ITERATIVELY LEARNING FROM THE BEST

Anonymous authors
Paper under double-blind review

ABSTRACT

We study a simple generic framework to address the issue of bad training data; both bad labels in supervised problems, and bad samples in unsupervised ones. Our approach starts by fitting a model to the whole training dataset, but then iteratively improves it by alternating between *(a)* revisiting the training data to select samples with lowest *current* loss, and *(b)* re-training the model on *only* these selected samples. It can be applied to any existing model training setting which provides a loss measure for samples, and a way to refit on new ones. We show the merit of this approach in both theory and practice. We first prove statistical consistency, and linear convergence to the ground truth and global optimum, for two simpler model settings: mixed linear regression, and gaussian mixture models. We then demonstrate its success empirically in *(a)* saving the accuracy of existing deep image classifiers when there are errors in the labels of training images, and *(b)* improving the quality of samples generated by existing DC-GAN models, when it is given training data that contains a fraction of the images from a different and unintended dataset. The experimental results show significant improvement over the baseline methods that ignore the existence of bad labels/samples.

1 INTRODUCTION

This paper is **motivated** by the problem that training large machine learning models requires well-curated training sets with lots of samples. This is an issue in both supervised and unsupervised settings. For *supervised* problems (e.g. multi-label classification) we need very many properly labeled examples; generating these requires a lot of human effort. Errors in labels can significantly affect the accuracy of naive training. For example, a faulty CIFAR-10 dataset with 20% of automobile images mis-labeled as "airplane" (and so on for the other classes) leads to the accuracy of a neural architecture like WRN of Zagoruyko & Komodakis (2016) to go from over 90% to about 70%. Even in *unsupervised* problems, where there are no labels, spurious samples are a nuisance. Consider for example the task of training a generative adversarial network (GAN) Goodfellow et al. (2014) to produce realistic images of faces. While this has been shown to work well using a dataset of face images [1], like Celeb-A Liu et al. (2015), it degrades quickly if the training dataset is corrupted with non-face images. Our experiments illustrate these issues.

Main idea We study a simple framework that addresses the problem of spurious data in both supervised and unsupervised settings. Note that if we knew which samples were clean, we could learn a model; conversely, if we had a good model, we could find the clean samples. But we do not have either; instead we have a chicken-and-egg problem: finding the best subset of samples needs a model, but finding a model needs a subset of samples. In this paper, we take the natural approach to this problem by *(0)* fitting an initial model on all samples, and then alternating between *(1)* finding the best subset of samples given a model, and *(2)* fitting a model given a subset. [2] Our approach is most related to the works of Bhatia et al. (2015) and Vainsencher et al. (2017), and we contrast to these in the related work section.

[1] https://github.com/carpedm20/DCGAN-tensorflow

[2] Our framework sounds initially similar to EM-style algorithms like k-means. Note however that EM needs to postulate a model for all the data points, while we search over a subset and do not worry about the loss on corrupted points. We are alternating between a simple search over subsets and a fitting problem on only the *selected* subset; this is not an instance of EM.

Our approach can be applied to any existing model that provides (a) confidence scores for samples, and (b) a way to train the model for a set of samples. Indeed in our experiments we use existing models and training code, but with iterative (re)selections of samples as above. In supervised settings, any model that maps features to responses typically associates a confidence to every sample, e.g., in linear regression, given a model θ, the confidence in a sample $(\mathbf{x}_i, y_i)$ increases as its squared error $(y_i - \mathbf{x}_i^\top \theta)^2$ decreases. For a neural network classifier, on the other hand, the confidence can be measured by the cross entropy loss between the output probability distribution and the true distribution. In unsupervised settings, the confidence of a sample is given by its "likelihood" under the model. If we are lucky, e.g., in Gaussian mixture models, this likelihood is explicit and can be evaluated. However, if there is no explicit likelihood, we need to rely on a surrogate. E.g., in our GAN setting below, we rely on the output of the discriminator as a confidence measure. We also note an important **caveat:** this paper is not focused on (and may not be effective for) the settings of adversarial examples, or of gross corruptions of features. Rather, our focus is to address the issue of bad labeling or curation of data, as motivated above.

Our Results and Paper Outline

We show that our framework (outlined in Section 3) can address the spurious sample problem, both theoretically and empirically and for both supervised and unsupervised problems:

(1) Mixed linear regression (Section 4) and Gaussian mixture model (Section 5) For each of these mixture problem settings, under the standard statistical assumptions, we establish both statistical consistency and linear convergence of our iteratively updated model parameters.

(2) Neural network classifiers (Section 6) : We investigate image classification in CIFAR-100, CIFAR-10 and MNIST datasets (each with a different and appropriate *existing* neural architecture) in the setting where there are *systematic* errors in the labels – i.e. every bad example in one label class comes from the same other label class. We show that our framework can significantly improve the classification accuracy over the baseline method that ignores the existence of bad samples.

(3) Deep generative models (Section 7): We show both quantitatively and qualitatively the improvement of using our algorithm for image generation when the dataset consists of two types of images. Specifically, investigate training a GAN to *(a)* generate face images as in the "intended" Celeb-A dataset, but when the given training data has a fraction of the samples being the "spurious" CIFAR-10 images, and *(b)* generate intended MNIST digits when training data contains a fraction of samples from the spurious Fashion-MNIST dataset. A *crucial innovation* here is that the output of the discriminator network in the GAN can be used as a confidence metric for the training samples.

In Section 8, we discuss the positives and negatives of our results, underlying intuition and also some future directions. All of our experiments were done on clearly identified publicly available neural network architectures, datasets, and training software. We also clearly describe our error and noise effects; we expect them to be easily reproducible.

2 Related Work

Alternating minimization for robustness Two works are most related to ours. Bhatia et al. (2015) proposes an iterative hard thresholding algorithm for adversarial noise in linear regression responses; they establish statistical guarantees and linear convergence under this harder noise model. We focus on multiple mixed linear regression, a different setting that results in an easier noise model, and we can handle a larger fraction ($\frac{1}{65}$ vs $\frac{1}{2} - \epsilon$) of bad samples. In addition we also prove these for Gaussian mixture models, which is not studied in their work. Vainsencher et al. (2017) proposes what is essentially a soft version of our method, and prove local convergence and generalization. However they do not have any initialization, and hence no global or consistency guarantees. Neither of these empirically explore the overall approach for more complex / neural network models.

Robust regression Recent work on robust regression consider strong robustness in terms of both the inputs and the outputs and provide guarantees for constant corruption ratio Diakonikolas et al.

(2018); Prasad et al. (2018); Klivans et al. (2018). Chen et al. (2013); Liu et al. (2018) study the high dimensional setting and provide algorithms for recovering the true parameters. These algorithms require much more computation, compared with methods for dealing with noisy outputs, e.g., Bhatia et al. (2015). Another type of robustness considers heteroscedastic noise for every sample. From the learning perspective, Anava & Mannor (2016); Chaudhuri et al. (2017) both require strong constraints on the noise model which largely reduces the degree of freedom. Anava & Mannor (2016) considers the online learning setting, while Chaudhuri et al. (2017) considers the active learning setting.

Mixed linear regression Alternating minimization type algorithms are used for mixed linear regression with convergence guarantee Yi et al. (2014). Chen et al. (2014) provides a convex formulation for the problem. (Balakrishnan et al., 2017) shows expectation-maximization (EM) algorithm is provably effective for a set of tasks including the basic setting of mixed linear regression and Gaussian mixture model. (Sedghi et al., 2016) extends to the multiple component setting, where provable tensor decomposition methods are used. More recently, Li & Liang (2018) gives a more general result for learning mixtures of linear regression. On the other hand, (Ray et al., 2018) studies the problem of finding a single component in mixed linear regression problem using side information.

Noisy labels Classification tasks with noisy labels are also of wide interest. Frénay et al. (2014) gives an overview of the related methods. Theoretical guarantee for noisy binary classification has been studied under different settings Scott et al. (2013); Natarajan et al. (2013); Menon et al. (2016). More recently, noisy label problems have been studied for deep neural network based models. Reed et al. (2014) and Malach & Shalev-Shwartz (2017) develop the idea of bootstrapping and query-by-committee into neural network based models. On the other hand, Khetan et al. (2017) and Zhang & Sabuncu (2018) provide new losses for training under the noise. Sukhbaatar & Fergus (2014) adds a noise layer into the training process, while Ren et al. (2018) provides a meta-algorithm for learning the weights of all samples by referencing to a clean subset of validation data during training.

Others (Bora et al., 2018) studies the problem of noisy images for GANs. In their setting, all images are of low quality under some measurements, while in our problem, we consider the image dataset consists of (possibly more than) two types of images, which is different. There are also studies that consider noisy samples more generally. The classical RANSAC method Fischler & Bolles (1981) provides a paradigm for fitting a model to experimental data, and can be potentially used for the noisy data setting. However, the algorithm requires multiple times of random sampling to find a valid consensus score, which is conceptually different from our idea of doing iterative filtering. EM algorithm Moon (1996) can also be used for detecting noisy samples, e.g., in the setting of mixture models Balakrishnan et al. (2017). However, our algorithm does not need to know the likelihood for the bad samples for the iterative updates.

3 ITERATIVELY LEARNING FROM THE BEST (ILFB)

We now describe the setup and algorithm, in a unified way that covers both supervised and unsupervised settings. We will then specialize to the many specific problems we study. We are provided with *samples* $s_1, \cdots, s_n$, using which we want to train a *model* parameterized by $\theta \in \mathbb{R}^d$. We are also provided with a loss function; let $f_\theta(s)$ denote the loss of a sample s when the parameters are θ. Let $\tau \in [0, 1]$ be the fraction of data we want to fit. With this setup, the idea of learning from the best can be viewed as trying to solve

$$\min_\theta \min_{S:|S|=\tau n} \sum_{i \in S} f_\theta(s_i).$$

We do this by alternating between finding S and finding θ, as described below in Alg. 1.

