [\boldsymbol{s}_{\star 1}, \ldots, \boldsymbol{s}_{\star N}]$, where $\boldsymbol{s}_{\star i}$ is the difference between the observed label $\boldsymbol{y}_i$ and the underlying true label, is a *sparse* matrix. We made no additional assumption on the sparsity pattern of $\boldsymbol{S}_\star$, other than the non-negative and non-positive constraints discussed in (4). In practice, it is usually the case that certain pairs of classes are more similar hence more easily confusing with each other than other pairs. As a result, certain blocks of $\boldsymbol{S}_\star$ tend to have more non-zero entries than the others. When there is a prior on which blocks may have more non-zero entries, our method may be adapted by using a *weighted* sparse regularization for the corresponding blocks. When there is no such prior, our method may be adapted by using a *group* sparse regularization (Neyshabur et al., 2014; Tibshirani, 2021).

## Acknowledgements

SL and QQ were partially supported by NSF grant DMS 2009752. SL was partially supported by NSF NRT-HDR Award 1922658 and Alzheimer's Association grant AARG-NTF-21-848627. QQ also acknowledge support from NSF CAREER 2143904, NSF CCF 2212066, and ONR N00014-22-1-2529. ZZ acknowledges support from NSF grants CCF 2008460 and CCF 2106881. Part of this work was done when CY was at University of California, Berkeley and was supported by Tsinghua-Berkeley Shenzhen Institute Research Fund. The authors acknowledge helpful discussion with Ryan Chan from Johns Hopkins University.

## References

Algan, G. and Ulusoy, I. Image classification with deep learning in the presence of noisy labels: A survey. *Knowledge-Based Systems*, 215:106771, 2021.

Amid, E., Warmuth, M. K., Anil, R., and Koren, T. Robust bi-tempered logistic loss based on bregman divergences. *Advances in Neural Information Processing Systems*, 32: 15013–15022, 2019.

Arora, S., Cohen, N., Hu, W., and Luo, Y. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, pp. 7411–7422, 2019.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A. Mixmatch: A holistic approach to semi-supervised learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Candes, E. J. and Tao, T. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

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

Chang, H.-S., Learned-Miller, E., and McCallum, A. Active bias: Training more accurate neural networks by emphasizing high variance samples. *Advances in Neural Information Processing Systems*, 30:1002–1012, 2017.

Chen, T., Zhang, Z., Balachandra, S., Ma, H., Wang, Z., Wang, Z., et al. Sparsity winning twice: Better robust generalization from more efficient training. In *International Conference on Learning Representations*, 2021.

Chen, X. and Gupta, A. Webly supervised learning of convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1431–1439, 2015.

Cheng, H., Zhu, Z., Li, X., Gong, Y., Sun, X., and Liu, Y. Learning with instance-dependent label noise: A sample sieve approach. In *International Conference on Learning Representations*, 2021.

Chi, Y., Lu, Y. M., and Chen, Y. Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269, 2019.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. *Advances in Neural Information Processing Systems*, 32:2937–2947, 2019.

Chou, H.-H., Maly, J., and Rauhut, H. More is less: Inducing sparsity via overparameterization. *arXiv preprint arXiv:2112.11027*, 2021.

Claerbout, J. F. and Muir, F. Robust modeling with erratic data. *Geophysics*, 38(5):826–844, 1973.

Cohen, A., Dahmen, W., and DeVore, R. Compressed sensing and best $k$-term approximation. *Journal of the American mathematical society*, 22(1):211–231, 2009.

Davenport, M. A. and Romberg, J. An overview of low-rank matrix recovery from incomplete observations. *IEEE Journal of Selected Topics in Signal Processing*, 10(4): 608–622, 2016.

Diamond, S. and Boyd, S. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Ding, L., Jiang, L., Chen, Y., Qu, Q., and Zhu, Z. Rank overspecified robust matrix recovery: Subgradient method and exact recovery. *Advances in Neural Information Processing Systems*, 34, 2021.

Domahidi, A., Chu, E., and Boyd, S. ECOS: An SOCP solver for embedded systems. In *European Control Conference (ECC)*, pp. 3071–3076, 2013.

Frénay, B. and Verleysen, M. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2013.

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pp. 797–842. PMLR, 2015.

Ghosh, A., Kumar, H., and Sastry, P. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Goldberger, J. and Ben-Reuven, E. Training deep neural-networks using a noise adaptation layer. 2017.

Gunasekar, S., Woodworth, B., Bhojanapalli, S., Neyshabur, B., and Srebro, N. Implicit regularization in matrix factorization. In *2018 Information Theory and Applications Workshop (ITA)*, pp. 1–10. IEEE, 2018.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hendrycks, D., Mazeika, M., Wilson, D., and Gimpel, K. Using trusted data to train deep networks on labels corrupted by severe noise. *Advances in Neural Information Processing Systems*, 31:10456–10465, 2018.

Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., and Peste, A. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *Journal of Machine Learning Research*, 22(241):1–124, 2021.

Hu, W., Li, Z., and Yu, D. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *International Conference on Learning Representations*, 2019.

Huang, L., Zhang, C., and Zhang, H. Self-adaptive training: beyond empirical risk minimization. *arXiv preprint arXiv:2002.10319*, 2020.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

Jacot, A., Ged, F., Gabriel, F., Simsek, B., and Hongler, C. Deep linear networks dynamics: Low-rank biases induced by initialization scale and l2 regularization. *arXiv preprint arXiv:2106.15933*, 2021.

Ji, Z., Dudík, M., Schapire, R. E., and Telgarsky, M. Gradient descent follows the regularization path for general losses. In *Conference on Learning Theory*, pp. 2109–2136. PMLR, 2020.

Jiang, L., Huang, D., Liu, M., and Yang, W. Beyond synthetic noise: Deep learning on controlled noisy labels. In *International Conference on Machine Learning*, pp. 4804–4815. PMLR, 2020.

Kalimeris, D., Kaplun, G., Nakkiran, P., Edelman, B., Yang, T., Barak, B., and Zhang, H. Sgd on neural networks learns functions of increasing complexity. *Advances in Neural Information Processing Systems*, 32:3496–3506, 2019.

Kim, T., Ko, J., Choi, J., Yun, S.-Y., et al. Fine samples for learning with noisy labels. *Advances in Neural Information Processing Systems*, 34, 2021.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Lee, J. D., Simchowitz, M., Jordan, M. I., and Recht, B. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257. PMLR, 2016.

Li, J., Wong, Y., Zhao, Q., and Kankanhalli, M. S. Learning to learn from noisy labeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5051–5059, 2019.

Li, J., Socher, R., and Hoi, S. C. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020a.

Li, J., Nguyen, T., Hegde, C., and Wong, K. W. Implicit sparse regularization: The impact of depth and early stopping. *Advances in Neural Information Processing Systems*, 34, 2021a.

Li, M., Soltanolkotabi, M., and Oymak, S. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International conference on artificial intelligence and statistics*, pp. 4313–4324. PMLR, 2020b.

Li, W., Wang, L., Li, W., Agustsson, E., and Van Gool, L. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

Li, X., Liu, T., Han, B., Niu, G., and Sugiyama, M. Provably end-to-end label-noise learning without anchor points. *arXiv preprint arXiv:2102.02400*, 2021b.

