Optimization Algorithms Exploiting Unitary Constraints

Jonathan H. Manton, *Member, IEEE*

***Abstract*—This paper presents novel algorithms that iteratively converge to a local minimum of a real-valued function $f(X)$ subject to the constraint that the columns of the complex-valued matrix X are mutually orthogonal and have unit norm. The algorithms are derived by reformulating the constrained optimization problem as an unconstrained one on a suitable manifold. This significantly reduces the dimensionality of the optimization problem. Pertinent features of the proposed framework are illustrated by using the framework to derive an algorithm for computing the eigenvector associated with either the largest or the smallest eigenvalue of a Hermitian matrix.**

***Index Terms*—Constrained optimization, eigenvalue problems, optimization on manifolds, orthogonal constraints.**

I. Introduction

THIS paper derives novel algorithms for numerically minimizing a cost function $f(X)$, $f: \mathbb{C}^{n\times p} \to \mathbb{R}$ subject to the orthogonality constraint $X^H X = I$, where H denotes Hermitian transpose, and I is the identity matrix. The complex-valued case is considered for generality; the results in this paper remain valid if all quantities are restricted to being real-valued. It has been shown recently [8] that the geometrically correct setting for this constrained minimization problem is on the Stiefel manifold in general and on the Grassmann manifold if $f(X)$ possesses the symmetrical property that $f(XQ) = f(X)$ for any unitary (that is, $Q^H Q = QQ^H = I$) matrix $Q \in \mathbb{C}^{p\times p}$. However, not only did [8] consider only the real-valued case, the approach therein relied on endowing the Stiefel manifold with a Riemannian structure. The present paper presents a simpler framework for orthogonally constrained optimization problems.

Orthogonally constrained optimization problems tend to occur in signal processing problems involving subspaces. This is because the constraint $X^H X = I$ requires the columns of X to form an orthonormal basis, meaning that the cost function $f(X)$ can be interpreted as a function of an ordered set of orthonormal basis vectors. Similarly, if $f(X) = f(XQ)$ for any unitary matrix Q, then $f(X)$ is a function of the subspace spanned by the columns of X, or equivalently, the range space of X. This is because the range spaces of X and XQ are the same.

Manuscript received November 13, 2000; revised November 20, 2001. This work was supported by the Australian Research Council and was performed in part while the author was visiting the Hong Kong University of Science and Technology. The associate editor coordinating the review of this paper and approving it for publication was Prof. Jian Li.

The author is with the ARC Special Research Centre for Ultra-Broadband Information Networks, Department of Electrical and Electronic Engineering, The University of Melbourne, Parkville, Australia (e-mail: j.manton@ee.mu.oz.au).

Publisher Item Identifier S 1053-587X(02)01329-6.

It is candidly stated that in terms of convergence speed and computational complexity, the proposed algorithms are not necessarily the best algorithms for any given cost function. Rather, the justification for this work is that it is believed to be the first work that presents general purpose and ready-to-use algorithms for solving optimization problems with complex-valued orthogonality constraints. Indeed, the only prerequisite for being able to implement the four main optimization algorithms (namely, Algorithm 15 of Section V-A, Algorithm 17 of Section V-B, Algorithm 24 of Section VII-A, and Algorithm 26 of Section VII-B) is to be able to compute the derivative and Hessian of a cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, as defined in Section II. The relevance of these algorithms to signal processing is now delineated.

A. Applications

As the opening paragraph of [29] states, many signal processing tasks involve the constrained minimization of the function $f(X) = \mathrm{tr}\{X^H R X\}$, where R is a possibly time-varying covariance matrix. If the constraint is $X^H X = I$, then it is by now well known [10], [29] that the minimum occurs when the p columns of X span the same subspace as spanned by the eigenvectors associated with the p smallest eigenvalues of R; this is an example where $f(X) = f(XQ)$ for any unitary Q. Therefore, the algorithms in this paper, when applied to the specific cost function $f(X) = \mathrm{tr}\{X^H R X\}$, complement the growing literature on subspace estimation and tracking problems [6], [7], [11], [14], [15], [19], [23], [25], [37] with applications in antenna array processing [27], [36], frequency estimation [26], and so forth. This is discussed further in Section VIII.

Other problems in linear algebra can also be expressed as orthogonally constrained minimization problems [3], [10]. Examples include finding the singular value decomposition (SVD) of a matrix and computing a total least-squares solution [9]. Whereas iterative methods, such as those presented here, have yet to outperform traditional methods for solving linear algebra problems in general, one advantage of iterative methods is their applicability to adaptive engineering applications where minor corrections to present estimates need to be performed regularly. Another advantage is their computational robustness; iterative refinement can be used to improve a solution obtained by traditional means.

When no closed-form solution exists to a problem, it becomes necessary to use an iterative method. Several examples appearing in the more recent literature are now given.

The joint diagonalization problem, which appears in blind source separation [1], [24], blind beamforming [4], [32], [34], and other [20], [33] applications, is to find a unitary matrix X

such that for given matrices $A_1, \ldots, A_m \in \mathbb{C}^{n\times n}$, the matrices $X^H A_i X$ for $i = 1, \ldots, m$ are all (approximately) diagonal. It can be posed as an orthogonally constrained minimization problem by choosing the cost function $f(X)$ to be the sum of the squares of the off-diagonal elements of the $X^H A_i X$ (cf. [1, Eq. (20)]).

The weighted low-rank approximation problem [12] is to find a matrix having a prespecified rank and that best approximates a given matrix under a weighted norm. It is shown in [18] that the weighted low rank approximation problem can be reformulated as an orthogonally constrained minimization problem, and moreover, it is proved that the formulation is the most natural one because it uses the least possible number of parameters. A similar idea is used in [17] to reformulate the convolutive reduced rank Wiener filtering problem as an orthogonally constrained minimization problem. (Prior to this reformulation, the authors were unaware of any general solution to the convolutive reduced-rank Wiener filtering problem.)

In summary, a significant number of engineering problems can be formulated naturally as an orthogonally constrained minimization problem. The present paper not only provides novel algorithms for solving such problems, but it provides a general framework in which existing methods can be better understood.

B. Related Work

Although the references in [31] show that the theory behind the minimization of a cost function on a manifold was already being studied in the seventies, general-purpose algorithms for solving orthogonally constrained minimization problems did not appear until the 1990s [8], [13], [28].

The key principle in [8], [13], [28] was to exploit the geometry of the constraint surface, and in all cases, the algorithms performed a series of descent steps, with each descent step taken along a geodesic. (A geodesic is the generalization of a straight line to curved surfaces.)

The present paper breaks with tradition by not moving along geodesics. The reason for this is now explained. There is no inherent connection between the (Riemannian) geometry imposed in [8], [13], and [28] on the constraint surface $\{X \in \mathbb{R}^{n\times p} : X^T X = I\}$ and an arbitrary cost function $f(X)$. (See Section IX for greater detail.) That is to say, although moving along geodesics is a sensible thing to do, it is not the only sensible thing that can be done. A disadvantage of moving along geodesics is the computational cost involved in computing the path of a geodesic. This paper, by choosing not to follow geodesics, is able to achieve a modest reduction in the computational complexity of the algorithms.

The main distinction of the present paper, however, is that it considers the complex-valued case. The reason why the algorithms in [8], [13], and [28] do not generalize immediately to the complex-valued case is because a nonconstant cost function $f : \mathbb{C}^{n\times p} \to \mathbb{R}$ cannot be analytic. This means that the gradient and Hessian of $f(X)$ on a complex Riemannian manifold are not well defined. Although the solution is straightforward—simply treat $f(X)$ as a function of the real and imaginary components of X (that is, $f : \mathbb{R}^{n\times p} \times \mathbb{R}^{n\times p} \to \mathbb{R}$) whenever appropriate—it necessitates the algorithms for the complex-valued case to be derived from scratch.

C. Outline of Paper

Two types of algorithms are derived in this paper. The first type is based on the traditional steepest descent algorithm coupled with Armijo's method for choosing the step size at each iteration [22]. Although steepest descent algorithms were derived in [7], only the case of X being a column vector was considered, and no step size rule was given. Steepest descent-type algorithms were not explicitly considered in [8], which concentrated instead on conjugate gradient and Newton-type methods.

The second type of algorithm derived in this paper is based on the traditional Newton algorithm [22]. Although [8] derived similar Newton type algorithms to the ones presented here (albeit for the real-valued case only), there is an important difference; in [8], the pertinent manifold was locally parameterized by using the exponential map, whereas this paper locally parameterizes the manifold by a Euclidean projection of the tangent space onto the manifold. This difference affects the computational complexity and the rate of convergence of the algorithms. The performance of the algorithms resulting from the two different parameterizations was compared in [18] for a particular cost function, and it was shown that the Euclidean-projection-based parameterization resulted in less computational complexity and faster convergence. (For other cost functions, the converse may well be true. A more detailed discussion appears in Section IX.)

Remark: Each algorithm is not simply an application of the steepest descent or Newton method in some parameter space of reduced dimension. The novel feature is that the local cost function to which the steepest descent or Newton method is applied changes at each iteration.

The reason for presenting both steepest descent and Newton type algorithms is because each one has its own advantages and disadvantages. Steepest descent-type algorithms coupled with Armijo's step-size rule almost always converge to a local minimum [22]. However, their rate of convergence is only linear, meaning that asymptotically, the number of correct digits increases by a fixed amount per iteration (see [22] for a precise definition). Newton-type algorithms, by using second-order derivatives, are able to achieve quadratic convergence, meaning that the number of correct digits ultimately doubles per iteration. This faster rate of convergence comes with two disadvantages: increased computational complexity per iteration and no guarantee that the algorithm will converge to a local minimum. Indeed, without appropriate checks, the Newton method will converge to the closest critical point, whether it is a local maximum, local minimum, or a saddle point. In practice, the steepest descent and Newton algorithms are often used together; a few iterations of the steepest descent algorithm are performed first to move close to a local minimum before the Newton algorithm is applied.

The rest of this paper is organized as follows. Section II defines the derivative and Hessian of a cost function $f : \mathbb{C}^{n\times p} \to \mathbb{R}$ and is the only prerequisite for being able to implement the algorithms in this paper. The theory behind these algorithms is covered in Sections III–VII. Section III derives formulae for calculating the critical point of a quadratic function defined on various vector spaces. These formulae are required in subsequent

sections of the paper. Section IV introduces the complex Stiefel manifold, its tangent space, and the Euclidean projection operator. Section V derives two algorithms for minimizing a cost function on the Stiefel manifold. Similarly, Section VI introduces the Grassmann manifold, and Section VII derives algorithms for minimizing a cost function on the Grassmann manifold. A worked example, which is of independent interest in its own right, is given in Section VIII. Section IX discusses various aspects of the optimization framework presented here. Section X concludes the paper.

Notation: The superscripts T and H denote transpose and Hermitian transpose, respectively. The Frobenius norm is used throughout, that is, $\|X\|^2 = \mathrm{tr}\{X^H X\}$, where $\mathrm{tr}\{\cdot\}$ is the trace operator. The vec operator $\mathrm{vec}\{X\}$ is the vector obtained by stacking the columns of the matrix X on top of each other. Kronecker's product is denoted by $\otimes$. The symbol I denotes the identity matrix whose size can be determined from its context. Similarly, I_n denotes the n-by-n identity matrix, whereas $I_{n,p}$ denotes the n-by-p matrix with ones along the diagonal. A square matrix X satisfying $X^H X = I$ is called unitary. Given a matrix $X \in \mathbb{C}^{n\times p}$ satisfying $X^H X = I$, its complement $X_\perp \in \mathbb{C}^{n\times(n-p)}$ is defined to be any matrix that satisfies $[X\ X_\perp]^H[X\ X_\perp] = I$. Since $X_\perp$ is not uniquely defined, implicit in any statement involving $X_\perp$ is that the statement holds for any choice of $X_\perp$. The square matrix A is skew-Hermitian if $A + A^H = 0$. It is Hermitian if $A = A^H$. The subspace spanned by the columns of a matrix X is denoted by $\lfloor X \rfloor$. The expression $O(t^n)$ denotes a (possibly matrix-valued) function of t such that $|t^{-n}O(t^n)|$ remains bounded as $t \to 0$, where $|\cdot|$ denotes absolute value (or norm). The symbol $\jmath$ denotes $\sqrt{-1}$, whereas $\Re$ and $\Im$ denote the real and imaginary parts of a complex quantity, respectively.

II. Second-Order Approximation

This paper chooses to express the second-order Taylor series approximation of an arbitrary (but sufficiently differentiable) function $f\colon \mathbb{C}^{n\times p} \to \mathbb{R}$ in the form

$$f(X+tZ) = f(X) + t\Re\,\mathrm{tr}\{Z^H D_X\} + \frac{t^2}{2}\left(\mathrm{vec}\{Z\}^H H_X \mathrm{vec}\{Z\} + \Re\left\{\mathrm{vec}\{Z\}^T C_X \mathrm{vec}\{Z\}\right\}\right) + O(t^3) \quad (1)$$

where $D_X \in \mathbb{C}^{n\times p}$ is the derivative of f evaluated at X, and $H_X, C_X \in \mathbb{C}^{np\times np}$ are the Hessian of f evaluated at X. To ensure uniqueness, it is required that H_X and C_X satisfy $H_X = H_X^H$ and $C_X = C_X^T$.

Remark: For the real-valued case $f\colon \mathbb{R}^{n\times p} \to \mathbb{R}$, the matrix C_X should be omitted from (1).

The term $\Re\,\mathrm{tr}\{Z^H D_X\}$ in (1) was chosen because the Euclidean inner product $\langle X, Y\rangle = \Re\,\mathrm{tr}\{Y^H X\}$ is the unique inner product inducing the Frobenius norm $\|X\|^2 = \mathrm{tr}\{X^H X\}$ on $\mathbb{C}^{n\times p}$ space. The consequence of this choice is that the ijth element of D_X equals

$$(D_X)_{ij} = \frac{\partial f}{\partial \Re X_{ij}} + \jmath\frac{\partial f}{\partial \Im X_{ij}}. \quad (2)$$

Although similar formulae can be derived for H_X and C_X, it is often easier to determine D_X, H_X, and C_X analogously to the following example.

