Meta-learning for Mixed Linear Regression

Weihao Kong 1  Raghav Somani 1  Zhao Song 2  Sham Kakade 1  Sewoong Oh 1

Abstract
In modern supervised learning, there are a large number of tasks, but many of them are associated with only a small amount of labelled data. These include data from medical image processing and robotic interaction. Even though each individual task cannot be meaningfully trained in isolation, one seeks to meta-learn across the tasks from past experiences by exploiting some similarities. We study a fundamental question of interest: When can abundant tasks with small data compensate for lack of tasks with big data? We focus on a canonical scenario where each task is drawn from a mixture of \( k \) linear regressions, and identify sufficient conditions for such a graceful exchange to hold; there is little loss in sample complexity even when we only have access to small data tasks. To this end, we introduce a novel spectral approach and show that we can efficiently utilize small data tasks with the help of \( \tilde{\Omega}(\frac{k^3}{2}) \) medium data tasks each with \( \tilde{\Omega}((\frac{k}{2})^2) \) examples.

1. Introduction
Recent advances in machine learning highlight successes on a small set of tasks where a large number of labeled examples have been collected and exploited. These include image classification with 1.2 million labeled examples (Deng et al., 2009) and French-English machine translation with 40 million paired sentences (Bojar et al., 2014). For common tasks, however, collecting clean labels is costly, as they require human expertise (as in medical imaging) or physical interactions (as in robotics), for example. Thus collected real-world datasets follow a long-tailed distribution, in which a dominant set of tasks only have a small number of training examples (Wang et al., 2017).

Inspired by human ingenuity in quickly solving novel problems by leveraging prior experience, meta-learning approaches aim to jointly learn from past experience to quickly adapt to new tasks with little available data (Schmidhuber, 1987; Thrun & Pratt, 2012). This has had a significant impact in few-shot supervised learning, where each task is associated with only a few training examples. By leveraging structural similarities among those tasks, one can achieve accuracy far greater than what can be achieved for each task in isolation (Finn et al., 2017; Ravi & Larochelle, 2017; Koch et al., 2015; Oreshkin et al., 2018; Triantafillou et al., 2019; Rusu et al., 2018). The success of such approaches hinges on the following fundamental question: When can we jointly train small data tasks to achieve the accuracy of large data tasks?

We investigate this trade-off under a canonical scenario where the tasks are linear regressions in \( d \)-dimensions and the regression parameters are drawn i.i.d. from a discrete set of a support size \( k \). Although widely studied, existing literature addresses the scenario where all tasks have the same fixed number of examples. We defer formal comparisons to Section 6.

On one extreme, when large training data of sample size \( \Omega(d) \) is available, each task can easily be learned in isolation; here, \( \Omega(k \log k) \) such tasks are sufficient to learn all \( k \) regression parameters. This is illustrated by a solid circle in Figure 1. On the other extreme, when each task has only one example, existing approaches require exponentially many tasks (see Table 1). This is illustrated by a solid square.

Several aspects of few-shot supervised learning makes training linear models challenging. The number of training examples varies significantly across tasks, all of which are significantly smaller than the dimension of the data \( d \). The number of tasks are also limited, which restricts any algorithm with exponential sample complexity. An example distribution of such heterogeneous tasks is illustrated in Figure 1 with a bar graph in blue, where both the solid circle and square are far outside of the regime covered by the typical distribution of tasks.

In this data scarce regime, we show that we can still efficiently achieve any desired accuracy in estimating the
2. Problem formulation and notations

There are two perspectives on approaching meta-learning: optimization based (Li et al., 2017; Bertinetto et al., 2019; Zhou et al., 2018; Zintgraf et al., 2019; Rajeswaran et al., 2019), and probabilistic (Grant et al., 2018; Finn et al., 2018; Kim et al., 2018; Harrison et al., 2018). Our approach is motivated by the probabilistic view and we present a brief preliminary in Section 2.1. In Section 2.2, we present a simple but canonical scenario where the tasks are linear regressions, which is the focus of this paper.

2.1. Review of probabilistic view on meta-learning

A standard meta-training for few-shot supervised learning assumes that we are given a collection of $n$ meta-training tasks $\{\tau_i\}_{i=1}^n$ drawn from some distribution $\mathbb{P}(\mathcal{T})$. Each task is associated with a dataset of size $t_i$, collectively denoted as a meta-training dataset $\mathcal{D}_{\text{meta-train}} = \{(x_{i,j}, y_{i,j}) \in \mathbb{R}^d \times \mathbb{R} \}_{j \in [t_i]}$, $i \in [n]$. Exploiting some structural similarities in $\mathbb{P}(\mathcal{T})$, the goal is to train a model for a new task $\mathcal{T}_{\text{new}}$, coming from $\mathbb{P}(\mathcal{T})$, from a small amount of training dataset $\mathcal{D} = \{(x_{j,\text{new}}, y_{j,\text{new}})\}_{j \in [t]}$.

Each task $\tau_i$ is associated with a model parameter $\phi_i$, where the meta-training data is independently drawn from: $(x_{i,j}, y_{i,j}) \sim \mathbb{P}_{\phi_i}(y|x)\mathbb{P}(x)$ for all $j \in [t_i]$. The prior distribution of the tasks, and hence the model parameters, is fully characterized by a meta-parameter $\theta$ such that $\phi_i \sim \mathbb{P}_\theta(\phi)$.