Algorithm 1 ILFB Update

1: **input:** samples $\{s_i\}_{i=1}^n$, fraction τ, max. iteration number T
2: **initialize:** set $t \leftarrow 0$ and initially fit all samples $\theta_0 \leftarrow \arg\min_\theta \sum_{i\in[n]} f_\theta(s_i)$,
3: **while** $t < T$ **do**
4: calculate current losses $f_{\theta_t}(s_i)$ for all samples s_i, $i \in [n]$
5: find new best set of samples

$$S_t \leftarrow \arg \min_{S:|S|=\tau n} \sum_{i\in S} f_{\theta_t}(s_i)$$

by sorting them according to losses $f_{\theta_t}(s_i)$ and choosing the ones with smallest losses.
6: fit a model for this new set

$$\theta_{t+1} \leftarrow \arg \min_\theta \sum_{i\in S_t} f_\theta(s_i)$$

7: $t \leftarrow t + 1$
8: **return:** θ_T

Comments Steps 4, 5 and 6 above can be done approximately, when exact solutions are unavailable. For example, in our experiments involving deep neural networks (both the classifiers and the generative models), for step 6 we do not fit a model exactly, but rather do training using stochastic gradient descent. Also, for our experiments on training GANs on the best subset, we do not have an explicit loss; we instead use the loss at the output of the discriminator as a surrogate (more details on that below). Similarly, the initialization step above can possibly be replaced with some problem-setting dependent alternative, if one is available.

In the following, we describe how this directly specializes to each of our four settings.

4 ILFB for Mixed Linear Regression

We first describe ILFB as specialized to the case when the θ represents the parameters of a linear regression, and then describe the statistical setting of mixed linear regression where we provide rigorous statistical guarantees on its performance.

Algorithm The samples are $s_i = (\mathbf{x}_i, y_i)$ with features $\mathbf{x}_i \in \mathbb{R}^d$ and responses $y_i \in \mathbb{R}$, and $\theta \in \mathbb{R}^d$ are the parameters of the linear model. The loss is $f_\theta(\mathbf{x}, y) = (y - \mathbf{x}^\top\theta)^2$. The algorithm thus initializes θ_0 by doing an ordinary least squares (OLS) on all the samples, and then alternates between finding the set of τn samples with the lowest squared error for the current θ, and then finding the new θ by doing an OLS on this set.

Statistical setting In the standard and widely studied mixed linear regression problem there are n samples $(\mathbf{x}_i, y_i)$, with features $\mathbf{x}_i \sim \mathcal{N}(0, I_d)$ being standard gaussians in d dimensions. The full set of samples $S = [n]$ can be splitted into a set of "good" samples S_{good}, and m sets of "bad" samples $\cup_{j\in[m]} S^j_{\text{bad}}$. The response variables y_i are given by:

$$y_i = \begin{cases} x_i^\top \theta^\star + \epsilon_i, & \text{if } i \in S_{\text{good}} \\ x_i^\top \theta^{(j)} + \epsilon_i, & \text{if } i \in S^j_{\text{bad}} \end{cases} \tag{1}$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ is the additive noise, and $\theta^\star$ as well as all the $\theta^{(j)}$ are assumed to be unit norm vectors. We analyze our algorithm in the fresh sample setting, where iteration t finds θ_{t+1} from θ_t using a new set of samples. This too is standard in the statistical analysis of mixed linear regression, and corresponds to the case of using mini-batches of samples to learn a model.

We now prove two results; the first shows that the iterates converge linearly to the true θ^* once θ_t is closer to θ^* than any of the bad $\theta^{(j)}$. The second shows that, with some additional conditions on the θ^* and $\theta^{(j)}$'s, the initialization condition ensures a θ_0 that satisfies this condition.

Theorem 1 (local linear convergence). *For the mixed linear regression setting as above, suppose that some iterate θ_t satisfies $\|\theta_t - \theta^\star\| < \min_{j\in[m]} \|\theta_t - \theta^{(j)}\|$, and the τ in Algorithm 1 is less than the true ratio of good samples. Then, for any $C < 1$, the next iterate θ_{t+1} of the algorithm satisfies*

$$\|\theta_{t+1} - \theta^\star\| \leq C\left(\|\theta_t - \theta^\star\| + \sigma\right)$$

with high probability (i.e., with probability $1 - n^{-c_0}$), provided the number of samples $n \geq c_1 \frac{d}{C^2\tau^4}$, where c_1 is a constant.

Theorem 1 establishes a linear dependence of sample complexity on the dimension d, and a fourth-order dependence on the fraction τ of good points. It would be interesting to see if this fourth-order dependence can be improved.

Theorem 2 (initialization). *Given the model described in (1), assume $\sum_{j\in[m]} \|\theta^\star - \theta^{(j)}\||S^j_{\text{bad}}| \leq \frac{n}{\kappa}\min_{j\in[m]} \|\theta^\star - \theta^{(j)}\|$ for some $\kappa > 2$, and define $d_m := \min_{j\in[m]} \|\theta^\star - \theta^{(j)}\|$. We have:*

$$\|\theta^\star - \theta_0\| \leq \frac{1}{2} d_m$$

with high probability, where $\theta_0 = \theta_{\text{MLE}}$, for $n \geq \max\{\frac{g(\kappa)^2 d\sigma^2}{d_m^2}, \frac{cg(\kappa)m\log m}{d_m}\}$, $g(\kappa) = \frac{\kappa(\kappa+2)}{\kappa-2}$.

The assumption in Theorem 2 requires the total sum of distance of the parameters for bad samples should not be too large. For a fixed d_m, if the distance for some $\theta^{(j)}$ becomes larger, it may become harder for the OLS estimator to be in the local region close to $\theta^\star$.

The proofs for the theorems are in Appendix C.1 and C.2, and synthetic experiments verifying the performance of ILFB is in Appendix C.3.

5 ILFB FOR GAUSSIAN MIXTURE MODEL

In this section, we describe ILFB applied to the setting of Gaussian mixture model, and provide statistical guarantees on its performance.

Algorithm The samples $s_i = \mathbf{x}_i$ are points in $\mathbb{R}^d$, $\theta \in \mathbb{R}^d$ is the mean we want to estimate, and $f_\theta(\mathbf{x})$ is proportional to the negative log-likelihood, i.e., $f_\theta(\mathbf{x}) = \|\mathbf{x} - \theta\|^2$. The algorithm thus initializes the mean by finding the average of all n points, and then iteratively finds the set of τn closest points closest to the current mean, and updating the mean to be the average of this set.

Statistical setting Given a dataset $\mathcal{D} = \{\mathbf{x}_i \mid i \in S = \{1, \cdots, n\}\}$, where S can be splitted into disjoint sets $S = S_{\text{good}} \cup S^1_{\text{bad}} \cup \cdots \cup S^m_{\text{bad}}$, the samples follows:

$$\begin{aligned} \mathbf{x}_i &\sim \mathcal{N}\left(\theta^\star, \sigma^2 I\right), \quad \text{if } i \in S_{\text{good}} \\ \mathbf{x}_i &\sim \mathcal{N}\left(\theta^{(j)}, \sigma^2 I\right), \quad \text{if } i \in S^j_{\text{bad}} \end{aligned} \tag{2}$$

We now prove two results: the first shows that iterates converge linearly under the fresh sample setting in a local region, while the second shows with some additional conditions, that the initialization of ILFB falls into this local region.

Theorem 3 (local linear convergence). *We run Alg. 1 on the given dataset $\mathcal{D}$ generated following (2), if $\|\theta_0 - \theta^\star\| \leq \frac{3}{4}\min_{j\in[m]} \|\theta_0 - \theta^{(j)}\|$, then, with high probability (i.e., with probability $1 - n^{-c_0}$),*

$$\|\theta_{t+1} - \theta^\star\| \leq c_1\|\theta_t - \theta^\star\| + c_2\sqrt{d}\sigma,$$

where $c_1 \in (0, 1), c_2$ are constant.

Here, the $3/4$ factor in the assumption is chosen for convenience of the analysis. Essentially, what we need is this value being less than one. Intuitively, given this initialization condition, the samples selected by the initialization parameter contains more than half good samples, and we can show that the algorithm is guaranteed to converge. Notice that there is no dependency between n and d. In the extreme case where $\sigma = 0$, we only need n to be a constant value: from any good initialization point, the algorithm will converge to $\theta^\star$ in one step.

Table 1: **Classification with systematic label error:** Performance for MNIST, CIFAR-10, CIFAR-100 datasets as the ratio of good samples varies from 60% to 90%. Here **Baseline** : Naive training using all the samples; **ILFB** : Our iterative update algorithm; **Oracle** : Training with all good samples. We see significant improvement of **ILFB** over **Baseline** for all the settings.

dataset	MNIST			CIFAR-10			CIFAR-100		
$\frac{\text{\# good}}{\text{\# total}}$	**Baseline**	**ILFB**	**Oracle**	**Baseline**	**ILFB**	**Oracle**	**Baseline**	**ILFB**	**Oracle**
60%	70.26	90.00	94.30	64.63	75.16	91.26	58.28	65.25	80.39
70%	85.95	92.09	94.50	71.05	86.09	92.25	66.75	73.83	81.65
80%	92.62	94.30	94.61	78.84	89.14	92.85	72.68	78.70	82.50
90%	93.88	94.41	94.92	84.12	91.03	93.47	78.12	81.10	83.41

Theorem 4 (initialization). *Given a dataset $\mathcal{D}$ generated following (2), define $d_m := \min_{j\in[m]} \|\theta^\star - \theta^{(j)}\|$, and assume $\left\|\sum_{j\in[m]}(\theta^\star - \theta^{(j)})|S^j_{\text{bad}}|\right\| \leq \alpha n d_m$. Consider taking the overall mean as the initialization θ_0, then, if $n \geq \frac{cd\sigma^2}{(\frac{3}{7}-\alpha)^2 d_m^2}$, we have $\|\theta_0 - \theta^\star\| \leq \frac{3}{4}\min_{j\in[m]}\left\|\theta_0 - \theta^{(j)}\right\|$ with high probability.*

The proofs for the theorems are in Appendix D.1 and D.2, and synthetic experiments verifying the performance of ILFB is in Appendix D.3.