Li, Y., Ma, T., and Zhang, H. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. In *Conference On Learning Theory*, pp. 2–47, 2018.

Li, Z., Luo, Y., and Lyu, K. Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. In *International Conference on Learning Representations*, 2020c.

Lin, J. Z. and Bradic, J. Learning to combat noisy labels via classification margins. *arXiv preprint arXiv:2102.00751*, 2021.

Liu, S., Niles-Weed, J., Razavian, N., and Fernandez-Granda, C. Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, 33, 2020.

Liu, S., Liu, K., Zhu, W., Shen, Y., and Fernandez-Granda, C. Adaptive early-learning correction for segmentation from noisy annotations. *ArXiv*, abs/2110.03740, 2021a.

Liu, S., Yin, L., Mocanu, D. C., and Pechenizkiy, M. Do we actually need dense over-parameterization? in-time over-parameterization in sparse training. In *International Conference on Machine Learning*, pp. 6989–7000. PMLR, 2021b.

Liu, T. and Tao, D. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2015.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Lukasik, M., Bhojanapalli, S., Menon, A., and Kumar, S. Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pp. 6448–6458. PMLR, 2020.

Ma, J. and Fattahi, S. Implicit regularization of sub-gradient method in robust matrix recovery: Don't be afraid of outliers. *arXiv preprint arXiv:2102.02969*, 2021.

Ma, X., Wang, Y., Houle, M. E., Zhou, S., Erfani, S., Xia, S., Wijewickrema, S., and Bailey, J. Dimensionality-driven learning with noisy labels. In *International Conference on Machine Learning*, pp. 3355–3364. PMLR, 2018.

Ma, X., Huang, H., Wang, Y., Romano, S., Erfani, S., and Bailey, J. Normalized loss functions for deep learning with noisy labels. In *International Conference on Machine Learning*, pp. 6543–6553. PMLR, 2020.

Menon, A. K., Rawat, A. S., Reddi, S. J., and Kumar, S. Can gradient clipping mitigate label noise? In *International Conference on Learning Representations*, 2019.

Nacson, M. S., Srebro, N., and Soudry, D. Stochastic gradient descent on separable data: Exact convergence with a fixed learning rate. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3051–3059. PMLR, 2019.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

Oymak, S. and Soltanolkotabi, M. Overparameterized nonlinear learning: Gradient descent takes the shortest path? In *International Conference on Machine Learning*, pp. 4951–4960, 2019.

Oymak, S., Fabian, Z., Li, M., and Soltanolkotabi, M. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392*, 2019.

Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1944–1952, 2017.

Razin, N. and Cohen, N. Implicit regularization in deep learning may not be explainable by norms. *Advances in Neural Information Processing Systems*, 33, 2020.

Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., and Rabinovich, A. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Song, H., Kim, M., and Lee, J.-G. Selfie: Refurbishing unclean samples for robust deep learning. In *International Conference on Machine Learning*, pp. 5907–5915. PMLR, 2019a.

Song, H., Kim, M., Park, D., and Lee, J.-G. Prestopping: How does early stopping help generalization against label noise? *ArXiv*, abs/1911.08059, 2019b.

Song, H., Kim, M., Park, D., Shin, Y., and Lee, J.-G. Learning from noisy labels with deep neural networks: A survey. *arXiv preprint arXiv:2007.08199*, 2020.

Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

Stöger, D. and Soltanolkotabi, M. Small random initialization is akin to spectral learning: Optimization and generalization guarantees for overparameterized low-rank matrix reconstruction. *Advances in Neural Information Processing Systems*, 34, 2021.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Tanaka, D., Ikami, D., Yamasaki, T., and Aizawa, K. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5552–5560, 2018.

Tibshirani, R. J. Equivalences between sparse models and neural networks. 2021.

Vaskevicius, T., Kanade, V., and Rebeschini, P. Implicit regularization for optimal sparse recovery. In *Advances in Neural Information Processing Systems*, pp. 2968–2979, 2019.

Wang, B., Meng, Q., Zhang, H., Sun, R., Chen, W., and Ma, Z.-M. Momentum doesn't change the implicit bias. *arXiv preprint arXiv:2110.03891*, 2021.

Wang, R., Liu, T., and Tao, D. Multiclass learning with partially corrupted labels. *IEEE transactions on neural networks and learning systems*, 29(6):2568–2580, 2017.

Wang, Y., Ma, X., Chen, Z., Luo, Y., Yi, J., and Bailey, J. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 322–330, 2019.

Wei, J. and Liu, Y. When optimizing $f$-divergence is robust with label noise. In *International Conference on Learning Representations*, 2021.

Wei, J., Liu, H., Liu, T., Niu, G., and Liu, Y. Understanding (generalized) label smoothing whenlearning with noisy labels. *arXiv preprint arXiv:2106.04149*, 2021a.

Wei, J., Zhu, Z., Cheng, H., Liu, T., Niu, G., and Liu, Y. Learning with noisy labels revisited: A study using real-world human annotations. *arXiv preprint arXiv:2110.12088*, 2021b.

Woodworth, B., Gunasekar, S., Lee, J. D., Moroshko, E., Savarese, P., Golan, I., Soudry, D., and Srebro, N. Kernel and rich regimes in overparametrized models. In *Conference on Learning Theory*, pp. 3635–3673. PMLR, 2020.

Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., and Ma, Y. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2008.

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G., and Sugiyama, M. Are anchor points really indispensable in label-noise learning? *Advances in Neural Information Processing Systems*, 32:6838–6849, 2019.

Xia, X., Liu, T., Han, B., Gong, C., Wang, N., Ge, Z., and Chang, Y. Robust early-learning: Hindering the memorization of noisy labels. In *International Conference on Learning Representations*, 2020.

Xiao, T., Xia, T., Yang, Y., Huang, C., and Wang, X. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2691–2699, 2015.

Xie, Q., Dai, Z., Hovy, E. H., Luong, M.-T., and Le, Q. V. Unsupervised data augmentation for consistency training. *arXiv: Learning*, 2020.

Yang, Z., Yu, Y., You, C., Steinhardt, J., and Ma, Y. Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*, pp. 10767–10777. PMLR, 2020.

You, C., Zhu, Z., Qu, Q., and Ma, Y. Robust recovery via implicit bias of discrepant learning rates for double over-parameterization. *Advances in Neural Information Processing Systems*, 33:17733–17744, 2020.

Yu, Y., Chan, K. H. R., You, C., Song, C., and Ma, Y. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. *Advances in Neural Information Processing Systems*, 33:9422–9434, 2020.

Zetterqvist, O., Jörnsten, R., and Jonasson, J. Robust neural network classification via double regularization. *arXiv preprint arXiv:2112.08102*, 2021.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021a.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Zhang, H., Xing, X., and Liu, L. Dualgraph: A graph-based method for reasoning about label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9654–9663, 2021b.

Zhang, Y., Niu, G., and Sugiyama, M. Learning noise transition matrix from only noisy labels via total variation regularization. *arXiv preprint arXiv:2102.02414*, 2021c.