Following the definition from (Grant et al., 2018), the meta-learning problem is defined as estimating the most likely meta-parameter given meta-training data by solving

$$\theta^* \in \arg\max_\theta \log \mathbb{P}(\theta | \mathcal{D}_{\text{meta-train}}) ,$$

which is a special case of empirical Bayes methods for learning the prior distribution from data (Carlin & Louis, 2010). Once meta-learning is done, the model parameter of a newly arriving task can be estimated by a Maximum a Posteriori (MAP) estimator:

$$\hat{\phi} \in \arg\max_\phi \log \mathbb{P}(\phi | \mathcal{D}, \theta^*) ,$$

or a Bayes optimal estimator:

$$\hat{\phi} \in \arg\min_\phi \mathbb{E}_{\phi' \sim \mathbb{P}(\phi'|\mathcal{D}, \theta^*)} [\ell(\phi, \phi')] ,$$

for a choice of a loss function $\ell$. This estimated parameter is then used for predicting the label of a new data point $x$ in task $\tau_{\text{new}}$ as

$$\hat{y} \in \arg\max_y \mathbb{P}_{\hat{\phi}}(y|x) .$$
2.2. Linear regression with a discrete prior

In general, the meta-learning problem of (1) is computationally intractable and no statistical guarantees are known. To investigate the trade-offs involved, we assume a simple but canonical scenario where the tasks are linear regressions:

\[ x_{i,j} \sim P_x, \quad y_{i,j} = \beta_i^\top x_{i,j} + \epsilon_{i,j}, \tag{5} \]

for the \( i \)-th task and \( j \)-th example. Each task is associated with a model parameter \( \phi_i = (\beta_i, \sigma_i) \in \mathbb{R}^d \times \mathbb{R}^+ \). The noise \( \epsilon_{i,j} \) is i.i.d. as \( \epsilon_{i,j} \sim P_{\epsilon_i} \), and \( P_{\epsilon_i} \) is a centered sub-Gaussian distribution with parameter \( \sigma_i^2 \). Without loss of generality, we assume that \( P_x \) is an isotropic (i.e., \( \mathbb{E} [x_{i,j} x_{i,j}^\top] = \mathbf{I}_d \)) centered sub-Gaussian distribution. If \( P_x \) is not isotropic, we assume there are large number of \( x_{i,j} \)'s for whitening such that \( P_x \) is sufficiently close to isotropic.

We do not make any assumption on the prior of \( \phi_i \)'s other than that they come from a discrete distribution of a support size \( k \). Concretely, the meta-parameter \( \theta = (W \in \mathbb{R}^{d \times k}, s \in \mathbb{R}^k, p \in \mathbb{R}^k \cap B_{1,k}(0, 1)) \) defines a discrete prior (which is also known as mixture of linear experts (Chaganty & Liang, 2013)) on \( \phi_i \)'s, where \( W = [w_1, \ldots, w_k] \) are the \( k \) candidate model parameters, and \( s = [s_1, \ldots, s_k] \) are the \( k \) candidate noise parameters. The \( i \)-th task is randomly chosen from one of the \( k \) components from distribution \( p \), denoted by \( z_i \sim \text{multinomial}(p) \). The training data is independently drawn from (5) for each \( j \in [t_i] \) with \( \beta_i = w_{z_i} \) and \( \sigma_i = s_{z_i} \).

We want to characterize the sample complexity of this meta-learning. This depends on how complex the ground truths prior \( \theta \) is. This can be measured by the number of components \( k \), the separation between the parameters \( W \), the minimum mixing probability \( p_{\min} \), and the minimum positive eigen-value \( \lambda_{\min} \) of the matrix \( \sum_{j=1}^k p_j w_j w_j^\top \).

Notations. We define \( \rho_i := \sqrt{s_{z_i}^2 + \|w_{z_i}\|_2^2} \) as the sub-Gaussian norm of a label \( y_{i,j} \) in the \( i \)-th task, and \( \rho^2 := \max_i \rho_i^2 \). Without loss of generality, we assume \( \rho = 1 \), which can be always achieved by scaling the meta-parameters appropriately. We also define \( p_{\min} := \min_{j \in [k]} p_j \) and \( \Delta := \min_{i,j \in [k], i \neq j} \|w_i - w_j\|_2 \) and assume \( p_{\min}, \Delta > 0 \). \( \omega \in \mathbb{R}^+ \) is such that two \( n \times n \) matrices can be multiplied in \( \mathcal{O}(n^\omega) \) time.

3. Algorithm

We propose a novel spectral approach (Algorithm 1) to solve the meta-learning linear regression, consisting of three sub-algorithms: subspace estimation, clustering, and classification. These sub-algorithms require different types of tasks, depending on how many labelled examples are available.

Clustering requires heavy tasks, where each task is associated with many labelled examples, but we need a smaller number of such tasks. On the other hand, for subspace estimation and classification, light tasks are sufficient, where each task is associated with a few labelled examples. However, we need a large number of such tasks. In this section, we present the intuition behind our algorithm design, and the types of tasks required. Precisely analyzing these requirements is the main contribution of this paper, to be presented in Section 4.

3.1. Intuitions behind the algorithm design

We give a sketch of the algorithm below. Each step of meta-learning is spelled out in full detail in Section 5. This provides an estimated meta-parameter \( \tilde{\theta} = (\tilde{W}, \tilde{s}, \tilde{p}) \). When a new task arrives, this can be readily applied to solve for prediction, as defined in Definition 4.4.

Algorithm 1

Meta-learning

1. Subspace estimation. Compute subspace \( U \) which approximates \( \text{span} \{w_1, \ldots, w_k\} \), with singular value decomposition.

2. Clustering. Project the heavy tasks onto the subspace of \( U \), perform distance-based \( k \) clustering, and estimate \( \tilde{w}_i \) for each cluster.

3. Classification. Perform likelihood-based classification of the light tasks using \( \tilde{w}_i \) estimated from the Clustering step, and compute the more refined estimates \( (\tilde{w}_i, \tilde{s}_i, \tilde{p}_i) \) of \( (w_i, s_i, p_i) \) for \( i \in [k] \).