6 ILFB for Deep Image Classification with Systematic Training Label Error

Experimental setting We consider *systematic errors in labels*, i.e. the setting where all the samples that actually come from a class "a" are given the *same* bad label "b", a setting that is less benign than one with more randomness in the errors. We investigated the ability of our framework to account for these errors for the following dataset – neural architecture pairs:
(a) MNIST (LeCun et al., 1998) dataset with a standard 2-layer CNN [3]
(b) CIFAR-10 (Krizhevsky & Hinton, 2009) with a 16-layer WideResNet [4] (Zagoruyko & Komodakis, 2016)
(c) CIFAR-100 [5] with a 28-layer WideResNet [6]
In each of these cases, the neural network has excellent performance when there are no label errors in training; this degrades significantly as the fraction of spurious samples increases.

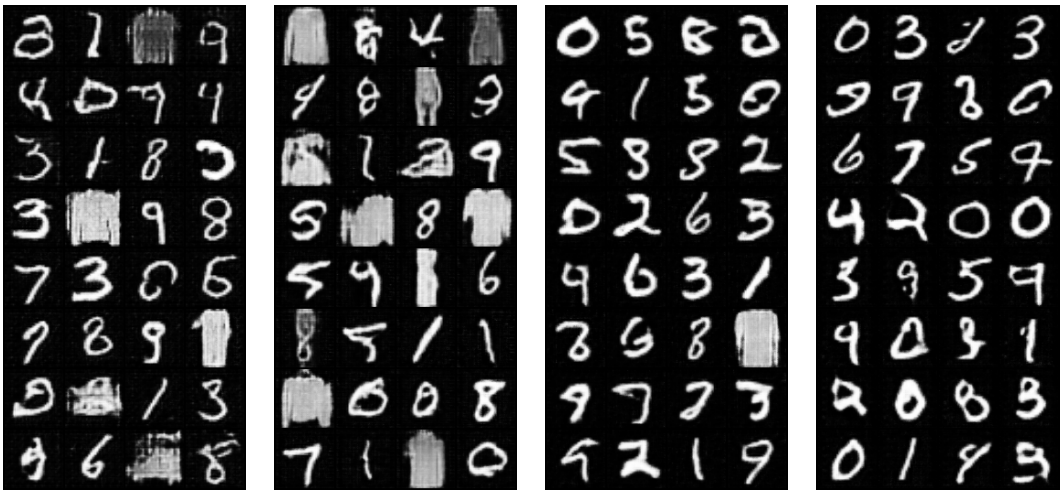
Algorithm In each case, the algorithm involves first training the NN on all the samples, and then iteratively alternating between *(a)* picking the fraction τ of samples in each label class with the lowest cross-entropy loss under the current θ, and *(b)* retraining the model from scratch with these picked samples, to get a new θ. Training was done on MXNet. Mapping back to the development in Section 3, here θ represents the weights of the neural network, and $f_\theta(s)$ is the standard cross-entropy loss used in training.

Further details When training on all the samples, we run simple stochastic gradient descent algorithm with initial learning rate 0.5/0.1/0.1 and batch size 1000/256/64 for MNIST/CIFAR-10/CIFAR-100, respectively. The learning rate is divided by 5 at the 50-th epoch and each experiment runs for 80 epochs. For each iteration of update using our ILFB algorithm, since less samples are used at each epoch, we adjust the total epoch number accordingly, so that the training for every iteration uses the same number of SGD updates. We set τ to be 5% less than the true ratio of "good" samples.

[3] follow the implementation in https://mxnet.incubator.apache.org/tutorials/python/mnist.html
[4] cifar_wideresnet16_10 from the online model zoo: https://gluon-cv.mxnet.io/api/model_zoo.html
[5] We use the coarse labels as the classification target
[6] WideResNet: depth 28, width factor 10, dropout rate 0.3

Table 2: **Generative models from mixed training data:** A **quantitative** measure of the efficacy of our approach is to find how many of the good training samples the final discriminator can identify; this is shown here for the three different "good"/"bad" dataset pairs. For each pair, the fraction of "good" samples is 90%, 80% or 70%. The table depicts the ratio of the good samples in the training data that are *recovered* by the discriminator when it is run on the training samples. The higher this fraction, the more effective the generator. For MNIST-Fashion and CelebA-CIFAR10, our approach shows significant improvements with iteration count. For CIFAR10-CelebA dataset, the error is extremely simple to be corrected, likely because faces are easier to discriminate against when compared to natural images.

	MNIST(good)-Fashion(bad)			CIFAR10(good)-CelebA(bad)			CelebA(good)-CIFAR10(bad)		
orig	90%	80%	70%	90%	80%	70%	90%	80%	70%
ILFB iter-1	91.90%	76.84%	77.77%	100.0%	99.99%	98.67%	97.12%	81.34%	75.57%
ILFB iter-2	96.05%	91.95%	79.12%	100.0%	99.85%	99.10%	97.33%	88.11%	76.45%
ILFB iter-3	99.15%	96.14%	85.66%	100.0%	99.91%	99.69%	97.43%	89.48%	86.63%
ILFB iter-4	100.0%	99.67%	91.51%	100.0%	99.96%	99.75%	97.53%	92.89%	82.15%
ILFB iter-5	100.0%	100.0%	97.00%	100.0%	99.99%	99.95%	98.14%	92.94%	94.02%



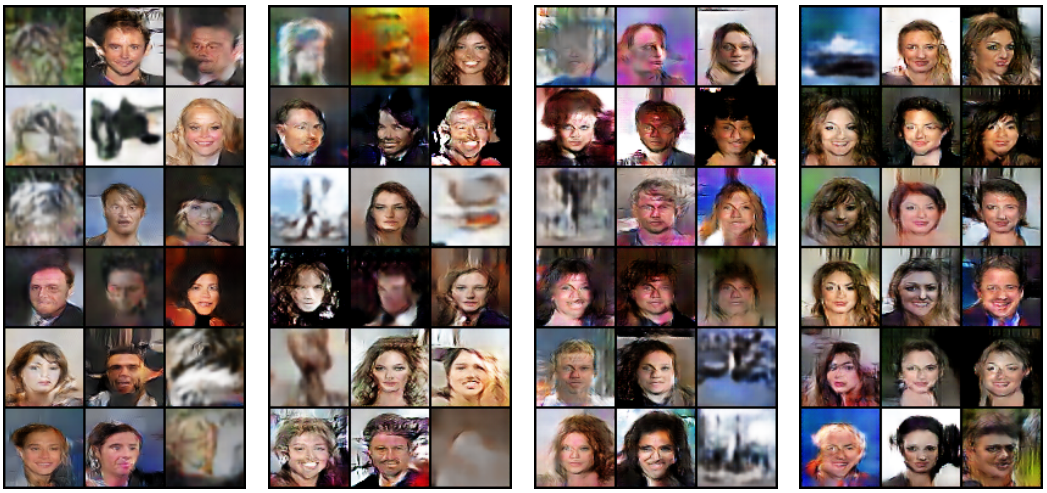
(a) baseline (b) 1st iteration (c) 3rd iteration (d) 5th iteration

Figure 1: **Qualitative performance of ILFB for GANs:** We apply ILFB to a dataset of 90% MNIST "good" images + 10% Fashion-MNIST "bad". The panels show the fake images from 32 randomly chosen (and then fixed) latent vectors, as ILFB iterations update the GAN weights. Baseline is the standard training of fitting to all samples. We can see that the baseline generates both digit and fashion images, but by the 5th iteration it hones in on digit images.

Results Table 1 shows the results for the baseline, oracle and our methods. Please see the caption thereof. We observe significant improvement over the baseline under all experiments. In Appendix B, we provide a comparison of using different initialization methods, where ILFB performs better than its counterpart with random initialization.

7 ILFB FOR DEEP GENERATIVE MODELS WITH MIXED TRAINING DATA

Experimental setting We consider training a GAN – specifically, the DC-GAN architecture Radford et al. (2015) – to generate images similar to those from a *good* dataset, but when the training data given to it contains some fraction of the samples from a different *bad* dataset. All images are unlabeled, and we do not know which training sample comes from which dataset. We investigated the efficacy of our approach in three such settings:
(a) When the good dataset is the Celeb-A face images, and the bad dataset is CIFAR-10.



