An Alternative DCA-based Approach for Reduced-Rank Multitask Linear Regression with Covariance Estimation

Hoai An Le Thi
Computer science and Applications Dept., LGIPM
University of Lorraine
Metz, France
hoai-an.le-thi@univ-lorraine.fr

Vinh Thanh Ho
Computer science and Applications Dept., LGIPM
University of Lorraine
Metz, France
vinh-thanh.ho@univ-lorraine.fr

Tao Pham Dinh
Laboratory of Mathematics, INSA - Rouen
University of Normandie
76801 Saint-Etienne-du-Rouvray Cedex, France
pham@insa-rouen.fr

***Abstract*—We investigate a nonconvex, nonsmooth optimization approach based on DC (Difference of Convex functions) programming and DCA (DC Algorithm) for the reduced-rank multitask linear regression problem with covariance estimation. The objective is to model the linear relationship between a multitask response and more explanatory variables by estimating a low-rank coefficient matrix and a covariance matrix. The problem is formulated as minimizing the constrained negative log-likelihood function of these two matrix variables. Then, we consider a reformulation of this problem which takes the form of a *partial* DC program i.e. it is a standard DC program for each variable when fixing the other variable. Next, an alternative version of DCA scheme is developed. Numerical results on synthetic multitask linear regression datasets and benchmark real datasets show the efficiency of our approach in comparison with the existing alternating/joint methods.**

***Index Terms*—multitask linear regression, reduced-rank, covariance estimation, DC programming, DCA, Alternative DCA**

I. INTRODUCTION

In this paper, we consider the reduced-rank multitask linear regression problem with covariance estimation (see, e.g., [1]). Given m different tasks with the d-dimensional feature vector denoted $\phi_i \in \mathbb{R}^d$, the corresponding respond denoted $z_i \in \mathbb{R}^m$ is generated using the linear model

$$z_i = X\phi_i + \epsilon_i, \tag{1}$$

where $X \in \mathbb{R}^{m\times d}$ is an unknown matrix whose rows represent the coefficient vector for each task; the error $\epsilon_i \in \mathbb{R}^m$ is assumed from a centered multivariate normal distribution with an unknown covariance matrix $\mathrm{Cov}(\epsilon_i) = (\Theta)^{-1}, \Theta \in \mathbb{R}^{m\times m}$.

The objective is to find the matrices X and Θ from n data points $\{(z_i, \phi_i)\}_{i=1,\ldots,n}$. In the high-dimensional setting, the problem aims to minimize the constrained negative log-likelihood function as follows.

$$\begin{aligned} \min \quad & \left[\frac{1}{n}\sum_{i=1}^{n}(z_i - X\phi_i)^\top \Theta (z_i - X\phi_i)\right] - \log\det(\Theta) \\ \text{s.t. } & X \in \mathcal{X}, \Theta \in \mathcal{Y}, \end{aligned} \tag{2}$$

where $\mathcal{X} = \{X \in \mathbb{R}^{m\times d} : \mathrm{rank}(X) = r\}$ represents the low-rank constraint set and $\mathcal{Y} = \{\Theta \in \mathbb{R}^{m\times m} : \Theta \succeq 0\}$ is the set of positive semi-definite matrices.

This problem has many real-world applications ranging from chemometrics (see e.g. [2]) to imaging neuroscience (see e.g. [3]), to quantitative finance and risk management (see e.g. [4]), to bioinformatics (see e.g. [5]), to robotics (see e.g. [6], [7]), to cite a few. For instance, in robotics, multivariate regression analysis is applied to evaluate the impact of robotic technique and high surgical volume on the cost of radical prostatectomy [6]. In another robotics application [7], linear regression analysis is performed to quantify the effect of surgeon experience on the operating time for each surgical step in the robotic-assisted laparoscopic prostatectomy procedure. In bioinformatics, the multitask regression algorithms are developed to solve the genomic selection problem in the fields of plant/animal breeding and genetic epidemiology (see [5] for more details).

In general, it is very hard to search globally optimal solutions to the problem (2) due to a double difficulty: first, the objective function of (2) is nonconvex in the variable (X, Θ), and, second, the rank function in the constraint set $\mathcal{X}$ is discontinuous and nonconvex.