Prediction

4. Prediction. Perform MAP or Bayes optimal prediction using the estimated meta-parameter as a prior.

Subspace estimation. The subspace spanned by the regression vectors, \( \text{span} \{w_1, \ldots, w_k\} \), can be easily estimated using data from the (possibly) light tasks with only \( t_i \geq 2 \). Using any two independent examples from the same task \( (x_{i,1}, y_{i,1}), (x_{i,2}, y_{i,2}) \), it holds that \( \mathbb{E} [y_{i,1} y_{i,2} x_{i,1} x_{i,2}^\top] = \sum_{j=1}^k p_j w_j w_j^\top \). With a total of \( \Omega(d \log d) \) such examples, the matrix \( \sum_{j=1}^k p_j w_j w_j^\top \) can be accurately estimated under spectral norm, and so is the column space.
Clustering. Given an accurate estimation of the subspace \( \text{span}\{\mathbf{w}_1, \ldots, \mathbf{w}_k\} \), we can reduce the problem from a \( d \)-dimensional to a \( k \)-dimensional regression problem by projecting \( \mathbf{x} \) onto the subspace of \( \mathbf{U} \). Tasks with \( t_i = \Omega(k) \) examples can be individually trained as the unknown parameter estimation (i.e. meta-learning) phase.

To this end, we borrow techniques from recent advances in parameter estimation (i.e. meta-learning) phase. We start the classification step, once each cluster has enough (i.e. \( \Omega(k) \)) regression vectors are used to check if the two tasks are in the same clusters. We can estimate \( \|\mathbf{w}_i - \mathbf{w}_j\|_2^2 \) accurately with \( t = O(\sqrt{d}) \). We use this technique to estimate \( \|\mathbf{w}_i - \mathbf{w}_j\|_2^2 \), whose value can be used to check if the two tasks are in the same clusters. We cluster the tasks with \( t_i = \Omega(\sqrt{k}) \) into \( k \) disjoint clusters. We call this step clustering.

After clustering, resulting estimated \( \hat{\mathbf{w}}_i \)'s have two sources of error: the error in the subspace estimation, and the error in the parameter estimation for each cluster. If we cluster more heavy tasks, we can reduce the second error but not the first. We could increase the samples used in subspace estimation, but there is a more sample efficient way: classification.

Classification. We start the classification step, once each cluster has enough (i.e. \( \Omega(k) \)) datapoints to obtain a rough estimation of their corresponding regression vector. In this regime, we have \( O(1) \) error in the estimated \( \hat{\mathbf{w}}_i \)'s. This is sufficient for us to add more datapoints to grow each of the clusters. When enough data points are accumulated (i.e. \( O(d) \) for each cluster), then we can achieve any desired accuracy with this larger set of accurately classified tasks. This separation of the roles of the three sub-algorithms is critical in achieving the tightest sample complexity.

In contrast to the necessary condition of \( t_i = \Omega(\sqrt{k}) \) for the clustering step, we show that one can accurately determine which cluster a new task belongs to with only \( t_i = \Omega(\log k) \) examples once we have a rough initial estimation \( \hat{\mathbf{W}} \) of the parameter \( \mathbf{W} \). We grow the clusters by adding tasks with a logarithmic number of examples until we have enough data points per cluster to achieve the desired accuracy. We call this step classification. This concludes our algorithm for the parameter estimation (i.e. meta-learning) phase.

4. Main results

Suppose we have \( n_H \) heavy tasks each with at least \( t_H \) training examples, and \( n_L \) light tasks each with at least \( t_L \) training examples. If heavy tasks are data rich (\( t_H \gg d \)), we can learn \( \mathbf{W} \) straightforwardly from a relatively small number, i.e. \( n_H = \Omega(k \log k) \). If the light tasks are data rich (\( t_L \gg k \)), they can be straightforwardly clustered on the projected \( k \)-dimensional subspace. We therefore focus on the following challenging regime of data scarcity.

Assumption 1. The heavy dataset \( \mathcal{D}_H \) consists of \( n_H \) heavy tasks, each with at least \( t_H \) samples. The first light dataset \( \mathcal{D}_{L1} \) consists of \( n_{L1} \) light tasks, each with at least \( t_{L1} \) samples. The second light dataset \( \mathcal{D}_{L2} \) consists of \( n_{L2} \) tasks, each with at least \( t_{L2} \) samples. We assume \( t_{L1}, t_{L2} < k \), and \( t_H < d \).

To give more fine grained analyses on the sufficient conditions, we assume two types of light tasks are available with potentially differing sizes (Remark 4.3). In meta-learning step in Algorithm 1, subspace estimation uses \( \mathcal{D}_{L1} \), clustering uses \( \mathcal{D}_H \), and classification uses \( \mathcal{D}_{L2} \). We provide proofs of the main results in Appendices A, B, and C.

4.1. Meta-learning

We characterize a sufficient condition to achieve a target accuracy \( \epsilon \) in estimating the meta-parameters \( \theta = (\mathbf{W}, \mathbf{s}, \mathbf{p}) \).