(a) baseline (b) 1st iteration (c) 3rd iteration (d) 5th iteration

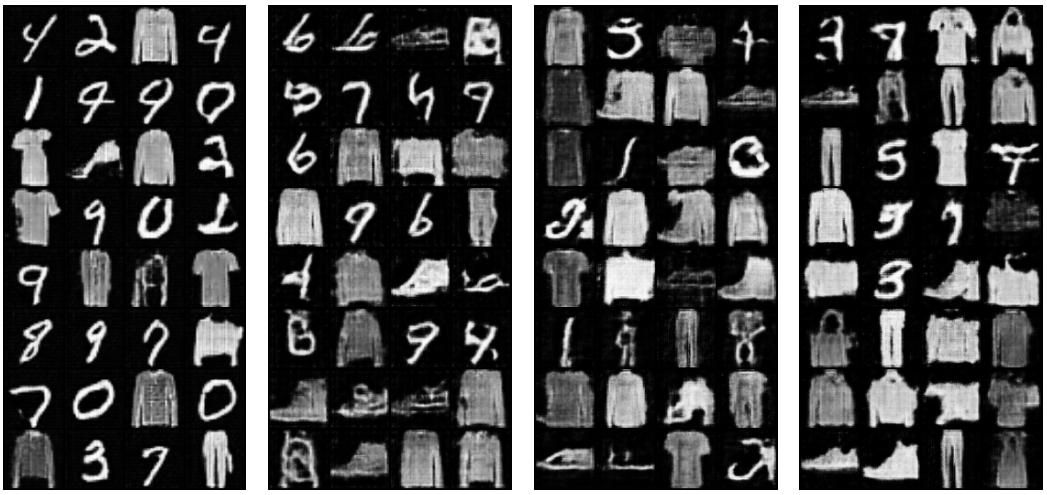
Figure 2: **Qualitative performance of ILFB for GANs:** Given training data consisting of 70% CelebA "good" images + 30% CIFAR-10 "bad" images, the four panels above each show the performance after iterations of our ILFB algorithm. First we choose 18 random vectors in latent pace, and fix them for all iterations. In each iteration, we retrain the GAN on corresponding selected samples to get new weighs, which are then used to generate the 18 fake images. Baseline refers to the standard training where we fit to all the samples (this is also our initialization). Visually, we can see that the generator is able to improve its generation quality and only generate face-like images after the 5th iteration.

(b) When the good dataset is the CIFAR-10 face images, and the bad dataset is Celeb-A.
(c) When the good dataset is MNIST digits, and the bad is Fashion-MNIST Xiao et al. (2017).
For each of these, we consider different fractions of bad samples in the training data, evaluate their effect on standard GAN training, and then the efficacy of our approach as we execute it for upto 5 iterations.

Algorithm Recall that training a GAN consists of updating the weights of both a generator network and a discriminator network; our model θ is the parameters of *both* of these networks. Unlike the classification seeing, we now need to develop a loss $f_\theta(s)$ – and unlike in the simple Gaussian mixture model setting we do not have explicit access to a likelihood function for the generative model. Our crucial innovation is to use the loss[7] at the output of the discriminator – the same one used to train the discriminator – as the $f_\theta(s)$. The algorithm starts by training on all samples, and then alternates between picking training samples with the smallest discriminator loss, and retraining the model on these picked samples. Here training means updating both the generator and discriminator network weights, which is done via SGD. Again, we set τ to be 5% less than the true ratio of "good" samples.

Results We evaluate the effect of bad training samples, and the improvement by our approach, in two ways: quantitatively in Table 2 and qualitatively in Figures 1 and 2. Figure 3 illustrates a failure case, when the bad set is too big, which we feel gives insight into what is going on. Please see the respective captions. The failure example happens for MNIST-Fashion dataset when ratio of MNIST images is 60%. In fact, for every iteration, ILFB selects all images from FashionMNIST (which counts for $0.4n$) and $0.15n$ MNIST images (the least number of MNIST images the algorithm can select). See Fig. 3 for qualitative evaluation of the results.

[7]Notice that for different GAN architectures, the loss function for training the discriminator varies, however, we can always find a surrogate loss by modifying the original loss function, and use the loss of the discriminator for real images as the surrogate loss for ILFB .



(a) baseline (b) 1st iteration (c) 3rd iteration (d) 5th iteration

Figure 3: **Illustrative failure case:** This figure shows that when the fraction of "bad" samples is too large, ILFB cannot clean them out. The setting is exactly the same as in Figure 1, but now with 60% MNIST "good" images + 40% Fashion-MNIST "bad" images. We can see that now the 5th iteration still retains the fake fashion images. Please see the discussion section for some intuition on this.

8 DISCUSSION

We demonstrated, both theoretically for simpler settings, and empirically for more challenging neural network ones, the merit of iteratively fitting to the best samples – in both supervised and unsupervised problems. The underlying **intuition** is that ILFB can focus in on the "good" samples, provided it is properly initailized; and when the number of bad samples is significant but not overwhelming, initial fitting on all the data is good enough to get started. We now add several discussions on the merits and otherwise of our results:
(1) One way to view the effect of ILFB in the GAN setting: a discriminator is likely least certain on the smallest modes of a distribution, and our process of dropping training samples based on this results in the elimination of smaller modes, which correspond to modes from the bad samples. This is illustrated by the failure case in Fig. 3, where the presence of a high fraction of "bad" fashion-MNIST samples makes their modes comparable to those of the "good" MNIST.
(2) Retaining samples best fit by the current model is inherently local, in the sense that samples with large errors are ignored. An initial fit on all the samples is thus important; without this we have bad performance.
(3) Our theory results show convergence to the noise floor when n is $O(d)$, but it is not clear that with much larger number of samples the algorithm cannot go below.

REFERENCES

Oren Anava and Shie Mannor. Heteroscedastic sequences: beyond gaussianity. In *International Conference on Machine Learning*, pp. 755–763, 2016.

Sivaraman Balakrishnan, Martin J Wainwright, Bin Yu, et al. Statistical guarantees for the em algorithm: From population to sample-based analysis. *The Annals of Statistics*, 45(1):77–120, 2017.

Kush Bhatia, Prateek Jain, and Purushottam Kar. Robust regression via hard thresholding. In *Advances in Neural Information Processing Systems*, pp. 721–729, 2015.

Ashish Bora, Eric Price, and Alexandros G. Dimakis. AmbientGAN: Generative models from lossy measurements. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Hy7fDog0b`.

Stéphane Boucheron, Maud Thomas, et al. Concentration inequalities for order statistics. *Electronic Communications in Probability*, 17, 2012.

Kamalika Chaudhuri, Prateek Jain, and Nagarajan Natarajan. Active heteroscedastic regression. In *International Conference on Machine Learning*, pp. 694–702, 2017.

Yudong Chen, Constantine Caramanis, and Shie Mannor. Robust sparse regression under adversarial corruption. In *International Conference on Machine Learning*, pp. 774–782, 2013.

Yudong Chen, Xinyang Yi, and Constantine Caramanis. A convex formulation for mixed regression with two components: Minimax optimal rates. In *Conference on Learning Theory*, pp. 560–604, 2014.

Ilias Diakonikolas, Gautam Kamath, Daniel M Kane, Jerry Li, Jacob Steinhardt, and Alistair Stewart. Sever: A robust meta-algorithm for stochastic optimization. *arXiv preprint arXiv:1803.02815*, 2018.

Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24 (6):381–395, 1981.

Benoît Frénay, Ata Kabán, et al. A comprehensive introduction to label noise. In *ESANN*, 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ashish Khetan, Zachary C Lipton, and Anima Anandkumar. Learning from noisy singly-labeled data. *arXiv preprint arXiv:1712.04577*, 2017.

Adam Klivans, Pravesh K Kothari, and Raghu Meka. Efficient algorithms for outlier-robust regression. *arXiv preprint arXiv:1803.03241*, 2018.

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

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yuanzhi Li and Yingyu Liang. Learning mixtures of linear regressions with nearly optimal complexity. *arXiv preprint arXiv:1802.07895*, 2018.

Liu Liu, Yanyao Shen, Tianyang Li, and Constantine Caramanis. High dimensional robust sparse regression. *arXiv preprint arXiv:1805.11643*, 2018.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*, pp. 961–971, 2017.

Aditya Krishna Menon, Brendan Van Rooyen, and Nagarajan Natarajan. Learning from binary labels with instance-dependent corruption. *arXiv preprint arXiv:1605.00751*, 2016.

Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6): 47–60, 1996.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pp. 1196–1204, 2013.

Adarsh Prasad, Arun Sai Suggala, Sivaraman Balakrishnan, and Pradeep Ravikumar. Robust estimation via robust gradient estimation. *arXiv preprint arXiv:1802.06485*, 2018.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Avik Ray, Joe Neeman, Sujay Sanghavi, and Sanjay Shakkottai. The search problem in mixture models. *Journal of Machine Learning Research*, 18(206):1–61, 2018.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

Clayton Scott, Gilles Blanchard, and Gregory Handy. Classification with asymmetric label noise: Consistency and maximal denoising. In *Conference On Learning Theory*, pp. 489–511, 2013.

Hanie Sedghi, Majid Janzamin, and Anima Anandkumar. Provable tensor methods for learning mixtures of generalized linear models. In *Artificial Intelligence and Statistics*, pp. 1223–1231, 2016.

Sainbayar Sukhbaatar and Rob Fergus. Learning from noisy labels with deep neural networks. *arXiv preprint arXiv:1406.2080*, 2(3):4, 2014.

Daniel Vainsencher, Shie Mannor, and Huan Xu. Ignoring is a bliss: Learning with large noise through reweighting-minimization. In *Conference on Learning Theory*, pp. 1849–1881, 2017.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Xinyang Yi, Constantine Caramanis, and Sujay Sanghavi. Alternating minimization for mixed linear regression. In *International Conference on Machine Learning*, pp. 613–621, 2014.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *arXiv preprint arXiv:1805.07836*, 2018.

A EFFECT OF MIS-SPECIFICATION

We test the performance of ILFB under mis-specification we choose random $\theta^\star \in \mathbb{R}^d, \|\theta^\star\| = 1$, $\tau = 1$, $\hat{\tau} = 0.95$, $d = 100$, $n \in \{500, 1000, 1500\}$, $T = 30$. As shown in Fig. 4, as σ gets larger, the gap between Top-Sel and OLS estimator (in terms of ℓ_2 distance) increases linearly.

