PCA by Determinant Optimisation has no Spurious Local Optima

RAPHAEL A. HAUSER, University of Oxford and Alan Turing Institute
ARMIN EFTEKHARI, Alan Turing Institute and University of Edinburgh
HEINRICH F. MATZINGER, Georgia Institute of Technology

Principal Component Analysis (PCA) finds the best linear representation for data and is an indispensable tool in many learning tasks. Classically, principal components of a dataset are interpreted as the directions that preserve most of its "energy", an interpretation that is theoretically underpinned by the celebrated Eckart-Young-Mirsky Theorem. There are yet other ways of interpreting PCA that are rarely exploited in practice, largely because it is not known how to reliably solve the corresponding non-convex optimisation programs. In this paper, we consider one such interpretation of principal components as the directions that preserve most of the "volume" of the dataset. Our main contribution is a theorem that shows that the corresponding non-convex program has no spurious local optima, and is therefore amenable to many convex solvers. We also confirm our findings numerically.

ACM Reference Format:
Raphael A. Hauser, Armin Eftekhari, and Heinrich F. Matzinger. 2018. PCA by Determinant Optimisation has no Spurious Local Optima. 1, 1 (May 2018), 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Let $A \in \mathbb{R}^{m\times n}$ be a data matrix whose rows correspond to m different items, and columns to n different features. For $p \le n$ and $X \in \mathbb{R}^{n\times p}$ with orthonormal columns, rows of $AX \in \mathbb{R}^{m\times p}$ correspond to the original data vectors projected onto range(X), the column span of X. In particular, the new data matrix AX has reduced dimension p, while the number m of items is unchanged. Successful dimensionality reduction is at the heart of classification, regression and other learning tasks that often suffer from the "curse of dimensionality", where having a small number of training samples in relation to the data dimension (namely, $m \ll n$) typically leads to overfitting [1].

Principal Component Analysis (PCA) is one of the oldest dimensionality reduction techniques going back to the work of Pearson [2] and Hotelling [3], which were motivated by the observation that often data is approximately located in a lower-dimensional subspace. PCA identifies this subspace by finding a suitable matrix X that retains in AX as much as possible of the energy of A. The optimal X is called the *loading matrix*. The columns of the loading matrix also yield important structural information about the data by identifying groups of variables that occur with jointly positive or jointly negative weights. In various application domains such groups of variables indicate patterns of functional dependencies, for example, genes that are jointly upregulated [4].

More formally, assume throughout this paper that the data matrix A is mean-centred, namely $\sum_{i=1}^m a_i = 0$ where $a_i \in \mathbb{R}^n$ is the i-th row of A. Consider $p \le n$ and let $\mathbb{R}_p^{n\times p}$ be the space of full column-rank $n \times p$ matrices and consider

Authors' addresses: Raphael A. Hauser, University of Oxford and Alan Turing Institute; Armin Eftekhari, Alan Turing Institute and University of Edinburgh; Heinrich F. Matzinger, Georgia Institute of Technology.

Manuscript submitted to ACM

the *trace inflation function*

$$\begin{aligned} g_t : \mathbb{R}_p^{n\times p} &\to \mathbb{R} \\ X &\mapsto \frac{\|AX\|_F^2}{\|X\|_F^2} = \frac{\mathrm{tr}(X^*A^*AX)}{\mathrm{tr}(X^*X)} \end{aligned} \tag{1}$$

and the program

$$\arg\max \left\{ g_t(X) : \ X \in \mathrm{St}(n,p) \right\}. \tag{2}$$

Above, $\|\cdot\|_F$ and $\mathrm{tr}(\cdot)$ return the Frobenius norm and trace of a matrix, respectively, and A^* is the transpose of matrix A. With $p \le n$, $\mathrm{St}(n,p)$ above denotes the the Stiefel manifold, namely the set of all $n \times p$ matrices with orthonormal columns. Program (2) performs PCA on the data matrix A, and it is a consequence of the Eckart-Young-Mirsky Theorem [5, 6] that a Stiefel matrix $X \in \mathrm{St}(n,p)$ is a global optimizer of Program (2) if and only if it is a p-leading right-singular factor $V_p \in \mathbb{R}^{n\times p}$ of A, namely if and only if V_p consists of the right singular vectors of A corresponding to its p largest singular values.

We remark that when $X \in \mathrm{St}(n,p)$, the denominator in the definition of $g_t(X)$ in (1) is constant and serves the purely cosmetic role of highlighting the similarities with an alternative program to compute principal components of A that we will propose below, a program that is unconstrained and does not require X to have orthogonal columns.

The interpretation of PCA as a tool for dimensionality reduction suggests that it should suffice to merely find a matrix $X \in \mathbb{R}_p^{n\times p}$ whose columns span the desired optimal subspace obtained by optimization over the Grassmannian $\mathrm{Gr}(n,p)$, defined as the set of p-dimensional subspaces of $\mathbb{R}^n$. However, program (2) does not fit that bill, for it is easy to see that $g_t(X)$ is not invariant under a change of basis for $\mathrm{range}(X)$.

