RANK-DEFICIENT QUADRATIC-FORM MAXIMIZATION OVER $M$-PHASE ALPHABET: POLYNOMIAL-COMPLEXITY SOLVABILITY AND ALGORITHMIC DEVELOPMENTS* 

Anastasios T. Kyrillidis
School of Computer & Communication Sciences
EPFL
CH-1015 Lausanne, Switzerland
E-mail: anastasios.kyrillidis@epfl.ch

George N. Karystinos
Electronic & Computer Engineering Department
Technical University of Crete
Chania, 73100, Greece
E-mail: karystinos@telecom.tuc.gr

ABSTRACT
The maximization of a positive (semi)definite complex quadratic form over a finite alphabet is $NP$-hard and achieved through exhaustive search when the form has full rank. However, if the form is rank-deficient, the optimal solution can be computed with only polynomial complexity in the length $N$ of the maximizing vector. In this work, we consider the general case of a rank-$D$ positive (semi)definite complex quadratic form and develop a method that maximizes the form with respect to a $M$-phase vector with polynomial complexity. The proposed method efficiently reduces the size of the feasible set from exponential to polynomial. We also develop an algorithm that constructs the polynomial-size candidate set in polynomial time and observe that it is fully parallelizable and rank-scalable.

1. INTRODUCTION
Unconstrained complex quadratic maximization over a finite alphabet captures many problems that are of interest to the communications and signal processing community. Interestingly, it has been recently proven that the maximization of a quadratic form with a binary vector argument is no longer $NP$-hard if the rank of the form is not a function of the problem size [1] and can be computed optimally and efficiently in polynomial time through a variety of computational geometry (CG) algorithms, such as the incremental algorithm for cell enumeration in arrangements [2] and the reverse search method [3],[4]. However, it should be noted that, although the incremental algorithm is applicable to $M$-phase vector-argument alphabets with $M > 4$, it is not known whether it has practical importance in higher dimensions due to lack of parallelizability and memory management inefficiency. On the other hand, the reverse search method is highly parallelizable and speed/memory efficient but has been applied, until today, only for $M = 2, 4$. From a communications-systems-optimization perspective, developments have led to algorithms that optimally solve the problem of maximization of a rank-$D$ quadratic form over the $M$-phase alphabet for $D = 1$ for any $M$ [5].

*This work was supported in part by Alexandros S. Onassis Foundation.

2. PROBLEM STATEMENT
We consider the computation of the $M$-phase vector that maximizes the quadratic form

$$s_{\text{opt}} \triangleq \arg\max_{s \in A^N_M} s^H V V^H s = \arg\max_{s \in A^N_M} \|V^H s\|^2$$  

(1)

where $V \in \mathbb{C}^{N \times D}$ is a rank-$D$ complex matrix, $s \in A^N_M$ is a $M$-phase $N$-tuple vector argument, $A^N_M = \{e^{j2\pi m/N} | m = 0, 1, \ldots, M - 1\}$ is the $M$-phase alphabet, and $M \in \{2^k | k = 1, 2 \ldots \}$. In the next section, we use the framework presented in [6] and propose a more generalized algorithm for the maximization of a rank-deficient quadratic form over any $M$-phase alphabet where $M \in \{2^k | k = 1, 2 \ldots \}$.

3. EFFICIENT RANK-DEFICIENT QUADRATIC-FORM MAXIMIZATION WITH A $M$-PHASE VECTOR ARGUMENT

3.1. Problem Reformulation
W.l.o.g., we assume that each row of $V$ has at least one nonzero element, i.e. $V_{n,1:D} \neq 0_{1 \times D}, \forall n \in \{1, 2, \ldots, N\}$.

Let $\phi_{i,j} \triangleq [\phi_1, \phi_{i+1}, \ldots, \phi_j]^T$. To develop an efficient method for the maximization in (1), we introduce $2D - 1$ auxiliary hyperspherical coordinates $\phi_{1:2D-1} \in [-\pi/T, \pi/T]^{2D-2} \times [-\pi/T, \pi/T]$. 

3856 ICASSP 2011
\((-\pi, \pi]\) and define the hyperspherical real vector with unit radial coordinate

\[
\tilde{c}(\phi_{1:2D-1}) \triangleq \begin{bmatrix}
\sin \phi_1 \\
\cos \phi_1 \sin \phi_2 \\
\cos \phi_1 \cos \phi_2 \sin \phi_3 \\
\vdots \\
\prod_{i=1}^{2D-2} \cos \phi_i \sin \phi_{2D-1} \\
\prod_{i=1}^{2D-2} \cos \phi_i \cos \phi_{2D-1}
\end{bmatrix}_{2D \times 1}
\]  \hspace{1cm} (2)

as well as the \(D \times 1\) hyperspherical complex vector \(c(\phi_{1:2D-1}) \triangleq \tilde{c}(\phi_{1:2D-1}) + j\tilde{c}(\phi_{1:2D-1})\).