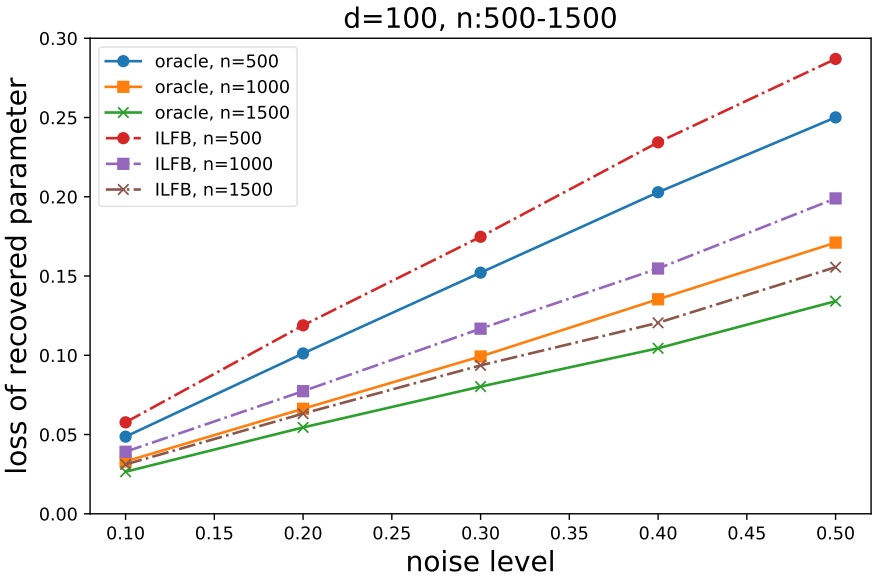


Figure 4: Effect of mis-specification for linear regression model

B RANDOM INITIALIZATION FOR CLASSIFICATION

We study the effect of initialization for the classification problem using MNIST (5%) dataset. In Table 3, we present the accuracy after 5-th iteration of training for both random initialization and the original ILFB methods. Even with random initialization, the performance is consistently better than the baseline, however, the improvement of accuracy is smaller when compared to the original ILFB , which uses all samples for training as initialization.

$\frac{\text{\# good}}{\text{\# total}}$	60%	70%	80%	90%
Baseline	70.26	85.95	92.62	93.88
Random Initialization + ILFB	82.05	91.48	93.80	93.79
ILFB	90.00	92.09	94.30	94.41

Table 3: MNIST-5% dataset with systematic label error using ILFB with different initialization methods. The accuracy at the 5-th iteration is shown for both initialization methods. Results are averaged over 5 runs.

C LINEAR REGRESSION MODEL

C.1 PROOF FOR UPDATE STEPS IN LINEAR REGRESSION MODEL

Theorem 5 (Restate of Theorem 1). *For the mixed linear regression setting as above, suppose that some iterate θ_t satisfies $\|\theta_t - \theta^\star\| < \min_{j\in[m]} \|\theta_t - \theta^{(j)}\|$, and the τ in Algorithm 1 is less than the true ratio of good samples. Then, for any $C < 1$, the next iterate θ_{t+1} of the algorithm satisfies*

$$\|\theta_{t+1} - \theta^\star\| \leq C\left(\|\theta_t - \theta^\star\| + \sigma\right) \tag{3}$$

with high probability (i.e., with probability $1 - n^{-c_0}$), provided the number of samples $n \geq c_1 \frac{d}{C^2 \tau^4}$, where c_1 is a constant.

For simplicity of the notation, we present the proof by assuming $\mathbf{y}_i$ for bad samples have the form $\mathbf{y}_i = \mathbf{n}_i + \mathbf{e}_i$, where $\mathbf{n}_i$ corresponds to $\mathbf{x}_i^\top \theta^{(j)}$, such that $i \in S_{\text{bad}}^j$.

Let θ_t be the current learned parameter. θ_{t+1} is learned by using one iteration in Algorithm 1. Namely, we first select a subset S_t of size $\hat{\tau} n$ with the smallest loss $(\mathbf{y}_i - \mathbf{x}_i^\top \theta_t)^2$, then re-calculate the MLE estimator only using those samples. Denote $\mathbf{W}_t$ as the diagonal matrix whose diagonal entry $\mathbf{W}_{ii}$ equals 1 when the i-th sample is in set S_t, otherwise equals 0. Then,

$$\theta_{t+1} = \left(\mathbf{X}^\top \mathbf{W}_t \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{W}_t \mathbf{y}, \tag{4}$$

where we used the fact that $\mathbf{W}_t^2 = \mathbf{W}_t$. Notice that $\mathbf{W}^\star$ is the ground truth for $\mathbf{W}_t$ (accordingly, define $S^\star$ for the ground truth of S_t). For clearness of the presentation, we ignore the subscript t when there is no ambiguation. Then, according to the definition of $\mathbf{y}$ and reorganizing the terms, (4) can be expanded as

$$\begin{aligned}
\theta_{t+1} - \theta^\star &= \left(\mathbf{X}^\top \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{W} \left(\mathbf{W}^\star \mathbf{X} \theta^\star + \left(\mathbf{I} - \mathbf{W}^\star\right) \mathbf{n} + \mathbf{e}\right) - \theta^\star \\
&= \left(\mathbf{X}^\top \mathbf{W} \mathbf{X}\right)^{-1} \left(\mathbf{X}^\top \mathbf{W} \mathbf{W}^\star \mathbf{X} \theta^\star + \mathbf{X}^\top \mathbf{W} \mathbf{n} - \mathbf{X}^\top \mathbf{W} \mathbf{W}^\star \mathbf{n} - \mathbf{X}^\top \mathbf{W} \mathbf{X} \theta^\star + \mathbf{X}^\top \mathbf{W} \mathbf{e}\right) \\
&= \left(\mathbf{X}^\top \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^\top \left(\mathbf{W} \mathbf{W}^\star - \mathbf{W}\right) \left(\mathbf{X} \theta^\star - \mathbf{n} - \mathbf{e}\right) + \left(\mathbf{X}^\top \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{W} \mathbf{W}^\star \mathbf{e}
\end{aligned}$$

where as we addressed at the beginning of this section, for vector $\mathbf{n}$, given $i \in S_{\text{bad}}$, $\mathbf{n}_i = \mathbf{x}_i^\top \theta^{(j)}$ for some $j \in [m]$. Therefore,

$$\|\theta_{t+1} - \theta^\star\| \tag{5}$$

$$\leq \underbrace{\left\| \left(\mathbf{X}^\top \mathbf{W} \mathbf{X}\right)^{-1} \right\|}_{T_1} \cdot \underbrace{\left\| \mathbf{X}^\top \left(\mathbf{W} \mathbf{W}^\star - \mathbf{W}\right) \left(\mathbf{X} \theta^\star - \mathbf{n} - \mathbf{e}\right) \right\|}_{T_2} + \underbrace{\left\| \left(\mathbf{X}^\top \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{W} \mathbf{W}^\star \mathbf{e} \right\|}_{T_3} \tag{6}$$

We bound the three terms on the right hand side of (6) separately.

C.1.1 HELPFUL LEMMAS

Lemma 6 (Sub-Gamma property for the q-quantile)**.** *Let $n \geq 3$, let $X_{(1)} \geq \cdots \geq X_{(n)}$ be the order statistics of the absolute value of standard Gaussian samples. qn is an integer for some constant $q \in (0, 1)$. Let $v_n = \frac{8}{qn \log 2}$. For all $0 \leq \lambda < \frac{qn}{2\sqrt{v_n}}$,*

$$\log \mathcal{E}\left[e^{\lambda\left(X_{(qn)} - \mathcal{E}\left[X_{(qn)}\right]\right)}\right] \leq \frac{v_n \lambda^2}{2\left(1 - 2\lambda\sqrt{\frac{v_n}{qn}}\right)} \tag{7}$$

This shows the sub-gamma property of $X_{(qn)}$, and as a result, for all $t > 0$, and constant $\frac{1}{2} < q < 1$,

$$\mathcal{E}\left[X_{(qn)}\right] = \psi^{-1}\left(\frac{1-q}{2}\right) \tag{8}$$

$$\mathcal{P}\left[\left|X_{(qn)} - \mathcal{E}\left[X_{(qn)}\right]\right| \geq \sqrt{2c_1 v_n \log n} + 2c_1\sqrt{\frac{v_n}{qn}} \log n\right] \leq n^{-c_1}. \tag{9}$$

Proof of Lemma 6. This result is generalized from the result in Boucheron et al. (2012). ■

Lemma 7. *Suppose we have two distributions $\mathcal{D}_1 = \mathcal{N}(0, \Delta^2), \mathcal{D}_2 = \mathcal{N}(0, 1)$ following normal Gaussian distribution. We have fn i.i.d. samples from $\mathcal{D}_1$ and $(1-f)n$ i.i.d. samples from $\mathcal{D}_2$. Denote the set of the top $\bar{f}n$ samples with smallest abstract values as $S_{\bar{f}n}$, where $\bar{f} \leq f$. Then, with high probability, for $\Delta \leq 1$, at least $\frac{1}{2}\bar{f}^2 n$ samples in $S_{\bar{f}n}$ are from distribution $\mathcal{D}_1$.*

Proof of Lemma 7. We only need to consider the boundary case where $\Delta = 1$, which makes $\mathcal{D}_1 = \mathcal{D}_2$. Now, the problem becomes counting the number of red balls in the first $\bar{f}n$ balls where each ball is colored with red with probability $\bar{f}$. The expected number of red balls is $\bar{f}^2 n$. By Chernoff lower tail bound, with probability at least $1 - e^{-\frac{\bar{f}^2 n}{12}}$, the number of red balls is at least $\frac{1}{2}\bar{f}^2 n$. ■

C.1.2 T_1 IN (6):

Lemma 8. *Assume θ_t satisfies $\|\theta_t - \theta^\star\| < \min_{j\in[m]} \|\theta_t - \theta^{(j)}\|$. For the term $\left(\mathbf{X}^\top \mathbf{W}\mathbf{X}\right)^{-1}$ in (6), with high probability, we have:*

$$\sigma_{\min}\left(\mathbf{X}^\top \mathbf{W}\mathbf{X}\right) \geq \frac{1}{2}\left|S_t \cap S^\star\right| \mathcal{C}(c_1, \tau), \tag{10}$$

where $|S_t \cap S^\star| \geq \frac{1}{2}\tau^2 n$, and $\mathcal{C}(c_1, \tau)$ is a function with positve value that is monotonically increasing with both c_1 and τ.

