Entropic Proximal Operators for Nonnegative Trigonometric Polynomials

Hsiao-Han Chao and Lieven Vandenberghe

***Abstract*—Signal processing applications of semidefinite optimization are often rooted in sum-of-squares representations of nonnegative trigonometric polynomials. Interior-point solvers for semidefinite optimization can handle constraints of this form with a per-iteration-complexity that is cubic in the degree of the trigonometric polynomial. The purpose of this paper is to discuss first-order methods with a lower complexity per iteration. The methods are based on generalized proximal operators defined in terms of the Itakura–Saito distance. This is the Bregman distance defined by the negative entropy function. The choice for the Itakura–Saito distance is motivated by the fact that the associated generalized projection on the set of normalized nonnegative trigonometric polynomials can be computed at a cost that is roughly quadratic in the degree of the polynomial. The generalized projection is the key operation in generalized proximal first-order methods that use Bregman distances instead of the squared Euclidean distance. The paper includes numerical results with Auslender and Teboulle's accelerated proximal gradient method for Bregman distances.**

I. Introduction

OPTIMIZATION problems over the cone of nonnegative trigonometric polynomials or its dual cone, the cone of positive semidefinite Toeplitz matrices, are common in signal processing and system identification [1]–[8]. Recent examples include superresolution techniques for spectrum estimation and gridless compressed sensing [9]–[11]. If the cost function admits an efficient semidefinite representation, such problems can be solved by general-purpose interior-point solvers for semidefinite optimization. To be specific, let K be the cone of nonnegative trigonometric polynomials of degree p or less:

$$K = \{x \in \mathbf{R}^{p+1} \mid F_x(e^{\mathrm{j}\omega}) \geq 0 \ \ \forall \omega\} \tag{1}$$

where $F_x(z) = x_0 + \sum_{k=1}^{p} x_k(z + z^{-1})$ is the Laurent polynomial with coefficients $x = (x_0, \ldots, x_p)$. The convex cone K can be expressed as the image of the positive semidefinite cone under a linear transformation,

$$K = \{\mathcal{D}(X) \mid X \in \mathbf{S}_+^{p+1}\} \tag{2}$$

where $\mathbf{S}_+^{p+1}$ is the set of symmetric positive semidefinite matrices of order $p+1$, and the linear mapping $\mathcal{D}$ maps X to the $(p+1)$-vector of its diagonal sums, *i.e.*,

$$\mathcal{D}(X)_k = \sum_{i=0}^{p-k} X_{i,i+k}, \quad k = 0, \ldots, p \tag{3}$$

(see [3], [5], [6]). If f is a cost function with an epigraph $\{(x,t) \mid f(x) \leq t\}$ that can be represented by linear matrix inequalities, then the equivalence between (1) and (2) allows us to formulate the problem of minimizing $f(x)$ over K as a semidefinite program (SDP), and solve it using general-purpose SDP solvers. The interior-point algorithms implemented in these solvers have a complexity of $O(p^4)$ per iteration, if we assume that the complexity is dominated by the cost of handling the constraint (2) (*i.e.*, ignoring the cost of handling the constraints that represent the epigraph of f). The special-purpose interior-point algorithms developed in [5], [12], [13] reduce the complexity to $O(p^3)$ per iteration. First-order proximal algorithms such as the proximal gradient algorithm [14], [15] or the alternating direction method of multipliers (ADMM) [16] offer no immediate improvement over the $O(p^3)$ per-iteration-complexity of the customized interior-point methods, since they require at each iteration a Euclidean projection on the positive semidefinite cone (*i.e.*, an eigenvalue decomposition of order $p+1$) and, moreover, converge more slowly than interior-point methods.

The purpose of this paper is to describe faster first-order methods, with a complexity of roughly $O(p^2)$ or $O(p(\log p)^2)$ operations per iteration. The algorithms are based on generalized proximal operators defined in terms of the *Itakura–Saito distance*

$$d(x,v) = \frac{1}{2\pi}\int_0^{2\pi} \Big(\frac{F_x(e^{\mathrm{j}\omega})}{F_v(e^{\mathrm{j}\omega})} - \log\frac{F_x(e^{\mathrm{j}\omega})}{F_v(e^{\mathrm{j}\omega})} - 1\Big)\, d\omega, \tag{4}$$

with domain $\operatorname{dom} d = (K \setminus \{0\}) \times (\operatorname{int} K)$. The Itakura–Saito distance is the Bregman distance associated with the negative entropy function

$$\phi(x) = -\frac{1}{2\pi}\int_0^{2\pi} \log F_x(e^{\mathrm{j}\omega})\, d\omega. \tag{5}$$

We present an efficient method for computing a generalized projection $x = \Pi(a, v)$, defined as the solution of the problem

$$\begin{array}{ll} \text{minimize} & \langle a, x\rangle + d(x,v) \\ \text{subject to} & x_0 = 1 \end{array} \tag{6}$$

for an arbitrary $(p+1)$-vector a and a vector $v \in \operatorname{int} K$. If we interpret $F_x(e^{\mathrm{j}\omega})$ as a power spectrum, then the constraint $x_0 = 1$ normalizes the total power

$$x_0 = \frac{1}{2\pi}\int_0^{2\pi} F_x(e^{\mathrm{j}\omega}) d\omega. \tag{7}$$

Our method for (6) reduces the problem to a nonlinear equation in one variable (equivalently, an unconstrained differentiable convex optimization problem in one variable) that can be solved using Newton's method. Each Newton iteration requires the solution of a positive definite Toeplitz equation,

The authors are with the Electrical and Computer Engineering Department, University of California, Los Angeles (e-mail: suzihchao@ucla.edu, vandenbe@ucla.edu). This work was supported in part by NSF Grant 1509789.

which takes $O(p^2)$ operations using Levinson's algorithm, or $O(p(\log p)^2)$ operations using superfast Toeplitz solvers. Since the number of Newton steps is small and weakly dependent on problem size, we conclude that the complexity of solving problem (6) is roughly $O(p^2)$ or $O(p(\log p)^2)$.

The Itakura–Saito projection operation (6) should be compared with the Euclidean projection of the vector $v-a$ on the set $\{x \in K \mid x_0 = 1\}$, *i.e.*, the solution of

$$\begin{array}{ll} \text{minimize} & \langle a, x\rangle + \frac{1}{2}\|x-v\|^2 \\ \text{subject to} & x \in K, \quad x_0 = 1 \end{array} \tag{8}$$

where $\|u\|^2 = \langle u, u\rangle$. This is a non-trivial convex optimization problem [3], [6].

To test the effectiveness of the entropic projection operator, we use it in an accelerated proximal gradient method [17], [18] for optimization problems of the form

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in K, \quad x_0 = 1, \end{array} \tag{9}$$

where f is a differentiable convex function. The entropic projection operator can be used in other types of first-order methods as well, for example, mirror descent [19].

The generalized projection can be further extended to define generalized proximal operators, which map vectors a and v to the solution of

$$\text{minimize} \quad \langle a, x\rangle + g(x_0) + \frac{1}{t}d(x, v) \tag{10}$$

where g is a possibly nondifferentiable convex function of one variable and $t > 0$. This is useful for optimization problems

$$\begin{array}{ll} \text{minimize} & f(x) + g(x_0) \\ \text{subject to} & x \in K, \end{array}$$

with differentiable f. The second term in the cost function assigns a cost to the total power (7).

The rest of the paper is organized as follows. In Section II we review some background material from statistical signal processing and numerical linear algebra related to positive definite Toeplitz systems. The negative entropy function (5) and its conjugate are discussed in Section III, and the associated Bregman distance (4) in Section IV. The main results of the paper are in Section V, where we describe the algorithm for the Itakura–Saito projection (6). Section VI contains numerical examples with a generalized proximal gradient method based on the Itakura–Saito distance. The Appendix gives more details and a proof of convergence of this proximal gradient method.

II. Forward and backward Levinson–Durbin algorithm

In this section we review some classical results and algorithms from statistical signal processing. We denote by $\mathcal{T}(y)$, where $y = (y_0, y_1, \ldots, y_p)$, the symmetric Toeplitz matrix with first column y:

$$\mathcal{T}(y) = \begin{bmatrix} y_0 & y_1 & \cdots & y_p \\ y_1 & y_0 & \cdots & y_{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_p & y_{p-1} & \cdots & y_0 \end{bmatrix}, \tag{11}$$

and by $\mathcal{J}(b)$, where $b = (b_0, b_1, \ldots, b_p)$, the matrix with elements

$$\begin{aligned} \mathcal{J}(b) \quad = \quad & \begin{bmatrix} b_0/2 & 0 & \cdots & 0 & 0 \\ b_1/2 & b_0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ b_{p-1}/2 & b_{p-2} & \cdots & b_0 & 0 \\ b_p/2 & b_{p-1} & \cdots & b_1 & b_0 \end{bmatrix} \\ & + \begin{bmatrix} b_0/2 & b_1 & \cdots & b_{p-1} & b_p \\ b_1/2 & b_2 & \cdots & b_p & 0 \\ \vdots & \vdots & \iddots & \vdots & \vdots \\ b_{p-1}/2 & b_p & \cdots & 0 & 0 \\ b_p/2 & 0 & \cdots & 0 & 0 \end{bmatrix}. \end{aligned} \tag{12}$$

This matrix is known as the *Jury matrix* [20]. Note that $\mathcal{T}(y)b = \mathcal{J}(b)y$ for all y and b.

The results in this section may be summarized as follows. Suppose $\mathcal{T}(y)$ is positive definite, and let $b = (b_0, \ldots, b_p)$ be the solution of the linear equation

$$\mathcal{T}(y)b = e, \tag{13}$$

where $e = (1, 0, \ldots, 0)$. Then the polynomial $b_0 z^p + b_1 z^{p-1} + \cdots + b_p$ is stable (has all its zeros inside the unit circle). The classical algorithm for solving this equation is the Levinson–Durbin algorithm, which computes a Cholesky factorization of $\mathcal{T}(y)^{-1}$ in order p^2 operations. Several more recent algorithms for positive definite Toeplitz systems are even faster, with an order $p(\log p)^2$ complexity [21], [22].

Second, suppose the polynomial $\mathcal{B}(z) = b_0 z^p + b_1 z^{p-1} + \cdots + b_p$ is stable. Then the Jury matrix $\mathcal{J}(b)$ is nonsingular, and the unique solution y of the equation

$$\mathcal{J}(b)y = e \tag{14}$$

defines a positive definite Toeplitz matrix $\mathcal{T}(y)$. This equation can be solved by a recursive algorithm that is essentially the Jury stability test applied to $\mathcal{B}(z)$. The algorithm computes a factorization of $\mathcal{J}(b)$ and can be interpreted as a backward Levinson–Durbin algorithm.