Example 1: If $f(X) = (1/2)\,\mathrm{tr}\{X^H A X\}$ with $A \in \mathbb{C}^{n\times n}$ Hermitian and $X \in \mathbb{C}^{n\times p}$, then

$$\begin{aligned} f(X+tZ) &= \tfrac{1}{2}\,\mathrm{tr}\{(X+tZ)^H A(X+tZ)\} && (3)\\ &= f(X) + \tfrac{1}{2}\,t\,\mathrm{tr}\{Z^H AX + X^H AZ\} + \tfrac{1}{2}\,t^2\,\mathrm{tr}\{Z^H AZ\} && (4)\\ &= f(X) + t\,\Re\,\mathrm{tr}\{Z^H AX\} + \tfrac{1}{2}\,t^2 \mathrm{vec}\{Z\}^H \mathrm{vec}\{AZ\} && (5)\end{aligned}$$

proving that $D_X = AX$, $C_X = 0$, and $H_X = (I_p \otimes A)$, the last equality following from the fact that $\mathrm{vec}\{AZ\} = (I_p \otimes A)\mathrm{vec}\{Z\}$. Note that $H_X = H_X^H$ and $C_X = C_X^T$, as required.

III. Critical Points of a Quadratic Function

This section derives formulae for finding critical points[1] of the quadratic function $\tilde{g}\colon \mathbb{C}^{n\times p} \to \mathbb{R}$ defined by

$$\tilde{g}(Z) = \Re\,\mathrm{tr}\{Z^H D\} + \tfrac{1}{2}\,\mathrm{vec}\{Z\}^H H\,\mathrm{vec}\{Z\} + \tfrac{1}{2}\,\Re\left\{\mathrm{vec}\{Z\}^T C\,\mathrm{vec}\{Z\}\right\} \quad (6)$$

where $D \in \mathbb{C}^{n\times p}$, H, $C \in \mathbb{C}^{np\times np}$ are arbitrary matrices satisfying $H = H^H$ and $C = C^T$, subject to Z being restricted to one of the following three vector spaces. The reason for considering these particular vector spaces will become clear in Sections V-B and VII-B. Let $X \in \mathbb{C}^{n\times p}$ and $X_\perp \in \mathbb{C}^{n\times(n-p)}$ be given matrices such that $[X\ X_\perp]^H[X\ X_\perp] = I$.

- **V1)** $V_1 = \{Z \in \mathbb{C}^{n\times p}\colon Z = XA + X_\perp B\}$, where $A \in \mathbb{C}^{p\times p}$ is skew-Hermitian, and $B \in \mathbb{C}^{(n-p)\times p}$ is arbitrary.
- **V2)** $V_2 = \{Z \in \mathbb{C}^{n\times p}\colon Z = XA + X_\perp B\}$, where $A \in \mathbb{C}^{p\times p}$ is skew-Hermitian and has zero diagonal elements ($A_{ii} = 0$), whereas $B \in \mathbb{C}^{(n-p)\times p}$ is arbitrary.
- **V3)** $V_3 = \{Z \in \mathbb{C}^{n\times p}\colon Z = X_\perp B\}$, where $B \in \mathbb{C}^{(n-p)\times p}$ is arbitrary.

Since $\tilde{g}(Z)$ is not analytic, it is necessary to think of it as a quadratic function in the real and imaginary parts of Z. The consequence is that the vector spaces V_1, V_2, V_3 are treated as real vector spaces.

Proposition 2: Let $V \subset \mathbb{C}^{n\times p}$ be a real vector space (such as V_1, V_2 or V_3), and define $\tilde{g}\colon V \to \mathbb{R}$ as in (6). The point $Z = Z^{(\mathrm{cp})}$ is a critical point of $\tilde{g}(Z)$ if and only if $Z^{(\mathrm{cp})}$ is a matrix in V satisfying the linear constraints

$$\forall\, Z \in V, \quad \Re\left\{\mathrm{tr}\{Z^H D\} + \left[\mathrm{vec}\{Z\}^H H + \mathrm{vec}\{Z\}^T C\right]\mathrm{vec}\{Z^{(\mathrm{cp})}\}\right\} = 0. \quad (7)$$

Proof: Let $E_i \in \mathbb{C}^{n\times p}$ for $i = 1, \ldots, d$ be an arbitrary basis for V, where d is the dimension of V. Express $Z = \sum_{i=1}^{d} \alpha_i E_i$, where $\alpha_i \in \mathbb{R}$. Then

$$\frac{\partial \tilde{g}(Z)}{\partial \alpha_i} = \Re\left\{\mathrm{tr}\{E_i^H D\} + \mathrm{vec}\{E_i\}^H H\,\mathrm{vec}\{Z\} + \mathrm{vec}\{E_i\}^T C\,\mathrm{vec}\{Z\}\right\}. \quad (8)$$

[1] A critical point of a function is a point at which the first-order directional derivatives are all zero. A nondegenerate quadratic function has a unique critical point.

TABLE I
MATLAB CODE THAT COMPUTES THE CRITICAL POINT OF THE QUADRATIC FUNCTION $\tilde{g}(Z) = \Re\,\mathrm{tr}\{Z^H D\} + (1/2)\mathrm{vec}\{Z\}^H H\,\mathrm{vec}\{Z\} + (1/2)\Re\{\mathrm{vec}\{Z\}^T C\,\mathrm{vec}\{Z\}\}$, WHERE Z IS RESTRICTED TO BE OF THE FORM $Z = XA + X_\perp B$ WITH A SKEW-HERMITIAN (cf. SECTION III). IT IS REQUIRED THAT $[X\ X_\perp]^H[X\ X_\perp] = I$, $H = H^H$ AND $C = C^T$. (`Xc` CORRESPONDS TO $X_\perp$.)

```
function [Z] = cpoint(X,Xc,D,H,C);
[n,p] = size(X); np = n*p; d = p*(2*n-p);
% Form basis for tangent space
E = zeros(n,p,d); i = 1;
for r=1:p  % Diagonal elements of A
    M = zeros(p,p); M(r,r) = 1j; E(:,:,i) = X*M; i = i+1;
end
for r=1:p-1  % Off-diagonal elements of A
for c=r+1:p
    M = zeros(p,p); M(r,c) = 1;  M(c,r) = -1; E(:,:,i) = X*M; i = i+1;
    M(r,c) = 1j; M(c,r) = 1j; E(:,:,i) = X*M; i = i+1;
end
end
for r=1:n-p  % Elements of B
for c=1:p
    M = zeros(n-p,p); M(r,c) = 1; E(:,:,i) = Xc*M; i = i+1;
    M(r,c) = 1j; E(:,:,i) = Xc*M; i = i+1;
end
end
% Form linear equation and solve for alpha
A = zeros(d,d); b = zeros(d,1); vD = reshape(D,np,1);
for r=1:d
    vEr = reshape(E(:,:,r),np,1); vErHC = vEr'*H + vEr.'*C;
    for c=1:d
        A(r,c) = real(vErHC*reshape(E(:,:,c),np,1));
    end
    b(r) = real(vEr'*vD);
end
alpha = -(A\b);
% Recover Z
Z = zeros(n,p);
for i=1:d
    Z = Z + alpha(i)*E(:,:,i);
end
```

If Z is a critical point, then $\partial\tilde{g}(Z)/\partial\alpha_i = 0$ for $i = 1, \ldots, d$. Since the E_i span V, (8) shows that this requirement is equivalent to (7). □

A matrix expression for $Z^{(\mathrm{cp})}$ if $V = V_3$ is given later in Proposition 3. If $V = V_1$ or $V = V_2$, however, a simple matrix expression does not exist. In order for the algorithms in this paper to be immediately implementable, the Matlab code for solving (7) for $Z^{(\mathrm{cp})}$ when $V = V_1$ is given in Table I. (It works by forming a basis E_i as in the proof of Proposition 2 and then finding α_i to make (8) zero.) If $V = V_2$, then the following modifications must be made to Table I. The dimension `d` defined on line 2 should be changed to $\mathtt{d} = \mathtt{p} * (2 * \mathtt{n} - \mathtt{p} - 1)$, and lines 5 to 7 (the `for` loop labeled "Diagonal elements of `A`") should be omitted.

Proposition 3: Define $\tilde{g}: V_3 \to \mathbb{R}$ as in (6). A critical point of $\tilde{g}(Z)$ occurs when $Z = X_\perp B^{(\mathrm{cp})}$, where $B^{(\mathrm{cp})} \in \mathbb{C}^{(n-p)\times p}$ satisfies

$$\begin{bmatrix} \Re\{G_1+G_2\} & -\Im\{G_1+G_2\} \\ \Im\{G_1-G_2\} & \Re\{G_1-G_2\} \end{bmatrix} \begin{bmatrix} \Re\,\mathrm{vec}\{B^{(\mathrm{cp})}\} \\ \Im\,\mathrm{vec}\{B^{(\mathrm{cp})}\} \end{bmatrix} = -\begin{bmatrix} \Re\,\mathrm{vec}\{X_\perp^H D\} \\ \Im\,\mathrm{vec}\{X_\perp^H D\} \end{bmatrix} \quad (9)$$

and where $G_1 = (I_p \otimes X_\perp)^H H (I_p \otimes X_\perp)$ and $G_2 = (I_p \otimes X_\perp)^T C (I_p \otimes X_\perp)$. If $C = 0$, then (9) simplifies to

$$[(I_p \otimes X_\perp)^H H (I_p \otimes X_\perp)]\,\mathrm{vec}\{B^{(\mathrm{cp})}\} = -\mathrm{vec}\{X_\perp^H D\}. \quad (10)$$

Proof: Since Z and $Z^{(\mathrm{cp})}$ are restricted to lie in V_3, they must be expressible as $Z = X_\perp B$ and $Z^{(\mathrm{cp})} = X_\perp B^{(\mathrm{cp})}$. Thus, (7) implies that $B^{(\mathrm{cp})}$ must satisfy

$$\begin{aligned} 0 &= \Re\{\mathrm{tr}\{(X_\perp B)^H D\} + \left[\mathrm{vec}\{X_\perp B\}^H H + \mathrm{vec}\{X_\perp B\}^T C\right] \\ &\qquad \cdot \mathrm{vec}\{X_\perp B^{(\mathrm{cp})}\} \quad (11) \\ &= \Re\Big\{\mathrm{vec}\{B\}^H\,\mathrm{vec}\{X_\perp^H D\} \\ &\qquad + \left[\mathrm{vec}\{B\}^H (I_p \otimes X_\perp)^H H + \mathrm{vec}\{B\}^T (I_p \otimes X_\perp)^T C\right] \\ &\qquad \cdot (I_p \otimes X_\perp)\mathrm{vec}\{B^{(\mathrm{cp})}\}\Big\} \quad (12) \end{aligned}$$

for all matrices B. Splitting all terms into their real and imaginary parts proves that $B^{(\mathrm{cp})}$ must be given by (9). If $C = 0$, then it is readily seen that (9) can be written in the complex-valued form (10). □

IV. Complex Stiefel Manifold

This section derives a number of fundamental properties of the complex Stiefel manifold. Although the complex Stiefel manifold has been studied from a number of perspectives in the literature [10], [21], the author has been unable to find explicit statements of this section's results elsewhere.

Definition 4 (Stiefel Manifold): The complex Stiefel manifold $St(n, p)$ is the set

$$St(n, p) = \{X \in \mathbb{C}^{n\times p} : X^H X = I\}. \tag{13}$$

The complex Stiefel manifold embeds naturally in $\mathbb{C}^{n\times p}$. It inherits the usual topology on $\mathbb{C}^{n\times p}$, and in particular, it is a compact manifold.

The projection of an arbitrary matrix X onto the Stiefel manifold is defined to be the point on the Stiefel manifold closest to X in the Euclidean norm. There is no unique solution if X does not have full column rank. Proposition 7, which is shown later, proves the converse; there is a unique solution if X has full column rank.

Definition 5 (Projection): Let $X \in \mathbb{C}^{n\times p}$ be a rank p matrix. The projection operator $\pi : \mathbb{C}^{n\times p} \to St(n, p)$ onto the Stiefel manifold $St(n, p)$ is defined to be

$$\pi(X) = \arg \min_{Q\in St(n,p)} \|X - Q\|^2. \tag{14}$$

The following useful lemma follows immediately from the fact that $\|UXV\| = \|X\|$ if U and V are unitary.

Lemma 6: If $U \in \mathbb{C}^{n\times n}$ and $V \in \mathbb{C}^{p\times p}$ are unitary matrices, then $\pi(UXV) = U\pi(X)V$.

Proposition 7: Let $X \in \mathbb{C}^{n\times p}$ be a rank p matrix. Then, $\pi(X)$ is well defined. Moreover, if the SVD of X is $X = U\Sigma V^H$, then $\pi(X) = UI_{n,p}V^H$.

Proof: It is shown that the unique solution of $\arg\min_{Q\in St(n,p)} \|\Sigma - Q\|^2$ is $Q = I_{n,p}$, which, when combined with Lemma 6, proves that (14) has the unique solution $Q = UI_{n,p}V^H$. Since $\|\Sigma - Q\|^2 = \mathrm{tr}\{(\Sigma - Q)^H(\Sigma - Q)\} = p + \mathrm{tr}\{\Sigma^T\Sigma\} - 2\Re\mathrm{tr}\{\Sigma^T Q\}$, it is sufficient to show that $\Re\mathrm{tr}\{\Sigma^T Q\} \le \Re\mathrm{tr}\{\Sigma^T I_{n,p}\}$ with equality if and only if $Q = I_{n,p}$. The inequality holds because $\Sigma_{ii} > 0$ and $\Re Q_{ii} \le |Q_{ii}| \le 1$, where the latter is implied by $Q^H Q = I$. Moreover, $\Re Q_{ii} = 1$ for $i = 1, \cdots, p$ if and only if $Q = I_{n,p}$, proving the only if part. □

Associated with each point of a manifold is a vector space called the tangent space. The tangent space of an abstract manifold is only unique up to isomorphism. For example, two different representations of the tangent space about each point of the real Stiefel manifold appear in [14] and [8]. The representation in [14] comes from the Lie group structure, whereas the representation in [8] comes from the embedding of the real Stiefel manifold in $\mathbb{R}^{n\times p}$. This paper chooses to use a representation of the tangent space that comes from the projection operator π. This choice fits in naturally with the optimization scheme proposed in the next section.