There are some existing approaches for solving the problem (2) which use an alternating optimization procedure on the variable (X, Θ). In particular, a classic Alternating Method (AM) will alternate between computing two variables X and

Θ at every iteration (see e.g. [8]). It leads to solving, at each iteration, a reduced-rank regression problem in X (see [9]) and a convex program in Θ. Recently, Ha and Foygel Barber [1] have proposed an Alternating method using Gradient Descent method (AGD) for solving (2). The AGD method differs from the AM method by the fact that the AGD performs one iteration of the gradient descent method for solving the reduced-rank regression problem. Another approach without computing two variables alternatively is the joint gradient descent (JGD) method [1] which takes one gradient descent step in (X, Θ). All three AM, AGD, and JGD algorithms are described completely in the Appendix.

In this work, we still use the alternating optimization procedure on the variable (X, Θ). However since the problem (2) is nonconvex in X, we will investigate an alternating approach for solving (2) based on DC (Difference of Convex functions) programming and DCA (DC Algorithm) (see e.g. [10]–[13] and the references in [11], [14]) which are well-known as powerful nonsmooth, nonconvex optimization tools. DCA aims to solve a standard DC program that consists in minimizing a DC function $f = g - h$ (with g, h being convex functions) over a convex set or on the whole space. Here $g - h$ is called a DC decomposition of f, while g and h are DC components of f. The idea of the standard DCA is, at each iteration k, approximating the second DC component h by its affine minorant and then solving the resulting convex subproblem.

Our contribution. First, we consider a reformulation of the problem (2) which can be expressed as a *partial DC program* i.e. for fixed variables, it is a standard DC program in other variables. Second, we propose an alternative DCA scheme for solving this problem. In particular, at each iteration, we perform one iteration of standard DCA for the corresponding DC program in each variable when fixing the other variable. Finally, we evaluate our alternating approach by comparing with three alternating/joint methods on six synthetic multitask linear regression datasets and eight benchmark real datasets.

II. Solution Method

A. A Brief Introduction to Partial DC Programming and Alternative DCA

DC programming and DCA were introduced by Pham Dinh Tao in a preliminary form in 1985 and have been extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994. DCA is well-known as an efficient approach in the non-convex programming framework. In recent years, numerous DCA-based algorithms have been developed for successfully solving large-scale nonsmooth/nonconvex programs in several application areas (see the list of references in [11], [14]). For a comprehensible survey on thirty years of development of DCA, the reader is referred to the recent work [11].

The standard DCA scheme is described below. Its convergence properties are given completely in [10], [12], [15].

Standard DCA scheme
Initialization: Let $x^0 \in \mathbb{R}^p$ be a best guess. Set $k = 0$.
repeat
1. Calculate $\overline{x}^k \in \partial h(x^k)$.
2. Calculate $x^{k+1} \in \text{argmin}\{g(x) - \langle x, \overline{x}^k\rangle : x \in \mathbb{R}^p\}$.
3. $k = k + 1$.

until convergence of $\{x^k\}$.

Next, we briefly introduce partial DC programming and Alternative DCA [16]. A partial DC program takes the form

$$\min F(x, y) := G(x, y) - H(x, y) \text{ s.t. } (x, y) \in \mathbb{R}^p \times \mathbb{R}^q, \quad (3)$$

where G and H are partial convex functions in the sense that they are convex in each variable when fixing all other variables. Such a function F is called a *partial DC function*.

An alternative version of DCA for solving (3) consists in, at the iteration k, *alternatively* computing x^{k+1} and y^{k+1} by performing one iteration of standard DCA for solving the following DC programs in variable x and y, respectively:

$$\min F(x, y^k) := G(x, y^k) - H(x, y^k) \text{ s.t. } x \in \mathbb{R}^p,$$

and

$$\min F(x^{k+1}, y) := G(x^{k+1}, y) - H(x^{k+1}, y) \text{ s.t. } y \in \mathbb{R}^q.$$

This version, named Alternative DCA, is described as follows.

Alternative DCA scheme
Initialization: Let $(x^0, y^0) \in \mathbb{R}^p \times \mathbb{R}^q$ be a best guess. Set $k = 0$.
repeat
1. Calculate $\overline{x}^k \in \partial_x H(x^k, y^k)$.
2. Calculate $x^{k+1} \in \text{argmin}\{G(x, y^k) - \langle x, \overline{x}^k\rangle : x \in \mathbb{R}^p\}$.
3. Calculate $\overline{y}^k \in \partial_y H(x^{k+1}, y^k)$.
4. Calculate $y^{k+1} \in \text{argmin}\{G(x^{k+1}, y) - \langle y, \overline{y}^k\rangle : y \in \mathbb{R}^q\}$.
5. $k = k + 1$.

until convergence of $\{(x^k, y^k)\}$.

