Alternating direction methods for non convex optimization with applications to second-order least-squares and risk parity portfolio selection

Xi Bai*[1] and Katya Scheinberg†[1]

[1]Department of Industrial and Systems Engineering, Lehigh University, USA

Original Publication: Sep. 2, 2014
Last Revised: Feb. 11, 2015

Abstract

In this paper we mainly focus on optimization of sums of squares of quadratic functions, which we refer to as second-order least-squares problems, subject to convex constraints. Our motivation arises from applications in risk parity portfolio selection. We generalize the setting further by considering a class of nonlinear, non convex functions which admit a (non separable) two-block representation with special structure. We then develop alternating direction and alternating linearization schemes for such functions and analyze their convergence and complexity. Due to the special structure of our functions, the steps of our methods reduce to solving convex optimization subproblems. We provide convergence rate results for the proposed methods. Furthermore, some global relaxation techniques are presented to find lower bounds and strengthen our local algorithms. We show the effectiveness of our techniques in application to risk parity optimization in portfolio management.

Keywords: Augmented Lagrangian, alternating direction method, alternating linearization method, quartic optimization, sum of squares, risk parity.

1 Introduction

Minimizing a sum of squares is a classical approach to finding an approximate solution to (possibly) overdetermined systems of equations. In this paper we are interested in the case when the equations are quadratic. Specifically, consider solving the following system of quadratic equations:

$$x^T M_i x = c_i, \forall i \in \{1, ..., m\}, \quad x \in \mathcal{X} \tag{1.1}$$

*E-mail: xib210@lehigh.edu,
†E-mail: katyas@lehigh.edu,

where $\mathcal{X}$ is some convex set and $M_i \in \mathbb{R}^{n\times n}$ are matrices, not necessarily symmetric. In this paper, we consider solving this problem by minimizing a sum of squares:

$$\min_{x\in\mathcal{X}} F = \sum_{i=1}^{m} F_i(x) = \sum_{i=1}^{m} (x^T M_i x - c_i)^2, \tag{1.2}$$

which we refer to, in this paper, as second-order least-squares problem. Thus, we aim to minimize a forth-order (quartic) polynomial, which is nonconvex in general, over some convex set. Solving a system of quadratic equations or minimizing a quartic polynomial is NP-hard in general (see, for instance, [8, 17]).

There are several simple extensions of (1.1) and (1.2) to which methods in this paper can be readily applied. For instance, (1.1) can be extended by adding a non-homogeneous term on the left hand side, i.e.

$$x^T M_i x + p_i^T x = c_i, \forall i \in \{1, ..., m\}, \tag{1.3}$$

where $m_i \in \mathbb{R}^n$ is a given vector.

Another natural extension is the introduction of a regularization term. System (1.3) may be under-determined or it only needs to be satisfied approximately, i.e. $x^T M_i x + p_i^T x \approx c_i, \forall i \in \{1, ..., m\}$. In these situations, we may consider a problem in this form:

$$\min_{x\in\mathcal{X}} F = \sum_{i} (x^T M_i x + p_i^T x - c_i)^2 + q(x), \tag{1.4}$$

where $q(x)$ is a regularization function that has some desired properties.

Below are the practical examples that provided initial motivation for the methods in this paper.

Example 1.1. Least-squares risk parity problem.

Risk parity arises in portfolio selection when the objective is to develop portfolios for which the contributions of risk from all assets are equally weighted [18]. If volatility of the returns is chosen as the risk measure, then risk parity problem can be represented as

$$\begin{array}{l} x_i(\Sigma x)_i = x_j(\Sigma x)_j, \ \forall i, j, \\ a_i \le x_i \le b_i \\ \sum_{i=1}^n x_i = 1, \end{array} \tag{1.5}$$

where x is the weight vector of individual assets, a_i and b_i are lower and upper bounds on the weight of the i-th asset, and Σ is the covariance matrix of the assets. Depending on the constraints on x, the existence of the exact solution of (1.5) may not be simple to establish. As an alternative, in [3], the following least-square model is proposed for risk parity portfolios:

$$\begin{array}{ll} \min_{x,\theta} & \sum_{i=1}^n (x_i(\Sigma x)_i - \theta)^2 + q(x) \\ \text{s.t.} & a_i \le x_i \le b_i \\ & \sum_{i=1}^n x_i = 1, \end{array} \tag{1.6}$$

where $q(x)$ is a customized measure function. In the standard risk parity portfolio selection problem, $q(x) = 0$. It was shown in [3] that risk parity solution may not be unique when shorting of assets is allowed ($a_i < 0$ for some i). In that case one may aim at finding a risk parity solution with the least variance, for instance. In that case $q(x) = \rho x^T \Sigma x$, where $\rho > 0$ is a regularization parameter.

The risk parity formulation (1.6) clearly fits the form (1.4), with $\mathcal{X} = \{x : a \le x \le b\} \otimes \mathbb{R}$, $M_i = \Sigma_i^T e_i$, and $p_i = (-e_{n+1})^T$, where $\Sigma_i \in \mathbb{R}^{1\times(n+1)}$ is the i-th row of the covariance matrix with a zero added as the $n+1$st element and $e_i \in \mathbb{R}^{1\times(n+1)}$ is the i-th row of the identity.

Example 1.2. Group risk parity problem.

More generally, one may divide assets into groups by sectors (of industry). In this case the risk parity between these sectors rather than individual assets is desired. This leads to the following grouped formulation [3]:

$$\begin{array}{ll} \min_{x,\theta} & \sum_{j=1}^{l}(\sum_{i\in\mathcal{G}_j} x_i(\Sigma x)_i - \theta)^2 \\ \text{s.t.} & a_i \le x_i \le b_i \\ & \sum_{i=1}^{n} x_i = 1, \end{array} \tag{1.7}$$

where $\mathcal{G}_j$ stands for the jth group, and l is the total number of sectors. Again, this problem can be written in form (1.4), with $\mathcal{X}$ and p_i defined as in individual risk parity case and $M_j = A_j^T B_j$, where $A_j \in \mathbb{R}^{m_j\times(n+1)}$ contains all rows of Σ indexed by $\mathcal{G}_j$ and appended by a zero element and $B_j \in \mathbb{R}^{m_j\times(n+1)}$ is defined as follows. Suppose the i-th row in A_j is the corresponding (k_i)th row in Σ, then

$$(B_j)_{i,k} = \begin{cases} 1, & k = k_i \\ 0, & otherwise. \end{cases}$$

For the main part of this paper we generalize the setting further. In particular, we consider optimizing (over a convex set $\mathcal{X}$) an objection function $F(x)$, which can be written as $F(x) = h(f(x), g(x))$, where f and g are (possibly) vector functions of x, and h is some function of the corresponding arguments. We make some standard assumptions on the smoothness and boundedness of F, but the key assumption for the purposes of this paper is that h, f and g are such that for any fixed $\bar{x}$, $h(f(\bar{x}), g(x))$ and $h(f(x), g(\bar{x}))$ are smooth and convex functions. This assumption applies to our second-order least squares problem (1.4).

Based on the representation of function $F(x)$ we propose an algorithmic framework based on variable splitting and augmented Lagrangian technique. Our framework consists of the well-known alternating direction method of multipliers (ADMM) and alternating linearization method (ALM) which has been widely studied in recent literature, primarily for convex optimization. Our focus and main contribution is to analyze these methods in a nonconvex setting where the objective function cannot be represented as a sum of multiple functions. The problem under discussion is potentially nonconvex, with convexity being assumed in some subspace when variable splitting is applied, and our methods are convergent to a local minimum. We provide global complexity analysis for both ADMM and ALM and show that they converge at the sub-linear rate of $O(1/\sqrt{k})$. The difference between the two frameworks is that ALM requires computing partial gradient information and as a result benefits from possibility of varying the choice of proximal parameter and choosing it via backtracking. It also requires constraints $x \in \mathcal{X}$ to be enforced for each subproblem optimization. In ADMM, on the other hand, the choice of the proximal parameter has to be fixed and sufficiently small (at least in theory), while the constraints are only enforced for one of the subproblems on each iteration. Both frameworks are simple and rely on solving a sequence of convex subproblems. Our experiments show that ALM is more efficient in terms of the number of convex subproblems that need to be solved. We also show, that our alternating linearization scheme is related to the classical Levenberg-Marquard method in case of solving least-squares problems.

Since our methods find local stationary points they provide upper bounds on objective function value. While we do not focus on global optimization techniques here, we provide some lower bounds for the risk parity problems specifically, to demonstrate that our local solutions happen to be global

in our numerical example. In particular, we consider the sum-of-square (SOS) techniques which is a popular approach for polynomial optimization [16, 21, 22]. SOS finds a global lower bound by checking the membership in the sum-of-square cone via semidefinite programming (SDP). There have been several variants of SOS methods and we discuss some of them in this paper. These techniques are usually computationally expensive as the dimension of the resulting SDP problem grows fast, but they appear to produce useful results for the applications that we are interested in.

The rest of the paper is organized as follows. After a brief discussion of the problem structure in Section 2, in Section 3 we introduce the class of algorithms based on variable splitting and augmented Lagrangian function and develop and analyze the ADMM method. We introduce and analyze the ALM method in Section 4. We discuss the application of SOS technique and its variants in Section 5. The experiments for risk parity problem and computational results are presented in Section 6, followed by conclusion remarks in Section 7.

2 Alternating direction schemes for minimizing a nonconvex objective that is *not* necessarily composite

2.1 Notations and preliminaries

Consider the following nonlinear optimization problem:

$$\min_{x\in\mathcal{X}} F(x), \tag{2.1}$$

where $\mathcal{X} \in \mathbb{R}^n$ is a simple convex set. We seek local solutions $\bar{x}$ that satisfy first-order optimality condition

$$\langle \nabla F(\bar{x}), x - \bar{x}\rangle \geq 0, \forall x \in \mathcal{X}. \tag{2.2}$$

Suppose that the function F can be written as a function of two blocks, i.e. $F(x) = h(f(x), g(x))$, where function f is $\mathbb{R}^n \to \mathbb{R}^{m_1}$, g is $\mathbb{R}^n \to \mathbb{R}^{m_2}$ and h is $\mathbb{R}^{m_1+m_2} \to \mathbb{R}$. Note that we do not assume that the objective can be decomposed as a sum of f and g.[1]

Consider the following function of x, which is restriction of F, with the second block, g, fixed, given a fixed $\bar{x}$.

$$F_1(x, \bar{x}) = h(f(x), g(\bar{x})). \tag{2.3}$$

Similarly, if we fix the first block, f, given $\bar{x}$, we have the following function

$$F_2(\bar{x}, x) = h(f(\bar{x}), g(x)). \tag{2.4}$$

In this section, we use the "2-block" notations so that the subscript $i = 1, 2$ indicates which block is variable. In particular the partial derivative of the objective over the ith block is denoted as $\nabla_i F$ $(i = 1, 2)$. Then $\nabla_1 F(x, \bar{x}) = \nabla_f h(f(x), g(\bar{x})) \cdot \nabla_x f(x)$ and $\nabla_2 F(\bar{x}, x) = \nabla_g h(f(\bar{x}), g(x)) \cdot \nabla_x g(x)$, are the gradients of (2.3) and (2.4), respectively, where $\nabla_f h$ and $\nabla_g h$ are the corresponding partial derivatives of h with respect to the first and second block. Note that, for any x, the relationship between the full gradient of F and our notations is simply

$$\nabla F(x) = \nabla_1 F(x, x) + \nabla_2 F(x, x). \tag{2.5}$$

[1]Such functions are called composite functions, in recent literate, and are assumed to have the form $F(x) = f(x) + g(x)$, or in multi-block case, $F(x) = \sum_{i=1}^n f_i(x)$.

In particular, this and (2.2) imply that x^* is a stationary point of (1.2) as long as

$$\langle \nabla_1 F(x^*, x^*) + \nabla_2 F(x^*, x^*), x - x^* \rangle \geq 0, \forall x \in \mathcal{X}. \tag{2.6}$$

We now list the key assumptions on the function F and the resulting functions F_1 and F_2. Two assumptions are standard - smoothness of the functions and boundedness from below. The third assumption - convexity of the functions F_1 and F_2 - is the key assumption which allows us to develop efficient framework based on alternating directions.

- **Lipschitz continuity of the gradients.** Gradients of functions $\nabla F(x)$, $\nabla_1 F(x, \bar{x})$ and $\nabla_2 F(\bar{x}, x)$ are Lipschitz continuous with Lipschitz constant L, for any $\bar{x} \in \mathcal{X}$, i.e. $F, F_1, F_2 \in C_L^{1,1}(\mathcal{X})$,
- **Blockwise convexity.** $F_1(x, \bar{x}), F_2(\bar{x}, x)$ are convex over x, for any $\bar{x} \in \mathcal{X}$,
- **Boundedness from below.** F (and hence $F_1(x, \bar{x})$ and $F_2(\bar{x}, x)$) is bounded from below, for any $\bar{x} \in \mathcal{X}$, i.e. $F > \infty$.

For each block, we define the following two functions as local approximations of F at any given $\bar{x} \in \mathbb{R}^n$.

$$\begin{aligned} Q_\mu^1(x, \bar{x}) &= F_1(x, \bar{x}) + \langle \nabla_2 F(\bar{x}, \bar{x}), x - \bar{x} \rangle + \tfrac{1}{2\mu} ||x - \bar{x}||_2^2 \\ Q_\mu^2(\bar{x}, x) &= F_2(\bar{x}, x) + \langle \nabla_1 F(\bar{x}, \bar{x}), x - \bar{x} \rangle + \tfrac{1}{2\mu} ||x - \bar{x}||_2^2, \end{aligned} \tag{2.7}$$

where μ is a positive scalar. These functions will be used by our alternating linearization method in Section 4.