2 MAIN RESULT AND PRIOR ART

In analogy to the function g_t in (1), we define a *volume inflation function* by

$$\begin{aligned} g_d : \mathbb{R}_p^{n\times p} &\to \mathbb{R} \\ X &\mapsto \frac{\det(X^*A^*AX)}{\det(X^*X)}, \end{aligned} \tag{3}$$

where det stands for determinant. Unlike g_t, note that g_d is invariant under a general change of basis. Indeed, consider $X \in \mathbb{R}_p^{n\times p}$, $\Theta \in \mathrm{GL}(p)$, and $Y = X\Theta$, where $\mathrm{GL}(p)$ is the general linear group, i.e., the set of nonsingular $p \times p$ matrices. Then we have that

$$g_d(Y) = \frac{\det(\Theta)^2 \det(X^*A^*AX)}{\det(\Theta)^2 \det(X^*X)} = g_d(X). \tag{4}$$

It follows that g_d is naturally defined on the Grassmannian $\mathrm{Gr}(n,p)$. In analogy to (2), consider the program

$$\arg\max \left\{ g_d(X) : \ X \in \mathbb{R}_p^{n\times p} \right\}. \tag{5}$$

Note that Program (5) is unconstrained because $\mathbb{R}_p^{n\times p}$ is an open subset of $\mathbb{R}^{n\times p}$ with nonempty relative interior, and that, having chosen $X \in \mathbb{R}_p^{n\times p}$, we explicitly exclude degenerate points.

Program (5) is a good model for dimensionality reduction because X^*A^*AX is the sample covariance of the dimensionality-reduced data matrix AX, and $\det(X^*A^*AX)^{\frac{1}{2}}$ is the volume of the smallest box that fits around the ellipsoid $\{y \in \mathbb{R}^p : \ y^*(X^*A^*AX)^{-1}y \le 1.96^2\}$, which closely approximates the smallest ellipsoid that encloses all of the projected (dimensionality-reduced) data points $(a_iX)^*$, where a_i is the feature vector of the i-th item, that is the

i-th row of matrix A. Thus, Program (5) may be thought of as maximizing the volume of point cloud of dimensionality reduced items. Program (2) in contrast maximizes the length of the diagonal of the aforementioned box. Thus, Programs (2) and (5) have different geometric justifications. Our main result is an extension of the Eckart-Young-Mirsky Theorem to Program (5), showing that the latter yields the same notion of PCA of A as Program (2):

Theorem 2.1. *The following statements hold true:*

i) *Any p-leading right-singular factor $V_p \in \mathbb{R}^{n\times p}$ of A is a global maximizer of Program (5).*
ii) *For any global maximizer $\widetilde{X}$ of Program (5), there exists a p-leading right-singular factor V_p of A such that* $\text{range}(\tilde{X}) = \text{range}(V_p)$.
iii) *Program (5) does not have any spurious local maximisers, namely any local maximizer of Program (5) is also a global maximizer.*

Due to the characterization of X in part ii), the singular value decomposition of $A\widetilde{X} = \widetilde{U}\widetilde{S}\widetilde{V}^*$, which can be computed in merely $\mathcal{O}(mp^2)$ time, yields $V_p = X\widetilde{V}$ as a p-leading right-singular factor of A, the diagonal coefficients of $\widetilde{S}$ as the p leading singular values of A, and the first p columns of $\widetilde{U}$ as the corresponding p-leading left singular factor of A. Parts i) and ii) of Theorem 2.1 are not new, as they can be easily proven using interlacing properties of singular values, see [7, Corollary 3.2], or via the Cauchy-Binet Formula [8]. See also [9, Lemma 4.6] for a related result that shows the stationarity of right singular factors of A for Program (5). What is new in our result, however, is Part iii), which is crucial in rendering the computation of a PCA via Program (5) practical: this property guarantees that any locally convergent descent algorithm from the standard literature (gradient descent, coordinate descent, block coordinate descent, trust-region methods, line search descent methods, cubic regularisation methods, etc.) applied to the negative objective function of (5) are automatically globally convergent to a correct PCA irrespective of the starting point used. Empirical confirmation of this property is provided in our numerical section where we compare the convergence of gradient descent, line-search gradient descent and accelerated gradient descent algorithms applied to Programs (2) and (5). To prove Property iii), we develop an exact differential geometric characterization of all KKT points of Program (5), including the case with coalescing singular values, and this also yields a novel, purely geometric proof of Parts i) and ii) of Theorem 2.1. In contrast to earlier proofs of these parts, our analysis does not depend on the columns of X being mutually orthogonal.

A string of conceptually related papers has appeared in the recent literature, all of which show the nonexistence of spurious local optima for different problems for the purposes of understanding the geometry and performance of iterative descent algorithms on nonconvex optimization problems [10–15]. Our theory fits nicely into this growing literature, although it is based on different techniques. Among this line of research papers, the work of [16] is most closely related to ours, as they consider functions $X \mapsto \phi(X^*BX)$ with B positive semidefinite and ϕ convex in matrix completion [17–19], while our objective function $X \mapsto \log(g_d(X))$ is a difference of two such functions.

3 TECHNICAL DETAILS

Before proving Theorem 2.1, a word on the notation is in order. We denote the singular value decomposition of a matrix $A \in \mathbb{R}^{m\times n}$ of rank k by $A = USV^*$, where $U \in \text{Orth}(m)$, $V \in \text{Orth}(n)$, and $S \in \mathbb{R}^{m\times n}$ is a diagonal matrix with the ordered singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$ of A on the diagonal and all other values equal to zero. If $k < n$ we complete the spectrum by setting $\sigma_{k+1} = \cdots = \sigma_n = 0$. The *thin SVD* of A will be denoted by $A = U_kS_kV_k^*$, where $U_k \in \text{St}(m,k)$ consists of the first k columns of U, S_k of the top left $k \times k$ block of S, and $V_k \in \text{St}(n,k)$ of the first k columns of V. For any index set $J \subseteq \{1,\ldots,n\}$, we write V_J for the matrix composed of the columns $\{v_j : j \in J\}$, and

S_J for the submatrix $[s_{ij}]$ of S with indices $j \in J$, $i \in J \cap \{1,\dots,m\}$, and U_J for the matrix consisting of the columns $\{u_i : i \in J \cap \{1,\dots,m\}$, leaving the original ordering intact in all cases. We also write $A_J = U_J S_J V_J^*$ for the low-rank approximation of A corresponding to singular vectors with index in J. In the special case $J = \{1,\dots,p\}$ that corresponds to the p-leading part SVD of A we use the shorthand notation U_p, S_p, V_p and A_p for U_J, S_J, V_J and A_J respectively.