In the sequel, we present a reformulation of (2) and then show that it takes the form of a partial DC program for which the Alternative DCA scheme can be investigated.

B. A Reformulation of the Problem (2)

We reformulate the problem (2) by penalizing the difficult low-rank constraint in $\mathcal{X}$. In particular, for a given positive parameter α, the problem (2) can be transformed into the following optimization problem

$$\begin{aligned} \min\ F(X, \Theta) := {} & \frac{1}{n} \sum_{i=1}^{n} (z_i - X\phi_i)^\top \Theta (z_i - X\phi_i) \\ & - \log\det(\Theta) + \alpha d^2_{\mathcal{X}}(X) + \chi_{\Theta \succeq 0}(\Theta), \quad (4) \\ \text{s.t. } & X \in \mathbb{R}^{m\times d}, \Theta \in \mathbb{R}^{m\times m}, \end{aligned}$$

where the squared distance function $d^2_{\mathcal{X}}$ is defined as

$$d^2_{\mathcal{X}}(X) := \min_{Y \in \mathcal{X}} \|Y - X\|^2_F,$$

$\|\cdot\|_F$ is a Frobenius norm, and $\chi_{\mathcal{C}}$ is an indicator function of $\mathcal{C}$, defined as $\chi_{\mathcal{C}}(x) = 0$ if $x \in \mathcal{C}$, $+\infty$ otherwise.

Note that if (X^*, Θ^*) is a globally optimal solution to the problem (4) and $(X^*, \Theta^*) \in \mathcal{X} \times \mathcal{Y}$, then (X^*, Θ^*) is also a globally optimal solution to the problem (2).

It is easy to see that the function $d^2_{\mathcal{X}}$ is a DC function with DC decomposition

$$d^2_{\mathcal{X}}(X) = \min_{Y\in\mathcal{X}} \|X-Y\|_F^2 = \|X\|_F^2 - \max_{Y\in\mathcal{X}}(2\langle X, Y\rangle - \|Y\|_F^2).$$

As a result, the problem (4) can be expressed as a partial DC program

$$\min F(X, \Theta) := G(X, \Theta) - H(X, \Theta) \tag{5}$$

where

$$\begin{aligned} G(X,\Theta) &:= \frac{1}{n}\sum_{i=1}^{n}(z_i - X\phi_i)^\top \Theta (z_i - X\phi_i) \\ &\quad - \log\det(\Theta) + \alpha\|X\|_F^2 + \chi_{\Theta\succeq 0}(\Theta), \\ H(X,\Theta) &:= \alpha \max_{Y\in\mathcal{X}}(2\langle X, Y\rangle - \|Y\|_F^2). \end{aligned}$$

Obviously, the functions G and H are partially convex.

C. Alternative DCA for solving the problem (5)

According to the Alternative DCA scheme in Section II-A, we need to construct two sequences $\{(X^k, \Theta^k)\}$ and $\{(U^k, V^k)\}$ such that

$$\begin{aligned} &U^k \in \partial_X H(X^k, \Theta^k), \\ &X^{k+1} \in \operatorname{argmin}\{G(X, \Theta^k) - \langle X, U^k\rangle : X \in \mathbb{R}^{m\times d}\}, \end{aligned} \tag{6}$$

and

$$\begin{aligned} &V^k \in \partial_\Theta H(X^{k+1}, \Theta^k), \\ &\Theta^{k+1} \in \operatorname{argmin}\{G(X^{k+1}, \Theta) - \langle \Theta, V^k\rangle : \Theta \in \mathbb{R}^{m\times m}\}. \end{aligned} \tag{7}$$

From the definition of the function H, we compute the partial subdifferentials of H as follows:

$$\partial_X H(X, \Theta) = 2\alpha \mathrm{co}\{\mathrm{Proj}_{\mathcal{X}}(X)\} \text{ and } \partial_\Theta H(X, \Theta) = \{0\}.$$

Here $\mathrm{Proj}_{\mathcal{C}}$ and $\mathrm{co}(\mathcal{C})$ denote, respectively, the projection operator on the set $\mathcal{C}$ and the convex hull of $\mathcal{C}$.

