A LEVINSON-GALERKIN ALGORITHM FOR REGULARIZED TRIGONOMETRIC APPROXIMATION*

THOMAS STROHMER†

Abstract. Trigonometric polynomials are widely used for the approximation of a smooth function from a set of nonuniformly spaced samples. If the samples are perturbed by noise, a good choice for the polynomial degree of the trigonometric approximation becomes an essential issue to avoid overfitting and underfitting of the data. Standard methods for trigonometric least squares approximation assume that the degree for the approximating polynomial is known a priori, which is usually not the case in practice. We derive a multi-level algorithm that recursively adapts to the least squares solution of suitable degree. We analyze under which conditions this multi-approach yields the optimal solution. The proposed algorithm computes the solution in at most $\mathcal{O}(rM+M^2)$ operations (M being the polynomial degree of the approximation and r being the number of samples) by solving a family of nested Toeplitz systems. It is shown how the presented method can be extended to multivariate trigonometric approximation. We demonstrate the performance of the algorithm by applying it in echocardiography to the recovery of the boundary of the Left Ventricle of the heart.

Key words. trigonometric approximation, Toeplitz matrix, Levinson algorithm, multi-level method.

AMS subject classifications. 65T10, 42A10, 65D10, 65F10

1. Introduction. The necessity of recovering a function from a finite set of nonuniformly spaced measurements arises in areas as diverse as digital signal processing, geophysics, spectroscopy or medical imaging. the measurements $\{s_j\}_{j=1}^r$ are often distorted by several kinds of error. Hence a complete reconstruction of the function from the perturbed data $s_j^\varepsilon = s_j + \nu_j$ is not possible. Often the function to be reconstructed is smooth, in which case a trigonometric polynomial of relatively low degree (compared to the possibly huge number of samples) can provide a good approximation to the function. This trigonometric approximation may be found by solving the least squares problem

$$\min_{p\in \boldsymbol{P}_M} \sum_{j=1}^{r} |p(x_j) - s_j^\varepsilon|^2 w_j, \tag{1.1}$$

where $w_j > 0$ are weights and $\boldsymbol{P}_M$ is the space of trigonometric polynomials of degree less than or equal to M.

Many efficient algorithms have been developed to solve (1.1), e.g., see the articles [22, 7, 25, 11, 10]. But surprisingly little attention has been paid to the problem of how to control the smoothness of the approximation in order to avoid overfitting and underfitting of the data. An adaptation of the smoothness of the approximation can

*The author has been supported by project S7001-MAT, Schrödinger fellowship J01388-MAT of the Austrian Science foundation FWF, and NSF DMS grant 9973373.

†Department of Mathematics, University of California, Davis, CA 95616-8633, USA; strohmer@math.ucdavis.edu.

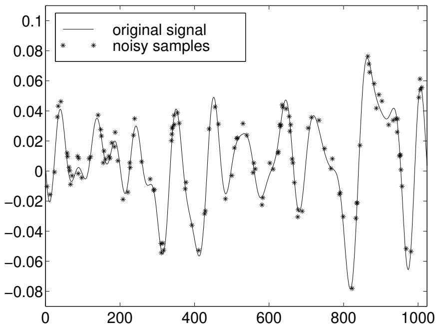
be achieved of instance by providing a suitable upper bound for the degree M of the space $\boldsymbol{P}_M$ in (1.1). In most of the aforementioned algorithms a necessary requirement to get useful results in applications is that a good a priori guess of the degree of the trigonometric approximation is available. However a priori it is not clear what is a suitable degree for the solution, in terms of how to choose a reasonable degree M when solving (1.1). Determining M by "trial and error" is certainly not a satisfying alternative.

It is the goal of this paper to derive an efficient algorithm that computes the trigonometric approximation which provides the "optimal" balance between fitting the given data and preserving smoothness of the solution. Here optimality is meant in the sense that the solution has minimal degree among all trigonometric polynomials that satisfy a certain least squares criterion. The algorithm recursively adapts to the least squares approximation of optimal degree by solving a family of nested Toeplitz systems in at most $\mathcal{O}(Mr + M^2)$ operations,

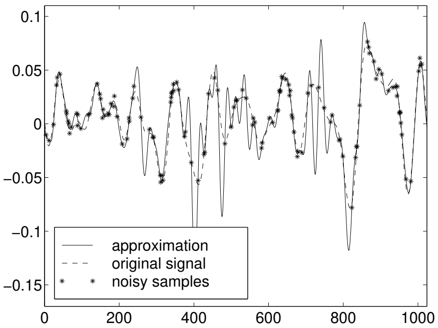
If the data $\{s_j^\varepsilon\}_{j=1}^r$ were (i) unperturbed and (ii) stem from sampling a trigonometric polynomial (with degree less than $r/2$), then the solution of (1.1) would automatically have the appropriate degree, since the original function could be completely recovered in this case. However the assumptions (i) and (ii) are rarely met in applications and controlling the smoothness of the solution becomes essential to avoid overfitting and underfitting of the data. If we choose the upper bound for the degree in (1.1) too large, the solution will almost always take on the maximal possible degree, hence being too wiggly and picking up too much noise (overfit), see also Figure 1.1 (a)–(b). In the extreme case $2M + 1 = r$ we will get an interpolating polynomial, mostly with strong oscillations and far away from approximating the function between the given samples. On the other hand, if we choose M too small, then the approximation will be very smooth but poorly fitting the given data (underfit). Figure 1.1(c) illustrates this behavior. The "regularized" trigonometric approximation obtained by the algorithm proposed in this paper – to which we will refer as *Levinson-Galerkin algorithm* – is shown in Figure 1.1(d).

The paper is organized as follows. In Sections 2 and 4 we present the main results, including the Levinson-Galerkin algorithm and a theoretical analysis that clarifies under which conditions this algorithm provides optimal results. In Section 3 we show how properly chosen weights can be used as simple but efficient tool to precondition the least squares problem. Some aspects of extending the algorithm to multivariate trigonometric polynomials are discussed in 5. In Section 6 we present some applications in echocardiography.

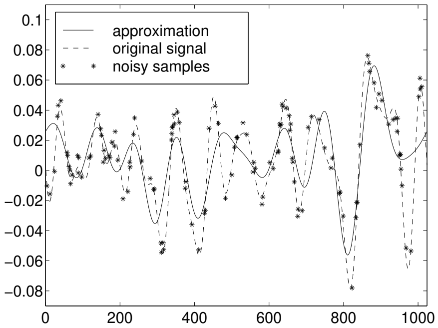
Before we proceed we introduce some notation and conventions. The inner product is denoted by $\langle \cdot, \cdot \rangle$, and the conjugate transpose of a matrix A by A^*. The space



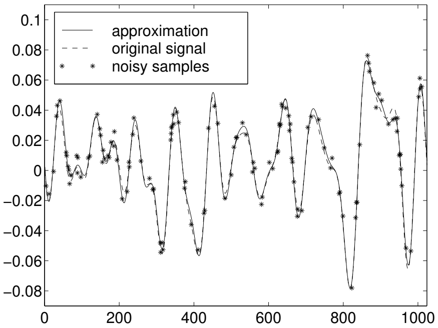
(a) Original signal and perturbed samples



(b) Least squares approximation using a too large upper bound for the degree of the polynomial (overfit)



(c) Least squares approximation using a too small upper bound for the degree of the polynomial (underfit)



(d) Regularized approximation by proposed Levinson-Galerkin algorithm

FIG. 1.1. *Controlling the smoothness of the solution is essential for trigonometric approximation from perturbed data in order to avoid overfitting and underfitting of the data. The proposed Levinson-Galerkin algorithm automatically adapts to the least squares solution of optimal degree.*

of trigonometric polynomial of degree equal to or less than M is defined as

$$\boldsymbol{P}_M = \left\{ p : p(x) = \sum_{k=-M}^{M} c_k e^{2\pi i k x} \right\} . \tag{1.2}$$

The norm of $p(x) = \sum_{k=-M}^{M} c_k e^{2\pi i k x} \in \boldsymbol{P}_M$ is given by

$$\|p\| = \left(\int_0^1 |p(x)|^2 \, dx \right)^{\frac{1}{2}} = \left(\sum_{k=-M}^{M} |c_k|^2 \right)^{\frac{1}{2}} = \|c\| , \tag{1.3}$$

where $c = \{c_k\}_{k=-M}^{M}$. In some applications it is advantageous to deal with complex-valued polynomials (see also Section 6), hence we do not restrict ourselves to the case of real-valued trigonometric approximation.