More details on these algorithms may be found in textbooks on statistical signal processing, for example, [23, chapter 10] or [24, chapter 11].

A. Levinson–Durbin algorithm

The Levinson–Durbin algorithm [25, §4.7] is a fast algorithm for the Cholesky factorization of the inverse of a positive definite Toeplitz matrix $\mathcal{T}(y)$, given its first column $y = (y_0, \ldots, y_p)$. The computed factorization is

$$U\mathcal{T}(y)U^T = \operatorname{diag}(\sigma_p^2, \ldots, \sigma_0^2), \tag{15}$$

where $0 < \sigma_p \leq \cdots \leq \sigma_0$ and

$$U = \begin{bmatrix} 1 & a_{p1} & a_{p2} & \cdots & a_{pp} \\ 0 & 1 & a_{p-1,1} & \cdots & a_{p-1,p-1} \\ 0 & 0 & 1 & \cdots & a_{p-2,p-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}. \tag{16}$$

For theoretical purposes later in the paper, it is useful to note that the algorithm can be extended to Toeplitz matrices that are positive semidefinite, but not positive definite. In that case we can still compute a factorization of the form (15), where $0 \le \sigma_p \le \cdots \le \sigma_0$.

Algorithm II.1. *Levinson–Durbin algorithm.*

Input. The coefficients $y_0, \ldots, y_p$ of a Toeplitz matrix $\mathcal{T}(y)$.

Output. If $\mathcal{T}(y)$ is positive semidefinite, a matrix U and coefficients $\sigma_0, \ldots, \sigma_p$ that satisfy (15), (16), and $0 \le \sigma_p \le \cdots \le \sigma_0$.

Algorithm. Define $\sigma_0 = \sqrt{y_0}$. For $k = 0, \ldots, p-1$, execute the following steps.

- If $\sigma_k = 0$, set $\kappa_k = 0$. Otherwise, define
$$\kappa_k = -\frac{y_{k+1} + y_k a_{k1} + \cdots + y_1 a_{kk}}{\sigma_k^2}. \qquad (17)$$
- If $|\kappa_k| > 1$, terminate. The matrix $\mathcal{T}(y)$ is not positive semidefinite.
- Compute $\sigma_{k+1} = \sigma_k(1-\kappa_k^2)^{1/2}$ and
$$\begin{bmatrix} a_{k+1,1} \\ \vdots \\ a_{k+1,k} \\ a_{k+1,k+1} \end{bmatrix} = \begin{bmatrix} a_{k1} & a_{kk} \\ \vdots & \vdots \\ a_{kk} & a_{k1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ \kappa_k \end{bmatrix}. \qquad (18)$$

The algorithm has complexity $O(p^2)$.

The update (18) can be written concisely using polynomial notation, if we define polynomials $\mathcal{A}_0(z) = 1$,

$$\mathcal{A}_k(z) = z^k + a_{k1} z^{k-1} + \cdots + a_{k,k-1} z + a_{kk}, \qquad (19)$$

for $k = 1, \ldots, p$, and the reversed polynomials $\hat{\mathcal{A}}_k(z) = z^k \mathcal{A}_k(1/z) = a_{kk} z^k + \cdots + a_{k1} z + 1$. With this notation, the update (18) can be written

$$\begin{bmatrix} \mathcal{A}_{k+1}(z) \\ \hat{\mathcal{A}}_{k+1}(z) \end{bmatrix} = \begin{bmatrix} 1 & \kappa_k \\ \kappa_k & 1 \end{bmatrix} \begin{bmatrix} z\mathcal{A}_k(z) \\ \hat{\mathcal{A}}_k(z) \end{bmatrix}, \qquad (20)$$

starting at $\mathcal{A}_0(z) = 1$. Another useful form is in terms of the $(p+1)$-vectors

$$a^{(k)} = (1, a_{k1}, \ldots, a_{kk}, 0, \ldots, 0). \qquad (21)$$

The recursion (18) is a linear transformation

$$a^{(k+1)} = H_k a^{(k)}, \qquad (22)$$

where

$$H_k = \begin{bmatrix} I_{k+2} + \kappa_k J_{k+2} & 0 \\ 0 & I_{p-k-1} \end{bmatrix} \qquad (23)$$

and J_r is the $r \times r$ identity with its columns reversed.

We mention two properties of the factorization (15) that will be useful later in the paper. The first column of the equation $\mathcal{T}(y)U^T = U^{-1}\operatorname{diag}(\sigma_p^2, \ldots, \sigma_0^2)$ is the *Yule–Walker equation*

$$\begin{bmatrix} y_0 & y_1 & \cdots & y_p \\ y_1 & y_0 & \cdots & y_{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_p & y_{p-1} & \cdots & y_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_{p1} \\ \vdots \\ a_{pp} \end{bmatrix} = \begin{bmatrix} \sigma_p^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \qquad (24)$$

The Levinson–Durbin algorithm solves this equation (*i.e.*, computes $a_{p1}, \ldots, a_{pp}, \sigma_p^2$, given y) in $O(p^2)$ operations. The solution is unique if the $p \times p$ Toeplitz matrix with first column $y_0, \ldots, y_{p-1}$ is positive definite. If $\mathcal{T}(y)$ is positive definite, then $\sigma_p^2 \ne 0$, and $b = \sigma_p^{-2}(1, a_{p1}, \ldots, a_{pp})$ is the solution of (13).

Second, if $\mathcal{T}(y)$ is positive definite, then the polynomials $\mathcal{A}_k(z)$ defined in (19) are stable. In particular, the polynomial

$$b_0 z^p + b_1 z^{p-1} + \cdots + b_p = \mathcal{A}_p(z)/\sigma_p^2$$

is stable. To show this, one can note that if $\mathcal{A}_k(z)$ is a stable polynomial, then $|\mathcal{A}_k(z)| \ge |\hat{\mathcal{A}}_k(z)|$ holds for $|z| \ge 1$. (This is easily seen from the fact that $|z-a|/|1-\bar{a}z| \ge 1$ if $|a| < 1$ and $|z| \ge 1$.) Therefore, if $\mathcal{A}_k(z)$ is stable and $|\kappa_k| < 1$, then $\mathcal{A}_{k+1}(z)$ defined in (20) is nonzero for $|z| \ge 1$. Since $\mathcal{A}_0(z) = 1$, stability of the polynomials $\mathcal{A}_k(z)$ follows by induction.

B. Jury stability test

In this section we discuss an algorithm that can be seen as the Levinson–Durbin algorithm run backwards. The algorithm is equivalent to the Jury test for determining the stability of a real polynomial. The connection between the Jury test and the Levinson–Durbin algorithm was made by Vieira and Kailath [26]. In the next section we will see that the algorithm also computes a factorization of the Jury matrix defined in (12).

We use the same notation (16) as before.

Algorithm II.2. *Jury stability test.*

Input. The coefficients $b_0, \ldots, b_p$ of a polynomial $\mathcal{B}(z) = b_0 z^p + \cdots + b_p$, with $b_0 > 0$.

Output. If $\mathcal{B}(z)$ is stable, a unit upper triangular matrix U with first row $(b_0, \ldots, b_p)/b_0$ and coefficients $0 < \sigma_p \le \cdots \le \sigma_0$ such that $U^{-1}\operatorname{diag}(\sigma_p^2, \ldots, \sigma_0^2)U^{-T}$ is Toeplitz.

Algorithm. Define $\sigma_p = 1/\sqrt{b_0}$ and

$$(a_{p1}, \ldots, a_{pp}) = (b_1/b_0, \ldots, b_p/b_0).$$

For $k = p-1, \ldots, 0$, execute the following steps.

- Define $\kappa_k = a_{k+1,k+1}$. If $|\kappa_k| \ge 1$, terminate. The polynomial $\mathcal{B}(z)$ is not stable.
- Otherwise, compute $\sigma_k = \sigma_{k+1}/\sqrt{1-\kappa_k^2}$ and
$$\begin{bmatrix} a_{k1} \\ \vdots \\ a_{kk} \end{bmatrix} = \frac{1}{1-\kappa_k^2} \begin{bmatrix} a_{k+1,1} & a_{k+1,k} \\ \vdots & \vdots \\ a_{k+1,k} & a_{k+1,1} \end{bmatrix} \begin{bmatrix} 1 \\ -\kappa_k \end{bmatrix}.$$

The Jury stability test is successful if $|\kappa_k| < 1$ for $k = p-1, \ldots, 0$, or, equivalently, $\sigma_0 \ge \sigma_1 \ge \cdots \ge \sigma_p > 0$. The complexity of this algorithm is $O(p^2)$.

The relation with the Levinson–Durbin algorithm is clear if we write the update in Algorithm II.2 as a recursion for the polynomials $\mathcal{A}_k(z)$ (with coefficients a_{ki}, as defined in (19)),

$$\begin{bmatrix} z\mathcal{A}_k(z) \\ \hat{\mathcal{A}}_k(z) \end{bmatrix} = \begin{bmatrix} 1 & \kappa_k \\ \kappa_k & 1 \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{A}_{k+1}(z) \\ \hat{\mathcal{A}}_{k+1}(z) \end{bmatrix},$$

and compare this with (20). The choice $\kappa_k = a_{k+1,k+1}$ ensures that $\mathcal{A}_k(z)$ is a polynomial of degree k. This form of the recursion also explains why the Jury test works. Suppose $\mathcal{A}_{k+1}(z)$ is stable. Then $|\kappa_k| < 1$, since $|a_{k+1,k+1}|$ is the product of the absolute values of the zeros of $\mathcal{A}_{k+1}(z)$. Moreover, since $|\mathcal{A}_{k+1}(z)| \geq |\hat{\mathcal{A}}_{k+1}(z)|$ for $|z| \geq 1$, the polynomial $\mathcal{A}_k(z)$ is nonzero for $|z| \geq 1$. Therefore if the recursion starts with a stable polynomial $\mathcal{A}_p(z) = (1/b_0)\mathcal{B}(z)$, then the polynomials $\mathcal{A}_k(z)$ are stable and $|\kappa_k| < 1$ for $k = p-1, \ldots, 0$. The converse can be shown as in the proof of stability of the polynomials generated by the Levinson–Durbin algorithm. If $|\kappa_k| < 1$ for $k = 0, \ldots, p-1$, then the recursion $\mathcal{A}_{k+1}(z) = z\mathcal{A}_k(z) + \kappa_k \hat{\mathcal{A}}_k(z)$ started at $\mathcal{A}_0(z) = 1$ generates a sequence of stable polynomials.