Proof of Lemma 8. We focus on finding concentration results for the smallest eigenvalue of $\mathbf{X}^\top \mathbf{W}\mathbf{X}$. Here, we can not directly apply classic concentration results since $\mathbf{W}$ is not fixed. Instead, each entry in the diagonal of $\mathbf{W}$ is determined by some rule based on $\mathbf{X}$, $\theta^\star$, and θ_t. Rewrite $\mathbf{X}^\top \mathbf{W}\mathbf{X}$ as

$$\begin{aligned}\mathbf{X}^\top \mathbf{W}\mathbf{X} =& \sum_i \mathbf{x}_i \mathbf{x}_i^\top \mathbf{1}\left[l_{i,t} \in \arg \tau n\text{-}\min\left(l_{1,t}, \cdots, l_{n,t}\right)\right] \\ \succeq& \sum_i \mathbf{x}_i \mathbf{x}_i^\top \mathbf{1}\left[l_{i,t} \in \arg \tau n\text{-}\min\left(l_{1,t}, \cdots, l_{n,t}\right)\right] \mathbf{1}\left[i \in S^\star\right]\end{aligned} \tag{11}$$

where $\mathbf{1}[\cdot]$ is the indicator function. Note that the values $\{l_{i,t}\}_{i=1}^n$ are defined as:

$$l_{i,t} = \left(\mathbf{y}_i - \mathbf{x}_i^\top \theta_t\right)^2.$$

More specifically, consider the model setting, we have:

$$\begin{aligned} l_{i,t} &= \left[\mathbf{x}_i^\top (\theta^\star - \theta_t) + \epsilon_i\right]^2, \ \ i \in S^\star \\ l_{i,t} &= \left(\mathbf{n}_i + \epsilon_i - \mathbf{x}_i^\top \theta_t\right)^2, \ \ i \in [n] \backslash S^\star \end{aligned}$$

For all $i \in S^\star$, $l_{i,t}$s have the same distribution as $\delta\mu$, where μ is a $\chi^2(1)$ random variable, and δ is the magnitude $\delta = \|\theta^\star - \theta_t\|_2^2 + \sigma^2$. The bad samples $i \in [n]\backslash S^\star$, for $i \in S_{\text{bad}}^j$ have the same distribution as $(\sigma^2 + \|\theta^{(j)} - \theta_t\|_2^2)\mu$. Also, the assumption in θ_t assumes that the magnitude of the good distribution is smaller. Applying the results in Lemma 6 and Lemma 7, for given $\theta_t, \theta^\star$, there exists a constant c_1 such that the separation value for deciding top-τn or not is lower bounded by $c_1 \mathcal{Q}(\frac{1}{2} + \frac{\tau}{4})\delta$ with high probability, where $\mathcal{Q}(q)$ is the q-quantile of a normal Gaussian distribution. For a fixed c_0, we consider the following quantity:

$$\mathbf{\Sigma}(c_0) = \mathcal{E}\left[\mathbf{x}_i \mathbf{x}_i^\top \mid |\mathbf{x}_i^\top (\theta^\star - \theta_t) + \epsilon_i| \leq c_0 \sqrt{\|\theta^\star - \theta_t\|_2^2 + \sigma^2}\right].$$

By the geometry of the constraint, we have that

$$\sigma_{\min}\left(\mathbf{\Sigma}\left(c_0\right)\right) \geq \int_{-c_0}^{c_0} s^2 \frac{1}{\sqrt{2\pi}} e^{-\frac{s^2}{2}} ds = 1 - 2\psi(c_0) - c_0 e^{-\frac{c_0^2}{2}}$$

In other words, the lower bound we have for the smallest eigenvalue of the expectation of (11) is monotonically increasing with c_0. We denote the value of $1 - 2\psi(c_0) - c_0 e^{-\frac{c_0^2}{2}}$ when $c_0 = c_1 \mathcal{Q}(\frac{1}{2} + \frac{\tau}{4})$ as $\mathcal{C}(c_1, \tau)$, which is monotonically increasing with both c_1 and τ. Therefore, considering the sub-exponential property, we get the concentration result based on the above expected value:

$$\begin{aligned} &\sigma_{\min}\left(\mathbf{X}^\top \mathbf{W}\mathbf{X}\right) \\ \geq& \sigma_{\min}\left(\sum_i \mathbf{x}_i \mathbf{x}_i^\top \mathbf{1}\left[l_{i,t} \in \arg \tau n\text{-}\min\left(l_{1,t}, \cdots, l_{n,t}\right)\right] \mathbf{1}\left[i \in S^\star\right]\right) \\ \geq& \frac{1}{2}\left|S_t \cap S^\star\right| \mathcal{C}(c_1, \tau), \end{aligned}$$

where for the term $|S_t \cap S^\star|$, according to Lemma 7, with high probability, $|S_t \cap S^\star| \geq \frac{1}{2}\hat{\tau}^2 n$.

■

According to Lemma 8,

$$T_1 \leq \frac{2}{|S_t \cap S^\star| \mathcal{C}(c_1, \tau)} \leq \frac{4}{\tau^2 n \mathcal{C}(c_1, \tau)}. \tag{12}$$

C.1.3 T_2 IN (6)

For $i \in S_t \backslash S^\star$, we know that:

$$\mathbf{y}_i = \mathbf{n}_i + \mathbf{e}_i, \left(\mathbf{y}_i - \mathbf{x}_i^\top \theta_t\right)^2 \leq \left(\mathbf{y}_j - \mathbf{x}_j^\top \theta_t\right)^2, \quad i \in S_t \backslash S^\star, \forall j \notin S_t.$$

That means, for $\forall j \in S^\star \backslash S_t$, we know

$$\left(\mathbf{n}_i + \mathbf{e}_i - \mathbf{x}_i^\top \theta_t\right)^2 \leq \left(\mathbf{x}_j^\top \left(\theta^\star - \theta_t\right) + \mathbf{e}_i\right)^2. \tag{13}$$

Now, define

$$\alpha_{j,t} \triangleq \frac{\|\langle \mathbf{x}_j, \theta^\star - \theta_t \rangle + \mathbf{e}_i\|}{\sqrt{\|\theta^\star - \theta_t\|_2^2 + \sigma^2}}, \quad \alpha_t = \min_{j \in S^\star \backslash S_t} \alpha_{j,t}. \tag{14}$$

We let $\tau^\star$ be the true ratio of good samples, $0 < \tau < \tau^\star \leq 1$. For simplicity, we may assume $\tau^\star / \tau$ is a constant.

Lemma 9 (Bounding α_t)**.** *Consider $\alpha_{j,t}$ as defined in (14). Define interval*

$$\mathcal{I}(q) := \left[\mathcal{Q}(q) - c_1 \sqrt{\frac{\log n}{q\tau^\star n}} - c_2 \frac{\log n}{q\tau^\star n}, \mathcal{Q}(q) + c_1 \sqrt{\frac{\log n}{q\tau^\star n}} + c_2 \frac{\log n}{q\tau^\star n}\right].$$

Then, with high probability,

$$\alpha_t = \min_{j \in S^\star \backslash S_t} \alpha_{j,t} \in \mathcal{I}\left(\frac{1}{2} + \frac{\tau}{2\tau^\star}\right).$$

Proof. We know that the relationship between the size of $S^\star \backslash S_t$ and $S_t \backslash S^\star$ as follows:

$$|S^\star \backslash S_t| = |S_t \backslash S^\star| + (\tau^\star - \tau) n.$$

This shows that the set $S^\star \backslash S_t$ has at least $(\tau^\star - \tau) n$ samples. According to the definition of $\alpha_{j,t}$, for a normal vector $\mathbf{x}_j \sim \mathcal{N}(0, \mathbf{I}_d)$, $\alpha_{j,t}$ follows the distribution of the abstract value of a normal random variable. We can not bound α_t by directly bounding the minimum value among all $\alpha_{j,t}$s in the set since the set $S^\star \backslash S_t$ is dependent with all $\mathbf{x}_i$s, and $\alpha_{j,t}$s are not independent anymore. However, we can bound α_t by only looking at the good samples in $S^\star$. Intuitively, α_t is approximately upper bounded by the $\left(\frac{1}{2} + \frac{\tau}{2\tau^\star}\right)$-quantile of a normal Gaussian distribution. We use concentration of order statistics, which is a generalized result based on Boucheron et al. (2012).

Let as consider the q-quantile of a normal Gaussian random variable with τn samples. Based on Lemma 6, we know that with high probability, the q-quantile value belongs to the range

$$\mathcal{I}(q) := \left[\mathcal{Q}(q) - c_1 \sqrt{\frac{\log n}{q\tau n}} - c_2 \frac{\log n}{q\tau n}, \mathcal{Q}(q) + c_1 \sqrt{\frac{\log n}{q\tau n}} + c_2 \frac{\log n}{q\tau n}\right].$$

As a result, $\alpha_t = \min_{j \in S^\star \backslash S_t} \alpha_{j,t} \in \mathcal{I}\left(\frac{1}{2} + \frac{\hat{\tau}}{2\tau}\right)$ with high probability.