For $a = [a_{-M}, a_{-M+1}, \dots, a_{M-1}, a_M] \in \mathbb{C}^{2M+1}$ we define the orthogonal projections P_N by

$$P_N a = [0, \dots, 0, a_{-N}, a_{-N+1}, \dots, a_r, a_N, 0, \dots, 0] \tag{1.4}$$

for $N = 1, 2, \dots, M$ and identify the image of P_N with the $2N+1$-dimensional space $\mathbb{C}^{2N+1}$.

Let p^M and p^N be trigonometric polynomials of degree M and N respectively, with coefficients vectors $c^M \in \mathbb{C}^{2M+1}, c^N \in \mathbb{C}^{2N+1}$. If $N < M$, then we can always interpret p^N as polynomial of degree M by adding to appropriate number of zero-coefficients and by doing so we are embedding the vector c^N into a zero-padded vector of length $2M+1$. We will henceforth tacitly assume that such an embedding has been made, when we compute expressions such as $\|c^M - c^N\|$.

2. Multi-level least squares approximation. A standard method in numerical analysis to find the optimal balance between fitting the given data and preserving smoothness of the solution is to introduce a regularization parameter. The best value of this regularization parameter is then determined for instance by generalized cross validation [15] or via the L-curve [20]. Here we understand regularization not as a way to stabilize ill-conditioned problems, but in a broader context as a means of finding the best compromise between fitting a given set of data and preserving smoothness of the solution. As we will see, in our case it is not necessary to introduce an additional parameter, since we can regularize the smoothness of the solution by varying the parameter M of the space $\boldsymbol{P}_M$ in which we are searching for the solution of (1.1).

For the derivation of the algorithm we consider first following situation. Assume $p_*(x) = \sum_{k=-N_*}^{N_*} (c_*)_k e^{2\pi i k x} \in \boldsymbol{P}_{N_*}$ and let $s^\varepsilon = \{s_j^\varepsilon\}_{j=1}^r, 2M+1 \le r$ with $s_j^\varepsilon = s_j + \nu_j = p_*(x_j) + \nu_j$ be given noisy samples satisfying

$$\|s^\varepsilon - s\|^2 \le \varepsilon \|s^\varepsilon\|^2. \tag{2.1}$$

For convenience we assume that r, the number of samples, is odd.

The aim is to approximate p_* from the data $\{s_j^\varepsilon\}_{j=1}^r$. Let us first assume that we already know that we are searching for our least squares solution in the space $\boldsymbol{P}_{N_*}$. In this case the coefficient vector of the polynomial that solves (1.1) is the least squares solution of

$$W V_{N_*} c = W s^\varepsilon \, , \tag{2.2}$$

where V_{N_*} is a $r \times (2N_* + 1)$ Vandermonde matrix with entries

$$(V_M)_{j,k} = e^{2\pi i k x_j}, \qquad j = 1, \dots, r, \; k = -N_*, \dots, N_* \tag{2.3}$$

and $W = \operatorname{diag}(\{\sqrt{w_j}\})$.

We will discuss the role and specific choice of the weights in more detail in Section 3. To reduce the notational burden we absorb the weight matrix W in the Vandermonde matrix and in the sampling values. Thus for given degree, M say, we consider the linear system of equations

$$V_M c = s^\varepsilon \tag{2.4}$$

where V_M is now the $r \times (2M+1)$ "weighted Vandermonde" matrix. We will denote the least squares solution of (2.4) by $c^{(M)} = \{c_k^{(M)}\}_{k=-M}^{M}$ and the corresponding polynomial is $p^{(M)}(x) = \sum_{k=-M}^{M} c_k^{(M)} e^{2\pi i k x}$.

Since in general we do not know the optimal degree or *level* M of the space $\boldsymbol{P}_M$ in which we should solve the least squares problem, the situation becomes considerably more complicated. If we want to solve (1.1) under the information (2.1) without knowing the degree of the polynomial, one may argue that we have to accept any trigonometric polynomial $p(x) = \sum_{k=-N}^{N} c_k e^{2\pi i k x}$ with $\|V_N c - s^\varepsilon\| \le \varepsilon \|s^\varepsilon\|$ as an approximate solution to p_*, since it is compatible with the only knowledge we have on the data.

In general there may be infinitely many such polynomials, which raises the questions of how to find a polynomial p that yields a small approximation error $\|p_* - p\|$ and at the same time can be computed efficiently.

2.1. A multi-level algorithm and an efficient stopping criterion. The heuristic considerations above suggest following approach.

ALGORITHM 1. *Set $N = 0$ and solve $V_0 c^{(0)} = s^\varepsilon$. If $c^{(0)}$ satisfies the condition $\|V_0 c^{(0)} - s^\varepsilon\| \le \varepsilon \|s^\varepsilon\|$, take $c^{(0)}$ as solution. Otherwise set $N = N+1$ and solve*

$$V_N c^{(N)} = s^\varepsilon\,, \tag{2.5}$$

until $c^{(N)}$ satisfies for the first time the stopping criterion

$$\|V_N c^{(N)} - s^\varepsilon\| \le \varepsilon \|s^\varepsilon\|. \tag{2.6}$$

at some level $N = N_0$. Set $c^{(N_0)} = c^{(N)}$. The approximation to p_* is then $p^{(N_0)}(x) = \sum_{k=-M}^{M} c_k^{(N_0)} e^{2\pi i k x}$.

The stopping criterion (2.6) is well-defined, since it is definitely satisfied for $N = (r-1)/2$, in which case the left side in (2.6) equals 0. Thus Algorithm 1 selects among all least squares solutions $p^{(N)}, N = 0, \ldots, (r-1)/2$ that polynomial with minimal degree.

Algorithm 1 and stopping criterion (2.6) can be justified by following theoretical considerations.

One readily verifies that the matrices $V_N, N = 0, \ldots, (r-1)/2$ satisfy the relations:

(i) there exists a left-inverse V_N^+ such that

$$V_N^+ V_N = I_N, \quad \text{with } V_N^+ = (V_N^* V_N)^{-1} V_N^*, \tag{2.7}$$

where I_N is the identity matrix on $\mathbb{C}^{2N+1}$.

(ii) Let $a \in \mathbb{C}^{2M+1}$ be the coefficient vector of some $p \in \boldsymbol{P}_M$. Then

$$V_N a = V_M a \qquad \text{for all } N > M,\ a \in \mathbb{C}^{2M+1}. \tag{2.8}$$

In (ii) we have made use of the fact that the coefficient vector a can be interpreted as coefficient vector of a polynomial of degree N by extending it to a vector of length $2N+1$ via zero-padding. The matrix-vector multiplication $V_M a$ and equation (2.8) should be understood in this sense.

Lemma 2.1. *If $N \geq N_*$ then $c^{(N)}$ satisfies $\|V_N c^{(N)} - s^\varepsilon\| \leq \varepsilon \|s^\varepsilon\|$, hence stopping criterion* (2.6) *always becomes active at some level $N_0 \leq N_*$.*

Proof. Note that $V_N V_N^+$ is the orthogonal projection into range(V_N) and $s \in$ range$(V_{N_*}) \subseteq$ range(V_N) for $N_* \leq N$, hence $V_N V_N^+ s = s$. Therefore

$$\|V_N c^{(N)} - s^\varepsilon\|^2 = \|V_N V_N^+(s+\nu) - (s+\nu)\|^2 = \|\nu - V_N V_N^+ \nu\|^2 \tag{2.9}$$

$$= \|\nu\|^2 - \|V_N V_N^+ \nu\|^2 \leq \varepsilon^2 \|s^\varepsilon\|^2, \tag{2.10}$$

where we have used condition (2.1) in the last step. It follows from (2.10) that Algorithm 1 terminates at some level $N_0 \leq N_*$. □

The following lemma shows that from the viewpoint of numerical stability it is advisable to keep the level N of the space $\boldsymbol{P}_N$ in which we search for our solution as small as possible.