Let us now turn to the proof of Theorem 2.1. Assuming $\mathrm{rank}(A) \geq p$, so that there exist X for which $\det(X^*A^*AX) \neq 0$, we may instead maximize the function $f_d(X) := \ln g_d(X)$ and reformulate Problem (5) in the following equivalent form,

$$X^* = \arg\max \left\{ f_d(X) : \ X \in \mathbb{R}_p^{n\times p} \right\} \tag{6}$$

which is unconstrained, because $\mathbb{R}_p^{n\times p}$ is an open full-dimensional domain in $\mathbb{R}^{n\times p}$. The stationary points of this model are therefore characterized by $\nabla f_d(X) = 0$. Using a Taylor expansion, it is easy to see that the gradient of f_d is given by

$$\nabla f_d(X) = 2X(X^*X)^{-1} - 2A^*AX(X^*A^*AX)^{-1} \tag{7}$$

and behaves as follows under a change of basis $Y = X\Theta$, where $\Theta \in \mathrm{GL}(p)$,

$$\nabla f_d(Y) = \nabla f_d(X)\Theta^{-*}. \tag{8}$$

Lemma 3.1. *Let $A \in \mathbb{R}^{m\times n}$ be of rank $k \geq p$, $X \in \mathbb{R}_p^{n\times p}$ and $AX \in \mathbb{R}_p^{m\times p}$. Then $\nabla f_d(X) = 0$ if and only if there exists a set of indices $J = \{i_1,\dots,i_\ell\} \subseteq \{1,\dots,k\}$ such that $|\{\sigma_{i_1},\dots,\sigma_{i_\ell}\}| \leq p$ (not counting multiplicities), and* $\mathrm{range}(X) \subseteq \mathrm{range}(V_J)$.

Proof: By virtue of Equation (8), $\nabla f_d(X) = 0$ if and only if $\nabla f_d(X\Theta) = 0$ for any $\Theta \in \mathrm{GL}(p)$, and since it is also the case that $\mathrm{range}(X) \subseteq \mathrm{range}(V_J)$ if and only if $\mathrm{range}(X\Theta) \subseteq \mathrm{range}(V_J)$, we may in fact assume without loss of generality that $X \in \mathrm{St}(n,p)$, so that $X^*X = \mathrm{I}_p$. Let $A = U_k S_k V_k^*$ be the thin SVD of A, which is an exact factorization because $\mathrm{rank}(A) = k$, so that $\sigma_1 \geq \cdots \geq \sigma_k > 0 = \sigma_{k+1} = \cdots = \sigma_n$. By (7), we then have

$$\begin{aligned} \nabla f_d(X) \quad &= \quad 0 \ \Leftrightarrow\ X = A^*AX(X^*A^*AX)^{-1} \ \Leftrightarrow\ (XX^*)V_kS_k^2V_k^*X = V_kS_k^2V_k^*X \ \Leftrightarrow\ \mathrm{range}(V_kS_k^2V_k^*X) \subseteq \mathrm{range}(X) \\ &\Leftrightarrow\quad \mathrm{range}(V_kS_k^2V_k^*X) = \mathrm{range}(X), \qquad (9) \\ &\Leftrightarrow\quad \exists\, Z \in \mathrm{GL}(p) \text{ s.t.} V_kS_k^2V_k^*X = XZ \ \Leftrightarrow\ S_k^2V_k^*X = V_k^*XZ \ \Leftrightarrow\ (X^*V_k)S_k^2 = Z^*(X^*V_k), \end{aligned}$$

where Equivalence (9) follows from $\mathrm{rank}(V_kS_k^2V_k^*X) = p$. Denoting the column vectors of X^*V_k by $(w_1,\dots,w_k)$, we have $w_j \in \mathbb{R}^p$ with

$$\sigma_j^2 w_j = Z^* w_j, \quad (j = 1,\dots,k). \tag{10}$$

Since Z^* has at most p different eigenvalues, it must be the case that $|\{\sigma_j : w_j \neq 0\}| \leq p$ (not counting multiplicities), and then $AX \in \mathbb{R}_p^{m\times p}$ implies that we have $X = V_kV_k^*X = V_JV_J^*X$, where $J = \{j : w_j \neq 0\}$ is the index set whose existence is claimed in the lemma. □

Lemma 3.2. *Let $A \in \mathbb{R}^{m\times n}$ be of rank $k \geq p$, $X \in \mathbb{R}_p^{n\times p}$ such that $AX \in \mathbb{R}_p^{m\times p}$ and $\nabla f_d(X) = 0$, and let $[w_1,\dots,w_k] = X^*V$. If $\{j_1,\dots,j_s\} \subseteq \{1,\dots,k\}$ is an index set such that for any strict subset $J \subset \{j_1,\dots,j_s\}$ the vectors $\{w_j : j \in J\}$ are linearly independent but the set $\{w_{j_i} : i = 1,\dots,s\}$ is linearly dependent, then $\sigma_{j_1} = \cdots = \sigma_{j_s}$.*

