Solving PhaseLift by low-rank Riemannian optimization methods for complex semidefinite constraints *

Wen Huang †¶ K. A. Gallivan ‡ Xiangxiong Zhang §

May 24, 2016

Abstract

A framework, PhaseLift, was recently proposed to solve the phase retrieval problem. In this framework, the problem is solved by optimizing a cost function over the set of complex Hermitian positive semidefinite matrices. This approach to phase retrieval motivates a more general consideration of optimizing cost functions on semidefinite Hermitian matrices where the desired minimizers are known to have low rank. This paper considers an approach based on an alternative cost function defined on a union of appropriate manifolds. It is related to the original cost function in a manner that preserves the ability to find a global minimizer and is significantly more efficient computationally. A rank-based optimality condition for stationary points is given and optimization algorithms based on state-of-the-art Riemannian optimization and dynamically reducing rank are proposed. Empirical evaluations are performed using the PhaseLift problem. The new approach is shown to be an effective method of phase retrieval with computational efficiency increased substantially compared to the algorithm used in original PhaseLift paper. A preliminary version can be found in [HGZ16].

1 Introduction

Recovering a signal given the modulus of its transform, e.g., Fourier or wavelet transform, is an important task in the phase retrieval problem. It is a key problem for many important applications, e.g., X-ray crystallography imaging [Har93], diffraction imaging [BDP+07], optics [Wal63] and microscopy [MISE08].

The continuous form of the problem with the Fourier transform recovers $x(t) : \mathbb{R}^s \to \mathbb{C}$ from $|\tilde{x}(u)|$, where $\tilde{x}(u) : \mathbb{R}^s \to \mathbb{C}$ is defined by

$$\tilde{x}(u) = \int_{\mathbb{R}^s} x(t) \exp\left(-2\pi u \cdot t\sqrt{-1}\right) dt,$$

and $\cdot$ denotes the Euclidean inner product. This paper considers the discrete form of the problem where an indexed set of complex numbers $\mathbf{x} \in \mathbb{C}^{n_1 \times n_2 \times \ldots \times n_s}$ is to be recovered from the modulus of its discrete Fourier transform $|\tilde{\mathbf{x}}(g_1, g_2, \ldots, g_s)|$, where $(g_1, g_2, \ldots, g_s) \in \Omega := G_1 \times G_2 \times \ldots G_s$ and Ω is a grid of an s-dimensional space. The discrete Fourier transform $\tilde{\mathbf{x}}$ is given by

$$\tilde{\mathbf{x}}(g_1, g_2, \ldots, g_s) = \frac{1}{\sqrt{n}} \sum_{i_1, i_2, \ldots, i_s} \mathbf{x}_{i_1 i_2 \ldots i_s} \exp\left(-2\pi \left(\frac{(i_1 - 1)g_1}{n_1} + \ldots + \frac{(i_s - 1)g_s}{n_s}\right)\sqrt{-1}\right), \tag{1.1}$$

*This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office. This work was supported by grant FNRS PDR T.0173.13.

†Department of Mathematical Engineering, ICTEAM Institute, Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium.

‡Department of Mathematics, 208 Love Building, 1017 Academic Way, Florida State University, Tallahassee FL 32306-4510, USA.

§Department of Mathematics, 150 N. University Street, Purdue University, West Lafayette, IN 47907-2067, USA.

¶Corresponding author. E-mail: huwst08@gmail.com.

where $n = n_1 n_2 \dots n_s$, i_j is an integer satisfying $1 \le i_j \le n_j$ for $j = 1, \dots s$, $\mathbf{x}_{i_1 i_2 \dots i_s}$ denotes the corresponding entry of $\mathbf{x}$ and $\tilde{\mathbf{x}}(g_1, g_2, \dots, g_s)$ denotes the corresponding entry of $\tilde{\mathbf{x}}$.

It is well-known that the solution of the phase retrieval is not unique. For example, when Ω is the uniform grid, i.e, $G_i = \{0, 1, \dots, n_i - 1\}, i = 1, 2, \dots, s$, if $\mathbf{x}$ is a solution, then

1. $\mathbf{y} = c\mathbf{x}$ is a solution where $c \in \mathbb{C}$ and $|c| = 1$;
2. $\mathbf{y} \in \mathbb{C}^{n_1 \times n_2 \times \dots n_s}$ such that $\mathbf{y}_{i_1 i_2 \dots i_s} = \mathbf{x}_{j_1 j_2 \dots j_s}$ is a solution, where $j_k = i_k + a_k \mod n_k, k = 1, \dots, s$ and $a_1, a_2, \dots a_s$ are integers;
3. $\mathbf{y} \in \mathbb{C}^{n_1 \times n_2 \times \dots n_s}$ such that $\mathbf{y}_{i_1, i_2, \dots i_s} = \bar{\mathbf{x}}_{j_1 j_2 \dots j_s}$ is a solution, where $j_k = -i_k \mod n_k, k = 1, \dots, s$ and $\bar{\mathbf{x}}$ is the conjugate of $\mathbf{x}$.

These equivalent solutions are called trivial associates of $\mathbf{x}$ and infinitely many additional solutions may exist [San85].

Oversampling in the Fourier domain is a standard method to obtain a unique solution and it has been shown to almost always give a unique solution for multiple dimensional problems for real-valued and non-negative signals [BS79, Hay82, San85]. Many algorithms based on alternating projection [GS72] have been developed to solve phase retrieval problem using the oversampling framework [Fie78, Fie82, Els03, Bla04, Mar07, CMWL07]. While these algorithms are efficient and effective in some problem settings, they may not perform well in other settings. For details on the capabilities and difficulties of these algorithms see [CESV13] and the references therein.

In recent years other frameworks, using multiple structured illuminations, or the mathematically equivalent construct of masks, combined with convex programming, have been proposed to recover the phase exactly, e.g., PhaseLift [CESV13] and PhaseCut [WDAM13]. It was later proved that a feasibility problem of two convex sets can be solved for PhaseLift in [CL13, DH14]. For the PhaseLift framework, four major results are of interest here. First, using a small number (related to s) of noiseless measurements of the modulus defined by certain carefully designed illuminations, the phase can be recovered exactly [CESV13]. Second, when these carefully designed measurements are not used, the phase can be recovered exactly with high probability using $O(n \log n)$ noiseless measurements of the modulus [CSV13]. Third, exact recovery is still possible using $O(n)$ noiseless measurements [CL13]. Finally, the stability of recovering the phase using noisy measurements is shown in [CSV13].

For the PhaseCut framework, it is known that if the phase can be recovered using PhaseLift, then it can also be recovered by a modified version of PhaseCut and that the PhaseCut is at least as stable as the weak formulation of PhaseLift for noisy measurements [WDAM13]. The weak formulation is formally defined in [WDAM13, Section 4.1], however, the idea of a weak formulation is also given earlier in the proof of [CESV13, Theorem 2.1]. Empirically, PhaseCut is observed to be more stable in the situation of sparse sampling of the modulus.

The problems in both PhaseLift and PhaseCut concern optimizing convex cost functions defined on a convex set of complex matrices, i.e.,

$$\min_{X \in \mathcal{D}_n} H(X), \tag{1.2}$$

where $H : \mathcal{D}_n \to \mathbb{R} : X \mapsto H(X)$, and $\mathcal{D}_n$ denotes the set of all n-by-n complex Hermitian positive semidefinite matrices. PhaseCut further requires that the diagonal entries of X are 1. However, the dimension of (1.2) is usually too large to be solved by standard convex programming techniques. For example, in order to recover an image of 100 by 100 pixels, i.e., $s = 2$ and $n_1 = n_2 = 100$, solving an optimization problem with an argument that is a 100^2 by 100^2 matrix is required. The complexity of solving PhaseLift and PhaseCut using standard semidefinite programming solvers, e.g., SDPT3 [TTT99], is discussed in [WDAM13, Section 4.6].

Since the desired optimum, X_*, is known to be a rank-one matrix, a low-rank matrix approximation of the argument matrix is used in [CESV13] to save computations for PhaseLift. While this approximation has good empirical performance, no convergence proof is given in [CESV13]. For PhaseCut, a block coordinate descent algorithm is proposed in [WDAM13] and the algorithm is shown to be computationally inexpensive

for each iteration. However, the block coordinate descent algorithm converges slowly, i.e., linear convergence [BV04, Section 9.4.3], and the overall computational cost can be unacceptably high.

This paper uses the framework of PhaseLift and an alternate cost function $F : \mathbb{C}^{n\times p} \to \mathbb{R} : Y \mapsto F(Y) = H(YY^*)$ defined by matrix factorization is considered. Even though F is not convex, it is shown to be a suitable replacement of the cost function H. Riemannian optimization methods on an appropriate quotient space are used for optimizing F. Using the cost function F with a small dimension p reduces storage and the computational complexity of each iteration. Fast convergence rate is also guaranteed theoretically by known Riemannian optimization results. This new approach is shown to perform empirically much better than the low-rank approximate version of the algorithm used for PhaseLift in [CESV13] from the points of view of efficiency and effectiveness. Finally, note that the analysis and algorithm presented is not specific to the cost function used for phase retrieval in PhaseLift but for a general cost function defined on $\mathcal{D}_n$ and therefore the approach has potential for optimization in other applications where the global optimum is known to have low rank.

A forerunner to this paper appeared in [HGZ16]. This paper differs from the conference paper in the following main aspects: (i) proofs of theoretical results and derivations of required ingredients for Riemannian optimization are given in this paper; (ii) a different Riemannian metric is used for the fixed-rank Hermitian positive semidefinite matrices and this metric yields a cheaper Riemannian gradient and isometric vector transport[1]; and (iii) the initial iterate in this paper is chosen by exploiting the approach in [CLS16, Algorithm 1] whereas [HGZ16] only uses a random initial iterate. The proposed initial iterate has an edge over the previous one in the sense that it significantly reduces computational time.