Below are some examples of $F(x)$ and the resulting functions F_1, F_2, Q_μ^1 and Q_μ^2 . We start with the standard composite function case.

Example 2.1. Assume that $h = f + g$ and thus

$$F(x) = f(x) + g(x),$$

where $f(x)$ and $g(x)$ are convex and smooth scalar functions. Then we define $F_1(x, \bar{x}) = f(x) + g(\bar{x})$ and $F_2(\bar{x}, x) = f(\bar{x}) + g(x)$, which clearly leads to

$$\begin{aligned} Q_\mu^1(x, \bar{x}) &= f(x) + g(\bar{x}) + \langle \nabla g(\bar{x}), x - \bar{x} \rangle + \tfrac{1}{2\mu} ||x - \bar{x}||_2^2 \\ Q_\mu^2(\bar{x}, x) &= f(\bar{x}) + g(x) + \langle \nabla f(\bar{x}), x - \bar{x} \rangle + \tfrac{1}{2\mu} ||x - \bar{x}||_2^2. \end{aligned} \tag{2.8}$$

Note that our approximation functions in this case are the same as the standard block-wise proximal functions from the composite optimization literature.

We now turn to more complex function structures.

Example 2.2. Assume that $h = f \cdot g$ and thus

$$F(x) = f(x)g(x),$$

where $f(x)$ and $g(x)$ are convex, smooth and nonnegative. Then we define $F_1(x,\bar{x}) = f(x)g(\bar{x})$ and $F_2(\bar{x},x) = f(\bar{x})g(x)$, which implies that F_1 and F_2 satisfy Assumption and we have

$$\begin{aligned} Q_\mu^1(x,\bar{x}) &= f(x)g(\bar{x}) + f(\bar{x})\left\langle \nabla g(\bar{x}), x-\bar{x}\right\rangle + \tfrac{1}{2\mu}||x-\bar{x}||_2^2 \\ Q_\mu^2(\bar{x},x) &= f(\bar{x})g(x) + g(\bar{x})\left\langle \nabla f(\bar{x}), x-\bar{x}\right\rangle + \tfrac{1}{2\mu}||x-\bar{x}||_2^2. \end{aligned} \tag{2.9}$$

Finally, we present another general setting which includes the objective function of (1.2) and satisfies Assumption 2.2.

Example 2.3. Assume that $h = \sum_i ((f^i)^T g^i - c_i)^2$ and thus

$$F(x) = \sum_i (f^i(x)^T g^i(x) - c_i)^2, \tag{2.10}$$

where each $f^i(x) = [f_1^i(x), f_2^i(x), ..., f_{m_i}^i(x)]$ and $g^i(x) = [g_1^i(x), g_2^i(x), ..., g_{m_i}^i(x)]$ is an affine function of x, $\mathbb{R}^n \to \mathbb{R}^{m_i}$.

It is easy to see, that is functions f^i and g^i are affine and homogeneous (that is they do not contain a constant term), then this form is equivalent to (1.2), with an appropriate choice of M_i and c_i parameters, otherwise formulation (1.4) applies with $q(x) = 0$. Clearly, when $q(x)$ is a smooth convex function, then $h = \sum_i (f_i^T g_i - c_i)^2 + q(x)$ should be considered.

In the risk parity case, in particular, we have

$$\begin{array}{ll} \min_{x,\theta} & F(x) = \sum_{i=1}^n (x^T M_i x - \theta)^2 \\ \text{s.t.} & x \in \mathcal{X}, \end{array}$$

which can be written as (2.10) with $f^i(x,\theta) = [x, -1]$ and $g^i(x,\theta) = [M_i x, \theta]$, $i = 1, \ldots, n$ are $(n+1)$-dimensional affine vector functions of x and θ and $c_i = 0$, $\forall i$. For any given $\bar{x}$: $F_1(x,\bar{x},\bar{\theta}) = \sum_{i=1}^m (x^T M_i \bar{x} - \bar{\theta})^2$ and $F_2(\bar{x},x) = \sum_{i=1}^m (\bar{x}^T M_i x - \theta)^2$. Both F_1 and F_2 are convex quadratic functions and Assumption 2.2 is satisfied.

3 Variable splitting and augmented Lagrangian based methods

In this section, we discuss several alternating direction methods, all of which are based on the augmented Lagrangian framework with variable splitting. Augmented Lagrangian method (with variable splitting) and its variants have been increasingly popular in recent literature [1, 7, 12, 28, 29].

In particular observe that (2.1) can be equivalently written as

$$\begin{array}{ll} \min_{x\in\mathcal{X},y} & F(x,y) = h(f(x), g(y)) \\ \text{s.t.} & x = y, \end{array} \tag{3.1}$$

where $x, y \in \mathbb{R}^n$. In other words, we map the dimension of decision variable from n in (2.1) to $2n$ in (3.1).

Consider problem in the form of (3.1). Provided a penalty parameter $1/\mu$ ($\mu > 0$), we have the following augmented Lagrangian function:

$$\mathcal{L}_A(x,y;\lambda) = F(x,y) - \lambda^T(x-y) + \frac{1}{2\mu}\|x-y\|^2, \tag{3.2}$$

and, hence, (3.1) can be solved by the augmented Lagrangian method described in Algorithm 1.

Algorithm 1 Augmented Lagrangian method (AL)

1. Choose μ^0, λ^0, and $x^0 = y^0$;
2. for $k = 0, 1, ...,$ do
   $[x^{k+1}, y^{k+1}] := \arg\min_{x\in\mathcal{X},y} \mathcal{L}_A(x, y; \lambda^k)$;
   update the multiplier $\lambda^{k+1} = \lambda^k - \frac{1}{\mu^k}(x^{k+1} - y^{k+1})$;
   possibly choose new penalty parameter μ^{k+1}.

Algorithm 2 Alternating direction methods of multipliers (ADMM)

1. Choose μ, λ^0, and $x^0 = y^0$;
2. for $k = 0, 1, ...,$ do
   $x^{k+1} := \arg\min_{x\in\mathcal{X}} \mathcal{L}_A(x, y^k; \lambda^k)$;
   $y^{k+1} := \arg\min_y \mathcal{L}_A(x^{k+1}, y; \lambda^k)$;
   update the multiplier $\lambda^{k+1} = \lambda^k - \frac{1}{\mu}(x^{k+1} - y^{k+1})$;

Example 3.1. In the second-order least-squares case, we split the variables as follows

$$\begin{array}{ll} \min_{x,y} & F(x) = \sum_{i=1}^n (x^T M_i y - c_i)^2 \\ \text{s.t.} & x = y \\ & x \in \mathcal{X}. \end{array} \tag{3.3}$$

Then the augmented Lagrangian function is defined as

$$\mathcal{L}_A(x, y; \lambda) = \sum_{i=1}^n (x^T M_i y - c_i)^2 - \lambda^T (x - y) + \frac{1}{2\mu}\|x - y\|^2.$$

The convergence of augmented Lagrangian method (see Algorithm 1) has been well studied (see, for instance, [5, 30]). Furthermore, the minimization of the augment Lagrangian in Algorithm 1, can be performed by applying block coordinate decent method (BCD) until first-order optimality is guaranteed. Convergence of block coordinate decent method, under the assumption of uniqueness of minimizers over blocks, and for simple convex constraints is studied, for instance, in [13].

3.1 Alternating direction methods of multipliers

Alternating direction methods of multipliers (ADMM), or alternating direction augmented Lagrangian method (ADAL), can be regarded as a variant of augmented Lagrangian with subproblems solved inexactly by BCD. ADMM and other alternating direction methods (ADMs) can be tracked back to the Douglas-Rachford method in the 1950s [9] and ADMs for solving variational problems associated with PDEs in the 1970s [10, 11]. In ADMM the multiplier is updated after only one minimization step over each x and y blocks instead of after minimizing the augmented Lagrangian over x and y jointly. A simple framework of ADMM for solving (3.1) is given in Algorithm 2.

ADMM has been widely used and well studied, in the large-scale convex optimization setting in the case of composite structure of the objective function (see [7] for a review). In the nonconvex setting, ADMM has been applied to obtain KKT solutions often obtaining competitive results, empirically. However, its theoretical properties are not well understood in the non convex case. Typically, convergence is shown under the assumption that the successive differences of the iterates

converge to zero (see, for instance, an ADMM for polynomial optimization described in [15]). While such assumption seems reasonable in that the iterates produced by the algorithm do not exhibit any erratic behavior, it is not clear if this condition can be verified in advance or enforced during the progress of the algorithm. Recently Hong, et al. show the convergence of a family of ADMMs without this assumption when applied to a family of n-block structured composite nonconvex problems [14]. Our results in this paper also do not rely on this assumption, but are different from [14] in that our objective function is not assumed to be a sum of multiple functions (composite form). Note that it is not trivial to extend our convergence result to problems with more than two blocks. However, the two-block results shown in this section readily apply to the second-order least-squares problem which is the focus of our work.

Our strategy to prove the convergence of ADMM is similar to that of [14], in that it relies on obtaining a sufficient function decrease of augmented Lagrangian. In our proof, we frequently use the term "x-update" and "y-update", which refer to the update rule of each block of variables x and y, respectively. First, we have the following result to bound the augmented Lagrangian function value, which is a relaxed version of Lemma 2.3 in [14].

Lemma 3.1. *Let Assumption 2.2 hold with the Lipschitz constant L and let $\mu \leq \frac{1}{4L}$. Then for the sequence of iterates $\{x^k, y^k, \lambda^k\}$ defined by Algorithm 2, the augmented Lagrangian function converges to some limit $\mathcal{L}^*$:*

$$\lim_{k\to\infty} \mathcal{L}_A(x^k, y^k; \lambda^k) = \mathcal{L}^*$$

Proof. By the first-order optimality condition of y-update in Algorithm 2, we have

$$\nabla_2 F(x^{k+1}, y^{k+1}) + \lambda^k - \frac{1}{\mu}(x^{k+1} - y^{k+1}) = 0,$$

which leads to

$$\lambda^{k+1} = \lambda^k - \frac{1}{\mu}(x^{k+1} - y^{k+1}) = -\nabla_2 F(x^{k+1}, y^{k+1}). \tag{3.4}$$

Thus, we can bound the change of λ:

$$\begin{aligned} \|\lambda^{k+1} - \lambda^k\| &= \|\nabla_2 F(x^{k+1}, y^{k+1}) - \nabla_2 F(x^k, y^k)\| \\ &\leq L\left(\|x^{k+1} - x^k\| + \|y^{k+1} - y^k\|\right), \end{aligned} \tag{3.5}$$

where L is a Lipschitz constant of the gradient.

Now, we can bound the change of the augmented Lagrangian function value, after the update of the primal variables. Since it is assumed that at each iteration $F_1(x, y^k)$ is convex, the function of subproblem of ADMM can be regarded as a sum of convex and strongly convex function and thus is strongly convex, i.e., we have $\mathcal{L}_A(x^k, y^k; \lambda^k) \geq \mathcal{L}_A(x^{k+1}, y^k; \lambda^k) - \left\langle \nabla_1 \mathcal{L}_A(x^{k+1}, y^k; \lambda^k), x^{k+1} - x^k \right\rangle + \frac{1}{2\mu}\|x^{k+1} - x^k\|^2$. From the optimality of the subproblem we obtain that

$$\mathcal{L}_A(x^k, y^k; \lambda^k) - \mathcal{L}_A(x^{k+1}, y^k; \lambda^k) \geq \frac{1}{2\mu}\|x^{k+1} - x^k\|^2. \tag{3.6}$$

Similarly, for y-update, it holds that

$$\mathcal{L}_A(x^{k+1}, y^k; \lambda^k) - \mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^k) \geq \frac{1}{2\mu}\|y^{k+1} - y^k\|^2. \tag{3.7}$$

We also bound the change of the augmented Lagrangian function value, after the update of the multipliers:

$$\begin{aligned}
&\mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^{k+1}) - \mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^k) \\
&= F(x^{k+1}, y^{k+1}) - \left\langle \lambda^{k+1}, x^{k+1} - y^{k+1} \right\rangle + \tfrac{1}{2\mu}\|x^{k+1} - y^{k+1}\|^2 \\
&- F(x^{k+1}, y^{k+1}) - \left\langle \lambda^{k}, x^{k+1} - y^{k+1} \right\rangle + \tfrac{1}{2\mu}\|x^{k+1} - y^{k+1}\|^2 \\
&= -\left\langle \lambda^{k+1} - \lambda^k, x^{k+1} - y^{k+1} \right\rangle \\
&= \mu\|\lambda^{k+1} - \lambda^k\|^2.
\end{aligned}$$

By using (3.5), we further have

$$\begin{aligned}
&\mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^{k+1}) - \mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^k) \\
&\leq \mu L^2 \left(\|x^{k+1} - x^k\| + \|y^{k+1} - y^k\|\right)^2 \\
&\leq 2\mu L^2 \left(\|x^{k+1} - x^k\|^2 + \|y^{k+1} - y^k\|^2\right).
\end{aligned} \tag{3.8}$$

With (3.6), (3.7) and (3.8), we finally have

$$\begin{aligned}
&\mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^{k+1}) - \mathcal{L}_A(x^{k}, y^{k}; \lambda^k) \\
&\leq \left(2\mu L^2 - \tfrac{1}{2\mu}\right)\|x^{k+1} - x^k\|^2 + \left(2\mu L^2 - \tfrac{1}{2\mu}\right)\|y^{k+1} - y^k\|^2,
\end{aligned} \tag{3.9}$$

which indicates a monotonic decrease of augmented Lagrangian function value as long as $\mu < \frac{1}{2L}$. For instance, we can choose μ to be $\frac{1}{4L}$.