We can choose the subgradients $U^k \in \partial_X H(X^k, \Theta^k)$ and $V^k \in \partial_\Theta H(X^{k+1}, \Theta^k)$ as follows:

$$U^k = 2\alpha W^k, \; W^k \in \mathrm{Proj}_{\mathcal{X}}(X^k), \text{ and } V^k = 0. \tag{8}$$

Solving the convex subproblem (6) amounts to solving the problem

$$\begin{aligned} \min_{X\in\mathbb{R}^{m\times d}} \; & \left[\frac{1}{n}\sum_{i=1}^{n}(z_i - X\phi_i)^\top \Theta^k (z_i - X\phi_i)\right] \\ & + \alpha\|X\|_F^2 - \langle U^k, X\rangle. \end{aligned} \tag{9}$$

By setting the derivative of the objective function of the last problem (9) to zero, we can see that its optimal solution X^{k+1} satisfies the Sylvester equation

$$A^k X + X B^k = C^k, \tag{10}$$

where the matrices $A^k \in \mathbb{R}^{m\times m}$, $B^k \in \mathbb{R}^{d\times d}$, and $C^k \in \mathbb{R}^{m\times d}$ are defined as

$$\begin{aligned} A^k &= \alpha(\Theta^k)^{-1}, \quad B^k = \frac{1}{n}\sum_{i=1}^{n}\left(\phi_i\phi_i^\top\right), \\ C^k &= \alpha(\Theta^k)^{-1}W^k + \frac{1}{n}\sum_{i=1}^{n}\left(z_i\phi_i^\top\right). \end{aligned}$$

From (7) and the definition of G, Θ^{k+1} is an optimal solution to the convex program

$$\min_{\Theta\succeq 0}\left[\frac{1}{n}\sum_{i=1}^{n}(z_i - X^{k+1}\phi_i)^\top \Theta (z_i - X^{k+1}\phi_i)\right] - \log\det(\Theta). \tag{11}$$

It is easy to check that the problem (11) has a closed-form optimal solution (see, e.g., [1]) as follows.

$$\Theta^{k+1} = \left(\frac{1}{n}\sum_{i=1}^{n}(z_i - X^{k+1}\phi_i)(z_i - X^{k+1}\phi_i)^\top\right)^{-1}. \tag{12}$$

Here Z^{-1} denotes an inverse of a matrix Z.

Finally, the Alternative DCA scheme applied to (5) can be summarized in Algorithm 1 (ADCA).

Algorithm 1 ADCA: Alternative DCA for solving (5)

Initialization: Let ε be a sufficiently small positive number. Let $X^0 \in \mathbb{R}^{m\times d}$, $\Theta^0 \in \mathbb{R}^{m\times m}$, $\Theta^0 \succeq 0$, $\alpha > 0$. Set $k = 0$.
repeat
1. Compute $W^k \in \mathrm{Proj}_{\mathcal{X}}(X^k)$.
2. Compute X^{k+1} by solving the Sylvester equation (10).
3. Compute Θ^{k+1} using (12).
4. $k = k + 1$.

until Stopping criteria are satisfied.

Remark 1: In numerical experiments, X^* obtained by ADCA does often not belong to $\mathcal{X}$. Thus, after stopping ADCA, we propose performing one projection step: projecting X^* into the set $\mathcal{X}$ and then updating Θ^* by (12).

III. Numerical experiments

Our experiments aim to compare the proposed alternative algorithm ADCA with other alternating/joint algorithms for the multitask linear regression problem (2).

Comparative algorithms. As listed in Section I, we consider three alternating/joint algorithms for solving the problem (2): classic alternating method (AM), alternating method using gradient descent method (AGD) [1], and joint gradient method (JGD) [1] (see the Appendix for more details).

Datasets. We test the four algorithms ADCA, AGD, JGD, and AM on six synthetic datasets and eight real datasets.

We generate synthetic datasets using the linear model (1) similarly to the works, e.g., [1], [17]–[19]. Specifically, the feature vector ϕ_i is drawn independently from a multivariate normal distribution $\mathcal{N}(0, \Sigma_\phi)$ where each element $\Sigma_\phi(i,j) = 0.5^{|i-j|}$. Similarly, the error ϵ_i is also generated from $\mathcal{N}(0, \sigma^2\Sigma_\epsilon)$ where σ^2 is chosen such that the corresponding signal-to-noise is equal to 1 (see, e.g., [1], [17]) and Σ_ϵ is defined by the following type: AR(1), denoted ar(ρ_ϵ), with $\Sigma_\phi(i, j) = (\rho_\epsilon)^{|i-j|}$. Here, ρ_ϵ represents a correlation parameter; the larger its value is, the more the degree of dependence of errors would be. The coefficient matrix X is computed as $X = AB$ where the orthonormal matrices $A \in \mathbb{R}^{m\times r}$ and $B \in \mathbb{R}^{r\times d}$ are generated form $\mathcal{N}(0, 1)$. Finally, the respond vector $z_i \in \mathbb{R}^m$ is computed using (1). By setting $r = 3$, $m \in \{10, 20, 60\}$, $d \in \{10, 20, 40\}$, $\rho_\epsilon \in \{0, 0.5\}$, we have six synthetic datasets which are summarized in Table I. For each synthetic dataset, we generate 50 training samples and 1000 test samples in each run time, and we repeat the whole process 30 times.