The idea of using low-rank factorization to solve positive semidefinite constrained problems is, of course, not new but all the research results of which the authors are aware, are for real positive semidefinite matrix constraints. Burer and Monteiro [BM03] first investigate this approach for semidefinite programming (SDP) in which the cost function is linear. Journée et al. [JBAS10] use low-rank factorization for a more general problem in the sense that the cost function H is not necessary linear,

$$\min_{X\in\mathbb{S}_n^+} H(X), \quad \text{such that } \operatorname{tr}(A_iX) = b_i, i = 1, \dots m,$$

where $\mathbb{S}_n^+$ denotes the set of all real n-by-n symmetric positive semidefinite matrices, $A_i \in \mathbb{R}^{n\times n}$, $A_i = A_i^T$ and $A_iA_j = 0$ for any $i \neq j$. The conditions that $A_i = A_i^T$ and $A_iA_j = 0$ for any $i \neq j$ implies the number of equality constraints m is at most n as pointed out in [JBAS10]. The complex problem (1.2) does not belong to this category of problem since m is much larger than n when the complex problem is written as a real problem, see details in Appendix A.

The paper is organized as follows. Section 2 presents the notation used. The derivation of the optimization problem framework in PhaseLift is given in Section 3. The alternate cost function and optimality conditions are derived in Section 4. Riemannian optimization methods and the required geometric objects are presented in Section 5. In Section 6, the effectiveness of the methods are demonstrated with several numerical experiments and, finally, conclusions are given in Section 7.

2 Notation

For any $\mathbf{z} \in \mathbb{C}^{n_1\times n_2\times\dots n_s}$, $\operatorname{vec}(\mathbf{z}) \in \mathbb{C}^n$, where $n = n_1n_2\dots n_s$, denotes the vector form of $\mathbf{z}$, i.e., $(\operatorname{vec}(\mathbf{z}))_k = \mathbf{z}_{i_1i_2\dots i_s}$, where $k = i_1 + \sum_{j=1}^{s-1} n_1n_2\dots n_j(i_{j+1} - 1)$. $\operatorname{Re}(\cdot)$ denotes the real part of the argument and superscript $*$ denotes the conjugate transpose operator. Given a vector v with length h, $\operatorname{Diag}(v)$ denotes an h-by-h diagonal matrix the diagonal entries of which are v.

$0_{s\times k}$ denotes an $s \times k$ zero matrix; $I_{s\times k}$ denotes a diagonal matrix with diagonal entries 1; and 0_s denote a vector with length s with entries all 0. $\operatorname{diag}(M)$ denotes a vector of the diagonal entries of $M \in \mathbb{C}^{s\times k}$ and $\operatorname{tr}(M)$ denotes the trace of M. If $s \geq k$, $M_\perp$ denotes an $s \times (s - k)$ matrix such that $M_\perp^*M_\perp = I_{(s-k)\times(s-k)}$

[1]The Riemannian gradient in [HGZ16] requires solving a Sylvester equation whereas the Riemannian gradient in this paper has a cheaper closed form. The isometric vector transports in this paper has complexity $O(np^2) + O(p^3)$ which is cheaper than the one of $O(np^2) + O(p^6)$ in [HGZ16].

and $M_\perp^* M = 0_{(s-k)\times k}$. $M(:, 1:k)$ denotes a matrix that is formed by the first k columns of matrix M. span(M) denotes the column space of M. E_{ij} denotes a matrix with i-th row j-th column entry be 1 and other entries be 0.

Given a manifold $\mathcal{M}$, $\mathrm{T}_x\,\mathcal{M}$ denotes the tangent space of $\mathcal{M}$ at $x \in \mathcal{M}$. $\mathcal{D}_k$ denotes set $\{X \in \mathbb{C}^{n\times n} | X = X^*, X \geq 0, \mathrm{rank}(X) \leq k\}, 1 \leq k \leq n$, where the statement $X \geq 0$ means that matrix X is positive semidefinite or definite. $\mathrm{St}(k, s)$ denotes the complex compact Stiefel manifold $\{A \in \mathbb{C}^{s\times k} | A^*A = I_{k\times k}\}$ with $s \geq k$. $\mathrm{S}_+^{\mathbb{C}}(k, s)$ denotes the set of all Hermitian positive semidefinite $s \times s$ matrices of fixed rank k. When elements of $\mathrm{S}_+^{\mathbb{C}}(k, s)$ are restricted to be real, it is denoted by $\mathrm{S}_+^{\mathbb{R}}(k, s)$. $\mathbb{C}_*^{s\times k}$ denotes the complex noncompact Stiefel manifold, i.e., the set of all $s \times k$ full column rank complex matrices. $\mathcal{O}_s$ denotes the group of s-by-s unitary matrices.

Given a function $f(x)$ on $\mathcal{M}$ or $\mathbb{C}^{s\times k}$, grad $f(x)$ denotes the gradient of f at x.

3 The PhaseLift approach to phase retrieval

The phase retrieval problem recovers $\mathbf{x}$ from its quadratic measurements of the form $\mathbb{A}(\mathbf{x}) = \{|\langle \mathbf{a}_k, \mathbf{x}\rangle|^2 : k = 1, 2, \ldots, m\}$, where $\mathbf{a}_k \in \mathbb{C}^{n_1\times n_2\times \ldots n_s}, k = 1, 2, \ldots, m$ are given. It is well-known that the quadratic measurements can be lifted up to be linear measurements about the rank-one matrix $X = xx^*$, where $x = \mathrm{vec}(\mathbf{x}) \in \mathbb{C}^n$. Specifically, the measurements are $|\langle \mathbf{a}_k, \mathbf{x}\rangle|^2 = \mathrm{tr}(a_k a_k^* x x^*) := \mathrm{tr}(A_k X)$, where $a_k = \mathrm{vec}(\mathbf{a}_k) \in \mathbb{C}^n$. Define $\mathcal{A}$ to be the linear operator mapping X into $b := \begin{bmatrix}\mathrm{tr}(A_1X) & \mathrm{tr}(A_2X) & \ldots & \mathrm{tr}(A_mX)\end{bmatrix}^T$. The goal of the phase retrieval problem is to

$$\text{find } X, \quad \text{such that } \mathcal{A}(X) = b, X \geq 0 \text{ and } \mathrm{rank}(X) = 1. \tag{3.1}$$

The alternative problem suggested in [CESV13] considers an optimization problem that does not force the rank of matrix to be one but adds a nuclear norm penalty term to favor low-rank solutions

$$\min_{X\in\mathcal{D}_n} \|b - \mathcal{A}(X)\|_2^2 + \kappa\,\mathrm{tr}(X), \tag{3.2}$$

where κ is a positive constant.

Measurements with noise, $b \in \mathbb{R}^m$, are assumed to have the form $b = \mathcal{A}(X) + \epsilon$, where $\epsilon \in \mathbb{R}^m$ is noise sampled from a distribution $p(:; \mu)$. The task suggested in [CESV13] is

$$\begin{aligned} &\min_X -\log(p(b; \mu)) + \kappa\,\mathrm{tr}(X) \\ &\text{such that } \mu = \mathcal{A}(X) \text{ and } X \in \mathcal{D}_n, \end{aligned} \tag{3.3}$$

or equivalently

$$\min_{X\in\mathcal{D}_n} -\log(p(b; \mathcal{A}(X))) + \kappa\,\mathrm{tr}(X) \tag{3.4}$$

where κ is a positive constant. Problems (3.3) and (3.4) are preferred over Problem (3.1), since they are convex programming problems when the log-likelihood function is concave.

4 Theoretical results

This section presents theoretical results that motivate the design of algorithms for optimizing a cost function H defined on $\mathcal{D}_n$. The analysis does not rely on the convexity of the cost function H.

4.1 Equivalent cost function

The cost functions generically denoted H all satisfy

$$H : \mathcal{D}_n \to \mathbb{R} : X \mapsto H(X). \tag{4.1}$$

It is well-known that for any $X \in \mathcal{D}_n$, there exists $Y_n \in \mathbb{C}^{n\times n}$ such that $Y_nY_n^* = X$. Furthermore, if X is rank p, then there exists $Y_p \in \mathbb{C}^{n\times p}$ such that $Y_pY_p^* = X$. Throughout this paper, the subscript of Y is used to emphasize the column size of Y. Therefore, a surjective mapping between $\mathbb{C}^{n\times p}$ and $\mathcal{D}_p$ is given by $\alpha_p : \mathbb{C}^{n\times p} \to \mathcal{D}_p : Y_p \mapsto Y_pY_p^*$. It is clear that α_p is not an injection. Specifically, given $X \in \mathcal{D}_p$, if Y_p satisfies $\alpha_p(Y_p) = Y_pY_p^* = X$, then Y_pO_p also satisfies $\alpha_p(Y_pO_p) = X$ for any $O_p \in \mathcal{O}_p$. Thus, if the desired solution of H is known to be at most rank p, then an alternate cost function to H can be used:

$$F_p : \mathbb{C}^{n\times p} \to \mathbb{R} : Y_p \mapsto H(\alpha_p(Y_p)) = H(Y_pY_p^*).$$

The subscripts of F and α indicate the column size of the argument. The domain of F_p has lower dimension than that of H which may yield computational efficiency. Therefore, instead of problem of (1.2), the problem $\min_{Y_p\in\mathbb{C}^{n\times p}} F_p(Y_p)$ is considered.

4.2 Optimality conditions

In this section, the characterizations of stationary points of F and H over $\mathcal{D}_n$ are used to derive the relationship between optimizing F and optimizing H over $\mathcal{D}_n$. Since H is defined on a constrained set, a stationary point of H does not simply satisfies $\operatorname{grad} H(X) = 0$. One can define the stationary points of H as follows by Lemma A.1:

Definition 4.1. *A stationary point of* (4.1) *is a matrix* $X \in \mathbb{D}_n$ *such that* $\operatorname{grad} H(X)X = 0$ *and* $\operatorname{grad} H(X) \geq 0$.

The gradient is easily computed and is given in Lemma 4.1 in terms of H.