Lemma 2.2. *$cond(V_N^* V_N) \geq cond(V_M^* V_M)$ for $N \geq M$.*

Proof. Since

$$P_M (V_N^* V_N) P_M = V_M^* V_M \quad \text{for } M \leq N,$$

Cauchy's Interlace Theorem [16] implies that cond$(V_N^* V_N) \geq$ cond$(V_M^* V_M)$ for $N \geq M$. □

In the sequel we demonstrate that the fact that Algorithm 1 terminates at some level $\leq N_*$ is a desired property in many cases. We show that stopping criterion (2.6) is even optimum in a number of cases.

Let us first consider two special cases: (i) noisefree samples and (ii) uniformly spaced samples.

2.1.1. Noisefree samples. Any reasonable stopping criterion has to satisfy following lemma.

Lemma 2.3. *For noisefree data the stopping criterion* (2.6) *yields the exact solution.*

Proof. One readily verifies that Algorithm 1 terminates at level N_*. Hence for $N = N_*$:

$$\|p^{(N)} - p_*\| = \|c^{(N)} - c_*\| = \|V_N^+ s^\varepsilon - c_*\| = \|V_N^+ V_N c_* - c_*\| = 0, \tag{2.11}$$

since $V_N^+ V_N c_* = c_*$ for $N \geq N_*$. □

Lemmas 2.3 and 2.2 together show that stopping criterion (2.6) yields the optimum solution for noisefree data while providing maximum numerical stability.

2.1.2. Uniformly spaced samples. If the sampling points $x_j, j = 1, \ldots, r$ are uniformly spaced and we choose $w_j = 1/r$ as weights then a simple calculation shows that V_N is unitary on $\mathbb{C}^{2N+1}$, i.e., $V_N^* V_N = I_N$ for $N = 0, 1, \ldots, (r-1)/2$.

In this case

$$\|c_* - c^{(N)}\| = \|c_* - V_N^* s^\varepsilon\| = \|c_* - V_N^* V_{N_*} c_* - V_N^* \nu\|. \tag{2.12}$$

$N \geq N_*$ implies $V_N^* V_{N_*} = I_N$ and hence

$$\|c_* - c^{(N)}\| = \|V_N^* \nu\|. \tag{2.13}$$

Note that

$$\|V_N^* \nu\| = \langle V_N^* \nu, V_N^* \nu \rangle = \langle V_N V_N^* \nu, \nu \rangle = \|V_N V_N^* \nu\|, \tag{2.14}$$

since $V_N V_N^*$ is an orthogonal projection. Equation (2.14) yields

$$\|V_M^* \nu\| \leq \|V_N^* \nu\| \qquad \text{for } M \leq N, . \tag{2.15}$$

Consequently

$$\|c_* - c^{(M)}\| \leq \|c_* - c^{(N)}\| \qquad \text{if } N_* \leq M \leq N. \tag{2.16}$$

Thus for uniformly spaced samples any stopping criterion should terminate Algorithm 1 at the latest at $N = N_*$. Under a mild condition on the coefficients c_* we can show that the proposed stopping criterion provides the optimal solution among all least squares solutions.

PROPOSITION 2.4. *Assume that the samples are regularly spaced. Then the solution $p^{(N_0)}$ computed via Algorithm 1 satisfies*

$$\|p_* - p^{(N_0)}\| \leq \|p_* - p^{(N)}\| \qquad \textit{for all } N \geq N_*. \tag{2.17}$$

If furthermore p_* satisfies

$$\|(I_{N_*} - P_N) c_*\| \geq \|(I_{N_*} - P_N) V_{N_*}^* \nu\| \tag{2.18}$$

then

$$\|p_* - p^{(N_0)}\| \leq \|p_* - p^{(N)}\| \qquad \textit{for all } N. \tag{2.19}$$

Condition (2.18) is satisfied e.g., if all coefficients of p_* are larger than the relative noise level, i.e., $|c_k| \geq \varepsilon \|s^\varepsilon\|$.

Proof. Lemma 2.1 yields that $N_0 \leq N_*$, thus (2.16) implies (2.17).

To prove assertion (2.19) we only have to show that

$$\|c_* - c^{(N_0)}\| \leq \|c_* - c^{(N)}\| \qquad \text{for all } N < N_*$$

For $N < N_*$ note that $(c_* - V_N^* V_{N_*} c_*)$ is orthogonal to $V_N^* \nu$, since

$$\langle c_*, V_N^* \nu \rangle = \langle V_{N_*}^* V_{N_*} c_*, V_N^* \nu \rangle \tag{2.20}$$

$$= \langle V_{N_*} c_*, V_{N_*}^* V_N^* \nu \rangle = \langle V_{N_*} c_*, V_N V_N^* \nu \rangle, \tag{2.21}$$

hence

$$\langle c_* - V_N^* V_{N_*} c_*, V_N^* \nu \rangle = 0.$$

Therefore

$$\|c_* - c^{(N)}\|^2 = \|c_* - V_N^+ V_{N_*} c_* + V_N^* \nu\|^2 = \|c_* - V_N^+ V_{N_*} c_*\|^2 + \|V_N^* \nu\|^2 .$$

In order to prove $\|c_* - c^{(N_0)}\| \leq \|c_* - c^{(N)}\|$ for all $N < N_*$ we need to verify $\|c_* - V_N^+ V_{N_*} c_*\|^2 + \|V_N^* \nu\|^2 \geq \|V_{N_*}^* \nu\|^2$. Since

$$\|c_* - V_N^+ V_{N_*} c_*\|^2 = \sum_{|k|=N+1}^{N_*} |(c_*)_k|^2 = \|(I_N - P_N) c_*\|^2$$

and

$$\|(V_{N_*})^* \nu\|^2 - \|V_N^* \nu\|^2 = \sum_{|k|=N+1}^{N_*} |(V_{N_*}^* \nu)_k|^2 = \|(I_N - P_N) V_{N_*}^* \nu\|^2,$$

the result follows now from the assumption (2.18). □

REMARK: Proposition 2.4 shows that the least squares polynomial that gives the best approximation to p_* is not necessarily of degree N_*.

2.1.3. Noisy nonuniform samples. For noisy nonuniformly spaced data we observe that

$$\|p_* - p^{(N)}\| = \|c_* - c^{(N)}\| \leq \|c_* - V_N^+ V_{N_*} c_*\| + \|V_N^+ \nu\|,$$

and for $N \geq N_*$

$$\|p_* - p^{(N)}\| \leq \|V_N^+ \nu\|, \tag{2.22}$$

since $\|c_* - V_N^+ V_{N_*} c_*\| = 0$ in this case.

If V_N is not unitary then $\|V_N^+ \nu\|$ is not necessarily monotonically increasing with increasing level N. One can argue heuristically that since $\|V_N^+\|$ is increasing with increasing level N due to Lemma 2.2, we may fairly assume that $\|V_N^+ \nu\|$ will also increase (although not strictly monotonically). Also from the viewpoint of numerical stability it is reasonable to keep the level N small, since by Lemma 2.2 we know that $\text{cond}(V_N^* V_N) \geq \text{cond}(V_M^* V_M)$ for $N \geq M$. This together with (2.22) suggests to choose a stopping criterion which terminates at or before level N_*, which is guaranteed for stopping criterion (2.6) by Lemma 2.1.

We can conclude that the stopping criterion will provide excellent results if the noise level ε is small or if the condition number of $V_N^* V_N$ is small (which implies that V_N is approximately unitary). In order to verify the latter it is useful to have estimates for the condition number of $V_N^* V_N$. We will address this issue in Proposition 3.1.