Proof: By the assumptions of the theorem, there exist unique scalars $\lambda_2,\dots,\lambda_s$, all nonzero, such that $w_{j_1} = \sum_{i=2}^s \lambda_i w_{j_i}$, and multiplying this equation with the matrix Z^* constructed in the proof of Lemma 3.1 yields $\sigma_{j_1}^2 w_{j_1} = \sum_{i=2}^s \sigma_{j_i}^2 \lambda_i w_{j_i}$. By the uniqueness and non-nullity of the scalars λ_i, and by strict positivity of σ_{j_1}, comparison of the two equations yields $\sigma_{j_i} = \sigma_{j_1}$ $(i = 2,\dots,s)$. □

Lemma 3.2 shows that proper linear dependence among the vectors w_j can only occur within subsets of vectors that correspond to the same eigenvalue of the matrix Z^* from the proof of Lemma 3.1. We also note that since $Z^* \in \mathrm{GL}(p)$, it has at most p linearly independent eigenvectors. This motivates the following definition, in which $A \in \mathbb{R}^{m\times n}$ is a matrix of rank $k \geq p$, $X \in \mathbb{R}_p^{n\times p}$ such that $AX \in \mathbb{R}_p^{m\times p}$. Let $[w_1, \ldots, w_k] := X^*V_k$, and $J := \{j : w_j \neq 0\}$. Further, let $\varsigma_1 > \cdots > \varsigma_\rho > 0$ be an enumeration of the nonzero singular values $\sigma_1 \geq \cdots \geq \sigma_k$ of A without repetition. For any index set $\mathcal{J} \subseteq \{1, \ldots, k\}$ let us write $\mathcal{I}(\mathcal{J}) = \{i : \varsigma_i \in \{\sigma_j : j \in J\}\}$, and conversely, for any index set $\mathcal{I} \subseteq \{1, \ldots, \rho\}$ we write $\mathcal{J}(\mathcal{I}) = \{j : \sigma_j \in \{\xi_i : i \in \mathcal{I}\}\}$. For $i = 1, \ldots, \rho$, we call $m(i) := |\mathcal{J}(\{i\})|$ is the multiplicity of ς_i in V, $m_\lambda(i,p) := \max\left(0, p - \sum_{\ell=1}^{i-1} m(\ell)\right)$ the p-leading multiplicity of ς_i in V, $m_\tau(i,p) := \max\left(0, p - \sum_{\ell=i+1}^{k} m(\ell)\right)$ the p-trailing multiplicity of ς_i in V, and $m(X,i) := \mathrm{rank}\{w_j : j \in J_i\}$ the multiplicity of ς_i in X, where $J_i = J \cap \mathcal{J}(\{i\})$. For an intuitive interpretation, note that $m(i)$ is the number of right-singular vectors of A associated with the singular value σ_i, $m_\lambda(i,p)$ the number of column vectors in any p-leading right-singular factor V_p of A associated with the singular value σ_i, and $m(X,i) = \dim(\mathrm{range}(X) \cap \mathrm{V}_i)$ where $\mathrm{V}_i := \mathrm{range}\{v_j : \sigma_j = \varsigma_i\}$ is the right-singular space of A associated with the singular value ς_i.

Corollary 3.3. *X is a stationary point of Program (6) if and only if $m(X,i) = \dim(\Pi_{\mathrm{V}_i}\mathrm{range}(X))$ for $i = 1, \ldots, \rho$, where Π_{V_i} denotes the orthogonal projection into the right-singular space of A associated with the singular value ς_i.*

Proof: Note that since $\mathrm{range}(X) \cap \mathrm{V}_i \subseteq \Pi_{\mathrm{V}_i}\mathrm{range}(X)$, we have

$$m(X,i) \leq \dim(\Pi_{\mathrm{V}_i}\mathrm{range}(X)) \tag{11}$$

for all $X \in \mathbb{R}^{n\times p}$ and $i = 1, \ldots, \rho$. Thus, the corollary says in fact that X is a stationary point of Program (6) if and only if $\mathrm{range}(X) \cap \mathrm{V}_i = \Pi_{\mathrm{V}_i}\mathrm{range}(X)$ for all i. By Lemma 3.1, $\nabla f_d(X) = 0$ if and only if $\mathrm{range}(X) \subseteq \mathrm{range}(V_J)$. Assuming that the latter condition is satisfied, we have $\mathrm{range}(X) \cap \mathrm{V}_i = \{0\} = \Pi_{\mathrm{V}_i}\mathrm{range}(X)$ for all $i \notin \mathcal{I}(J)$, while the condition also implies step (13) in the following sequence of equalities,

$$\begin{aligned}
\sum_{i=1}^{\rho} m(X,i) &= \sum_{i=1}^{\rho} \mathrm{rank}(X^*V_{J_i}) \;=\; \mathrm{rank}(X^*V_J) && (12)\\
&= \mathrm{rank}(X^*V_k) && (13)\\
&= p, && (14)
\end{aligned}$$

where (12) follows from Lemma 3.2 and (14) from $AX \in \mathbb{R}_p^{n\times p}$. But this sequence of equalities can only hold if (11) holds as an equality for all ($i = 1, \ldots, \rho$), and hence, $\mathrm{range}(X) \cap \mathrm{V}_i = \Pi_{\mathrm{V}_i}\mathrm{range}(X)$ for all $i \in \mathcal{I}(J \cap \mathcal{J}(\{1, \ldots, \rho\}))$. Conversely, if $\mathrm{range}(X) \cap \mathrm{V}_i = \Pi_{\mathrm{V}_i}\mathrm{range}(X)$ for all i, then

$$\begin{aligned}
\mathrm{range}(X) &= \mathrm{range}(VV^*X) \;=\; \mathrm{range}(V_kV_k^*X) && (15)\\
&= \mathrm{range}(V_JV_J^*X) && (16)\\
&\subseteq \mathrm{range}(V_J),
\end{aligned}$$

as claimed, where (15) follows from $AX \in \mathbb{R}_p^{n\times p}$, and (16) from the definition of J, by which $X^*v_j = 0$ for $j \notin J$. □

Theorem 3.4. *If X is a stationary point of Program (6), then*

i) $\sum_{i=1}^{\rho} m(X,i) = p$,
ii) $f_d(X) = \sum_{i=1}^{\rho} 2m(X,i)\ln\varsigma_i$.