Lemma 4.1. *The gradient of* F_p *at* Y_p *is given by*

$$\operatorname{grad} F_p(Y_p) = 2\operatorname{grad} H(Y_pY_p^*)Y_p. \tag{4.2}$$

Proof. On one hand, it satisfies that for all $\eta_p \in \mathbb{C}^{n\times p}$

$$\mathrm{D}\, F_p(Y_p)[\eta_p] = g^E(\operatorname{grad} F_p(Y_p), \eta_p).$$

On the other hand, we have

$$\begin{aligned}\mathrm{D}\, F_p(Y_p)[\eta_p] =& \mathrm{D}\, H(Y_pY_p^*)[Y_p\eta_p^* + \eta_pY_p^*] = g^E(\operatorname{grad} H(Y_pY_p^*), Y_p\eta_p^* + \eta_pY_p^*)\\ =& g^E((\operatorname{grad} H(Y_pY_p^*) + \operatorname{grad} H(Y_pY_p^*)^*)Y_p, \eta_p),\end{aligned}$$

which implies $\operatorname{grad} F_p(Y_p) = (\operatorname{grad} H(Y_pY_p^*) + \operatorname{grad} H(Y_pY_p^*)^*)Y_p$. Since H is defined on Hermitian matrices, $\operatorname{grad} H$ can be written as a Hermitian matrix. It follows that $\operatorname{grad} F_p(Y_p) = 2\operatorname{grad} H(Y_pY_p^*)Y_p$ which is (4.2). □

Theorem 4.1 and [JBAS10, Theorem 7] show similar results under different frameworks. Both results suggest considering the cost function F_p if the desired minimizer of H is known to have rank smaller than p, as is the case with PhaseLift for phase retrieval. This is formalized in Theorem 4.1 and has critical algorithmic, efficiency and optimality implications when H has suitable structure such as convexity as in the case of PhaseLift. These implications for PhaseLIft are discussed in Section 6.1.

Theorem 4.1. *Suppose* $Y_p = K_sQ^*$ *is a rank deficient minimizer of* F_p*, where* $K_s \in \mathcal{C}_*^{n\times s}$ *with* $s < p$ *and* $Q \in \operatorname{St}(s,p)$*. Then* $\operatorname{grad} H(Y_pY_p^*)$ *is a positive semidefinite matrix and, therefore,* $X = Y_pY_p^*$ *is a stationary point of* H*. If furthermore* H *is convex, then* X *is a global minimizer of* (4.1)*.*

Proof. We first show that $(K_s)_\perp^* \operatorname{grad} H(X)(K_s)_\perp$ is a positive semidefinite matrix. This is proved by contradiction. If $(K_s)_\perp^* \operatorname{grad} H(X)(K_s)_\perp$ is not a positive semidefinite matrix, then it has at least one negative eigenvalue. If μ and v denote a negative eigenvalue and the corresponding eigenvector then the

semidefinite positive matrix $\eta = -(K_s)_\perp (v\mu v^*)(K_s)_\perp^*$ satisfies $g^E(\eta, \operatorname{grad} H(X)) < 0$. Thus, a smooth curve $\gamma(t) = X + t\eta$ satisfies that $\dot\gamma(0) = \eta$, $\gamma(t) \in \mathcal{D}_p$ for all $t \in [0, \delta)$ and $\gamma(0) = X$, where δ is a positive constant. The derivative $\frac{d}{dt} H(\gamma(t))|_{t=0}$ by definition is $g^E(\eta, \operatorname{grad} H(X))$ and, therefore, $\frac{d}{dt} H(\gamma(t))|_{t=0} < 0$.

Decomposing $\gamma(t)$ yields

$$\gamma(t) = \begin{pmatrix} K_s & \tilde v \end{pmatrix} \begin{pmatrix} I_s & \\ & -t\mu \end{pmatrix} \begin{pmatrix} K_s & \tilde v \end{pmatrix}^*,$$

where $\tilde v = (K_s)_\perp v$. Define $r(t) = \begin{pmatrix} K_s & \tilde v \end{pmatrix} \operatorname{Diag}(I_s, \sqrt{-t\mu})$. Therefore, $\gamma(t) = r(t) r^*(t)$. The derivative of $r(t^2)$ is $\xi(t) = \begin{pmatrix} 0_{n\times s} & \sqrt{-\mu}\tilde v \end{pmatrix}$. It follows that

$$\frac{d^2}{dt^2} H(\gamma(t^2))|_{t=0} = \frac{d^2}{dt^2} F_n(r(t^2))|_{t=0} = g^E(\xi(0), \operatorname{Hess} F_p(\tilde Y_p)[\xi(0)]) \geq 0. \tag{4.3}$$

Let $a(t) = H(\gamma(t))$ and so $\dot a(0) < 0$. It follows that

$$\frac{d^2}{dt^2} H(\gamma(t^2))|_{t=0} = \frac{d^2}{dt^2} a(t^2) = (4t^2 \ddot a(t^2) + 2\dot a(t^2))|_{t=0} = 2\dot a(0) < 0,$$

which conflicts with (4.3). Therefore, $(K_s)_\perp^* \operatorname{grad} H(Y_p Y_p^*)(K_s)_\perp$ is a positive semidefinite matrix which is a contradiction with the initial assumption of the proof.

Let Q_s denote an orthonormal basis of $\operatorname{span}(K_s)$. $\operatorname{grad} H(X)$ can be written as

$$\operatorname{grad} H(X) = \begin{pmatrix} Q_s & (K_s)_\perp \end{pmatrix} \begin{pmatrix} S & A^* \\ A & R \end{pmatrix} \begin{pmatrix} Q_s & (K_s)_\perp \end{pmatrix}^*,$$

where $S \in \mathbb{C}^{s\times s}$, $S^* = S$, $A \in \mathbb{C}^{(n-s)\times p}$ and $R = (K_s)_\perp^* \operatorname{grad} H(X)(K_s)_\perp$. Since Y_p is a stationary point of F, we have $\operatorname{grad} H(X) K_s Q^* = \operatorname{grad} F(Y_p) = 0_{n\times s}$. It follows that $S = 0_{s\times s}$ and $A = 0_{(n-s)\times s}$. Therefore, $R \geq 0$ implies $\operatorname{grad} H(X) \geq 0$ which means X is a stationary point of H by Definition 4.1. □

5 A Riemannian approach

Riemannian optimization is an active research area and recently many Riemannian optimization methods have been systemically analyzed and efficient libraries designed, e.g., Riemannian trust-region Newton method (RTR-Newton) [Bak08], Riemannian Broyden family method including BFGS method and its limited-memory version (RBroyden family, RBFGS, LRBFGS) [RW12, Hua13, HGA15], Riemannian trust-region symmetric rank-one update method and its limited-memory version (RTR-SR1, LRTR-SR1) [Hua13, HAG15], Riemannian Newton method (RNewton) and Riemannian non-linear conjugate gradient method (RCG) [AMS08, SI15, Sat15].

Journée et al. [JBAS10] have proposed a method that combines a Riemannian optimization method on a fixed rank manifold with a procedure of increasing rank for their semidefinite constrained problem setting. Specifically, given an iterate with rank r, a Riemannian optimization method is applied for a cost function on a manifold with rank r. If the limit point is not rank deficient, then either a descent direction to a higher rank space can be found or a desired stationary point is obtained. For the former case, a descent algorithm is applied to find a next descent iterate which is used to be the initial point for a Riemannian optimization method on a manifold with larger rank. For the latter case, the convergence rate can be obtained and depends on the Riemannian optimization algorithm. If the limit point is rank deficient, then all existing Riemannian convergence analyses are not applicable. This case was ignored in [JBAS10] since situations of a limit point being rank deficient were not encountered in their experiments.

If the rank of the desired minimizer is known, such as in the problems in PhaseLift, then [BM03] suggests to choose the rank of initial point to be that rank. However, this, in fact, is not the appropriate response due to complexity considerations as the theory and algorithms derived in this section indicate, and the numerical experiments in Section 6 demonstrate. To see this, let r_* denote the desired rank of the global minimizer.

Note that there may exist a stationary point Y_{r_*} of F_{r_*} for which $Y_{r_*}Y_{r_*}^*$ is not a stationary point of H. It follows that forcing iterates to be rank r_* is not appropriate and starting from a higher rank or moving to a higher rank to move to the minimizer of H is necessary. This is discussed further later in this section.

There is also a potential problem of using a rank increasing procedure. If Y_p is a stationary point of F_p, then $(Y_p, 0_{n\times(k-p)})$ is also a stationary point of F_k. A procedure that increases rank starting from Y_p may find a point Y_k which is close to $(Y_p, 0_{n\times(k-p)})$. It follows that using Y_k to be an initial point of the iteration on the rank-k manifold may not work efficiently since Y_k may be too close to a stationary point. Therefore, an algorithm based on Riemannian optimization methods on a fixed rank manifold and a procedure to decrease rank without using any rank increase technique is proposed in this section. Since it is known that the minimizer for PhaseLift phase retrieval has rank 1 this is sufficient to allow global minimization using only rank decreases.

5.1 Riemannian optimization on fixed rank manifold

In order to make use of Riemannian optimization theory and algorithms on a fixed rank manifold, the Riemannian gradient of the cost function, the tangent space of an element in the manifold, the retraction operation on the manifold, and an appropriate vector transport are needed. The definitions of Riemannian gradient, tangent space, retraction and vector transport are standard and can be found, e.g., in [Boo86, AMS08].

Derivations for Riemannian objects of $\mathrm{S}_+^{\mathbb{R}}(p,n)$ have been given in [AIDV09]. This section includes derivations of Riemannian objects for the complex case, i.e., $\mathrm{S}_+^{\mathbb{C}}(p,n)$. Since the mapping α_p is not an injection, all the minimizers of F_p are degenerate, which causes difficulties in some algorithms, e.g., Riemannian and Euclidean Newton method. In order to overcome this difficulty, a function defined on a quotient manifold with fixed rank is considered. To this end, define the mapping β_p to be the mapping α_p restricted on $\mathbb{C}_*^{n\times p}$, i.e., $\beta_p : \mathbb{C}_*^{n\times p} \to \mathrm{S}_+^{\mathbb{C}}(p,n) : Y \mapsto \alpha_p(Y) = YY^*$, and function G_p to be the function F_p restricted on $\mathbb{C}_*^{n\times p}$, i.e., $G_p : \mathbb{C}_*^{n\times p} \to \mathbb{R} : Y \mapsto F_p(Y) = H(\beta_p(Y))$. Like α_p, the mapping β_p is a surjection but not an injection and there are multiple matrices in $\mathbb{C}_*^{n\times p}$ mapping to a single point in $\mathrm{S}_+^{\mathbb{C}}(p,n)$. Nevertheless, given an $X \in \mathrm{S}_+^{\mathbb{C}}(p,n)$, $\beta_p^{-1}(X)$ is a manifold while $\alpha_p^{-1}(X)$ is not a manifold. Therefore, using the mapping β_p, a quotient manifold can be used to remove the degeneracy by defining the equivalence class $\beta_p^{-1}(YY^*) = [Y] = \{YO | O \in \mathcal{O}_p\}$ and the set

$$\mathbb{C}_*^{n\times p}/\mathcal{O}_p = \{[Y] | Y \in \mathbb{C}_*^{n\times p}\}.$$

This set can be shown to be a quotient manifold over $\mathbb{R}$. To clarify the notation, $\pi(Y)$ is used to denote $[Y]$ viewed as an element in $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ and $\pi^{-1}(\pi(Y))$ is used to denote $[Y]$ viewed as a subset of $\mathbb{C}_*^{n\times p}$. The function $m_p : \pi(Y) \mapsto YY^*$ is a diffeomorphism between $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ and $\mathrm{S}_+^{\mathbb{C}}(p,n)$.