Theorem 1 (Meta-learning). For any failure probability \( \delta \in (0,1) \), and accuracy \( \epsilon \in (0,1) \), given three batches of samples under Assumption 1, meta-learning step of Algorithm 1 estimates the meta-parameters with accuracy

\[
\|\hat{\mathbf{w}}_i - \mathbf{w}_i\|_2 \leq \epsilon s_i, \\
|\hat{s}_i^2 - s_i^2| \leq \frac{\epsilon s_i^2}{\sqrt{d}}, \text{ and} \\
|\hat{p}_i - p_i| \leq \frac{\epsilon}{\sqrt{d}} p_i,
\]

with probability at least \( 1 - \delta \), if the following holds. The numbers of tasks satisfy

\[
n_{L1} = \Omega\left( \frac{d \log^3 \left( \frac{d}{\min \Delta^2} \right)}{t_{L1}} \right) \cdot \min \left\{ \Delta^{-6} p_{min}^{-2}, \Delta^{-2} \lambda_{min}^{-2} \right\},
\]

\[
n_H = \Omega\left( \frac{\log(k/\delta)}{t_H p_{min} \Delta^2} \left( k + \Delta^{-2} \right) \right),
\]

\[
n_{L2} = \Omega\left( \frac{d \log^2 (k/\delta)}{t_{L2} p_{min} \Delta^2} \right),
\]

and the numbers of samples per task satisfy \( t_{L1} \geq 2 \), \( t_{L2} = \Omega \left( \log(k/\min \delta) / \Delta^2 \right) \), and \( t_H = \Omega \left( \Delta^{-2} \sqrt{k} \log(k/(\min \delta)) \right) \), where \( \lambda_{min} \) is the smallest non-zero eigen value of \( \mathbf{M} := \sum_{j=1}^k p_j \mathbf{w}_j \mathbf{w}_j^\top \in \mathbb{R}^{d \times d} \).
In the following remarks, we explain each of the conditions.

**Remark 4.1** (Dependency in $D_{1,1}$). The total number of samples used in subspace estimation is $n_{1,1}t_{1,1}$. The sufficient condition scales linearly in $d$ which matches the information theoretically necessary condition up to logarithmic factors. If the matrix $M$ is well conditioned, for example when $w_i$’s are all orthogonal to each other, subspace estimation is easy, and $n_{2,1}t_{1,1}$ scales as $\Delta^{-2} \min \{ p_{\min}, \}$.

**Remark 4.2** (Dependency in $D_H$). The clustering step requires $t_H = \Omega(\sqrt{k})$, which is necessary for distance-based clustering approaches such as single-linkage clustering. From (Kong & Valiant, 2018; Kong et al., 2019) we know that it is necessary (and sufficient) to have $t = \Theta(\sqrt{k})$, even for a simpler testing problem between $\beta_1 = \beta_2$ or $\|\beta_1 - \beta_2\|^2 > 0$, from two labelled datasets with two linear models $\beta_1$ and $\beta_2$.

Our clustering step is inspired by (Vempala & Wang, 2004) on clustering under Gaussian mixture models, where the algorithm succeeds if $t_H = \Omega(\Delta^{-2} \sqrt{k})$. Although a straightforward adaptation fails, we match the sufficient condition, we only require the number of heavy samples $n_{Ht_H}$ to be $\Omega(k/p_{\min})$ up to logarithmic factors, which is information theoretically necessary.

**Remark 4.3** (Gain of using two types of light tasks). To get the tightest guarantee, it is necessary to use a different set of light tasks to perform the final estimation step. First notice that the first light dataset $D_{1,1}$ does not cover the second light dataset since we need $t_{1,2} \geq \Omega(\log(kd))$ which does not need to hold for the first dataset $D_{1,1}$. On the other hand, the second light dataset does not cover the first light dataset in the setting where $\Delta$ or $p_{\min}$ is very small.

**Remark 4.4** (Dependency in $D_{1,2}$). Classification and prediction use the same routine to classify the given task. Hence, the log $k$ requirement in $t_{1,2}$ is tight, as it matches our lower bound in Proposition 4.6. The extra terms in the log factor come from the union bound over all $n_{1,2}$ tasks to make sure all the tasks are correctly classified. It is possible to replace it by $\log(1/\epsilon)$ by showing that $\epsilon$ fraction of incorrectly classified tasks does not change the estimation by more than $\epsilon$. We only require $n_{1,2}t_{1,2} = \Omega(d/p_{\min})$ up to logarithmic factors, which is information theoretically necessary.

### 4.2. Prediction

Given an estimated meta-parameter $\tilde{\theta} = (\tilde{W}, \tilde{\beta}, \tilde{p})$, and a new dataset $D = \{(x_i^{\text{new}}, y_i^{\text{new}})\}_{i \in [\tau]}$, we make predictions on the new task with unknown parameters using two estimators: MAP estimator and Bayes optimal estimator.

**Definition 4.5.** Define the maximum a posterior (MAP) estimator as

$$\hat{\beta}_{\text{MAP}}(D) = \tilde{w}^\top; \quad \hat{\theta} = \arg\max_{\theta} \log\hat{L}_i, \quad \hat{L}_i = \exp\left(-\sum_{j=1}^\tau \frac{(y_i^{\text{new}} - \tilde{w}^\top x_i^{\text{new}})^2}{2\tilde{s}_i^2} - \tau \log \tilde{s}_i + \log \tilde{p}_i\right).$$

Define the posterior mean estimator as

$$\hat{\beta}_{\text{Bayes}}(D) = \sum_{i=1}^k \hat{L}_i \tilde{w}_i.$$ 

If the true prior, $\{(w_i, s_i, p_i)\}_{i \in [k]}$, is known. The posterior mean estimator achieves the smallest expected squared $\ell_2$ error, $E_{D,\beta_{\text{new}}} \left[\|\beta(D) - \beta_{\text{new}}\|_2^2\right]$. Hence, we refer to it as Bayes optimal estimator. The MAP estimator maximizes the probability of exact recovery.

**Theorem 2** (Prediction). Under the hypotheses of Theorem 1 with $\epsilon \leq \min \left\{\Delta/10, \Delta^2\sqrt{d}/50\right\}$, the expected prediction errors of both the MAP and Bayes optimal estimators $\beta(D)$ are bound as

$$E \left[(\hat{x}^\top \beta(D) - y)^2\right] \leq \delta + (1 + \epsilon^2) \sum_{i=1}^k p_i s_i^2,$$  

if $\tau \geq \Theta \left(\log(k)/\Delta^4\right)$, where the true meta-parameter is $\theta = \{(w_i, s_i, p_i)\}_{i=1}^k$, the expectation is over the new task with model parameter $\phi_{\text{new}} = (\beta_{\text{new}}, \sigma_{\text{new}}) \sim P_{\theta}$, training dataset $D \sim P_{\phi_{\text{new}}}$, and test data $(x, y) \sim P_{\phi_{\text{new}}}$.