In terms of the vectors $a^{(k)}$ defined as in (21), the recursion in Algorithm II.2 can be written as

$$a^{(k)} = H_k^{-1} a^{(k+1)}, \tag{25}$$

with H_k defined in (23), where κ_k is the reflection coefficient computed by Algorithm II.2.

C. Factorization of Jury matrix

Vostrý in [27] points out that Algorithm II.2 computes a factorization of the Jury matrix (12). Using the formula (25) for the recursion of Algorithm II.2, we find that

$$b_0^{-1} H_0^{-1} \cdots H_{p-1}^{-1} \mathcal{J}(b) = \begin{bmatrix} 1 & 0 \\ 0 & L \end{bmatrix}, \tag{26}$$

where L is the $p \times p$ unit lower-triangular matrix

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ a_{11} & 1 & 0 & \cdots & 0 & 0 \\ a_{22} & a_{21} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{p-2,p-2} & a_{p-2,p-3} & a_{p-2,p-4} & \cdots & 1 & 0 \\ a_{p-1,p-1} & a_{p-1,p-2} & a_{p-1,p-3} & \cdots & a_{p-1,1} & 1 \end{bmatrix}.$$

The factorization shows that the Jury matrix is nonsingular if the vector b defines a stable polynomial. It also provides an $O(p^2)$ algorithm for solving equations with coefficient matrix $\mathcal{J}(b)$. In particular, it can be used to solve (14). To compute $y = \mathcal{J}(b)^{-1} e$, we first compute

$$\frac{1}{b_0} H_0^{-1} \cdots H_{p-1}^{-1} e = (\sigma_0^2, -\kappa_0\sigma_0^2, \ldots, -\kappa_{p-1}\sigma_{p-1}^2)$$

and then calculate y using forward substitution with the triangular matrix on the right-hand side of (26). In other words, from the output of Algorithm II.2, the solution of (14) can be computed as $y_0 = \sigma_0^2$ and

$$y_{k+1} = -\sigma_k^2 \kappa_k - y_1 a_{kk} - \cdots - y_k a_{k1}, \tag{27}$$

for $k = 0, \ldots, p-1$.

III. Entropy

Recall the definition of the cone of nonnegative trigonometric polynomials in (1) and its semidefinite characterization (2). To simplify notation, we use the inner product

$$\begin{aligned} \langle x, y \rangle &= x_0 y_0 + 2x_1 y_1 + \cdots + 2x_p y_p \\ &= \frac{1}{2\pi} \int_0^{2\pi} F_x(e^{\mathrm{j}\omega}) F_y(e^{\mathrm{j}\omega})\, d\omega, \end{aligned} \tag{28}$$

on $\mathbf{R}^{p+1}$. The adjoint of the linear mapping $\mathcal{D}$, for this inner product on $\mathbf{R}^{p+1}$ and the trace inner product on $\mathbf{S}^{p+1}$, is the function $\mathcal{T} : \mathbf{R}^{p+1} \to \mathbf{S}^{p+1}$ that maps y to the symmetric Toeplitz matrix (11). The dual cone $K^* = \{y \mid \langle y, x \rangle \geq 0 \ \forall x \in K\}$ is the cone of positive semidefinite Toeplitz matrices

$$K^* = \{y \mid \mathcal{T}(y) \succeq 0\}. \tag{29}$$

We will discuss two related convex functions, associated with the cones K and K^*. The first function is

$$\phi(x) = -\frac{1}{2\pi} \int_0^{2\pi} \log F_x(e^{\mathrm{j}\omega}) d\omega, \tag{30}$$

with domain $\operatorname{dom} \phi = K \setminus \{0\}$. The second function is

$$\psi(y) = \log(e^T \mathcal{T}(y)^{-1} e) \tag{31}$$

with domain $\operatorname{dom} \psi = \operatorname{int} K^*$. (Recall that $e = (1, 0, \ldots, 0)$, so $e^T \mathcal{T}(y)^{-1} e = (\mathcal{T}(y)^{-1})_{00}$.) The function ψ can be evaluated by solving the Yule–Walker equation (24) with coefficient matrix $\mathcal{T}(y)$, since $e^T \mathcal{T}(y)^{-1} e = 1/\sigma_p^2$. Another useful expression is

$$\psi(y) = -\log(y_0 - \tilde{y}^T \tilde{\mathcal{T}}(y)^{-1} \tilde{y}) \tag{32}$$

where we refer to a partition of $\mathcal{T}(y)$ as

$$\mathcal{T}(y) = \left[\begin{array}{c|ccc} y_0 & y_1 & \cdots & y_p \\ \hline y_1 & y_0 & \cdots & y_{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_p & y_{p-1} & \cdots & y_0 \end{array}\right] = \begin{bmatrix} y_0 & \tilde{y}^T \\ \tilde{y} & \tilde{\mathcal{T}}(y) \end{bmatrix}.$$

The expression in (32) shows that ψ is a convex function, since the argument of the logarithm is concave in y.

Up to a change of sign and a constant, the two functions form a pair of conjugates; we will see that

$$\phi^*(y) = \psi(-y) - 1, \qquad \psi^*(x) = \phi(-x) - 1. \tag{33}$$

Discussions of the duality relations between the two functions and their importance in signal processing can be found in [28], [29]. In Section IV the function ϕ is used as the kernel to define a Bregman distance.

A. Semidefinite representations

It will be useful to express the functions $\phi(x)$ and $\psi(y)$ as optimal values of convex optimization problems.

We first consider the negative entropy function ϕ. If $x \in K \setminus \{0\}$, then $F_x(z)$ has a spectral factorization

$$F_x(z) = \mathcal{B}_*(z)\mathcal{B}(z) \tag{34}$$

where $\mathcal{B}(z) = b_0 + b_1 z^{-1} + \cdots + b_p z^{-p}$ and $\mathcal{B}_*(z) = \mathcal{B}(1/z)$, with real coefficients b_0, …, b_p and $b_0 > 0$. The factor $\mathcal{B}(z)$

can be chosen to have all its zeros on or inside the unit circle ($\mathcal{B}(z) \neq 0$ for $|z| > 1$). If $x \in \operatorname{int} K$, then $\mathcal{B}(z)$ can be chosen to have its zeros inside the unit circle. This choice of $\mathcal{B}(z)$ is known as the *minimum-phase* spectral factor and is unique. From the minimum-phase spectral factors we immediately obtain the value of the negative entropy function:

$$\phi(x) = -2 \log b_0. \tag{35}$$

The minimum-phase spectral factorization of positive trigonometric polynomials is efficiently computed by the *cepstral method*, described in [30, appendix D] [31, §5.4], or by the Newton method proposed by Tunnicliffe Wilson [32]. Tunnicliffe Wilson's method finds the coefficients b in the spectral factorization (34) by solving the equivalent set of quadratic equations $\mathcal{D}(bb^T) = x$ via Newton's method.

It is also known that spectral factorization problems can be formulated as semidefinite programming problems with low-rank solutions. Replacing bb^T with a positive semidefinite matrix $X \in \mathbf{S}^{p+1}$ gives a convex relaxation

$$\mathcal{D}(X) = x, \qquad X \succeq 0.$$

The feasible solution X with maximum element $X_{00} = e^T X e$ can be shown to be equal to $X = bb^T$, where b is the vector of coefficients of the minimum-phase spectral factor; see [33] [34, theorem 6.6] [6, theorem 2.15]. If we combine this fact with the expression (35), we see that the negative entropy $\phi(x)$ is the optimal value of the convex optimization problem

$$\begin{array}{ll} \text{minimize} & -\log(e^T X e) \\ \text{subject to} & \mathcal{D}(X) = x, \quad X \succeq 0 \end{array} \tag{36}$$

in the variable X, as a function of the right-hand side x of the equality constraint.

Convex duality then gives another expression for $\phi(x)$. A convenient dual for (36) can be derived starting from the reformulation

$$\begin{array}{ll} \text{minimize} & -\log v \\ \text{subject to} & e^T X e = v \\ & \mathcal{D}(X) = x, \quad X \succeq 0, \end{array} \tag{37}$$

with an extra scalar variable v. The Lagrange dual is

$$\begin{array}{ll} \text{maximize} & \log w - \langle x, y \rangle + 1 \\ \text{subject to} & wee^T \preceq \mathcal{T}(y), \end{array} \tag{38}$$

with a scalar variable w and a vector variable y. These variables are the Lagrange multipliers for the equality constraints in (37). Since strong duality holds (the dual problem is strictly feasible), $\phi(x)$ is also equal to the optimal value of (38).

The dual problem (38) can be further simplified by eliminating w. Dual feasibility requires the Toeplitz matrix $\mathcal{T}(y)$ to be positive semidefinite. It therefore has a factorization (15), and the inequality in the dual problem can be written as

$$\operatorname{diag}(\sigma_p^2, \ldots, \sigma_0^2) = U \mathcal{T}(y) U^T \succeq w(Ue)(Ue)^T = wee^T.$$

If $\mathcal{T}(y)$ is singular, we have $\sigma_p^2 = 0$ and there exists no solution with positive w, so the problem is infeasible. If $\mathcal{T}(y)$ is nonsingular, we have $0 < \sigma_p^2 \leq \cdots \leq \sigma_0^2$, so $w = \sigma_p^2 = 1/(e^T \mathcal{T}(y)^{-1} e)$ at the optimum. The result of this elimination step is an unconstrained optimization problem in the variable y:

$$\text{maximize} \quad -\psi(y) - \langle x, y \rangle + 1. \tag{39}$$

The optimal value of this problem is again equal to $\phi(x)$.

Using similar arguments, we derive semidefinite programming representations of $\psi(y)$. If $\mathcal{T}(y)$ is positive definite, then $\psi(y)$ is the optimal value of the problem

$$\begin{array}{ll} \text{minimize} & -\log w \\ \text{subject to} & wee^T \preceq \mathcal{T}(y), \end{array} \tag{40}$$

with variable w. The dual of this problem is

$$\begin{array}{ll} \text{maximize} & \log(e^T X e) - \langle \mathcal{D}(X), y \rangle + 1 \\ \text{subject to} & X \succeq 0, \end{array} \tag{41}$$

with a symmetric variable X. Since $\phi(x)$ is the optimal value of (36), the dual can be written as

$$\text{maximize} \quad -\phi(x) - \langle x, y \rangle + 1, \tag{42}$$

with variable x. By strong duality, the optimal values of (41) and (42) are also equal to $\psi(y)$.

B. Gradients

We have seen how $\phi(x)$ can be evaluated via spectral factorization, and $\psi(y)$ by solving a Yule–Walker equation. We now discuss algorithms for computing the gradients of the two functions.