Proof: In assuming that X is a stationary point of Program (6), we make the assumption that $A \in \mathbb{R}^{m\times n}$ is a matrix of rank $k \geq p$, and $X \in \mathbb{R}_p^{n\times p}$ such that $AX \in \mathbb{R}_p^{m\times p}$ and $\nabla f_d(X) = 0$, so that all of the above lemmas apply. Part i) is then an equivalent reformulation of Corollary 3.3, by virtue of Equation (14). For Part ii), we may assume without loss of generality that $X^*X = \mathrm{I}_p$, for the same reasons as in the proof of Lemma 3.1, so that $f_d(X) = \ln\det(X^*V_J S_J^2 V_J^* X)$. Then there exists $\Theta \in \mathrm{Orth}(p)$ such that $X\Theta = [Y_1, \ldots, Y_\rho]$, where some of the blocks may have zero columns, and where each block $Y_i = [y_1^i, \ldots, y_{m(X,i)}^i] \in \mathrm{St}\,(n, m(X,i))$ satisfies $\mathrm{range}(Y_i) \subseteq \mathrm{range}(V_{J_i})$ so that $Y_i^* V_{J_j} = 0$ $(j \neq i)$ and $V_{J_i} V_{J_i}^* Y_i = Y_i$. We have

$$\begin{aligned} f_d(X) &= \ln\det\left(\Theta^* X^* V_J S_J^2 V_J^* X \Theta\right) = \ln\det\left(\left[\oplus_{i=1}^\rho Y_i^* V_{J_i}\right]\left[\oplus_{i=1}^\rho \varsigma_i^2 \mathrm{I}_{m(X,i)}\right]\cdot\left[\oplus_{i=1}^\rho V_{J_i}^* Y_i\right]\right) \\ &= \ln\prod_{i=1}^\rho \det\left(Y_i^* V_{J_i}\left(\varsigma_i^2 \mathrm{I}_{m(X,i)}\right) V_{J_i}^* Y_i\right) = \ln\prod_{i=1}^\rho \varsigma_i^{2m(X,i)} \det\left[\left(Y_i^* V_{J_i} V_{J_i}^*\right)\left(V_{J_i}^* V_{J_i} Y_i\right)\right] = \sum_{i=1}^\rho 2m(X,i)\ln\varsigma_i, \end{aligned}$$

where $\oplus$ denotes the composition of a block-diagonal matrix out of its constituent blocks. □

Theorem 3.5. *Let X be a stationary point of Program (6). The following hold true:*

i) *X is a global maximizer of f_d if and only if $m(X,i) = m_\lambda(i,p)$ for all $(i = 1, \ldots, \rho)$.*
ii) *When $\mathrm{rank}(A) = n$, then X is a global minimizer of f_d if and only if $m(X,i) = m_\tau(i,p)$ for all $(i = 1, \ldots, \rho)$.*
iii) *In all other cases X is a saddle point of f_d.*

Proof: It suffices to prove Claim iii), as Claims i) and ii) will then follow immediately from Theorem 3.4.iii) and the fact that f_d is C^∞ on $\{X \in \mathbb{R}^{n\times p} : \mathrm{rank}(AX) = p\}$. Using the same transformation $Y = X\Theta$ as in the proof of Theorem 3.4, we may replace X by Y and assume without loss of generality that X consists of p different columns of V, ordered by non-increasing corresponding singular values, $X = [v_{j_1}, \ldots, v_{j_p}]$, $\sigma_{j_1} \geq \cdots \geq \sigma_{j_p}$, and since we are neither in the case of Claims i) or ii), there exist $\mu, \eta \in \{1, \ldots, k\} \setminus \{j_1, \ldots, j_p\}$ such that $\sigma_\mu > \sigma_{j_p}$ and $\sigma_\eta < \sigma_{j_1}$. Now let us write $c = \cos\theta$, $s = \sin\theta$, and consider the matrix $X(\theta) = \begin{bmatrix} v_\eta & v_{j_1} & \ldots & v_{j_p}\end{bmatrix}\begin{bmatrix}\begin{bmatrix} c & s\\ -s & c\end{bmatrix} & 0\\ 0 & \mathrm{I}_{p-1}\end{bmatrix}\begin{bmatrix} 0\\ \mathrm{I}_p\end{bmatrix}$. We have $X(0) = X$, $X(\theta) \in \mathrm{St}(n,p)$ for all θ, and

$$\begin{aligned} &\det\left(X(\theta)^* A^* A X(\theta)\right) = \det\left(X(\theta)^* V S^2 V^* X(\theta)\right) \\ &= \det\left(\begin{bmatrix} 0 & \mathrm{I}_p\end{bmatrix}\begin{bmatrix}\begin{bmatrix} c & -s\\ s & c\end{bmatrix} & 0\\ 0 & \mathrm{I}_{p-1}\end{bmatrix}\begin{bmatrix}\sigma_\eta^2 & & & 0\\ & \sigma_{j_1}^2 & & \\ & & \ddots & \\ 0 & & & \sigma_{j_p}^2\end{bmatrix}\begin{bmatrix}\begin{bmatrix} c & s\\ -s & c\end{bmatrix} & 0\\ 0 & \mathrm{I}_{p-1}\end{bmatrix}\begin{bmatrix} 0\\ \mathrm{I}_p\end{bmatrix}\right) \\ &= (\sigma_\eta^2 s^2 + \sigma_{j_1}^2 c^2)\sigma_{j_2}^2 \ldots \sigma_{j_p}^2. \end{aligned}$$