Zhang, Z. and Sabuncu, M. R. Generalized cross entropy loss for training deep neural networks with noisy labels. In *32nd Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

Zhao, P., Yang, Y., and He, Q.-C. Implicit regularization via hadamard product over-parametrization in high-dimensional linear regression. *arXiv preprint arXiv:1903.09367*, 2019.

Zheng, S., Wu, P., Goswami, A., Goswami, M., Metaxas, D., and Chen, C. Error-bounded correction of noisy labels. In *International Conference on Machine Learning*, pp. 11447–11457. PMLR, 2020.

Zhu, Z., Song, Y., and Liu, Y. Clusterability as an alternative to anchor points when learning with noisy labels. *arXiv preprint arXiv:2102.05291*, 2021.

# Appendices

This appendix is organized as follows. In Section A we provide additional details for reproducing experimental results presented in Section 2. In Section B we provide proofs for the theoretical results presented in Section 3.

## A. Training Details for Robust Classification with Label Noise

### A.1. Choice of Loss Function

The cross-entropy loss in (8) cannot be used to optimize $\{\boldsymbol{v}_i\}$ as we explain below. Consider a data point $\boldsymbol{x}$ with a one-hot label $\boldsymbol{y}$. With the CE loss in (8) rewritten below for convenience:

$$L_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{x}, \boldsymbol{y}) \doteq \ell_{\mathrm{CE}}\Big(\phi\big(f(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{s}\big),\, \boldsymbol{y}\Big),\ \text{with } \boldsymbol{s} \doteq \boldsymbol{u} \odot \boldsymbol{u} \odot \boldsymbol{y} - \boldsymbol{v} \odot \boldsymbol{v} \odot (1 - \boldsymbol{y}), \tag{A.1}$$

we may compute its gradient with respect to (w.r.t.) $\boldsymbol{v}$ as

$$\frac{\partial L_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{v}} = \frac{2\boldsymbol{v} \odot (1 - \boldsymbol{y})}{\mathbf{1}^\top (f(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{s})}. \tag{A.2}$$

This shows that the gradient w.r.t. different entries of $\boldsymbol{v}$ does not depend on the output $f(\boldsymbol{x}, \boldsymbol{\theta})$ of the model at all modulo the divider shared by all entries. Hence, $\boldsymbol{v}$ cannot correctly learn the label noise.

We now consider the MSE loss in (9) rewritten below for convenience:

$$L_{\mathrm{MSE}}(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{x}, \boldsymbol{y}) \doteq \ell_{\mathrm{MSE}}\Big(f(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{s},\, \boldsymbol{y}\Big),\ \text{with } \boldsymbol{s} \doteq \boldsymbol{u} \odot \boldsymbol{u} \odot \boldsymbol{y} - \boldsymbol{v} \odot \boldsymbol{v} \odot (1 - \boldsymbol{y}). \tag{A.3}$$

The gradient w.r.t. $\boldsymbol{v}$ can be computed as

$$\frac{\partial L_{\mathrm{MSE}}(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}; \boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{v}} = 4(f(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{s} - \boldsymbol{y}) \odot \boldsymbol{v} \odot (1 - \boldsymbol{y}). \tag{A.4}$$

Here the gradient w.r.t. different entries of $\boldsymbol{v}$ varies depending on how well the model prediction $f(\boldsymbol{x}, \boldsymbol{\theta}) + \boldsymbol{s}$ matches the given label $\boldsymbol{y}$ at the corresponding entry. Hence, when the model prediction deviates from the given label which may occur when the label is corrupted, $\boldsymbol{v}$ is able to learn the underlying corruption to the label.

### A.2. Definition of Label Noise

In this paper, we consider two types of widely existed label noise, namely symmetric label noise and asymmetric label noise. For symmetric noise with noise level $\alpha$, the labels are generated as follows:

$$y = \begin{cases} y^{GT} \text{ with the probability of } 1 - \alpha \\ \text{random one hot vector with the probability of } \alpha. \end{cases}$$

We consider noise level $\alpha \in \{0.2, 0.4, 0.5, 0.6, 0.8\}$. For asymmetric noise, following (Patrini et al., 2017), we flip labels between TRUCK $\rightarrow$AUTOMOBILE, BIRD $\rightarrow$ AIRPLANE, DEER $\rightarrow$ HORSE, and CAT $\leftrightarrow$ DOG. We randomly choose 40% training data with their labels to be flipped according to this asymmetric labeling rule. For real world datasets, Clothing1M has noise level estimated at around 38.5% (Song et al., 2019b), and for WebVision, the noise level is estimated to be at around 20% (Li et al., 2017).

### A.3. Implementation Details of SOP+

We considered two separate regularization terms to further boost the results and stabilize training. We will describe the definitions and roles of them below:

**Consistency regularizer $\mathcal{L}_C$.** We use a regularizer $\mathcal{L}_C$ to encourage consistency of network prediction on a original image and the corresponding augmented image. Such a regularizer is commonly used in semi-supervised learning and label noise learning literature, see e.g., (Berthelot et al., 2019; Li et al., 2019). Specifically, the consistency regularizer $\mathcal{L}_C$ is defined as the Kullback-Leibler (KL)-divergence between the softmax predictions from the images with augmentations (described in Section A.4) and the softmax predictions for the corresponding images generated with Unsupervised Data Augmentation

*Table A.1.* Hyper-parameters for SOP on CIFAR-10/100, Clothing-1M and Webvision datasets.

| | CIFAR-10 | | CIFAR-100 | | Clothing-1M | Webvision |
|---|---|---|---|---|---|---|
| architecture | ResNet34 | PreAct PresNet18 | ResNet34 | PreAct PresNet18 | ResNet-50 (pretrained) | InceptionResNetV2 |
| batch size | 128 | 128 | 128 | 128 | 64 | 32 |
| learning rate (lr) | 0.02 | 0.02 | 0.02 | 0.0 2 | 0.002 | 0.02 |
| lr decay | 40th & 80th | Cosine Annealing | 40th & 80th | Cosine Annealing | 5th | 50th |
| weight decay (wd) | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ | $1 \times 10^{-3}$ | $5 \times 10^{-4}$ |
| training epochs | 120 | 300 | 150 | 300 | 10 | 100 |
| training examples | 45,000 | 50,000 | 45,000 | 50,000 | 1,000,000 | 66,000 |
| lr for $\{\boldsymbol{u}_i, \boldsymbol{v}_i\}$ | Sym: $\alpha_u = 10, \alpha_v = 10$<br>Asym: $\alpha_u = 10, \alpha_v = 100$ | Sym: $\alpha_u = 10, \alpha_v = 10$<br>Asym: $\alpha_u = 10, \alpha_v = 100$ | Sym: $\alpha_u = 1, \alpha_v = 10$<br>Asym: $\alpha_u = 1, \alpha_v = 100$ | Sym: $\alpha_u = 1, \alpha_v = 10$<br>Asym: $\alpha_u = 1, \alpha_v = 100$ | $\alpha_u = 0.1$ , $\alpha_v = 1$ | $\alpha_u = 0.1$ , $\alpha_v = 1$ |
| wd for $\{\boldsymbol{u}_i, \boldsymbol{v}_i\}$ | 0 | 0 | 0 | 0 | 0 | 0 |
| init. std for $\{\boldsymbol{u}_i, \boldsymbol{v}_i\}$ | $10^{-8}$ | $10^{-8}$ | $10^{-8}$ | $10^{-8}$ | $10^{-8}$ | $10^{-8}$ |
| $\lambda_C$ | 0.0 | 0.9 | 0.0 | 0.9 | 0.0 | 0.0 |
| $\lambda_B$ | 0.0 | 0.1 | 0.0 | 0.1 | 0.0 | 0.0 |

