Multi-Stage Multi-Task Learning with Reduced Rank

Lei Han[1] **and Yu Zhang**[2*]
[1]Department of Statistics, Rutgers University
[2]Department of Computer Science and Engineering, Hong Kong University of Science and Technology
[1]lhan@stat.rutgers.edu, leihan.cs@gmail.com; [2]yu.zhang.ust@gmail.com

Abstract

Multi-task learning (MTL) seeks to improve the generalization performance by sharing information among multiple tasks. Many existing MTL approaches aim to learn the low-rank structure on the weight matrix, which stores the model parameters of all tasks, to achieve task sharing, and as a consequence the trace norm regularization is widely used in the MTL literature. A major limitation of these approaches based on trace norm regularization is that all the singular values of the weight matrix are penalized simultaneously, leading to impaired estimation on recovering the larger singular values in the weight matrix. To address the issue, we propose a Reduced rAnk MUlti-Stage multi-tAsk learning (RAMUSA) method based on the recently proposed capped norms. Different from existing trace-norm-based MTL approaches which minimize the sum of all the singular values, the RAMUSA method uses a capped trace norm regularizer to minimize only the singular values smaller than some threshold. Due to the non-convexity of the capped trace norm, we develop a simple but well guaranteed multi-stage algorithm to learn the weight matrix iteratively. We theoretically prove that the estimation error at each stage in the proposed algorithm shrinks and finally achieves a lower upper-bound as the number of stages becomes large enough. Empirical studies on synthetic and real-world datasets demonstrate the effectiveness of the RAMUSA method in comparison with the state-of-the-art methods.

Introduction

Multi-task learning (MTL) (Caruana 1997) seeks to improve the generalization performance of multiple learning tasks by sharing common information among those tasks. Many MTL methods have been proposed by learning common feature representations (Caruana 1997; Argyriou, Evgeniou, and Pontil 2008; Liu, Ji, and Ye 2009; Zhang, Yeung, and Xu 2010), learning task groups (Han and Zhang 2015a), learning task relations (Zhang and Yeung 2010a; Zhang 2013; Zhang and Yeung 2014), utilizing or learning tree structure among tasks (Han et al. 2014; Han and Zhang 2015b), and so on. MTL has been successfully applied to a wide range of applications including medical risk evaluation (Caruana, Baluja, and Mitchell 1996), image annotation (Fan, Gao, and Luo 2008), speech recognition (Parameswaran and Weinberger 2010), age estimation (Zhang and Yeung 2010b), disease progression predication (Zhou et al. 2011), and so on. One common assumption in MTL is that all the learning tasks share some common structure including, for example, hidden units in neural networks, probabilistic priors in Bayesian model (Zhang, Ghahramani, and Yang 2005; Yu, Tresp, and Schwaighofer 2005), and feature representations (Argyriou, Evgeniou, and Pontil 2008; Liu, Ji, and Ye 2009; Zhang, Yeung, and Xu 2010).

In many real-world MTL problems, challenge arises with high-dimensional data. To address this issue, several approaches (Chen, Zhou, and Ye 2011; Chen, Liu, and Ye 2012) have been proposed to learn the low-rank structure on the weight matrix which stores the model parameters of all tasks. Then the objective functions are usually formulated by minimizing the empirical loss as well as the rank of the weight matrix, which is also known as the reduced rank regression problem (Reinsel and Velu 1998) if the learning tasks solve regression problems. Unfortunately, the rank minimization problem is NP-hard due to the combinatorial nature of the rank function. A commonly used approach to overcome the computational infeasibility is to resort to minimizing the trace norm, the sum of the singular values of a matrix, which is shown as the tightest convex lower bound of the rank function (Recht, Fazel, and Parrilo 2010). As a consequence, the trace norm regularization has been widely used in MTL literature and empirical results have shown that the trace norm regularization leads to improved performance.

For a matrix, its rank is determined by the number of its singular values being 0. So in order to obtain low-rank matrices, the small singular values should be penalized to approach zero but the large ones can take any positive value. In other words, to achieve low-rank, only part of the singular values but not all of them should be penalized. Unfortunately, all the existing MTL methods to seek for the low-rank structure use the trace norm regularization to penalize all the singular values of the weight matrix simultaneously, and thus the estimation may not recover the underlying low-rank structure well, leading to the suboptimal performance.

Recently, the capped norms (Zhang 2009; 2010; 2011; Gong, Ye, and Zhang 2012; Sun, Xiang, and Ye 2013) have gained the popularity since those norms are capable

*Both authors contributed equally.

of penalizing part of the parameters lower than a threshold. For learning low-rank matrices, the capped trace norm has been used to improve the robust principal component analysis (Sun, Xiang, and Ye 2013). Moreover, the truncated trace norm, which is related to the capped trace norm, has been used for matrix completion (Zhang et al. 2012; Hu et al. 2013). However, to our knowledge, no effort has been made to learn a better low-rank structure for the weight matrix in MTL problem via the capped trace norm.

In this paper, we aim to fill this gap by investigating the use of the capped trace norm in MTL problems. Specifically, we propose a Reduced rAnk MUlti-Stage multi-tAsk learning (RAMUSA) method, which uses the capped trace norm regularizer to only penalize the singular values smaller than a threshold. Similar to most of the capped norms, the capped trace norm regularizer is non-convex. In order to solve the resulting non-convex optimization problem, we develop a simple and efficient multi-stage algorithm to learn the weight matrix iteratively. We further show that at each stage of the proposed algorithm, the subproblem reduces to a problem regularized with the truncated trace norm (Zhang et al. 2012; Hu et al. 2013), and we use an alternating optimization method to solve the subproblems in a way similar to (Zhang et al. 2012; Hu et al. 2013). Moreover, we theoretically prove that for any initial value of the weight matrix, the parameter estimation error shrinks after each learning stage and achieves a lower upper-bound when the number of learning stage becomes large enough, if the threshold constant is appropriately chosen. Such theoretical results provide important guarantees for the proposed multi-stage algorithm to achieve good estimation performance. For empirical studies, we first evaluate the proposed RAMUSA method on synthetic data, and the experimental results well match the properties revealed in the theoretical analysis. Then we evaluate on five real-world datasets with distinct application scenarios. The experimental results on those datasets demonstrate the effectiveness of the proposed RAMUSA method.