Thus, writing $\phi(\theta) = f_d(X(\theta))$, we find $\phi''(\theta) = -\left(\frac{2\sigma_\eta^2 \sin\theta\cos\theta - 2\sigma_{j_1}^2 \cos\theta\sin\theta}{\sigma_\eta^2 \sin^2\theta + \sigma_{j_1}^2\cos^2\theta}\right)^2 + \frac{2(\sigma_\eta^2 - \sigma_{j_1}^2)(\cos^2\theta - \sin^2\theta)}{\sigma_\eta^2\sin^2\theta + \sigma_{j_1}^2\cos^2\theta}$, and hence, $\phi''(0) = 2(\sigma_\eta^2 - \sigma_{j_1}^2)/\sigma_{j_1}^2 < 0$. Analogously, using $X(\theta) = \begin{bmatrix} v_{j_1} & \ldots & v_{j_p} & v_\mu\end{bmatrix}\begin{bmatrix}\mathrm{I}_{p-1} & 0\\ 0 & \begin{bmatrix} c & s\\ -s & c\end{bmatrix}\end{bmatrix}\begin{bmatrix}\mathrm{I}_p\\ 0\end{bmatrix}$ yields $\phi''(0) = 2(\sigma_\mu^2 - \sigma_{j_p}^2)/\sigma_{j_p}^2 > 0$. The Hessian of f_d at X has thus both directions of positive and negative curvature, and X is a saddle point, as claimed. □

Theorem 2.1 is now an easy corollary of Theorem 3.5:

Proof of Theorem 2.1. Part i) follows from the trivial fact that $m(V_p, i) = m_\lambda(i, p)$, $(i = 1, \ldots, \rho)$ and the "if part" of Theorem 3.5.i), whereas Part ii) follows from Corollary 3.3 and the "only if part" of Theorem 3.5.i). □

4 EXPERIMENTAL VERIFICATION OF RESULTS

The theory of Section 3 gives deep insight into the geometry of low-rank factorizations. Part iii) of Theorem 2.1 furthermore shows that Model (5), or equivalently, (6), is of practical use in conjunction with any standard descent algorithm that is known to be locally convergent, as the Theorem then implies that all such algorithms are also globally convergent irrespective of the starting point. In this section we conduct a series of experiments that give empirical confirmation of this finding. For the purposes of these experiments we chose but a few of the simplest and easiest to implement algorithms: steepest ascent with constant step size, steepest ascent with variable step size, and steepest ascent with Scieur-d'Aspremont-Bach acceleration. In order to get a qualitative understanding of the behavior of Model (6), we compare it against numerical results obtained by applying the same algorithms to model

$$\arg\max \{ f_t(X) := \log g_t(X) : \ X \in \mathrm{St}(n, p) \}, \tag{17}$$

which is often used in the sparse PCA context [20]. As we will see below, determinant maximization has significant advantages over trace maximization. It is important to keep in mind that we are comparing two optimization *models*, rather than *algorithms*, as any number of known optimization algorithms could have been applied instead of the steepest ascent variants we discuss, and a comprehensive comparative study of algorithms is beyond the scope of this paper. However, based on our extensive numerical experiments we believe that the qualitative differences between the determinant and trace models are robust under the choice of algorithm for their solution. Recall that the steepest ascent algorithm applied to the unconstrained maximization of a C^1 function $f : \mathbb{R}_p^{n\times p} \to \mathbb{R}$ is an iterative procedure with updating rule

$$X_{i+1} = X_i + \eta \nabla f(X_i), \tag{18}$$

where $\eta > 0$ is a steplength multiplier. When η is chosen constant, a good choice is $\eta = 1/\Lambda$, where Λ is a global Lipschitz constant of ∇f if such a constant is know. If Λ is unknown, it has to be estimated numerically. The case where η is chosen as the exact minimizer of the one-dimensional optimization problem $\min_\eta f(X_i + \eta \nabla f(X_i))$ is called steepest ascent with exact line-search. Since this is computationally expensive, practical line-search methods have been developed to find an approximately optimal η via computationally inexpensive interval searches. One of the best standard choices of such a method is a line-search with *Wolfe Conditions*. A standard stopping criterion for unconstrained iterative optimization methods is $\|\nabla f(X)\| < \epsilon$, that is, the algorithm is stopped when the objective gradient falls below a chosen threshold. All of the above is standard textbook material, see e.g. [21]. For ease of reference, we call any algorithm applied to Model (6) as a *determinant flow*, while referring to any algorithm applied to Program (17) as a *trace flow*. Note that while (6) is an unconstrained optimization problem, (17) is constrained by the requirement that X be a Stiefel matrix of size $n \times p$. To get around this problem, [20] apply a standard steepest ascent update (18) followed by re-orthogonalization of the columns of X, which is obtained by applying a thin QR factorization to X [22] and replacing X by Q. This method is guaranteed to work generically, and we adopted it for all implementations of determinant and trace flows, in order to make running times more comparable. In our experiments we observed that numerical estimates of the local Lipschitz constant of $\nabla f_d(X)$ was nearly constant over the entire domain of the determinant flow, hence it appears that numerically this model is globally Lipschitz constant and amenable to steepest