As for real datasets, we test on eight benchmark multitask regression datasets[1]. These datasets are collected from various interesting applications and can be found in the recent work [20] (see the references therein). The parameters of these datasets and the given values of r are provided in Table III. We split each real dataset into a training set containing the first 75% of dataset and a test set containing the rest of dataset.

Comparison criteria and stopping criteria. We are interested in the following aspects: prediction error and CPU time (in seconds) for training the solution (X^*, Θ^*). As for synthetic datasets, the prediction error is defined by the mean squared error (MSE) [17]

$$\text{MSE} = \frac{\sum_{i=1}^{n} \|X\phi_i - AB\phi_i\|_2^2}{nm}, \tag{13}$$

while the relative root mean squared error (RRMSE) on real datasets is used to measure the prediction error of the algorithm on each task and defined as [20]

$$\text{RRMSE} = \sqrt{\frac{\sum_{i=1}^{n} \|\hat{z}_i - z_i\|_2^2}{\sum_{i=1}^{n} \|\overline{z}_i - z_i\|_2^2}}, \tag{14}$$

where $\hat{z}_i$ is a respond vector estimated by the algorithm and $\overline{z}_i$ is the mean value of the respond vectors on the training set. We stop the algorithms if the relative difference between two consecutive points (X^{k-1}, Θ^{k-1}) and (X^k, Θ^k) or between two corresponding objective function values is less than or equal to ε.

Set up parameters. Our experiment is performed in MATLAB R2016b on a PC Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz of 8GB RAM. The MATLAB's `sylvester` function is used for solving Sylvester equation (10). All algorithms start with the same point (X^0, Θ^0). The starting point X^0 is set to a zero matrix in $\mathbb{R}^{m\times d}$, and the matrix Θ^0 is computed using (12). To validate the performance of the algorithms on all synthetic/real datasets, we consider the following validation procedure: first we run the algorithm with the different parameters on the training set, then choose the solution (X^*, Θ^*) that furnishes the best objective function value $F(X^*, \Theta^*)$, and finally evaluate the obtained model using MSE (13) or RRMSE (14) on the test set. The ranges of parameters η_X, η_Θ, and α are defined as: $\alpha \in \{5, 10, 100\}$, $\eta_X \in \{10^{-5}, 10^{-4}, \ldots, 10^2\}$, η_Θ belongs in a geometric sequence from 5 to 400 [1]. The default tolerance is $\varepsilon = 10^{-3}$.

[1]For the detailed descriptions of all datasets, the reader is referred to [20] and the website http://mulan.sourceforge.net/datasets-mtr.html.

TABLE I

COMPARATIVE RESULTS OF ADCA, AGD, JGD, AND AM IN TERMS OF THE AVERAGE OF MEAN-SQUARED-ERROR MSE DEFINED BY (13) (UPPER ROW) AND ITS STANDARD DEVIATION (LOWER ROW) ON SIX SYNTHETIC DATASETS OVER 30 RUN TIMES. BOLD VALUES INDICATE THE BEST RESULT.

d	m	Σ_ϵ	ADCA	AGD	JGD	AM
10	60	ar(0.0)	**2.44e-02**	2.65e-02	2.64e-02	5.73e-01
			6.47e-03	5.78e-03	5.71e-03	8.94e-01
		ar(0.5)	**2.00e-02**	2.42e-02	2.40e-02	4.34e-01
			6.15e-03	4.64e-03	4.81e-03	6.04e-01
20	10	ar(0.0)	**4.79e-02**	5.83e-02	1.37e-01	5.04e-02
			1.67e-02	2.39e-02	2.24e-02	1.65e-02
		ar(0.5)	**3.30e-02**	5.47e-02	1.33e-01	3.56e-02
			1.24e-02	2.87e-02	2.01e-02	1.10e-02
40	20	ar(0.0)	**6.13e-02**	6.15e-02	6.23e-02	2.34e+00
			8.01e-03	8.07e-03	8.28e-03	9.66e+00
		ar(0.5)	6.35e-02	**6.27e-02**	6.31e-02	5.72e-01
			8.76e-03	8.54e-03	8.42e-03	5.96e-01

Descriptions of result tables. The average MSE and its standard deviation obtained by all comparative algorithms on six synthetic datasets over 30 run times are reported in Table I. The average results of training time of the algorithms on synthetic datasets are given in Table II. Table III shows the experimental results on real datasets in terms of RRMSE and training time.