The RAMUSA Model

For clear presentation, we list notations frequently used in Table 1. Suppose we have m learning tasks and the training data for the i-th task is denoted by $(\mathbf{X}_i, \mathbf{y}_i)$ where $\mathbf{X}_i \in \mathbb{R}^{n_i \times d}$ is the data matrix with n_i training samples stored in the rows, d is the feature dimensionality, and $\mathbf{y}_i \in \mathbb{R}^{n_i}$ is the vector of the labels corresponding to the n_i training samples in $\mathbf{X}_i$. If the values in $\mathbf{y}_i$ are continuous, the i-th task is a regression problem and otherwise a classification problem. The linear function for the i-th task is defined as $f_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x}$, where $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_m] \in \mathbb{R}^{d\times m}$ is the weight matrix or parameter matrix. In order to estimate the low-rank task structure, the widely used trace norm regularization solves the objective function $\min_{\mathbf{W}} L(\mathbf{W}) + \lambda\|\mathbf{W}\|_*$, where $L(\mathbf{W})$ is the empirical loss on the training data and λ is a positive regularization parameter. Since the trace norm penalizes all the singular values of the weight matrix simultaneously, the low-rank structure may not be well estimated. In order to achieve a better recovery of the low-rank structure, the capped trace norm only penalizes the sum of some small singular values:

$$\sum_{i=1}^{R} \min(\sigma_i(\mathbf{W}), \tau), \tag{1}$$

where τ is a threshold. Note that with the threshold τ, only the singular values smaller than τ contribute to the capped trace norm regularizer in Eq. (1), while the singular values larger than τ are capped. This is one reason that the capped trace norm regularizer could provide a better approximation of the rank function than the trace norm. Based on the capped trace norm defined in Eq. (1), the objective function of the RAMUSA model is formulated as

$$\min_{\mathbf{W}} \; L(\mathbf{W}) + \lambda \sum_{i=1}^{R} \min(\sigma_i(\mathbf{W}), \tau). \tag{2}$$

When $\tau = 0$, problem (2) reduces to the empirical risk minimization over the m tasks, and when $\tau \to \infty$, problem (2) becomes the trace norm regularization problem. So the capped trace norm regularization is a generalization of the trace norm regularization. In this paper, we focus on the square loss, i.e., $L(\mathbf{W}) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{n_i}\|\mathbf{y}_i - \mathbf{X}_i\mathbf{w}_i\|_2^2$. Other loss functions, e.g., the hinge loss, can be handled in a similar way. Obviously problem (2) is non-convex due to the capped trace norm regularizer, and thus it is not easy to solve. In the next section, we show how to solve it.

Table 1: Notations used in this paper.

Notation	Description
$\mathbf{w} \in \mathbb{R}^m$	A vector $\mathbf{w}$ with length m.
$\mathbf{W} \in \mathbb{R}^{d\times m}$	A matrix $\mathbf{W}$ with size $d \times m$.
$\mathbf{w}^j, \mathbf{w}_i, w_{ji}$	The j-th row, i-th column, and (j, i)-th element of matrix $\mathbf{W}$.
$\mathbf{I}_a$	An $a \times a$ identity matrix.
$\|\cdot\|_2$	The ℓ_2 norm for any vector.
$\|\cdot\|_F$	The matrix Frobenius norm.
$\langle\cdot,\cdot\rangle$	The inner product.
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with mean μ and variance σ^2.
$\mathbb{N}_m$	The index set $\{1, 2, \cdots, m\}$.
$\mathrm{tr}(\cdot)$	The trace operator.
$\{\sigma_i(\mathbf{W})\}_{i=1}^R$	The set of non-increasing ordered singular values of matrix $\mathbf{W} \in \mathbb{R}^{d\times m}$, where $R = \min(d, m)$.
$\|\mathbf{W}\|_*$	The matrix trace norm, which is $\sum_{i=1}^R \sigma_i(\mathbf{W})$.
$\|\mathbf{W}\|_{r-}$	$\sum_{i=r+1}^R \sigma_i(\mathbf{W})$ for any non-negative integer r ($r \le R$).
$\|\mathbf{W}\|_{r+}$	$\sum_{i=1}^r \sigma_i(\mathbf{W})$ for any non-negative integer r ($r \le R$).
$\mathbb{I}(\cdot)$	The indicator function.

The Multi-Stage Algorithm for RAMUSA

In this section, we propose a multi-stage algorithm to solve problem (2).

The multi-stage algorithm is an instance of the majorization-minimization (MM) algorithm (Hunter and Lange 2004), an iterative algorithm, which in each iteration constructs a surrogate function as the upper-bound of the original objective function based on the solution of the previous iteration and then minimizes the surrogate function instead. The MM algorithm is guaranteed to converge to a local optimum and so is the proposed multi-stage algorithm.

One benefit of the MM algorithm is that the constructed surrogate function is usually easier to be solved than the original optimization problem, leading to a more efficient solution. The detailed algorithm is shown in Algorithm 1.