From Cauchy-Schwarz Inequality, we observe that for any \(a \in \mathbb{C}^D\), \(|a^H c(\phi_{1:2D-1})| \leq ||a|| ||c(\phi_{1:2D-1})|| = ||a||\) with equality if and only if \(\phi_{1:2D-1} \in (-\pi, \pi]\) are the hyperspherical coordinates of vector \(a\), i.e., \(1^2 c(\phi_{1:2D-1}) = \frac{a}{||a||}\) since \(a^H c(\phi_{1:2D-1}) = \frac{\hat{a}^H c}{||a||} = ||a||\). Using the above, a critical equality for our subsequent developments is

\[
s_{\text{opt}} = \arg \max_{\theta \in A_M} \max_{s(\phi_{1:2D-1}) \in (-\pi, \pi]} ||s^H V c(\phi_{1:2D-1})||. \hspace{1cm} (11)
\]

Furthermore, we observe that for any \(a \in \mathbb{C}^D\) and any \(\theta \in (-\pi, \pi]\),

\[
\Re \left\{ a^H c(\phi_{1:2D-1}) e^{-j\theta} \right\} \leq ||a|| c(\phi_{1:2D-1}) \hspace{1cm} (3)
\]

with equality if and only if \(\theta = \bar{\theta} \triangleq \arg \{a^H c(\phi_{1:2D-1})\}\).

We observe that Cauchy-Schwarz Inequality and (4) are simultaneously satisfied with equality if and only if \(\phi_{1:2D-1} \in (-\pi, \pi]\) are the hyperspherical coordinates of vector \(a\) and \(\bar{\theta} = \theta\). Then, applying some further developments, (3) can be further transformed into:

\[
s_{\text{opt}} = \arg \max_{\theta \in A_M} \max_{s(\phi_{1:2D-1}) \in (-\pi, \pi]} ||s^H V c(\phi_{1:2D-1})||. \hspace{1cm} (5)
\]

We note that given a hyperspherical complex vector \(c(\phi_{1:2D-1})\) and \(\phi_{2D-1} \in (-\pi, \pi]\), there always exists an angle \(\alpha \in \arg\{A_M\} = \left\{ \frac{2\pi m}{M} : m = 0, 1, \ldots, M - 1 \right\}\) that relocates the angular coordinate \(\phi_{2D-1}\) of the hyperspherical vector \(\{c(\phi_{1:2D-1})\}_{\phi}\) in the interval \((-\pi, \pi]\) and results in the same value in (5).

Thus, without loss of optimality, we choose \(\alpha \in \arg\{A_M\}\) such that \(\phi_{2D-1} \in (-\pi, \pi]\). Then, (5) becomes

\[
s_{\text{opt}} = \arg \max_{\theta \in A_M} \max_{s(\phi_{1:2D-1}) \in (-\pi, \pi]} ||s^H V c(\phi_{1:2D-1})||. \hspace{1cm} (6)
\]

We drop the arg operator, interchange the maximizations in (6), and obtain the equivalent problem (for \(\Psi \triangleq (-\pi, \pi]\))

\[
\max_{\phi_{1:2D-1} \in (-\pi, \pi]} \sum_{n=1}^{N} \max_{s_{n} \in A_M} \Re \{s_{n}^H V c(\phi_{1:2D-1})\}. \hspace{1cm} (7)
\]

3.2. Decision Functions and Candidate Vector Set \(S(V_{N \times D})\)

We observe that the original problem in (1) is decomposed into symbol-by-symbol maximizations for a given \(\phi_{1:2D-1} \in (-\pi, \pi]\). For such a set of angles, the maximization argument of the sum in (7), e.g. symbol \(s_n\), depends only on the corresponding row of matrix \(V\). As \(\phi_{2D-1}\) varies, the decision in favor of \(s_n\) is maintained as long as a decision boundary is not crossed.

Due to the structure of \(A_M\) and the definitions above, the \(\frac{M}{2}\) decision boundaries for the determination of \(s_n\) are given by \(V_{n,1:D} c(\phi_{1:2D-1}) = Ae^{j\pi \frac{2n+1}{M}}, A \in \mathbb{R}\), \(k = 0, 1, \ldots, \frac{M}{2} - 1\), or, equivalently,

\[
\Re \{e^{-j\pi \frac{2n+1}{M}} V_{n,1:D} c(\phi_{1:2D-1})\} = 0, \hspace{0.5cm} k = 0, 1, \ldots, \frac{M}{2} - 1, \hspace{1cm} (8)
\]