■

T_2 can be bounded as follows:

$$
\begin{aligned}
&\left\|\mathbf{X}^\top\left(\mathbf{W}\mathbf{W}^\star-\mathbf{W}\right)\left(\mathbf{X}\theta^\star-\mathbf{n}-\mathbf{e}\right)\right\|_2\\
=&\left\|\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta^\star-\mathbf{n}_i-\mathbf{e}_i\right)\mathbf{x}_i\right\|_2\\
=&\left\|\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta^\star-\mathbf{x}_i^\top\theta_t+\mathbf{x}_i^\top\theta_t-\mathbf{n}_i-\mathbf{e}_i\right)\mathbf{x}_i\right\|_2\\
\leq&\left\|\sum_{i\in S_t\backslash S^\star}\mathbf{x}_i\mathbf{x}_i^\top\right\|\cdot\|\theta^\star-\theta_t\|+\left\|\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta_t-\mathbf{n}_i-\mathbf{e}_i\right)\mathbf{x}_i\right\|_2\\
\leq&\left\|\sum_{i\in[n]}\mathbf{x}_i\mathbf{x}_i^\top\right\|\cdot\|\theta^\star-\theta_t\|+\left\|\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta_t-\mathbf{n}_i-\mathbf{e}_i\right)\mathbf{x}_i\right\|_2\\
\leq&\sqrt{n}\left(1+c\sqrt{\frac{\log n}{n}}\right)\cdot\|\theta^\star-\theta_t\|+\left\|\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta_t-\mathbf{n}_i-\mathbf{e}_i\right)\mathbf{x}_i\right\|_2
\end{aligned}
\tag{15}
$$

The second term in (15) is bounded as follows:

$$
\begin{aligned}
&\left\|\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta_t-\mathbf{n}_i-\mathbf{e}_i\right)\mathbf{x}_i\right\|_2^2\\
=&\left(\mathbf{X}\theta_t-\mathbf{n}-\mathbf{e}\right)^\top\left(\mathbf{W}\mathbf{W}^\star-\mathbf{W}\right)\mathbf{X}\mathbf{X}^\top\left(\mathbf{W}\mathbf{W}^\star-\mathbf{W}\right)\left(\mathbf{X}\theta_t-\mathbf{n}-\mathbf{e}\right)\\
\leq&\sigma\left(\mathbf{X}\mathbf{X}^\top\right)\left(\mathbf{X}\theta_t-\mathbf{n}-\mathbf{e}\right)^\top\left(\mathbf{W}\mathbf{W}^\star-\mathbf{W}\right)\left(\mathbf{X}\theta_t-\mathbf{n}-\mathbf{e}\right)\\
\leq&\left(d+c\sqrt{d\log n}\right)\sum_{i\in S_t\backslash S^\star}\left(\mathbf{x}_i^\top\theta_t-\mathbf{n}_i-\mathbf{e}_i\right)^2\\
\leq&\left(d+c\sqrt{d\log n}\right)|S_t\backslash S^\star|\,\alpha_t^2\left(\|\theta^\star-\theta_t\|^2+\sigma^2\right)
\end{aligned}
$$

with high probability.

As a consequence,

$$
\begin{aligned}
T_2\leq&\left(2\sqrt{n}+\alpha_t\sqrt{\left(d+c\sqrt{d\log n}\right)|S_t\backslash S^\star|}\right)\|\theta^\star-\theta_t\|+\alpha_t\sqrt{\left(d+c\sqrt{d\log n}\right)|S_t\backslash S^\star|}\sigma\\
\leq&c_2\sqrt{n}\|\theta^\star-\theta_t\|_2+c_2\mathcal{Q}(\frac{1}{2}+\frac{\tau}{\tau^\star})\sqrt{\tau^\star n}\left(\sqrt{d}+\sqrt[4]{d\log n}\right)\left(\|\theta^\star-\theta_t\|_2+\sigma\right)
\end{aligned}
\tag{16}
$$

C.1.4 T_3 IN (6)

For this part, we can reuse the result we proved for T_1. Also, notice that with high probability:

$$
\left\|\mathbf{X}^\top\mathbf{W}\mathbf{W}^\star\mathbf{e}\right\|_2^2=\mathbf{e}^\top\mathbf{W}^\star\mathbf{W}\mathbf{X}\mathbf{X}^\top\mathbf{W}\mathbf{W}^\star\mathbf{e}\leq\sigma(\mathbf{X}\mathbf{X}^\top)\mathbf{e}^\top\mathbf{e}\leq 2\left(d+c\sqrt{d\log n}\right)n\sigma^2
$$

Therefore,

$$
T_3\leq\frac{c_3\left(\sqrt{nd}+\sqrt{n\sqrt{d\log n}}\right)}{|S_t\cap S^\star|\mathcal{C}(c_1,\tau)}\sigma
\tag{17}
$$

C.1.5 PUTTING THINGS TOGETHER

Combining (12), (16), and (17), we have:

$$
\begin{aligned}
&\|\theta_{t+1} - \theta^\star\|_2 \\
\leq &\frac{c_2\sqrt{n} + c_2\mathcal{Q}(\frac{1}{2} + \frac{\tau}{\tau^\star})\sqrt{\tau^\star n}\left(\sqrt{d} + \sqrt[4]{d\log n}\right)}{\frac{1}{4}\tau^2 n c_1 \mathcal{Q}(\frac{1}{2} + \frac{\tau}{4})}\|\theta^\star - \theta_t\|_2 \\
&+ \frac{c_2\mathcal{Q}(\frac{1}{2} + \frac{\tau}{\tau^\star})\sqrt{\tau^\star n}\left(\sqrt{d} + \sqrt[4]{d\log n}\right) + c_3\left(\sqrt{nd} + \sqrt{n\sqrt{d\log n}}\right)}{\frac{1}{4}\tau^2 n c_1 \mathcal{Q}(\frac{1}{2} + \frac{\tau}{4})}\sigma
\end{aligned}
$$

with high probability, since $\mathcal{C}(c_1, \tau) = c_1\mathcal{Q}(\frac{1}{2} + \frac{\tau}{4})$. When $n \geq \frac{128c_2^2\mathcal{Q}^2\left(\frac{1}{2}+\frac{\tau}{\tau^\star}\right)\tau^\star d + 32c_2^2 + 128dc_3}{C^2c_1^2\tau^4\mathcal{Q}^2\left(\frac{1}{2}+\frac{\tau}{4}\right)} = O(\frac{d}{C^2\tau^4})$, we have:

$$
\|\theta_{t+1} - \theta^\star\|_2 \leq C\left(\|\theta_t - \theta^\star\|_2 + \sigma\right).
$$

C.2 PROOF OF THE INITIALIZATION FOR SE MODEL

Proof of Theorem 2. We write out the MLE estimator as follows:

$$
\begin{aligned}
\theta_{\mathrm{MLE}} &= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{y} \\
&= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\left(\mathbf{W}^\star\mathbf{X}\theta^\star + \left(\mathbf{I} - \mathbf{W}^\star\right)\mathbf{n} + \mathbf{e}\right) \\
&= \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\left(\mathbf{X}\theta^\star - (\mathbf{I} - \mathbf{W}^\star)\mathbf{X}\theta^\star + (\mathbf{I} - \mathbf{W}^\star)\mathbf{n} + \mathbf{e}\right) \\
&= \theta^\star - \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\left(\mathbf{I} - \mathbf{W}^\star\right)\left(\mathbf{X}\theta^\star - \mathbf{n}\right) + \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{e}
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&\|\theta_{\mathrm{MLE}} - \theta^\star\| \\
\leq &\left\|\sum_{j=1}^{M}\sum_{i\in S_{\mathrm{bad}}^j}\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{x}_i\mathbf{x}_i^\top\left(\theta^\star - \theta^{(j)}\right)\right\| + \left\|\left(\mathbf{X}^\top\mathbf{X}\right)^{-1}\mathbf{X}^\top\mathbf{e}\right\| \\
\leq &\frac{\sum_{j\in[m]}\|\theta^\star - \theta^{(j)}\|\cdot|S_{\mathrm{bad}}^j| + mc\log n}{n - c\log n} + \frac{\sqrt{(d + c\sqrt{d\log n})(n + c\sqrt{\log n})}}{n - c\log n}\sigma \\
\leq &\frac{\sum_{j\in[m]}\|\theta^\star - \theta^{(j)}\|\cdot|S_{\mathrm{bad}}^j| + mc\log n}{n}\left(1 + \frac{2c\log n}{n}\right) + \frac{\sqrt{2(d + c\sqrt{d\log n})}}{\sqrt{n}}\left(1 + \frac{2c\log n}{n}\right)\sigma \\
\leq &\frac{\sum_{j\in[m]}\|\theta^\star - \theta^{(j)}\|\cdot|S_{\mathrm{bad}}^j|}{n}\left(1 + \frac{2c\log n}{n}\right) \\
&+ \left(\frac{mc\log n}{n} + \frac{\sqrt{2(d + c\sqrt{d\log n})}}{\sqrt{n}}\sigma\right)\left(1 + \frac{2c\log n}{n}\right) \qquad (18) \\
\leq &\left(\frac{1}{\kappa}d_m + \frac{mc\log n}{n} + \frac{\sqrt{2(d + c\sqrt{d\log n})}}{\sqrt{n}}\sigma\right)\left(1 + \frac{2c\log n}{n}\right) \\
\leq &\frac{1}{2}d_m
\end{aligned}
$$

for $n \geq c\max\{\frac{\kappa^2(\kappa+2)^2 d\sigma^2}{(\kappa-2)^2 d_m^2}, \frac{m\kappa(\kappa+1)\log m}{(\kappa-2)d_m}\}$, where in (18), we use the assumption $\sum_{j\in[m]}\|\theta^\star - \theta^{(j)}\||S_{\mathrm{bad}}^j| \leq \frac{n}{\kappa}\min_{j\in[m]}\|\theta^\star - \theta^{(j)}\|$.