For our multi-stage algorithm, the new regularizer $\|\cdot\|_{r^-}$ in step 4 of Algorithm (1) is a surrogate function of the capped trace norm regularizer by omitting some constant with r defined based on the solution of the last stage $l-1$ ($l \geq 2$). The regularizer $\|\cdot\|_{r^-}$ in step 4 of Algorithm (1) is essentially the truncated trace norm introduced in (Zhang et al. 2012; Hu et al. 2013), which only penalizes the smallest $R-r$ singular values of $\mathbf{W}$. The difference between our work and (Zhang et al. 2012; Hu et al. 2013) is that r needs to be pre-defined in (Zhang et al. 2012; Hu et al. 2013) while in our case, r is obtained from the previous estimation on $\mathbf{W}$ at each stage of Algorithm 1.

Algorithm 1 Multi-Stage Algorithm for the RAMUSA Model

Input: $\mathbf{X}$, $\mathbf{Y}$, λ, τ;
Output: $\hat{\mathbf{W}}$;
1: $R := \min(d, m)$;
2: $r := 0$;
3: **for** $l = 1, 2, \cdots, \mathcal{L}$ **do**
4: Solve problem $\min_{\mathbf{W}} \{L(\mathbf{W}) + \lambda\|\mathbf{W}\|_{r^-}\}$.
5: $r := \sum_{i=1}^{R} \mathbb{I}(\sigma_i(\hat{\mathbf{W}}^{(l)}) \geq \tau)$;
6: **end for**

As we will see later, the problem in step 4 of Algorithm 1 is easier to be optimized than problem (2), which is one computational advantage of our multi-stage algorithm. Moreover, the problem in step 4 at each stage tends to shrink only small singular values of $\mathbf{W}$. Hence, the RAMUSA method can overcome the limitation of the trace norm regularization by adaptively regularizing the singular values of $\mathbf{W}$ according to the solution obtained from the last stage. In the first iteration, the problem in step 4 is just the trace norm regularization problem since r is initialized to 0. When $l \geq 2$, the operator $\|\mathbf{W}\|_{r^-}$ is non-convex and so is the problem in step 4. Therefore, the key step in Algorithm 1 is to solve the problem in step 4 efficiently when $l \geq 2$.

Next, we use an alternating optimization method to solve the problem in step 4 efficiently. Before presenting the detailed algorithm, we first introduce an useful lemma.

Lemma 1 (Zhang et al. 2012; Hu et al. 2013) *Suppose $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the singular value decomposition (SVD) of $\mathbf{W}$, where $\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_d) \in \mathbb{R}^{d\times d}$ and $\mathbf{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_d) \in \mathbb{R}^{m\times m}$ are unitary matrices, and $\Sigma \in \mathbb{R}^{d\times m}$. Define $r = \sum_{i=1}^{R} \mathbb{I}(\sigma_i(\hat{\mathbf{W}}^{(l)}) \geq \tau)$ and let $\hat{\mathbf{A}} = (\mathbf{u}_1, \cdots, \mathbf{u}_r)^T, \hat{\mathbf{B}} = (\mathbf{v}_1, \cdots, \mathbf{v}_r)^T$, then $\|\mathbf{W}\|_{r^+} = \max_{\mathbf{A}\in\mathcal{C}_{r,d}, \mathbf{B}\in\mathcal{C}_{r,m}} \mathrm{tr}\left(\mathbf{A}\mathbf{W}\mathbf{B}^T\right) = \mathrm{tr}\left(\hat{\mathbf{A}}\mathbf{W}\hat{\mathbf{B}}^T\right)$, where $\mathcal{C}_{a,b} = \{\mathbf{A} | \mathbf{A} \in \mathbb{R}^{a\times b}, \mathbf{A}\mathbf{A}^T = \mathbf{I}_a\}$.*

Lemma 1 shows that an lower bound for the sum of largest singular values of a matrix, which brings a reformulation for the capped trace norm and facilitates the optimization of the problem in step 4 of Algorithm 1. Based on Lemma 1, we can reformulate the problem in step 4 of Algorithm 1 as

$$\min_{\mathbf{W},\mathbf{A},\mathbf{B}} \left\{ L(\mathbf{W}) + \lambda\|\mathbf{W}\|_* - \lambda \mathrm{tr}\left(\mathbf{A}\mathbf{W}\mathbf{B}^T\right) \right\}. \tag{3}$$

Algorithm 2 The Alternating Optimization Method for Solving Problem (3)

Input: $\mathbf{X}$, $\mathbf{Y}$, λ, l, r, $\hat{\mathbf{W}}_0$, N;
Output: $\hat{\mathbf{W}}^{(l)}$;
1: **for** $t = 0, 1, \cdots, N-1$ **do**
2: Compute $\hat{\mathbf{A}}_t$, $\hat{\mathbf{B}}_t$ according to Lemma 1;
3: Compute $\hat{\mathbf{W}}_{t+1}$ by solving the problem in Eq. (4);
4: **end for**
5: $\hat{\mathbf{W}}^{(l)} = \hat{\mathbf{W}}_t$;

Problem (3) can be solved via an alternating way when $l \geq 2$ and the detailed procedure is shown in Algorithm 2. At the t-th iteration, we first fix the value of $\mathbf{W}$ and compute $\hat{\mathbf{A}}_t$ and $\hat{\mathbf{B}}_t$ according to Lemma 1, and then update $\mathbf{W}$ with fixed $\hat{\mathbf{A}}_t$ and $\hat{\mathbf{B}}_t$ by solving the following problem as

$$\hat{\mathbf{W}}_{t+1} = \arg\min_{\mathbf{W}} \left\{ L(\mathbf{W}) + \lambda\|\mathbf{W}\|_* - \lambda \mathrm{tr}\left(\hat{\mathbf{A}}_t\mathbf{W}\hat{\mathbf{B}}_t^T\right) \right\}. \tag{4}$$