Suppose $x \in \operatorname{int} K = \operatorname{int} \operatorname{dom} \phi$. The optimal value of (36) and of the dual problem (38) is $\phi(x)$. From convex duality theory, if the dual has a unique optimal solution y, then the optimal value is differentiable and

$$\nabla \phi(x) = -y.$$

The techniques described in Sections II-B and II-C allow us to construct the unique dual optimal solution y from the primal optimal solution, as follows. A primal feasible X and dual feasible y, w are optimal for (36) and (38) if they satisfy

$$w = X_{00}^{-1}, \qquad (\mathcal{T}(y) - wee^T) X = 0 \tag{43}$$

(see [35, chapter 5]). The second equality is known as *complementary slackness*. Now let b be the vector of coefficients of the minimum-phase spectral factor, so $X = bb^T$ is optimal for (36). Then, from (43) the dual optimal solution w, y satisfies $w = 1/b_0^2$ and $\mathcal{J}(b) y = \mathcal{T}(y) b = (1/b_0) e$. The solution y can be computed using Algorithm II.2 and the factorization of $\mathcal{J}(b)$ given in Section II-C. Algorithm II.2 thus provides an $O(p^2)$ algorithm for computing the gradient of ϕ at a point $x \in \operatorname{int} K$, from its spectral factor.

The function ψ is clearly differentiable, with gradient

$$\nabla \psi(y) = -\frac{1}{e^T \mathcal{T}(y)^{-1} e} \mathcal{D}(\mathcal{T}(y)^{-1} e e^T \mathcal{T}(y)^{-1}). \tag{44}$$

The gradient is easily obtained from the solution of the Yule–Walker equation (24). If we define $a = (1, a_{p1}, \ldots, a_{pp})$, then $\nabla \psi(y) = -(1/\sigma_p^2) \mathcal{D}(aa^T)$.

C. Legendre property

We have shown that $\phi(x)$ is the optimal value of (39):

$$\phi(x) = \sup_y \left(-\langle x, y\rangle - \psi(y)\right) + 1 = \psi^*(-x) + 1. \tag{45}$$

This is the second of the conjugacy relations (33). Similarly, from the fact that $\psi(y)$ is the optimal value of (42) we conclude that

$$\psi(y) = \sup_x \left(-\langle x, y\rangle - \phi(x)\right) + 1 = \phi^*(-y) + 1. \tag{46}$$

This gives the first identity in (33). The relation (46) can also be obtained directly from (45) by noting that ψ is a closed convex function (closed because its domain is open, and its value $\psi(y)$ tends to infinity as y approaches the boundary of its domain and $\sigma_p \to 0$). Therefore $\psi^{**} = \psi$ [36, theorem 12.2], and the identity $\phi^*(y) = \psi(-y) - 1$ follows by taking the conjugates of the two sides of (45).

In addition to being closed, convex, and differentiable on an open domain $\operatorname{int} K^*$, the function ψ is strictly convex. It is therefore a convex function of *Legendre type* [36, p.258]. By [36, theorem 26.5] the pair $(\operatorname{int} K, \phi)$ is also of Legendre type, and the gradient $\nabla\psi$ is a one-to-one mapping from $\operatorname{int} K^*$ to $-\operatorname{int} K$, with inverse

$$(\nabla\psi)^{-1}(x) = -\nabla\phi(-x). \tag{47}$$

Algorithms II.1 and II.2 give efficient algorithms for evaluating the two gradient mappings. Even though the function ϕ is finite on the boundary of K (except at the origin), it is *essentially smooth, i.e.*, $\|\nabla\phi(x)\|$ grows unboundedly as x approaches the boundary [36, p.251].

IV. Itakura–Saito distance

Let $h : \mathbf{R}^n \to \mathbf{R}$ be a closed strictly convex function with $\operatorname{int}(\operatorname{dom} h) \neq \emptyset$, and assume h is differentiable on $\operatorname{int}(\operatorname{dom} h)$. The *Bregman distance* with *kernel* h is the function

$$d_h(x, v) = h(x) - h(v) - \langle \nabla h(v), x - v\rangle, \tag{48}$$

with domain $\operatorname{dom} d_h = \operatorname{dom} h \times \operatorname{int}(\operatorname{dom} h)$. For example, the squared Euclidean distance $d_h(x, v) = (1/2)\|x - v\|_2^2$ is the Bregman distance for $h(x) = (1/2)\|x\|_2^2$ and the standard inner product $\langle u, v\rangle = u^T v$. The best known non-quadratic example is the relative entropy

$$d_h(x, v) = \sum_{i=1}^n (x_i \log(x_i/v_i) - x_i + v_i), \tag{49}$$

which is the Bregman distance for the negative entropy function $h(x) = \sum_i x_i \log x_i$ and the standard inner product.

From the definition (48) it is clear that d_h is convex in x for fixed v. By convexity of h, we also have $d_h(x, v) \geq 0$ for all $(x, v) \in \operatorname{dom} d_h$. Strict convexity of h further implies that $d_h(x, v) = 0$ only if $x = v$. However, $d_h(x, v) \neq d_h(v, x)$ in general, so $d_h(x, v)$ is not a true distance.

A. Itakura–Saito and Kullback–Leibler distance

The Bregman distance d_ϕ for the negative entropy kernel (30) and the inner product (28) is called the *Itakura–Saito distance*. To simplify notation we omit the subscript in d_ϕ, and define

$$\begin{aligned} d(x, v) &= \phi(x) - \phi(v) - \langle \nabla\phi(v), x - v\rangle \\ &= \frac{1}{2\pi}\int_0^{2\pi} \left(\frac{F_x(e^{\mathrm{j}\omega})}{F_v(e^{\mathrm{j}\omega})} - \log\frac{F_x(e^{\mathrm{j}\omega})}{F_v(e^{\mathrm{j}\omega})} - 1\right) d\omega. \end{aligned} \tag{50}$$

The domain of d is $(K \setminus \{0\}) \times (\operatorname{int} K)$. The Itakura–Saito distance was first proposed and has been studied extensively in speech processing [37], [38]. For surveys of the Itakura–Saito and other spectral distance measures, see [39]–[41].

The Legendre property of the underlying kernel function ϕ makes the Itakura–Saito distance well suited for the generalized proximal methods discussed in sections V and VI. This is a key difference with the better known *Kullback–Leibler divergence,*

$$\frac{1}{2\pi}\int_0^{2\pi} (F_x(e^{\mathrm{j}\omega}) \log\frac{F_x(e^{\mathrm{j}\omega})}{F_v(e^{\mathrm{j}\omega})} - F_x(e^{\mathrm{j}\omega}) + F_v(e^{\mathrm{j}\omega}))d\omega,$$

which is also a Bregman distance, namely for the kernel function

$$\tilde\phi(x) = \frac{1}{2\pi}\int_0^{2\pi} F_x(e^{\mathrm{j}\omega}) \log F_x(e^{\mathrm{j}\omega}) d\omega. \tag{51}$$

The function $\tilde\phi$ is not essentially smooth, however, *i.e.*, its gradient does not necessarily go to infinity as x approaches the boundary of K. Figure 1 illustrates the different behavior of ϕ and $\tilde\phi$ near the boundary of K.

B. Strong convexity

Another important property of the Itakura–Saito distance, required for its use in generalized proximal gradient methods, follows from the strong convexity of the negative entropy function $\phi(x)$ when restricted to a bounded set. We define the norm

$$\|x\|_1 = \frac{1}{2\pi}\int_0^{2\pi} |F_x(e^{\mathrm{j}\omega})| d\omega.$$

With respect to this norm the function ϕ is 1-strongly convex on the set $\{x \mid \|x\| \leq 1\}$ where $\|x\| = \langle x, x\rangle^{1/2}$. In other words,

$$d(x, v) \geq \frac{1}{2}\|x - v\|_1^2 \tag{52}$$

for all $(x, v) \in \operatorname{dom} d$ with $\|x\| \leq 1$ and $\|v\| \leq 1$. To see this, we consider $v \in \operatorname{int} K$ and $x \in K \setminus \{0\}$, and define $g(t) = \phi(v + tw)$ with $w = x - v$. The second derivative is

$$\begin{aligned} g''(t) &= \frac{1}{2\pi}\int_0^{2\pi} \frac{F_w(e^{\mathrm{j}\omega})^2}{F_{v+tw}(e^{\mathrm{j}\omega})^2} d\omega \\ &\geq \left(\frac{1}{2\pi}\int_0^{2\pi} \frac{F_w(e^{\mathrm{j}\omega})^2}{F_{v+tw}(e^{\mathrm{j}\omega})^2} d\omega\right) \|v + tw\|^2 \\ &\geq \|w\|_1^2. \end{aligned}$$

The first inequality follows from $\|v+tw\| \leq 1$, and the second inequality from the Cauchy–Schwarz inequality. Integrating the inequality $g''(t) \geq \|w\|_1^2$ twice gives (52). More generally, $d(x, v) \geq (\sigma/2)\|x - v\|_1^2$ when $\|x\| \leq 1/\sqrt{\sigma}$ and $\|v\| \leq 1/\sqrt{\sigma}$.

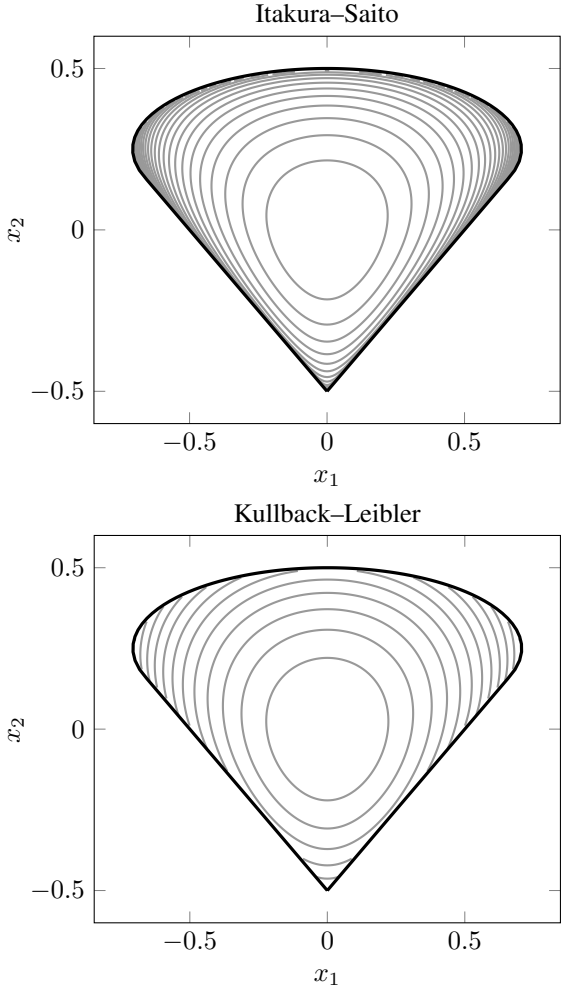

Fig. 1. *Top.* Contour lines of the function $\phi(1, x_1, x_2)$, defined in (30), on the set $\{(x_1, x_2) \mid (1, x_1, x_2) \in K\}$. *Bottom.* Contour lines of the function $\tilde{\phi}(1, x_1, x_2)$ defined in (51).