Let $X \in St(n, p)$, and consider the perturbed point $\pi(X + \epsilon Z) \in St(n, p)$ for some matrix $Z \in \mathbb{C}^{n\times p}$ and scalar $\epsilon \in \mathbb{R}$. (For $|\epsilon|$ sufficiently small, $X + \epsilon Z$ has full rank, and hence, $\pi(X + \epsilon Z)$ is well defined.) Since $St(n, p)$ does not fill up the whole $\mathbb{C}^{n\times p}$ space, certain directions Z do not cause $\pi(X+\epsilon Z)$ to move away from X as ϵ increases. The collection of directions Z such that $\pi(X+\epsilon Z) = X+O(\epsilon^2)$ is called the normal space at X of $St(n, p)$. The tangent space is defined to be the orthogonal complement of the normal space. (Orthogonality is with respect to the Euclidean inner product because this inner product induces the Euclidean norm; see Section II.) The tangent and normal spaces are determined as follows.

Remark: It can be shown that the above nonstandard definition of a tangent space meets all the criteria required of a tangent space in differential geometry. Moreover, the above definition leads to a concrete representation of the tangent space, which is the most suitable one for this paper.

Lemma 8: Let $X \in St(n, p)$, and choose any $X_\perp \in \mathbb{C}^{n\times(n-p)}$ satisfying $[X\, X_\perp]^H[X\, X_\perp] = I$. An arbitrary matrix $Z \in \mathbb{C}^{n\times p}$ is uniquely decomposable as $Z = XA + X_\perp B + XC$, where $A \in \mathbb{C}^{p\times p}$ is skew-Hermitian, $B \in \mathbb{C}^{(n-p)\times p}$ is arbitrary, and $C \in \mathbb{C}^{p\times p}$ is Hermitian. Furthermore

$$\pi(X + tZ) = X + t(XA + X_\perp B) + O(t^2). \tag{15}$$

Proof: That $XA + X_\perp B + XC$ is a unique decomposition of Z is clear. Define $X(t) = \pi(X + tZ)$, and let D be its derivative at $t = 0$ so that $X(t) = X + tD + O(t^2)$. Since $X(t)^H X(t) = I + t(X^H D + D^H X) + O(t^2) = I$, D must satisfy $X^H D + D^H X = 0$. This constraint is most easily enforcible by expressing D as $D = X\tilde{A} + X_\perp\tilde{B} + X\tilde{C}$, where $\tilde{A} \in \mathbb{C}^{p\times p}$ is skew-Hermitian, $\tilde{B} \in \mathbb{C}^{(n-p)\times p}$ is arbitrary, and $\tilde{C} \in \mathbb{C}^{p\times p}$ is Hermitian. Then, $X^H D + D^H X = 0$ is equivalent to $\tilde{C} = 0$. By definition, $X(t)$ is the closest point on the Stiefel manifold to $X + tZ$. Thus, $D = X\tilde{A} + X_\perp\tilde{B}$ must minimize $\|X + tZ - X - tD\|^2 = t^2\|X(A - \tilde{A}) + X_\perp(B - \tilde{B}) + XC\|^2$ for sufficiently small t. The minimum occurs when $\tilde{A} = A$ and $\tilde{B} = B$ (which is a consequence of Lemma 10 shown later), completing the proof. □

Lemma 8 shows that $\pi(X + \epsilon XC) = X + O(\epsilon^2)$, and in fact, it can be shown that $\pi(X + \epsilon XC) = X$ for $|\epsilon| > 0$ sufficiently small. This leads to the following definition.

Definition 9 (Normal Space): The normal space $N_X(n, p)$ at $X \in St(n, p)$ of the Stiefel manifold $St(n, p)$ is

$$N_X(n, p) = \left\{N \in \mathbb{C}^{n\times p} : N = XC,\, C \in \mathbb{C}^{p\times p},\, C = C^H\right\}. \tag{16}$$

Lemma 8 suggests that the tangent space consists of directions of the form $XA + X_\perp B$. The following lemma confirms that these directions are indeed orthogonal to the normal space.

Lemma 10: Let $X \in St(n, p)$ and $X_\perp \in \mathbb{C}^{n\times(n-p)}$ satisfy $[X\, X_\perp]^H[X\, X_\perp] = I$. Then, for any skew-Hermitian $A \in \mathbb{C}^{p\times p}$, arbitrary $B \in \mathbb{C}^{(n-p)\times p}$, and Hermitian $C \in \mathbb{C}^{p\times p}$, $\Re\mathrm{tr}\{(XA + X_\perp B)^H XC\} = 0$. That is, $XA + X_\perp B$ is orthogonal to XC, and furthermore, $\|XA + X_\perp B + XC\|^2 = \|XA + X_\perp B\|^2 + \|XC\|^2$.

Proof: Since A is skew-Hermitian and C is Hermitian, $\Re\mathrm{tr}\{(XA + X_\perp B)^H XC\} = \Re\mathrm{tr}\{A^H C\} = 0$. Furthermore, $\|XA + X_\perp B + XC\|^2 = \mathrm{tr}\{(XA + X_\perp B)^H(XA + X_\perp B) + (XC)^H(XC)\} + 2\Re\mathrm{tr}\{(XA + X_\perp B)^H XC\} = \|XA + X_\perp B\|^2 + \|XC\|^2$. □

Definition 11 (Tangent Space): The tangent space $T_X(n, p)$ at $X \in St(n, p)$ of the Stiefel manifold $St(n, p)$ is

$$T_X(n, p) = \Big\{ Z \in \mathbb{C}^{n\times p} \colon Z = XA + X_\perp B,\ A \in \mathbb{C}^{p\times p} \\ A + A^H = 0,\ B \in \mathbb{C}^{(n-p)\times p} \Big\}. \quad (17)$$

It is readily verified that $T_X(n, p)$ is a real vector space. Its dimension, when considered as a vector space over $\mathbb{R}$, is computed as follows. Since B has $(n-p)p$ complex-valued elements, it contributes an amount $2(n-p)p$ to the overall dimension of $T_X(n, p)$. Because A is skew-Hermitian, it is completely specified once its $1 + \cdots + (p-1)$ complex-valued upper diagonal elements are given as well as its p diagonal elements, which must be purely imaginary. Thus, the overall dimension is

$$2(n-p)p + 2(1 + \cdots + (p-1)) + p = p(2n-p). \quad (18)$$

The second-order approximation of the projection from the tangent space to the Stiefel manifold is required in the next section.

Proposition 12: If $X \in St(n, p)$ and $Z \in T_X(n, p)$, then $\pi(X + tZ) = X + tZ - (1/2)t^2 XZ^H Z + O(t^3)$.

Proof: Lemma 8 proves that the first-order term is tZ. Define $X(t) = \pi(X + tZ)$, and let H be such that $X(t) = X + tZ + t^2 H + O(t^3)$. Since $X(t)^H X(t) = I$, direct expansion shows that H must satisfy $H^H X + X^H H = -Z^H Z$. Subject to this constraint, H must also minimize $||X + tZ - X - tZ - t^2 H||^2 = t^4 ||H||^2$ for sufficiently small t. Applying the Lagrange multiplier technique proves that $H = -(1/2)XZ^H Z$. □

V. Optimization on the Complex Stiefel Manifold

This section derives two algorithms (one is a modified steepest descent method, and the other is a modified Newton method) for minimizing $f(X)$ subject to $X^H X = I$, where $X \in \mathbb{C}^{n\times p}$, and $f(X)$ is real valued. Note that if the cost function $f(X)$ is such that $f(XQ) = f(X)$ for any unitary matrix $Q \in \mathbb{C}^{p\times p}$, then the algorithms in Section VII should be used instead.

The main principle behind optimization on manifolds is to rewrite the optimization problem in terms of a local parameterization at each iteration. A local parameterization about the point $X \in St(n, p)$ is a mapping $h\colon \Omega \to St(n, p)$ from an open subset Ω of the vector space $T_X(n, p)$ to the Stiefel manifold with the property that any point $Y \in St(n, p)$ sufficiently close to X can be uniquely written as $Y = h(Z)$ with $Z \in \Omega$. There are an infinite number of local parameterizations to choose from. For instance, the exponential map (which is only defined after the Stiefel manifold is endowed with a Riemannian structure) is used in [8] as a local parameterization. The present paper proposes to use the local parameterization $h\colon T_X(n, p) \to St(n, p)$ defined by $h(Z) = \pi(X + Z)$, where π is the projection operator (see Definition 5). Later, lemma 13 will prove that h is well defined. This local parameterization is not only simpler than the one in [8], but simulations in [18] demonstrate that it leads to faster convergence in certain applications.

Lemma 13: For arbitrary $X \in St(n, p)$ and $Z \in T_X(n, p)$, the matrix $X + Z$ has full rank. In particular, the local parameterization $h(Z) = \pi(X + Z)$ is well defined.

Proof: As in Definition 11, write $Z = XA + X_\perp B$. If $X + Z$ did not have full rank, then there would exist a nonzero vector z such that $(X + XA + X_\perp B)z = 0$. Premultiplying by X^H shows that this implies $Az = -z$. However, this is not possible since A, being skew-Hermitian, has purely imaginary eigenvalues. That $h(Z)$ is well-defined now follows from Proposition 7. □

Given a local parameterization h, the local cost function is defined to be $f \circ h$, which is the composition of f and h. For instance, if $h(Z) = \pi(X + Z)$, then the local cost function $g\colon T_X(n, p) \to \mathbb{R}$ is

$$g(Z) = f(\pi(X + Z)). \quad (19)$$

Whereas the cost function f is defined on a $2np$-dimensional vector space ($\mathbb{C}^{n\times p}$-space has $2np$ dimensions when considered as a vector space over $\mathbb{R}$), the local cost function g is defined on a $p(2n-p)$-dimensional vector space [cf. (18)]. This reduction in dimension is especially significant when p is large.

The general framework for optimization on manifolds is as follows. Given a point $X \in St(n, p)$, choose $Z \in T_X(n, p)$ so that $f(h(Z)) = g(Z) < g(0) = f(X)$. Move to the new point $X := h(Z)$, and repeat until convergence. Sections V-A and B propose two ways of choosing Z at each step, leading to two algorithms for solving orthogonally constrained optimization problems.

A. Modified Steepest Descent on the Complex Stiefel Manifold

This section derives an algorithm for minimizing $f(X)$ subject to $X^H X = I$. It requires the evaluation of $f(X)$ and its first derivative D_X at each step.

For a given $X \in St(n, p)$, let $g\colon T_X(n, p) \to \mathbb{R}$ be the local cost function $g(Z) = f(\pi(X + Z))$. Since $T_X(n, p)$ is a vector space, the well-known steepest descent algorithm (see, for instance, [22]) can be used to find a Z, which locally minimizes $g(Z)$. However, since the range of $\pi(X + Z)$ covers only part of the Stiefel manifold, it is more sensible to perform just a single descent step using the local cost function $g(Z)$.

Performing a descent step requires the computation of the gradient of $g(Z)$. The gradient is only defined once $T_X(n, p)$ is given an inner product. Ideally, the inner product should be chosen to make the level sets of $g(Z)$ approximately spherical [22]. However, since $g(Z)$ is not known in advance, it is necessary to make an arbitrary choice for the inner product. Just as the Euclidean inner product is customarily used for optimization on $\mathbb{R}^n$, the inner product

$$\langle Z_1, Z_2 \rangle = \Re \operatorname{tr} \left\{ Z_2^H \left(I - \tfrac{1}{2} XX^H \right) Z_1 \right\} \\ Z_1, Z_2 \in T_X(n, p), \quad X \in St(n, p) \quad (20)$$

is commonly used on $T_X(n, p)$ [see [8] for the derivation of (20) in the real-valued case].

The reason why (20) is a natural choice for an inner product is because it has the following geometrically pleasing interpretation. Express $Z \in T_X(n, p)$ as $Z = XA + X_\perp B$, and let E_{ij} be the matrix whose elements are all zero except for the ijth element, which is one. Then, under (20), the "elementary" tangent directions $X_\perp E_{ij}$ and $X(E_{ij} - E_{ji})$ for appropriate values of i and j are mutually orthogonal and have unit norm [the norm of a tangent direction $Z \in T_X(n, p)$ is $\sqrt{\langle Z, Z \rangle}$]. This is a

desirable property because the perturbation $\pi(X+\epsilon X_\perp E_{ij})$ is obtained by rotating the jth column of X in the direction of the ith column of $X_\perp$ by an angle of $\epsilon+O(\epsilon^2)$, whereas the perturbation $\pi(X+\epsilon X(E_{ij}-E_{ji}))$ is obtained by rotating both the ith and jth columns of X by an angle of $\epsilon+O(\epsilon^2)$.

The gradient $G \in T_X(n, p)$ of the local cost function $g(Z)$ at the origin is, by definition, the unique G in $T_X(n, p)$ such that

$$g(Z) = f(X) + \langle G, Z\rangle + O(\|Z\|^2) \tag{21}$$

holds for all $Z \in T_X(n, p)$. The steepest descent direction is the negative of the gradient. It is expressible in terms of the first derivative of $f(X)$ as follows.

Theorem 14 (Steepest Descent): Given the cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, let $g(Z) = f(\pi(X+Z))$ be the local cost function about a given point $X \in St(n, p)$. The steepest descent direction of $g(Z)$ at the origin $Z=0$ under the canonical inner product (20) is

$$Z^{(\mathrm{sd})} = XD_X^HX - D_X \tag{22}$$

where $D_X \in \mathbb{C}^{n\times p}$ is any matrix such that

$$\begin{aligned} &\forall\, Z \in T_X(n, p),\\ &f(X+Z) = f(X) + \Re\,\mathrm{tr}\{Z^HD_X\} + O(\|Z\|^2). \end{aligned} \tag{23}$$

Proof: It follows from Lemma 8 and (23) that

$$g(Z) = f(X) + \Re\,\mathrm{tr}\{Z^HD_X\} + O(\|Z\|^2). \tag{24}$$

The following calculation shows that $G = D_X - XD_X^HX$ makes (21) and (24) equivalent

$$\begin{aligned} \langle G, Z\rangle &= \Re\,\mathrm{tr}\{Z^H\left(I - \tfrac{1}{2}XX^H\right)(D_X - XD_X^HX)\} && (25)\\ &= \Re\,\mathrm{tr}\{Z^HD_X\}\\ &\quad - \tfrac{1}{2}\Re\,\mathrm{tr}\{Z^HX(D_X^HX + X^HD_X)\} && (26)\\ &= \Re\,\mathrm{tr}\{Z^HD_X\} && (27) \end{aligned}$$

where the last equality follows from Lemma 10 and the fact that $X(D_X^HX+X^HD_X) \in N_X(n, p)$. Finally, it can be shown that X^HG is skew-Hermitian, verifying that G is an element of the tangent space $T_X(n, p)$ and completing the proof. □

Remark: Although one choice of D_X in (23) is the derivative of $f(X)$ (see Section II), other choices are possible since (23) only requires D_X to be the derivative of $f(X)$ in the tangent directions $Z \in T_X(n, p)$. This fact sometimes can be exploited to simplify the computation of D_X.