Note that the $\sum_{i=1}^k p_i s_i^2$ term in (6) is due to the noise in $y$, and can not be avoided by any estimator. With an accurate meta-learning, we can achieve a prediction error arbitrarily close to this statistical limit, with $\tau = \mathcal{O}(\log k)$. Although both predictors achieve the same guarantee, Bayes optimal estimator achieves smaller training and test errors in Figure 2, especially in challenging regimes with small data.

We show that $\tau = \Omega(\log k)$ training samples are necessary (even if the ground truths meta-parameter $\theta$ is known) to achieve error approaching this statistical limit. Let $\Theta_{k, \Delta, \sigma}$ denote the set of all meta-parameters with $k$ components, satisfying $\|w_i - w_j\|_2 \geq \Delta$ for $i \neq j \in [k]$ and $s_i \leq \sigma$ for all $i \in [k]$. The following minimax lower bound shows that there exists a threshold scaling as $\mathcal{O}(\log k)$ below which no algorithm can achieve the fundamental limit of $\sigma^2$, which is $\sum_{i=1}^k p_i s_i^2$ in this minimax setting.
We explain and analyze each step in Algorithm 1. These analyses imply our main result in meta-learning, which is a continuation of simulations discussed in Appendix E, where we provide extensive experiments confirming our analyses.

Remark 4.6 (Lower bound for prediction). For any \( \sigma, \Delta > 0 \), if \( \tau = \left( (1 + \Delta^2)/\sigma^2 \right)^{-1} \log(k - 1) \), then

\[
\inf \sup_{\theta} \mathbb{E} \left[ (\hat{y}(D, \theta) - y)^2 \right] = \sigma^2 + \Omega(\Delta^2),
\]

where the minimization is over all measurable functions of the meta-parameter \( \theta \) and the training data \( D \) of size \( \tau \).

5. Details of the algorithm and the analyses

We explain and analyze each step in Algorithm 1. These analyses imply our main result in meta-learning, which is explicitly written in Appendix A.

5.1. Subspace estimation

In the following, we use \( k \_SV \) routine that outputs the top \( k \) singular vectors. As \( \mathbb{E}[\hat{\Sigma}] = \Sigma := \sum_{j=1}^{k} p_{ij} \_U \_j \_W \_j \), this outputs an estimate of the subspace spanned by the true parameters. We show that as long as \( t_{L1} \geq 2 \), the accuracy only depends on the total number of examples, and it is sufficient to have \( n_{L1} t_{L1} = \Omega(d) \).

Algorithm 2 Subspace estimation

**Input:** data \( D_{L1} = \{(x_{ij}, y_{ij})\}_{i \in [n_{L1}], j \in [t_{L1}]} \), \( k \in \mathbb{N} \)

**Output:** \( u = k \_SV \_D_{L1} \)

1. **compute** for all \( i \in [n_{L1}] \)
2. \( \hat{\beta}_i^{(1)} \leftarrow \frac{2}{t_{L1}} \sum_{j=1}^{t_{L1}/2} y_{ij} x_{ij}, \hat{\beta}_i^{(2)} \leftarrow \frac{2}{t_{L1}} \sum_{j=t_{L1}/2+1}^{t_{L1}} y_{ij} x_{ij} \)
3. \( \hat{\Sigma} \leftarrow (2n_{L1})^{-1} \sum_{i=1}^{n_{L1}} \left( \hat{\beta}_i^{(1)} \hat{\beta}_i^{(2)} \right) \)
4. \( \Sigma \leftarrow k \_SV \_D_{L1} \)

The dependency on the accuracy \( \epsilon \) changes based on the ground truths meta-parameters. In an ideal case when \( W \) is an orthonormal matrix (condition number one), the sample complexity is \( \mathcal{O}(d/(p_{min}^2 \epsilon^2)) \). For the worst case \( W \), it is \( \mathcal{O}(d/(p_{min}^2 \epsilon^6)) \).

**Lemma 5.1** (Learning the subspace). Suppose Assumption 1 holds, and let \( U \in \mathbb{R}^{d \times k} \) be the matrix with top \( k \) eigen vectors of matrix \( \hat{\Sigma} \in \mathbb{R}^{d \times d} \). For any failure probability \( \delta \in (0, 1) \) and accuracy \( \epsilon \in (0, 1) \), if the sample size is large enough such that

\[
n_{L1} = \Omega \left( dt_{L1}^{-1} \min \left\{ \epsilon^{-6} p_{min}^{-2}, \epsilon^{-2} \lambda_{min}^{-2} \right\} \cdot \log^2(n_{L1} d / \delta) \right),
\]

then \( 2 \leq t_{L1} < d \), we have

\[
\| (UU^\top - I) w_i \|_2 \leq \epsilon,
\]

for all \( i \in [k] \) with probability at least \( 1 - \delta \), where \( \lambda_{min} \) is the smallest non-zero eigen value of \( \Sigma \).

**Time complexity:** \( \mathcal{O} \left( (n_{L1}^{-1} + n_{L1} t_{L1}) d \right) \) for computing \( \hat{\Sigma} \), and \( \mathcal{O} \left( kd^2 \right) \) for \( k \_SV \_D \) (Allen-Zhu & Li, 2016).