Since all the terms $L(\mathbf{W})$, $\|\mathbf{W}\|_*$ and $-\mathrm{tr}\left(\hat{\mathbf{A}}_t\mathbf{W}\hat{\mathbf{B}}_t^T\right)$ are convex, problem (4) is convex. We use the FISTA method (Beck and Teboulle 2009) to solve problem (4). The FISTA method solves problems as

$$\min_{\mathbf{W}} f(\mathbf{W}) + g(\mathbf{W}), \tag{5}$$

where function $f(\cdot)$ is convex and smooth, and function $g(\cdot)$ is convex but possibly non-smooth. In order to solve problem (4), we can define $g(\mathbf{W}) = \lambda\|\mathbf{W}\|_*$ and $f(\mathbf{W}) = L(\mathbf{W}) - \lambda \mathrm{tr}\left(\hat{\mathbf{A}}_t\mathbf{W}(\hat{\mathbf{B}}_t)^T\right)$. The FISTA method solves the proximal function of problem (5) at $\hat{\mathbf{W}}_k$ as

$$\begin{aligned}\min_{\mathbf{W}} Q(\mathbf{W}, \hat{\mathbf{W}}_k) =& f(\hat{\mathbf{W}}_k) + \langle \mathbf{W} - \hat{\mathbf{W}}_k, \nabla f(\hat{\mathbf{W}}_k)\rangle \\ &+ \frac{\rho}{2}\|\mathbf{W} - \hat{\mathbf{W}}_k\|_F^2 + g(\mathbf{W}),\end{aligned}$$

which can be rewritten as

$$\hat{\mathbf{W}}_{k+1} = \arg\min_{\mathbf{W}} \frac{\rho}{2}\|\mathbf{W} - (\hat{\mathbf{W}}_k - \frac{1}{\rho}\nabla f(\hat{\mathbf{W}}_k))\|_F^2 + g(\mathbf{W}), \tag{6}$$

where ρ is the Lipschitz constant that can be determined w.r.t. (Beck and Teboulle 2009) and $\nabla f(\hat{\mathbf{W}}_k)$ can be computed as

$$\nabla f(\hat{\mathbf{W}}_k) = \nabla L(\hat{\mathbf{W}}_k) - \lambda\hat{\mathbf{A}}_t^T\mathbf{B}_t. \tag{7}$$

Algorithm 3 FISTA Algorithm for Solving Problem (4)

Input: $\mathbf{X}$, $\mathbf{Y}$, λ, $\hat{\mathbf{A}}_t$, $\hat{\mathbf{B}}_t$, $\hat{\mathbf{W}}_0$, N', $\theta_0 = 1$;
Output: The optimal solution of Eq. (4);
1: **for** $k = 0, 1, \cdots, N'-1$ **do**
2: Compute $\nabla f(\hat{\mathbf{W}}_k)$ as in Eq. (7);
3: Compute the closed-form solution of problem (6);
4: $\theta_{k+1} := \frac{1+\sqrt{1+4\theta_k^2}}{2}$;
5: $\hat{\mathbf{W}}_{k+1} := \hat{\mathbf{W}}_{k+1} + \frac{\theta_k - 1}{\theta_{k+1}}(\hat{\mathbf{W}}_{k+1} - \hat{\mathbf{W}}_k)$;
6: **end for**

Since $g(\mathbf{W}) = \lambda\|\mathbf{W}\|_*$, Problem (6) has an analytical solution, which can be obtained via the soft-thresholding operation on the singular values of $\hat{\mathbf{W}}_k - \frac{1}{\rho}\nabla f(\hat{\mathbf{W}}_k)$ according

to (Cai, Candès, and Shen 2010). The whole procedure for the FISTA algorithm is depicted in Algorithm 3.

In order to analyze the complexity of the whole algorithm, we first analyze the complexity of Algorithm 3 which is the innermost one. The main computational cost in each iteration of Algorithm 3 comes from calculating $\nabla f(\hat{\mathbf{W}}_k)$ and the SVD operation. For simplicity, we assume that all tasks have the same number of training samples. Since the gradient of the square loss needs to compute $\mathbf{X}_i^T\mathbf{X}_i$ and $\mathbf{X}_i^T\mathbf{y}_i$, which can be pre-computed and stored, $\nabla L(\hat{\mathbf{W}}_k)$ can be computed in $O(d^2m)$ time. Moreover, computing $\hat{\mathbf{A}}_t^T\mathbf{B}_t$ takes at most $O\left(dm\min(d,m)\right)$ time. The closed-form solution of problem (6) needs to do matrix SVD which will cost $O(md\min(m,d))$. In total, Algorithm 3 can be completed in $O(N'd^2m)$ where N' is the number of iterations. The time complexity of Algorithm 2 is N times higher than that of Algorithm 3, and the time complexity of Algorithm 1 is $\mathcal{L}$ times higher than that of Algorithm 2 with N and $\mathcal{L}$ as the numbers of iterations in the respective algorithms. From the experimental results, we find that both Algorithms 1 and 2 need very small numbers of iterations to converge and hence the whole algorithm to solve RAMUSA is still very efficient.

Theoretical Analysis

In this section, we show that the estimation performance of the multi-stage algorithm can be theoretically guaranteed. We first present a performance bound for the innermost convex problem (4), and then we extend this bound to show that the parameter estimation error bound of the RAMUSA method based on Algorithm 1 shrinks after each stage and can finally achieve a lower upper-bound when the number of the learning stage becomes large enough. For notational simplicity, we assume that the numbers of training samples for all the tasks are the same and denote it by n. The general case that different tasks have different training sizes can be similarly analyzed.