■

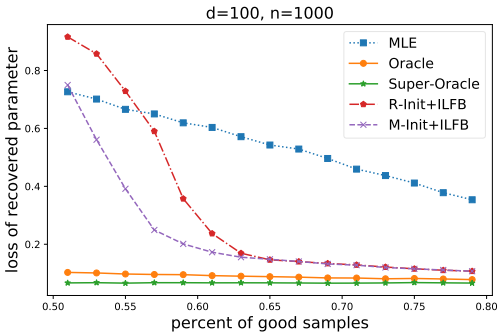


Figure 5: ℓ_2 loss of recovered parameter to the true parameter for (a) MLE: naive MLE/OLS estimator; (b) Oracle: MLE estimator for the subset of good samples; (c) Super-Oracle: MLE estimator for the whole set of samples given the correct output for bad samples; (d) R-Init+ILFB: our algorithm with random initialization; (e)M-Init+ILFB: our algorithm with MLE as initialization. $\sigma = 0.2$, systematic error setting.

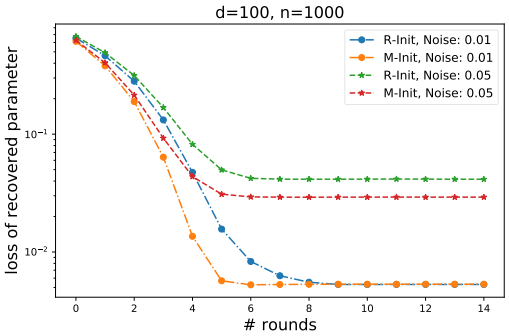


Figure 6: Speed of convergence under systematic error setting, number of good samples is set as 600. M-Init: ILFB using MLE initialization; R-Init: ILFB using random initialization.

C.3 Simulation Results

We verify the performance of ILFB on linear regression model via synthetic experiments. When constructing the dataset, we consider a special case where all bad samples are generated from a single β_j. We test the performance when $\tau^\star$ varies in the interval $[0.5, 0.8]$ with $\sigma = 0.2$ in Fig. 5 We compare: (1) *MLE*: All samples are treated as good samples to find the best parameter; (2) *Oracle*: learn from the $\tau^\star n$ good samples given oracle access; (3) *Super-Oracle*: learn from n samples given oracle access; (4) *R-Init+ILFB* : ILFB with random initialization; (5) *M-Init+ILFB* : ILFB using *MLE* as initialization. The parameter recovery (y-axis) is measured by $\|\theta_{\text{alg}} - \theta^\star\|_2^2$. We can see that ILFB with MLE as initialization performs better than the random initialization counterpart, which shows the effectiveness of taking good initailizations. The speed of convergence is shown in Fig. 6 with multiple levels of measurement noise. The plots show linear convergence of each update step, which matches the result in Theorem 1.

D Gaussian Mixture Model

D.1 Proof for update steps in Gaussian mixture model

Proof. Similar to the proof for linear regression setting, we use S_t to denote the set of selected samples according to parameter θ_t. Then,

$$\theta_{t+1} = \frac{1}{\tau n} \sum_{i \in S_t} \mathbf{x}_i. \tag{19}$$

For $i \in S_{\text{bad}} \cap S_t$, the norm of $\mathbf{x}_i - \theta_t$ is small. More specifically,

$$\|\mathbf{x}_i - \theta_t\| \leq \|\mathbf{x}_l - \theta_t\|, \forall i \in S_{\text{bad}} \cap S_t, \forall l \in S^\star \backslash S_t \tag{20}$$

For any good sample with index l, the value $\|\mathbf{x}_l - \theta_t\| \leq \|\mathbf{x}_l - \theta^\star\| + \|\theta^\star - \theta_t\|$, where the square of $\frac{1}{\sigma^2}\|\mathbf{x}_l - \theta^\star\|$ follows $\chi^2(d)$ distribution. Denote $\mathcal{B}(q)$ be the q-quantile value of $\chi^2(d)$ distribution, with high probability, the q-quantile of $\frac{1}{\sigma^2}\|\mathbf{x}_l - \theta^\star\|$ for all good samples is bounded by $2\mathcal{B}(q)$.

Then,

$$\theta_{t+1} - \theta^\star = \frac{1}{\tau n}\left(\sum_{i\in S_t\setminus S^\star}\mathbf{x}_i - \sum_{i\in S^\star\setminus S_t}\mathbf{x}_i\right) \tag{21}$$

$$\|\theta_{t+1} - \theta^\star\| = \frac{1}{\tau n}\left\|\sum_{i\in S_t\setminus S^\star}(\mathbf{x}_i - \theta_t + \theta_t) - \sum_{i\in S^\star\setminus S_t}(\mathbf{x}_i - \theta^\star + \theta^\star)\right\| \tag{22}$$

$$\leq \frac{1}{\tau n}\left(\left\|\sum_{j\in[m]}\sum_{i\in S^j_{\text{bad}}\cap S_t}(\mathbf{x}_i - \theta_t)\right\| + |S^\star\setminus S_t|\,\|\theta^\star - \theta_t\| + \left\|\sum_{i\in S^\star\setminus S_t}\mathbf{x}_i - \theta^\star\right\|\right) \tag{23}$$

$$\leq \frac{1}{\tau n}|S^\star\setminus S_t|\left(2\,\|\theta^\star - \theta_t\| + 4\sigma\sqrt{d}\mathcal{B}\left(\frac{|S^\star\setminus S_t|}{\tau n}\right)\right) \tag{24}$$

$$\leq \frac{2|S^\star\setminus S_t|}{\tau n}\,\|\theta^\star - \theta_t\| + \frac{c_0|S^\star\setminus S_t|\sqrt{d}}{\tau n}\sigma \tag{25}$$

where $\mathcal{B}\left(\frac{|S^\star\setminus S_t|}{\tau n}\right)$ is uppder bounded by some constant c_0, since $|S^\star\setminus S_t| < \frac{\tau n}{2}$ with high probability.

Given $\|\theta_t - \theta^\star\| \leq \frac{3}{4}\min_{j\in[m]}\|\theta_t - \theta^j\|$, there exists $c_1 \in (0,1)$, such that $|S^\star\setminus S_t| \leq \frac{c_1}{2}\tau n$ with high probability. As a consequence,

$$\|\theta_{t+1} - \theta^\star\| \leq c_1\,\|\theta_t - \theta^\star\| + c_0 c_1\sigma\sqrt{d}. \tag{26}$$

■

D.2 Proof of initialization for Gaussian mixture model

Assume $\left\|\sum_{j\in[m]}(\theta^\star - \theta^{(j)})|S^j_{\text{bad}}|\right\| \leq \alpha(1-\tau)n\min_{j\in[m]}\left\|\theta^\star - \theta^{(j)}\right\|$. Consider the initialization

$$\theta_0 = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i \tag{27}$$

$$\|\theta_0 - \theta^\star\| = \frac{1}{n}\left\|\sum_{i=1}^n \mathbf{x}_i - \theta^\star\right\| \tag{28}$$

$$= \frac{1}{n}\left\|\left[\sum_{i\in S_{\text{good}}}\mathbf{x}_i - \theta^\star + \sum_{i\in S_{\text{bad}}}\mathbf{x}_i - \theta^\star\right]\right\| \tag{29}$$

$$\leq \frac{1}{n}\left\|\sum_{j\in[m]}(\theta^\star - \theta^{(j)})|S^j_{\text{bad}}|\right\| + c\sigma\sqrt{\frac{d}{n}} \tag{30}$$

$$\leq \frac{1}{n}\alpha(1-\tau)n\min_{j\in[m]}\left\|\theta^\star - \theta^{(j)}\right\| + c\sigma\sqrt{\frac{d}{n}} \tag{31}$$

$$\leq \frac{3}{7}\min_{j\in[m]}\left\|\theta^\star - \theta^{(j)}\right\| \tag{32}$$

if $n \geq \frac{cd\sigma^2}{(\frac{3}{7}-\alpha(1-\tau))^2 d_k^2}$. This will imply $\|\theta_0 - \theta^\star\| \leq \frac{3}{4}\min_{j\in[m]}\left\|\theta_0 - \theta^{(j)}\right\|$.

D.3 Simulation results

We also verify the performance of ILFB for Gaussian mixture model through synthetic experiments. In Gaussian mean estimation, we are interested in recovering the mean of the dominant mixture

component. The bad samples are generated from another Gaussian distribution, and the distance of the two centers is set as 1 for convenience. According to Fig. 7, ILFB performs close to Oracle, and is much better than the naive method and ILFB with random initialization. In 8, we see linear convergence at the first few rounds until reaching a static point.

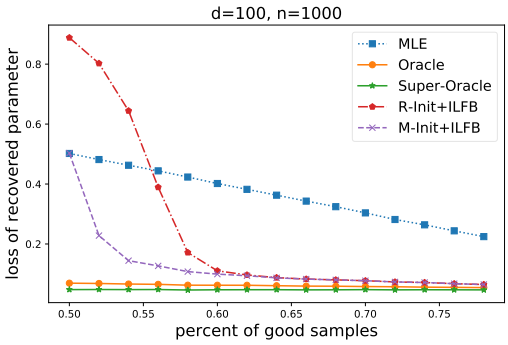


Figure 7: Performance of different algorithms for Gaussian mixture model, measured by the ℓ_2 loss of the recovered parameter to the true parameter, $\sigma = 0.15$

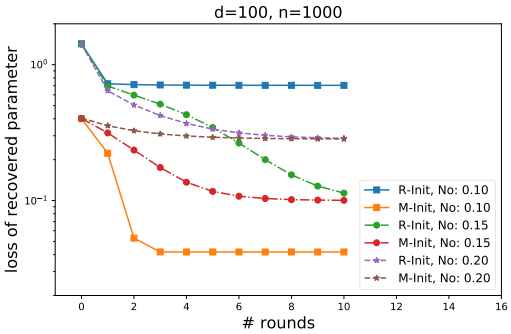


Figure 8: Convergence speed of ILFB under different noise levels, using both MLE initialization and random initialization.