5.2. Clustering

Once we have the subspace, we can efficiently cluster any task associated with \( t_{H} = \Omega(\sqrt{k}) \) samples. In the following, the matrix \( H \in \mathbb{R}^{n \times n} \) estimates the distance between the parameters in the projected \( k \)-dimensional space. If there is no error in \( U \), then \( \mathbb{E}[H_{i,j}] \geq \Omega(\Delta^2) \) if \( i \) and \( j \) are from different components, and zero otherwise. Any clustering algorithm can be applied treating \( H \) as a distance matrix.

**Algorithm 3** Clustering and estimation

**Input:** data \( D_{H} = \{(x_{i,j}, y_{i,j})\}_{i \in [n_{H}], j \in [t_{H}]} \), \( 2L \leq t_{H} \), \( k \in \mathbb{N}, L \in \mathbb{N}, U \in \mathbb{R}^{d \times k} \)

**compute** for all \( \ell \in [L] \) and \( i \in [n_{H}] \)

\[
\hat{\beta}_i^{(H)} \leftarrow (2L / t_{H}) \sum_{j=\ell-1}^{\ell+L} (y_{i,j} x_{i,j})
\]

compute for all \( \ell \in [L] \) and \( (i, j) \in [n_{H}] \times [n_{H}] \)

\[
H_{i,j}^{(H)} \leftarrow (\hat{\beta}_i^{(H)} - \hat{\beta}_j^{(H)})^\top U^\top (\hat{\beta}_i^{(H)} + \hat{\beta}_j^{(H)} - \hat{\beta}_i^{(H+L)} - \hat{\beta}_j^{(H-L)})
\]

compute for all \( (i, j) \in [n_{H}] \times [n_{H}] \)

\[
H_{i,j} \leftarrow \text{median} \left\{ H_{i,j}^{(H)} \right\}_{H \in [L]}
\]

Cluster \( D_{H} \) using \( H \) and return its partition \( \{ C_{\ell} \}_{\ell \in [k]} \). For all \( \ell \in [k] \)

\[
\bar{w}_\ell \leftarrow \frac{t_{H} |C_{\ell}|^{-1} \sum_{i \in C_{\ell} \in [t_{H}]} y_{i,j} x_{i,j}}{t_{H} |C_{\ell}|^{-1} \sum_{i \in C_{\ell} \in [t_{H}]} (y_{i,j} - x_{i,j} \_w_\ell)^2}
\]

output \( \{ C_{\ell}, \bar{w}_{\ell}, \_\ell \} \}_{\ell \in [k]} \)

This is inspired by (Vempala & Wang, 2004), where clustering mixture of Gaussians is studied. One might wonder if it is possible to apply their clustering approach to \( \hat{\beta}_i^{(H)} \). This approach fails as it crucially relies on the
Meta-learning for Mixed Linear Regression

fact that \( \|x - \mu\|_2 = \sqrt{k} + O(1) \) with high probability for \( x \sim N(0, I_k) \). Under our linear regression setting, \( \|y x - \beta\|_2 \) does not concentrate. We instead propose median of estimates, to get the desired \( t_H = \Omega(\sqrt{k}) \) sufficient condition.

Lemma 5.2 (Clustering and initial parameter estimation). Under Assumption 1, and given an orthonormal matrix \( U \in \mathbb{R}^{d \times k} \) satisfying (8) with any \( \epsilon \in (0, \Delta/4) \), Algorithm 3 correctly clusters all tasks with \( t_H = \Omega(\Delta^{-2} \sqrt{k} \log(n_H/\delta)) \) with probability at least \( 1 - \delta \), \( \forall \delta \in (0, 1) \). Further, if

\[
n_H = \Omega \left( \frac{k \log(k/\delta)}{t_H r_i^2 p_{\min}} \right),
\]

for any \( \epsilon > 0 \), with probability at least \( 1 - \delta \),

\[
\|U^T (\tilde{w}_i - w_i)\|^2_2 \leq \epsilon, \quad (10a)
\]

\[
|\tilde{r}_i^2 - r_i^2| \leq \frac{\epsilon}{\sqrt{k}} r_i^2, \quad (10b)
\]

where \( r_i^2 := (s_i^2 + \|\tilde{w}_i - w_i\|_2^2) \) for all \( i \in [k] \).

Time complexity: It takes \( O(n_H d t_H + n_H d k) \) time to compute \( \{U^T \beta_i^{(t)}\}_{i \in [n_H], t \in [L]} \). Then by using matrix multiplication, it takes \( O(n_H^2 d k^{-2}) \) time to compute the matrix \( H \), and the single linkage clustering algorithm takes \( O(n_H^2) \) time (Sibson, 1973).

Experiments. We set \( d = 8k \), \( p = 1_k / k \), \( s = 1_k \), and \( \mathcal{P}_x \) and \( \mathcal{P}_s \) are standard Gaussian distributions. Given an estimated subspace with estimation error \( \sim 0.1 \), the clustering step is performed with \( n_H = \max \{k^{3/2}, 256\} \) tasks for various \( t_H \). The minimum \( t_H \) such that the clustering accuracy is above 99% for at least \( 1 - \delta \) fraction of 10 random trials is denoted by \( t_{\min}(1 - \delta) \). Figure 3 and Table 3 illustrate the dependence of \( k \) on \( t_{\min}(0.5) \), and \( t_{\min}(0.9) \). More experimental results are provided in Appendix E.

5.3. Classification

Once we have \( \{\tilde{w}_i\}_{i=1}^k \) from the clustering step, we can efficiently classify any task with \( t_{L_2} = \Omega(\log k) \) samples, and an extra \( \log n_{L_2} \) samples are necessary to apply the union bound. This allows us to use the light samples, in order to refine the clusters estimated with heavy samples. This separation allows us to achieve the desired sample complexity on light tasks \( (t_{L_2} = \Omega(\Delta^{-2} \log d), n_{L_2} t_{L_2} p_{\min} = \Omega(\epsilon^{-2}) \) ), and heavy tasks \( (t_H = \Omega(\Delta^{-2} \sqrt{k}), n_H t_H p_{\min} = \Omega(\Delta^{-2} \sqrt{k}) \) ).