(UDA) (Xie et al., 2020):

$$\mathcal{L}_c(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} D_{KL} \left( f(\boldsymbol{x}_i; \boldsymbol{\theta}) \parallel f(\text{UDA}(\boldsymbol{x}_i); \boldsymbol{\theta}) \right).$$

**Class-balance regularizer $\mathcal{L}_B$.** We use a regularizer $\mathcal{L}_B$ to prevent the network from assigning all data points to the same class. Following (Tanaka et al., 2018), we use the prior information on the probability distribution $p$ of class labels and minimize its distance in terms of KL-divergence to the mean prediction of each batch $\mathcal{B}$:

$$\mathcal{L}_b(\boldsymbol{\theta}) = \sum_{k=1}^{K} p_k \log \frac{p_k}{\overline{f_k}(\boldsymbol{x}, \boldsymbol{\theta})} = -\sum_{k=1}^{K} p_k \log \overline{f_k}(\boldsymbol{x}; \boldsymbol{\theta}),$$

where $\overline{f_k}(\boldsymbol{x}; \boldsymbol{\theta}) \approx \frac{1}{|\mathcal{B}|} \sum_{\boldsymbol{x} \in \mathcal{B}} f(\boldsymbol{x}; \boldsymbol{\theta})$, and $p_k$ stands for the prior probability of the $k$th class.

The final loss function for SOP+ is therefore constructed by three terms as follows

$$L(\boldsymbol{\theta}, \{\boldsymbol{u}_i, \boldsymbol{v}_i\}) + \lambda_C \mathcal{L}_C(\boldsymbol{\theta}) + \lambda_B \mathcal{L}_B(\boldsymbol{\theta}),$$

where $\lambda_c, \lambda_B > 0$ are the hyper-parameters.

### A.4. Experimental Settings

**Data processing:** For experiments on CIFAR10/100 (Krizhevsky et al., 2009) without extra techniques, we use simple data augmentations including random crop and horizontal flip following previous works (Patrini et al., 2017; Liu et al., 2020). For SOP+, we use the default setting from unsupervised data augmentation (Xie et al., 2020) to apply efficient data augmentation to create another view of the data for consistency training. For Clothing-1M (Xiao et al., 2015), we first resize images to $256 \times 256$, and then random crop to $224 \times 224$, following a random horizontal flip. For WebVision (Li et al., 2017), we randomly crop the images into size of $227 \times 227$. All images are standardized by their means and variances.

**Hyper-parameters of SOP:** We adopt a SGD optimizer without weight decay for $U$ and $V$. We keep all the hyper-parameters fixed for different levels of noise. For fair comparison, we adopt two settings of hyper-parameters and architectures for SOP and SOP+. More details of hyper-parameters can be found in Table A.4. Note that the method is not very sensitive to hyper-parameters $\lambda_C$ and $\lambda_B$.

## B. Proofs for Theoretical Analysis with Linear Models

### B.1. Proof of Proposition 3.2

We first present a simple but useful lemma.

**Lemma B.1.** *Let $(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v})$ be a critical point to (15) that is not a global minimum, i.e.,*

$$\boldsymbol{r} \;:=\; \boldsymbol{J}\boldsymbol{\theta} + \boldsymbol{u} \odot \boldsymbol{u} - \boldsymbol{v} \odot \boldsymbol{v} - \boldsymbol{y} \neq \boldsymbol{0}.$$

*Then there exists an index $i$ such that*

$$u^i = v^i = 0,\ r^i \neq 0, \tag{B.1}$$

*where $u^i$, $v^i$, and $r^i$ denote the $i$-th elements of $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{r}$, respectively.*

*Proof.* We may compute the gradient of the objective function $h$ in (15) as

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) &= \boldsymbol{J}^\top \boldsymbol{r},\\ \nabla_{\boldsymbol{u}} h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) &= 2\boldsymbol{r} \odot \boldsymbol{u},\\ \nabla_{\boldsymbol{v}} h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) &= -2\boldsymbol{r} \odot \boldsymbol{v}.\end{aligned}$$

Since $\boldsymbol{r} \neq \boldsymbol{0}$ but $\nabla_{\boldsymbol{u}} h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) = \nabla_{\boldsymbol{v}} h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{0}$, we must have $u^i = v^i = 0$ and $r^i \neq 0$ for some $i$. □

We now prove Proposition 3.2 as follows.

*Proof of Proposition 3.2.* We compute the hessian $\nabla^2 h$ of the objective function $h$ in (15) as

$$\nabla^2 h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) \ = \ \begin{bmatrix} \boldsymbol{J}^\top \boldsymbol{J} & 2\boldsymbol{J}^\top \operatorname{diag}(\boldsymbol{u}) & -2\boldsymbol{J}^\top \operatorname{diag}(\boldsymbol{v}) \\ 2\operatorname{diag}(\boldsymbol{u})\boldsymbol{J} & 2\operatorname{diag}(\boldsymbol{r} + 2\boldsymbol{u} \odot \boldsymbol{u}) & -4\operatorname{diag}(\boldsymbol{v} \odot \boldsymbol{u}) \\ -2\operatorname{diag}(\boldsymbol{v})\boldsymbol{J} & -4\operatorname{diag}(\boldsymbol{v} \odot \boldsymbol{u}) & -2\operatorname{diag}(\boldsymbol{r} - 2\boldsymbol{v} \odot \boldsymbol{v}) \end{bmatrix}.$$

For any direction $\boldsymbol{d} = \begin{bmatrix} \boldsymbol{d}_\theta^\top & \boldsymbol{d}_{\boldsymbol{u}}^\top & \boldsymbol{d}_{\boldsymbol{v}}^\top \end{bmatrix}^\top$, the quadratic form of the Hessian $\nabla^2 h$ along this direction is given by

$$\begin{aligned}\boldsymbol{d}^\top \nabla^2 h(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v}) \boldsymbol{d} = \|\boldsymbol{J}\boldsymbol{d}_{\boldsymbol{\theta}}\|_2^2 + 4\|\boldsymbol{u} \odot \boldsymbol{d}_{\boldsymbol{u}}\|_2^2 + 4\|\boldsymbol{v} \odot \boldsymbol{d}_{\boldsymbol{v}}\|_2^2 \\ + 2\left\langle \boldsymbol{r}, \boldsymbol{d}_{\boldsymbol{u}} \odot \boldsymbol{d}_{\boldsymbol{u}} - \boldsymbol{d}_{\boldsymbol{v}} \odot \boldsymbol{d}_{\boldsymbol{v}} \right\rangle + 4\left\langle \boldsymbol{J}\boldsymbol{d}_{\boldsymbol{\theta}}, \boldsymbol{u} \odot \boldsymbol{d}_{\boldsymbol{u}} - \boldsymbol{v} \odot \boldsymbol{d}_{\boldsymbol{v}} \right\rangle - 8\left\langle \boldsymbol{u} \odot \boldsymbol{d}_{\boldsymbol{u}}, \boldsymbol{v} \odot \boldsymbol{d}_{\boldsymbol{v}} \right\rangle. \end{aligned} \tag{B.2}$$

We now consider an arbitrary critical point $(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v})$ of (15) that is not a global minimum. By Lemma B.1, there exists an $i$ such that $r^i \neq 0$ while $u^i = v^i = 0$. We divide the discussion into two cases.