V. Entropic proximal operators

Proximal algorithms, such as the projected and proximal gradient methods and their accelerated variants [14], [42], the Douglas-Rachford method and alternating direction method of multipliers [16], [43], or Dykstra's sequential projection method [44], depend on efficient methods for evaluating the proximal operators of cost functions. The proximal operator of a convex function g is the mapping

$$\mathrm{prox}_g(u) = \operatorname*{argmin}_x \left(g(x) + \frac{1}{2}\|x - u\|_2^2\right), \tag{53}$$

where $\|\cdot\|_2$ is the Euclidean norm. If g is the indicator function of a set, this is the Euclidean projection of u on the set.

Useful extensions of the proximal methods are obtained by replacing the squared Euclidean distance in the definition by a generalized Bregman distance function, in the hope of making the generalized proximal operators or projections easier to compute. There is an extensive literature on methods of this type (see, for example, the book [45]), and the properties that the generalized distance function must satisfy depend on the algorithm in which they are applied [46]. In the numerical experiments of the next section we will apply one specific algorithm, Auslender and Teboulle's generalization of an accelerated proximal gradient method due to Nesterov [17], [18]. The generalized proximal operator used in this method is defined as

$$\mathrm{prox}_g^h(a, v) = \operatorname*{argmin}_x \left(\langle a, x\rangle + g(x) + d_h(x, v)\right), \tag{54}$$

where d_h is a Bregman distance (48). On the right-hand side of (54), the vectors a and v are given, with $v \in \operatorname{int}(\operatorname{dom} h)$. The variable in the minimization problem is x and the feasible set is $\operatorname{dom} g \cap \operatorname{dom} h$. This is a generalization of (53): if $d_h(x, v) = (1/2)\|x - v\|_2^2$ and $\langle a, x\rangle = a^T x$, then the solution of (54) is $\mathrm{prox}_g(v - a)$.

Proximal algorithms that use the generalized definition (54) require that for every a and every $v \in \operatorname{int}(\operatorname{dom} h)$, the minimizer in (54) is a unique and easily computed point $\hat{x} \in \operatorname{int}(\operatorname{dom} h)$. The classical example is the indicator $g(x) = \delta_C(x)$ of the hyperplane $C = \{x \mid \mathbf{1}^T x = 1\}$, and the relative entropy function (49). With this choice of g and d_h, the solution of the optimization problem in (54) is

$$\hat{x}_i = \frac{v_i e^{-a_i}}{\sum_{j=1}^n v_j e^{-a_j}}, \quad i = 1, \ldots, n.$$

Sufficient conditions that guarantee existence in $\operatorname{int}(\operatorname{dom} h)$ and uniqueness of the solution of (54) are discussed in papers on generalized distances (for example, [19], [46]).

In the following sections we consider the generalized proximal operator (54) defined by the Itakura–Saito distance (50) and the inner product (28).

A. Projection

We first take for g the indicator function of $\{x \mid x_0 = 1\}$ and denote the generalized proximal operator by

$$\Pi(a, v) = \operatorname*{argmin}_{x_0 = 1} \left(\langle a, x\rangle + d(x, v)\right) \tag{55}$$

To simplify notation we define $c = a - \nabla\phi(v)$ and write the minimization problem in the definition as

$$\begin{array}{ll} \text{minimize} & \langle c, x\rangle + \phi(x) \\ \text{subject to} & x_0 = 1. \end{array} \tag{56}$$

The feasible set is a compact set $\{x \in K \mid x_0 = 1\}$. Since ϕ is strictly convex and essentially smooth, the problem has a unique solution in $\operatorname{int} K$, for every c. The optimality conditions for the projection problem are

$$\nabla\phi(x) = -c - \lambda e, \qquad \langle e, x\rangle = 1.$$

The variable λ is a Lagrange multiplier for the equality constraint in (56). The unknown x can be eliminated from the first equation, using the inverse gradient mapping in (47). Substituting $x = -\nabla\psi(c + \lambda e)$ in the second equation gives a nonlinear equation

$$\langle e, \nabla\psi(c + \lambda e)\rangle + 1 = 0.$$

More explicitly, in view of (44), λ is the root of the equation

$$-\frac{e^T(\mathcal{T}(c) + \lambda I)^{-2} e}{e^T(\mathcal{T}(c) + \lambda I)^{-1} e} + 1 = 0 \tag{57}$$

in the interval $(-\lambda_{\min}(\mathcal{T}(c)), \infty)$. After solving the nonlinear equation for λ, we compute the solution of the Yule–Walker equation with coefficient matrix $\mathcal{T}(c)+\lambda I$, and obtain x from the expression (44).

Solving (57) is equivalent to solving the dual of problem (56), which is given by

$$\text{max.} \quad -\phi^*(-c-\lambda e) - \lambda = -\psi(c+\lambda e) - \lambda + 1.$$

As we have seen, the negative of the cost function

$$\begin{aligned} h(\lambda) &= \psi(c+\lambda e) + \lambda - 1 \\ &= \log(e^T(\mathcal{T}(c)+\lambda I)^{-1}e) + \lambda - 1 \end{aligned}$$

is strictly convex and differentiable on the interval $(-\lambda_{\min}(\mathcal{T}(c)), \infty)$. It increases to ∞ as $\lambda \to -\lambda_{\min}(\mathcal{T}(c))$ and as $\lambda \to \infty$. The optimal λ can therefore be found by setting the derivative of h to zero. The derivative $h'(\lambda)$ is the left-hand side of (57).

To solve the nonlinear equation (57), one can minimize $h(\lambda)$ by Newton's method with a backtracking line search, or use a safeguarded Newton method. To check whether $\lambda > -\lambda_{\min}(\mathcal{T}(c))$, one can use the Levinson–Durbin algorithm and terminate the recursion early, as soon as a reflection coefficient with $|\kappa_k| \geq 1$ is found.

A starting value $\lambda > -\lambda_{\min}(\mathcal{T}(c))$ is easily found by embedding the $\mathcal{T}(c)$ in a symmetric circulant matrix. The smallest eigenvalue of the circulant matrix is a lower bound on the smallest eigenvalue of $\mathcal{T}(c)$ and can be computed by the discrete Fourier transform.

The derivative $h'(\lambda)$ is the left-hand side of (57). The second derivative is

$$-\frac{(e^T(\mathcal{T}(c)+\lambda I)^{-2}e)^2}{(e^T(\mathcal{T}(c)+\lambda I)^{-1}e)^2} + 2\frac{e^T(\mathcal{T}(c)+\lambda I)^{-3}e}{e^T(\mathcal{T}(c)+\lambda I)^{-1}e}.$$

The value of $h(\lambda)$ and its derivatives follow from the solution of the Yule–Walker equation with coefficient matrix $\mathcal{T}(c)+\lambda I$. They can be computed in order p^2 operations by the Levinson–Durbin algorithm, or in order $p(\log p)^2$ operations by superfast algorithms for positive definite Toeplitz systems.

Let $\lambda^\star$ be the solution of (57). The derivative $h'(\lambda)$, which is given by the left-hand side of (57), increases monotonically from $-\infty$ to zero on the interval $(-\lambda_{\min}(\mathcal{T}(c)), \lambda^\star]$ and from zero to one on the interval $[\lambda^\star, \infty)$. When started at a point $\lambda^{(0)} \in (-\lambda_{\min}(\mathcal{T}(c)), \lambda^\star)$, Newton's method with unit steps produces an increasing sequence of values that converges to $\lambda^\star$ from the left. When started at a point $\lambda^{(0)} \in (\lambda^\star, \infty)$, the Newton update may be infeasible, and backtracking or bisection steps can be taken to find a point in $(-\lambda_{\min}(\mathcal{T}(c)), \lambda^{(0)})$.

In practice, a small number of Newton iterations (on the order of 10) is sufficient, almost independent of problem size. The cost of the projection algorithm is therefore a small multiple of the cost of solving positive definite Toeplitz system.

B. Proximal operator

The method of the previous section can be extended to generalized proximal operators (54) where $g(x)$ has the form $g(x) = \tilde{g}(x_0)$, with $\tilde{g}$ a convex function of one variable. This generalized proximal operator maps vectors $a \in \mathbf{R}^{p+1}$ and $v \in \operatorname{int} K$ to the vector

$$\operatorname*{argmin}_x \left(\langle a, x\rangle + \tilde{g}(x_0) + d(x, v)\right). \tag{58}$$

The projection operator discussed in Section V-A is a special case with $\tilde{g}$ the indicator function of $\{1\}$. Other interesting choices are the indicator function of $[0, 1]$ and $\tilde{g}(t) = t^2$ on $t \geq 0$. We will assume that $\lim_{t\to\infty} \tilde{g}(t)/t = \infty$ and, without loss of generality, that $\operatorname{dom} \tilde{g} \subseteq \mathbf{R}^+$. This implies that the conjugate $\tilde{g}^*(\lambda)$ is defined for all λ [36, corollary 13.3.1].

If we introduce an auxiliary variable u and define $c = a - \nabla\phi(v)$, the minimization in the definition (58) is

$$\begin{array}{ll} \text{minimize} & \langle c, x\rangle + \tilde{g}(u) + \phi(x) \\ \text{subject to} & \langle e, x\rangle = u. \end{array}$$

The Lagrange dual of this problem is

$$\text{maximize} \quad -\psi(c+\lambda e) - \tilde{g}^*(\lambda) + 1.$$

If $\tilde{g}^*$ is simple, as in the examples mentioned above, this optimization problem with one variable can be solved by modifying the methods described in Section V-A.