Once the steepest descent direction $Z^{(\mathrm{sd})}$, which is defined in (22), has been calculated, it is necessary to choose a positive step size $\gamma \in \mathbb{R}$ so that $g(\gamma Z^{(\mathrm{sd})}) < g(0)$. The Armijo step size rule [22] states that γ should be chosen to satisfy the inequalities

$$g(0) - g\left(\gamma Z^{(\mathrm{sd})}\right) \geq \tfrac{1}{2}\gamma\left\langle Z^{(\mathrm{sd})}, Z^{(\mathrm{sd})}\right\rangle \tag{28}$$

$$g(0) - g\left(2\gamma Z^{(\mathrm{sd})}\right) < \gamma\left\langle Z^{(\mathrm{sd})}, Z^{(\mathrm{sd})}\right\rangle. \tag{29}$$

Rule (28) ensures that the step $\gamma Z^{(\mathrm{sd})}$ will "significantly" decrease the cost, whereas (29) ensures that the step $2\gamma Z^{(\mathrm{sd})}$ would not be a better choice. A straightforward method for finding a suitable γ is to keep on doubling γ until (29) no longer holds and then halving γ until it satisfies (28). [From (21), it is readily seen that such a γ can always be found.]

Consolidating the above ideas yields the following algorithm. The algorithm almost always converges to a local minimum, the exception being if it lands directly on a saddle point first. Indeed, it is proved in [22] that the Armijo step size rule ensures that $f(X)$ decreases to a critical point, provided f is differentiable and that the level sets of $f(X)$ are bounded. The latter criteria is always true here because the Stiefel manifold $St(n, p)$ is compact.

Algorithm 15 (Modified Steepest Descent on Stiefel Manifold): Given a cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, the following algorithm almost always converges to a local minimum of $f(X)$ subject to the constraint that $X^HX = I$. It requires that a matrix $D_X \in \mathbb{C}^{n\times p}$ satisfying (23) can be computed for any $X \in St(n, p)$.

1) Choose $X \in \mathbb{C}^{n\times p}$ such that $X^HX = I$. Set step size $\gamma := 1$.
2) Compute D_X, which is the derivative of f at X [cf. (23)].
3) Compute the descent direction $Z := XD_X^HX - D_X$.
4) Evaluate $\langle Z, Z\rangle = \mathrm{tr}\{Z^H(I - (1/2)XX^H)Z\}$. If $\sqrt{\langle Z, Z\rangle}$ is sufficiently small, then stop.
5) If $f(X) - f(\pi(X+2\gamma Z)) \geq \gamma\langle Z, Z\rangle$, then set $\gamma := 2\gamma$, and repeat Step 5. [The projection $\pi(\cdot)$ can be evaluated using the SVD; see Proposition 7.]
6) If $f(X) - f(\pi(X+\gamma Z)) < (1/2)\gamma\langle Z, Z\rangle$, then set $\gamma := (1/2)\gamma$, and repeat Step 6.
7) Set $X := \pi(X+\gamma Z)$. Go to Step 2.

B. Modified Newton Method on the Complex Stiefel Manifold

This section derives an alternative algorithm for minimizing $f(X)$ subject to $X^HX = I$. It requires the evaluation of $f(X)$ and its first two derivatives at each step. Unlike Algorithm 15, the algorithm in this section only converges to a local minimum if it is initialized to a point sufficiently close to the local minimum. However, when it does converge, it achieves[2] a quadratic rate of convergence. By comparison, Algorithm 15 exhibits only a linear rate of convergence.

As in Section V-A, let $g: T_X(n, p) \to \mathbb{R}$ be the local cost function $g(Z) = f(\pi(X+Z))$ about the point $X \in St(n, p)$. This section proposes to take a Newton step at each iteration based on a quadratic approximation of $g(Z)$ at the origin. Unlike in Section V-A, it is not necessary to give $T_X(n, p)$ an inner product.

The following proposition derives the second-order Taylor series approximation of the local cost function $g(Z)$ at the origin.

Proposition 16: Given the cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, let $g(Z) = f(\pi(X+Z))$ be the local cost function about a given point $X \in St(n, p)$. Then, for any $Z \in T_X(n, p)$

$$\begin{aligned} g(Z) = f(X) &+ \Re\,\mathrm{tr}\{Z^HD_X\} + \tfrac{1}{2}\,\mathrm{vec}\{Z\}^H\\ &\cdot\left[H_X - \tfrac{1}{2}((X^HD_X + D_X^HX)^T \otimes I_n)\right]\mathrm{vec}\{Z\}\\ &+ \tfrac{1}{2}\Re\{\mathrm{vec}\{Z\}^TC_X\,\mathrm{vec}\{Z\}\} + O(\|Z\|^3) \end{aligned} \tag{30}$$

where $D_X \in \mathbb{C}^{n\times p}$ and H_X, $C_X \in \mathbb{C}^{np\times np}$ are the derivative and Hessian of f at X, respectively (cf. Section II).

[2]Although this fact is not proved here, the convergence proofs in [14] and [16] can be adapted to the algorithms in this paper.

Proof: Using Proposition 12 and (1) gives

$$g(tZ) = f\left(X + t\left(Z - \frac{t}{2}XZ^HZ\right)\right) + O(t^3) \tag{31}$$

$$= f(X) + t\Re\mathrm{tr}\left\{\left(Z - \frac{t}{2}XZ^HZ\right)^H D_X\right\} + \frac{t^2}{2}\,\mathrm{vec}\left\{Z - \frac{t}{2}XZ^HZ\right\}^H H_X \mathrm{vec}\left\{Z - \frac{t}{2}XZ^HZ\right\} + \frac{t^2}{2}\,\Re\left\{\mathrm{vec}\left\{Z - \frac{t}{2}XZ^HZ\right\}^T \cdot C_X\,\mathrm{vec}\left\{Z - \frac{t}{2}XZ^HZ\right\}\right\} + O(t^3) \tag{32}$$

$$= f(X) + t\Re\mathrm{tr}\{Z^HD_X\} + \frac{t^2}{2}[\mathrm{vec}\{Z\}^H H_X\,\mathrm{vec}\{Z\} - \Re\mathrm{tr}\{Z^HZX^HD_X\}] + \frac{t^2}{2}\,\Re\{\mathrm{vec}\{Z\}^T C_X\,\mathrm{vec}\{Z\}\} + O(t^3) \tag{33}$$

$$= f(X) + t\Re\mathrm{tr}\{Z^HD_X\} + \frac{t^2}{2}\left[\mathrm{vec}\{Z\}^H H_X\,\mathrm{vec}\{Z\} - \tfrac{1}{2}\mathrm{tr}\{Z^HZ(X^HD_X + D_X^HX)\}\right] + \frac{t^2}{2}\,\Re\{\mathrm{vec}\{Z\}^T C_X\,\mathrm{vec}\{Z\}\} + O(t^3). \tag{34}$$

The result now follows from the fact that $\mathrm{tr}\{Z^HZA\} = \mathrm{vec}\{Z\}^H(A^T \otimes I_n)\mathrm{vec}\{Z\}$ for any matrix $A \in \mathbb{C}^{p\times p}$. □

The Newton step is defined to be the value of Z confined to the tangent space $T_X(n, p)$ at which the quadratic approximation in (30) has its critical point. The location of the critical point can be found by applying the theory in Section III; observe that V_1 in Section III is the vector space $T_X(n, p)$.

If the Hessian of f is singular, then the critical point of the quadratic approximation of the local cost function is not unique. It is therefore important to ensure that the cost function f does not possess symmetries. The two most common symmetries that f may possess are $f(XQ) = f(X)$ for Q unitary and $f(X\Theta) = f(X)$, where $\Theta = \mathrm{diag}\{e^{j\theta_1}, \ldots, e^{j\theta_p}\}$ is a diagonal unitary matrix. In the former case, the theory in Section VII-B must be used. In the latter case, it suffices to restrict Z to lie in V_2, which is the vector space defined in Section III. This necessitates a minor modification of the following algorithm; see Remark 1 later.

Algorithm 17 (Modified Newton Method on Stiefel Manifold): Given a twice-differentiable cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, the following algorithm attempts to converge to a local minimum of $f(X)$ subject to the constraint that $X^HX = I$. It requires that the derivative $D_X \in \mathbb{C}^{n\times p}$ and the Hessian $H_X, C_X \in \mathbb{C}^{np\times np}$ of f at the point X can be computed for any $X \in St(n, p)$.

1) Choose $X \in \mathbb{C}^{n\times p}$ and $X_\perp \in \mathbb{C}^{n\times(n-p)}$ such that $[X\,X_\perp]^H[X\,X_\perp] = I$.
2) Compute D_X, which is the derivative of f at X, and H_X, C_X, which is the Hessian of f at X, as defined in Section II. If $\sqrt{\mathrm{tr}\{D_X^HD_X - X^HD_XX^HD_X\}}$ is sufficiently small, then stop.
3) Compute the Newton step $Z := \mathtt{cpoint}(X, X_\perp, D_X, H_X - (1/2)[(X^HD_X + D_X^HX)^T \otimes I_n], C_X)$, where `cpoint` is defined in Table I. (See Remark 1 later.)
4) If $f(X) \le f(\pi(X+Z))$, then abort. [The projection $\pi(\cdot)$ can be evaluated using the SVD; see Proposition 7.]
5) Use the SVD to compute U, Σ and V such that $X + Z = U\Sigma V^H$. Set $X := UI_{n,p}V^H$, and set $X_\perp$ to the last $n-p$ columns of U. Go to Step 2.

Remarks:

1) If $f(X\Theta) = f(X)$ for any diagonal unitary matrix Θ, then the following modifications must be made to `cpoint` in Table I. The dimension `d` defined on line 2 should be changed to $\mathtt{d} = \mathtt{p} * (2 * \mathtt{n} - \mathtt{p} - 1)$, and lines 5 to 7 (the `for` loop labeled "Diagonal elements of `A`") should be omitted.
2) The quantity $\sqrt{\mathrm{tr}\{D_X^HD_X - X^HD_XX^HD_X\}}$ in Step 2 of Algorithm 17 is equal to $\sqrt{\langle Z, Z\rangle}$ in Step 4 of Algorithm 15, that is, it equals the norm of the gradient of f at X.
3) Step 5 of Algorithm 17 results in $X := \pi(X + Z)$ and an $X_\perp$ such that $[X\,X_\perp]^H[X\,X_\perp] = I$.

As is the case with all Newton-type algorithms, Algorithm 17 can fail to take a descent direction if the current point X is not sufficiently close to a minimum to ensure that the Hessian is positive definite. If Algorithm 17 fails to take a descent step, several iterations of Algorithm 15 can be used to move closer to a minimum before restarting Algorithm 17.

VI. Complex Grassmann Manifold

If the cost function $f(X)$ is such that $f(XQ) = f(X)$ for any unitary matrix Q, it should be minimized on the Grassmann manifold rather than on the Stiefel manifold. This is because the Grassmann manifold treats points X and XQ as being equivalent, leading to a further reduction in the dimension of the optimization problem.

This section derives a number of fundamental properties of the complex Grassmann manifold. The derivation of the results in this and the next section parallels that done in the last two sections for the complex Stiefel manifold. However, unlike the complex Stiefel manifold, the complex Grassmann manifold does not embed in $\mathbb{C}^{n\times p}$ space. Therefore, the derivations here are not identical to the derivations in previous sections.

Definition 18 (Complex Grassmann Manifold): The complex Grassmann manifold $Gr(n, p)$ is the set of all p-dimensional complex subspaces of $\mathbb{C}^n$.

There is a close connection between the Grassmann and Stiefel manifolds. If $X \in St(n, p)$ is a point on the Stiefel manifold, then its p columns form an orthonormal basis for a p-dimensional subspace. That is, if $\lfloor X \rfloor$ denotes the subspace spanned by the columns of X, then $X \in St(n, p)$ implies $\lfloor X \rfloor \in Gr(n, p)$. Conversely, every point in $Gr(n, p)$ is obtainable in this way.

For the purposes of this paper, the Grassmann manifold is best thought of as a quotient space of the Stiefel manifold. This is now explained. Define two points $X, Y \in St(n, p)$ on the Stiefel manifold to be equivalent and denoted $X \equiv Y$ if there

exists a unitary matrix $Q \in \mathbb{C}^{p\times p}$ such that $Y = XQ$. It is clear that $X \equiv Y$ if and only if $\lfloor X \rfloor = \lfloor Y \rfloor$. Therefore, there is a one-to-one correspondence between points on the Grassmann manifold $Gr(n, p)$ and equivalence classes of $St(n, p)$.

Since the Grassmann manifold does not embed in $\mathbb{C}^{n\times p}$ space, the projection of an arbitrary matrix X onto the Grassmann manifold is defined in terms of the projection onto the Stiefel manifold. There is no unique solution if X does not have full column rank. Later, Proposition 20 will prove the converse; there is a unique solution if X has full column rank.

Definition 19 (Projection): Let $X \in \mathbb{C}^{n\times p}$ be a rank p matrix. The projection operator $\pi: \mathbb{C}^{n\times p} \rightarrow Gr(n, p)$ onto the Grassmann manifold $Gr(n, p)$ is defined to be

$$\pi(X) = \left\lfloor \arg \min_{Q\in St(n,p)} ||X - Q||^2 \right\rfloor. \tag{35}$$

Whereas π was used in earlier sections to denote projection onto the Stiefel manifold, henceforth, π refers to the projection operator onto the Grassmann manifold, unless otherwise stated.

Proposition 20: Let $X \in \mathbb{C}^{n\times p}$ be a rank p matrix. Then, $\pi(X) = \lfloor X \rfloor$. Moreover, if the SVD of X is $X = U\Sigma V^H$, then $\pi(X) = \lfloor UI_{n,p} \rfloor$, and if the QR decomposition of X is $X = QR$, then $\pi(X) = \lfloor QI_{n,p} \rfloor$.