ascent with constant step size, an algorithm we denote by `det-flow`. In contrast, the trace flow numerically rotates the initial iterates X very fast for most starting points, resulting in large local Lipschitz constants for $\nabla f_t(X)$ initially, and becoming small asymptotically. The use of a constant step size η is not appropriate for this model, as it would force a choice that renders all but the initial few step sizes too short. We therefore implemented the trace flow with a practical line search based on Wolfe Conditions [21] and refer to this algorithm as `trace-flow`. To give the determinant flow the same chance at running with a variable step size, we implemented several other variants: `det-LS` is the steepest ascent method with the same practical line-search, but applied to the determinant flow model rather than the trace flow. A third variant, `acc-det-flow` is an implementation of the acceleration scheme of [23] applied to the determinant flow. This scheme computes $k+1$ consecutive updates (18) and then uses the gradients $\nabla f_d(X_k)$ to compute an improved single update direction Δ_i applied to X_i,

$$X_{i+1} = X_i + \xi\Delta_i.$$

We used constant values $\eta = 1/\Lambda$ for the $k+1$ updates (18), where Λ is a numerical estimate of the Lipschitz constant of ∇f_d, as in algorithm `det-flow`. We tested three variants of this method: `acc-det-flow-k=4` is a basic version with $\xi = 1$ throughout, `acc-det-LS` is a version with $k = 4$ and ξ chosen by a practical line search based on Wolfe Conditions, and `acc-det-BT` is a version with $k = 4$ and Δ_i computed via a backtracking method proposed in [23]. The technical details of the acceleration method goes beyond the limited space available to this paper, but our implementation is fully reproducible by referring to [23]. Furthermore, we will make our code publicly available upon acceptance of our paper. We compared `trace-flow`, `det-flow`and all its variants on the following two random matrix models: Left and right singular vectors of matrices were drawn from the uniform (Haar) distribution on the corresponding spaces. For example, the left singular vectors of a random 2000×3000 matrix were drawn from the uniform distribution on the orthogonal group Orth(2000), whereas its right singular vectors were drawn from the uniform distribution on the Stiefel manifold St(3000, 2000). For the singular values, the first model uses an "easy" spectrum that has a fairly flat scree plot. In the second model, the singular spectrum exhibits a "hockey-stick" shape. We computed $p = 15$ right-leading singular vectors of matrices generated from these three models and with different matrix dimensions. The error is expressed as the principal angle [22] $\arcsin(\|V_p V_p^* \widehat{X} - \widehat{X}\|)$ between $\mathrm{range}(\widehat{X})$ and the ground truth $\mathrm{range}(V_p)$. Here, $\|\cdot\|$ is the spectral norm, $\widehat{X}$ is the output of the algorithm invoked, and V_p is a p-leading right-singular factor of the random data matrix at hand. Error of all algorithms versus time are plotted in Figures 1 and 2. Our code was implemented in MATLAB (Release R2015a), and all numerical experiments were carried out on a MacBook Air equipped with a 1.4 GHz Intel Core i5 processor with 8 GB of memory. While the purpose of this section is primarily to illustrate that iterative algorithms applied to the determinant flow model (6) have the regular convergence behavior predicted by the theory of Section 3 and to give a qualitative comparison with Program (17), it is also interesting for comparison to run the compiled Fortran code of the LAPACK implementation of Lanczos' Method on the same input data. The Lanczos Method (see e.g. [22]) was the algorithm of choice for the computation of p-leading part SVDs for many decades, although it has now been superseeded. An asterisk is plotted if all p right-leading singular vectors were computed to the accuracy specified on the vertical axis in the time shown on the horizontal axis. In some of the experiments the error is shown as 10^0, and this means that not all p vectors were computed to the required accuracy. This may not be reflective of the maximum error among the p vectors, as the Lanczos code only outputs the vectors it could compute to the required accuracy. In contrast, the error of the outputs of our own implementation are reflective of the maximum error, namely the principal angle $\arcsin(\|V_p V_p^* \widehat{X} - \widehat{X}\|)$ described earlier. On the difficult spectra the `det-flow` algorithm significantly and consistently outperformed the `trace-flow`, the latter being competitive only for random matrices with an easy spectrum. We

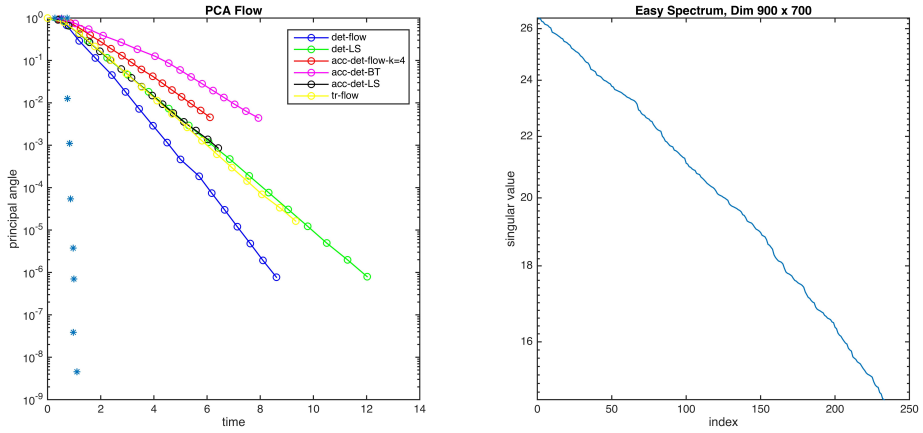

Fig. 1. Computing the principal components of a 900×700 random matrix an "easy" spectrum using `trace-flow`, `det-flow`, and its variants. The spectrum is displayed on the right and the error versus time for various algorithms are plotted on the left.