TABLE II

COMPARATIVE RESULTS OF ADCA, AGD, JGD, AND AM IN TERMS OF THE AVERAGE OF TRAINING TIME IN SECONDS (UPPER ROW) AND ITS STANDARD DEVIATION (LOWER ROW) ON SIX SYNTHETIC DATASETS OVER 30 RUN TIMES. BOLD VALUES INDICATE THE BEST RESULT.

d	m	Σ_ϵ	ADCA	AGD	JGD	AM
10	60	ar(0.0)	1.52e-02	**4.08e-03**	4.79e-03	5.81e-03
			9.29e-03	1.60e-03	1.62e-03	2.88e-03
		ar(0.5)	1.44e-02	**3.92e-03**	9.79e-03	4.81e-03
			5.35e-03	2.00e-03	3.78e-03	1.34e-03
20	10	ar(0.0)	3.14e-02	4.25e-02	**8.82e-04**	2.32e-03
			1.98e-02	1.65e-02	2.30e-04	1.04e-03
		ar(0.5)	2.06e-02	3.50e-02	**9.33e-04**	1.97e-03
			5.48e-03	1.75e-02	8.11e-05	3.48e-04
40	20	ar(0.0)	1.20e-01	7.35e-02	**1.40e-03**	4.87e-03
			2.60e-02	7.10e-02	6.87e-04	7.55e-04
		ar(0.5)	9.77e-02	6.10e-02	**8.92e-04**	4.64e-03
			1.96e-02	4.54e-02	1.63e-04	2.04e-04

Comments on numerical results

Synthetic datasets. We observe from Table I that, in terms of MSE, ADCA is more efficient than AGD, JGD, and AM. To be specific, ADCA is the best on $5/6$ datasets – the ratio of gain of ADCA versus AGD, JGD, and AM varies from 0.32% to 39.6%, from 1.60% to 75.1% and from 4.96% to 97.3%, respectively. Moreover, ADCA well performs for two model errors (independent, moderately correlated). In terms of training time, all four algorithms run very fast (less than 0.1 seconds).

Real datasets. The error RRMSE obtained by ADCA is the best on $6/8$ datasets, especially the rf2 dataset with more than 7000 samples. In particular, as for the rf2 dataset, ADCA significantly outperforms AGD, JGD and AM with the ratio of gain of 92.6%, 92.6% and 85.8%, respectively. On other datasets, the ratio of gain varies from 1.18% to 77.5%, from 4.07% to 77.5% and from 22.5% to 99.9%. Comparing with AM, ADCA is worse on $2/8$ datasets with the ratio from 5.36% to 9.19%. In Table III, training times of ADCA are reasonable (less than 1 seconds on $6/8$ datasets and 25 seconds on the atp7d and rf2 datasets).

IV. Conclusions

We have investigated a new approach based on DC programming and DCA for solving the reduced-rank multitask linear regression problem with covariance estimation. An Alternative version of DCA, ADCA, has been developed. Numerical results on synthetic/real datasets have turned out that the ADCA is more efficient than exiting alternating/joint methods in terms of the prediction error and runs within a reasonable consuming time. In the future, we plan to extend this work in the future to study the convergence properties of ADCA and show the efficiency of ADCA on many other synthetic/real datasets with different model errors as well as various applications.

Appendix
Comparative Algorithms for Solving the Problem (2)

The AM method alternates between computing the variable X and Θ at every iteration. In particular, at iteration k, for fixed Θ, we need to compute X^{k+1}, an optimal solution to the following problem (see, e.g., [9])

$$\min \frac{1}{n}\sum_{i=1}^{n}(z_i - X\phi_i)^\top \Theta^k (z_i - X\phi_i) \text{ s.t. } \operatorname{rank}(X) = r. \quad (15)$$

Let us denote by Z (resp. Φ) a matrix in $\mathbb{R}^{m\times n}$ (resp. $\mathbb{R}^{d\times n}$) whose each column is a vector z_i (resp. ϕ_i); and define $D^k := (\Phi\Phi^\top)^{(-1/2)}(\Phi Z^\top)(\Theta^k)^{(1/2)}$. A reduced-rank regression estimate X^{k+1} of (15) is given by

$$X^{k+1} = \sum_{t=1}^{r} \lambda_t \left[(1/n)\Phi\Phi^\top\right]^{(-1/2)} u_t v_t^\top (\Theta^k)^{(-1/2)}, \quad (16)$$

where the sequence $\{\lambda_t\}$ is the singular values of matrix D^k; $\{u_t\}$ and $\{v_t\}$ are the left-hand and right-hand singular vectors of D^k. For fixed X, the AM computes the point Θ^{k+1} using (12) at X^{k+1}. Note that the AM method does not have any parameters.