Proof: Proposition 7 implies that $\pi(X)$ is well defined and equals $\lfloor UI_{n,p}V^H \rfloor$. Moreover, $\lfloor UI_{n,p}V^H \rfloor = \lfloor UI_{n,p} \rfloor$. Since X has rank p, $\lfloor UI_{n,p} \rfloor = \lfloor X \rfloor$. Finally, $\lfloor QI_{n,p} \rfloor = \lfloor X \rfloor$ follows immediately from [9, Th. 5.2.1]. □

Remark: The usefulness of expressing the projection in terms of the SVD or QR decomposition is that both $UI_{n,p}$ and $QI_{n,p}$ are elements of $St(n, p)$.

Since the Grassmann manifold is a quotient space of the Stiefel manifold, its tangent space is a subspace of the Stiefel manifold's tangent space. This fact can also be seen from the nonstandard definition of a tangent space in terms of the projection operator given in Section IV; certain elements Z in the tangent space of the Stiefel manifold will no longer cause $\pi(X + \epsilon Z)$ to move away from X as ϵ increases. Specifically, for a given $X \in St(n, p)$, let $Z = XA + X_\perp B$ be an element of the tangent space of $St(n, p)$. Applying Lemma 8 shows that

$$\begin{aligned}\pi(X + t(XA + X_\perp B)) &= \lfloor X + t(XA + X_\perp B) + O(t^2) \rfloor \\ &= \lfloor X + tX_\perp B + O(t^2) \rfloor \end{aligned} \tag{36}$$

where the last equality follows from the fact that $\lfloor X(I + tA) \rfloor = \lfloor X \rfloor$, provided $I + tA$ is invertible. Thus, the subspace generated by XA is not part of the tangent space of $Gr(n, p)$. Conversely, $\lfloor X + X_\perp B \rfloor = \lfloor X \rfloor$ implies $B = 0$, proving that the tangent space of $Gr(n, p)$ is the subspace generated by matrices of the form $X_\perp B$.

Definition 21 (Tangent Space): Let $X \in St(n, p)$. Then, the tangent space $T_{\lfloor X \rfloor}(n, p)$ at $\lfloor X \rfloor \in Gr(n, p)$ of the Grassmann manifold $Gr(n, p)$ is

$$T_{\lfloor X \rfloor}(n, p) = \{Z \in \mathbb{C}^{n\times p}: Z = X_\perp B,\ B \in \mathbb{C}^{(n-p)\times p}\}. \tag{37}$$

The dimension of $T_{\lfloor X \rfloor}(n, p)$, which is considered to be a vector space over $\mathbb{R}$, is $2(n - p)p$ [cf. (18)].

The second-order approximation of the projection from the tangent space to the Grassmann manifold is required in the next section.

Proposition 22: Let $X \in St(n, p)$. If $Z \in T_{\lfloor X \rfloor}(n, p)$ is an element of the tangent space at $\lfloor X \rfloor \in Gr(n, p)$ of the Grassmann manifold $Gr(n, p)$, then $\pi(X+tZ) = \lfloor X+tZ+O(t^3) \rfloor$.

Proof: From Proposition 12

$$\begin{aligned}\pi(X + tZ) &= \left\lfloor X + tZ - \tfrac{1}{2}t^2XZ^HZ + O(t^3) \right\rfloor && (38)\\ &= \left\lfloor X\left(I - \tfrac{1}{2}t^2Z^HZ\right) + tZ + O(t^3) \right\rfloor && (39)\\ &= \left\lfloor X + tZ\left(I - \tfrac{1}{2}t^2Z^HZ\right) + O(t^3) \right\rfloor && (40)\\ &= \lfloor X + tZ + O(t^3) \rfloor. && (41)\end{aligned}$$

□

VII. Optimization on the Complex Grassmann Manifold

Whereas Section V derived algorithms for minimizing a general cost function $f(X)$ subject to $X^HX = I$, this section derives specialized algorithms for the case when $f(X)$ satisfies either one or both of the following assumptions.

- **A1)** $f(XQ) = f(X)$ for all $X \in St(n, p)$ and unitary $Q \in \mathbb{C}^{p\times p}$.
- **A2)** $f(XS) = f(X)$ for all $X \in St(n, p)$ and invertible $S \in \mathbb{C}^{p\times p}$.

Such symmetries occur in subspace tracking [5], [23], [25] as well as in blind identification [1], [30]. In the former case, it is because a subspace is invariant to unitary transformations (that is, $\lfloor XQ \rfloor = \lfloor X \rfloor$), whereas in the latter case, it is caused by the inability of second-order statistics to discriminate between unitary transformations of certain parameters of interest.

Whereas Section V considered f as a function on the Stiefel manifold $St(n, p)$, this section considers f as a function on the Grassmann manifold. Specifically, since f satisfies A1), there exists a function $\tilde{f}: Gr(n, p) \rightarrow \mathbb{R}$ such that

$$\forall X \in St(n, p), \qquad \tilde{f}(\lfloor X \rfloor) = f(X). \tag{42}$$

The local cost function $g: T_{\lfloor X \rfloor}(n, p) \rightarrow \mathbb{R}$ about the point $\lfloor X \rfloor \in Gr(n, p)$, where $X \in St(n, p)$, is defined to be (cf. Section V)

$$g(Z) = \tilde{f}(\pi(X + Z)) \tag{43}$$

where π is the projection onto the Grassmann manifold defined in Definition 19. (It follows from Lemma 13 that $g(Z)$ is defined for all Z.) One advantage of using the Grassmann manifold rather than the Stiefel manifold is that $T_{\lfloor X \rfloor}(n, p)$ has only $p(2n - 2p)$ dimensions, whereas $T_X(n, p)$ in Section V has $p(2n - p)$ dimensions.

From a numerical point of view, it is not practical to work explicitly with the cost function $\tilde{f}$. Instead, the local cost function (43) can be rewritten in the alternative form

$$g(Z) = f\big(\mathrm{qf}_p\{X + Z\}\big) \tag{44}$$

where $\mathrm{qf}_p\{\cdot\}$ is the "Q-Factor" operator defined as follows. If $X = QR$ is the QR decomposition of the matrix $X \in \mathbb{C}^{n\times p}$, then $\mathrm{qf}_p\{X\}$ is defined to be the first p columns of Q. That (44) is equivalent to (43) follows immediately from Proposition 20.

Using the same framework as in Section V, Section VII-A derives a modified steepest descent algorithm, whereas Section VII-B derives a modified Newton algorithm.

A. Modified Steepest Descent on the Complex Grassmann Manifold

This section derives an algorithm for minimizing $f(X)$ subject to $X^HX = I$ when $f(X)$ satisfies either A1) or both A1) and A2) in Section VII. It requires the evaluation of $f(X)$ and its derivative D_X at each iteration.

The steepest descent direction of the local cost function (43) is only defined once $T_{\lfloor X \rfloor}(n, p)$ is given an inner product structure. Since $T_{\lfloor X \rfloor}(n, p) \subset T_X(n, p)$, the natural choice of an inner product on $T_{\lfloor X \rfloor}(n, p)$ is the one obtained by restricting the canonical inner product (20) on $T_X(n, p)$ to $T_{\lfloor X \rfloor}(n, p)$. In fact, when restricted to $T_{\lfloor X \rfloor}(n, p)$, (20) simplifies to the Euclidean inner product

$$\langle Z_1, Z_2 \rangle = \Re \operatorname{tr}\{Z_2^H Z_1\}, \quad Z_1, Z_2 \in T_{\lfloor X \rfloor}(n, p) \quad X \in St(n, p). \tag{45}$$

Theorem 23 (Steepest Descent): Given the cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, let $g(Z) = \tilde{f}(\pi(X + Z))$ be the local cost function about a given point $X \in St(n, p)$, where $\tilde{f}$ is defined in (42). The steepest descent direction of $g(Z)$ at the origin $Z = 0$ under the canonical inner product (45) is

$$Z^{(\mathrm{sd})} = -(I - XX^H)D_X \tag{46}$$

where $D_X \in \mathbb{C}^{n\times p}$ is any matrix such that

$$\forall\, Z \in T_{\lfloor X \rfloor}(n, p) \quad f(X + Z) = f(X) + \Re \operatorname{tr}\{Z^H D_X\} + O(\|Z\|^2). \tag{47}$$

Proof: It is clear that $g(Z) = f(\pi(X + Z))$, where π is the projection operator onto the Stiefel manifold (Definition 5). Thus, $g(Z)$ is given by (24). The following calculation shows that $G = (I - XX^H)D_X$ makes (21) and (24) equivalent for all $Z \in T_{\lfloor X \rfloor}(n, p)$

$$\langle G, Z \rangle = \Re \operatorname{tr}\{Z^H(I - XX^H)D_X\} \tag{48}$$
$$= \Re \operatorname{tr}\{Z^H D_X\} - \Re \operatorname{tr}\{Z^H XX^H D_X\} \tag{49}$$
$$= \Re \operatorname{tr}\{Z^H D_X\} \tag{50}$$

where the last equality follows from the fact that if $Z \in T_{\lfloor X \rfloor}(n, p)$, then $Z = X_\perp B$ and so $Z^HX = 0$. Finally, it is readily seen that $X^HG = 0$, verifying that G is an element of the tangent space $T_{\lfloor X \rfloor}(n, p)$ and completing the proof. □

Remark: Although one choice of D_X in (47) is the derivative of $f(X)$ (see Section II), other choices are possible since (47) only requires D_X to be the derivative of $f(X)$ in the tangent directions $Z \in T_{\lfloor X \rfloor}(n, p)$.

Combining Theorem 23 with the Armijo step-size rule described in Section V-A leads to the following analog of Algorithm 15.

Algorithm 24 (Modified Steepest Descent on Grassmann Manifold): Given a differentiable cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$ that satisfies (A1) in Section VII, the following algorithm almost always converges to a local minimum of $f(X)$ subject to the constraint that $X^HX = I$. It requires that a matrix $D_X \in \mathbb{C}^{n\times p}$ satisfying (47) can be computed for any $X \in St(n, p)$.

1) Choose $X \in \mathbb{C}^{n\times p}$ such that $X^HX = I$. Set step size $\gamma := 1$.
2) Compute D_X, which is the derivative of f at X [cf. (47)].
3) Compute the descent direction $Z := -(I - XX^H)D_X$.
4) Evaluate $\langle Z, Z \rangle = \operatorname{tr}\{Z^HZ\}$. If $\sqrt{\langle Z, Z \rangle}$ is sufficiently small, then stop.
5) If $f(X) - f(\mathrm{qf}_p\{X + 2\gamma Z\}) \geq \gamma \langle Z, Z \rangle$, then set $\gamma := 2\gamma$, and repeat Step 5. [The Q-Factor operator $\mathrm{qf}_p\{\cdot\}$ is defined in (44).]
6) If $f(X) - f(\mathrm{qf}_p\{X + \gamma Z\}) < (1/2)\gamma \langle Z, Z \rangle$, then set $\gamma := (1/2)\gamma$, and repeat Step 6.
7) Set $X := \mathrm{qf}_p\{X + \gamma Z\}$. Go to Step 2.

If the cost function $f(X)$ satisfies A2) in Section VII, then Algorithm 24 simplifies slightly; under A2), $f(\mathrm{qf}_p\{Y\}) = f(Y)$ holds for any full rank matrix $Y \in \mathbb{C}^{n\times p}$. Therefore, the Q-Factor operator can be omitted from Steps 5 and 6 of Algorithm 24.

B. Modified Newton Method on the Complex Grassmann Manifold

This section derives another algorithm for minimizing $f(X)$ subject to $X^HX = I$ when $f(X)$ satisfies either A1) or both A1) and A2) in Section VII. It requires the evaluation of $f(X)$ and its first two derivatives at each iteration. Unlike Algorithm 24, the algorithm in this section only converges to a local minimum if it is initialized to a point sufficiently close to the local minimum. However, when it does converge, it achieves a quadratic rate of convergence, as opposed to the linear rate of convergence exhibited by Algorithm 24.

By definition, the Newton step moves to the critical point of the quadratic approximation of the local cost function (43). The quadratic approximation is given in the following proposition.

Proposition 25: Given a cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$ satisfying A1) of Section VII, let $g(Z) = \tilde{f}(\pi(X + Z))$ be the local cost function about a given point $X \in St(n, p)$, where $\tilde{f}$ is defined in (42). Then, for any $Z \in T_{\lfloor X \rfloor}(n, p)$

$$g(Z) = f(X) + \Re \operatorname{tr}\{Z^H D_X\} + \tfrac{1}{2} \operatorname{vec}\{Z\}^H \cdot \left[H_X - \tfrac{1}{2}\left((X^H D_X + D_X^H X)^T \otimes I_n\right)\right] \operatorname{vec}\{Z\} + \tfrac{1}{2} \Re\left\{\operatorname{vec}\{Z\}^T C_X \operatorname{vec}\{Z\}\right\} + O(\|Z\|^3) \tag{51}$$

where $D_X \in \mathbb{C}^{n\times p}$ and H_X, $C_X \in \mathbb{C}^{np\times np}$ are the derivative and Hessian of f at X, respectively (see Section II). If f satisfies A2) of Section VII, then (51) simplifies to

$$g(Z) = f(X) + \Re \operatorname{tr}\{Z^H D_X\} + \tfrac{1}{2} \operatorname{vec}\{Z\}^H H_X \operatorname{vec}\{Z\} + \tfrac{1}{2} \Re\{\operatorname{vec}\{Z\}^T C_X \operatorname{vec}\{Z\}\} + O(\|Z\|^3). \tag{52}$$

Proof: It is clear that $g(Z) = f(\pi(X + Z))$, where π is the projection operator onto the Stiefel manifold (Definition 5). Thus, $g(Z)$ is given by (30) in general. If f satisfies A2), then $\tilde{f}(\lfloor X \rfloor) = f(X)$ for all $X \in \mathbb{C}^{n\times p}$. Thus, Proposition 22 shows that

$$g(tZ) = \tilde{f}(\pi(X + tZ)) = \tilde{f}(\lfloor X + tZ + O(t^3) \rfloor) = f(X + tZ) + O(t^3) \tag{53}$$

from which (52) follows from (1). □

The location of the critical point of (51) can be found by applying Proposition 3; observe that V_3 in Section III is the vector space $T_{\lfloor X \rfloor}(n, p)$.