Moreover, we can bound the value of $\mathcal{L}_A$ by F:

$$\begin{aligned}
&\mathcal{L}_A(x^{k+1}, y^{k+1}; \lambda^{k+1}) \\
&= F(x^{k+1}, y^{k+1}) - \left\langle \lambda^{k+1}, x^{k+1} - y^{k+1} \right\rangle + \tfrac{1}{2\mu}\|x^{k+1} - y^{k+1}\|^2 \\
&= F(x^{k+1}, y^{k+1}) + \left\langle \nabla_2 F(x^{k+1}, y^{k+1}), x^{k+1} - y^{k+1} \right\rangle + \tfrac{1}{2\mu}\|x^{k+1} - y^{k+1}\|^2 \\
&\geq F(x^{k+1}),
\end{aligned}$$

for $\mu \leq \frac{1}{L}$. Since it is assumed that F is bounded from below, it follows that the sequence $\{\mathcal{L}_A(x^k, y^k; \lambda^k)\}$ is also bounded from below and thus the augmented Lagrangian function value converges. □

It is possible to prove this result without assuming convexity of F_1, F_2, by choosing μ large enough so that the augmented Lagrangian is strongly convex at each iteration.

Now we show that, under further assumptions, any limit point of the sequence $\{x^k\}$ is a stationary point of (2.1).

Theorem 3.1. *Let Assumption 2.2 hold with the Lipschitz constant L and let $\mu < \frac{1}{4L}$. Then any limit point of sequence $\{x^k\}$ generated by Algorithm 2 is a stationary point of* (2.1).

Proof. As $\mathcal{L}_A(x^k, y^k; \lambda^k)$ converges by Lemma 3.1, it follows from (3.5) and (3.9) that

$$\begin{aligned}
x^{k+1} - x^k &\to 0 \\
y^{k+1} - y^k &\to 0 \\
\lambda^{k+1} - \lambda^k &\to 0,
\end{aligned}$$

which indicates that $x^k - y^k \to 0$ from (3.4), since μ is bounded from above.

Assume $\{\bar{x}, \bar{y}, \bar{\lambda}\}$ is a limit point of the sequence $\{x^k, y^k, \lambda^k\}$. Our goal is to prove the first-order condition, i.e. for any x such that $x \neq \bar{x}$ and $x \in \mathcal{X}$, it holds that $\left\langle \nabla F(\bar{x}), \frac{x-\bar{x}}{\|x-\bar{x}\|} \right\rangle \geq 0$, which is equivalent to (2.2).

In fact, for any $x \neq \bar{x}$,

$$\begin{aligned}
&\left\langle \nabla F(\bar{x}), \tfrac{x-\bar{x}}{\|x-\bar{x}\|} \right\rangle \\
&= \tfrac{1}{\|x-\bar{x}\|} \langle \nabla_1 F(\bar{x}, \bar{y}) + \nabla_2 F(\bar{x}, \bar{y}), x - \bar{x} \rangle \\
&+ \tfrac{1}{\|x-\bar{x}\|} \langle \nabla_1 F(\bar{x}, \bar{x}) - \nabla_1 F(\bar{x}, \bar{y}), x - \bar{x} \rangle + \tfrac{1}{\|x-\bar{x}\|} \langle \nabla_2 F(\bar{x}, \bar{x}) - \nabla_2 F(\bar{x}, \bar{y}), x - \bar{x} \rangle \\
&= \tfrac{1}{\|x-\bar{x}\|} \left\langle \nabla_1 F(\bar{x}, \bar{y}) - \bar{\lambda}, x - \bar{x} \right\rangle \\
&+ \tfrac{1}{\|x-\bar{x}\|} \langle \nabla_1 F(\bar{x}, \bar{x}) - \nabla_1 F(\bar{x}, \bar{y}), x - \bar{x} \rangle + \tfrac{1}{\|x-\bar{x}\|} \langle \nabla_2 F(\bar{x}, \bar{x}) - \nabla_2 F(\bar{x}, \bar{y}), x - \bar{x} \rangle ,
\end{aligned} \tag{3.10}$$

where $\langle \nabla_1 F(\bar{x}, \bar{x}) - \nabla_1 F(\bar{x}, \bar{y}), x - \bar{x} \rangle \leq L\|\bar{x} - \bar{y}\|\|x - \bar{x}\|, \langle \nabla_2 F(\bar{x}, \bar{x}) - \nabla_2 F(\bar{x}, \bar{y}), x - \bar{x} \rangle \leq L\|\bar{x} - \bar{y}\|\|x - \bar{x}\|$ and thus the last two terms of (3.10) vanish as $\bar{x} - \bar{y} \to 0$. From the first order optimality of the x-update, we also have $\left\langle \nabla_1 F(\bar{x}, \bar{y}) - \bar{\lambda}, x - \bar{x} \right\rangle \geq 0, \forall x \in \mathcal{X}$. Thus, $\left\langle \nabla F(\bar{x}), \frac{x-\bar{x}}{\|x-\bar{x}\|} \right\rangle \geq 0$ which indicates that $\bar{x}$ is a stationary point of F. □

We will now prove the convergence rate result for Algorithm 2. We will show that the sequence of iterates converges to the first order optimality conditions at a sublinear rate, where we say that a two-block pair $(\bar{x}, \bar{y})$ satisfies the first-order optimality conditions for (3.1) when

$$\begin{aligned}
&\langle \nabla_1 F(\bar{x}, \bar{y}), x - \bar{x} \rangle + \langle \nabla_2 F(\bar{x}, \bar{y}), x - \bar{x} \rangle \geq 0, \ \forall x \in \mathcal{X} \\
&\bar{x} - \bar{y} = 0.
\end{aligned} \tag{3.11}$$

Clearly, (3.11) is equivalent to (2.2).

Theorem 3.2. *Let Assumption 2.2 hold with the Lipschitz constant L and let $\mu < \frac{1}{4L}$. Let $\{x^k, y^k, \lambda^k\}$ be the sequence of iterates defined by Algorithm 2, Denote $\hat{g}_k \equiv \min_{1 \leq i \leq k}\{\|x^i - x^{i-1}\| + \|y^i - y^{i-1}\|\}$ and $\bar{\mathcal{I}}_k \equiv \{(x^i, y^i) : \|x^i - x^{i-1}\| + \|y^i - y^{i-1}\| = \hat{g}_k, \ 1 \leq i \leq k\}$. Then for any fixed $x \in \mathcal{X}$, which is not a stationary point of* (3.1)*, and any k, one (and only one) of the following is true:*

1. $\hat{g}_k = 0$, and (3.11) *is satisfied by $\bar{x} = x^k$ and $\bar{y} = y^k$.*

2. $\hat{g}_k > 0$. Consider a subsequence $\{\hat{x}_k, \hat{y}_k\}$, where $(\hat{x}_k, \hat{y}_k) \in \bar{\mathcal{I}}_k$. Then $\nabla F(\hat{x}_k, \hat{y}_k)$ satisfies first-order optimality conditions (3.11) *with an error, which converges to zero at a sublinear rate of $\mathcal{O}(\frac{1}{\sqrt{k}})$.*

Proof. Consider the sequence of two-block variables $\{x^k, y^k\}$.

If $\hat{g}_k = 0$, then it is obvious that a limit point is obtained, and it is a stationary point by Theorem 2.1.

Now suppose $\hat{g}_k > 0$, and we aim to bound each condition of (3.11) and show that each condition is satisfied with an error which converges to zero at the rate of $\mathcal{O}(\frac{1}{\sqrt{k}})$.

First, we bound the change of primal variables in $\bar{\mathcal{I}}_k$ by the change of augmented Lagrangian

function value. From (3.9), we have

$$\begin{aligned}
&\mathcal{L}_A(x^0, y^0; \lambda^0) - \mathcal{L}_A(x^*, y^*; \lambda^*) \\
&\geq \mathcal{L}_A(x^0, y^0; \lambda^0) - \mathcal{L}_A(x^k, y^k; \lambda^k) \\
&\geq \textstyle\sum_{i=1}^k (\frac{1}{2\mu} - 2\mu L^2)\|x^i - x^{i-1}\|^2 + \sum_{i=1}^k (\frac{1}{2\mu} - 2\mu L^2)\|y^i - y^{i-1}\|^2 \\
&\geq \tfrac{1}{2} k\delta \left(\|x^i - x^{i-1}\| + \|y^i - y^{i-1}\|\right)^2 \\
&= \tfrac{1}{2} k\delta \hat{g}_k^2,
\end{aligned} \tag{3.12}$$

where $\delta = \frac{1}{2\mu} - 2\mu L^2$. Since we choose $0 < \mu \leq \frac{1}{4L}$, $\delta > 0$.

Thus, we have

$$\hat{g}_k \leq \sqrt{\frac{2(\mathcal{L}_A(x^0, y^0; \lambda^0) - \mathcal{L}_A(x^*, y^*; \lambda^*))}{k\delta}}. \tag{3.13}$$

Now we bound each condition of (3.11) by (3.13). From the x-update, we have $\left\langle \nabla_1 \mathcal{L}(x^k, y^{k-1}; \lambda^{k-1}), x - x^k \right\rangle$ $0,\ \forall x \in \mathcal{X}$, which is written as

$$\left\langle \nabla_1 F(x^k, y^{k-1}) + \nabla_2 F(x^k, y^k) + \frac{1}{\mu}(y^k - y^{k-1}), x - x^k \right\rangle \geq 0,\ \forall x \in \mathcal{X}. \tag{3.14}$$

Here we use the fact that $\nabla_1 \mathcal{L}(x^k, y^{k-1}; \lambda^{k-1}) = \nabla_1 F(x^k, y^{k-1}) - \lambda^{k-1} + \frac{1}{\mu}(x^k - y^{k-1})$ and $\lambda^{k-1} = \lambda^k + \frac{1}{\mu}(x^k - y^k) = -\nabla_2 F(x^k, y^k) + \frac{1}{\mu}(x^k - y^k)$.

Similarly to (3.10), we now bound $\left\langle \nabla_1 F(x^k, y^k), x - x^k \right\rangle + \left\langle \nabla_2 F(x^k, y^k), x - x^k \right\rangle$ as following

$$\begin{aligned}
&\left\langle \nabla_1 F(x^k, y^k), x - x^k \right\rangle + \left\langle \nabla_2 F(x^k, y^k), x - x^k \right\rangle \\
&= \left\langle \nabla_1 F(x^k, y^{k-1}) + \nabla_2 F(x^k, y^k) + \tfrac{1}{\mu}(y^k - y^{k-1}), x - x^k \right\rangle \\
&+ \left\langle \nabla_1 F(x^k, y^k) - \nabla_1 F(x^k, y^{k-1}), x - x^k \right\rangle - \left\langle \tfrac{1}{\mu}(y^k - y^{k-1}), x - x^k \right\rangle \\
&\geq -(L + \tfrac{1}{\mu})\|x - x^k\|\|y^k - y^{k-1}\|.
\end{aligned} \tag{3.15}$$

Thus, it follows that

$$\left\langle \nabla_1 F(x^k, y^k) + \nabla_2 F(x^k, y^k), \frac{x - x^k}{\|x - x^k\|} \right\rangle \geq -(L + \frac{1}{\mu})\|y^k - y^{k-1}\|. \tag{3.16}$$

Consider a subsequence $\{(\hat{x}_k, \hat{y}_k)\}$, where $(\hat{x}_k, \hat{y}_k) \in \bar{\mathcal{I}}_k$. From (3.16), we have

$$\left\langle \nabla_1 F(\hat{x}_k, \hat{y}_k) + \nabla_2 F(\hat{x}_k, \hat{y}_k), \frac{x - \hat{x}_k}{\|x - \hat{x}_k\|} \right\rangle \geq -(L + \frac{1}{\mu})\hat{g}_k \geq -(L + \frac{1}{\mu})\sqrt{\frac{2(\mathcal{L}_A(x^0, y^0; \lambda^0) - \mathcal{L}_A(x^*, y^*; \lambda^*))}{k\delta}}. \tag{3.17}$$

On the other hand, for any k, we have $x^k - y^k = -\mu(\lambda^k - \lambda^{k-1})$, which combined with (3.5) implies that

$$\|x^k - y^k\| = \mu\|\lambda^k - \lambda^{k-1}\| \leq \mu L \left(\|x^k - x^{k-1}\| + \|y^k - y^{k-1}\|\right). \tag{3.18}$$

(3.18) indicates that, for $(\hat{x}_k, \hat{y}_k) \in \bar{\mathcal{I}}_k$, we have

$$\|\hat{x}_k - \hat{y}_k\| \leq \bar{\mu} L \hat{g}_k \leq \bar{\mu} L \sqrt{\frac{2(\mathcal{L}_A(x^0, y^0; \lambda^0) - \mathcal{L}_A(x^*, y^*; \lambda^*))}{k\delta}}. \tag{3.19}$$

(3.17) and (3.19) demonstrate that sequences x_k, y_k and $\nabla F(\hat{x}_k, \hat{y}_k)$ satisfy first-order optimality conditions with errors that converge to zero at a sublinear rate $\mathcal{O}(\frac{1}{\sqrt{k}})$. Thus, our proof is complete. □

Algorithm 3 Alternating linearization method (ALM)

1. Choose $\mu_1^0 = \mu_2^0 = \mu^0$, and $x^0 = y^0$;
2. for $k = 0, 1, ...$
   (a) $x^{k+1} := \arg\min_{x\in\mathcal{X}} Q^1_{\mu_1^k}(x, y^k)$; choose μ_1^{k+1} such that (4.1) holds;
   (b) $y^{k+1} := \arg\min_{y\in\mathcal{X}} Q^2_{\mu_2^k}(x^{k+1}, y)$; choose μ_2^{k+1} such that (4.2) holds;

When $\mathcal{X} = \mathbb{R}^n$, by choosing $x = \hat{x}_k - \frac{\nabla F(\hat{x}_k,\hat{y}_k)}{\|\nabla F(\hat{x}_k,\hat{y}_k)\|}$ in (3.17) we have

$$\|\nabla F(\hat{x}_k, \hat{y}_k)\| \leq (L + \frac{1}{\mu})\sqrt{\frac{2(\mathcal{L}_A(x^0, y^0; \lambda^0) - \mathcal{L}_A(x^*, y^*; \lambda^*))}{k\delta}}, \tag{3.20}$$

which recovers the standard complexity result in the unconstrained case, i.e. it takes $\mathcal{O}(\frac{1}{\epsilon^2})$ iterations for the norm of $\nabla F(\hat{x}_k, \hat{y}_k)$ to reach a value below ϵ.