VI. Numerical experiments

In this section we use the generalized projection $\Pi(a, v)$ in an accelerated proximal gradient method for solving convex problems of the form

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in K, \quad x_0 = 1, \end{array} \tag{59}$$

where f is convex and differentiable. The algorithm is IGA (Improved Interior Gradient Algorithm) from [17], and is also discussed in [18, Algorithm 1]. It is an extension to non-Euclidean projections of an accelerated proximal gradient algorithm by Nesterov. The algorithm generates three strictly feasible sequences v^k, x^k, y^k, using the following recursion started at a strictly feasible $v^0 = x^0$:

$$\begin{aligned} y^k &= (1-\theta_k)x^{k-1} + \theta_k v^{k-1} && \text{(60a)} \\ v^k &= \Pi(\tau_k \nabla f(y^k), v^{k-1}) && \text{(60b)} \\ x^k &= (1-\theta_k)x^{k-1} + \theta_k v^k. && \text{(60c)} \end{aligned}$$

The Appendix gives more details, including strategies for choosing the parameters $\theta_k \in (0, 1)$ and $\tau_k > 0$. (In the experiments we used the monotonic search strategy.)

A. Covariance estimation

As a first example, we consider a variation of the line spectrum estimation example in [47, §5.1]. We estimate the parameters in a signal model

$$s(t) = \sum_{k=1}^{\rho} c_k e^{\mathrm{j}\omega_k t} + w(t), \tag{61}$$

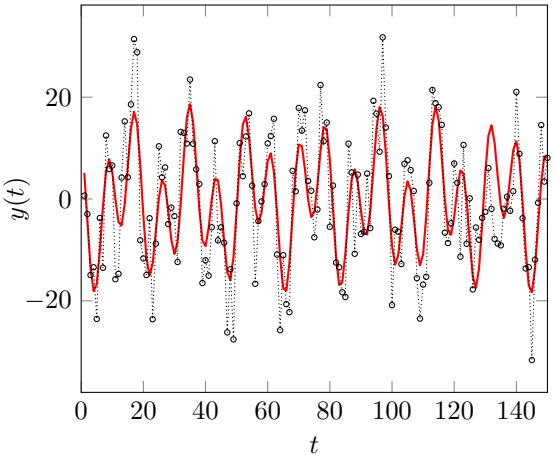

Fig. 2. True signal (solid line) and noisy samples (circles).

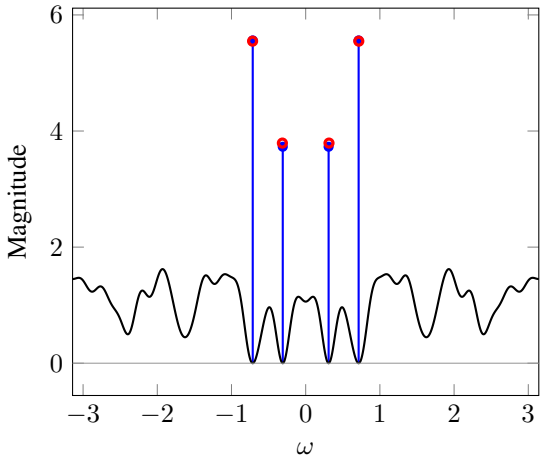

Fig. 3. True and estimated line spectrum (red circles and blue stem lines, respectively) and dual optimal polynomial $F_x(e^{j\omega})$ (solid line).

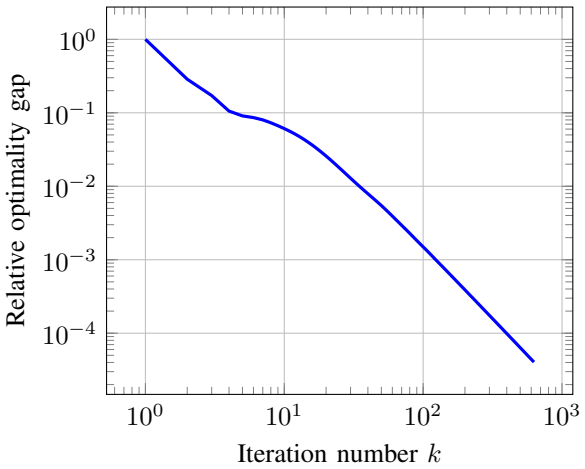

Fig. 4. Convergence of the generalized proximal gradient method applied to (64).

where $w(t)$ is white noise with variance σ^2. Under standard assumptions [48, §4.1] the covariance matrix of $s(t)$ of order $p+1$ is given by

$$R = \sigma^2 I + \sum_{k=1}^{\rho} |c_k|^2 \begin{bmatrix} 1 \\ e^{j\omega_k} \\ \vdots \\ e^{jp\omega_k} \end{bmatrix} \begin{bmatrix} 1 \\ e^{j\omega_k} \\ \vdots \\ e^{jp\omega_k} \end{bmatrix}^H, \qquad (62)$$

i.e., a positive multiple of the identity plus a rank-ρ positive semidefinite Toeplitz matrix. If the line spectrum has Hermitian symmetry, the signal is real and the covariance matrix is symmetric. To fit a covariance matrix $R = \mathcal{T}(r)$ to observed data, we introduce variables $t = \sigma^2$ and $y = r - te$, and solve a convex problem

$$\begin{array}{ll} \text{minimize} & y_0 + \gamma \tilde{f}(y + te) \\ \text{subject to} & y \in K^*. \end{array} \qquad (63)$$

The second term in the objective measures the quality of the fit of the matrix $\mathcal{T}(y)+tI = \mathcal{T}(y+te)$ to the observed data. The first term in the objective is a multiple of the trace of $\mathcal{T}(y)$ and is added to encourage low-rank solutions. The coefficient γ is a positive regularization parameter. The dual of this problem can be written as

$$\begin{array}{ll} \text{maximize} & -\gamma \tilde{f}^*((x-e)/\gamma) \\ \text{subject to} & x \in K, \quad x_0 = 1, \end{array} \qquad (64)$$

where $\tilde{f}^*$ is the conjugate of $\tilde{f}$. If $\tilde{f}^*$ is differentiable, this is of the form (59) with $f(x) = \gamma \tilde{f}^*((x-e)/\gamma)$.

In the example we take a quadratic penalty function $\tilde{f}(r) = \|\mathcal{T}(r) - R_s\|_F^2$, where R_s is a sample covariance matrix. With this choice, $\tilde{f}^*$ is quadratic. The sample covariance matrix is constructed from $N = 150$ samples of a time series $s(t)$ of the form (61), shown in Figure 2. We take $\rho = 4$, and the frequencies ω_k and magnitudes $|c_k|$ indicated with red circles in Figure 3. The noise is Gaussian white noise with variance $\sigma^2 = 64$. The sample covariance matrix of order $p+1 = 30$ is constructed as $R_s = HH^T/(N-p)$ where H is the $(p+1) \times (N-p)$ Hankel matrix with $s(1), \ldots, s(N-p)$ in its first row.

Figure 3 shows the result for $\gamma = 2 \cdot 10^{-4}$. As can be seen, the recovered spectrum is quite accurate. The estimated noise variance σ^2 is 77.2.

Figure 4 shows the error versus iteration number. The relative optimality gap is computed as $(f(x^k) - f^{\text{opt}})/|f^{\text{opt}}|$, where $f(x)$ denotes the negative of the dual objective value in (64) and f^{opt} is the optimal value computed by CVX [49]. The error decreases roughly as $1/k^2$.

B. Euclidean projection on nonnegative polynomials

To evaluate the complexity for large p, we test the generalized proximal gradient method on a family of test problems (59) with $f(x) = \sum_{k=1}^{p} (x_k - a_k)^2$. This problem arises in signal processing, as the problem of finding the normalized autocorrelation sequence closest to a given sequence [3], [6].

The experiment was performed on an Intel Core i5-2410M 2.30GHz CPU with 6GB RAM and 64-bit operating system, using MATLAB version 7.12 (R2011a). The initial stepsize is $\tau_0 = 10/p$. The monotone search strategy in the Appendix (with $\beta = 2$) is used. In most problems less than five search

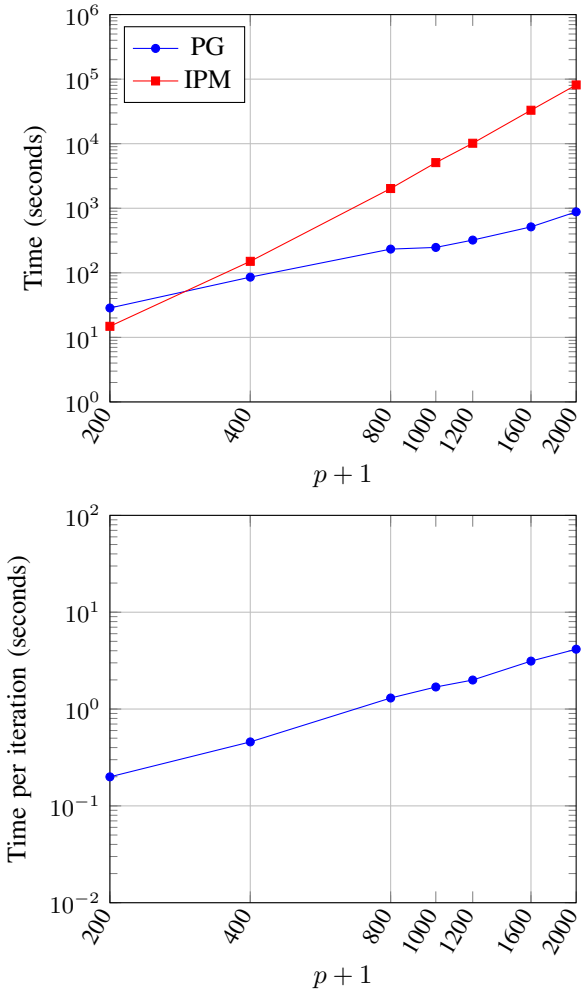

Fig. 5. *Top.* Time for proximal gradient method and general-purpose interior-point methods (IPM) versus problem size. *Bottom.* Time per iteration for the proximal gradient method.

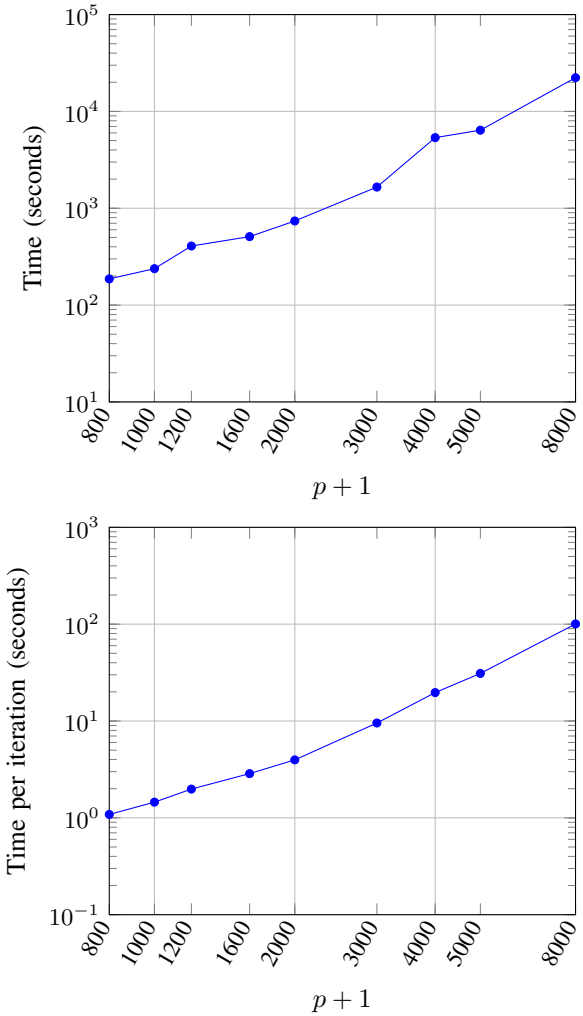

Fig. 6. Time for the proximal gradient method versus problem size.

steps during the first few iterations of the algorithm were needed.