2.2. A Toeplitz system and trigonometric approximation. Instead of directly solving $V_M c^{(M)} = s^\varepsilon$ it is more efficient in our case to consider the normal equations

$$V_M^* V_M^* c^{(M)} = V_M^* s^\varepsilon. \tag{2.23}$$

The reason is that from a numerical point of view the structural properties of the matrix $V_M^* V_M$ are much more attractive than those of V_M, which in turn leads to faster numerical algorithms, see also Section 4.

Set $T_M = V_M^* V_M$ then a simple calculation shows that the entries of the hermitian matrix T_M are

$$(T_M)_{k,l} = \sum_{j=1}^{r} w_j e^{2\pi i(k-l)x_j}, \qquad k,l = -M,\ldots,M. \tag{2.24}$$

T_M is a Toeplitz matrix, since the entries $(T_M)_{k,l}$ depend only on the difference $k-l$. Obviously T_M is invertible if $2M+1 \le r$.

Following result is just a reformulation of (2.23) together with relation (2.8), but since it plays a key role in Section 4 it is helpful to state it in detail (cf. also [18]).

THEOREM 2.5. *Given the sampling points* $0 \le x_1 < \ldots, x_r < 1$, *samples* $\{s_j^\varepsilon\}_{j=1}^N$, *positive weights* $\{w_j\}_{j=1}^r$ *and the degree* M *with* $2M+1 \le r$. *The polynomial* $p^{(M)} \in \boldsymbol{P}_M$ *that solves* (1.1) *is given by*

$$p^{(M)}(x) = \sum_{k=-M}^{M} c_m^{(M)} e^{2\pi i k x} \in \boldsymbol{P}_M\,. \tag{2.25}$$

where its coefficients $c_k^{(M)}$ *satisfy*

$$T_M c^{(M)} = b^{(M)} \ \in \mathbb{C}^{(2M+1)^2}, \tag{2.26}$$

with

$$b_k^{(M)} = \sum_{j=1}^{r} s_j^\varepsilon w_j e^{2\pi i k x_j} \qquad \text{for } |k| \le M, \tag{2.27}$$

and T_M *as defined in* (2.24).

3. Weights as simple preconditioner. Vandermonde matrices are known to be ill-conditioned, if the nodes x_j are clustered [13]. To improve the stability of the systems (2.4) and (2.26) we can use the weights as simple diagonal preconditioner. This leads to the problem of how to choose the weights w_j.

We propose to use the size of the area of the Voronoi region [23] associated with the sampling point x_j as weight w_j. In 1-D this reduces to

$$w_j = \frac{x_{j+1} - x_{j-1}}{2}\,. \tag{3.1}$$

This choice is motivated by following observations.

In this section we let V_N denote the Vandermonde matrix defined in (2.3) without weights. Let $p \in \boldsymbol{P}_M$ with coefficient vector c. Since

$$\langle T_M c, c\rangle = \langle WV_M c, WV_M c\rangle = \|WV_M c\|^2 = \sum_{j=1}^{r} |p(x_j)|^2 w_j, \tag{3.2}$$

the inequality

$$C_1\|c\|^2 \le \sum_{j=1}^{r} |p(x_j)|^2 w_j \le C_2\|c\|^2 \tag{3.3}$$

holds for all $c \in \mathbb{C}^{2M+1}$ with constants $C_1 = \lambda_{min}$ and $C_2 = \lambda_{max}$, where λ_{min} and λ_{max} denote the minimal and maximal eigenvalue, respectively, of T_M.

(i) The lower bound of T_M is mainly determined by the large gaps in the sampling set. Suppose there is a large gap in the sampling set and denote the interval corresponding to this gap by Γ (hence $x_j \notin \Gamma$). Choose a trigonometric polynomial $p \in \boldsymbol{P}_M$ which, like the prolate spheroidal functions, concentrates most of its energy in the interval Γ. Then the sampling values of p will not pick up any information about the main concentration of the polynomial energy. Consequently if we use no weights (or set $w_j = 1$) we get

$$\sum_{j\notin\Gamma} |p(x_j)|^2 \ll \|p\|^2 = \|c\|^2.$$

For such a sampling set the lower frame bound C_1 in the inequality (3.3) must be small. Generically, large gaps and the ensuing lack of information always results in bad condition numbers. This problem cannot be fixed by preconditioning.

(ii) On the other hand, we can choose a trigonometric polynomial that is mainly concentrated in the region where the sampling points are located. In this case the same local information is counted and added several times. Thus

$$\sum_{j\notin\Gamma} |p(x_j)|^2 \gg \|p\|^2 = \|c\|^2$$

and the upper constant C_2 in (3.3) will be large. Yet, as mentioned in (i) a cluster will not contribute much to the lower bound and to the uniqueness of the problem. In this case the condition number is large, because too much local information is given in certain areas of the polynomial.

Problem (ii) can be addressed by introducing properly chosen weights. The idea is to compensate for the local variation of the sampling density by using weights in inequality (3.3). Suppose that $0 \le x_1 \le x_2 \le \cdots \le x_r < 1$ is a sampling set in $[0,1]$. Then a natural choice for the weights is $w_j = (x_{j+1} - x_{j-1})/2$. Thus if many samples are clustered near a point x_j, then the weight w_j is small. If x_j is the only sampling point in a large neighborhood, then the corresponding weight is large. This choice has not only been confirmed by extensive numerical experiments [11], but also by following optimization approach.

A standard approach for the construction of preconditioners for a matrix A is the following. One attempts to find the matrix P in a given class $\mathcal{M}$ of matrices (e.g., the class of all circulant matrices or the class of all diagonal matrices) which solves

$$\min_{P \in \mathcal{M}} \|I - PA\|_F \,, \tag{3.4}$$

where $\|.\|_F$ denotes the Frobenius norm.

In our setting this translates to following optimization problem

$$\min_{W \in \mathcal{D}} \|I - (WV)^* WV\|_F \,, \tag{3.5}$$

where $\mathcal{D}$ is the class of all $r \times r$ diagonal matrices and I is the $(2M+1) \times (2M+1)$ identity matrix.

Note that we require that $w_j > 0$ whereas (3.5) could in principle yield weights that violate this condition. However since we will make use of (3.5) in our actual algorithm, we are somewhat sloppy here.

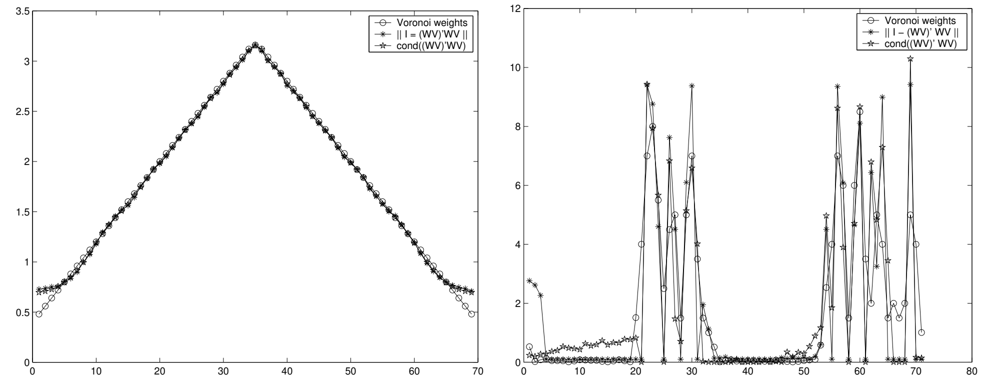
An alternative approach is to consider the solution of

$$\begin{gathered} \min\{\mathrm{cond}[(WV)^* WV]\} \\ \text{subject to } W \in \mathcal{D}. \end{gathered} \tag{3.6}$$

This optimization problem can be transformed to a general eigenvalue problem, see [4], which can be solved by convex optimization algorithms.