- **Case 1:** $r^i > 0$. We set $\boldsymbol{d}_\theta = \boldsymbol{0}, \boldsymbol{d}_{\boldsymbol{u}} = \boldsymbol{0}$, and $\boldsymbol{d}_{\boldsymbol{v}}$ to be such that all of its entries are zero except for the $i$-th entry which is given by $\boldsymbol{d}_{\boldsymbol{v}}^i = 1$. Plugging this direction into (B.2), we obtain
$$\boldsymbol{d}^\top \nabla^2 h(\theta, \boldsymbol{u}, \boldsymbol{v}) \boldsymbol{d} = 4\underbrace{[v^i]^2}_{v^i=0}[d_{\boldsymbol{v}}^i]^2 - 2r^i \underbrace{[d_{\boldsymbol{v}}^i]^2}_{d_{\boldsymbol{v}}^i=1} \ = \ -2r^i < 0$$
- **Case 2:** $r^i < 0$. We set $\boldsymbol{d}_\theta = \boldsymbol{0}, \boldsymbol{d}_{\boldsymbol{v}} = \boldsymbol{0}$, and $\boldsymbol{d}_{\boldsymbol{u}}$ to be such that all of its entries are zero except for the $i$-th entry which is given by $\boldsymbol{d}_{\boldsymbol{u}}^i = 1$. Plugging this direction into (B.2), we obtain
$$\boldsymbol{d}^\top \nabla^2 h(\theta, \boldsymbol{u}, \boldsymbol{v}) \boldsymbol{d} = 4\underbrace{[u^i]^2}_{u^i=0}[d_{\boldsymbol{u}}^i]^2 + 2r^i \underbrace{[d_{\boldsymbol{u}}^i]^2}_{d_{\boldsymbol{u}}^i=1} \ = \ 2r^i < 0$$

In both cases above we have constructed a direction of negative curvature, hence $(\boldsymbol{\theta}, \boldsymbol{u}, \boldsymbol{v})$ is a strict saddle.

□

### B.2. Proof of Proposition 3.3

The proof is based on the following lemma which follows trivially from KKT conditions:

**Lemma B.2** (KKT condition)**.** *Given any $\boldsymbol{J}$ and $\boldsymbol{y}$, if there exists $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}}, \widehat{\boldsymbol{\nu}})$ satisfying*

$$\begin{aligned}\boldsymbol{y} &= \boldsymbol{J}\widehat{\boldsymbol{\theta}} + \widehat{\boldsymbol{s}},\\ \widehat{\boldsymbol{\theta}} &= \boldsymbol{J}^\top \widehat{\boldsymbol{\nu}}, \ \ \textit{and}\\ \widehat{\boldsymbol{\nu}} &\in \lambda \operatorname{sign}(\widehat{\boldsymbol{s}}),\end{aligned} \tag{B.3}$$

*then $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}})$ is an optimal solution to (26). In above, $\operatorname{sign}(z)$ is defined entrywise on $z$ as*

$$\textit{sign}(z) = \begin{cases} z/|z| & \textit{if } \widehat{z} \neq 0, \\ [-1, 1] & \textit{if } \widehat{z} = 0. \end{cases} \tag{B.4}$$

*Proof of Proposition 3.3.* We divide the proof into two parts.

**Global convergence.** In this part we show that $\big(\boldsymbol{\theta}_\infty(\gamma,\alpha), \boldsymbol{u}_\infty(\gamma,\alpha), \boldsymbol{v}_\infty(\gamma,\alpha)\big)$ is a global solution to (15) for any fixed $(\gamma,\alpha)$. Denote

$$\boldsymbol{r}_\infty(\gamma,\alpha) \doteq \lim_{t\to\infty} \boldsymbol{r}_t(\gamma,\alpha). \tag{B.5}$$

It follows from (18) and (22) that the limit $\boldsymbol{r}_\infty(\gamma,\alpha)$ exists and can be written as

$$\boldsymbol{r}_\infty(\gamma,\alpha) = \boldsymbol{J}\boldsymbol{\theta}_\infty(\gamma,\alpha) + \boldsymbol{u}_\infty(\gamma,\alpha)\odot\boldsymbol{u}_\infty(\gamma,\alpha) - \boldsymbol{v}_\infty(\gamma,\alpha)\odot\boldsymbol{v}_\infty(\gamma,\alpha). \tag{B.6}$$

Suppose for the purpose of obtaining a contradiction that $\big(\boldsymbol{\theta}_\infty(\gamma,\alpha), \boldsymbol{u}_\infty(\gamma,\alpha), \boldsymbol{v}_\infty(\gamma,\alpha)\big)$ is not a global solution to (15). It follows from Lemma B.1 that there exists an $i$ such that

$$u^i_\infty(\gamma,\alpha) = v^i_\infty(\gamma,\alpha) = 0, \ \text{ and } \ r^i_\infty(\gamma,\alpha) \neq 0. \tag{B.7}$$

Without loss of generality we assume that $C \doteq r^i_\infty(\gamma,\alpha) > 0$ so that $r^i_t(\gamma,\alpha) \to C$ with $t\to\infty$. For any $\epsilon \in (0, C)$, there exists a $t_0 > 0$ such that

$$C - \epsilon \le r^i_t(\gamma,\alpha) \le C + \epsilon, \ \ \forall t > t_0. \tag{B.8}$$

It follows from (B.8) and (21) that

$$\begin{aligned}\nu^i_t(\gamma,\alpha) = -\int_0^t r^i_\tau(\gamma,\alpha)d\tau = \nu^i_{t_0}(\gamma,\alpha) - \int_{t_0}^t r^i_\tau(\gamma,\alpha)d\tau \\ \in \Big(\nu^i_{t_0}(\gamma,\alpha) - (C+\epsilon)(t-t_0),\ \nu^i_{t_0}(\gamma,\alpha) - (C-\epsilon)(t-t_0)\Big), \ \ \forall t > t_0.\end{aligned} \tag{B.9}$$

Using this bound on $\nu^i_t(\gamma,\alpha)$ in (20), we obtain

$$v^i_t(\gamma,\alpha) = \gamma\exp\Big(-2\alpha\nu^i_t(\gamma,\alpha)\Big) \ge \gamma\exp\Big(-2\alpha\nu^i_{t_0}(\gamma,\alpha)\Big)\exp\Big(2\alpha(C-\epsilon)(t-t_0)\Big), \ \ \forall t > t_0. \tag{B.10}$$

Taking the limit of $t\to\infty$, we obtain $v^i_\infty(\gamma,\alpha) = \infty$ which contradicts $v^i_\infty(\gamma,\alpha) = 0$ in (B.7). Therefore, we conclude that $\big(\boldsymbol{\theta}_\infty(\gamma,\alpha), \boldsymbol{u}_\infty(\gamma,\alpha), \boldsymbol{v}_\infty(\gamma,\alpha)\big)$ is a global solution to (15).