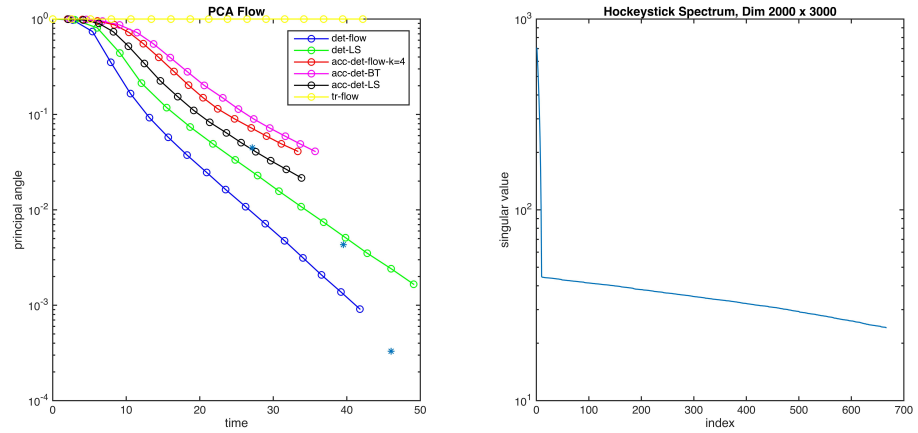

Fig. 2. Similar to Figure 1 but for a 2000×3000 random matrix with a "hockey-stick" spectrum.

remark that although `det-LS`, `acc-det-flow-k=4`, `acc-det-BT` and `acc-det-LS` all have lower iteration complexity than `det-flow`, that is, these algorithms converge to a given target accuracy in fewer iterations, the cost per iteration is much higher due to the evaluation of a variable step size and/or the accelerated update direction Δ_i, so that the overall clock time of these algorithms is slower than `det-flow`. In the case of the trace flow however, the line search is very cheap, because evaluating a trace takes only $O(p)$ time. Algorithm `tr-flow` has by far the lowest cost per iteration, but its iteration complexity is much higher than in all of the determinant flow variants.

REFERENCES

[1] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer Series in Statistics. Springer New York, 2013.

[2] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572, 1901.

[3] H. Hotelling. Relations between two sets of variates. *Biometrika*, (28):321–377, 1936.

[4] Orly Alter and Gene Golub. Singular value decomposition of genome-scale mrna lengths distribution reveals asymmetry in rna gel electrophoresis band broadening. *PNAS*, 103(32):11828–11833, 2006.

[5] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.

[6] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *Quart. J. Math. Oxford*, pages 1156–1159, 1966.

[7] R.A. Horn and C.R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, 1994.

[8] Samuel Karlin and Yosef Rinott. A generalized cauchy-binet formula and applications to total positivity and majorization. *Journal of multivariate analysis*, 27(1):284–299, 1988.

[9] Mark D Plumbley. Lyapunov functions for convergence of principal component algorithms. *Neural Networks*, 8(1):11–23, 1995.

[10] R. Ge, J Lee, and T. Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016.

[11] Ju Sun, Qing Qu, and John Wright. When are nonconvex problems not scary? *arXiv preprint arXiv:1510.06096*, 2015.

[12] R. Ge, F. Huang, Ch. Jin, and Y. Yuan. Escaping from saddle points – online stochastic gradient for tensor decomposition. arXiv:1503.02101, 2015.

[13] J. Sun, Q. Qu, and J. Wright. When are nonconvex problems not scary? arXiv:1510.06096, 2015.

[14] S. Bhojanapalli, B. Neyshabur, and N. Srebro. Global optimality of local search for low rank matrix recovery. ArXiv e-prints, 2016.

[15] Armin Eftekhari and Michael B Wakin. Greed is super: A fast algorithm for super-resolution. *arXiv preprint arXiv:1511.03385*, 2015.

[16] Q. Li and G. Tang. The nonconvex geometry of low-rank matrix optimizations with general objective functions. arXiv:1611.03060v1 [cs.IT], 2016.

[17] Mark A Davenport and Justin Romberg. An overview of low-rank matrix recovery from incomplete observations. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):608–622, 2016.

[18] Armin Eftekhari, Michael B Wakin, and Rachel A Ward. MC 2: A two-phase algorithm for leveraged matrix completion. *arXiv preprint arXiv:1609.01795*, 2016.

[19] Armin Eftekhari, Dehui Yang, and Michael B Wakin. Weighted matrix completion and recovery with prior subspace information. *IEEE Transactions on Information Theory*, 2018.

[20] L. Cambier and P.-A. Absil. Robust low-rank matrix completion by Riemannian optimization. *SIAM J. Sci. Comput.*, 38(5):440–460, 2016.

[21] J. Nocedal and S. Wright. *Numerical Optimization.* Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.

[22] G. Golub and Ch. Van Loan. *Matrix Computations.* The Johns Hopkins University Press, 1996.

[23] Damien Scieur, Alexandre d'Aspremont, and Francis Bach. Regularized nonlinear acceleration. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 712–720. Curran Associates, Inc., 2016.