An element of a quotient manifold is an equivalence class which is often cumbersome computationally . Fortunately, choosing a representative for an equivalence class and definitions of related mathematical objects have been developed in many papers in the literature of computation on manifolds, e.g., [AMS08]. The vertical space at $Y \in \pi^{-1}(\pi(Y))$, which is the tangent space of $\pi^{-1}(\pi(Y))$ at Y, is

$$\mathcal{V}_Y = \{Y\Omega | \Omega^* = -\Omega, \Omega \in \mathbb{C}^{p\times p}\}.$$

The horizontal space at Y, $\mathcal{H}_Y$, is defined to be a subspace of $\mathrm{T}_Y\,\mathbb{C}_*^{n\times p} = \mathbb{C}^{n\times p}$ that is orthogonal to $\mathcal{V}_Y$, i.e., satisfying $\mathcal{H}_A \oplus \mathcal{V}_A = \mathrm{T}_A\,\mathrm{GL}(n,\mathbb{C})$. Therefore, a Riemannian metric of $\mathbb{C}_*^{n\times p}$ is required to define the meaning of orthogonal. The metric used is

$$\hat{g}_Y(\eta_Y, \xi_Y) = \mathrm{Re}(\mathrm{tr}((Y^*Y)\eta_Y^*\xi_Y)), \tag{5.1}$$

for all $\eta_Y, \xi_Y \in \mathrm{T}_Y\,\mathbb{C}_*^{n\times p}$ and $Y \in \mathbb{C}_*^{n\times p}$. The metric (5.1) yields a cheap vector transport by parallelization which is discussed later. The horizontal space is therefore

$$\mathcal{H}_Y = \{YS + Y_\perp K | S^* = S, S \in \mathbb{C}^{p\times p}, K \in \mathbb{C}^{(n-p)\times p}\}.$$

The horizontal space $\mathcal{H}_Y$ is a representation of the tangent space $\mathrm{T}_{\pi(Y)}\,\mathbb{C}_*^{n\times p}/\mathcal{O}_p$. It is known that for any $\eta_{\pi(Y)} \in \mathrm{T}_{\pi(Y)}\,\mathbb{C}_*^{n\times p}/\mathcal{O}_p$, there exists a unique vector in $\mathcal{H}_Y$, called the horizontal lift of $\eta_{\pi(Y)}$ and denoted by $\eta_{\uparrow_Y}$, satisfying $\mathrm{D}\,\pi(Y)[\eta_{\uparrow_Y}] = \eta_{\pi(Y)}$, see e.g., [AMS08]. Lemma 5.1 gives a relationship among horizontal lifts of a tangent vector $\eta_{\pi(Y)}$ when different representations in $\pi^{-1}(\pi(Y))$ are chosen. The result follows from [Hua13, Theorem 9.3.1].

Lemma 5.1. *A horizontal vector field $\hat{\eta}$ of $\mathbb{C}_*^{n\times p}$ is the horizontal lift of a vector field η on $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ if and only if, for each $Y \in \mathbb{C}_*^{n\times p}$, we have $\hat{\eta}_{YO} = \hat{\eta}_Y O$ for all $O \in \mathcal{O}_p$.*

The orthogonal projections on to the horizontal space or the vertical space are also easily characterized.

Lemma 5.2. *The orthogonal projection to vertical space $\mathcal{V}_Y$ is $P_Y^v(\eta) = Y\Omega$, where $\Omega = ((Y^*Y)^{-1}Y^*\eta - \eta^*Y(Y^*Y)^{-1})/2$ is a skew Hermitian matrix. The orthogonal projection to Horizontal space $\mathcal{H}_Y$ is $P_Y^h(\eta) = \eta - Y\Omega$.*

Proof. By definition of $\mathcal{H}_Y$ and $\mathcal{V}_Y$, $P_Y^h(\eta)$ satisfies that $(Y^*Y)^{-1}Y^*P_Y^h(\eta) = P_Y^h(\eta)^*Y(Y^*Y)^{-1}$ and can be expressed as $\eta - Y\Omega$. It follows that $\Omega = ((Y^*Y)^{-1}Y^*\eta - \eta^*Y(Y^*Y)^{-1})/2$ which gives the desired results. □

Finally, the desired cost function that removes the equivalence can be defined as

$$f_p : \mathbb{C}_*^{n\times p}/\mathcal{O}_p \to \mathbb{R} : \pi(Y) \mapsto f_p(\pi(Y)) = G_p(Y) = F_p(Y). \tag{5.2}$$

The function f_p in 5.2 has the important property that $\pi(Y)$ is a nondegenerate minimizer of f over $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ if and only if YY^* is a nondegenerate minimizer of H over $\mathrm{S}_+^{\mathbb{C}}(p,n)$.

The gradient is given in Lemma 5.3.

Lemma 5.3. *The horizontal lift of the gradient of* (5.2) *at Y is*

$$(\operatorname{grad} f(\pi(Y)))_{\uparrow_Y} = P_Y^h(\operatorname{grad} F(Y)).$$

Proof. The directional derivative of f along any $\eta_{\pi(Y)} \in \mathrm{T}_{\pi(Y)}\,\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ is

$$\begin{aligned}\mathrm{D}\,f(\pi(Y))[\eta_{\pi(Y)}] &= \mathrm{D}\,f(\pi(Y))[\mathrm{D}\,\pi(Y)[\eta_{\uparrow_Y}]] \\ = \mathrm{D}\,F(Y)[\eta_{\uparrow_Y}] &= \hat{g}_Y(\operatorname{grad} F(Y), \eta_{\uparrow_Y}) = \hat{g}_Y(P_Y^h(\operatorname{grad} F(Y)), \eta_{\uparrow_Y}).\end{aligned}$$

Additionally using the definition of gradient [AMS08, (3.31)], i.e., $\mathrm{D}\,f(\pi(Y))[\eta_{\pi(Y)}] = g_{\pi(Y)}(\operatorname{grad} f(\pi(Y)), \eta_{\pi(Y)})$, and the equation $g_{\pi(Y)}(\operatorname{grad} f(\pi(Y)), \eta_{\pi(Y)}) = \hat{g}_Y((\operatorname{grad} f(\pi(Y)))_{\uparrow_Y}, \eta_{\uparrow_Y})$, yields the result. □

Retraction is used in updating iterates in a Riemannian algorithm. Vector transport is used to compare tangent vectors in different tangent spaces. Specifically, a retraction R is a smooth mapping from the tangent bundle $\mathrm{T}\,\mathcal{M}$, which is the set of all tangent spaces, onto $\mathcal{M}$ such that (i) $R(0_x) = x$ for all $x \in \mathcal{M}$ (where 0_x denotes the origin of $\mathrm{T}_x\,\mathcal{M}$) and (ii) $\frac{d}{dt}R(t\xi_x)|_{t=0} = \xi_x$ for all $\xi_x \in \mathrm{T}_x\,\mathcal{M}$. The restriction of R to $\mathrm{T}_x\,\mathcal{M}$ is denoted by R_x. A vector transport $\mathcal{T} : \mathrm{T}\,\mathcal{M} \oplus \mathrm{T}\,\mathcal{M} \to \mathrm{T}\,\mathcal{M}, (\eta_x, \xi_x) \mapsto \mathcal{T}_{\eta_x}\xi_x$ with associated retraction R is a smooth mapping such that, for all (x, η_x) in the domain of R and all $\xi_x \in \mathrm{T}_x\,\mathcal{M}$, it holds that (i) $\mathcal{T}_{\eta_x}\xi_x \in \mathrm{T}_{R(\eta_x)}\,\mathcal{M}$, (ii) $\mathcal{T}_{0_x}\xi_x = \xi_x$, (iii) $\mathcal{T}_{\eta_x}$ is a linear map. The retraction used in the Riemannian optimization methods is

$$R_{\pi(Y)}(\eta_{\pi(Y)}) = \pi(Y + \eta_{\uparrow_Y}), \tag{5.3}$$

and the vector transport used is the vector transport by parallelization [HAG16]:

$$\mathcal{T}_{\eta_x}\xi_x = B_y B_x^{\dagger},$$

where B is a smooth tangent basis field defined on an open set $\mathcal{V}$ of $\mathcal{M}$ and $B_x^{\dagger}$ denotes the pseudo-inverse of B_x. A smooth orthonormal tangent basis of $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ can be defined as follows: given $\pi(Z) \in \mathbb{C}_*^{n\times p}/\mathcal{O}_p$,

the horizontal lifts of columns in $B_{\pi(Z)}$ at Z is

$$\begin{aligned}
&\left\{ZL^{-*}e_ie_i^TL^{-1}, i=1,\dots,p\right\}\\
\bigcup&\left\{\frac{1}{\sqrt{2}}ZL^{-*}(e_ie_j^T-e_je_i^T)L^{-1}, i=1,\dots,p, j=i+1,\dots,p\right\}\\
\bigcup&\left\{\frac{1}{\sqrt{2}}ZL^{-*}(e_ie_j^T+e_je_i^T)\sqrt{-1}L^{-1}i=1,\dots,p, j=i+1,\dots,p\right\}\\
&\bigcup\left\{Z_\perp\tilde{e}_ie_j^TL^{-1}, i=1,\dots n-p, j=1,\dots p\right\}\\
&\bigcup\left\{Z_\perp\tilde{e}_ie_j^T\sqrt{-1}L^{-1}, i=1,\dots n-p, j=1,\dots p\right\},
\end{aligned}$$

where $(e_1,\dots,e_p)$ is the canonical basis of $\mathbb{R}^p$, $(\tilde{e}_1,\dots,\tilde{e}_{(n-p)})$ is the canonical basis of $\mathbb{R}^{n-r}$, and $Z^TZ=LL^*$ is the Cholesky decomposition.