Algorithm 26 (Modified Newton Method on Grassmann Manifold): Given a cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$, which is twice differentiable and satisfies A1) of Section VII, the following algorithm attempts to converge to a local minimum of $f(X)$ subject to the constraint that $X^H X = I$. It requires that the derivative $D_X \in \mathbb{C}^{n\times p}$ and the Hessian H_X, $C_X \in \mathbb{C}^{np\times np}$ of f at the point X can be computed for any $X \in St(n, p)$.

1) Choose $X \in \mathbb{C}^{n\times p}$ and $X_\perp \in \mathbb{C}^{n\times(n-p)}$ such that $[X\ X_\perp]^H[X\ X_\perp] = I$.
2) Compute D_X, which is the derivative of f at X, and H_X, C_X, which is the Hessian of f at X, as defined in Section II. If $\sqrt{\mathrm{tr}\{D_X^H X_\perp X_\perp^H D_X\}}$ is sufficiently small, then stop.
3) Compute the Newton step $Z \in \mathbb{C}^{n\times p}$ as follows. Set $G_1 := (I_p \otimes X_\perp)^H H_X (I_p \otimes X_\perp) - (1/2)(X^H D_X + D_X^H X)^T \otimes I_{n-p}$ and $G_2 := (I_p \otimes X_\perp)^T C_X (I_p \otimes X_\perp)$. Solve

$$\begin{bmatrix} \Re\{G_1+G_2\} & -\Im\{G_1+G_2\} \\ \Im\{G_1-G_2\} & \Re\{G_1-G_2\} \end{bmatrix} \begin{bmatrix} \Re\,\mathrm{vec}\{B\} \\ \Im\,\mathrm{vec}\{B\} \end{bmatrix} = -\begin{bmatrix} \Re\,\mathrm{vec}\{X_\perp^H D_X\} \\ \Im\,\mathrm{vec}\{X_\perp^H D_X\} \end{bmatrix} \quad (54)$$

   for $B \in \mathbb{C}^{(n-p)\times p}$. Set $Z := X_\perp B$.
4) If $f(X) \le f(\mathrm{qf}_p\{X+Z\})$, then abort. [The Q-Factor operator $\mathrm{qf}_p\{\cdot\}$ is defined in (44).]
5) Set $[X\ X_\perp] := \mathrm{qf}_n\{X+Z\}$. Go to Step 2.

Remark: The quantity $\sqrt{\mathrm{tr}\{D_X^H X_\perp X_\perp^H D_X\}}$ in Step 2 of Algorithm 26 equals $\sqrt{\langle Z, Z\rangle}$ in Step 4 of Algorithm 24, that is, it equals the norm of the gradient of f at X.

If the cost function $f(X)$ satisfies (A2) of Section VII, then Proposition 25 shows that G_1 in Step 3 of Algorithm 26 simplifies to $G_1 := (I_p \otimes X_\perp)^H H_X (I_p \otimes X_\perp)$. Furthermore, as in Section VII-A, the Q-Factor operator can be omitted in Step 4 of Algorithm 26.

As is the case with all Newton-type algorithms, Algorithm 26 can fail to take a descent direction if the current point X is not sufficiently close to a minimum to ensure that the Hessian is positive definite. If Algorithm 26 fails to take a descent step, several iterations of Algorithm 24 can be used to move closer to a minimum before restarting Algorithm 26.

VIII. Computing an Extreme Eigenvector

A. Introduction

It is well known [9], [10] that for a Hermitian matrix $A \in \mathbb{C}^{n\times n}$, $f(\boldsymbol{x}) = (1/2)\boldsymbol{x}^H A \boldsymbol{x}$ achieves its minimum, subject to $\boldsymbol{x}^H\boldsymbol{x} = 1$, when $\boldsymbol{x} \in \mathbb{C}^n$ corresponds to a minimal eigenvector, that is, an eigenvector associated with the smallest eigenvalue of A. This section specializes Algorithm 24, which is the modified steepest descent on the Grassmann manifold algorithm, to this particular cost function. An attractive feature of this specialization is that the Armijo step size rule is replaced by the optimal step size rule.

The reasons for deriving a novel algorithm for computing a minimal eigenvector are now listed.

1) It serves as a worked example of how to apply the optimization algorithms in this paper.
2) It is used to demonstrate that the algorithms in this paper can have significantly different properties compared with classical algorithms for solving the same problem.
3) It can be used in a number of signal processing applications [6], [36] that require the computation and subsequent tracking of a minimal eigenvector.
4) It has several advantages over existing algorithms for computing a minimal eigenvector.

Computing a minimal eigenvector appears to be intrinsically more difficult than computing a maximal eigenvector [7], [9], [11], [14], [15], [19], [29], [35]–[37]. The two standard methods that almost always converge to a minimal eigenvector are the inverse iteration method (described later) and steepest descent methods [7], [15]. Other methods, such as Newton methods [14], Rayleigh quotient iterations [9], and so forth, converge to the "nearest" eigenvector rather than to a minimal eigenvector.

One advantage of the novel algorithm derived here is that unlike the recently proposed steepest descent algorithms in [7] and [15], it takes a step of optimal size at each iteration. This feature is particularly attractive in tracking applications where A varies over time. Furthermore, it suggests that in certain applications at least, the Euclidean-projection-based parameterization of the Grassmann manifold in this paper is a more useful choice than the geodesic-based parameterization used in [7] and [15]; it does not appear to be possible to compute the optimal step size for the algorithms in [7] and [15].

The other advantages are that the algorithm is guaranteed to converge to a minimal eigenvector, provided the initial vector is not orthogonal to the space spanned by the minimal eigenvectors, and unlike the classical inverse iteration method, the algorithm is not sensitive to closely spaced eigenvalues. These properties are proved in the following section and corroborated by simulations in Section VIII-C.

B. Algorithm and Its Derivation

The notation and results in Sections II and VII are used here with the minor change that the matrices X and Z are replaced by the vectors $\boldsymbol{x}$ and $\boldsymbol{z}$. The steepest descent direction $\boldsymbol{z}$ defined in Step 3 of Algorithm 24 is readily calculated from Example 1 of Section II under the assumption that $\boldsymbol{x}^H\boldsymbol{x} = 1$:

$$f(\boldsymbol{x}) = \tfrac{1}{2}\boldsymbol{x}^H A \boldsymbol{x}, \quad D_{\boldsymbol{x}} = A\boldsymbol{x}$$
$$\boldsymbol{z} = -(I - \boldsymbol{x}\boldsymbol{x}^H)A\boldsymbol{x} = -A\boldsymbol{x} + (\boldsymbol{x}^H A \boldsymbol{x})\boldsymbol{x}. \quad (55)$$

It is convenient to interpret $\boldsymbol{x}^H A \boldsymbol{x}$ as a weighted average of the eigenvalues of A. Let

$$A = \sum_{i=1}^{n} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^H, \quad \boldsymbol{v}_i^H \boldsymbol{v}_j = \delta(i-j), \quad \lambda_1 \le \cdots \le \lambda_n \quad (56)$$

be the eigendecomposition of A, where δ is Dirac's delta function, and let $c_i \in \mathbb{R}$ be such that $\boldsymbol{x} = \sum_{i=1}^{n} c_i \boldsymbol{v}_i$. Then

$$\overline{\lambda} = \frac{\boldsymbol{x}^H A \boldsymbol{x}}{\boldsymbol{x}^H \boldsymbol{x}} = \frac{\sum_{i=1}^{n} c_i^2 \lambda_i}{\sum_{i=1}^{n} c_i^2} \tag{57}$$

is a weighted sum of the eigenvalues λ_i of A. In particular

$$\lambda_1 \leq \overline{\lambda} \leq \lambda_n. \tag{58}$$

The Q-Factor operator appearing in Algorithm 24 is readily evaluated when applied to a vector. Indeed

$$f(\mathrm{qf}_1\{\boldsymbol{y}\}) = \tfrac{1}{2}(\boldsymbol{y}^H \boldsymbol{y})^{-1} \boldsymbol{y}^H A \boldsymbol{y} \tag{59}$$

for arbitrary $\boldsymbol{y} \in \mathbb{C}^n$. It will be shown that the decrease in cost $f(\boldsymbol{x}) - f(\mathrm{qf}_1\{\boldsymbol{x} + \gamma \boldsymbol{z}\})$ can be expressed in terms of the following variables.

$$\overline{A} = A - \overline{\lambda} I, \quad \alpha = \boldsymbol{x}^H \overline{A}^2 \boldsymbol{x}, \quad \beta = \boldsymbol{x}^H \overline{A}^3 \boldsymbol{x}. \tag{60}$$

Note that α and β are both real-valued since $\overline{A} = \overline{A}^H$ and $\boldsymbol{x}^H \overline{A} \boldsymbol{x} = 0$ since it is assumed that $\boldsymbol{x}^H \boldsymbol{x} = 1$. Straightforward manipulation shows that

$$\begin{aligned} f(\boldsymbol{x}) - f(\mathrm{qf}_1\{\boldsymbol{x} + \gamma \boldsymbol{z}\}) &= \tfrac{1}{2}\{\overline{\lambda} - [(\boldsymbol{x} - \gamma \overline{A} \boldsymbol{x})^H (\boldsymbol{x} - \gamma \overline{A} \boldsymbol{x})]^{-1} \\ &\quad \cdot (\boldsymbol{x} - \gamma \overline{A} \boldsymbol{x})^H (\overline{A} + \overline{\lambda} I)(\boldsymbol{x} - \gamma \overline{A} \boldsymbol{x})\} \qquad (61) \\ &= \frac{\gamma\left(\alpha - \frac{1}{2}\gamma\beta\right)}{1 + \alpha\gamma^2}. \qquad (62) \end{aligned}$$

Differentiating (62) with respect to γ and setting the result to zero shows that the greatest decrease in cost occurs when γ is the unique positive root of the quadratic equation

$$\alpha^2 \gamma^2 + \beta\gamma - \alpha = 0. \tag{63}$$

Let $\gamma^{(\mathrm{opt})}$ denote this optimal value. It is interesting to note (cf. Step 6 of Algorithm 24) that

$$f(\boldsymbol{x}) - f\left(\mathrm{qf}_1\left\{\boldsymbol{x} + \gamma^{(\mathrm{opt})} \boldsymbol{z}\right\}\right) = \tfrac{1}{2}\gamma^{(\mathrm{opt})} \langle \boldsymbol{z}, \boldsymbol{z} \rangle. \tag{64}$$

Algorithm 24 specializes to the following. Note that the expressions for α and β in Step 3 of Algorithm 27 are equivalent to those in (60). In addition, note that α, β and $\overline{\lambda}$ must be real valued.

Algorithm 27 (Computing a Minimal Eigenvector): Let $A \in \mathbb{C}^{n \times n}$ be an arbitrary Hermitian matrix. The following algorithm converges to a minimal eigenvector of A with probability one (see Theorem 28 later).

1) Randomly choose an $\boldsymbol{x} \in \mathbb{C}^n$ with unit norm ($\boldsymbol{x}^H \boldsymbol{x} = 1$).
2) Compute the descent direction $\boldsymbol{z} := \overline{\lambda} \boldsymbol{x} - A \boldsymbol{x}$, where $\overline{\lambda} := \boldsymbol{x}^H A \boldsymbol{x}$. If $\sqrt{\boldsymbol{z}^H \boldsymbol{z}}$ is sufficiently small, then stop.
3) Compute $\alpha := \boldsymbol{x}^H A^2 \boldsymbol{x} - \overline{\lambda}^2$ and $\beta := \boldsymbol{x}^H A^3 \boldsymbol{x} - 3\alpha\overline{\lambda} - \overline{\lambda}^3$. Set γ to the positive root of $\alpha^2\gamma^2 + \beta\gamma - \alpha = 0$.
4) Set $\boldsymbol{x} := \boldsymbol{x} + \gamma \boldsymbol{z}$. Renormalize by setting $\boldsymbol{x} := \boldsymbol{x}/\sqrt{\boldsymbol{x}^H \boldsymbol{x}}$. Go to Step 2.

Before proving global convergence, two properties of Algorithm 27 are stated. Algorithm 27 is invariant to shifts; replacing A with $A - \lambda I$ for any $\lambda \in \mathbb{R}$ has no effect. This supports the empirical evidence (see Section VIII-C) that closely spaced eigenvalues, which are known to reduce severely the rate of convergence of power methods [9], do not affect the performance of Algorithm 27. Algorithm 27 is also invariant to orthogonal changes of coordinates. That is, if Algorithm 27 produces the sequence $\{\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \cdots\}$, then replacing A with QAQ^H and $\boldsymbol{x}^{(0)}$ with $Q\boldsymbol{x}^0$ will produce the sequence $\{Q\boldsymbol{x}^{(0)}, Q\boldsymbol{x}^{(1)}, \cdots\}$.

Theorem 28 (Global Convergence): Let $\boldsymbol{x}$ be the initial vector chosen in Step 1 of Algorithm 27. If λ_1 is the smallest eigenvalue of A and there exists an eigenvector $\boldsymbol{v}_1$ satisfying both $A\boldsymbol{v}_1 = \lambda_1 \boldsymbol{v}_1$ and $\boldsymbol{v}_1^H \boldsymbol{x} \neq 0$, then Algorithm 27 converges to an eigenvector $\boldsymbol{v}$ satisfying $A\boldsymbol{v} = \lambda_1 \boldsymbol{v}$.

Proof: Referring to Algorithm 27, since $\boldsymbol{z}^H \boldsymbol{z} = 0$ if and only if $\boldsymbol{x}$ is an eigenvector of A, it is clear that Algorithm 27 converges to an eigenvector $\boldsymbol{v}$ of A. Let λ be the eigenvalue associated with $\boldsymbol{v}$. Assume to the contrary that $\lambda > \lambda_1$. Since $\boldsymbol{v}$ must then be orthogonal to $\boldsymbol{v}_1$, this implies $|\boldsymbol{v}_1^H \boldsymbol{x}| \to 0$. It will be shown below that one iteration of Algorithm 27 increases $|\boldsymbol{v}_1^H \boldsymbol{x}|$ if the step size $\gamma > 0$ satisfies

$$\gamma[\alpha - (\overline{\lambda} - \lambda_1)^2] < 2(\overline{\lambda} - \lambda_1). \tag{65}$$

Since $\boldsymbol{x} \to \boldsymbol{v}$, it follows that $\overline{\lambda} = \boldsymbol{x}^H A \boldsymbol{x} \to \lambda$, and $\alpha = \boldsymbol{x}^H A^2 \boldsymbol{x} - \overline{\lambda}^2 \to 0$. This means there will come a time when $\alpha - (\overline{\lambda} - \lambda_1)^2 < 0$, and hence, (65) will also hold for all subsequent iterations. This contradicts $|\boldsymbol{v}_1^H \boldsymbol{x}| \to 0$, proving that $\lambda = \lambda_1$.