Setup

We assume that the ground truth for the relation between the data sample and its label is a linear function plus a Gaussian noise, which is defined as $y_{ji} = \mathbf{x}_j^{(i)}\bar{\mathbf{w}}_i + \delta_{ji}$ for $i \in \mathbb{N}_m$ and $j \in \mathbb{N}_n$, where y_{ji} is the jth element in $\mathbf{y}_i$, $\bar{\mathbf{W}} = [\bar{\mathbf{w}}_1, \ldots, \bar{\mathbf{w}}_m]$ is the true weight matrix, δ_{ji} is a Gaussian noise. Each noise δ_{ji} follows a normal distribution as $\delta_{ji} \sim \mathcal{N}(0, \sigma^2)$ and different noises are assumed to be independent of each other. For notational simplicity, we define $\mathbf{y}_i = \bar{f}_i + \boldsymbol{\delta}_i$ for $i \in \mathbb{N}_m$, where $\bar{f}_i = \mathbf{X}_i\bar{\mathbf{w}}_i$ and $\boldsymbol{\delta}_i = [\delta_{i1}, \cdots, \delta_{in}]^T \in \mathbb{R}^n$. Let $\mathcal{X} \in \mathbb{R}^{mn\times md}$ be a block-diagonal matrix with its i-th block formed by the data matrix $\mathbf{X}_i \in \mathbb{R}^{n\times d}$, $i \in \mathbb{N}_m$. Define a diagonalization operator $\mathcal{D}$ on any matrix $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_m] \in \mathbb{R}^{d\times m}$ such that $\mathcal{D}(\mathbf{W}) \in \mathbb{R}^{md\times m}$ is a block diagonal matrix with its i-th block formed by the column $\mathbf{w}_i$. Let $\bar{\mathbf{F}} = [\bar{f}_1, \cdots, \bar{f}_m]$. We define $\bar{\mathcal{F}} = \{i : \sigma_i(\bar{\mathbf{W}}) \neq 0\}$ and $\hat{\mathcal{G}} = \{i : \sigma_i(\hat{\mathbf{W}}) < \tau\}$ for an estimator $\hat{\mathbf{W}}$. For any set $\mathcal{A}$, let $\mathcal{A}^c$ be the complement set of $\mathcal{A}$, and $|\mathcal{A}|$ denotes the cardinality of $\mathcal{A}$.

We first present a basic assumption before we state the main theoretical results.

Assumption 1 *For any matrix $\Delta \in \mathbb{R}^{d\times m}$, we assume that there exist a constant*

$$\kappa = \min_{\Delta\in\mathcal{R}(s)} \frac{\|\mathcal{X}\mathcal{D}(\Delta)\|_F}{\sqrt{mn}\|\Delta\|_*} > 0, \tag{8}$$

where the restricted set $\mathcal{R}(s)$ is defined as $\mathcal{R}(s) = \left\{\Delta \in \mathbb{R}^{d\times m} : \Delta \neq 0, rank(\mathcal{Q}(\Delta)) = s \leq R\right\}$.

Assumption 1 refers to the widely used eigenvalue assumption. Similar assumptions are also used in several existing MTL works (Lounici et al. 2009; Chen, Zhou, and Ye 2011).

Estimation Error Bounds

We first present an important property for the innermost convex problem (4).[1]

Theorem 1 *Let $\hat{\mathbf{W}}$ be the optimal solution of problem (4) at any iteration t ($t = 1, \cdots, N$) in Algorithm 2 of the $(l+1)$-th stage in Algorithm 1. Let $\hat{\mathbf{W}}_\star^{(l)}$ be the optimal solution at the l-th stage. Define $r_l = \sum_{i=1}^R \mathbb{I}(\sigma_i(\hat{\mathbf{W}}_\star^{(l)}) \geq \tau)$ and note that r_l is unchanged within the $(l+1)$-th stage. For any matrix $\mathbf{W} \in \mathbb{R}^{d\times m}$, if we choose λ as $\lambda \geq \frac{2\phi}{mn}\sqrt{d+c}$, then with the probability of at least $1 - m\exp(-\frac{1}{2}(c - d\ln(1+\frac{c}{d})))$, we have*

$$\frac{1}{mn}\|\mathcal{X}\mathcal{D}(\hat{\mathbf{W}}) - \mathcal{D}(\bar{\mathbf{F}})\|_F^2 \leq \frac{1}{mn}\|\mathcal{X}\mathcal{D}(\mathbf{W}) - \mathcal{D}(\bar{\mathbf{F}})\|_F^2 + \lambda(1+\sqrt{m})\|\hat{\mathbf{W}} - \mathbf{W}\|_* + \lambda\|\hat{\mathbf{W}} - \mathbf{W}\|_{r_l^+}, \tag{9}$$

where c is some positive scalar.

Theorem 1 reveals that for any matrix $\mathbf{W} \in \mathbb{R}^{d\times m}$, the estimation error for problem (4) is upper-bounded by Eq. (9). Based on this theorem, we state the important estimation error bound of the RAMUSA model based on Algorithm 1 in the following theorem.

Theorem 2 *Let $\hat{\mathbf{W}}_\star^{(l+1)}$ be the optimal solution at the $(l+1)$-th stage. If we choose λ as in Theorem 1 and choose τ as $\tau > \frac{\lambda\sqrt{R-\bar{r}}}{\kappa^2}$, then based on Assumption 1, with probability of at least $1 - m\exp(-\frac{1}{2}(c - d\ln(1+\frac{c}{d})))$, we have*

$$\|\hat{\mathbf{W}}_\star^{(l+1)} - \bar{\mathbf{W}}\|_* \leq \left(\frac{\lambda\sqrt{R-\bar{r}}}{\tau\kappa^2}\right)^l \|\hat{\mathbf{W}}^{(0)} - \bar{\mathbf{W}}\|_* + \frac{\tau\lambda(\sqrt{\bar{r}}+1+\sqrt{m})}{\tau\kappa^2 - \lambda\sqrt{R-\bar{r}}}, \tag{10}$$

where $\bar{r}$ is the true rank of $\bar{\mathbf{W}}$, and $\hat{\mathbf{W}}^{(0)}$ is the initial weight matrix for the first stage. When $l \to \infty$, we have

$$\|\hat{\mathbf{W}}_\star^{(l+1)} - \bar{\mathbf{W}}\|_* \leq \frac{\tau\lambda(\sqrt{\bar{r}}+1+\sqrt{m})}{\tau\kappa^2 - \lambda\sqrt{R-\bar{r}}}. \tag{11}$$

[1] Due to page limit, we put all the proofs in the supplementary material (`http://www.stat.rutgers.edu/home/lhan/`).