In Figure 5 we compare the complexity of the generalized proximal gradient method (60) with general-purpose interior-point solvers called via CVX. The problem instances are randomly generated, with a from the normal distribution $N(0, I)$. For the first three data points ($p+1 = 200, 400, 800$), SDPT3 [50] was used as the interior-point method. Each of these data points is an average over 10 instances. For $p+1 = 1000$ and higher, SeDuMi [51] with the low-precision option was used. The first three of these data points are averages over five instances. For $p+1 = 2000$, only one instance was used. The blue curve is the total time for the proximal gradient method, averaged over the same instances as the interior-point solvers. The iteration was terminated when the relative suboptimality was less than 10^{-4}. The CVX solution was used to evaluate the suboptimality. The number of iterations for the interior-point solvers was generally between 10 and 30, and for the proximal gradient method between 100 and 200. On average about 10 Newton iterations were sufficient to evaluate the generalized projections. From Figure 5, it can be observed that the proximal gradient method exhibits a complexity under $O(p^2)$, whereas the SDP solvers have a complexity close to $O(p^4)$.

In Figure 6 we show results for larger problems of size up to 8000. Each data point is an average over five instances, and the iteration was terminated when the relative improvement in the cost function, defined as $|\min_{i<k} f(x^i) - f(x^k)|/\min_{i\leq k} f(x^i)$, was below 10^{-6}.

VII. Conclusion

We discussed a generalized proximal operator for the cone of nonnegative trigonometric polynomials, based on the Itakura–Saito distance. Projections in this distance have a complexity that is roughly quadratic in the degree of the polynomial. Proximal algorithms based on the generalized distance therefore scale better than standard (Euclidean) proximal algorithms, which require eigenvalue decompositions, and interior-point methods, which have a complexity that is cubic or higher.

Appendix

The Appendix describes the accelerated proximal gradient method used in the experiments, including a convergence

proof. The proof follows [18] and is included to clarify where our assumptions on the problem and the Bregman distance are needed. These conditions are slightly weaker than the ones stated in [18, p.17]. The proof also justifies the third parameter selection strategy discussed below.

We consider an optimization problem

$$\text{minimize} \quad F(x) = f(x) + g(x), \tag{65}$$

in which the objective is split as a sum of two convex functions. We assume that $\emptyset \neq \operatorname{dom} g \subseteq \operatorname{dom} f$ and that f is differentiable with a Lipschitz continuous gradient on $\operatorname{dom} g$, *i.e.*, there exists a constant L such that

$$f(x) \leq f(y) + \langle \nabla f(y), x - y \rangle + \frac{L}{2}\|x - y\|^2 \tag{66}$$

for all $x, y \in \operatorname{dom} g$. In addition, we assume that d_h is a Bregman distance with kernel h, and that for every a and every $v \in \operatorname{int}(\operatorname{dom} h)$, the generalized proximal operator $\operatorname{prox}^h_{\tau g}(\tau a, v)$ defined in (54), is well defined, *i.e.*, the optimization problem

$$\text{minimize} \quad \langle a, x \rangle + g(x) + \frac{1}{\tau} d_h(x, v) \tag{67}$$

has a unique solution in $\operatorname{dom} g \cap \operatorname{int}(\operatorname{dom} h)$. Here τ is a positive proximal stepsize. We also assume that

$$d_h(x, y) \geq \frac{1}{2}\|x - y\|^2 \tag{68}$$

for all $x \in \operatorname{dom} g \cap \operatorname{dom} h$ and $y \in \operatorname{dom} g \cap \operatorname{int}(\operatorname{dom} h)$. The norm on the right-hand side of (68) is the same norm as in (66). Finally, we assume that the problem (65) is solvable and has a solution $x^\star \in \operatorname{dom} g \cap \operatorname{dom} h$.

The following algorithm is IGA in [17] and Algorithm 1 in [18]. We start at $x^0 = v^0 \in \operatorname{dom} g \cap \operatorname{int}(\operatorname{dom} h)$ and run the iteration

$$\begin{aligned} y^k &= (1 - \theta_k) x^{k-1} + \theta_k v^{k-1} && \text{(69a)} \\ v^k &= \operatorname{prox}^h_{\tau_k g}(\tau_k \nabla f(y^k), v^{k-1}) && \text{(69b)} \\ x^k &= (1 - \theta_k) x^{k-1} + \theta_k v^k. && \text{(69c)} \end{aligned}$$

Suitable choices for the parameters $\theta_k \in [0, 1]$ and $\tau_k > 0$ are discussed below. Since the minimizer v^k in step (69b) is in the convex set $\operatorname{dom} g \cap \operatorname{int}(\operatorname{dom} h)$, all iterates y^k, v^k, x^k are in $\operatorname{dom} g \cap \operatorname{int}(\operatorname{dom} h)$. The update in the second step (69b) is therefore well defined at all iterations.

We discuss three strategies for choosing θ_k and τ_k. The first option requires knowledge of L, the Lipschitz constant in (66) with respect to a norm that also satisfies (68). Several strategies have been proposed to avoid this and replace L with an adaptively adjusted estimate λ_k [18], [42], [52]–[54]. The second and third methods below are examples of this.

In each method we will choose $\theta_1 = 1$, $\theta_k \in (0, 1)$ for $k > 1$, and $\tau_k > 0$ subject to the two conditions:

$$\begin{aligned} F(x^k) \leq (1 - \theta_k) F(x^{k-1}) + \theta_k (g(v^k) + f(y^k) \\ + \langle \nabla f(y^k), v^k - y^k \rangle + \frac{1}{\tau_k} d_h(v^k, v^{k-1})), \end{aligned} \tag{70}$$

and

$$\tau_k (1 - \theta_k) \theta_{k-1} \leq \tau_{k-1} \theta_k. \tag{71}$$

We will see that these conditions imply that

$$F(x^k) - F(x^\star) \leq \frac{\theta_k}{\tau_k} d_h(x^\star, x^0). \tag{72}$$

Each of the following three parameter selection methods satisfies (70) and (71), with $\theta_k/\tau_k = O(1/k^2)$.

a) Known Lipschitz constant: We choose $\tau_k = 1/(L\theta_k)$ and a sequence θ_k that satisfies $\theta_1 = 1$ and

$$(1 - \theta_k)\theta_{k-1}^2 \leq \theta_k^2, \quad k > 1. \tag{73}$$

A simple choice is $\theta_k = 2/(k + 1)$. The sequence that decreases most quickly, subject to the constraint (73), is obtained by imposing equality in (71). This gives the recursion

$$\theta_k = \frac{-\theta_{k-1}^2 + \sqrt{\theta_{k-1}^4 + 4\theta_{k-1}^2}}{2}.$$

To show that (70) holds, we apply (66) with $x = x^k$ and $y = y^k$, substitute (69c) for x^k on the right-hand side, simplify the argument of the norm using (69a), and apply (68) to obtain

$$\begin{aligned} f(x^k) \leq (1 - \theta_k)(f(y^k) + \langle \nabla f(y^k), x^{k-1} - y^k \rangle) \\ + \theta_k (f(y^k) + \langle \nabla f(y^k), v^k - y^k \rangle + \frac{1}{\tau_k} d(v^k, v^{k-1})). \end{aligned}$$

The inequality (70) now follows from convexity of f and Jensen's inequality for g applied to (69c).

b) Monotonic search: This is the strategy of [18], [42]. We choose a fixed sequence θ_k that satisfies (73), as in the previous strategy. We choose $\lambda_0 > 0$, and at iteration k choose for λ_k the smallest element of $\{\beta^i \lambda_{k-1} \mid i = 0, 1, 2, \ldots\}$, for which $\tau_k = 1/(\lambda_k \theta_k)$ satisfies (70). Here $\beta > 1$.

The inequality (71) holds because

$$\tau_k \theta_k = 1/\lambda_k \leq 1/\lambda_{k-1} = \tau_{k-1}\theta_{k-1}$$

and (73) holds. The procedure guarantees that $\lambda_k \leq \lambda_{\max} = \max\{\lambda_0, \beta L\}$ because, as shown above, (70) holds for $\theta_k \tau_k \leq 1/L$. Therefore

$$\frac{\theta_k}{\tau_k} \leq \theta_k^2 \lambda_{\max} \leq \frac{4\lambda_{\max}}{(k+1)^2} = O(\frac{1}{k^2}).$$

In this method, testing a candidate λ_k requires the evaluation of the generalized proximal operator in step (69b), and evaluations of $f(x^k)$, $g(x^k)$, $g(v^k)$, and $d_h(v^k, v^{k-1})$. These function values are needed to verify whether the inequality (70) holds.

c) Non-monotonic search: The third method does not force λ_k to be monotonically increasing as in the second method. At each iteration, choose some $\hat{\lambda}_k > 0$, and take the smallest λ_k in $\{\beta^i \hat{\lambda}_k \mid i = 0, 1, 2, \ldots\}$ that satisfies (70) with θ_k defined as the positive root of

$$\lambda_k \theta_k^2 = \lambda_{k-1}\theta_{k-1}^2(1 - \theta_k),$$

and $\tau_k = 1/(\theta_k \lambda_k)$.

Lipschitz continuity of ∇f guarantees that (70) holds if $\lambda_k = 1/(\theta_k \tau_k) \geq L$. Therefore the selected parameter satisfies $\lambda_k \leq \max\{\hat{\lambda}_k, \beta L\}$. The second condition (71) is satisfied by construction of θ_k. Finally, it can be shown that $\theta_k/\tau_k = O(1/k^2)$ [52, lemma 2.2]. The steps in this method are more expensive than in the second method. When testing

a candidate λ_k, we also change θ_k and therefore y^k, so we need to recompute the $f(y^k)$ and $\nabla f(y^k)$.