In the following, we use Least_Squares() routine that outputs the least-squares estimate of all the examples in each cluster. Once each cluster has \( O(d) \) samples, we can accurately estimate the meta-parameters.

Algorithm 4 Classification and estimation

```
Input: data \( \mathcal{D}_{L_2} = \{(x_{i,j}, y_{i,j})\}_{i \in [n_{L_2}], j \in [p_{L_2}]} \),
\{c_\ell, \tilde{w}_\ell, \tilde{r}_\ell^2\}_{\ell \in [k]}

compute for all \( i \in [n_{L_2}] \)

\[ h_i \leftarrow \min_{\ell \in [k]} \frac{1}{2 r_i^2} \sum_{j \in [L]} (y_{i,j} - x_{i,j}^T \tilde{w}_\ell)^2 + t_{L_2} \log \tilde{r}_\ell \]

\[ C_{h_i} \leftarrow C_{h_i} \cup \{(x_{i,j}, y_{i,j})\}_{j=1}^{t_{L_2}} \]

\[ \tilde{w}_\ell \leftarrow \text{Least_Squares}(C_{\ell}) \]

\[ s_\ell^2 \leftarrow (t_{L_2} |C_{\ell}| - d)^{-1} \sum_{i \in C_{\ell}} (y_{i,j} - x_{i,j}^T \tilde{w}_\ell)^2 \]

\[ \hat{p}_\ell \leftarrow |C_{\ell}| / n_{L_2} \]

output \( \{c_\ell, \tilde{w}_\ell, s_\ell^2, \hat{p}_\ell\}_{\ell=1}^k \)
```

Lemma 5.3 (Refined parameter estimation via classification). Under Assumption 1 and given estimated parameters \( \tilde{w}_\ell, \tilde{r}_\ell \), satisfying \( \|\tilde{w}_i - w_i\|_2 \leq \Delta/10, (1 - \Delta^2/50) \tilde{r}_i^2 \leq s_i^2 + \|\tilde{w}_i - w_i\|_2^2 \leq (1 + \Delta^2/50) \tilde{r}_i^2 \) for all \( i \in [k] \) and \( n_{L_2} \) task with \( t_{L_2} = \Omega(\log(k n_{L_2}/\delta)/\Delta^2) \) examples per task, with probability \( 1 - \delta \), Algorithm 4 correctly classifies all the \( n_{L_2} \) tasks. Further, for any \( 0 < \epsilon \leq 1 \) if

\[
n_{L_2} = \Omega \left( \frac{d \log^2(k/\delta)}{t_{L_2} p_{\min} \epsilon^2} \right),
\]

the following holds for all \( i \in [k] \),

\[
\|\tilde{w}_i - w_i\|_2 \leq \epsilon s_i, \quad (12a)
\]

\[
|s_i^2 - \tilde{s}_i^2| \leq \epsilon \sqrt{d} s_i^2, \quad \text{and} \quad (12b)
\]

\[
|\hat{p}_i - p_i| \leq \epsilon \sqrt{t_{L_2}/d} p_i. \quad (12c)
\]

Time complexity: Computing \( \{h_i\}_{i \in [n_{L_2}]} \) takes \( O(n_{L_2} d k) \) time, and least square estimation takes \( O(n_{L_2} d e^{-1}) \) time.
Table 1. Sample complexity for previous work in MLR to achieve small constant error on parameters recovery of the mixed linear regression problem. We ignore the constants and poly log factors. Let $n$, $d$, and $k$ denote the number of samples, the dimension of the data points, and the number of clusters, respectively. (Yi et al., 2016) and (Chaganty & Liang, 2013) requires $\sigma_k$, the $k$-th singular value of some moment matrix. (Sedghi et al., 2016) requires $s_{\text{min}}$, the $k$-th singular value of the matrix of the regression vectors. Note that $1/s_{\text{min}}$ and $1/\sigma_k$ can be infinite even when $\Delta > 0$. (Zhong et al., 2016) algorithm requires $\Delta_{\text{max}}/\Delta_{\text{min}} = O(1)$ and some spectral properties.

<table>
<thead>
<tr>
<th>References</th>
<th>Noise</th>
<th>$# \text{samples}$ $n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Chaganty &amp; Liang, 2013)</td>
<td>YES</td>
<td>$d^6 \cdot \text{poly}(k, 1/\sigma_k)$</td>
</tr>
<tr>
<td>(Yi et al., 2016)</td>
<td>NO</td>
<td>$d \cdot \text{poly}(k, 1/\Delta, 1/\sigma_k)$</td>
</tr>
<tr>
<td>(Zhong et al., 2016)</td>
<td>NO</td>
<td>$d \cdot \exp(k \log(k \log d))$</td>
</tr>
<tr>
<td>(Sedghi et al., 2016)</td>
<td>YES</td>
<td>$d^4 \cdot \text{poly}(k, 1/s_{\text{min}})$</td>
</tr>
<tr>
<td>(Li &amp; Liang, 2018)</td>
<td>NO</td>
<td>$d \cdot \text{poly}(k/\Delta) + \exp(k^2 \log(k/\Delta))$</td>
</tr>
<tr>
<td>(Chen et al., 2020)</td>
<td>NO</td>
<td>$d \cdot \exp(\sqrt{k}) \cdot \text{poly}(1/\Delta)$</td>
</tr>
</tbody>
</table>

6. Related Work

Meta-learning linear models have been studied in two contexts: mixed linear regression and multi-task learning.  