In summary, this section provides the objects used in Riemannian optimization methods, i.e., the horizontal space, the projection to a horizontal space, the Riemannian metric, the retraction, the vector transport, and the Riemannian gradient.

5.2 Dynamic rank reduction

Since the domain of f_p, $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$, is not closed, i.e., a sequence $\{W^{(i)}\}$ representing $\{\pi(W^{(i)})\}$ generated by an algorithm may have a limit point $\hat{W}$ with rank less than p, a simple well-known strategy for dynamically reducing rank is adapted and used. Since it is impossible in practice to check whether a limit point of iterates $\{W^{(i)}\}$ is a lower rank matrix or just close to one of lower rank, the idea suggested below makes more sense when the desired rank of the minimizer is known and the current iterate $W^{(i)}$ has a higher rank than the desired rank. This is the case with PhaseLift for phase retrieval.

The thin singular value decomposition of the i-th iterate is $W^{(i)}=U\Sigma V^*$ and $\Sigma=\operatorname{Diag}(\sigma_1,\sigma_2,\dots,\sigma_p)$, where $\sigma_1\geq\sigma_2\geq\ldots\geq\sigma_p\geq 0$. Let $\tilde{\sigma}$ be $\|\operatorname{Diag}(\sigma_1,\dots,\sigma_p)\|_F/\sqrt{p}$. If there exists $q<p$ such that $\sigma_q/\tilde{\sigma}>\delta$ and $\sigma_{q+1}/\tilde{\sigma}\leq\delta$ for a given threshold δ, then $\hat{W}=U(:,1:q)\operatorname{Diag}(\sigma_1,\dots\sigma_q)V(:,1:q)^*$ is chosen to be the initial point for optimizing cost function f_q over $\mathbb{C}_*^{n\times q}/\mathcal{O}_q$. The details of reducing rank are given in Algorithm 1. Note that the step of decreasing the rank may produce an iterate that increases the cost function value. This facilitates global optimization by allowing nondescent steps.

Algorithm 1 Reduce Rank

Require: $Y\in\mathbb{C}^{n\times p}$; threshold δ;
Ensure: $W\in\mathbb{C}^{n\times q}$;
1: Take thin singular value decomposition for Y, i.e., $Y=U\operatorname{Diag}(\sigma_1,\dots,\sigma_p)V^*$, where $U\in\mathbb{C}^{n\times p}$, $V\in\mathbb{C}^{p\times p}$ and $\sigma_1\geq\ldots\geq\sigma_p\geq 0$;
2: Set $\tilde{\sigma}=\|\operatorname{Diag}(\sigma_1,\dots,\sigma_p)\|_F/\sqrt{p}$;
3: **if** $\sigma_p/\tilde{\sigma}>\delta$ **then**
4: $q\leftarrow p$, $W\leftarrow Y$ and return;
5: **else**
6: Find q such that $\sigma_q/\tilde{\sigma}>\delta$ and $\sigma_{q+1}/\tilde{\sigma}\leq\delta$;
7: Let $W=U(:,1:q)\operatorname{Diag}(\sigma_1,\dots\sigma_q)V(:,1:q)^*$ and return;
8: **end if**

Combining a Riemannian optimization method with the procedure of reducing rank gives Algorithm 2.

6 Experiments

In this section, numerical simulations for noiseless problems and those with Gaussian noise are used to illustrate the performance of the proposed method. The required Riemannian objects are derived in Section 6.1

Algorithm 2 Rank Reduce Algorithm

Require: $p > 0$; $Y_p^{(0)} \in \mathbb{C}^{n\times p}$ a representation of initial point $\pi(Y_p^{(0)})$ for f; Stopping criterion threshold ϵ; rank reducing threshold δ; a Riemannian optimization method;

Ensure: W

1: **for** $k = 0, 1, 2, \ldots$ **do**
2: Apply Riemannian method for cost function f over $\mathbb{C}_*^{n\times p}/\mathcal{O}_p$ with initial point $\pi(Y_p^{(k)})$ until i-th iterate $W^{(i)}$ satisfying $g(\operatorname{grad} f, \operatorname{grad} f) < \epsilon^2$ or the requirement of reducing rank with threshold δ;
3: **if** $g(\operatorname{grad} f, \operatorname{grad} f) < \epsilon_1^2$ **then**
4: Set $W \leftarrow W^{(i)}$ and return;
5: **else** {iterate in the Riemannian optimization method meets the requirements of reducing rank}
6: Apply Algorithm 1 with threshold δ and obtain an output $\hat{W} \in \mathbb{C}^{n\times q}$;
7: $p \leftarrow q$ and set $Y_p^{(k+1)} = \hat{W}$;
8: **end if**
9: **end for**

and the experimental environment and parameters are defined in Section 6.2. Algorithm 2 with LRBFGS is compared for a range of parameters in Section 6.3. In Section 6.4, the Riemannian approach is compared to the algorithm used in the convex programming approach of [CESV13, CSV13] that represents the current PhaseLift state-of-the-art. Finally, the performance is evaluated for various sizes of natural images.

6.1 Cost function, gradient, and complexity for PhaseLift

The known random masks or illumination fields defined on the discrete signal domain are denoted $\mathbf{w}_r \in \mathbb{C}^{n_1\times n_2\times\ldots n_s}$, $r = 1, \ldots l$. It follows that $\{\langle \mathbf{a}_k, \mathbf{x}\rangle, k = 1, \ldots m\}$ is

$$\begin{pmatrix} (\mathcal{F}_{n_s} \otimes \mathcal{F}_{n_{s-1}} \otimes \ldots \mathcal{F}_{n_1}) \operatorname{Diag}(w_1)x \\ \vdots \\ (\mathcal{F}_{n_s} \otimes \mathcal{F}_{n_{s-1}} \otimes \ldots \mathcal{F}_{n_1}) \operatorname{Diag}(w_l)x \end{pmatrix},$$

where $\otimes$ denotes the Kronecker product and $\mathcal{F}_{n_i} \in \mathbb{C}^{t_i\times n_i}, i = 1, \ldots, s$ denotes the one-dimensional Discrete Fourier Transform (DFT). Let Z_i denote $(\mathcal{F}_{n_s} \otimes \mathcal{F}_{n_{s-1}} \otimes \ldots \mathcal{F}_{n_1}) \operatorname{Diag}(w_i)$, Z denote $(Z_1^T\ Z_2^T\ \ldots\ Z_l^T)^T$. We have $\mathbb{A}(\mathbf{x}) = \operatorname{diag}(Zxx^*Z^*)$, which implies that $\mathcal{A}(X) = \operatorname{diag}(ZXZ^*)$.

When the entries in the noise ϵ are drawn from the normal distribution with mean 0 and variance τ, the cost functions of (3.4) and (3.2) are essentially identical, i.e., for (3.2), $H_1(X) = \|b - \operatorname{diag}(ZXZ^*)\|_2^2 + \kappa \operatorname{tr}(X)$, and for (3.4), $H_2(X) = \frac{1}{\tau^2}\|b - \operatorname{diag}(ZXZ^*)\|_2^2 + \kappa \operatorname{tr}(X)$. Without loss of generality, only the cost function $H(X) = \|b - \operatorname{diag}(ZXZ^*))\|_2^2/\|b\|_2^2 + \kappa \operatorname{tr}(X)$ is considered. It can be shown that the Euclidean gradient of H is $\operatorname{grad} H(X) = \frac{2}{\|b\|_2^2} Z^* \operatorname{Diag}(\operatorname{diag}(ZXZ^*) - b)Z + \kappa I_{n\times n}$. The gradients of functions F_p and f_p can be constructed by using Lemmas 4.1 and 5.3.

The complexities of evaluations of the function value, and gradient of F_p are all of the same order, $O(pms \max_i(\log(n_i))$. The complexities of evaluations of the function value, and gradient of f_p are $O(pms \max_i(\log(n_i))$ and $O(pms \max_i(\log(n_i)) + O(np^2) + O(p^3)$ respectively. The complexities of the retraction and the vector transport by parallelization are $O(pn)$ and $O(np^2) + O(p^3)$ respectively. If $p << n$ then all these complexities are dominated by $O(pms \max_i(\log(n_i))$.

For the optimization problems in the PhaseLift framework for phase retrieval, Theorem 4.1 is important due to the following reasons. First, the cost function H in PhaseLift is convex over a convex domain $\mathcal{D}_n$. Therefore, finding a stationary point of H by using the cost function F is sufficient to find a global minimizer of H. Second, the rank of the desired minimizer of H in PhaseLift is one. It follows that by using a low-rank factorization-based cost function F_p with small $p > 1$ it is possible to find the desired unique rank-one minimizer of H and optimizing F_p with small $p > 1$. (This approach also has lower storage and computational complexity compared to optimizing H.) The theorem guarantees that any minimizer, Y_p, of F_p with rank

less than p must have rank 1 and $Y_pY_p^*$ must be the global minimizer of H. Stationary points, including local minimizers, of F_p with rank p can be discarded if found and the algorithm restarted appropriately. If an X with numerical rank $1 < r < p$ is encountered when iterating using F_p then X is not a stationary point and the rank reduction strategy increases efficiency by removing the unnecessary directions from X and continuing the iteration on F_r.

Even though stationary points with rank 1 that are not local minimizers of F_1 may exist, their presence tends to simply slow the algorithm rather than stopping the iteration at the saddle point. As expected, therefore, running with $p > 1$ avoids this issue completely. There is no theorem guaranteeing that the iterates generated by optimizing F_p with adapting but remaining greater than 1 always converge to an approximation of the rank-one minimizer of H in PhaseLift, such convergence occurred in all of the experiments below and if it were to occur Theorem 4.1 allows detection and restarting as discussed above.

6.2 Data, parameters and notations

All codes are written in Matlab and all experiments are performed in Matlab R2014a on a 64 bit Ubuntu system with 3.6 GHz CPU (Intel (R) Core (TM) i7-4790).

Unless indicated in the description of the experiments, the following test data are used. A complex number $a + b\sqrt{-1}$ is said to be drawn from a distribution in this paper if both a and b are drawn from the distribution independently. The entries of the true solution x_* and Gaussian masks $w_i, i = 1, \dots l$ are drawn from the standard normal distribution. The entries of x_* are further normalized by $\|x_*\|_2$ and the $w_i, i = 1, \dots, l$ are further normalized by $\sqrt{n}$. For the noiseless problem, the measurement b is set to be $\mathrm{diag}(Zx_*x_*^*Z^*)$ and for Gaussian noise problem, the measurement b is set to be $\mathrm{diag}(Zx_*x_*^*Z^*) + \epsilon$, where the entries of $\epsilon \in \mathbb{R}^m$ are drawn from the normal distribution with mean 0 and variance τ that is specified later for each experiment.