AM: classic Alternating Method for solving (2)

Initialization: Let ε be a sufficiently small positive number. Let $X^0 \in \mathbb{R}^{m\times d}$, $\Theta^0 \in \mathbb{R}^{m\times m}$, $\Theta^0 \succeq 0$. Set $k = 0$.

repeat

1. Compute X^{k+1} using (16).
2. Compute Θ^{k+1} using (12).
3. $k = k + 1$.

until Stopping criteria are satisfied.

The AGD method differs from the AM method by the fact that the AGD performs one iteration of gradient descent method for solving the convex problem (15). In particular, X^{k+1} is computed as follows [1]:

$$X^{k+1} = \operatorname{Proj}_{\mathcal{X}}\left(X^k + \frac{2\eta_X}{n}\Theta^k \sum_{i=1}^{n}(z_i - X^k\phi_i)\phi_i^\top\right), \quad (17)$$

where the step size η_X is a tuning parameter. It is similar for the AM to update the point Θ^{k+1} using (12) at X^{k+1}.

AGD: Alternating method using Gradient Descent method for solving (2)

Initialization: Let ε be a sufficiently small positive number. Let $X^0 \in \mathbb{R}^{m\times d}$, $\Theta^0 \in \mathbb{R}^{m\times m}$, $\Theta^0 \succeq 0$. Set $k = 0$.

repeat

1. Compute X^{k+1} using (17).
2. Compute Θ^{k+1} using (12).
3. $k = k + 1$.

until Stopping criteria are satisfied.

The JGD method does not compute two variables alternatively, but takes one gradient descent step in the joint variable (X, Θ). For estimating X^{k+1}, it is the same as (17), while the point Θ^{k+1} is computed by using gradient descent method for (11) at the point (X^k, Θ^k) as follows [1]:

$$\Theta^{k+1} = \operatorname{Proj}_{\mathcal{Y}}\left(\Theta^k + \eta_\Theta \Delta^k\right), \quad (18)$$

where the step size η_Θ is a tuning parameter and

$$\Delta^k = (\Theta^k)^{(-1)} - \left[\frac{1}{n}\Theta^k \sum_{i=1}^{n}(z_i - X^k\phi_i)(z_i - X^k\phi_i)^\top\right].$$

JGD: Joint Gradient Descent method for solving (2)

Initialization: Let ε be a sufficiently small positive number. Let $X^0 \in \mathbb{R}^{m\times d}$, $\Theta^0 \in \mathbb{R}^{m\times m}$, $\Theta^0 \succeq 0$. Set $k = 0$.

repeat

1. Compute X^{k+1} using (17).
2. Compute Θ^{k+1} using (18).
3. $k = k + 1$.

until Stopping criteria are satisfied.

References

[1] W. Ha and R. Foygel Barber, "Alternating minimization and alternating descent over nonconvex sets," *ArXiv e-prints*, Sep. 2017.

[2] S. Wold, M. Sjöström, and L. Eriksson, "PLS-regression: a basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109–130, 2001.

[3] L. Harrison, W. Penny, and K. Friston, "Multivariate autoregressive modeling of fMRI time series," *NeuroImage*, vol. 19, pp. 1477–1491, 09 2003.

[4] C. L. Lee, A. C. Lee, and J. Lee, *Handbook of Quantitative Finance and Risk Management*. Springer US, 01 2010.

TABLE III
COMPARATIVE RESULTS OF ADCA, AGD, JGD, AND AM IN TERMS OF THE RELATIVE ROOT-MEAN-SQUARED ERROR RRMSE DEFINED BY (14) (UPPER ROW) AND TRAINING TIME IN SECONDS (LOWER ROW) ON EIGHT REAL DATASETS. BOLD VALUES INDICATE THE BEST RESULT.