**Implicit regularization.** In this part we prove that $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}})$ is an optimal solution to the regularized convex optimization problem in (26). Let $\boldsymbol{\nu}_\infty(\gamma,\alpha)$ be the limit of $\boldsymbol{\nu}_t(\gamma,\alpha)$ in (21) at $t\to\infty$, and let

$$\widehat{\boldsymbol{\nu}} \doteq \lim_{\gamma\to 0} \boldsymbol{\nu}_\infty(\gamma,\alpha(\gamma)), \tag{B.11}$$

with $\alpha(\gamma)$ defined in (23). We only need to show that the triplet $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{s}}, \widehat{\boldsymbol{\nu}})$ with $\widehat{\boldsymbol{\theta}}$ defined in (24), $\widehat{\boldsymbol{s}}$ defined in (25) and $\widehat{\boldsymbol{\nu}}$ defined in (B.11) satisfies the KKT conditions in (B.3).

1. Because $\big(\boldsymbol{\theta}_\infty(\gamma,\alpha), \boldsymbol{u}_\infty(\gamma,\alpha), \boldsymbol{v}_\infty(\gamma,\alpha)\big)$ is a global solution to (15), we have
$$\boldsymbol{J}\boldsymbol{\theta}_\infty(\gamma,\alpha) + \boldsymbol{u}_\infty(\gamma,\alpha)\odot\boldsymbol{u}_\infty(\gamma,\alpha) - \boldsymbol{v}_\infty(\gamma,\alpha)\odot\boldsymbol{v}_\infty(\gamma,\alpha) = \boldsymbol{y}, \ \ \forall\gamma>0, \alpha>0. \tag{B.12}$$
Taking the limit of $\gamma\to 0$ with $\alpha = \alpha(\gamma)$ and noting the assumption that all limits in (24) exist, we obtain
$$\boldsymbol{J}\widehat{\boldsymbol{\theta}} + \widehat{\boldsymbol{u}}\odot\widehat{\boldsymbol{u}} - \widehat{\boldsymbol{v}}\odot\widehat{\boldsymbol{v}} = \boldsymbol{y}. \tag{B.13}$$
Plugging in the definition of $\widehat{\boldsymbol{s}}$ in (25), we obtain
$$\boldsymbol{y} \ = \ \boldsymbol{J}\widehat{\boldsymbol{\theta}} + \widehat{\boldsymbol{s}}.$$

2. By taking the limit of the relation $\boldsymbol{\theta}_t(\gamma,\alpha) = \boldsymbol{J}^\top\boldsymbol{\nu}_t(\gamma,\alpha)$ in (20) and noting the assumptions that all relevant limits exist, we obtain
$$\widehat{\boldsymbol{\theta}} = \lim_{\gamma\to 0}\lim_{t\to\infty}\boldsymbol{\theta}_t(\gamma,\alpha(\gamma)) = \lim_{\gamma\to 0}\lim_{t\to\infty}\boldsymbol{J}^\top\boldsymbol{\nu}_t(\gamma,\alpha(\gamma)) = \boldsymbol{J}^\top\widehat{\boldsymbol{\nu}}. \tag{B.14}$$

3. Denote $\boldsymbol{s}_\infty(\gamma,\alpha) \doteq \boldsymbol{u}_\infty(\gamma,\alpha)\odot\boldsymbol{u}_\infty(\gamma,\alpha) - \boldsymbol{v}_\infty(\gamma,\alpha)\odot\boldsymbol{v}_\infty(\gamma,\alpha)$. By (20), we have
$$s^i_\infty(\gamma,\alpha) \doteq u^i_\infty(\gamma,\alpha)^2 - v^i_\infty(\gamma,\alpha)^2 = \gamma^2\exp(4\alpha\nu^i_\infty(\gamma,\alpha)) - \gamma^2\exp(-4\alpha\nu^i_\infty(\gamma,\alpha)). \tag{B.15}$$
For each entry of $\widehat{\boldsymbol{s}} = \lim_{\gamma\to 0}\boldsymbol{s}_\infty(\gamma,\alpha(\gamma))$ (recall that $\alpha(\gamma)$ is defined in (23)), we may have three cases:

   - **Case 1:** $\widehat{s}^i > 0$. From (B.15), we must have $\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma)) \to +\infty$ as $\gamma\to 0$ so that
   $$\lim_{\gamma\to 0}\exp\Big(4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = \infty, \ \text{ and } \ \lim_{\gamma\to 0}\exp\Big(-4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = 0. \tag{B.16}$$

Hence,

$$\begin{aligned} & \lim_{\gamma\to 0} \gamma^2 \exp\Big(4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = \widehat{s}^i \\ \Longrightarrow \quad & \lim_{\gamma\to 0} 2\log\gamma + 4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma)) = \log\widehat{s}^i \\ \Longrightarrow \quad & \lim_{\gamma\to 0} \nu^i_\infty(\gamma,\alpha(\gamma)) = \lim_{\gamma\to 0}\left(\frac{\log\widehat{s}^i}{4\alpha(\gamma)} - \frac{\log\gamma}{2\alpha(\gamma)}\right). \end{aligned} \tag{B.17}$$

Plugging in the relation $\alpha(\gamma) = -\frac{\log\gamma}{2\lambda}$ in (23), we have

$$\lim_{\gamma\to 0} \nu^i_\infty(\gamma,\alpha(\gamma)) = -\lim_{\gamma\to 0}\frac{\lambda\log\widehat{s}^i}{2\log\gamma} + \lambda = \lambda.$$

- **Case 2:** $\widehat{s}^i < 0$. Similar to case 1, from (B.15) we must have

$$\lim_{\gamma\to 0}\exp\Big(4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = 0, \ \text{ and } \ \lim_{\gamma\to 0}\exp\Big(-4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = \infty. \tag{B.18}$$

Hence,

$$\begin{aligned} & \lim_{\gamma\to 0} -\gamma^2\exp\Big(-4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = \widehat{s}^i \\ \Longrightarrow \quad & \lim_{\gamma\to 0} 2\log\gamma - 4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma)) = \log(-\widehat{s}^i) \\ \Longrightarrow \quad & \lim_{\gamma\to 0} \nu^i_\infty(\gamma,\alpha(\gamma)) = \lim_{\gamma\to 0}\left(-\frac{\log(-\widehat{s}^i)}{4\alpha(\gamma)} + \frac{\log\gamma}{2\alpha(\gamma)}\right). \end{aligned} \tag{B.19}$$