Theorem 2 provides important estimation error bounds *in terms of the trace norm* on the difference between the estimator and the true weight matrix: (1) Eq. (10) implies that given any initial value $\hat{\mathbf{W}}^{(0)}$ for the weight matrix, the upper bound of $\|\hat{\mathbf{W}}_\star^{(l)} - \bar{\mathbf{W}}\|_*$ is shrinkable after each stage l, since $\left(\frac{\lambda\sqrt{R-\bar{r}}}{\tau\kappa^2}\right) < 1$ by choosing τ as in Theorem 2; (2) when l is large enough, we can obtain a lower upper-bound in Eq. (11), which is a constant and irrelevant with the initial value $\hat{\mathbf{W}}^{(0)}$. Under Theorem 2, the estimation performance of the multi-stage algorithm is well guaranteed even if the initial guess for the weight matrix $\hat{\mathbf{W}}^{(0)}$ is not very good.

Related Work

In (Sun, Xiang, and Ye 2013), the capped trace norm is used to improve the robust principal component analysis and the truncated trace norm is introduced in (Zhang et al. 2012; Hu et al. 2013) for the matrix completion problems. Compared to those works, our RAMUSA model has good theoretical properties. Moreover, all the previous works, using either the truncated trace norm or the capped trace norm, are for matrix completion problems, while our work is to accurately estimate the model parameters for multiple tasks via the capped trace norm.

Experiments

In this section, we conduct empirical experiments on one synthetic dataset and five real-world datasets to study the proposed RAMUSA method.

The baseline algorithms used for comparison include: (1) The ℓ_1-norm regularized single-task algorithm (Lasso) (Tibshirani 1996); (2) the ℓ_2-norm single-task ridge regression (RR) model with $\lambda\|\mathbf{W}\|_F^2$ as the regularizer; (3) the multi-task feature learning (MTFL) algorithm introduced in (Argyriou, Evgeniou, and Pontil 2008) which utilizes the trace norm as a regularizer.

Experiments on Synthetic Data

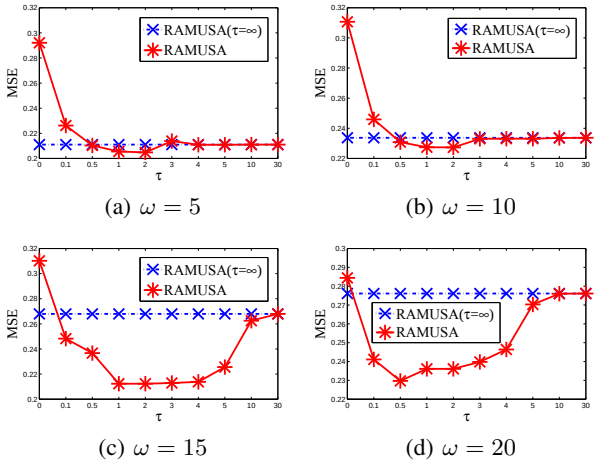
We first conduct experiments on synthetic data. We study multi-task regression problems. The number of tasks is assumed to be $m = 20$. The columns of the true weight matrix $\bar{\mathbf{W}}$, i.e. $\{\bar{\mathbf{w}}_1, \cdots, \bar{\mathbf{w}}_m\}$, are sampled from 5-dimensional Gaussian distribution with zero mean and covariance $\omega\mathrm{diag}([1, 0.64, 0.49, 0.36, 0.25])$, where ω is a weight to control the magnitude of the diagonal elements and operator $\mathrm{diag}(\cdot)$ converts a vector to a diagonal matrix. To construct low-rank weight matrix, we add 10 irrelevant features and therefore the feature dimensionality is 15. Note that a larger ω leads to larger singular values of $\bar{\mathbf{W}}$. Moreover, we assume that all the tasks have the same sample size n. For the i-th task, each column of the data matrix $\mathbf{X}_i \in \mathbb{R}^{n\times d}$ is generated from a normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I}_n)$, where $\mathbf{0}$ denotes a zero vector or matrix with appropriate size. The label $\mathbf{y}_i$ for the i-th task is generated as $\mathbf{y}_i = \mathbf{X}_i\bar{\mathbf{w}}_i + \boldsymbol{\epsilon}_i$, where $\boldsymbol{\epsilon}_i$ is a noise vector generated from $\mathcal{N}(\mathbf{0}, \mathbf{I}_n)$.

Since the true weight matrix $\bar{\mathbf{W}}$ is given in the synthetic data, the mean square error (MSE), which is defined as $\mathrm{MSE}(\mathbf{W}) = \frac{1}{mn}\sum_{i=1}^m (\mathbf{w}_i - \bar{\mathbf{w}}_i)^T\mathbf{X}_i^T\mathbf{X}_i(\mathbf{w}_i - \bar{\mathbf{w}}_i)$, is used to measure the performance of the estimation. We generate 50 and 200 samples for training and testing separately and use another 200 samples as a validation set to select the regularization parameters and hyperparameters in all the compared methods including the parameter τ in the RAMUSA method. We vary the value of ω to simulate different conditions on singular values in $\bar{\mathbf{W}}$.

Table 2 shows the average MSE of various methods over 10 simulations in terms of mean±standard deviation. From the results shown in Table 2, we have the following conclusions: (1) the RAMUSA method outperforms all other competing algorithms; (2) the multi-task learning algorithms, i.e. the MTFL and RAMUSA methods, outperform the single-task learning algorithms, i.e. the Lasso and RR models.