To show that (65) implies that $|\boldsymbol{v}_1^H \boldsymbol{x}|$ will increase, note first that direct substitution proves that

$$\boldsymbol{v}_1^H \frac{\boldsymbol{x} + \gamma \boldsymbol{z}}{\sqrt{(\boldsymbol{x} + \gamma \boldsymbol{z})^H (\boldsymbol{x} + \gamma \boldsymbol{z})}} = \frac{1 - \gamma(\lambda_1 - \overline{\lambda})}{\sqrt{1 + \alpha\gamma^2}}. \tag{66}$$

Since $\alpha \geq 0$, it is readily verified that $\left[1 - \gamma(\lambda_1 - \overline{\lambda})/\sqrt{1 + \alpha\gamma^2}\right]^2 > 1$ if and only if (65) holds. That is, (65) implies that $|\boldsymbol{v}_1^H \boldsymbol{x}|$ will increase unless $|\boldsymbol{v}_1^H \boldsymbol{x}| = 0$. However, the latter cannot occur because, from (58), $\lambda_1 \leq \overline{\lambda}$, and therefore, $1 - \gamma(\lambda_1 - \overline{\lambda})$ can never be zero. □

Finally, it is remarked that Algorithm 26 may also be applied to the cost function $f(\boldsymbol{x}) = (1/2)\boldsymbol{x}^H A \boldsymbol{x}$. Since $D_{\boldsymbol{x}} = A\boldsymbol{x}$, $H_{\boldsymbol{x}} = A$, and $C_{\boldsymbol{x}} = 0$ (see Example 1), Step 3 of Algorithm 26 becomes $Z := X_\perp \boldsymbol{b}$, where $\boldsymbol{b} \in \mathbb{C}^{n-1}$ satisfies the linear equation

$$[(\boldsymbol{x}^H A \boldsymbol{x}) I_{n-1} - X_\perp^H A X_\perp] \boldsymbol{b} = X_\perp^H A \boldsymbol{x}. \tag{67}$$

C. Simulations

This section studies the convergence rate of Algorithm 27 and compares it with traditional methods for calculating extremal eigenvectors. It is demonstrated that the performance of Algorithm 27 is relatively insensitive to the actual eigenvalue distribution.

The inverse iteration method [9] for finding an eigenvector of the matrix A associated with the eigenvalue having the smallest

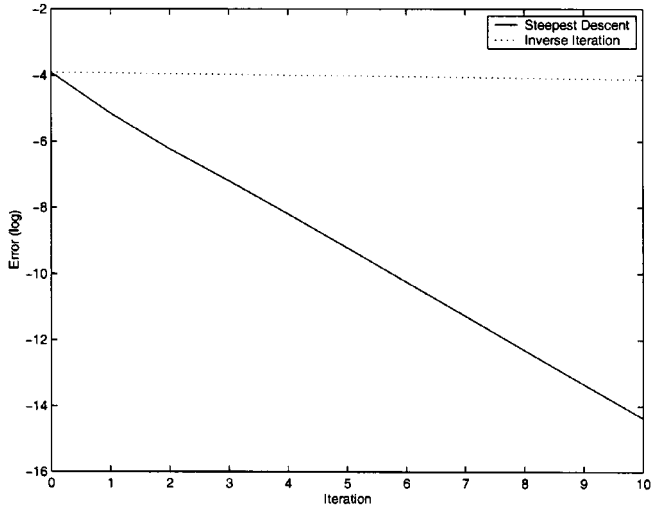

Fig. 1. Graph comparing the convergence rates of the steepest descent and inverse iteration algorithms when applied to the matrix $A = \text{diag}\{1,\ 1.01,\ 1.02,\ 1.03,\ 1.04\}$.

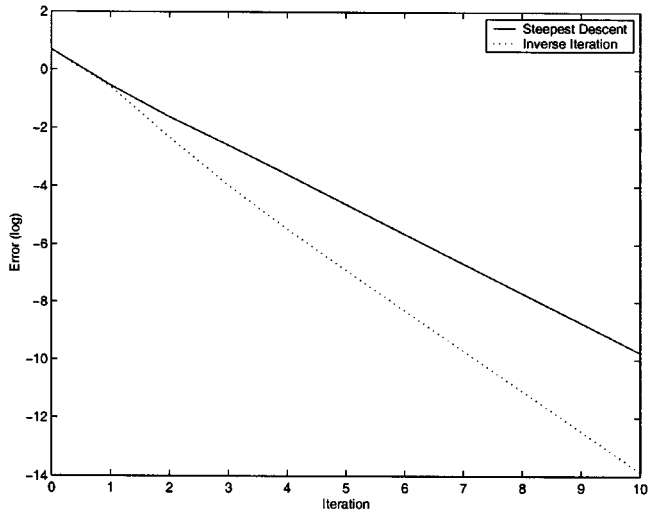

Fig. 2. Graph comparing the convergence rates of the steepest descent and inverse iteration algorithms when applied to the matrix $A = \text{diag}\{1,\ 2,\ 3,\ 4,\ 5\}$.

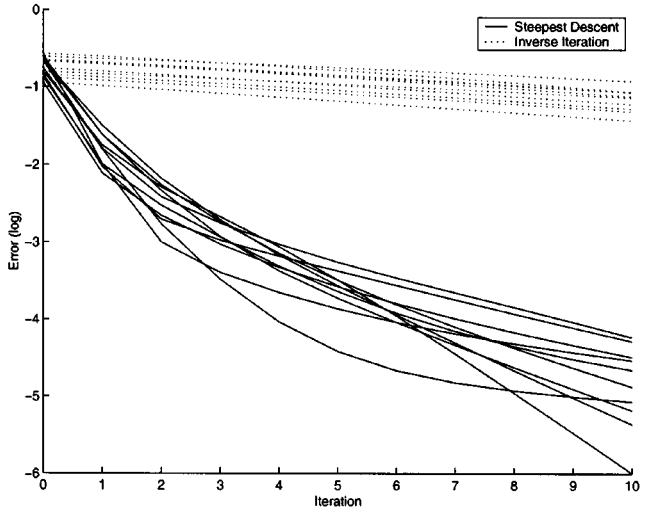

Fig. 3. Graph comparing the convergence rates of the steepest descent and inverse iteration algorithms when applied to ten randomly generated 20-by-20 matrices with eigenvalues uniformly distributed between 10 and 11.

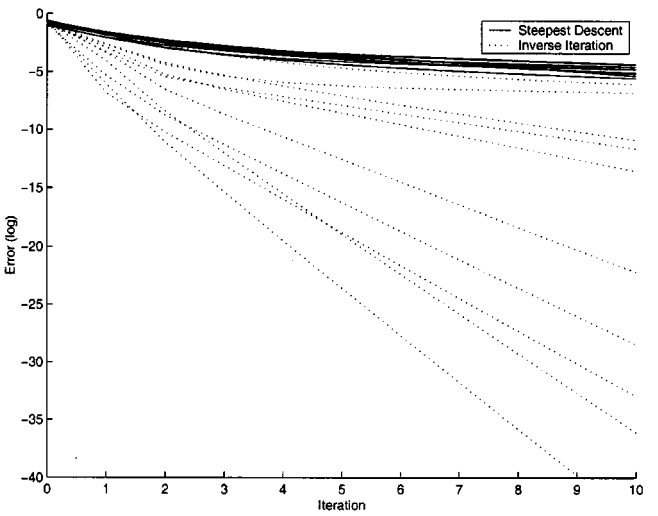

Fig. 4. Graph comparing the convergence rates of the steepest descent and inverse iteration algorithms when applied to ten randomly generated 20-by-20 matrices with eigenvalues uniformly distributed between 0 and 1.

absolute value[3] is to generate a sequence $\{\boldsymbol{x}^{(k)}\}$ of vectors according to the rule

$$\boldsymbol{x}^{(k+1)} = \frac{A^{-1}\boldsymbol{x}^{(k)}}{\|A^{-1}\boldsymbol{x}^{(k)}\|}. \tag{68}$$

Figs. 1–4 compare the inverse iteration method (68) with the steepest descent method (Algorithm 27). Figs. 1 and 3 show that Algorithm 27 outperforms (68) if the eigenvalues of A are closely spaced, whereas Figs. 2 and 4 demonstrate that the converse holds as well. This is now explained in more detail.

It is well-known that the convergence rate of the power and inverse iteration methods [9] applied to the matrix A critically depends on the eigenvalue distribution of A. Indeed, replacing A with $A + \lambda I$ for some constant $\lambda \in \mathbb{R}$ (which is known as a shift in the literature) significantly alters the convergence rate of (68). In comparison, Section VIII shows that such shifts do not alter Algorithm 27 at all. It is therefore expected that the inverse iteration method will exhibit convergence rates ranging from extremely poor to extremely good, depending on the eigenvalue distribution of A, whereas Algorithm 27 is expected to achieve a steady rate of convergence over a wide range of eigenvalue distributions.

This hypothesis was tested by plotting the log of the error, which is defined as $\log((\boldsymbol{x}^{(k)})^H A\boldsymbol{x}^{(k)} - \lambda_{\min}\{A\})$, where $\lambda_{\min}\{A\}$ is the smallest eigenvalue of A, against the iteration number k. (The fact that the resulting graphs in Figs. 1–4 are essentially straight lines shows that both algorithms achieve a linear rate of convergence [22].) Fig. 1 was generated by applying the algorithms to the matrix $A = \text{diag}\{1, 1.01, 1.02, 1.03, 1.04\}$. (In all

[3]It is important to note that steepest descent algorithms converge to the smallest eigenvalue, whereas the inverse iteration method converges to the eigenvalue having the smallest absolute value. Similarly, the power method (which will be mentioned later) converges to the eigenvalue having the largest absolute value, whereas steepest ascent algorithms converge to the largest eigenvalue. Therefore, when comparing algorithms, it is important to choose A to be positive definite.

simulations, the initial starting vector was chosen to be $\boldsymbol{x}^{(0)} = [1 \quad 1 \quad 1 \quad 1 \quad 1]^T$.) Since the eigenvalues are closely spaced, Algorithm 27 significantly outperforms (68). Conversely, Fig. 2 shows that (68) outperforms Algorithm 27 when applied to the matrix $A = \text{diag}\{1, 2, 3, 4, 5\}$. Figs. 3 and 4 suggest that this behavior is typical. Fig. 4 shows the performance of the two algorithms when applied to ten randomly generated 20-by-20 matrices with eigenvalues uniformly distributed between 0 and 1. The same ten matrices were then shifted so that their eigenvalues lay between 10 and 11 (that is, each A was replaced with $A + 10I$) and the results plotted in Fig. 3. Whereas the performance of Algorithm 27 is unaltered, (68) performs badly in Fig. 3 but exceptionally well in Fig. 4.

The steepest ascent method, which is obtained by replacing A with $-A$ in Algorithm 27, was compared with the power method for converging to an eigenvector associated with the largest eigenvalue of A. The power method updates $\boldsymbol{x}^{(k)}$ according to the rule [cf. (68)] $\boldsymbol{x}^{(k+1)} = A\boldsymbol{x}^{(k)}/\|A\boldsymbol{x}^{(k)}\|$). Figs. 5 and 6 were generated analogously to Figs. 3 and 4. They demonstrate that Algorithm 27 achieves a convergence rate that is much less sensitive to the location of the eigenvalues of A than the power method does.

Finally, the rapid convergence of the Newton method (Algorithm 26 applied to the cost function $f(\boldsymbol{x}) = (1/2)\boldsymbol{x}^H A\boldsymbol{x}$ as in Section VIII) for finding a minimal eigenvector is illustrated in Fig. 7. (The Newton method converged to the exact answer up to machine precision on the third iteration.) Note that two iterations of Algorithm 27 were performed before running the Newton method since otherwise, the Newton method would fail to converge to a minimal eigenvector.

IX. Discussion

This section discusses the conceptual differences between the optimization approach in this paper and the approach in [8]. It also gives a qualitative description of when the Newton algorithms here are expected to outperform the Newton algorithms in [8].

It is first noted that the general framework in Section V for minimizing a function on a manifold—namely, given a point X on the manifold, apply a single iteration of the steepest descent or Newton algorithm to the local cost function $g(Z) = f(h(Z))$, where h is a local parameterization about X, then move to the new point $X := h(Z)$ and repeat—is more general than the framework in [8]. Choosing $h(Z)$ to be the exponential map (which corresponds to using geodesics to locally parameterise the manifold) results in the Newton algorithm in [8]. The differences between the algorithms can therefore be understood by determining what effect the choice of the local parameterization $h(Z)$ has on the computational complexity and the rate of convergence of the algorithms.

As mentioned earlier, the local parameterization used in this paper is computationally simpler to compute than the one in [8]. The asymptotic rate of convergence of the modified Newton algorithms here is the same as for the Newton methods in [8], that is, they all asymptotically achieve a quadratic rate of convergence. However, for a given cost function, it can be expected that one algorithm will converge faster than the other one. (Which

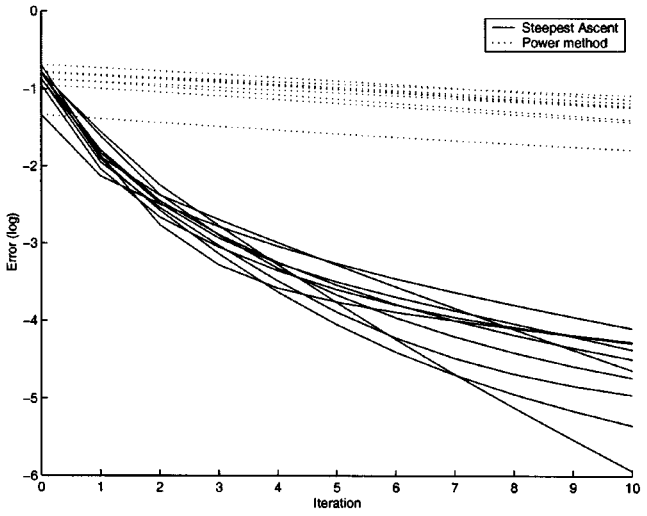

Fig. 5. Graph comparing the convergence rates of the steepest ascent and power method algorithms when applied to ten randomly generated 20-by-20 matrices with eigenvalues uniformly distributed between 10 and 11.