The initial iterate $Y_p^{(0)}$ is computed by Algorithm 3, which generalizes [CLS16, Algorithm 1] such that the values $p > 1$ is allowed. We set the number of iterations N in Algorithm 3 to be 20 in all the experiments.

Note the initial iterate is chosen such that its singular values are identical. This choice of initial point

Algorithm 3 Initialization

Require: $Y \in \mathbb{C}^{n\times p}$ drawn from the standard normal distribution;
Ensure: Initial iterate $Y_p^{(0)}$;
1: $Y \leftarrow \mathrm{orth}(Y)$, where $\mathrm{orth}(M)$ is computed by orthonormalizing the matrix M.
2: **for** i = 1, . . . , N **do**
3: $\quad Y \leftarrow \mathrm{orth}(Z^* \,\mathrm{Diag}(b) ZY)$;
4: **end for**
5: $Y_p^{(0)} \leftarrow Y$;

minimizes the influence of magnitudes of singular values of the initial point. In other words, if a bias of magnitudes of singular values is shown during iteration, one knows that the bias is generated by the algorithm and the surface of the cost function not the initial iterate.

The limited-memory version of Riemannian BFGS (LRBFGS) method is chosen to be the representative Riemannian method in Step 2 of Algorithm 2. The stopping criterion of Algorithm 2 requires the norm of gradient to less than 10^{-8} and the minimum number of iterations at each rank is 10. The parameter κ is chosen to be $1/\sqrt{n}$ if $p > 1$ and 0 if $p = 0$ for Algorithm 2.

To obtain sufficiently stable timing results, an average time is taken of several runs with identical parameters for a total runtime of at least 1 minute. The notation used when reporting the experimental results is given in Table 1.

Table 1: Notation for reporting the experimental results.

iter	summation of numbers of iterations in Step 2 of Algorithm 2
nf	number of function evaluations
ng	number of gradient evaluations
f_f	the function value of the final iterate
t	average wall time (seconds)

Table 2: The mean and the standard derivation of computational time of 20 runs of Algorithm 2 using LRBFGS with variant p_0 and δ and output format is (mean)/(the standard derivation). Since δ does not take effect for $p_0 = 1$, the row corresponding to $p_0 = 1$ has only one result.

	δ	0.95	0.9	0.85	0.8	0.75
p_0	1	$9.06_{-1}/3.75_{-1}$				
	2	$5.67_{-1}/7.95_{-2}$	$5.75_{-1}/8.05_{-2}$	$5.89_{-1}/8.32_{-2}$	$6.02_{-1}/8.84_{-2}$	$6.11_{-1}/8.64_{-2}$
	4	$5.99_{-1}/6.25_{-2}$	$6.63_{-1}/7.73_{-2}$	$7.03_{-1}/7.60_{-2}$	$7.62_{-1}/7.75_{-2}$	$7.98_{-1}/7.72_{-2}$

6.3 Choices of initial point size and rank reducing threshold

Table 2 presents the experimental results of Algorithm 2 using the representative Riemannian method LRBFGS with $l = 6$, $s = 2$ and several values of p_0 and δ. The noiseless problem is used. The mean and the standard derivation of computational time of 20 runs of Algorithm 2 are reported.

The average computational time and the standard derivation of $p_0 = 1$ are much larger relatively than the other starting ranks. This clearly shows that the performance simply optimizing over matrices with the fixed optimal rank is not a reliable and efficient method. Additionally, note that, when the initial point is close to the global rank-one minimizer, then Algorithm 2 with $p_0 = 1$ is fast, otherwise Algorithm 2 with $p_0 = 1$ is usually very slow. This explains the big standard derivation of computational time for $p = 1$. As expected, using $p_0 > 1$ significantly improves the performance of the algorithm. It allows the algorithm to search on a larger dimensional space and find a more reasonable initial point for Algorithm 2 when p finally reduces to 1. The values $p_0 = 4$ and $\delta = 0.9$ are chosen for use with Algorithm 2 in the later comparisons.

6.4 Comparisons with a standard low-rank method

Candes et al., [CESV13, CSV13] use a Matlab library TFOCS [BCG11] that contains a variety of accelerated first-order methods given in [Nes04] and, in particular, the method based on FISTA [BT09] is used to optimize the cost functions in PhaseLift. FISTA [BT09] works as follows. Given an initial point $X^{(0)}$, set $B^{(0)} = X^{(0)}$ and $\theta^{(0)} = 1$, and inductively define

$$X^{(i)} = P_{\mathcal{D}_n}(B^{(i-1)} - t^{(i)} \operatorname{grad} H(B^{(i-1)})), \tag{6.1}$$

$$\theta^{(i)} = 2(1 + \sqrt{1 + \frac{4}{(\theta^{(i-1)})^2}})^{-1},$$

$$\beta^{(i)} = \theta^{(i)}((\theta^{(i-1)})^{-1} - 1),$$

$$B^{(i)} = X^{(i)} + \beta^{(i)}(X^{(i)} - X^{(i-1)}), \tag{6.2}$$

where $t^{(i)}$ is an appropriate step size, e.g., by back tracking. For large scale problems, matrix $X^{(i)}$ is stored by its low-rank approximation computed via projection, i.e., $\sum_{j=1}^{k} \max(\sigma_j^{(i)}, 0) v_j^{(i)} (v_j^{(i)})^*$, where $\sum_{j=1}^{n} \sigma_j^{(i)} v_j^{(i)} (v_j^{(i)})^*$ is an eigenvalue decomposition of $X^{(i)}$ and eigenvalues satisfies $\sigma_1^{(i)} \geq \sigma_2^{(i)} \geq \ldots \geq \sigma_n^{(i)}$. The orthogonal projection (6.1) is obtained by using "eigs" with function handle providing matrix vector multiplication since the matrix vector multiplication of $(B^{(i-1)} - t^{(i)} \operatorname{grad} H(B^{(i-1)}))v$ is cheap for any

Table 3: Comparisons of Algorithm 2 and LR-FISTA for the noiseless PhaseLift problem (3.2) with $n_1 = n_2 = 64$ and several values of k. ♯ represents the number of iterations reach the maximum.

noiseless	Algorithm 2	LR-FISTA				
		1	2	4	8	16
$iter$	104	492	484	521	1538	2000
nf	108	1064	1004	1113	3323	4317
ng	104	532	502	556	1661	2158
f_f	3.88_{-15}	3.54_{-12}	3.20_{-12}	7.41_{-12}	1.52_{-11}	1.20_{-9}
RMSE	1.93_{-7}	6.20_{-6}	4.68_{-6}	1.59_{-5}	8.92_{-5}	2.64_{-3}
t	8.37_{-1}	59.6	53.3	80.7	337	676

$v \in \mathbb{C}^n$. The low-rank subtraction (6.2) is computed exactly by doubling the storage. LR-FISTA is used to denote the low-rank version of FISTA.

As in [CESV13], the difference between the true solution and the minimizer is measured by the relative mean-square error (RMSE) that is defined to be $\min_{a:|a|=1} \|ax - x_*\|_2/\|x_*\|_2$ and RMSE in dB is defined by $10\log_{10}(\text{RMSE})$. The scale of the noise is measured by the signal-to-noise ratio (SNR) in dB that is defined to be $\text{SNR} = 10\log_{10}(\|b\|_2^2/\|b - \hat{b}\|_2^2)$, where $b = \text{diag}(Zx_*x_*^*Z^*)$ and $\hat{b}$ is the noise measurements.

The stopping criterion of LR-FISTA requires that the Frobenius norm of the relative difference between $X^{(i)}$ and $X^{(i-1)}$ is less than 10^{-6} or the number of iterations is greater than 2000, i.e., $\|X^{(i)} - X^{(i-1)}\|_F/\|X^{(i)}\|_F < 10^{-6}$ or $iter > 2000$.

In practice, the choice of κ needs careful consideration. The standard golden section search [Kie53] is used by Candes et al. [CESV13] to find the best κ that gives the smallest RMSE. This method can be used only when the true solution x_* is known. In addition, Candes et al. indicate that one would have to find the best κ via a strategy like cross validation or generalized cross validation. However, since Algorithm 2 was designed with rank in mind, the default choice for problems with and without noise is $\kappa = 1/\sqrt{n}$ if $p > 1$ and $\kappa = 0$ if $p = 1$. κ is chosen to be 0 in LR-FISTA for noiseless measurements in order to recover the exact solution. The effect of using $\kappa > 0$ in LR-FISTA is discussed below when Gaussian noisy measurements are used.

Tables 3 and 4 report experimental results of comparisons of Algorithm 2 and LR-FISTA for the noiseless and Gaussian noise problems (3.2) and (3.4) respectively. For the Gaussian noise problem, τ is 10^{-4} and the corresponding SNR is 31.05 dB in this experiment. Multiple examples with different random seeds and different SNR show similar results. First, increasing k for LR-FISTA usually does not improve the performance in the sense of efficiency and effectiveness for both noiseless and Gaussian noise problems. Second, increasing κ usually does not reduce the RMSE. When it does, the RMSE values are not reduced significantly. Third, Algorithm 2 outperforms LR-FISTA significantly in the sense that Algorithm 2 provides similar accuracy usually while requiring fewer operations of all types (cost function evaluation, gradients etc.) and yielding a significantly smaller computational time.

6.5 Performance of PhaseLift on natural images

Three images of different sizes, shown in Figures 1, 2 and 3, are used to illustrate the performance of Algorithm 2 for noiseless measurements. Table 5 reports the computational time with the default parameters and variant number of masks l for Figures 1 and 2. Even though using 6 masks empirically is sufficient to recover the images, it may not be a good choice in the sense of efficiency. Among the number of masks l we tested, 18 masks $l = 18$ requires least computational time. As shown in Figure 3, the Riemannian method is able to recover an 2D image of 1800 by 2880 pixels within an hour.