Table 2: Averaged MSE of various methods over 10 simulations on the synthetic data (mean±standard deviation).

ω	Lasso	RR	MTFL	RAMUSA
5	0.762±0.223	1.264±0.253	0.210±0.018	**0.205±0.021**
10	0.888±0.289	1.405±0.382	0.232±0.043	**0.227±0.051**
15	0.917±0.314	1.640±0.497	0.261±0.029	**0.212±0.040**
20	0.922±0.274	1.488±0.333	0.283±0.038	**0.230±0.037**



(a) $\omega = 5$ (b) $\omega = 10$

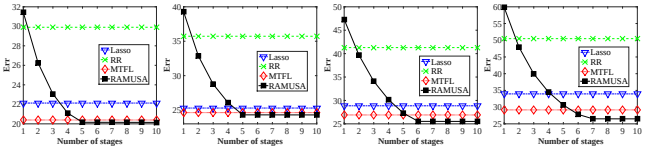
(c) $\omega = 15$ (d) $\omega = 20$

Figure 1: The performance of RAMUSA when τ changes. The performance of RAMUSA when $\tau = \infty$ (i.e. the trace norm regularization) is also plotted as the baseline.

Figure 1 shows the performance of RAMUSA when τ changes. The performance of the trace norm regularization corresponding to $\tau = \infty$ in RAMUSA is also provided as a baseline. According to Figure 1, we can find that when τ is increasing from 0, the MSE of the RAMUSA method first decreases and then increases, and when τ is large enough, the performance of RAMUSA keeps stable and approaches that of the trace norm regularization, which is accordance with the property of the RAMUSA model implied by Theorem 2.

Figure 2 plots the parameter estimation errors defined in Theorem 2, i.e. $\|\hat{\mathbf{W}}_\star^{(l)} - \bar{\mathbf{W}}\|_*$, against the number of stages l when we set $\tau = 1$. We see that the results are in line with the theoretical results revealed in Theorem 2 that the param-

eter estimation error (Err) shrinks after each stage according to Eq. (10), and the error will level off after several stages according to Eq. (11), where only 5-7 stages are needed to reach the convergence.



(a) $\omega = 5$ (b) $\omega = 10$ (c) $\omega = 15$ (d) $\omega = 20$

Figure 2: The change of Err when increasing the number of stages, where $\text{Err} = \|\hat{\mathbf{W}}_\star^{(l)} - \bar{\mathbf{W}}\|_*$.

Experiments on Real-World Data

In this section, we evaluate the empirical performance on five real-world datasets with distinct application fields, where both regression and classification problems are involved. The datasets include the School data[2], the SARCOS data[3], the Microarray data[4], the Traffic data, and the handwritten letter data[5]. The first four datasets correspond to multi-task regression problems where the normalized mean squared error (nMSE) is employed as the performance measure, but the last one, the handwritten letter dataset, is a multi-task classification problem with the classification error as the performance measure. The descriptions of those datasets, whose summary is shown in Table 3, are shown as follows:

School Data: the objective is to predict the student exam scores in different schools. Tasks correspond to schools, features are attributes for describing students, and each task has a different number of samples corresponding to students. We randomly select 10%, 20% and 30% of the samples from each task as the training set and the rest as the test set;

SARCOS Data: the problem is an inverse dynamics prediction problem for a seven degrees-of-freedom anthropomorphic robot arm, which needs to map from the feature space to seven joint torques. We randomly select 100, 200 and 300 samples to form the training set and randomly select 5000 samples to form the test set;

Microarray Data: this is a gene expression data set related to isoprenoid biosynthesis. The tasks are finding the cross-talks from the mevalonate genes to the plastidial genes. We randomly select 20% and 40% of the samples as the training set and use the rest for testing;

Traffic Data: this is to find the casual relationships from the entries to the exits in a highway traffic network, where each exit corresponds to one task and the information collected in entries is considered as the features shared by all the tasks. The settings are the same as those in the Microarray data;

Handwritten Letter Data: the goal is to discriminate between 7 pairs of letters, i.e. c/e, g/y, m/n, a/g, a/o, f/t and h/n. The features are pixel values of the handwritten letter. We randomly choose 10% and 20% of the samples as the training sets and the rest as the test set.

Table 3: Summary of the five real-world datasets.

	School	SARCOS	Microarray	Traffic	Letter
m	139	7	18	136	7
d	27	21	21	136	128
n	15362	48933	118	384	2000

Table 4: The averaged nMSE for the (1) School, (2) SARCOS, (3) Microarray, and (4) Traffic datasets, and the averaged test error (%) for the (5) Handwritten Letter data of various methods over 15 repetitions (mean±standard deviation).

	Train	Lasso	RR	MTFL	RAMUSA
(1)	10%	2.283±0.008	1.854±0.026	0.570±0.015	**0.518±0.012**
	20%	2.129±0.130	1.694±0.028	0.483±0.007	**0.458±0.005**
	30%	2.080±0.111	1.650±0.028	0.452±0.005	**0.436±0.004**
(2)	100	2.606±0.035	2.439±0.021	0.181±0.010	**0.170±0.010**
	200	2.604±0.035	2.289±0.027	0.159±0.007	**0.146±0.006**
	300	2.620±0.030	2.221±0.026	**0.136±0.003**	**0.137±0.004**
(3)	20%	0.794±0.092	0.788±0.028	0.746±0.038	**0.739±0.046**
	40%	0.700±0.043	0.715±0.039	0.680±0.025	**0.675±0.031**
(4)	20%	0.566±0.015	0.617±0.032	0.328±0.006	**0.316±0.006**
	40%	0.552±0.020	0.600±0.016	0.310±0.010	**0.300±0.008**
(5)	10%	31.74±16.30	31.22±21.22	13.45±7.50	**11.64±7.58**
	20%	31.73±17.61	31.08±20.95	11.86±6.49	**8.48±6.08**

Each setting is repeated for 15 times to test the average performance of various methods. For the parameter τ in the RAMUSA method, we choose it in a candidate set $[10^{-3}, 10^{-2}, \cdots, 10^{3}]$ via 5-fold cross validation. According to the results shown in Table 4, the multi-task learning algorithms, i.e. the MTFL and RAMUSA methods, outperform the single-task learning algorithms, i.e., the Lasso and RR methods, under all the settings, and our RAMUSA method achieves the best performance in every setting. Due to the different application scenarios in the five datasets, we think that the RAMUSA method is able to have good performance in various MTL applications.