**Mixed Linear Regression (MLR).** When each task has only one sample, (i.e. $t_i = 1$), the problem has been widely studied. Prior work in MLR are summarized in Table 1. We emphasize that the sample and time complexity of all the previous work either has a super polynomial dependency on $k$ (specifically at least $\exp(\sqrt{k})$) as in (Zhong et al., 2016; Li & Liang, 2018; Chen et al., 2020), or depends on the inverse of the $k$-th singular value of some moment matrix as in (Chaganty & Liang, 2013; Yi et al., 2016; Sedghi et al., 2016), which can be infinite. (Chen et al., 2020) cannot achieve vanishing error when there is noise.

**Multi-task learning.** (Baxter, 2000; Ando & Zhang, 2005; Rish et al., 2008; Orlitsky, 2005) address a similar problem of finding an unknown $k$-dimensional subspace, where all tasks can be accurately solved. The main difference is that all tasks have the same number of examples, and the performance is evaluated on the observed tasks used in training. Typical approaches use trace-norm to encourage low-rank solutions of the matrix $[\hat{\beta}_1, \ldots, \hat{\beta}_n] \in \mathbb{R}^{d \times n}$. This is posed as a convex program (Argyriou et al., 2008; Harchaoui et al., 2012; Amit et al., 2007; Pontil & Maurer, 2013).

Closer to our work is the streaming setting where $n$ tasks are arriving in an online fashion and one can choose how many examples to collect for each. (Balcan et al., 2015) provides an online algorithm using a memory of size only $O(kn + kd)$, but requires some tasks to have $t_i = \Omega(\sqrt{k}/\epsilon^2)$ examples. In comparison, we only need $t_H = \tilde{O}(\sqrt{k})$ but use $O(d^2 + kn)$ memory. (Bullins et al., 2019) also use only small memory, but requires $\tilde{O}(d^2)$ total samples to perform the *subspace estimation* under the setting studied in this paper.

**Empirical Bayes/Population of parameters.** A simple canonical setting of probabilistic meta-learning is when $\mathbb{P}_{\phi_i}$ is a univariate distribution (e.g. Gaussian, Bernoulli) and $\phi_i$ is the parameter of the distribution (e.g. Gaussian mean, success probability). Several related questions have been studied. In some cases, one might be interested in just learning the prior distribution $\mathbb{P}_{\phi}(\phi)$ or the set of $\phi_i$’s. For example, if we assume each student’s score of one particular exam $x_i$ is a binomial random variable with mean $\phi_i$ (true score), given the scores of the students in a class, an ETS statistician (Lord, 1969) might want to learn the distribution of their true score $\phi_i$’s. Surprisingly, the minimax rate on estimating the prior distribution $\mathbb{P}_{\phi}(\phi)$ was not known until very recently (Tian et al., 2017; Vinayak et al., 2019) even in the most basic setting where $\mathbb{P}_{\phi_i}(x)$ is Binomial.

In some cases, similar to the goal of meta-learning, one might want to accurately estimate the parameter of the new task $\phi_{\text{new}}$ given the new data $x_{\text{new}}$, perhaps by leveraging an estimation of the prior $\mathbb{P}_{\phi}(\phi)$. This has been studied for decades under the **empirical bayes** framework in statistics (see, e.g. the book by Efron (Efron, 2012) for an introduction of the field).

7. Discussion

We investigate how we can meta-learn when we have multiple tasks but each with a small number of labelled examples. This is also known as a few-shot supervised learning setting. When each task is a linear regression, we propose a novel spectral approach and show that we can leverage past experience on small data tasks to accurately learn the meta-parameters and predict new tasks.

When each task is a logistic regression coming from a mixture model, then our algorithm can be applied seamlessly. However, the notion of separation $\Delta = \min_{i \neq j} \|w_i - w_j\|_2$ does not capture the dependence on the statistical complexity. Identifying the appropriate notion of complexity on the groundtruths meta-parameters is an interesting research question.
The subspace estimation algorithm requires a total number of $\Omega(dk^2)$ examples. It is worth understanding whether this is also necessary.

Compared to online algorithms for multi-task learning, whose focus is in using small memory, our approach achieves better sample complexity at the expense of requiring a larger memory scaling as $O(d^2 + kn)$; this is larger than the $O(kd + kn)$ memory required by (Balcan et al., 2015). An interesting research direction is to achieve the sample complexity of our approach with only $O(kd + kn)$ memory.

Handling the setting where $P_x$ has different covariances in different tasks is a challenging problem. There does not seem to exist an unbiased estimator for $W$. Nevertheless, (Li & Liang, 2018) study the $t = 1$ case in this setting and come up with an exponential time algorithm. Studying this general setting and coming up with a polynomial time algorithm for meta-learning in a data constrained setting is an interesting direction.

Our clustering algorithm requires the existence of medium data tasks with $t_H = \Omega(\sqrt{k})$ examples per task. It is worth investigating whether there exists a polynomial time and sample complexity algorithms that learns with $t_H = o(\sqrt{k})$. We conjecture that with the techniques developed in the robust clustering literature (Diakonikolas et al., 2018; Hopkins & Li, 2018; Kothari et al., 2018), it is possible to learn with $t_H = o(\sqrt{k})$ in the expense of larger $n_H$, and higher computation complexity. For a lower bound perspective, it is worth understanding the information theoretic trade-off between $t_H$ and $n_H$ when $t_H = o(\sqrt{k})$.

8. Acknowledgement

Sham Kakade acknowledges funding from the Washington Research Foundation for Innovation in Data-intensive Discovery, and the NSF Awards CCF-1637360, CCF-1703574, and CCF-1740551. Sewoong Oh acknowledges funding from the Google Faculty Research Award, Intel Future Wireless Systems Research Award, NSF awards CCF-1927712, CCF-1705007, IIS-1929955, and CNS-2002664.

References


Meta-learning for Mixed Linear Regression