Table 4: Comparisons of Algorithm 2 and LR-FISTA for the noise PhaseLift problem (3.4) with SNR be 31.05 dB, $n_1 = n_2 = 64$ and several values of k and κ. $\sharp$ represents the number of iterations reach the maximum.

noise	Algorithm 2	κ	LR-FISTA				
			1	2	4	8	16
iter	79	10^{-2}	288	1574	$2000^{\sharp}$	$2000^{\sharp}$	$2000^{\sharp}$
		10^{-6}	515	$2000^{\sharp}$	$2000^{\sharp}$	$2000^{\sharp}$	$2000^{\sharp}$
		0	509	$2000^{\sharp}$	$2000^{\sharp}$	$2000^{\sharp}$	$2000^{\sharp}$
nf	83	10^{-2}	620	3378	4261	4285	4275
		10^{-6}	1108	4286	4315	4323	4323
		0	1098	4314	4309	4331	4323
ng	79	10^{-2}	387	2694	3651	3344	3584
		10^{-6}	554	2143	2157	2161	2161
		0	549	2157	2154	2165	2161
f_f	4.08_{-7}	10^{-2}	1.63_{-1}	1.76_{-1}	2.24_{-1}	2.75_{-1}	3.04_{-1}
		10^{-6}	1.84_{-5}	1.92_{-5}	2.37_{-5}	3.56_{-5}	7.82_{-5}
		0	4.08_{-7}	1.17_{-6}	6.01_{-6}	2.54_{-5}	9.31_{-5}
RMSE	6.71_{-4}	10^{-2}	1.80_{-1}	2.63_{-1}	3.60_{-1}	4.19_{-1}	4.45_{-1}
		10^{-6}	6.72_{-4}	1.11_{-3}	2.13_{-3}	3.52_{-3}	6.37_{-3}
		0	6.70_{-4}	1.10_{-3}	2.14_{-3}	4.05_{-3}	7.46_{-3}
t	6.59_{-1}	10^{-2}	34.0	304	445	571	948
		10^{-6}	61.6	255	302	452	701
		0	59.8	261	292	442	687

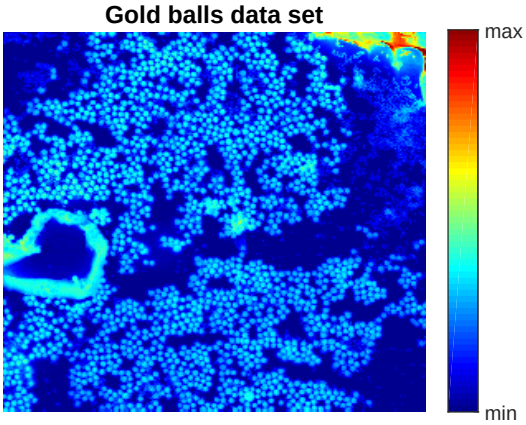

Figure 1: A gold balls data set image of 256 by 256 pixels. The values of pixels are complex numbers. The computational time is given in Table 5. The RMSE is -80.1 dB when l is 18.

Table 5: Computational time on the natural images in Figures 1 and 2. t_0 and t_R denote the computational time (in second) of initial iterate (Algorithm 3) and Algorithm 2, respectively.

	l	6	12	18	24	30
Figure 1	t_0	2.23	3.53	4.86	6.61	8.08
	t_R	22.14	9.15	7.61	8.45	10.08
Figure 2	t_0	79.8	150.7	216.1	286.4	345.9
	t_R	1374.4	605.6	470.0	622.5	425.9

Figure 2: A gray image of 1224 by 1632 pixels in Keukenhof park. The values of pixels are real numbers. The computational time is given in Table 5. The RMSE is -81.3 dB when l is 18.

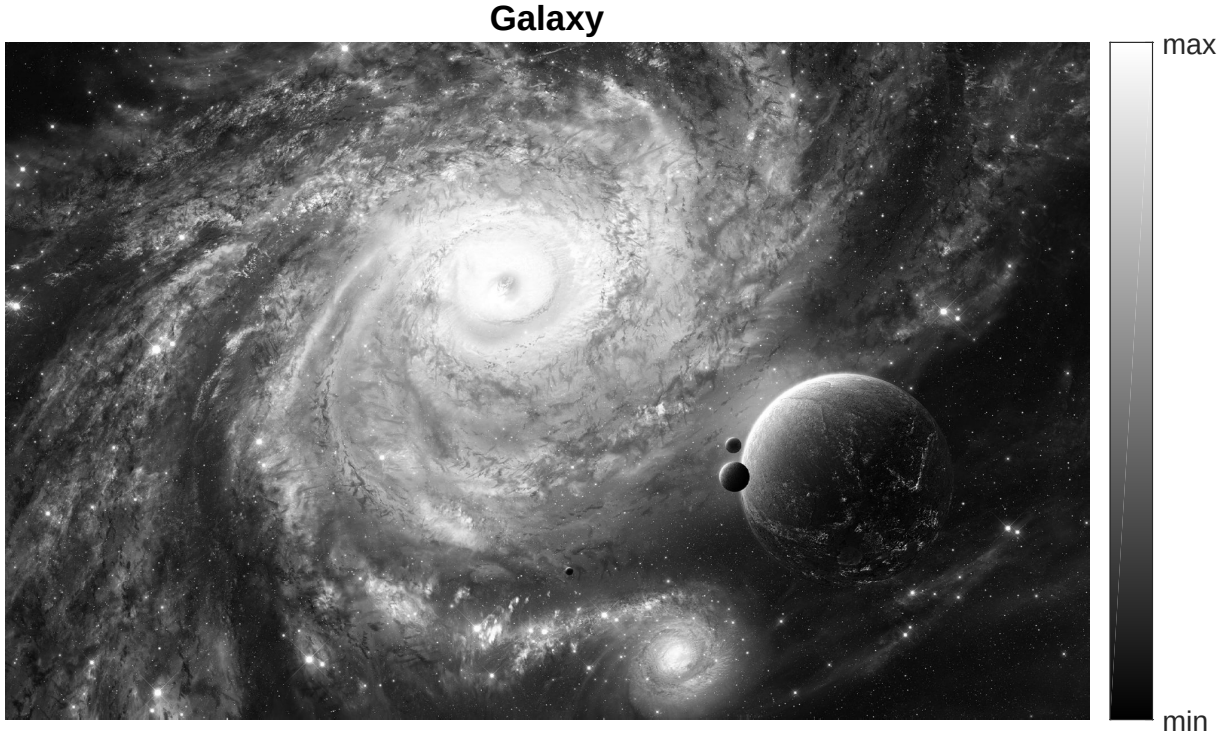

Figure 3: A gray galaxy image of 1800 by 2880 pixels. The values of pixels are real numbers. The computational times of initial iterate and Algorithm 2 are 536.6 seconds and 1826 seconds, respectively. The RMSE is -82.3 dB for the chosen $l = 18$.

7 Conclusion

In this paper, the recently proposed PhaseLift framework for solving the phase retrieval problem has motivated the consideration of cost functions H on the set of complex Hermitian positive semidefinite matrices $\mathcal{D}_n$ that include the PhaseLift cost function.

An alternate cost function F related to factorization is used to replace the cost function H, i.e., $F(Y) = H(YY^*)$. The optimality conditions of H are related to the properties of F and the important optimality condition, Theorem 4.1, shows that if Y_p is a rank deficient minimizer of F_p, then $Y_pY_p^*$ is a stationary point of H. For general problems defined on $\mathcal{D}_n$, if r_*, the rank of the desired minimizer of cost function H, is low, the optimality condition suggests the use of the alternate cost function F with $p > r_*$. If r_* is small, then a small p can be used and optimization on F_p can be more efficient than optimization on H.

Additionally, Algorithm 2 based on optimization on a fixed rank manifold and dynamically reducing is developed for optimizing the cost function F. For optimization on a fixed rank manifold, recently developed state-of-the-art Riemannian optimization methods on a quotient space are used.

For the case of the noiseless phase retrieval problem in the PhaseLift framework, obtaining a rank-one minimizer Y_p of F_p with $p > 1$ is shown to be equivalent to obtaining the rank-one global minimizer $Y_pY_p^*$ of H. Empirically, in finite precision arithmetic, choosing $p_0 > 1$ and using Algorithm 2 always yields a (approximately) rank-one minimizer for both the noiseless and noisy phase retrieval problems. The computational time of Algorithm 2 with LRBFGS is demonstrated to be significantly smaller than that of LR-FISTA with accuracy is at least as good as LR-FISTA when the latter manages to converge in an acceptable amount of time. In [CESV13], it is pointed out that the algorithm LR-FISTA may be too slow for large-scale images and development of a fast algorithm is a future research. The unacceptably large computational time of LR-FISTA is empirically verified here and Algorithm 2 using the limited memory forms of the Riemannian fixed-rank optimization algorithms, specifically LRBFGS for the phase retrieval problems, are clearly seen to be practical in terms of computational time and recovery quality for a wide range of problem sizes including those for which LR-FISTA fails.

Acknowledgement

We thank Stefano Marchesini at Lawrence Berkeley Notional Laboratory for providing the gold balls data set and granting permission to use it.

References

[AIDV09] P.-A. Absil, M. Ishteva, L. De Lathauwer, and S. Van Huffel. A geometric Newton method for Oja's vector field. *Neural Computation*, 21(5):1415–33, May 2009. doi:10.1162/neco.2008.04-08-749.

[AMS08] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, Princeton, NJ, 2008.

[Bak08] C. G. Baker. *Riemannian manifold trust-region methods with applications to eigenproblems*. PhD thesis, Florida State University, Department of Computational Science, 2008.

[BCG11] S. Becker, E. J. Cand, and M. Grant. Templates for convex cone problems with applications to sparse signal recovery. *Mathematical Programming Computation*, 3:165–218, 2011.

[BDP+07] O. Bunk, A. Diaz, F. Pfeiffer, C. David, B. Schmitt, D. K. Satapathy, and J. F. van der Veen. Diffractive imaging for periodic samples: retrieving one-dimensional concentration profiles across microfluidic channels. *Acta crystallographica. Section A, Foundations of crystallography*, 63(Pt 4):306–314, July 2007. doi:10.1107/S0108767307021903.

[Bla04] R. Blankenbecler. Three-dimensional image reconstruction. II. Hamiltonian method for phase recovery. *Physical Review B*, 69(6):064108, February 2004. doi:10.1103/PhysRevB.69.064108.

[BM03] S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, February 2003. doi:10.1007/s10107-002-0352-8.