Name	n	d	m	r	ADCA	AGD	JGD	AM
andro	49	30	6	4	**8.42e-01**	3.75e+00	3.75e+00	1.22e+00
					5.17e-01	1.21e-03	**6.32e-04**	4.06e-03
atp7d	296	411	6	4	**3.63e+00**	4.60e+00	4.83e+00	2.74e+04
					2.45e+01	7.93e-03	**3.05e-03**	1.12e-01
oes10	403	298	16	4	**5.18e-01**	1.18e+00	1.26e+00	1.95e+01
					5.41e-01	2.82e-02	**3.56e-03**	7.53e-02
osales	639	376	12	4	**9.89e-01**	1.05e+00	1.05e+00	1.09e+03
					4.71e-02	4.61e-03	**4.50e-03**	1.34e-01
rf2	7679	576	8	6	**1.09e-01**	1.48e+00	1.48e+00	7.70e-01
					2.67e+00	2.54e-02	**2.52e-02**	3.48e-01
scpf	1137	8	3	3	9.88e-01	1.03e+00	1.03e+00	**9.35e-01**
					6.48e-03	**6.22e-04**	6.66e-04	1.65e-03
sf1	323	7	3	3	**9.61e-01**	1.02e+00	1.07e+00	1.24e+00
					1.49e-03	1.40e-03	**4.12e-04**	1.51e-03
wq	1060	16	14	4	9.98e-01	1.01e+00	1.62e+00	**9.14e-01**
					1.48e-02	1.45e-02	**9.50e-04**	2.85e-03

[5] D. He, L. Parida, and D. Kuhn, "Novel applications of multitask learning and multiple output regression to multiple genetic trait prediction," *Bioinformatics*, vol. 32, no. 12, pp. i37–i43, 2016.

[6] E. Hyams, J. Mullins, P. Pierorazio, A. Partin, M. Allaf, and B. Matlaga, "Impact of robotic technique and surgical volume on the cost of radical prostatectomy," *Journal of Endourology*, vol. 27, no. 3, pp. 298–303, 2013.

[7] H. Dev, N. L. Sharma, S. N. Dawson, D. E. Neal, and N. Shah, "Detailed analysis of operating time learning curves in robotic prostatectomy by a novice surgeon," *BJU International*, vol. 109, no. 7, pp. 1074–1080, 2012.

[8] J. Ortega and W. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, 1970.

[9] M. Aldrin, *Reduced-Rank Regression*. John Wiley & Sons, 2002, vol. 3, pp. 1724–1728.

[10] H. A. Le Thi and T. Pham Dinh, "The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems," *Annals of Operation Research*, vol. 133, no. 1–4, pp. 23–46, 2005.

[11] H. A. Le Thi and T. Pham Dinh, "DC programming and DCA: thirty years of developments," *Mathematical programming, Special Issue: DC Programming - Theory, Algorithms and Applications*, vol. 169, no. 1, pp. 5–68, 2018.

[12] T. Pham Dinh and H. A. Le Thi, "Convex analysis approach to DC programming: theory, algorithms and applications," *Acta Mathematica Vietnamica*, vol. 22, no. 1, pp. 289–355, 1997.

[13] ——, "Recent Advances in DC Programming and DCA," in *Transactions on Computational Intelligence XIII*, N. T. Nguyen and H. A. Le Thi, Eds. Springer Berlin Heidelberg, 2014, vol. 8342, pp. 1–37.

[14] H. A. Le Thi, "DC Programming and DCA: http://www.lita.univ-lorraine.fr/~lethi/index.php/en/research/dc-programming-and-dca.html (homepage)," 2005.

[15] T. Pham Dinh and H. A. Le Thi, "DC optimization algorithms for solving the trust region subproblem," *SIAM Journal of Optimization*, vol. 8, no. 2, pp. 476–505, 1998.

[16] H. A. Le Thi, V. N. Huynh, and T. Pham Dinh, "Alternative DC algorithm for partial DC programming," 2016, technical report, University of Lorraine.

[17] L. Chen and J. Z. Huang, "Sparse reduced-rank regression with covariance estimation," *Statistics and Computing*, vol. 26, no. 1, pp. 461–470, 2016.

[18] A. J. Rothman, E. Levina, and J. Zhu, "Sparse multivariate regression with covariance estimation," *Journal of Computational and Graphical Statistics*, vol. 19, no. 4, pp. 947–962, 2010.

[19] M. Yuan and Y. Lin, "Model selection and estimation in the gaussian graphical model," *Biometrika*, vol. 94, no. 1, pp. 19–35, 2007.

[20] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas, "Multi-target regression via input space expansion: treating targets as inputs," *Machine Learning*, vol. 104, no. 1, pp. 55–98, 2016.