4 Alternating linearization method

In this section we consider alternating linearization method (ALM), which is closely related to ADMM. The relationship between the two methods in the composite convex setting is derived in [12]. In this paper, we discuss alternating linearization method in a nonconvex setting where the objective may not be decomposable.

Recall the notations in (2.7). Suppose that both sequences $\{x^k\}$ and $\{y^k\}$ are bounded, and that the function F can be bounded locally from above by the following approximation term with sufficiently small μ:

$$Q^1_\mu(x, y^k) = F_1(x, y^k) + \left\langle \nabla_2 F(y^k), x - y^k \right\rangle + \frac{1}{2\mu}||x - y^k||_2^2.$$

Formally put, the following condition holds for sufficiently small μ:

$$F(x^{k+1}) \leq Q^1_\mu(x^{k+1}, y^k), \tag{4.1}$$

where $x^{k+1} := \arg\min_{x\in\mathcal{X}} Q^1_{\mu_1^k}(x, y^k)$. Similarly,

$$F(y^{k+1}) \leq Q^2_\mu(x^{k+1}, y^{k+1}) \tag{4.2}$$

holds for small enough μ, where $y^{k+1} := \arg\min_{y\in\mathcal{X}} Q^2_{\mu_2^k}(x^{k+1}, y)$. In the next section, it will be shown through simple expansion that our assumption holds for the second-order least-squares case. Furthermore, such μ can be found by backtracking at each iteration. The basic ALM algorithm is presented in Algorithm 3 and the backtracking version is in Algorithm 4

The convergence analysis of ALM is similar as the analysis in the Iterative Shrinkage-Thresholding Algorithm (ISTA and FISTA) by Beck and Teboulle [4], Alternating Linearization Method (ALM) in the convex setting by Goldfarb *et al.* [12] and Block Coordinate Decent (BCD) method by Xu and Yin [31]. However, all of the above results apply only to the convex domain, which cannot be easily embedded into the case of problem (1.2). Moreover, unlike the ALM in [12], we do not have a composite function form where the linearly separable structure can be taken advantage of (see [12] for details to apply ALM to composite convex functions).

The descent step of Algorithm 3 can be guaranteed by the following result.

Lemma 4.1. *Let the objective function and its block-wise representation be defined as* (2.1), (2.4) *and* (2.3). *Define*

$$\hat{x} := \arg\min_{x\in\mathcal{X}} Q_\mu^1(x,u) \equiv \arg\min_{x\in\mathcal{X}} F_1(x,u) + \langle \nabla_2 F(u,u), x-u\rangle + \frac{1}{2\mu}||x-u||^2,$$

where $u \in \mathbb{R}^n$. *Suppose the following condition holds*

$$F(\hat{x}) \le Q_\mu^1(\hat{x},u). \tag{4.3}$$

Then we have

$$F(u) - F(\hat{x}) \ge \frac{1}{2\mu}(||\hat{x}-u||^2). \tag{4.4}$$

Similarly, define

$$\hat{y} := \arg\min_{y\in\mathcal{Y}} Q_\mu^2(u,y) \equiv \arg\min_{y\in\mathcal{Y}} F_2(u,y) + \langle \nabla_1 F(u,u), y-u\rangle + \frac{1}{2\mu}||y-u||^2,$$

where $u \in \mathbb{R}^n$. *Suppose the following condition holds*

$$F(\hat{y}) \le Q_\mu^2(u,\hat{y}). \tag{4.5}$$

Then we have

$$F(u) - F(\hat{y}) \ge \frac{1}{2\mu}(||\hat{y}-u||^2). \tag{4.6}$$

Proof. From the condition (4.3), we have

$$\begin{aligned} F(u) - F(\hat{x}) &\ge F(u) - Q_\mu^1(\hat{x},u) \\ &= F(u) - (F_1(\hat{x},u) + \langle \nabla_2 F(u,u), \hat{x}-u\rangle + \tfrac{1}{2\mu}||\hat{x}-u||^2). \end{aligned} \tag{4.7}$$

Since $F_1(x,u)$ is convex for any given u, we have

$$F(u) \ge F_1(\hat{x},u) + \langle \nabla_1 F(\hat{x},u), u-\hat{x}\rangle . \tag{4.8}$$

It follows that

$$\begin{aligned} F(u) - F(\hat{x}) &\ge F(u) - Q_\mu^1(\hat{x},u) \\ &\ge F_1(\hat{x},u) + \langle \nabla_1 F(\hat{x},u), u-\hat{x}\rangle - Q_\mu^1(\hat{x},u) \\ &= \langle \nabla_1 F(\hat{x},u) + \nabla_2 F(u,u), u-\hat{x}\rangle - \tfrac{1}{2\mu}||\hat{x}-u||^2. \end{aligned} \tag{4.9}$$

Since the subproblem is solved for optimality, it satisfies the first-order optimality condition:

$$\left\langle \nabla_1 Q_\mu^1(\hat{x},u), x-\hat{x}\right\rangle \ge 0,\ \forall x \in \mathcal{X},$$

which means that this condition holds for $x = u$:

$$\left\langle \nabla_1 Q_\mu^1(\hat{x},u), u-\hat{x}\right\rangle \ge 0,$$

i.e.

$$\left\langle \nabla_1 F(\hat{x},u) + \nabla_2 F(u,u) + \frac{1}{\mu}(\hat{x}-u), u-\hat{x}\right\rangle \ge 0. \tag{4.10}$$

From (4.9) and (4.10), we have

$$\begin{aligned} F(u) - F(\hat{x}) &\geq \langle \nabla_1 F(\hat{x}, u) + \nabla_2 F(u, u), u - \hat{x} \rangle - \tfrac{1}{2\mu}||\hat{x} - u||^2 \\ &\geq -\tfrac{1}{\mu}(\hat{x} - u)^T(u - \hat{x}) - \tfrac{1}{2\mu}||\hat{x} - u||^2 \\ &= \tfrac{1}{2\mu}||\hat{x} - u||^2. \end{aligned} \tag{4.11}$$

Similarly, the other part of the proof follows.

□

The following result shows the convergence of the primal function value for Algorithm 3.

Theorem 4.1. *Let the objective function, its block-wise representation and its approximation be defined as* (2.1), (2.3), (2.4), *and* (2.7), *respectively. Let Assumption 2.2 hold with the Lipschitz constant L. Suppose at iteration k, μ_1^k, μ_2^k are chosen to satisfy* (4.1) *and* (4.2), *respectively, and are bounded from zero by some constant $\tau > 0$. Then, by Algorithm 3, the sequence $\{F(x^k)\}$ converges to $F(x^*)$, where x^* is a stationary point of F. Furthermore, denote $\hat{g}_k \equiv \min_{1 \leq i \leq k} ||x^i - y^{i-1}||$ and $\bar{\mathcal{I}}_k \equiv \{x^i : ||x^i - y^{i-1}|| = \hat{g}_k,\ 1 \leq i \leq k\}$. Then one and only one of the following is true:*

1. $\hat{g}_k = 0$, and the limit point is obtained;

2. $\hat{g}_k > 0$. Consider a subsequence $\{\hat{x}_k\}$, where $\hat{x}_k \in \bar{\mathcal{I}}_k$. Then $\nabla F(\hat{x}_k)$ satisfies first-order optimality condition (2.2) *with an error that converges to zero at a sublinear rate $\mathcal{O}(\frac{1}{\sqrt{k}})$.*

Proof. From Algorithm 3 we have

$$x^k := \arg\min_{x \in \mathcal{X}} Q^1_{\mu_1^k}(x, y^{k-1})$$

for $k \geq 1$.

Let $u = y^{k-1}$ in Lemma 4.1. Then we have

$$2\mu_1^k(F(y^{k-1}) - F(x^k)) \geq ||x^k - y^{k-1}||^2 \geq 0. \tag{4.12}$$

Similarly, for the subproblem updating y, we have

$$2\mu_2^k(F(x^k) - F(y^k)) \geq ||x^k - y^k||^2 \geq 0. \tag{4.13}$$

Hence,

$$F(x^{k-1}) \geq F(x^k) \text{ and } F(y^{k-1}) \geq F(y^k), \quad \text{for all } k. \tag{4.14}$$

Now we have a non-increasing sequence $F(x^k)$ bounded from below. Thus, $\{F(x^k)\}$ converges. Since the left hand sides of (4.12) and (4.13) go to zero when $k \to \infty$, we have $x^k - y^k \to 0$, and also

$$x^k - x^{k-1} \to 0. \tag{4.15}$$

If it happens that $y^{k-1} = \arg\min_{x \in \mathcal{X}} Q^1_{\mu_1^k}(x, y^{k-1})$, then it naturally follows that $x^k = y^{k-1}$ is a stationary point. If $y^{k-1} \neq \arg\min_{x \in \mathcal{X}} Q^1_{\mu_1^k}(x, y^{k-1})$, since x^k is a minimizer of $Q^1_{\mu_1^k}(x, y^{k-1})$, it satisfies the foliowing first-order condition:

$$\left\langle \nabla_1 F(x^k, y^{k-1}) + \nabla_2 F(y^{k-1}, y^{k-1}) + \frac{1}{\mu_1^k}(x^k - y^{k-1}), x - x^k \right\rangle \geq 0,\ \forall x \in \mathcal{X}. \tag{4.16}$$

Hence, we have

$$\begin{aligned}
&\left\langle \nabla_1 F(x^k, x^k) + \nabla_2 F(x^k, x^k), x - x^k \right\rangle \\
&= \left\langle \nabla_1 F(x^k, y^{k-1}) + \nabla_2 F(y^{k-1}, y^{k-1}) + \tfrac{1}{\mu_1^k}(x^k - y^{k-1}), x - x^k \right\rangle \\
&+ \left\langle \nabla_1 F(x^k, x^k) - \nabla_1 F(x^k, y^{k-1}), x - x^k \right\rangle \\
&+ \left\langle \nabla_2 F(x^k, x^k) - \nabla_2 F(y^{k-1}, y^{k-1}), x - x^k \right\rangle - \left\langle \tfrac{1}{\mu_1^k}(x^k - y^{k-1}), x - x^k \right\rangle.
\end{aligned} \tag{4.17}$$

As $x^k - x^{k-1} \to 0$, $\nabla_1 F(x^k, x^k) - \nabla_1 F(x^k, x^{k-1}) \to 0$ since $||\nabla_1 F(x^k, x^k) - \nabla_1 F(x^k, y^{k-1})||_2 \leq L||x^k - y^{k-1}||$. Similarly, $\nabla_2 F(x^k, x^k) - \nabla_2 F(y^{k-1}, y^{k-1}) \to 0$. That, combined with (4.15) and (4.17), leads to

$$\left\langle \nabla_1 F(x^k, x^k) + \nabla_2 F(x^k, x^k), x - x^k \right\rangle \geq 0, \ k \to \infty \ \forall x \in \mathcal{X}.$$

Thus, $\{F(x^k)\}$ converges to $F(x^*)$, where x^* is a stationary point.

Furthermore, from (4.17) it leads to

$$\begin{aligned}
&\left\langle \nabla_1 F(x^k, x^k) + \nabla_2 F(x^k, x^k), x - x^k \right\rangle \\
&\geq -(2L + \tfrac{1}{\mu_1^k})||x - x^k|| \cdot ||x^k - y^{k-1}||,
\end{aligned} \tag{4.18}$$

where L is a Lipschitz constant for the gradients of F_1, F_2. For any $x \neq x^k$ and $x \in \mathcal{X}$, we can bound $\left\langle \nabla_1 F(x^k) + \nabla_2 F(x^k), \frac{x - x^k}{\|x - x^k\|} \right\rangle$ from below by $-(2L + \frac{1}{\mu_1^k})||x^k - y^{k-1}||$.