Conclusion and Future Work

In this paper, we proposed a reduced rank multi-stage MTL approach, RAMUSA, to learn the low-rank structure contained in the weight matrix under the multi-task setting. We developed a simple multi-stage algorithm to solve the RAMUSA model where theoretical guarantees are provided for the estimation performance.

In our future work, we will extend our RAMUSA model to utilize other loss functions such as the hinge loss. Moreover, currently the threshold τ is learned via the cross validation method. In our future study, we are interested in learning τ and the weight matrix simultaneously in a principled framework.

Acknowledgment

This work is supported by NSFC (61305071, 61473087) and Nature Science Foundation of Jiangsu Provice of China (BK20141340).

[2] http://www.cs.ucl.ac.uk/staff/A.Argyriou/code/

[3] http://www.gaussianprocess.org/gpml/data/

[4] http://www.ncbi.nlm.nih.gov/pmc/articles/PMC545783/

[5] http://multitask.cs.berkeley.edu/

References

Argyriou, A.; Evgeniou, T.; and Pontil, M. 2008. Convex multi-task feature learning. *MLJ*.

Beck, A., and Teboulle, M. 2009. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*.

Cai, J.; Candès, E. J.; and Shen, Z. 2010. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*.

Caruana, R.; Baluja, S.; and Mitchell, T. 1996. Using the future to "sort out" the present: Rankprop and multitask learning for medical risk evaluation. In *NIPS*.

Caruana, R. 1997. Multitask learning. *MLJ*.

Chen, J.; Liu, J.; and Ye, J. 2012. Learning incoherent sparse and low-rank patterns from multiple tasks. *TKDD*.

Chen, J.; Zhou, J.; and Ye, J. 2011. Integrating low-rank and group-sparse structures for robust multi-task learning. In *KDD*.

Fan, J.; Gao, Y.; and Luo, H. 2008. Integrating concept ontology and multitask learning to achieve more effective classifier training for multilevel image annotation. *IEEE Transactions on Image Processing*.

Gong, P.; Ye, J.; and Zhang, C. 2012. Multi-stage multi-task feature learning. In *NIPS*.

Han, L., and Zhang, Y. 2015a. Learning multi-level task groups in multi-task learning. In *AAAI*.

Han, L., and Zhang, Y. 2015b. Learning tree structure in multi-task learning. In *KDD*.

Han, L.; Zhang, Y.; Song, G.; and Xie, K. 2014. Encoding tree sparsity in multi-task learning: A probabilistic framework. In *AAAI*.

Hu, Y.; Zhang, D.; Ye, J.; Li, X.; and He, X. 2013. Fast and accurate matrix completion via truncated nuclear norm regularization. *TPAMI*.

Hunter, D. R., and Lange, K. 2004. A tutorial on MM algorithms. *The American Statistician*.

Liu, J.; Ji, S.; and Ye, J. 2009. Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization. In *UAI*.

Lounici, K.; Pontil, M.; Tsybakov, A. B.; and van de Geer, S. 2009. Taking advantage of sparsity in multi-task learning. *COLT*.

Parameswaran, S., and Weinberger, K. Q. 2010. Large margin multi-task metric learning. In *NIPS*.

Recht, B.; Fazel, M.; and Parrilo, P. A. 2010. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*.

Reinsel, G. C., and Velu, R. P. 1998. *Multivariate Reduced-Rank Regression*.

Sun, Q.; Xiang, S.; and Ye, J. 2013. Robust principal component analysis via capped norms. In *KDD*.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*.

Yu, K.; Tresp, V.; and Schwaighofer, A. 2005. Learning Gaussian processes from multiple tasks. In *ICML*.

Zhang, Y., and Yeung, D.-Y. 2010a. A convex formulation for learning task relationships in multi-task learning. In *UAI*.

Zhang, Y., and Yeung, D.-Y. 2010b. Multi-task warped Gaussian process for personalized age estimation. In *CVPR*.

Zhang, Y., and Yeung, D.-Y. 2014. A regularization approach to learning task relationships in multitask learning. *ACM TKDD*.

Zhang, D.; Hu, Y.; Ye, J.; Li, X.; and He, X. 2012. Matrix completion by truncated nuclear norm regularization. In *CVPR*.

Zhang, J.; Ghahramani, Z.; and Yang, Y. 2005. Learning multiple related tasks using latent independent component analysis. In *NIPS*.

Zhang, Y.; Yeung, D.-Y.; and Xu, Q. 2010. Probabilistic multi-task feature selection. In *NIPS*.

Zhang, T. 2009. Multi-stage convex relaxation for learning with sparse regularization. In *NIPS*.

Zhang, T. 2010. Analysis of multi-stage convex relaxation for sparse regularization. *JMLR*.

Zhang, T. 2011. Multi-stage convex relaxation for feature selection. *arXiv preprint arXiv:1106.0565*.

Zhang, Y. 2013. Heterogeneous-neighborhood-based multi-task local learning algorithms. In *NIPS*.

Zhou, J.; Yuan, L.; Liu, J.; and Ye, J. 2011. A multi-task learning formulation for predicting disease progression. In *KDD*.