In the simple case of regular sampling it is easy to check that the solution of both optimization problems is given by $W = \mathrm{diag}(\{\sqrt{w_j}\})$ with $w_j = (x_{j+1} - x_{j-1})/2 = 1/r$. However in the more interesting case of nonuniform sampling neither problem (3.5) nor (3.6) does in general have an analytic solution. Thus using these approaches for the actual construction of a preconditioner would be ridiculous, since the computational costs to solve these optimization problems are considerably larger than solving the trigonometric approximation problem. Nevertheless, solving (3.5) and (3.6) numerically for a variety of different examples is useful to get insight in the type of weights obtained by these approaches.

The numerical results confirm the choice of the Voronoi-type weights defined in (3.1). Sampling points in densely sampled areas are assigned a small weight, whereas sampling points in sparse sampled regions are assigned a large weight. Two typical comparisons of the weights obtained via optimization and the Voronoi weights are illustrated in Figure 3. In the first case we consider a sampling set with high density at the endpoints and strongly decreasing density towards the center. The weights obtained by solving (3.5) and (3.6) are almost identical and are very close to the Voronoi weights, as can be seen in Figure 3(a). The difference at the endpoints is probably due to boundary effects.



(a) (b)

FIG. 3.1. *Comparison of weights obtained by different approaches.*

In the second example we consider a random sampling set with several areas with high sampling density and relatively few samples between these clusters. Again all three approaches give weights that show a similar behavior, see Figure 3(b). The condition number of the non-weighted Toeplitz matrix in this example is 33, compared to the significantly smaller condition number 3.3 when using Voronoi-type weights. Using the weights obtained via (3.5) gives $\mathrm{cond}(T_M) = 3.1$, and for the weights resulting from (3.6) we get $\mathrm{cond}(T_M) = 2.9$, which is only a slight improvement compared to the Voronoi-type weights.

Obtaining good estimates for the condition number of a Toeplitz matrix is a difficult problem. It is gratifying that by using the weights defined in (3.1) it is possible to get an upper bound for the condition number.

PROPOSITION 3.1 (Gröchenig, [18]). *Assume that the sampling set* $\{x_j\}_{j=1}^r$ *satisfies*

$$\max(x_{j+1} - x_j) := \gamma < \frac{1}{2M}$$

and set $w_j = (x_{j+1} - x_{j-1})/2$. *Then the condition number of the Toeplitz matrix* T_M *defined in* (2.24) *is bounded by*

$$\kappa(T_M) \leq \left(\frac{1+\gamma}{1-\gamma}\right)^2 . \tag{3.7}$$

4. A Levinson-Galerkin algorithm for trigonometric approximation. The method described in Algorithm 1 can be seen as a Galerkin-type approach, since we try to determine an approximation by searching for a solution in a finite-dimensional space spanned by orthogonal polynomials, and by increasing the dimension of the space we increase the resolution of our approximation by adding more and more details.

When we use Levinson's algorithm [16] to solve (2.26) for $M = 0, 1, \dots, N_0$ the total computational effort would be of $\mathcal{O}(N_0^3)$, since the solution of each system $T_M c^{(M)} = b^{(M)}$ requires $\mathcal{O}(M^2)$ operations. Using one of the fast Toeplitz algorithms [2, 5] reduces this effort to $\mathcal{O}(kM \log M)$ for each level M, where k is the number of iterations, thus leading to a total of $\mathcal{O}(kN_0^2 \log N_0)$ operations. In this section we show that the systems $T_M c^{(M)} = b^{(M)}, M = 0, 1, \dots, N_0$ can be solved in $\mathcal{O}(N_0^2)$ operations and the total effort (including the calculation of the entries of T_M and the evaluation of the stopping criterion (2.6)) for computing $p^{(N_0)}$ is $\mathcal{O}(rN_0 + N_0^2)$ operations.

Following observation is crucial for the derivation of the proposed Levinson-Galerkin algorithm.

LEMMA 4.1. *For fixed degree M and $M+1$ let $T_M, b^{(M)}$ and $T_{M+1}, b^{(M+1)}$ be the Toeplitz matrices and right hand sides as defined in* (2.24) *and* (2.27), *respectively. Then T_M and $b^{(M)}$ are embedded in T_{M+1} and $b^{(M+1)}$ in the following way:*

$$T_{M+1} = \begin{bmatrix} t_0 & \cdots & \overline{t}_{2(M+1)} \\ \vdots & \boxed{T_M} & \vdots \\ t_{2(M+1)} & \cdots & t_0 \end{bmatrix}, \quad b^{(M+1)} = \begin{bmatrix} b_{-(M+1)} \\ \boxed{b^{(M)}} \\ b_{M+1} \end{bmatrix}. \tag{4.1}$$

Proof. (4.1) follows immediately from the definition of T_M and $b^{(M)}$ and (2.8). □

Unfortunately the solutions $c^{(M)}$ and $c^{(M+1)}$ of the systems $T_M c^{(M)} = b^{(M)}$ and $T_{M+1} c^{(M+1)} = b^{(M+1)}$ are not related is such a simple manner. But we can exploit the nested structure of the family $\{T_M\}_{M=1}^{N_0}$ by solving the systems $T_M c^{(M)} = b^{(M)}$ recursively via a modified Levinson algorithm. The standard Levinson algorithm cannot be applied directly, since it only addresses Toeplitz systems, where the principal leading sub-matrix and the principal leading sub-vector of the right hand side stay unchanged during the recursion, which is not the case here. For T_{M+1} it does not matter, if we enlarge T_M by appending new entries below or above, whereas the right hand side $b^{(M)}$ cannot be rearranged in such a way, the principal leading subvector of the right hand side will be changed if we switch from $b^{(M)}$ at level M to $b^{(M+1)}$ at level $M + 1$.

To adapt Levinson's algorithm to our situation, we have to split up the change from the system $T_M c^{(M)} = b^{(M)}$ at level M to the system $T_{M+1} c^{(M+1)} = b^{(M+1)}$ at level $M+1$ into two separate steps. Instead of indexing the matrix T_M and the vectors $b^{(M)}, c^{(M)}$ by the degree M, it is therefore advantageous to index them according to their dimension. For clarity of presentation we reserve the subscript (M) for the degree of the polynomial and its coefficient vector respectively, and use the subscript

(ℓ) when we refer to the dimension of the corresponding coefficient vector in $\mathbb{C}^\ell$. Thus for even ℓ, $b^{(\ell)} = [b_{-\frac{\ell}{2}+1}, \dots, b_{\frac{\ell}{2}}]^T \in \mathbb{C}^\ell$, and for odd ℓ we set $b^{(\ell)} = [b_{-\frac{\ell-1}{2}}, \dots, b_{\frac{\ell-1}{2}}]^T$ (whence $b^{(1)} = b_0$), analogously for $c^{(\ell)}$. Further it is useful in the sequel to denote $t^{(\ell)} = [t_1, \dots, t_\ell]^T$. Then the Toeplitz matrix T_ℓ of size $\ell \times \ell$ is generated by the vector $[t_0, (t^{(\ell-1)})^T]^T$ with $t_k = \sum_{j=1}^r w_j e^{2\pi i k x_j}$ according to (2.24).