Denote $\mu^{max} = \max_{1 \leq i \leq k}\{\mu_1^i\}, \mu^{min} = \min_{1 \leq i \leq k}\{\mu_1^i\}$. From monotonicity of the function value (4.12) and (4.13), we have

$$F(x^{i-1}) - F(x^i) \geq F(y^{i-1}) - F(x^i) \geq \frac{1}{2\mu_1^i}\|x^i - y^{i-1}\|^2,$$

and further

$$\begin{aligned}
&F(x^0) - F(x^*) \\
&\geq F(x^0) - F(x^k) \\
&\geq \textstyle\sum_{i=1}^k \frac{1}{2\mu_1^i}||x^i - y^{i-1}||^2 \\
&\geq \textstyle\sum_{i=1}^k \frac{1}{2\mu^{max}}||x^i - y^{i-1}||^2 \\
&\geq \tfrac{k}{2\mu^{max}}(\hat{g}_k)^2.
\end{aligned} \tag{4.19}$$

Thus, we have

$$\hat{g}_k \leq \sqrt{\frac{2\mu^{max}(F(x^0) - F(x^*))}{k}}. \tag{4.20}$$

Consider a subsequence $\{\hat{x}_k\}$, where $\hat{x}_k \in \bar{\mathcal{I}}_k$. Combining (4.18) and (4.20), we come to the following inequality for any $x \in \mathcal{X}$:

$$\begin{aligned}
\left\langle \nabla_1 F(\hat{x}_k, \hat{x}_k) + \nabla_2 F(\hat{x}_k, \hat{x}_k), \tfrac{x - \hat{x}_k}{\|x - \hat{x}_k\|} \right\rangle &\geq -(2L + \tfrac{1}{\mu^{min}})\hat{g}_k \\
&\geq -(2L + \tfrac{1}{\mu^{min}})\sqrt{\tfrac{2\mu^{max}(F(x^0) - F(x^*))}{k}}.
\end{aligned} \tag{4.21}$$

While the right-hand-side of (4.21) goes to zero at $\mathcal{O}(\frac{1}{\sqrt{k}})$, it follows that $\nabla F(\hat{x}_k)$ converges to first-order optimality at a sublinear rate.

Algorithm 4 Alternating linearization method with backtracking (ALM)

1. Choose $\mu_1^0 = \mu_2^0 = \mu^0$, and $x^0 = y^0$;
2. for $k = 0, 1, ...$
   (a) $x^{k+1} := \arg\min_{x\in\mathcal{X}} Q^1_{\mu^k}(x, y^k)$;
   (b) if $F(x^{k+1}) \le Q^1_{\mu_1^k}(x^{k+1}, y^k)$ then
   $\mu_1^{k+1} := \mu_1^k$;
   else
   find the smallest n s.t. $\bar{\mu} := \mu_1^k\beta^n, \bar{x} := \arg\min_{x\in\mathcal{X}} Q^1_{\bar{\mu}}(x, y^k)$ and $F(\bar{x}) \le Q^1_{\bar{\mu}}(\bar{x}, y^k)$;
   $\mu_1^{k+1} := \mu_1^k\beta^n$, $x^{k+1} := \arg\min_x Q^1_{\mu_1^{k+1}}(x, y^k)$;
   (c) $y^{k+1} := \arg\min_{y\in\mathcal{X}} Q^2_{\mu_2^k}(x^{k+1}, y)$;
   (d) if $F(y^{k+1}) \le Q^2_{\mu_2^k}(x^{k+1}, y^{k+1})$ then
   $\mu_2^{k+1} := \mu_2^k$;
   else
   find the smallest n s.t. $\bar{\mu} := \mu_2^k\beta^n, \bar{y} := \arg\min_{y\in\mathcal{X}} Q^2_{\bar{\mu}}(x^{k+1}, y)$ and $F(\bar{y}) \le Q^2_{\bar{\mu}}(x^{k+1}, \bar{y})$;
   $\mu_2^{k+1} := \mu_2^k\beta^n$, $y^{k+1} := \arg\min_{y\in\mathcal{X}} Q^2_{\mu_2^{k+1}}(x^{k+1}, y^{k+1})$.

When $\mathcal{X}$ is $\mathbb{R}^n$, by choosing $x = \hat{x}_k - \frac{\nabla F(\hat{x}_k)}{\|\nabla F(\hat{x}_k)\|}$, we have

$$\|\nabla F(\hat{x}_k)\| \quad \le (2L + \tfrac{1}{\mu^{min}})\sqrt{\tfrac{2\mu^{max}(F(x^0)-F(x^*))}{k}}, \tag{4.22}$$

which recovers the standard complexity result in the unconstrained case, i.e. it takes $\mathcal{O}(\frac{1}{\sqrt{k}})$ iterations for the norm of $\nabla F(\hat{x}_k)$ to go to zero.

□

4.1 Practical ALM with backtracking and skipping

In the previous section we discuss the alternating linearization framework. At each iteration, the sufficient function decrease is obtained by the condition that the resulting function value is no larger than the approximation function value. In the application that we are interested in, a prox parameter μ can be conveniently found so that such condition can be satisfied.

However, in some cases, finding μ that satisfies function reduction conditions may be time consuming, especially when one of the two block gradients has a large Lipschitz constant. When this happens to only one block, we can simply skip the step related to that block and still obtain the convergence. This idea was first proposed in [12] for convex composite case.

When skipping step is constantly applied to one block, we have what we call a partial linearization method (PLM), presented in Algorithm 5. The term "partial" is in a sense that only one block of variables is linearized. In the next section, we will give a specific interpretation when the second-order least-squares problem is considered.

4.2 Connection between ALM and ADMM.

Let us establish the connection between Algorithms 2 (ADMM) and 3 (ALM). Both algorithm perform optimization of two convex functions on each step. In particular, given y^k and λ^k ADMM

Algorithm 5 Partial linearization method (PLM)

1. Choose μ^0, $\beta \in (0,1)$ and x^0;
2. for $k = 0, 1, ...$
   (a) $x^{k+1} := \arg\min_{x\in\mathcal{X}} Q^1_{\mu^k}(x, x^k)$;
   (b) if $F(x^{k+1}) \le Q^1_{\mu^k}(x^{k+1}, x^k)$ then
   $\mu^{k+1} := \mu^k$;
   else
   find the smallest n s.t. $\bar{\mu} := \mu^k\beta^n$, $\bar{x} := \arg\min_{x\in\mathcal{X}} Q^1_{\bar{\mu}}(x, x^k)$ and $F(\bar{x}) \le Q^1_{\bar{\mu}}(\bar{x}, x^k)$;
   $\mu^{k+1} := \mu^k\beta^n$, $x^{k+1} := \arg\min_{x\in\mathcal{X}} Q^1_{\mu^{k+1}}(x, x^k)$; go to (b).

optimizes

$$\mathcal{L}_A(x, y^k; \lambda^k) = F_1(x, y^k) - \left\langle \lambda^k, x - y^k \right\rangle + \frac{1}{2\mu}\|x - y^k\|^2,$$

while ALM optimizes

$$Q^1_\mu(x, y^k) = F_1(x, y^k) + \left\langle \nabla_2 F(y^k), x - y^k \right\rangle + \frac{1}{2\mu}\|x - y^k\|^2.$$

Hence the two steps are identical if $\lambda^k = -\nabla_2 F(y^k) \equiv -\nabla_2 F(y^k, y^k)$ and then same value of μ is used. Consider the optimality conditions satisfied by y^k, since it is an unconstrained minimizer of

$$\mathcal{L}_A(x^k, y; \lambda^{k-1}) = F_2(x^k, y) - \left\langle \lambda^{k-1}, y - x^k \right\rangle + \frac{1}{2\mu}\|x^k - y\|^2,$$

which is

$$\nabla_y F_2(x^k, y^k) + \lambda^{k-1} + \frac{1}{\mu}(y^k - x^k) = 0,$$

Hence, from the update rule of λ^k, we have

$$\lambda^k = \lambda^{k-1} - \frac{1}{\mu}(x^k - y^k) = -\nabla_y F_2(x^k, y^k) \equiv -\nabla_2 F(x^k, y^k).$$

Similarly, if the order of subproblems is reversed in Algorithms 2, and if $\mathcal{X} = \mathbb{R}^n$, then

$$\lambda^k = -\nabla_1 F(x^k, y^k).$$

Hence, ADMM can be regarded as an inexact version of ALM, where λ is not updated as the exact partial gradient but some "mixture" of gradient information. In practice, the update of multipliers in ADMM is less costly than computing the gradient information in ALM, but as a results ADMM does not guarantee a descent step. In our analysis this difference inflicts limitation on ADMM convergence results to the case of constant, sufficiently small parameter μ, while in the case of ALM we were able to analyze the case where μ^k is a variable parameter chosen to satisfy sufficient decrease condition.

4.3 The connection between A/PLM and Levenberg-Marquardt method.

If we consider problem (1.2) simply as a nonlinear least squares problem, then we can apply classical methods such as Gauss-Newton method and Levenberg-Marquardt (LM) method [30]. By exploiting the structure of the least-squares problems, these methods compute exact gradient and partial Hessian information. In particular, they approximate the Hessian using Jacobian of the constraints (the functions inside the squares) and omit the second order information from those functions, since in many cases the contribution of this second order information is negligible compared with the Jacobian induced Hessian approximation. In this section, we relate these classic methods and our alternating linearization schemes, in the second-order least-squares setting. We show that our ALM method is a simplified version of the LM method.

We will consider the least-squares problem (1.2) instead of the more general form (1.4) for simplicity of notation. We can write (1.2) as follows:

$$\min_{x\in\mathcal{X}} F = \sum_{i=1}^{m} r_i^2(x) = \sum_{i=1}^{m} (x^T M_i x - c_i)^2, \tag{4.23}$$

where $r(x)$ is a residual vector and $r_i(x) = x^T M_i x - c_i$ is the ith residual. Then the Jacobian of r function is:

$$\begin{aligned} J(x) &= \begin{bmatrix} (\frac{\partial r_1(x)}{\partial x})^T \\ \vdots \\ (\frac{\partial r_m(x)}{\partial x})^T \end{bmatrix} \\ &= \begin{bmatrix} (M_1 x + M_1^T x)^T \\ \vdots \\ (M_m x + M_m^T x)^T \end{bmatrix}. \end{aligned} \tag{4.24}$$

Both Gauss-Newton method and Levenberg-Marquardt method store the Jacobian matrix after each iteration and compute the gradient as

$$\begin{aligned} \nabla F(x) &= J(x)^T r(x) \\ &= \textstyle\sum_{i=1}^{m} (x^T M_i x - c_i)(M_i x + M_i^T x). \end{aligned} \tag{4.25}$$

Furthermore, the Hessian can be written explicitly as

$$\begin{aligned} \nabla^2 F(x) &= J(x)^T J(x) + \textstyle\sum_i r_i(x) \nabla^2 r_i(x) \\ &= \begin{bmatrix} (M_1 + M_1^T)x & \dots & (M_n + M_n^T)x \end{bmatrix} \begin{bmatrix} (M_1 x + M_1^T x)^T \\ \vdots \\ (M_m x + M_m^T x)^T \end{bmatrix} + \textstyle\sum_{i=1}^{m} (x^T M_i x - c_i)(M_i + M_i^T). \end{aligned} \tag{4.26}$$

In the case of both, Gauss-Newton and Levenberg-Marquardt, methods the second term in (4.26) is ignored. This gives an efficient and relatively accurate approximation of the Hessian, in the cases when the magnitude of the second term in (4.26) is significantly smaller than that of the first. In particular, this is the case, when the residual $x^T M_i x - \theta$ is approximately zero for all i. For instance, the risk parity problems (when risk parity exists) belong to this case.

Levenberg-Marquardt method is considered more robust than Gauss-Newton because it adds a positive definite matrix to $J(x)^T J(x)$, (usually an identity matrix multiplied by a scalar) and solves the modified subproblem. Specifically, at each iteration, it solves

$$x^{k+1} = \arg\min_x F(x^k) + \nabla F(x^k)^T(x - x^k) + \frac{1}{2}(x - x^k)^T[J(x)^T J(x)](x^k) + \frac{1}{\mu}I)(x - x^k). \quad (4.27)$$

The relationship between Levenberg-Marquardt method and our alternating linearization method can be analyzed as follows. Recall our partial linearization method (Algorithm 5). At each iteration, the following function is minimized:

$$Q_\mu^1(x) = F(x, x^k) + \nabla_2 F(x^k)^T(x - x^k) + \frac{1}{2\mu}||x - x^k||^2, \quad (4.28)$$

and

$$F(x, x^k) = F(x^k) - \sum_i [(x^k)^T M_i^T (x - x^k)]^2 + 2\sum_i (x^T M_i x^k - c_i)(x^k)^T M_i^T (x - x^k).$$

Note that

$$\begin{aligned} 2\sum_i (x^T M_i x^k - c_i)(M_i x^k) &= 2\sum_i ((x^k)^T M_i x^k - c_i)(M_i x^k) + 2\sum_i ((x - x^k)^T M_i x^k - c_i)(M_i x^k) \\ &= \nabla_1 F(x^k) + 2\sum_i ((x - x^k)^T M_i x^k - c_i)(M_i x^k) \end{aligned}$$

and recall that $\nabla_2 F(x^k) + \nabla_1 F(x^k) = \nabla F(x^k)$. Thus, function $Q_\mu^1(x)$ in (4.28) can be written as

$$Q_\mu^1(x) = F(x^k) + \nabla F(x^k)^T(x - x^k) + G(x),$$

where the first two terms are exactly the same as (4.27), and $G(x)$ contains all the remaining terms of $Q_\mu^1(x)$, and, hence,

$$G(x) = \sum_i (x - x^k)^T [M_i x^k (x^k)^T M_i^T + \frac{1}{2\mu} I](x - x^k).$$

Recall that, in Levenberg-Marquardt method,

$$J(x^k)^T J(x^k) = \sum_i (M_i x^k + M_i^T x^k)(M_i x^k + M_i^T x^k)^T$$

Hence, our linearization approach can be regarded as a cheap variant of Levenberg-Marquardt method when only part of the second-order information is considered.