For \(n = 1, \ldots, N\) and \(k = 0, 1, \ldots, \frac{M}{2} - 1\), (8) becomes

\[
\tilde{V}_{l,1:D} \tilde{c}(\phi_{1:2D-1}) = 0, \hspace{0.5cm} l = 1, 2, \ldots, \frac{MN}{2}, \hspace{1cm} (9)
\]

where \(\tilde{V}_{l,1:D} c(\phi_{1:2D-1}) = 0, \hspace{0.5cm} l = 1, 2, \ldots, \frac{MN}{2}\)

Motivated by the statements above and the inner maximization rule in (7), for each \(D \times 1\) complex vector \(v\) we define the decision function \(s\) that maps \(\phi_{1:2D-1}\) to \(A_M\) according to

\[
s(v^T; \phi_{1:2D-1}) \triangleq \arg \max_{s \in A_M} \Re \{s^H v c(\phi_{1:2D-1})\}. \hspace{1cm} (10)
\]

Furthermore, for the given \(N \times D\) complex observation matrix \(V\), we can construct the vector decision function \(s\) using (10) where each point \(\phi_{1:2D-1} \in \Phi_{2D-2} \times (-\pi, \pi]\) is mapped to a candidate \(M\)-phase vector according to

\[
s(V_{N \times D}; \phi_{1:2D-1}) \triangleq \begin{bmatrix} s(V_{1,1:D}; \phi_{1:2D-1}) \\ \vdots \\ s(V_{N,1:D}; \phi_{1:2D-1}) \end{bmatrix}. \hspace{1cm} (11)
\]
Computing \( s(V_{N \times D}; \phi_{1:2D-1}) \) for \( \forall \phi_{1:2D-1} \in \Phi_{2D-2} \times (-\frac{\pi}{M}, \frac{\pi}{M}] \), we collect all \( M \)-phase candidate vectors into set 
\[ S(V_{N \times D}) = \cup_{\phi_{1:2D-1} \in \Phi_{2D-2}} s(V_{N \times D}; \phi_{1:2D-1}) \] \( \subseteq \mathcal{A}_M \).

Since \( \phi_{1:2D-1} \) takes values from the set \( \Phi_{2D-2} \times (-\frac{\pi}{M}, \frac{\pi}{M}] \), our problem in (1) becomes \( s_{\text{opt}} = \arg \max_{s \in S(V)} \| \mathbf{H}^T s \| \), i.e. the \( M \)-phase candidate vector \( s_{\text{opt}} \) that maximizes the expression above belongs into the set \( S(V_{N \times D}) \). In the following, we (i) show that \( |S(V_{N \times D})| = \sum_{d=1}^{D} \sum_{i=0}^{N} \binom{N}{d} \binom{2(d-i)-2}{\frac{(d-i)-1}{2}} \), and (ii) develop an algorithm for the construction of \( S(V_{N \times D}) \) with complexity \( O((\frac{MN}{2})^{2D}) \).

### 3.3. Hypersurfaces and Cardinality of \( S(V_{N \times D}) \)

According to (9), the rows of \( \mathbf{V}_{\text{disp} \times 2D} \) determine \( \frac{MN}{2} \) hypersurfaces \( \mathcal{H} = \{ H(V_{1:2D-1}), \ldots, H(V_{2D-1}) \} \) that partition the \((2D - 1)\)-dimensional hypercube \( \Phi_{2D-2} \times (-\frac{\pi}{M}, \frac{\pi}{M}] \) into cells \( C_1, \ldots, C_K \) such that the union of all cells equals \( \Phi_{2D-2} \times (-\frac{\pi}{M}, \frac{\pi}{M}] \) and the intersection of any two distinct cells, say \( C_k, C_j \), for \( k \neq j \), is empty. Each cell \( C_k \) corresponds to a distinct \( s_k \in \mathcal{A}_M \) in the sense that \( s(V_{N \times D}; \phi_{1:2D-1}) = s_k \) for any \( \phi_{1:2D-1} \in C_k \) and \( s_k \neq s_j \) if \( k \neq j \). From (10), we observe that \( |J(V_{N \times D})| = \sum_{d=0}^{D-1} \sum_{i=0}^{N} \binom{N}{d} \binom{2(d-i)-2}{\frac{(d-i)-1}{2}} \), i.e. there are \( |J(V_{N \times D})| \) candidate vectors \( s \in \Phi_{2D-2} \times (-\frac{\pi}{M}, \frac{\pi}{M}] \), associated with cells each of which minimizes \( \phi_{2D-1} \) component at a single point that constitutes the intersection of the corresponding \( 2D - 1 \) hypersurfaces. It can also be shown that if we take into consideration all regions in \( \Phi_{2D-2} \times (-\frac{\pi}{M}, \frac{\pi}{M}] \), all candidates form the candidate set \( S \) given by

\[
S(V_{N \times D}) = \bigcup_{d=0}^{D-1} J(V_{N \times D-1}) \tag{12}
\]

with cardinality \( |S(V_{N \times D})| = \sum_{d=0}^{D-1} \sum_{i=0}^{N} \binom{N}{d} \binom{2(d-i)-2}{\frac{(d-i)-1}{2}} = O((\frac{MN}{2})^{2D-1}) \).