Plugging in the relation $\alpha(\gamma) = -\frac{\log\gamma}{2\lambda}$ in (23), we have

$$\lim_{\gamma\to 0} \nu^i_\infty(\gamma,\alpha(\gamma)) \ = \ -\lambda.$$

- **Case 3:** $\widehat{s}^i = 0$. From (B.15), we must have

$$\lim_{\gamma\to 0}\gamma^2\exp\Big(4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = 0, \ \text{ and } \ \lim_{\gamma\to 0}\gamma^2\exp\Big(-4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big) = 0. \tag{B.20}$$

Hence, for any small $\epsilon\in(0,1)$, there exists $\gamma_0 > 0$ such that for all $\gamma\in(0,\gamma_0)$, we have

$$\begin{aligned} & \gamma^2\cdot\max\Big\{\exp\Big(4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big), \exp\Big(-4\alpha(\gamma)\nu^i_\infty(\gamma,\alpha(\gamma))\Big)\Big\} < \epsilon \\ \Longrightarrow \quad & 2\log\gamma + 4\alpha(\gamma)\cdot|\nu^i_\infty(\gamma,\alpha(\gamma))| < \log\epsilon \\ \Longrightarrow \quad & |\nu^i_\infty(\gamma,\alpha(\gamma))| < \frac{\log\epsilon}{4\alpha(\gamma)} - \frac{\log\gamma}{2\alpha(\gamma)}. \end{aligned} \tag{B.21}$$

Now, plugging $\alpha(\gamma) = -\frac{\log\gamma}{2\lambda}$ in, we have

$$|\nu^i_\infty(\gamma,\alpha(\gamma)))| < -\frac{\lambda\log\epsilon}{2\log\gamma} + \lambda \ < \ \lambda.$$

Therefore, we have

$$\lim_{\gamma\to 0}\big|\nu^i_\infty(\gamma,\alpha(\gamma))\big| \ < \ \lambda.$$

Synthesizing all the above three cases, we obtain:

$$\widehat{\boldsymbol{\nu}} \ \in \ \lambda\mathrm{sign}(\widehat{\boldsymbol{s}}).$$

□

### B.3. Proof of Proposition 3.5

We begin with introducing the null space property that is widely used for providing necessary and sufficient conditions for exact recovery of sparse signals in compressive sensing.

**Definition B.3** ((Cohen et al., 2009)). We say a matrix $\boldsymbol{A}\in\mathbb{R}^{m\times n}$ satisfies the null space property with constant $\rho\in(0,1)$ relative to $S\in[n]$ if

$$\|\boldsymbol{v}_S\|_1 \le \rho\|\boldsymbol{v}_{S^c}\|_1 \text{ for all } \boldsymbol{v}\in\ker\boldsymbol{A},$$

where $\ker\boldsymbol{A}$ is the null space of $\boldsymbol{A}$.

With Definition B.3, we prove Proposition 3.5 by using the following two lemmas. The first lemma establishes correct recovery of $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ from (26) under the null space property.

**Lemma B.4.** *Given matrix $\boldsymbol{J}$ and a matrix $\boldsymbol{A}$ that annihilates $\boldsymbol{J}$ on the left (i.e. such that $\boldsymbol{A}\boldsymbol{J} = 0$). If $\boldsymbol{A}$ satisfies the stable null space property with constant $\rho \in (0,1)$ relative to the support of $\boldsymbol{s}_\star$, then the solution to (26) is $(\boldsymbol{\theta}_\star, \boldsymbol{s}_\star)$ for any $\lambda > \lambda_0$ where $\lambda_0$ is a scalar that depends only on $(\boldsymbol{J}, \boldsymbol{\theta}_\star, \rho)$.*

The second lemma shows that the null space property is satisfied under the incoherent condition in (28).

**Lemma B.5.** *Given matrix $\boldsymbol{J}$ and a matrix $\boldsymbol{A}$ that annihilates $\boldsymbol{J}$ on the left, if*

$$k^2 r \leq \frac{N}{\mu(\boldsymbol{J})} \left(\frac{\rho}{\rho+1}\right)^2, \tag{B.22}$$

*then $\boldsymbol{A}$ satisfies null space property with constant $\rho$ relative to any $S$ that satisfies $|S| = k$.*

*Proof of Proposition 3.5.* Assume that the condition in (28) is satisfied. Then there exists a $\rho \in (0,1)$ such that the condition in (B.22) holds. Hence, $\boldsymbol{A}$ satisfies null space property with constant $\rho$ relative to any $S$ that satisfies $|S| = k$. Since $\boldsymbol{s}_\star$ is $k$-sparse, we have that $\boldsymbol{A}$ satisfies null space property with constant $\rho$ relative to the support of $\boldsymbol{s}_\star$. Then the conclusion of Proposition 3.5 follows from applying Lemma B.4. Finally, from Lemma B.4 we have that $\lambda_0$ is a function of $(\boldsymbol{J}, \boldsymbol{\theta}_\star, \rho)$, wherein $\rho$ is determined by $\boldsymbol{A}$ (hence $\boldsymbol{J}$) and the associated sparsity $k$. Hence $\lambda_0$ can be determined with a given $(\boldsymbol{J}, \boldsymbol{\theta}_\star, k)$. □

In the rest of this section we prove Lemma B.4 and Lemma B.5.

**Proof of Lemma B.4.** We first introduce the following result on a useful property of the stable null space property.

**Theorem B.6** (Useful property of stable null space property)**.** *Suppose a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ satisfies the null space property with constant $\rho \in (0,1)$ relative to $S \in [n]$. Then for every vector $\boldsymbol{x}$ supported on $S$, we have*

$$\|\boldsymbol{z} - \boldsymbol{x}\|_1 \leq \frac{1+\rho}{1-\rho}(\|\boldsymbol{z}\|_1 - \|\boldsymbol{x}\|_1)$$

*for any $\boldsymbol{z}$ with $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$.*

*Proof of Theorem B.6.* Since $\boldsymbol{A}(\boldsymbol{z} - \boldsymbol{x}) = \boldsymbol{0}$, i.e., $\boldsymbol{z} - \boldsymbol{x} \in \ker \boldsymbol{A}$, the null space property of $\boldsymbol{A}$ implies

$$\|(\boldsymbol{z} - \boldsymbol{x})_{S^c}\|_1 \leq \rho \|(\boldsymbol{z} - \boldsymbol{x})_{S^c}\|_1,$$

which further gives that

$$\|\boldsymbol{z} - \boldsymbol{x}\|_1 \leq (1+\rho)\|(\boldsymbol{z} - \boldsymbol{x})_{S^c}\|_1.$$

We now use these properties to prove the main result as

$$\begin{aligned}
\|\boldsymbol{z}\|_1 &= \|\boldsymbol{z}_S\|_1 + \|\boldsymbol{z}_{S^c}\|_1 = \|(\boldsymbol{z} - \boldsymbol{x} + \boldsymbol{x})_S\|_1 + \|\boldsymbol{z}_{S^c}\|_1 \\
&\geq \|\boldsymbol{x}\|_1 - \|(\boldsymbol{z} - \boldsymbol{x})_S\|_1 + \|(\boldsymbol{z} - \boldsymbol{x})_{S^c}\|_1 \\
&\geq \|\boldsymbol{x}\|_1 + (1-\rho)\|(\boldsymbol{z} - \boldsymbol{x})_{S^c}\|_1 \\
&\geq \|\boldsymbol{x}\|_1 + \frac{1+\rho}{1-\rho}\|\boldsymbol{z} - \boldsymbol{x}\|_1,
\end{aligned}$$

where the first inequality follows because $\boldsymbol{x}$ is only supported on $S$. □

We are now ready to prove Lemma B.4.