We now prove the inequality (72). We will need the following lemma [18, proposition 1]. If $\hat{x} \in \operatorname{int}(\operatorname{dom} h)$ is a solution of (67), then for all $x \in \operatorname{dom} g \cap \operatorname{dom} h$,

$$\begin{aligned}&\langle a, \hat{x}\rangle + g(\hat{x}) - \langle a, x\rangle - g(x)\\ &\qquad \leq \frac{1}{t}(d_h(x, v) - d_h(\hat{x}, v) - d_h(x, \hat{x})).\end{aligned} \tag{74}$$

Suppose (70) holds. By definition, v^k satisfies an inequality of the form (74), *i.e.*, for $x \in \operatorname{dom} g \cap \operatorname{dom} h$,

$$\begin{aligned}&\langle \nabla f(y^k), v^k\rangle + g(v^k) - \langle \nabla f(y^k), x\rangle - g(x)\\ &\quad \leq \frac{1}{\tau_k}(d_h(x, v^{k-1}) - d_h(v^k, v^{k-1}) - d_h(x, v^k)).\end{aligned}$$

Evaluating this at $x = x^\star$ and combining the result with (70) gives

$$\begin{aligned}F(x^k) - (1-\theta_k)F(x^{k-1}) + \frac{\theta_k}{\tau_k}(d_h(x^\star, v^k) - d_h(x^\star, v^{k-1}))\\ \leq\ \theta_k(f(y^k) + \langle \nabla f(y^k), x^\star - y^k\rangle + g(x^\star))\\ \leq\ \theta_k F(x^\star).\end{aligned}$$

Re-arranging gives

$$\begin{aligned}&\frac{\tau_k}{\theta_k}(F(x^k) - F(x^\star)) + d_h(x^\star, v^k)\\ &\quad \leq \frac{(1-\theta_k)\tau_k}{\theta_k}(F(x^{k-1}) - F(x^\star)) + d_h(x^\star, v^{k-1}).\end{aligned}$$

Combining these inequalities recursively using (71) gives (72).

REFERENCES

[1] S.-P. Wu, S. Boyd, and L. Vandenberghe, "FIR filter design via spectral factorization and convex optimization," in *Applied and Computational Control, Signals, and Circuits*, B. Datta, Ed. Birkhauser, 1998, vol. 1, pp. 215–245.

[2] P. Stoica, T. McKelvey, and J. Mari, "MA estimation in polynomial time," *IEEE Transactions on Signal Processing*, vol. 48, no. 7, pp. 1999–2012, Jul. 2000.

[3] B. Dumitrescu, I. Tabus, and P. Stoica, "On the parametrization of positive real sequences and MA parameter estimation," *IEEE Transactions on Signal Processing*, vol. 49, no. 11, pp. 2630–9, 2001.

[4] T. N. Davidson, Z.-Q. Luo, and J. F. Sturm, "Linear matrix inequality formulation of spectral mask constraints," *IEEE Transactions on Signal Processing*, vol. 50, no. 11, pp. 2702–2715, 2002.

[5] B. Alkire and L. Vandenberghe, "Convex optimization problems involving finite autocorrelation sequences," *Mathematical Programming Series A*, vol. 93, pp. 331–359, 2002.

[6] B. Dumitrescu, *Positive Trigonometric Polynomials and Signal Processing Applications*. Springer, 2007.

[7] T. T. Georgiou and A. Lindquist, "A convex optimization approach to ARMA modeling," *IEEE Transactions on Automatic Control*, vol. 53, pp. 1108–1119, 2008.

[8] M. Annergren, C. A. Larsson, H. Hjalmarsson, X. Bombois, and B. Wahlberg, "Application-oriented input design in system identification optimal input design for control," *IEEE Control Systems Magazine*, vol. 37, pp. 31–56, 2017.

[9] E. J. Candès and C. Fernandez-Granda, "Super-resolution from noisy data," *Journal of Fourier Analysis and Applications*, vol. 19, pp. 1229–1254, 2013.

[10] G. Tang, B. N. Bhaskar, P. Shah, and B. Recht, "Compressed sensing off the grid," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7465–7490, 2013.

[11] E. J. Candès and C. Fernandez-Granda, "Towards a mathematical theory of super-resolution," *Communications of Pure and Applied Mathematics*, vol. 67, no. 6, pp. 906–956, 2014.

[12] Y. Genin, Y. Hachez, Y. Nesterov, and P. Van Dooren, "Optimization problems over positive pseudopolynomial matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 25, no. 1, pp. 57–79, 2003.

[13] T. Roh and L. Vandenberghe, "Discrete transforms, semidefinite programming, and sum-of-squares representations of nonnegative polynomials," *SIAM Journal on Optimization*, vol. 16, no. 4, pp. 939–964, 2006.

[14] Y. Nesterov, *Introductory Lectures on Convex Optimization*. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2004.

[15] A. Beck and M. Teboulle, "Gradient-based algorithms with applications to signal recovery," in *Convex Optimization in Signal Processing and Communications*, Y. Eldar and D. Palomar, Eds. Cambridge University Press, 2009.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[17] A. Auslender and M. Teboulle, "Interior gradient and proximal methods for convex and conic optimization," *SIAM Journal on Optimization*, vol. 16, no. 3, pp. 697–725, 2006.

[18] P. Tseng, "On accelerated proximal gradient methods for convex-concave optimization," 2008.

[19] A. Beck and M. Teboulle, "Mirror descent and nonlinear projected subgradient methods for convex optimization," *Operations Research Letters*, vol. 31, pp. 167–175, 2003.

[20] C. J. Demeure and C. T. Mullis, "A Newton-Raphson method for moving-average spectral factorization using the Euclid algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 18, pp. 1697–1709, 1990.

[21] G. S. Ammar and W. B. Gragg, "Superfast solution of real positive definite Toeplitz systems," *SIAM Journal on Matrix Analysis and Applications*, vol. 9, no. 1, pp. 61–76, 1988.

[22] M. Stewart, "A superfast Toeplitz solver with improved numerical stability," *SIAM Journal on Matrix Analysis and Applications*, vol. 25, no. 3, pp. 669–693, 2003.

[23] L. L. Scharf, *Statistical Signal Processing*. Addison-Wesley, 1991.

[24] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing. Principles, Algorithms, and Applications*, 3rd ed. Prentice-Hall, 1996.

[25] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

[26] A. Vieira and T. Kailath, "On another approach to the Schur-Cohn criterion," *IEEE Transactions on Circuits and Systems*, vol. 24, pp. 218–220, 1977.

[27] Z. Vostrý, "New algorithm for polynomial spectral factorization with quadratic convergence. Part I," *Kybernetika*, vol. 11, pp. 415–422, 1975.

[28] C. I. Byrnes, S. V. Gusev, and A. Lindquist, "A convex optimization approach to the rational covariance extension problem," *SIAM Journal on Control and Optimization*, vol. 37, no. 1, pp. 211–229, 1998.

[29] ——, "From finite covariance windows to modeling filters: a convex optimization approach," *SIAM Review*, vol. 4, no. 4, pp. 645–675, 2001.

[30] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*. Prentice Hall, 1993.

[31] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, revised ed. Wellesley-Cambridge Press, 1997.

[32] G. Wilson, "Factorization of the covariance generating function of a pure moving average process," *SIAM J. on Numerical Analysis*, vol. 6, pp. 1–7, 1969.

[33] J. W. McLean and H. J. Woerdeman, "Spectral factorizations and sums of squares representations via semidefinite programming," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 3, pp. 646–655, 2001.

[34] Y. Hachez, "Convex optimization over non-negative polynomials: Structured algorithms and applications," Ph.D. dissertation, Université catholique de Louvain, 2003.

[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge: Cambridge University Press, 2004.

[36] R. T. Rockafellar, *Convex Analysis*. Princeton: Princeton Univ. Press, 1970.

[37] A. H. Gray and J. D. Markel, "Distance measures for speech processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-24, pp. 380–391, 1976.

[38] R. Gray, A. Buzo, A. Gray, and Y. Matsuyama, "Distortion measures for speech processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 367–376, 1980.

[39] M. Basseville, "Distance measures for signal processing and pattern recognition," *Signal Processing*, vol. 18, pp. 349–369, 1989.

[40] T. T. Georgiou, J. Karlsson, and M. S. Takyar, "Metrics for power spectra: an axiomatic approach," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 859–867, 2009.

[41] M. Basseville, "Divergence measures for statistical data processing—an annotated bibliography," *Signal Processing*, vol. 93, pp. 621–633, 2013.

[42] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[43] P. L. Combettes and J.-C. Pesquet, "A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 564–574, 2007.

[44] J. P. Boyle and R. L. Dykstra, "A method for finding projections onto the intersection of convex sets in Hilbert spaces," in *Advances in Order Restricted Statistical Inference*, ser. Lecture Notes in Statistics, R. Dykstra, T. Robertson, and F. T. Wright, Eds., vol. 37. Springer-Verlag, 1986, pp. 28–47.

[45] Y. Censor and S. A. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*, ser. Numerical Mathematics and Scientific Computation. New York: Oxford University Press, 1997.

[46] H. H. Bauschke and A. S. Lewis, "Dykstra's algorithm with Bregman projections: A convergence proof," *Optimization: A Journal of Mathematical Programming and Operations Research*, vol. 48, no. 4, pp. 409–427, 2000.

[47] H.-H. Chao and L. Vandenberghe, "Semidefinite representations of gauge functions for structured low-rank matrix decomposition," *SIAM Journal on Optimization*, vol. 27, pp. 1362–1389, 2017.

[48] P. Stoica and R. L. Moses, *Introduction to Spectral Analysis*. London: Prentice Hall, 1997.

[49] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[50] K. C. Toh, R. H. Tütüncü, and M. J. Todd, *SDPT3 version 3.02. A Matlab software for semidefinite-quadratic-linear programming*, 2002, available from `www.math.nus.edu.sg/~mattohkc/sdpt3.html`.

[51] J. F. Sturm, "Using SEDUMI 1.02, a Matlab toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11-12, pp. 625–653, 1999.

[52] O. Güler, "New proximal point algorithm for convex minimization," *SIAM Journal on Optimization*, vol. 2, no. 4, pp. 649–664, 1992.

[53] Y. Nesterov, "Gradient methods for minimizing composite functions," *Mathematical Programming, Series B*, vol. 140, pp. 125–161, 2013.

[54] K. Scheinberg, D. Goldfarb, and X. Bai, "Fast first-order methods for composite convex optimization with backtracking," *Foundations of Computational Mathematics*, vol. 14, pp. 389–417, 2014.