### 4. Algorithmic Developments and Complexity

In this section, we present the basic steps of the proposed algorithm for the construction of \( S(V_{N \times D}) \) for arbitrary \( N, D \in \mathbb{N}, D < N \) and \( M \in \{ 2^k \mid k = 1,2,\ldots \} \). From (12), we observe that the initial problem of the determination of \( S(V_{N \times D}) \) can be divided into smaller parallel construction problems of \( J(V_{N \times d}) \) for \( d = 1, \ldots, D \). Moreover, the construction of \( J(V_{N \times d}) \) can be fully parallelized since the candidate vector(s) \( s(V_{N \times d}; I_{2d-1}) \) can be computed independently for each \( I_{2d-1} \).

In the following, we assume a certain value for \( D \in \{ 1, \ldots, D \} \) and a certain set of indices \( I_{2d-1} = \{i_1, \ldots, i_{2d-1} \} \). According to the derivations in the previous section, the combination of hypersurfaces \( H(V_{1:2D-1}), \ldots, H(V_{2d-1}) \) intersects at a single point \( \phi(V_{N \times d}; I_{2d-1}) \) that “leads” \( Q \) cells associated with \( Q \) different \( M \)-phase candidate vectors \( s_q(V_{N \times d}; I_{2d-1}) \), \( q = 1,2,\ldots,Q \). It can be shown that the evaluation of the decision function in (10) at the intersection of the \( 2D - 1 \) hypersurfaces under consideration determines definitely the corresponding symbol \( s_q \) if and only if no hypersurface originates from \( V_{1:2D-1} \). For the hypersurfaces that pass through the intersection, the rule in (10) becomes ambiguous. In such a case, we have constructed disambiguation rules that solve the ambiguity in polynomial
time with respect to the length $N$. The algorithm visits independently $|S(V_{N \times D})| = O\left(\left(\frac{MN}{2}\right)^{2D-1}\right)$ intersections and computes the candidate $M$-phase vector(s) associated with each intersection. For each $J_{2d-1}$, the cost of the algorithm is $O\left(\frac{MN}{2}\right)$. Therefore the overall complexity of the algorithm for the computation of $S(V_{N \times D})$ with fixed $D < N$ becomes $O\left(\left(\frac{MN}{2}\right)^{2D-1}\right)O\left(\frac{MN}{2}\right) = O\left(\left(\frac{MN}{2}\right)^{2D}\right)$.

We observe that the computation of the candidate vectors of $S(V_{N \times D})$ is performed independently from cell to cell, which implies that there is no need to store the data that have been used for each candidate and we only have to store the “best” vector that has been met. Therefore, the proposed method is fully parallelizable and rank-scalable and its memory utilization is efficiently minimized, in contrast to the incremental algorithm in [2].

Compared to previous works on the maximization of a complex rank-deficient quadratic form over a finite field, we recall that the reverse search method [3, 4] computes $\sum_{i=0}^{D-1} \binom{N-1}{i}$ (as many as our proposed algorithm) candidates for $M = 2$ and $\sum_{i=0}^{2D-1} \binom{2N-1}{i}$ (twice as many as our proposed algorithm) candidates for $M = 4$ [see Fig. 1]. Additionally, the corresponding complexity of the algorithm proposed in [3, 4] is of the order $O(N^{2D}LP(\frac{MN}{2}, 2D))$ and $O((2N)^{2D}LP(\frac{MN}{2}, 2D))$ for $M = 2, 4$, respectively, where $LP(\frac{MN}{2}, 2D)$ is the time to solve a linear programming (LP) optimization problem with $\frac{MN}{2}$ inequalities in $2D$ variables. It turns out that the complexity of reverse search method is $O\left(\left(\frac{MN}{2}\right)^{2D+1}\right)$ for $M = 2, 4$, i.e. one order of magnitude higher than the proposed algorithm. In addition, the reverse search method is restricted to $M = 2, 4$. On the other hand, the incremental algorithm proposed in [2, 8] is a time-efficient algorithm that solves the maximization problem of interest but becomes impractical even for moderate values of $D$ since it follows an “incremental” strategy to construct the candidate set: it solves the problem inductively and, thus, it is too complicated to be implemented. Furthermore, the critical disadvantage of this method is its memory inefficiency since it needs to store all the extreme points, all faces and their incidences in memory. Finally, the algorithm proposed in [5] deals optimally the problem of the maximization of a rank-deficient quadratic form for any $M = \{2^k|k = 0, 1, \ldots\}$ but only for $D = 1$.

5. REFERENCES