[Boo86] W. M. Boothby. *An introduction to differentiable manifolds and Riemannian geometry*. Academic Press, second edition, 1986.

[BS79] Y. M. Bruck and L. G. Sodin. On the ambiguity of the image recontruction problem. *Optics Communications*, 30(3):304–308, 1979.

[BT09] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, January 2009. doi:10.1137/080716542.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004. doi:10.1017/CBO9780511804441.

[CESV13] E. J. Candès, Y. C. Eldar, T. Strohmer, and V. Voroninski. Phase retrieval via matrix completion. *SIAM Journal on Imaging Sciences*, 6(1):199–225, 2013. arXiv:1109.0573v2.

[CL13] E. J. Candès and X. Li. Solving quadratic equations via phaselift when there are about as many equations as unknowns. *Foundations of Computational Mathematics*, June 2013. doi:10.1007/s10208-013-9162-z.

[CLS16] E. J. Candés, X. Li, and M. Soltanolkotabi. Phase retrieval via Wirtinger flow: theory and algorithms. *IEEE Transactions on Information Theory*, 64(4):1985–2007, 2016.

[CMWL07] C. Chen, J. Miao, C. w. Wang, and T. K. Lee. Application of optimization technique to noncrystalline x-ray diffraction microscopy: Guided hybrid input-output method. *Physical Review B*, 76(6):064113, August 2007. doi:10.1103/PhysRevB.76.064113.

[CSV13] E. J. Candès, T. Strohmer, and V. Voroninski. PhaseLift : Exact and stable signal recovery from magnitude measurements via convex programming. *Communications on Pure and Applied Mathematics*, 66(8):1241–1274, 2013.

[DH14] L. Demanet and P. Hand. Stable optimizationless recovery from phaseless linear measurements. *Journal of Fourier Analysis and Applications*, 20(1):199–221, 2014.

[Els03] V. Elser. Solution of the crystallographic phase problem by iterated projections. *Section A: Foundations of Crystallography*, pages 201–209, 2003.

[Fie78] J. R. Fienup. Reconstruction of an object from the modulus of its Fourier transform. *Optics letters*, 3(1):27–29, 1978.

[Fie82] J. R. Fienup. Phase retrieval algorithms: a comparison. *Applied optics*, 21(15):2758–69, August 1982.

[GS72] R. W. Gerchberg and W. O. Saxton. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35:237–246, 1972.

[GW04] M. X. Goemans and D. P. Williamson. Approximation algorithms for max-3-cut and other problems via complex semidefinite programming. *Journal of Computer and System Sciences*, 68(2):442–470, March 2004. doi:10.1016/j.jcss.2003.07.012.

[HAG15] W. Huang, P.-A. Absil, and K. A. Gallivan. A Riemannian symmetric rank-one trust-region method. *Mathematical Programming*, 150(2):179–216, February 2015.

[HAG16] Wen Huang, P.-A. Absil, and K. A. Gallivan. Intrinsic representation of tangent vectors and vector transport on matrix manifolds. Technical Report UCL-INMA-2016.08, U.C.Louvain, 2016.

[Har93] R. W. Harrison. Phase problem in crystallography. *Journal of the Optical Society of America A*, 10(5):1046–1055, May 1993. doi:10.1364/JOSAA.10.001046.

[Hay82] M. H. Hayes. The reconstruction of a multidimensional sequence from the phase or magnitude of its fourier transform. *IEEE Transactions on Acoustics speech and signal processing*, 30(2):140–154, 1982.

[HGA15] Wen Huang, K. A. Gallivan, and P.-A. Absil. A Broyden Class of Quasi-Newton Methods for Riemannian Optimization. *SIAM Journal on Optimization*, 25(3):1660–1685, 2015.

[HGZ16] Wen Huang, K. A. Gallivan, and Xiangxiong Zhang. Solving phaselift by low rank Riemannian optimization methods. In *Proceedings of of the International Conference on Computational Science (ICCS2016), accepted*, 2016.

[Hua13] W. Huang. *Optimization algorithms on Riemannian manifolds with applications*. PhD thesis, Florida State University, Department of Mathematics, 2013.

[JBAS10] M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

[Kie53] J. Kiefer. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4:502–506, 1953.

[Mar07] S. Marchesini. Invited article: a unified evaluation of iterative projection algorithms for phase retrieval. *Review of scientific instruments*, 78(1):011301, January 2007. doi:10.1063/1.2403783.

[MISE08] J. Miao, T. Ishikawa, Q. Shen, and T. Earnest. Extending X-ray crystallography to allow the imaging of noncrystalline materials, cells, and single protein complexes. *Annual review of physical chemistry*, 59:387–410, January 2008. doi:10.1146/annurev.physchem.59.032607.093642.

[Nes04] Y. Nesterov. *Introductory lectures on convex programming: a basic course*, volume I. Springer, 2004.

[RW12] W. Ring and B. Wirth. Optimization methods on Riemannian manifolds and their application to shape space. *SIAM Journal on Optimization*, 22(2):596–627, January 2012. doi:10.1137/11082885X.

[San85] J. L. C. Sanz. Mathematical considerations for the problem of Fourier transform phase retrieval from magnitude. *SIAM Journal on Applied Mathematics*, 45(4):651–664, 1985.

[Sat15] H. Sato. A Dai-Yuan-type Riemannian conjugate gradient method with the weak Wolfe conditions. *Computational Optimization and Applications*, 2015. to appear.

[SI15] H. Sato and T. Iwai. A new, globally convergent Riemannian conjugate gradient method. *Optimization*, 64(4):1011–1031, February 2015.

[TTT99] K. C. Toh, M. J. Todd, and R. H. Tutuncu. SDPT3 a matlab software package for semidefinite programming. *Optimization methods and software*, 11:545–581, 1999.

[Wal63] A. Walther. The question of phase retrieval in optics. *Optica Acta: International Journal of Optics*, 10(1):41–49, January 1963. doi:10.1080/713817747.

[WDAM13] I. Waldspurger, A. D´ Aspremont, and S. Mallat. Phase recovery, maxcut and complex semidefinite programming. *Mathematical Programming*, December 2013. doi:10.1007/s10107-013-0738-9.

A Stationary Points

The definition of stationary points in real semidefinite programming can be used to characterize stationary points of (4.1). To this end, we first define a mapping, denoted by a superscript $\check{}$:

$$\check{}: \mathbb{C}^{n\times p} \to \mathbb{R}^{2n\times 2p} : Y = \check{Y}_1 + \check{Y}_2\sqrt{-1} \mapsto \check{Y} = \begin{pmatrix} \check{Y}_1 & -\check{Y}_2 \\ \check{Y}_2 & \check{Y}_1 \end{pmatrix}$$

which is an isometry from $\mathbb{C}^{n\times p}$ to $\mathbb{R}^{2n\times 2p}$ [GW04]. Therefore, (1.2) can be formulated as a problem with real semidefinite constraints:

$$\begin{aligned} \min_{\hat{X}\in\mathbb{S}^+_{2n}} \tilde{H}(\hat{X}) &:= H(\check{X}_1 + \check{X}_2\sqrt{-1}) \\ \text{such that } \operatorname{tr}(A_k\hat{X}) &= 0, k = 1, 2, \ldots, n(n+1), \end{aligned} \tag{A.1}$$

where A_k, $k = 1, \ldots, n(n+1)/2$ are given by

$$A_k = \begin{pmatrix} 0_{n\times n} & E_{ij} + E_{ji} \\ E_{ij} + E_{ji} & 0_{n\times n} \end{pmatrix}, \quad i = 1, \ldots, n, \;\; j = i, \ldots, n$$

and the $n(n+1)/2$ remaining A_k, are given by

$$A_k = \begin{pmatrix} E_{ij} + E_{ji} & 0_{n\times n} \\ 0_{n\times n} & -E_{ij} - E_{ji} \end{pmatrix}, \quad i = 1, \ldots, n, \;\; j = i, \ldots, n$$

where $E_{ij} \in \mathbb{R}^{n\times n}$ are the standard basis matrices.

Since $\tilde{H}$ in (A.1) is defined on a real space, [JBAS10, Definition 1] is applicable:

Definition A.1. *A stationary point of* (A.1) *is a symmetric matrix $\hat{X} \in \mathbb{R}^{2n\times 2n}$ for which there exists a vector $\delta = (\delta_1, \ldots, \delta_m)^T \in \mathbb{R}^m$ and a symmetric matrix $S \in \mathbb{R}^{2n\times 2n}$ such that the first-order optimality conditions hold:* $\operatorname{tr}(A_i\hat{X}) = 0, \quad \hat{X} \geq 0, \quad S \geq 0, S\hat{X} = 0, \quad S = \operatorname{grad}\tilde{H}(\hat{X}) - \sum_{i=1}^m \delta_i A_i$, *where* $m = n(n+1)$.

One can define X to be a stationary point of H if and only if $\tilde{X}$ is a stationary point of $\tilde{H}$, since $\tilde{H}$ is a reformulation of H. Therefore, by using Definition A.1, Lemma A.1 provides a necessary and sufficient condition for X to be a stationary point of H.

Lemma A.1. *Suppose $X \in \mathcal{D}_n$.* $\operatorname{grad} H(X)X = 0$ *and* $\operatorname{grad} H(X) \geq 0$ *if and only if X is a stationary point of H.*

Proof. $\Rightarrow$ The conditions $\operatorname{grad} H(X)X = 0$ and $\operatorname{grad} H(X) \geq 0$ imply $\operatorname{grad}\tilde{H}(\tilde{X}) \geq 0$ and $\operatorname{grad}\tilde{H}(\tilde{X})\tilde{X} = 0$. Therefore, choosing $S = \operatorname{grad}\tilde{H}(\tilde{X})$ and $\delta = 0_m$ in Definition A.1 yields that X is a stationary point of H.

$\Leftarrow$ Let G denote $\operatorname{grad} H(X)$. The gradient of $\tilde{H}$ can be written as $\tilde{G}$. Therefore, using $S = \tilde{G} - \sum_{i=1}^m \delta_i A_i$ in Definition A.1 yields $\delta_i = 0$ for all i. It follows from $S\tilde{X} = 0, S \geq 0$ that $\tilde{G}\tilde{X} = 0$ and $\tilde{G} \geq 0$. This implies $\operatorname{grad} H(X)X = 0$ and $\operatorname{grad} H(X) \geq 0$. □