*Proof of Lemma B.4.* Let $\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the compact SVD of $\boldsymbol{J}$ and $\boldsymbol{V}_\perp$ be an orthonormal basis that complements $\boldsymbol{V}$. Then, the constraint in (26) is equivalent to

$$\boldsymbol{A}\boldsymbol{y} = \boldsymbol{A}\boldsymbol{s}, \quad \boldsymbol{\theta} = \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{y} - \boldsymbol{s}) + \boldsymbol{V}_\perp \boldsymbol{h}.$$

Thus, the problem (26) is equivalent to

$$\begin{aligned}
&\min_{\boldsymbol{s},\boldsymbol{h}} \frac{1}{2}\|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{y} - \boldsymbol{s}) + \boldsymbol{V}_\perp \boldsymbol{h}\|_2^2 + \lambda\|\boldsymbol{s}\|_1 \\
&\text{s.t.} \quad \boldsymbol{A}\boldsymbol{y} = \boldsymbol{A}\boldsymbol{s},
\end{aligned} \tag{B.23}$$

which is further equivalent to

$$\begin{aligned} \min_{\boldsymbol{s}} g(\boldsymbol{s}) &:= \frac{1}{2}\|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{y}-\boldsymbol{s})\|_2^2 + \lambda\|\boldsymbol{s}\|_1 \\ \text{s.t.} \quad & \boldsymbol{A}\boldsymbol{s}_\star = \boldsymbol{A}\boldsymbol{s}. \end{aligned} \tag{B.24}$$

Assume $\boldsymbol{A}$ satisfies the stable null space property with constant $\rho \in (0,1)$ relative to the support of $\boldsymbol{s}_\star$. Now for any $\boldsymbol{s}$ with $\boldsymbol{A}\boldsymbol{s}_\star = \boldsymbol{A}\boldsymbol{s}$, by Theorem B.6, we have

$$\|\boldsymbol{s}\|_1 - \|\boldsymbol{s}_\star\|_1 \geq \frac{1-\rho}{1+\rho}\|\boldsymbol{s}-\boldsymbol{s}_\star\|_1,$$

which ensures $\boldsymbol{s} = \boldsymbol{s}_\star$ if we only minimize $\|\boldsymbol{s}\|_1$. The first term in (B.24) can be written as

$$\|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{y}-\boldsymbol{s})\|_2^2 = \|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{s}_\star - \boldsymbol{s} + \boldsymbol{J}\boldsymbol{\theta}_\star)\|_2^2,$$

where

$$\boldsymbol{\theta}_\star \;=\; \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{y}-\boldsymbol{s}_\star).$$

This together with the previous equation gives

$$\begin{aligned} & g(\boldsymbol{s}) - g(\boldsymbol{s}_\star) \\ & \geq \lambda\frac{1-\rho}{1+\rho}\|\boldsymbol{s}-\boldsymbol{s}_\star\|_1 + \|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{s}_\star - \boldsymbol{s} + \boldsymbol{J}\boldsymbol{\theta}_\star)\|_2^2 - \|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top\boldsymbol{J}\boldsymbol{\theta}_\star\|_2^2 \\ & = \lambda\frac{1-\rho}{1+\rho}\|\boldsymbol{s}-\boldsymbol{s}_\star\|_1 + \|\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^\top(\boldsymbol{s}_\star - \boldsymbol{s})\|_2^2 + 2\langle \boldsymbol{s}_\star - \boldsymbol{s}, \boldsymbol{U}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^\top\theta_\star\rangle \\ & \geq \lambda\frac{1-\rho}{1+\rho}\|\boldsymbol{s}-\boldsymbol{s}_\star\|_1 - 2\|\boldsymbol{U}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^\top\theta_\star\|_\infty\|\boldsymbol{s}-\boldsymbol{s}_\star\|_1 \\ & = \left(\lambda\frac{1-\rho}{1+\rho} - 2\|\boldsymbol{U}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^\top\theta_\star\|_\infty\right)\|\boldsymbol{s}-\boldsymbol{s}_\star\|_1. \end{aligned}$$

Thus, if $\lambda > \lambda_0$ with

$$\lambda_0 = 2\frac{1+\rho}{1-\rho}\|\boldsymbol{U}\boldsymbol{\Sigma}^{-1}\boldsymbol{V}^\top\boldsymbol{\theta}_\star\|_\infty, \tag{B.25}$$

we have $g(\boldsymbol{s}) - g(\boldsymbol{s}_\star) > 0$ whenever $\boldsymbol{s} \neq \boldsymbol{s}_\star$.

□

**Proof of Lemma B.5.**

*Proof.* Let $\boldsymbol{J} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ be the compact SVD of $\boldsymbol{J}$. From (B.22) we have

$$(\rho+1)k\sqrt{\frac{r}{N}\mu(\boldsymbol{J})} \leq \rho. \tag{B.26}$$

Let $S \subseteq [N]$ with $|S| = k$ and $\boldsymbol{a} \in \mathbb{R}^r$ be an arbitrary vector. We have

$$\|[\boldsymbol{U}\boldsymbol{a}]_S\|_1 = \sum_{i\in S}|\boldsymbol{e}_i^\top\boldsymbol{U}\boldsymbol{a}| = \sum_{i\in S}|\langle \boldsymbol{U}^\top\boldsymbol{e}_i, \boldsymbol{a}\rangle| \leq k\|\boldsymbol{a}\|_2 \cdot \max_{i\in S}\|\boldsymbol{U}^\top\boldsymbol{e}_i\|_2 \leq k\|\boldsymbol{a}\|_2\sqrt{\frac{r}{N}\mu(\boldsymbol{J})}, \tag{B.27}$$

where the last inequality is obtained from Definition 3.4. In addition, we have

$$\|\boldsymbol{U}\boldsymbol{a}\|_1 \geq \|\boldsymbol{U}\boldsymbol{a}\|_2 = \|\boldsymbol{a}\|_2. \tag{B.28}$$

Combining (B.26), (B.27) and (B.28), we get

$$(\rho+1)\|[\boldsymbol{U}\boldsymbol{a}]_S\|_1 \leq (\rho+1)k\|\boldsymbol{a}\|_2\sqrt{\frac{r}{N}\mu(\boldsymbol{J})} \leq \rho\|\boldsymbol{a}\|_2 \leq \rho\|\boldsymbol{U}\boldsymbol{a}\|_1, \tag{B.29}$$

hence,

$$\|[\boldsymbol{U}\boldsymbol{a}]_S\|_1 \leq \rho\|[\boldsymbol{U}\boldsymbol{a}]_{S^c}\|_1. \tag{B.30}$$

Noting that $\{\boldsymbol{U}\boldsymbol{a}|\boldsymbol{a} \in \mathbb{R}^r\} = \ker \boldsymbol{A}$, this finishes the proof by Definition B.3. □