Assume we have already solved the system $T_M c^{(M)} = b^{(M)}$ at level M (with $\ell = 2M+1$) and now we want to switch to the next level $M+1$. As we have agreed, we do this in two steps. In the first step ($\ell \to \ell+1$) the Toeplitz system can be written as

$$\begin{bmatrix} T_\ell & E_\ell \overline{t^{(\ell)}} \\ (t^{(\ell)})^T E_\ell & t_0 \end{bmatrix} \begin{bmatrix} v^{(\ell)} \\ v_{\frac{\ell+1}{2}} \end{bmatrix} = \begin{bmatrix} b^{(\ell)} \\ b_{\frac{\ell+1}{2}} \end{bmatrix}, \tag{4.2}$$

where E_ℓ is the rotated identity matrix on $\mathbb{C}^\ell$, i.e.,

$$E_\ell = \begin{bmatrix} 0 & & 1 \\ & \iddots & \\ 1 & & 0 \end{bmatrix}.$$

System (4.2) can be solved recursively by the standard Levinson algorithm [21, 16]. To be more detailed, assume that we have already solved the system $T_\ell c^{(\ell)} = b^{(\ell)}$ for $\ell = 2M+1$ and assume further that the solution of the ℓ-th order *Yule-Walker system* $T_\ell y^{(\ell)} = -t^{(\ell)}$ is available. Then the solution of (4.2) can be computed recursively by

$$\begin{aligned} v_{\frac{\ell+1}{2}} &= (b_{\frac{\ell+1}{2}} - [t^{(\ell)}]^T E_\ell c^{(\ell)})/\beta_\ell \\ v^{(\ell)} &= c^{(\ell)} + v_{\frac{\ell+1}{2}} E_\ell \overline{y^{(\ell)}} \end{aligned}$$

where

$$\begin{aligned} \beta_\ell &= t_0 + [t^{(\ell)}]^T \overline{y^{(\ell)}} = (1 - \alpha_{\ell-1}\overline{\alpha_{\ell-1}})\beta_{\ell-1} \\ \alpha_\ell &= -(t_{\ell+1} + [t^{(\ell)}]^T E_\ell y^{(\ell)})/\beta_\ell \\ z^{(\ell)} &= y^{(\ell)} + \alpha_\ell E_\ell \overline{y^{(\ell)}} \\ y^{(\ell+1)} &= \begin{bmatrix} z^{(\ell)} \\ \alpha_\ell \end{bmatrix}. \end{aligned}$$

Now we can proceed to the second step ($\ell+1 \to \ell+2 = 2(M+1)+1$), where the Toeplitz system can be expressed as

$$\begin{bmatrix} t_0 & (t^{(\ell+1)})^* \\ t^{(\ell+1)} & T_{\ell+1} \end{bmatrix} \begin{bmatrix} v_{-\frac{\ell+1}{2}} \\ v^{(\ell+1)} \end{bmatrix} = \begin{bmatrix} b_{-\frac{\ell+1}{2}} \\ b^{(\ell+1)} \end{bmatrix} \tag{4.3}$$

with $c^{(\ell+2)} = [v_{-\frac{\ell+1}{2}}, (v^{(\ell+1)})^T]^T = c^{(M+1)}$. Observe that (4.3) cannot be transformed to a system of the form (4.2) by simple permutations, i.e. just by interchanging rows and columns. Since we have already solved the systems $T_{\ell+1}c^{(\ell+1)} = b^{(\ell+1)}$ and $T_{\ell+1}y^{(\ell+1)} = -t^{(\ell+1)}$ we can write

$$v^{(\ell+1)} = (T_{\ell+1})^{-1}(b^{(\ell+1)} - t^{(\ell+1)}v_{-\frac{\ell+1}{2}}) = c^{(\ell+1)} + v_{-\frac{\ell+1}{2}}y^{(\ell+1)}$$

and

$$\begin{aligned} v_{-\frac{\ell+1}{2}} =& (b_{-\frac{\ell+1}{2}} - [t^{(\ell+1)}]^* v^{(\ell+1)})/t_0 \\ =& (b_{-\frac{\ell+1}{2}} - [t^{(\ell+1)}]^* c^{(\ell+1)} - [t^{(\ell+1)}]^* v_{-\frac{\ell+1}{2}} y^{(\ell+1)})/t_0 \\ =& (b_{-\frac{\ell+1}{2}} - [t^{(\ell+1)}]^* c^{(\ell+1)})/\beta_{\ell+1}\,, \end{aligned}$$

where we have used in the last step that $T_\ell = [T_\ell]^*$ which implies that $(t^{(\ell+1)})^* y^{(\ell+1)}$ is real and therefore $t_0 + (t^{(\ell+1)})^* y^{(\ell+1)} = t_0 + (t^{(\ell+1)})^T \overline{y^{(\ell+1)}} = \beta_{\ell+1}$.

Note that at each level M we have to check if the stopping criterion (2.6) is satisfied. The evaluation of the expression

$$\sum_{j=1}^{r} |p^{(M)}(x_j) - s_j^\varepsilon|^2 w_j \tag{4.4}$$

can be considerably simplified and by avoiding the evaluation of $p^{(M)}$ at the nonuniformly spaced points x_j we can reduce the computational effort from $\mathcal{O}(Mr)$ to $\mathcal{O}(M)$ operations.

To do this we define the subspace $\mathcal{R} = \left\{\{p(x_j)\}_{j=1}^r : p \in \boldsymbol{P}_M\right\} \subseteq \mathbb{C}^r$ with the weighted inner product $\langle y, z\rangle_{\mathcal{R}} = \sum_{j=1}^r y_j \bar{z}_j w_j$ for $y, z \in \mathbb{C}^r$. The solution of the least squares problem (1.1) is the orthogonal projection of the vector $\{s_j^\varepsilon\}_{j=1}^r \in \mathbb{C}^r$ onto $\mathcal{R}$ and therefore must satisfy

$$\langle \{p^{(M)}(x_j)\} - \{s_j^\varepsilon\}, \{p^{(M)}(x_j)\}\rangle_{\mathcal{R}} = \sum_{j=1}^{r} (p^{(M)}(x_j) - s_j^\varepsilon)\overline{p^{(M)}(x_j)} w_j = 0$$

which implies

$$\langle \{p^{(M)}(x_j)\}, \{s_j^\varepsilon\}\rangle_{\mathcal{R}} = \langle \{p^{(M)}(x_j)\}, \{p^{(M)}(x_j)\}\rangle_{\mathcal{R}}\,. \tag{4.5}$$

Since

$$\begin{aligned} \sum_{j=1}^{r} |s_j^\varepsilon - p^{(M)}(x_j)|^2 w_j =& \sum_{j=1}^{r} |s_j^\varepsilon|^2 w_j - 2\,\mathrm{Re}\,\langle s^\varepsilon, \{p^{(M)}(x_j)\}\rangle_{\mathcal{R}} + \sum_{j=1}^{r} |p^{(M)}(x_j)|^2 w_j \\ =& \sum_{j=1}^{r} |s_j^\varepsilon|^2 w_j - \sum_{j=1}^{r} |p^{(M)}(x_j)|^2 w_j \end{aligned}$$

by (4.5), and because

$$\begin{aligned}\sum_{j=1}^{r}|p^{(M)}(x_j)|^2 w_j &= \sum_{j=1}^{r} w_j\Big(\sum_{m=-M}^{M} c_m^{(M)} e^{2\pi i m x_j}\Big)\Big(\sum_{n=-M}^{M}\overline{c_n^{(M)} e^{2\pi i n x_j}}\Big)\\ &= \sum_{m=-M}^{M}\sum_{n=-M}^{M} c_m^{(M)}\overline{c_n^{(M)}}\Big(\sum_{j=1}^{r} w_j e^{2\pi i(m-n)x_j}\Big)\\ &= \langle T_M c^{(M)}, c^{(M)}\rangle = \langle b^{(M)}, c^{(M)}\rangle\,, \end{aligned}\tag{4.6}$$

it follows that

$$\sum_{j=1}^{r}|s_j^{\varepsilon} - p^{(M)}(x_j)|^2 w_j = \sum_{j=1}^{r}|s_j^{\varepsilon}|^2 w_j - \langle b^{(M)}, c^{(M)}\rangle\,. \tag{4.7}$$

Since $\sum_{j=1}^{r}|s_j^{\varepsilon}|^2 w_j$ has to be computed only once at the beginning of the algorithm, the evaluation of (4.4) can be carried out in $\mathcal{O}(M)$ operations.

Summing up we have arrived at following algorithm to compute $p^{(N_0)}$.