For alternating linearization method, the analysis is similar, in which case partial Hessian approximations $M_i x^k (x^k)^T M_i^T$ and $M_i^T x^k (x^k)^T M_i$ are used in an alternating manner. Specifically, for each subproblem,

$$\begin{aligned} &Q_{\mu_1}^1(x) = F(y^k) + \nabla F(y^k)^T(x - y^k) + \textstyle\sum_{i=1}^m (x - y^k)^T [M_i y^k (y^k)^T M_i^T + \frac{1}{2\mu_1} I](x - y^k) \\ &Q_{\mu_2}^2(y) = F(x^{k+1}) + \nabla F(x^{k+1})^T(y - x^{k+1}) + \textstyle\sum_{i=1}^m (y - x^{k+1})^T [M_i^T x^{k+1} (x^{k+1})^T M_i + \frac{1}{2\mu_2} I](y - x^{k+1}), \end{aligned}$$

where μ_1, μ_2 are positive scalars.

5 Relaxations and lower bounds

We have presented several algorithms based on alternating direction and variable splitting techniques. All these algorithms are local algorithms, hence they provide no guarantee that a global minimum will be found. Solutions obtained by these methods provide an upper bound of the global minimum. In this section, we apply relaxation schemes in an attempt to find a (possibly) good lower bound of the global optimum. In particular, in the case of risk parity problems, when risk parity exists, the global optimal solution of (1.1) is zero, otherwise it is positive. Assume that our ALM or ADMM methods find a local minimum with a positive value of the objective function. If a lower bound can be computed which is positive, this will provide a certification that a risk parity solution does not exists. Clearly, if the lower bound is equal to the local optimal value, then this certifies that a global optimum has actually been obtained.

5.1 SOS relaxations

The objective function of (1.1) is a quartic polynomial. Hence recent advances in polynomial optimization are generally applicable for our problem (see, for instance, [19] for a review). It is well known that polynomial optimization problem can be reduced to the problem of determining whether a polynomial is nonnegative (we expand on this below). It is also known that a polynomial is nonnegative if it has a sum-of-squares (SOS) decomposition (while the reverse in not true, examples can be seen in [25] for instance). Existence of an SOS for a given polynomial, can, in turn, be determined by solving a semidefinite programming (SDP) feasibility problem, and thus is considered as tractable for small to medium scaled problem [22]. Hence, SOS can be used to compute a global lower bound for a nonconvex optimization problem.

Consider a multivariate polynomial $p(x_1, ..., x_n) \triangleq p(x)$ of degree $2d$. Then $p(x)$ is representable as an SOS if there exists a positive semidefinite matrix Q, such that

$$p(x) = z^T Q z, \tag{5.1}$$

where $z = [1, x_1, x_2, ..., x_n, x_1x_2, ..., x_n^d]^T$. The size of z is $\begin{pmatrix} n+d \\ d \end{pmatrix}$. It can be shown that (5.1) is equivalent to $p(x) = \sum_i f_i^2(x)$, where $f_i(x)$ are polynomials [22]. Note that the matrix Q may not be unique.

Example 5.1. Recall the objective function in our least-squares formulation in risk parity problem: $p(x, \theta) = \sum_{i=1}^n (x^T M_i x - \theta)^2$. In order to have the form (5.1), we can first homogenize the degree of the polynomial by substituting $\theta = \frac{1}{n}\sum_{j=1}^n x^T M_j x$ (more on validity of that choice below). Also, note that, for any $M \in \mathbb{R}^{n\times n}$, there always exists a vector $p \in \mathbb{R}^n$ such that $x^T M x = p^T \bar{x}$, where $\bar{x} = [x_1^2, x_1x_2, ..., x_1x_n, x_2^2, ..., x_n^2]^T$ is a vector of all the monomials of degree 2. The size of $\bar{x}$ is $\begin{pmatrix} n+1 \\ 2 \end{pmatrix}$. Then we have:

$$\begin{aligned} & \sum_{i=1}^n (x^T M_i x - \tfrac{1}{n}\sum_{j=1}^n x^T M_j x)^2 \\ = \; & \sum_{i=1}^n (x^T M_i x)^2 - \tfrac{1}{n}(\sum_{j=1}^n x^T M_j x)^2 \\ = \; & \sum_{i=1}^n \bar{x}^T p_i p_i^T \bar{x} - \tfrac{1}{n}\bar{x}^T(\sum_{j=1}^n p_i)(\sum_{j=1}^n p_i)^T \bar{x} \\ = \; & \bar{x}^T Q \bar{x}, \end{aligned}$$

where $Q = \sum_{i=1}^n p_i p_i^T - \frac{1}{n}(\sum_{j=1}^n p_i)(\sum_{j=1}^n p_i)^T$, and the positive definiteness of Q follows naturally from Cauchy-Schwarz.

All of the problems of form of (1.2)) considered in this paper can be reformulated into (5.1). The existence of an SOS decomposition of a polynomial in n variables of degree $2d$ can be decided efficiently by solving a semidefinite programming feasibility problem [21].

Example 5.2. Consider the following 2×2 risk parity problem. The covariance matrix is given by

$$\Sigma = \begin{bmatrix} 1 & \\ & 4 \end{bmatrix}.$$

Thus, the objective function is given by $F(x) = \sum_{i=1}^2 (x_i(\Sigma x)_i - \frac{1}{2}x^T \Sigma x)^2 = \frac{1}{2}(x_1^2 - 4x_2^2)^2$. Thus, we can write

$$\begin{aligned} F(x) &= \tfrac{1}{2}(x_1^2 - 4x_2^2)^2 \\ &= [x_1^2 \ x_1x_2 \ x_2^2] \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{bmatrix} \\ &= q_{11}x_1^4 + 2q_{12}x_1^3x_2 + (2q_{13} + q_{22})x_1^2x_2^2 + 2q_{23}x_1x_2^3 + q_{33}x_2^4. \end{aligned}$$

Then we can determine an SOS decomposition by equating the corresponding coefficients and thus obtaining the following SDP feasibility problem:

$$\text{Find } Q \succeq 0, \quad \text{s.t. } q_{11} = 1, \ 2q_{13} + q_{22} = -8, \ q_{33} = 16.$$

5.1.1 Global bounds for polynomial functions

SOS technique can be used to compute the global lower bounds for polynomial functions. Consider the following problem:

$$\begin{array}{ll} \min & F(x) \\ \text{s.t.} & x \in \mathcal{X}. \end{array} \tag{5.2}$$

Problem (5.2) is equivalent to

$$\begin{array}{ll} \max & \gamma \\ \text{s.t.} & F(x) - \gamma \geq 0, \quad \forall x \in \mathcal{X}. \end{array} \tag{5.3}$$

Further, (5.3) can be approximated by

$$\begin{array}{ll} \max & \gamma \\ \text{s.t.} & F(x) - \gamma \text{ is SOS}, \quad \forall x \in \mathcal{X}. \end{array} \tag{5.4}$$

Obviously, the optimal value of (5.4) is a lower bound for the global minimum of the original problem (5.2).

If a polynomial is SOS, then it is nonnegative for any x. To take the constraints $x \in \mathcal{X}$ into account and hence to strengthen the lower bound on $F(x)$ the concept of Schmdgen Positivstellensatz can be applied.

Theorem 5.1. *(Schmdgen Theorem, '1991, [27]) Let $K = \{x \in \mathbb{R}^n : g_1(x) \geq 0, ..., g_m(x) \geq 0\}$ be a compact set. If a polynomial $p(x)$ is positive on K, then $p(x) \in P(g_1, ..., g_m)$, where*

$$P(g_1, ..., g_m) = \left\{ \sum_{\nu \in \{0,1\}^m} \sigma_\nu(x) g_1(x)^{\nu_1} ... g_m(x)^{\nu_m} : \text{ each } \sigma_\nu \text{ is SOS} \right\}.$$

Suppose we have the following constrained optimization problem:

$$\begin{array}{ll} \min & F(x) \\ \text{s.t.} & g_i(x) \geq 0, \quad i = 1, ..., m_1 \\ & h_j(x) = 0, \quad j = 1, ..., m_2. \end{array} \tag{5.5}$$

Then a lower bound of $F(x)$ can be computed by:

$$\begin{array}{lll} \max & \gamma & \\ \text{s.t.} & F(x) - \gamma & = \sigma_0(x) + \sigma_1(x) g_1(x) + ... + \sigma_{m_1}(x) g_{m_1}(x) \\ & & + \sigma_{12}(x) g_1(x) g_2(x) + ... + \sigma_{12...m}(x) g_1(x) ... g_{m_1}(x) + \sum_j^{m_2} \lambda_j(x) h_j(x), \end{array} \tag{5.6}$$

where $\sigma_i(x)$'s are a set of SOSs and $\lambda_j(x)$'s are a set of polynomials. Similarly to the unconstrained case, we can find such decomposition by solving an SDP. In fact, the SOSTOOLS that we apply for an implementation for risk parity optimization in the next section solves (5.6) to find a global bounds. We refer interested readers to [23] for more details.

As an alternative of Schmdgen's Positivstellensatz, we can apply Putinar's Positivstellensatz which needs a stronger assumption but also has a stronger conclusion. For a detailed review on Positivstellensatz and its applications, interested readers can refer to [19].

Theorem 5.2. *(Putinar Theorem, '1993, [24]) Let $K = \{x \in \mathbb{R}^n : g_1(x) \geq 0, ..., g_m(x) \geq 0\}$ be a compact set, and the quadratic module be defined as*

$$M(g_1, ..., g_m) = \left\{ \sum_{i=1}^{m} \sigma_i(x) g_i(x) : \text{ each } \sigma_i \text{ is SOS} \right\}.$$

Suppose there exists N such that $N - \|x\|_2^2 \in M(g_1, ..., g_m)$. If $p(x)$ is positive on K, then $p(x) \in M(g_1, ..., g_m)$.

The condition $N - \|x\|_2^2 \in M(g_1, ..., g_m)$ is called archimedean condition. Note that the risk parity optimization problem satisfies the archimedean condition. The reason is that in portfolio optimization the leveraging level is always bounded, which means the weights always satisfy $\sum_{i=1}^n x_i^2 \leq R$ for some large enough R and the archimedean condition holds.

Theorem 5.2 states that, under certain assumptions, a polynomial positive on K can be represented in Putinar's way as long as the degree of each σ_i is high enough. There exist upper bounds on the degree, for instance see [20], however, these bounds usually do not present a practical approach, as high degree SOS problems lead to very large SDPs. In practice, the maximum degree of the SOS is increased sequentially and for each degree an SOS approach is applied and an SDP is solved. This is referred to as increasing the hierarchy of SOS relaxations.

5.2 DSOS and SDSOS optimization: alternatives to SOS optimization

While computing SOS decomposition is generally tractable, the size of resulting SDP problem grows very quickly with the original dimension and the degree of the polynomial. Hence further relaxations, via diagonally dominant sum-of-squares (DSOS) and scaled-diagonally-dominant-sum-of-squares (SDSOS) decompositions, which lead to linear programs and second order cone programs, respectively, have recently been proposed [2]. The cones of polynomials that admit DSOS and SDSOS decompositions are subsets of the cone of SOS polynomials, but they lead to more tractable optimization problems.

A symmetric matrix A is diagonally dominant (dd) if $a_{ii} \geq \sum_{j \neq i} |a_{ij}|$ for all i. A is scaled diagonally dominant (sdd) if there exists an elementwise positive vector y such that $a_{ii} y_i \geq \sum_{j \neq i} |a_{ij}| y_j$ for all i.

Then two subsets of SOS cone, DSOS and SDSOS, are defined as follows.

(Ahmadi and Majumdar, '2013)

A polynomial p is diagonally-dominant-sum-of-squares (DSOS) if it can be written as

$$p = \sum_i \alpha_i m_i^2 + \sum_{i,j} \beta_{i,j} (m_i \pm m_j)^2,$$

for some monomials m_i, m_j and some nonnegative scalars $\alpha_i, \beta_{i,j}$.

A polynomial p is scaled-diagonally-dominant-sum-of-squares (SDSOS) if it can be written as

$$p = \sum_i \alpha_i m_i^2 + \sum_{i,j} (\beta_i m_i \pm \gamma_j m_j)^2,$$

for some monomials m_i, m_j and some constants $\alpha_i \geq 0, \beta_i, \gamma_i$.