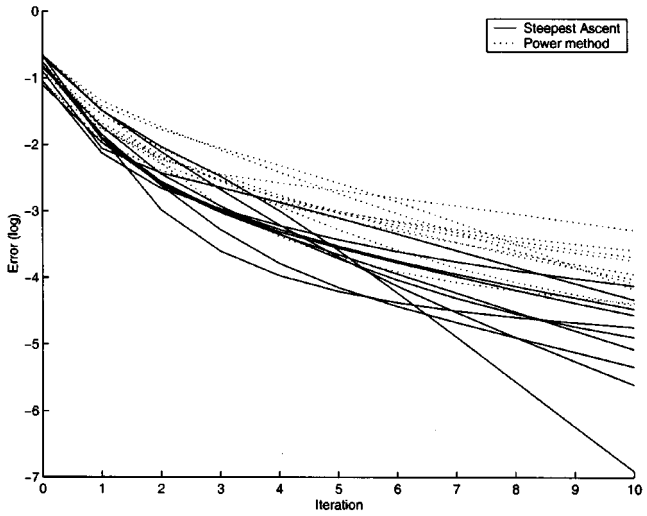

Fig. 6. Graph comparing the convergence rates of the steepest ascent and power method algorithms when applied to ten randomly generated 20-by-20 matrices with eigenvalues uniformly distributed between 0 and 1.

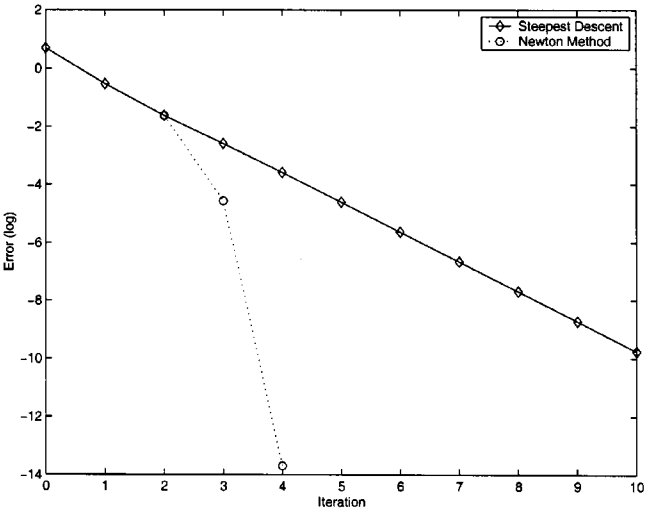

Fig. 7. Graph showing the rapid convergence of the Newton method when used to find the minimal eigenvector of the matrix $A = \text{diag}\{1, 2, 3, 4, 5\}$.

algorithm is the faster depends on the cost function.) The following example is used to explain this phenomenon.

Consider the ordinary Newton method applied to the two cost functions $f(x, y) = (x-2)^2 + (y-3)^2$ and $g(r, \theta) = (r\cos\theta - 2)^2 + (r\sin\theta - 3)^2$. Note that f and g represent the same function expressed in different coordinate systems. Specifically, $f = g \circ p$, where $p: \mathbb{R}^2 \to \mathbb{R}^2$ is the mapping from Cartesian to polar coordinates. The Newton method approximates the cost function by a quadratic cost at each iteration. Since $f(x - x_0, y - y_0)$ is quadratic about any point (x_0, y_0), the Newton method applied to f converges in a single iteration. However, since $g(r - r_0, \theta - \theta_0)$ is not quadratic about any point (r_0, θ_0), the Newton method will not converge in a single iteration when applied to g. Conversely, a "Newton algorithm in polar coordinates" would converge in one iteration when applied to g but take longer to converge when applied to f.

In the above example, the change of coordinates p is analogous to the local parameterization h used to define Newton algorithms on a manifold. In a qualitative sense, it implies that the algorithm that achieves the faster convergence for a particular cost function $f(X)$ is the algorithm that uses the local parameterization $h(Z)$ (either the exponential map in [8] or the Euclidean projection operator in this paper), resulting in the local cost function $f(h(Z))$ more closely approximating a quadratic function. Simulations in [18] show that for one particular class of cost functions, the algorithms here converge faster than the algorithms in [8]. However, the preceding argument suggests that there may exist another class of cost functions for which the converse is true.

Since the performance of the algorithms depends on the choice of local parameterization $h(Z)$ relative to the cost function $f(X)$, it is worthwhile understanding the motivation behind the choice of local parameterization here and in [8].

In [8], the Stiefel manifold was made into a Riemannian manifold by endowing it with its canonical metric.[4] A connection function[5] (the Levi-Civita connection) was also given to the manifold. Doing this made it possible to calculate the gradient and Hessian of a function on the manifold. Roughly speaking, the classical formula for computing the Newton step was generalized in [8] by replacing the first- and second-order derivatives in Newton's formula by the gradient and Hessian of the cost function on the manifold. Newton's formula results in a vector pointing in the direction to move, and since following a geodesic corresponds to walking in a straight line, it is natural for [8] to interpret Newton's formula as requiring the Newton step to be taken along a geodesic.

The motivation for considering a different approach in this paper is that there does not seem to be an intrinsic connection between the Riemannian geometry of the Stiefel manifold and the minimization of an arbitrary cost function. Why follow a geodesic, which is costly to compute, if it is not essential?

The algorithms in this paper avoid giving the Stiefel manifold a metric structure and a connection function. They are able to do this because they never compute a gradient or a Hessian of a function on the manifold. Instead, at each iteration, they form a local cost function whose domain is a vector space and not a manifold. Although the derivative and Hessian of this local cost function can only be calculated once the vector space is given an inner product structure (which is tantamount to giving the Stiefel manifold a metric structure except that there is no smoothness requirement), *the Newton step is independent of the inner product structure chosen*. Therefore, the Newton algorithms in this paper can claim (for better or for worse) to be unrelated to any Riemannian structure put on the manifold. (They do, however, depend on the choice of norm used to define the projection operator.)

While the approach in [8] depends on the Riemannian structure given to the Stiefel manifold, the approach in this paper depends on the projection operator used to define the local parameterization. The Euclidean norm in Definition 5 was chosen somewhat arbitrarily; a number of interesting cost functions can be written in terms of a Euclidean norm so that it seemed sensible to use the Euclidean norm to define the projection operator as well. A positive consequence of this choice is that it allows the optimal step size to be computed in Section VIII for a particular cost function.

X. Conclusion

This paper derived novel algorithms for the minimization of a cost function $f: \mathbb{C}^{n\times p} \to \mathbb{R}$ subject to the constraint $X^H X = I$. The key feature of the algorithms is that they reduce the dimensionality of the optimization problem by reformulating the optimization problem as an unconstrained one on either the Stiefel or Grassmann manifolds. A consequence of this is that the convergence properties of the algorithms may be different from those of traditional methods. To verify this assertion, the algorithms were applied to the problem of finding an eigenvector associated with the smallest eigenvalue of a Hermitian matrix. Simulations showed that the performance of the resulting algorithms exhibited quite different behavior from the traditional power and inverse iteration methods for computing an extremal eigenvector.

Acknowledgment

The author wishes to thank R. Mahony for many fruitful discussions centered around the problem of minimizing a cost function on a manifold. The author also wishes to thank the anonymous reviewers for their insightful comments, which led to a significantly improved presentation of the paper.

References

[1] A. Belouchrani, K. Abed-Meraim, J.-F. Cardoso, and E. Moulines, "A blind source separation technique using second-order statistics," *IEEE Trans. Signal Processing*, vol. 45, pp. 434–444, Feb. 1997.

[2] W. M. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*, second ed. New York: Academic, 1986.

[3] R. W. Brockett, "Dynamical systems that sort lists, diagonalise matrices, and solve linear programming problems," *Linear Algebra Appl.*, vol. 146, pp. 79–91, 1991.

[4] J. Cardoso and A. S. Soulomiac, "Blind beamforming for non-Gaussian signals," *Proc. Inst. Elect. Eng. F*, vol. 140, pp. 362–370, Dec. 1993.

[5] B. Champagne and Q.-G. Liu, "Plane rotation-based EVD updating schemes for efficient subspace tracking," *IEEE Trans. Signal Processing*, vol. 46, pp. 1886–1900, July 1998.

[4] A metric is an inner product structure given to each tangent space that varies in a smooth way [2].

[5] A connection function is required before "second-order derivatives" can be meaningfully calculated on a manifold.

[6] P. Comon and G. H. Golub, "Tracking a few extreme singular values and vectors in signal processing," *Proc. IEEE*, vol. 78, pp. 1327–1343, Aug. 1990.
[7] S. C. Douglas, S. Amari, and S.-Y. Kung, "On gradient adaptation with unit-norm constraints," *IEEE Trans. Signal Processing*, vol. 48, pp. 1843–1847, June 2000.
[8] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. Applicat.*, vol. 20, no. 2, pp. 303–353, 1998.
[9] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1996.
[10] U. Helmke and J. B. Moore, *Optimization and Dynamical Systems*. New York: Springer-Verlag, 1994.
[11] Z. Kang, C. Chatterjee, and V. P. Roychowdhury, "An adaptive quasi-Newton algorithm for eigensubspace estimation," *IEEE Trans. Signal Processing*, vol. 48, pp. 3328–3333, Dec 2000.
[12] W.-S. Lu, S.-C. Pei, and P.-H. Wang, "Weighted low-rank approximation of general complex matrices and its application in the design of 2-D digital filters," *IEEE Trans. Circuits Syst. I*, vol. 44, pp. 650–655, July 1997.
[13] R. E. Mahony, "Optimization algorithms on homogeneous spaces: with applications in linear systems theory," Ph.D. dissertation, Australian Nat. Univ., Canberra, 1994.
[14] R. E. Mahony, "The constrained Newton method on a Lie group and the symmetric eigenvalue problem," *Linear Algebra Applicat.*, vol. 248, pp. 67–89, 1996.
[15] R. E. Mahony, U. Helmke, and J. B. Moore, "Gradient algorithms for principal component analysis," *J. Austr. Math. Soc. B*, vol. 37, no. 4, pp. 430–450, 1996.
[16] R. E. Mahony and J. H. Manton, "The geometry of the Newton method on noncompact Lie groups," *J. Global Optimiz.*, to be published.
[17] J. H. Manton and Y. Hua, "Convolutive reduced rank Wiener filtering," in *Proc. IEEE Conf. Acoust., Speech, Signal Process.*, Salt Lake City, UT, May 2001.
[18] J. H. Manton, R. Mahony, and Y. Hua, "The geometry of weighted low rank approximations," *IEEE Trans. Signal Processing*, submitted for publication.
[19] G. Mathew, V. U. Reddy, and S. Dasgupta, "Adaptive estimation of eigensubspace," *IEEE Trans. Signal Processing*, vol. 43, pp. 401–411, Feb. 1995.
[20] O. J. Micka and A. J. Weiss, "Estimating frequencies of exponentials in noise using joint diagonalization," *IEEE Trans. Signal Processing*, vol. 47, pp. 341–348, Feb. 1999.
[21] P. Petersen, *Riemannian Geometry*. New York: Springer-Verlag, 1998.
[22] E. Polak, *Optimization: Algorithms and Consistent Approximations*. New York: Springer-Verlag, 1997.
[23] D. J. Rabideau, "Fast, rank adaptive subspace tracking and applications," *IEEE Trans. Signal Processing*, vol. 44, pp. 2229–2244, Sept. 1996.
[24] K. Rahbar and J. P. Reilly, "Blind source separation of convolved sources by joint approximate diagonalization of cross-spectral density matrices," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Salt Lake City, UT, May 2001.
[25] E. C. Real, D. W. Tufts, and J. W. Cooley, "Two algorithms for fast approximate subspace tracking," *IEEE Trans. Signal Processing*, vol. 47, pp. 1936–1945, July 1999.
[26] V. U. Reddy, B. Egardt, and T. Kailath, "Least squares type algorithm for adaptive implementation of pisarenko's harmonic retrieval method," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP–30, pp. 399–403, June 1982.
[27] R. Schreiber, "Implementation of adaptive array algorithms," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 1038–1045, Oct. 1986.
[28] S. T. Smith, "Geometric optimization methods for adaptive filtering," Ph.D. dissertation, Harvard Univ., Cambridge, MA, 1993.
[29] P. Strobach, "Square-root QR inverse iteration for tracking the minor subspace," *IEEE Trans. Signal Processing*, vol. 48, pp. 2994–2999, Nov. 2000.
[30] L. Tong, R.-W. Liu, V. C. Soon, and Y.-F. Huang, "Indeterminacy and identifiability of blind identification," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 499–509, May 1991.
[31] C. Udrişte, *Convex Functions and Optimization Methods on Riemannian Manifolds*. Boston, MA: Kluwer, 1994.
[32] A.-J. Van Der Veen, "Algebraic methods for deterministic blind beamforming," *Proc. IEEE*, vol. 86, pp. 1987–2008, Oct. 1998.
[33] A. J. Van der Veen and A. Paulraj, "An analytical constant modulus algorithm," *IEEE Trans. Signal Processing*, vol. 44, pp. 1136–1155, May 1996.
[34] M. Wax and J. Sheinvald, "A least-squares approach to joint diagonalization," *IEEE Signal Processing Lett.*, vol. 4, pp. 52–53, Feb. 1997.
[35] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*. Oxford, U.K.: Oxford Science, 1988.
[36] J.-F. Yang and M. Kaveh, "Adaptive eigensubspace algorithms for direction or frequency estimation and tracking," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 241–251, Feb. 1988.
[37] X. Yang, T. K. Sarkar, and E. Arvas, "A survey of conjugate gradient algorithms for solution of extreme eigen-problems of a symmetric matrix," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 1550–1556, Oct. 1989.

Jonathan H. Manton (M'98) was born in April 1973. He received the B.S. degree in mathematics and the B.Eng. degree in electrical engineering, both in 1995, and his Ph.D. degree in 1998, all from the University of Melbourne, Parkville, Australia.

In 2000, he received the prestigious Research Fellow Award from the Australian Research Council, which he chose to take up at the University of Melbourne. His current research interests include precoding for wireless communications, discrete-time polynomial systems, and optimization on manifolds.

Dr. Manton is currently an Associate Editor on the Conference Editorial Board of the IEEE Control and Systems Society.