Algorithm 2 (Levinson-Galerkin algorithm for trigonometric polynomials). *Let the sampling points $\{x_j\}_{j=1}^{r}$, sampling values $\{s_j^{\varepsilon}\}_{j=1}^{r}$, weights $w_j > 0$ and the data error estimate ε be given. Then the trigonometric polynomial $p^{(N_0)}$ determined in Algorithm 1 can be computed in $\mathcal{O}(rN_0 + N_0^2)$ operations by following algorithm.*

Initialize: $t_0 = \sum_{j=1}^{r} w_j, t_1 = \sum_{j=1}^{r} w_j e^{2\pi i x_j}, b_0 = \sum_{j=1}^{r} s_j^{\varepsilon} w_j, \sigma = \sum_{j=1}^{r}|s_j^{\varepsilon}|^2 w_j,$ $y^{(1)} = -t_1/t_0, c^{(1)} = b_0/t_0, \beta_0 = t_0, \alpha_0 = -t_1/t_0, \varepsilon_1 = (\sigma - b_0^2/t_0)/\sigma, \ell = 1.$

while $\varepsilon_\ell > \varepsilon$

$\beta_\ell = (1-\alpha_{\ell-1}\overline{\alpha_{\ell-1}})\beta_{\ell-1}$

if $\ell \equiv 1 \mod 2$

$$b_{\frac{\ell+1}{2}} = \sum_{j=1}^{r} s_j^{\varepsilon} w_j e^{\pi i(\ell+1)x_j}$$

$$v_{\frac{\ell+1}{2}} = \frac{b_{\frac{\ell+1}{2}} - \langle E_\ell\overline{c^{(\ell)}}, t^{(\ell)}\rangle}{\beta_\ell}$$

$$v^{(\ell)} = c^{(\ell)} + v_{\frac{\ell+1}{2}} E_\ell\overline{y^{(\ell)}}$$

$$c^{(\ell+1)} = \begin{bmatrix} v^{(\ell)} \\ v_{\frac{\ell+1}{2}}\end{bmatrix}$$

$$b^{(\ell+1)} = \begin{bmatrix} b^{(\ell)} \\ b_{\frac{\ell+1}{2}}\end{bmatrix}$$

elseif $\ell \equiv 0 \mod 2$

$$b_{-\frac{\ell}{2}} = \sum_{j=1}^{r} s_j^{\varepsilon} w_j e^{-\pi i \ell x_j}$$

$$v_{-\frac{\ell}{2}} = \frac{b_{-\frac{\ell}{2}} - \langle \overline{c^{(\ell)}}, t^{(\ell)} \rangle}{\beta_\ell}$$

$$v^{(\ell)} = c^{(\ell)} + v_{-\frac{\ell}{2}} y^{(\ell)}$$

$$c^{(\ell+1)} = \begin{bmatrix} v_{-\frac{\ell}{2}} \\ v^{(\ell)} \end{bmatrix}$$

$$b^{(\ell+1)} = \begin{bmatrix} b_{-\frac{\ell}{2}} \\ b^{(\ell)} \end{bmatrix}$$

$$\varepsilon_{\ell+1} = |\sigma - \langle b^{(\ell+1)}, c^{(\ell+1)} \rangle| / \sigma$$

end

$$t^{(\ell+1)} = \sum_{j=1}^{r} w_j e^{2\pi i(\ell+1)x_j}$$

$$\alpha_\ell = -\frac{t^{(\ell+1)} + \langle E_\ell \overline{y^{(\ell)}}, t^{(\ell)} \rangle}{\beta_\ell}$$

$$z^{(\ell)} = y^{(\ell)} + \alpha_\ell E_\ell \overline{y^{(\ell)}}$$

$$y^{(\ell+1)} = \begin{bmatrix} z^{(\ell)} \\ \alpha_\ell \end{bmatrix}$$

$$t^{(\ell+1)} = \begin{bmatrix} t^{(\ell)} \\ t_{\ell+1} \end{bmatrix}$$

$$\ell = \ell + 1$$

end

$$N_0 = \ell/2$$

$$p^{(N_0)}(x) = \sum_{k=-N_0}^{N_0} c_k^{(N_0)} e^{2\pi i k x}$$

REMARK: Usually one evaluates the final approximation on regularly spaced grid points, hence the last step of the algorithm can be realized by a Fast Fourier transform. The most costly steps are the computation of the entries of $t^{(\ell)}$ and $b^{(\ell)}$. According to Corollary 1 in [11] the entries of T_M and $b^{(M)}$ can also be computed via FFT by embedding the x_j into a regular grid (since the x_j can be stored only in finite precision). In this case one automatically gets all entries $t_0, \dots, t_r$ at once. However this trick is only useful if the number of points of the regular grid is of the same magnitude as the number of sampling points. Alternatively one may use the numerical attractive formulas of Rokhlin [9] or Beylkin [3] for a fast evaluation of trigonometric sums at unequally spaced nodes.

Algorithm 2 can be simplified for real-valued data, this modification is left to the reader.

Fast Vandermonde solvers require $\mathcal{O}(Mr)$ operations for the solution of $V_M c^{(M)} = s^\varepsilon$, cf. [25]. It is not clear however if these algorithms can utilize the nested structure of the sequence of matrices $\{V_M\}_M$ in order to give rise to an efficient implementation of Algorithm 1. Moreover it is an open problem if the Vandermonde solvers can be extended to multivariate trigonometric approximation. We will see in the next section that the extension of Algorithm 2 to higher dimensions is straightforward.

5. Multivariate trigonometric approximation. An advantage of the proposed approach, besides its numerical efficiency, is the fact that it can be easily extended to multivariate trigonometric approximation. In this section we briefly discuss some results for the two-dimensional case.

We define the space of 2-D trigonometric polynomials $\boldsymbol{P}_M^2$ by

$$\boldsymbol{P}_M^2 = \left\{ p : p(x,y) = \sum_{j,k=-M}^{M} c_{j,k} e^{2\pi i(jx+ky)} \right\} . \tag{5.1}$$

To reduce the notational burden, we have assumed in (5.1) that p has equal degree M in each coordinate, the extension to polynomials with different degree in each coordinate is straightforward.

For an arbitrary sampling set $\{(x_j, y_j)\}_{j=1}^{r} \in [0,1)^2$ and given degree M the system matrix according to the 2-D version of Theorem 2.5 is [28]

$$(T_M)_{k,l} = \sum_{j=1}^{r} w_j e^{2\pi i(k-l)(x_j+y_j)}\,, \quad k,l = 0,\dots 2M\,. \tag{5.2}$$

One can easily verify that T_M is a hermitian block Toeplitz matrix with $2M+1$ different Toeplitz blocks of size $(2M+1) \times 2M+1$, cf. [28]. For a given sampling set let T_M be the block Toeplitz matrix for degree M and T_{M+1} the block Toeplitz matrix for degree $M+1$. There is a similar relationship between T_M and T_{M+1} as in the 1-D case. More precisely, denote the Toeplitz blocks of T_M and T_{M+1} by $(B_M)_k, k = 0,\dots,2M$, and $(B_{M+1})_k, k = 0,\dots,2M+2$, respectively. Then one readily verifies following embedding:

$$T_{M+1} = \begin{bmatrix} (B_{M+1})_0 & \cdots & (B_{M+1})^*_{2(M+1)} \\ \vdots & \boxed{T_M} & \vdots \\ (B_{M+1})_{2(M+1)} & \cdots & (B_{M+1})_0 \end{bmatrix}, \tag{5.3}$$

$$B_{M+1} = \begin{bmatrix} t_0 & \cdots & \bar{t}_{2(M+1)} \\ \vdots & \boxed{B_M} & \vdots \\ t_{2(M+1)} & \cdots & t_0 \end{bmatrix}. \tag{5.4}$$

In [1] Levinson's algorithm has been extended to general block Toeplitz systems. With this extension and relation (5.3) at hand, we can easily generalize Algorithm 2 to 2-D (and along the same lines to multivariate) trigonometric approximation.

The analysis of the stopping criterion (2.6) in Section 2 can be applied line by line to the 2-D (actually to the n-D) setting. The only difficulty arises in the search for simple criteria for the invertibility of the block Toeplitz matrix T_M. The condition

$$(2M+1)^d \le r$$

is necessary in dimension $d > 1$, but no longer sufficient, since the fundamental theorem of algebra does not hold in dimensions larger than one. In [19] Gröchenig has derived estimates for the condition number of T_M in higher dimensions. In 2-D these estimates can be stated as follows.