Let $DSOS_{n,2d}$, $SDSOS_{n,2d}$, $SOS_{n,2d}$ and $PSD_{n,2d}$, denote cones of, respectively, DSOS, SDSOS, SOS and nonnegative polynomials of degree $2d$ in dimension n. It is clear that $DSOS_{n,2d} \subseteq SDSOS_{n,2d} \subseteq SOS_{n,2d} \subseteq PSD_{n,2d}$. It is shown in [2] that a polynomial p of degree $2d$ is DSOS (or SDSOS) if and only if it has a representation $p(x) = z^T(x) Q z(x)$, where $z(x)$ is the standard monomial vector of degree d, and Q is a DD (or SDD) matrix. It has also been proven that the set of $DSOS_{n,2d}$ (or $SDSOS_{n,2d}$) has a polyhedral (second order cone) representation and thus the search over $DSOS_{n,2d}$ ($SDSOS_{n,2d}$) for a fixed d reduces to a linear programming (second-order cone programming) problem.

In the next section we will show that applying the SOS, DSOS and SDSOS relaxation to the risk parity problem produces useful lower bounds and in the case of SDSOS good lower bounds can be obtained efficiently.

6 Numerical results on risk parity portfolio selection problem

In this section, we use risk parity portfolio selection problem (1.6) (with $q(x) = 0$) as the specific application to perform numerical experiments. It was shown in [3] that if all $a_i = 0$, $b_i = 1$ for all i, then a unique risk parity solution exists. It was further shown in [3] that if $a_i < 0$ for some i then there may be multiple risk parity solutions and in the case when bounds are not tight at the solution, all local minima are global. Tighter box constraints, on the other hand, may result in no risk parity solution and multiple local optima may exist. We will demonstrate on some simple examples, that our alternating direction approaches find the global optimum in each case.

In [3] these methods were used in case of multiple risk parity solutions and regularization term $q(x) = \frac{1}{2}x^T\Sigma x$ and also produced global solutions.

6.1 A comparison of local alternating direction algorithms.

We compare algorithms described in Sections 3 and 4 on randomly generated data and real data sets.

In risk parity problem (1.6) θ is a free variable of dimension one. It is possible to derive simplified versions of ALM and PLM in such a way, that optimization over θ is performed as a separate step after optimization over x. In particular, θ can be updated by the exact minimization at the end of each iteration. We observe that,

$$\frac{\partial F}{\partial \theta} = -2\sum_{i=1}^{n}(x^T M_i x - \theta),$$

which leads to the following simple update $\theta^k = \frac{1}{n}\sum_{i=1}^{n}(x^k)^T M_i x^k$. Note that if such an update is used instead of simultaneous optimization over x and θ, then our ALM scheme performs three optimization steps over three blocks of variables instead of two and our theory may no longer apply. In fact, as our computational results show, updating θ separately leads to inferior results. Optimizing over x and θ simultaneously reduced overall number of iterations, while it does not result in substantially more difficult subproblems. Hence, this modification of PLM and ALM is undesirable both in theory and in practice. We simply include it here for comparison.

For both PLM and ALM, we have shown that the convergence holds if μ^k is chosen so that the function value at the new iterate is not larger than the value of the approximation function Q. Choosing small value of μ^k a priori will guarantee this, but will result in slow progress of the algorithm. Hence, we apply backtracking procedure to find an acceptable value of details about applying backtracking alternating linearization method for convex composite optimization can be found in [26].

In the case of augmented Lagrangian method and alternating direction augmented Lagrangian method (ADMM) μ should be selected sufficiently small and constant, according to our theory. In order to avoid computing the Lipschitz constant of the gradient we allow some parameter tuning to improve the results. In augmented Lagrangian method, we also allow inexact minimization for the subproblem at beginning iterations, to achieve fast convergence (see, e.g. [6]).

In what follows we compare the following algorithms.

1. ALM-θ - Algorithm 3 with separate θ update.
2. ALM - Algorithm 3 with simultaneous optimization over x and θ;
3. PLM-θ - Algorithm 5 with separate θ update;
4. PLM - Algorithm 5 with simultaneous optimization over x and θ;
5. AL-BCD - Augmented Lagrangian method with block coordinate descent method to solve the subproblem;
6. ADMM: Algorithm 2 with properly chosen μ;

Our implementations and experiments were performed in MATLAB R2013a on a laptop with Intel Core Duo 1.8 GHz CPU and 2GB RAM. Mosek 7.0 was applied to solve the QP subproblem. In [3], the basic versions of ALM were shown to be superior to MATLAB fmincon, hence we do not include these comparisons here. In Table 6.1 and Table 6.2, we compare the algorithms on random data. An arbitrary symmetric positive semidefinite matrix can be generated as $\Sigma = AA^T$, where A_{ij} is uniformly distributed within the interval $[0, 1]$. Recall that the main computational cost of each algorithm lies in the number of quadratic models it solves as the subproblem. Hence, in each table we report the number of iterations and the total number of QP solved. We set the termination criterion to when the largest KKT violation falls below ϵ, with ϵ chosen to be 10^{-3}, 10^{-5}, etc. We recorded the number of the iterations and the number of subproblem (QP) solves it took each algorithm to reach this threshhold. The maximum iteration number is 10000.

Table 6.1: *A comparison of algorithms on a randomly generated instance* (20×20). *The starting point is chosen to be equally weighted portfolio, i.e.,* $x_i^0 = 1/n$. *The lower and upper bounds are chosen to be* $a = 0, b = 1$. *The starting* μ *is chosen to be 0.01.*

Algorithm	iter. (10^{-3})	QP	F-value	iter. (10^{-5})	QP	F-value	iter. (10^{-7})	QP	F-value
PLM-θ	18	29	1.06×10^{-8}	32	50	1.28×10^{-12}	45	70	1.44×10^{-16}
PLM	7	7	2.67×10^{-9}	9	9	2.94×10^{-13}	11	11	2.53×10^{-17}
ALM-θ	12	40	2.20×10^{-8}	21	66	1.03×10^{-12}	28	89	1.93×10^{-16}
ALM	5	12	9.29×10^{-10}	6	14	1.11×10^{-12}	8	21	1.18×10^{-19}
AL-BCD	3	52	1.11×10^{-8}	5	94	3.56×10^{-14}	6	116	4.03×10^{-17}
ADMM	19	38	9.43×10^{-9}	33	66	1.27×10^{-12}	48	96	1.02×10^{-16}

Table 6.2: *A comparison of algorithms with fixed steplengths on a randomly generated instance* (200×200). *The starting point is chosen to be equally weighted portfolio, i.e.,* $x_i^0 = 1/n$. *The lower and upper bounds are chosen to be* $a = -1, b = 2$. *The starting* μ *is chosen to be 0.01.*

Algorithm	iter. (10^{-3})	QP	F-value	iter. (10^{-5})	QP	F-value	iter. (10^{-7})	QP	F-value
PLM-θ	15	30	7.28×10^{-10}	27	50	5.65×10^{-14}	44	77	1.54×10^{-17}
PLM	4	4	1.51×10^{-10}	5	5	2.63×10^{-13}	7	7	1.16×10^{-18}
ALM-θ	14	55	7.33×10^{-10}	23	82	2.57×10^{-14}	28	100	2.16×10^{-18}
ALM	3	13	1.66×10^{-12}	4	16	3.53×10^{-17}	5	20	7.36×10^{-18}
AL-BCD	12	790	1.88×10^{-8}	17	1496	3.04×10^{-12}	22	2110	5.55×10^{-17}
ADMM	45	90	6.15×10^{-6}	92	184	9.84×10^{-12}	103	206	1.96×10^{-15}

Tables 6.1 and 6.2 contain results for randomly generated data of two different sizes. We observe that all algorithms find the global minimum. The AL-BCD method requires the least number of iterations, but the largest number of QP solves, since each iteration requires multiple such solves. Both PLM and ALM tend to require fewer QP solves, compared to ADMM and AL-BCD, which is likely the result of using the gradient information backtracking approach for selecting μ^k.

We also compare the algorithms on one data set created to simmulate difficult risk parity cases with 5 risky assets [3]. The covariance matrix of the percentage annual return is given by:

$$\Sigma = \begin{bmatrix} 94.868 & 33.750 & 12.325 & -1.178 & 8.778 \\ 33.750 & 445.642 & 98.955 & -7.901 & 84.954 \\ 12.325 & 98.955 & 117.265 & 0.503 & 45.184 \\ -1.178 & -7.901 & 0.503 & 5.460 & 1.057 \\ 8.778 & 84.954 & 45.184 & 1.057 & 34.126 \end{bmatrix}.$$

In Table 6.3 we present results for the case of lower and upper bounds set to 0 and 1, respectively. In this case risk parity solution exists and is unique [3]. In Table 6.4 we show the results for the case of tight upper and lower bounds, where a risk parity solution, satisfying these bounds, does not exists.

Finally we test the algorithms on real instances of covariance matrices of different sizes using risk parity model (1.6) and group risk parity model (1.7). These results are listed in Tables 6.5-6.7 and also include the case where risk parity does not exist.

All our results show that ALM and PLM are comparable to each other in terms of the number of QP solves and they both typically outperform other methods, such as ALM-θ and PLM-θ, as well as ADMM. The only exception in which ALM-θ outperforms ALM in terms of speed is in Table 6.7. However, ALM achieves a smaller objective function value (1.22×10^{-13}) than ALM-θ (5.41×10^{-13}).

Table 6.3: *A comparison of algorithms on 5×5 instance. The starting point is chosen to be equally weighted portfolio, i.e., $x_i^0 = 1/n$. The lower and upper bounds are chosen to be $a = 0, b = 1$. The starting μ is chosen to be 0.1.*

Algorithm	iter. (10^{-3})	QP solved	F-value	iter. (10^{-5})	QP solved	F-value	Final solution
PLM-θ	55	95	4.87×10^{-9}	77	131	1.34×10^{-12}	[0.125;0.047;0.083;0.613;0.132]
PLM	41	84	1.59×10^{-9}	50	101	5.41×10^{-14}	[0.125;0.047;0.083;0.613;0.132]
ALM-θ	47	174	3.08×10^{-9}	64	230	1.26×10^{-13}	[0.125;0.047;0.083;0.613;0.132]
ALM	20	82	1.65×10^{-9}	26	102	4.52×10^{-14}	[0.125;0.047;0.083;0.613;0.132]
AL-BCD	18	2058	7.04×10^{-13}	23	2744	5.67×10^{-16}	[0.125;0.047;0.083;0.613;0.132]
ADMM	91	182	3.67×10^{-9}	106	212	1.18×10^{-13}	[0.125;0.047;0.083;0.613;0.132]

Table 6.4: *A comparison of algorithms on 5×5 instance. The starting point is chosen to be equally weighted portfolio, i.e., $x_i^0 = 1/n$. The lower and upper bounds are chosen to be $a = 0.05, b = 0.35$. The starting μ is chosen to be 0.1.*

Algorithm	iter. (10^{-3})	QP solved	F-value	iter. (10^{-5})	QP solved	F-value	Final solution
PLM-θ	28	52	16.0344	61	87	16.0344	[0.204;0.060;0.130;0.350;0.256]
PLM	23	46	16.0344	29	55	16.0344	[0.204;0.060;0.130;0.350;0.256]
ALM-θ	19	79	16.0344	26	98	16.0344	[0.204;0.060;0.130;0.350;0.256]
ALM	14	62	16.0344	21	82	16.0344	[0.204;0.060;0.130;0.350;0.256]
AL-BCD	507	15786	16.0344	522	15846	16.0344	[0.204;0.060;0.130;0.350;0.256]
ADMM	309	618	16.0344	325	650	16.0344	[0.204;0.060;0.130;0.350;0.256]

6.2 Implementation of SOS optimization on risk parity optimization problem

In this section, we discuss the application of SOS relaxation and its variants discussed in Section 5 to risk parity. As mentioned above, it has been shown in [3], that all unconstrained local optima of

Table 6.5: *A comparison of algorithms on on asset allocation instance (14×14). The starting point is chosen to be equally weighted portfolio, i.e., $x_i^0 = 1/n$. The lower and upper bounds are chosen to be $a = 0, b = 1$. The starting μ is chosen to be 1.*

Algorithm	iter. (10^{-3})	QP solved	F-value	iter. (10^{-5})	QP solved	F-value	Final solution
PLM-θ	127	212	8.61×10^{-8}	210	343	7.78×10^{-12}	
PLM	131	225	7.15×10^{-8}	215	363	8.62×10^{-12}	$x_1 \sim x_3$: 0.0416, 0.0360, 0.032
ALM-θ	31	111	2.30×10^{-8}	45	159	2.95×10^{-12}	$x_4 \sim x_6$: 0.0313, 0.0334, 0.068
ALM	10	58	3.40×10^{-8}	14	69	5.82×10^{-12}	$x_7 \sim x_9$: 0.0249, 0.0506, 0.166
AL-BCD	42	1276	1.19×10^{-10}	44	1676	3.11×10^{-14}	$x_{10} \sim x_{12}$: 0.2892, 0.0630, 0.053
ADMM	85	170	2.88×10^{-8}	106	212	6.26×10^{-13}	$x_{13} \sim x_{14}$: 0.0760, 0.0332

Table 6.6: *A comparison of algorithms on asset allocation instance with tight bounds (14×14). The starting point is chosen to be equally weighted portfolio, i.e., $x_i^0 = 1/n$. The lower and upper bounds are chosen to be $a = 0.06, b = 0.08$. The starting μ is chosen to be 1.*

Algorithm	iter. (10^{-3})	QP solved	F-value	iter. (10^{-5})	QP solved	F-value	Final solution
PLM-θ	11	24	0.06645	17	32	0.06645	
PLM	11	24	0.06645	14	29	0.06645	$x_1 \sim x_3$: 0.0594, 0.0500, 0.0500
ALM-θ	7	34	0.06645	8	37	0.06645	$x_4 \sim x_6$: 0.0500, 0.0500, 0.0984
ALM	4	25	0.06645	5	27	0.06645	$x_7 \sim x_9$: 0.0500, 0.0715, 0.1000
AL-BCD	772	14432	0.06645	1211	16188	0.06645	$x_{10} \sim x_{12}$: 0.1000, 0.0957, 0.0796
ADMM	980	1960	0.06645	1508	3016	0.06645	$x_{13} \sim x_{14}$: 0.1000, 0.0500

(1.6) are actually risk parity solutions. However, if a local solution has an active bound constraint, then this may be a local solution, as seen in some examples in Section 6.1, and it is unclear if it also is a global one. Here, we can apply SOS techniques to find a global lower bounds for such cases.

We use SOSTOOLS MATLAB toolbox for constructing and solving SOS relaxation [23]. SOSTOOLS reformulate SOSs as semidefinite programs (SDPs), which is then solved by a standard SDP solver such as SeDuMi or SDPT3.

Example 6.1. Consider the 5×5 example introduced in Section 6.1. As shown in Table 6.4, by imposing sufficiently tight bound constraints, a risk parity solution is not reached. We now would like to verify that risk parity solution does not indeed exist, and, if possible, verify the global optimality of our local solution by constructing a lower bound. We used function `findbound` in SOSTOOLS. In this case, after reformulation, SDPT3 solved an SDP feasibility problem and gave a lower bound of 16.0344 which is equal to the objective value obtained by local algorithms, as is shown in Table 6.4.

While constructing an SOS representation of a nonnegative polynomial reduces to solving an SDP feasibility problem and can be done in polynomial time, the size of the resulting SDP grows very rapidly with the dimension and the degree of the polynomial. Moreover, in the constrained case, such as ours, we need to apply Positivstellensatz (5.6) which increases the size of the SDP even further. To improve efficiency of the SOS relaxations, we propose a simple adaptive strategy, where we add some of the box constraints to the formulation is a sequential manner.

Our simple sequential SOS method is stated as Algorithm 6 and the results of a comparison between applying SOSTOOLS to the original formulation and our adaptive strategy can be seen

Table 6.7: *A comparison of algorithms on equity market instance (*482×482 *) with group risk parity enforced. The starting point is chosen to be equally weighted portfolio, i.e.,* $x_i^0 = 1/n$*. The lower and upper bounds are chosen to be* $a = 0, b = 1$*. The starting* μ *is chosen to be 0.1.*

Algorithm	iter. (10^{-3})	QP solved	F-value	iter. (10^{-5})	QP solved	F-value
PLM-θ	38	66	4.62×10^{-9}	568	610	1.39×10^{-13}
PLM	47	88	8.45×10^{-9}	338	400	1.11×10^{-13}
ALM-θ	11	41	2.18×10^{-9}	101	228	5.41×10^{-13}
ALM	10	43	4.38×10^{-10}	261	556	1.22×10^{-13}
AL-BCD	19	184	1.75×10^{-10}	30	890	1.67×10^{-15}
ADMM	48	96	6.51×10^{-9}	77	154	7.64×10^{-13}

in Table 6.8. Instead of imposing all constraints at once, we first use our local algorithm to obtain a stationary point and thus have an initial "guess" which bound should be activated. We add the corresponding bounds to the formulation and solve the corresponding subproblem with SOSTOOLS and SeDuMi. Clearly each SOS subproblem provides a lower bound for our main problem, since it is a relaxation applied to a relaxed feasible set. SOSTOOLS also provides another solution x, which is a new stationary point. We then proceed by adding new constraints if they are violated by the new solutions x. Hence we obtain a sequence of tighter relaxations and nondecreasing lower bounds. In our experiments this simple strategy substantially reduced the size of the SDPs and the overall complexity. For instance, for the 5×5 case with bounds [0.05, 0.35] the original constrained problem results in a 126×2808 SDP (using SOSTOOLS). If we apply Algorithm 6, then we solve two subproblems whose sizes are 126×534 and 126×863, respectively. As can be observed in Table 6.8, this can significantly accelerate the algorithm.

We have tested the SOS approach on two 5×5 instances, in each case using two settings - tight bounds and the full $[0, 1]$ interval for each variable. Clearly, when the bounds are not tight, then risk parity solution exists and the lower bound is known to be 0 and hence there is no need for applying SOS tools. However, we use this setting for testing efficiency and accuracy of the SOS approach and Algorithm 6. In fact in some cases Algorithm 6 produced the accurate bound after solving only one SDP problem.

To demonstrate the challenge of using SOS we tested the 14×14 asset allocation instance with bounds to be $[0, 1]$. As we can see from Table 6.8, when we test the our approach find an inaccurate (positive) bound after more than 1 hour, while directly applying SOSTOOLS fails to provide an answer within 24 hours.

It is important to note that in order to apply Schmdgen Positivstellensatz in (5.6), we need the assumption that the solution lies in a compact set. Here, although we do not add all constraints at once, we can still assume, without loss of generality, that the solution lies in a large enough bounded set.

6.2.1 DSOS and SDSOS optimization

As discussed in Section 5, we can apply DSOS and SDSOS relaxations as potential alternatives of SOS. The hope is that these techniques could provide us with a quality bound at a much smaller computational cost.

Specifically, *SDSOS* approach reduces to the following problem when seeking a lower bound

Algorithm 6 Sequential sos method for risk parity optimization

1. Apply local algorithms (ADMM or ALM) to solve (1.6) and obtain a stationary point x^0 and the corresponding objective function value $f(x^0)$;
2. Initialize the constraint set $\mathcal{X}^0$, where $\mathcal{X}^0$ contains only the bounds which are activated by x^0 and all the equality constraints.
3. for $k = 0, 1, ...$

(a) Solve (5.6) by SOSTOOLS, and obtain new local solution $x^{k+1} \in \mathcal{X}^k$ and a global lower bound γ^{k+1};

(b) if $x^{k+1} \in \mathcal{X}^{true}$ then

break, and γ^{k+1} is a global lower bound for f;

else

add bounds that are violated by x^{k+1} and obtain new feasible set $\mathcal{X}^{k+1}$. Go to (a).

Table 6.8: *A comparison of algorithms on instances with different bounds. Original SOSTOOLS application (denoted as Org. in the table) is compared with relaxed sequential algorithm (Rel.). The default tolerance on duality gap is set to* 10^{-8}*).*

Instance		CPU time (s)		Global lower bounds		
Name (size)	Bounds	Org.	Rel.	Org.	Rel.	Opt.
5 assets-scaled (5)	[0.05, 0.35]	160.13	16.94	16.03	16.03	16.03
5 assets-unscaled (5)	[0, 1]	115.83	24.34	2.48×10^{-10}	1.54×10^{-11}	0
Rand I (5)	[0.15, 0.25]	152.97	8.46	2.01×10^{-3}	2.01×10^{-3}	2.01×10^{-3}
Rand I (5)	[0, 1]	115.98	5.65	2.70×10^{-10}	9.28×10^{-10}	0
Asset Allocation (14)	[0,1]	-	4855.32	-	2.12×10^{-7}	0

for risk parity problem:

$$\begin{array}{ll} \max & \lambda \\ \text{s.t.} & \sum_{i=1}^n (x_i(\Sigma x)_i - \frac{1}{n} x^T \Sigma x)^2 - \lambda - \sum_{i=1}^n L(x) g(x) - r(x)(\sum_{i=1}^n x_i - 1) \in SDSOS_{n,d} \\ & L(x) \in SDSOS_{n,d}, \end{array} \tag{6.1}$$

where n is the dimension (the number of assets), d is the degree of $SDSOS$.

The result using $DSOS$ and $SDSOS$ solving the 5×5 instance is listed in Table 6.9. We used package called SPOTless (see: github.com/spot-toolbox/spotless) to model the $DSOS$ and $SDSOS$ cones and apply Mosek 7.0 to solve the resulting convex optimization problem (LP and SOCP, respectively). We can see that DSOS approximation gave a poor bound (but positive) when the maximum degree was set to 4 and improved the bound to optimal when the degree was increased to 6. SDSOS relaxation obtained good results at (surprisingly) slightly lower computational cost. Further extensive testing is needed to explore the usefulness of these approaches and is a subject of future research.

7 Conclusion remarks

In this paper, we proposed a family of alternating direction methods for minimizing nonlinear non convex problems with special structure which allows convenient 2-block variable splitting. In

Table 6.9: *A comparison of DSOS and SDSOS approach, on solving a 5×5 example. We compare the final lower bound found with a increase of the degree.*

Method	Bound ($d = 4$)	CPU Time	Bound ($d = 6$)	CPU Time	OPT
DSOS	9.983	2.198	14.958	12.858	16.0344
SDSOS	16.021	1.986	16.027	11.142	16.0344

particular these methods apply to minimizing sums of squares of quadratic functions. We propose an alternating directions method of multipliers and an alternating linearization method and we provide convergence rate results for both classes of methods. The experiments on risk parity optimization problem shows the efficiency of these methods and their ability to recover a global minimum for this application. Global optimization techniques from polynomial optimization literature are applied to complement our local methods and to provide lower bounds. Exploring new applications of our methods is subject of future study.

References

[1] Manya Afonso, José Bioucas-Dias, and Mário Figueiredo. An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems. *Image Processing, IEEE Transactions on*, 20(3):681–695, 2011.

[2] Amir Ahmadi and Anirudha Majumdar. DSOS and SDSOS optimization: LP and SOCP-based alternatives to sum of squares optimization. In *Information Sciences and Systems (CISS), 2014 48th Annual Conference on*, pages 1–5. IEEE, 2014.

[3] Xi Bai, Katya Scheinberg, and Reha Tutuncu. Least-squares approach to risk parity in portfolio selection. *Available at SSRN*, 2013.

[4] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[5] Dimitri Bertsekas. *Constrained optimization and Lagrange multiplier methods*, volume 1. Academic Press, 1982.

[6] Dimitri Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[7] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[8] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology-EUROCRYPT 2000*, pages 392–407. Springer, 2000.

[9] Jim Douglas and Henry Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American mathematical Society*, 82(2):421–439, 1956.

[10] Michel Fortin and Roland Glowinski. *Augmented Lagrangian methods: applications to the numerical solution of boundary-value problems*. Elsevier, 2000.

[11] Roland Glowinski and Patrick Le Tallec. *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*, volume 9. SIAM, 1989.

[12] Donald Goldfarb, Shiqian Ma, and Katya Scheinberg. Fast alternating linearization methods for minimizing the sum of two convex functions. *Mathematical Programming*, 141(1-2):349–382, 2013.

[13] Luigi Grippo and Marco Sciandrone. On the convergence of the block nonlinear gauss–seidel method under convex constraints. *Operations Research Letters*, 26(3):127–136, 2000.

[14] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *arXiv preprint arXiv:1410.1390*, 2014.

[15] Bo Jiang, Shiqian Ma, and Shuzhong Zhang. Alternating direction method of multipliers for real and complex polynomial optimization models. *Optimization*, 63(6):883–898, 2014.

[16] Jean Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[17] Zhi-Quan Luo and Shuzhong Zhang. A semidefinite relaxation scheme for multivariate quartic polynomial optimization with quadratic constraints. *SIAM Journal on Optimization*, 20(4):1716–1736, 2010.

[18] Sébastien Maillard, Thierry Roncalli, and Jérôme Teïletche. The properties of equally weighted risk contribution portfolios. *The Journal of Portfolio Management*, 36(4):60–70, 2010.

[19] Martin Mevissen. Introduction to concepts and advances in polynomial optimization, 2007.

[20] Jiawang Nie and Markus Schweighofer. On the complexity of Putinar's positivstellensatz. *Journal of Complexity*, 23(1):135–150, 2007.

[21] Pablo Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, Citeseer, 2000.

[22] Pablo Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical programming*, 96(2):293–320, 2003.

[23] Stephen Prajna, Antonis Papachristodoulou, and Pablo Parrilo. Introducing SOSTOOLS: A general purpose sum of squares programming solver. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 1, pages 741–746. IEEE, 2002.

[24] Mihai Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993.

[25] Bruce Reznick. Some concrete aspects of Hilbert's 17th problem. *Contemporary Mathematics*, 253:251–272, 2000.

[26] Katya Scheinberg, Donald Goldfarb, and Xi Bai. Fast first-order methods for composite convex optimization with backtracking. *Foundations of Computational Mathematics*, 14(3):389–417, 2014.

[27] Konrad Schmüdgen. The k-moment problem for compact semi-algebraic sets. *Mathematische Annalen*, 289(1):203–206, 1991.

[28] Yuan Shen, Zaiwen Wen, and Yin Zhang. Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software*, 29(2):239–263, 2014.

[29] Suvrit Sra, Sebastian Nowozin, and Stephen Wright. *Optimization for machine learning*. MIT Press, 2012.

[30] Stephen Wright and Jorge Nocedal. *Numerical optimization*, volume 2. Springer New York, 1999.

[31] Yangyang Xu and Wotao Yin. A block coordinate descent method for multi-convex optimization with applications to nonnegative tensor factorization and completion. Technical report, DTIC Document, 2012.