Let $D_\delta(a,b) = \{(x,y) \in \mathbb{R}^2 : (x-a)^2 + (y-b)^2 < \delta^2\}$ be the disc of radius δ centered at (a,b). We say that a set $\{(x_j,y_j), j = 1,\dots,r\}$ is δ-dense in $[0,1]\times[0,1]$, if $\bigcup_{j=1}^r D_\delta(x_j,y_j) \supseteq [0,1]\times[0,1]$. In other words, the distance of a given sample (x_j,y_j) to its nearest neighbor $(x_k,y_k), k \neq j$ is at most 2δ.

Analogously to Section 3 we choose the size of the Voronoi region V_j associated with x_j as weight w_j in the computation of the block Toeplitz matrix T_M in (5.2). Suppose that the sampling set $\{(x_j,y_j), j = 1,\dots,r\} \subseteq [0,1]\times[0,1]$ is δ-dense and

$$\delta < \frac{\log 2}{4\pi M}\,. \tag{5.5}$$

Gröchenig [19] has shown that under these conditions

$$\mathrm{cond}(T_M) \le \frac{4}{(2 - e^{4\pi M\delta})^2}. \tag{5.6}$$

In particular, for arbitrary δ-dense sampling sets, the block Toeplitz matrix T_M is invertible and the 2-D version of Algorithm 2 is applicable.

5.1. Line-type nonuniform sampling in 2-D. In the following we consider a special case of trigonometric approximation in two dimensions. This case arises when a function is irregularly sampled along lines. A typical example is illustrated in Figure 5.1. Such sampling patterns are encountered for instance in geophysics and medical imaging, see also Section 6.2.

Corollary 5.1. *Let $p \in \boldsymbol{P}_M^2$ and let $\{x_j, y_{j,k}\}, j = 1,\dots,r, k = 1,\dots,r_j$ be a sampling set in $[0,1)^2$ such that*

$$A_1\|p\|^2 \le \sum_{k=1}^{r_j} |p(y_{j,k})|^2 \le B_1\|p\|^2 \qquad A_1, B_1 > 0 \tag{5.7}$$

for every $p \in \boldsymbol{P}_M$ and for all j. Further assume that $\{x_j\}_{j\in\mathbb{Z}}$ is a sampling set such that

$$A_2\|p\|^2 \le \sum_{j=1}^{r} |p(x_j)|^2 \le B_2\|p\|^2 \qquad A_2, B_2 > 0 \tag{5.8}$$

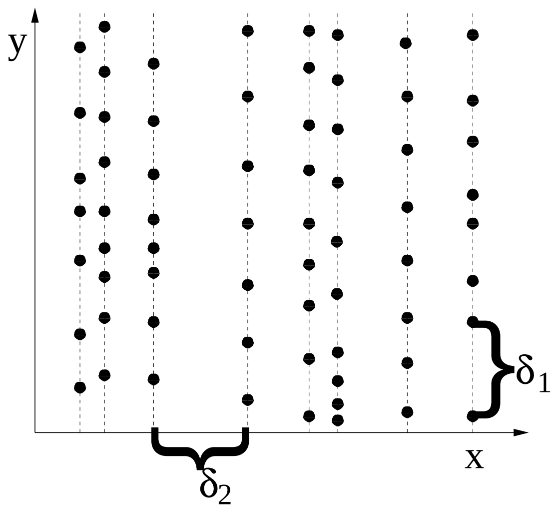


FIG. 5.1. *Line-type nonuniform sampling set*

for every $p \in \boldsymbol{P}_M$. Then

$$A_1 A_2 \|p\|^2 \le \sum_{j=1}^{r} \sum_{k=1}^{r_j} |p(x_j, y_{j,k})|^2 \le B_1 B_2 \|p\|^2 \tag{5.9}$$

for every $p \in \boldsymbol{P}_M^2$.

If $\{x_j\}$ and $\{y_{j,,k}\}$ are sampling sets with $\sup_k(x_{k+1} - x_k) = \delta_2 < \frac{1}{2M}$ and $\sup_{j,k}(y_{j,k+1} - y_{j,k}) = \delta_1 < \frac{1}{2M}$, then $A_l = (1-\delta_l)^2, B_l = (1+\delta_l)^2, l = 1,2$ and the condition number of the block Toeplitz matrix T_M is bounded by

$$\kappa(T_M) \le \frac{(1+\delta_1)^2(1+\delta_2)^2}{(1-\delta_1)^2(1-\delta_2)^2}\,. \tag{5.10}$$

Proof. Let x be fixed. Then $y \to p(x,y) \in \boldsymbol{P}_M$ and hence for all j

$$A_1 \int_0^1 |p(x_j,y)|^2\,dy \le \sum_{k=1}^{r_j} |p(x_j, y_{j,k})|^2 \le B_1 \int_0^1 |p(x_j,y)|^2\,dy \tag{5.11}$$

by assumption (5.7). It follows that

$$A_1 \sum_{j=1}^{r} \int_0^1 |p(x_j,y)|^2\,dy \le \sum_{j,k} |p(x_j, y_{j,k})|^2 \le B_1 \sum_{j=1}^{r} \int_0^1 |p(x_j,y)|^2\,dy \tag{5.12}$$

Now let y be fixed. Then $x \to p(x,y) \in \boldsymbol{P}_M$ and

$$A_2 \int_0^1 |p(x,y)|^2\,dx \le \sum_{j=1}^{r} |p(x_j,y)|^2 \le B_2 \int_0^1 |p(x,y)|^2\,dx\,. \tag{5.13}$$

Since

$$\sum_{j=1}^{r} \int_0^1 |p(x_j,y)|^2 \, dy = \int_0^1 \sum_{j=1}^{r} |p(x_j,y)|^2 \, dy \,, \tag{5.14}$$

assertion (5.9) follows by combining (5.12) and (5.13) with (5.14). The estimate of the constants A_l, B_l and of the condition number of the block Toeplitz matrix T_M follow from Theorem 2.5. ▯

The proof of Corollary 5.1 is due to Gröchenig [17]. Corollary 5.1 does not only guarantee that $p \in \boldsymbol{P}_M^2$ can be recovered from its samples $p(x_j, y_{j,k})$, it provides more. An immediate consequence is, that it can be reconstructed by an efficient algorithm relying on an successive application of Algorithm 2 and the Gohberg-Semencul representation of the inverse of a Toeplitz matrix. See Section 6.2 for more details and an application in medical imaging.

6. Curve and surface approximation by trigonometric polynomials. Trigonometric polynomials can be used to model the boundary or the surface of smooth objects. Let us consider a two-dimensional object, obtained e.g. by a planar cross-section from a 3-D object and assume that the boundary of this 2-D object is a closed curve in $\mathbb{R}^2$. We denote this curve by f and parameterize it by $f(u) = (x_u, y_u)$, where x_u and y_u are the coordinates of f at "time" u in the x- and y-direction respectively. Obviously we can interpret f as a one-dimensional continuous, complex, and periodic function, where x_u represents the real part and y_u represents the imaginary part of $f(u)$. It follows from the Theorem of Weierstrass (and from the Theorem of Stone-Weierstrass [26] for higher dimensions) that a continuous periodic function can be approximated uniformly by trigonometric polynomials. If f is smooth, we can fairly assume that trigonometric polynomials of low degree provide an approximation of sufficient precision.

Assume that we know only some arbitrary, perturbed points $s_j = (x_{u_j}, y_{u_j}) = f(u_j) + \delta_j, j = 1, \ldots, r$ of f, and we want to recover f from these points. By a slight abuse of notation we interpret s_j as complex number and write

$$s_j = x_j + iy_j \,. \tag{6.1}$$

We relate the curve parameter u to the boundary points s_j by computing the distance between two successive points s_{j-1}, s_j via

$$u_1 = 0 \tag{6.2}$$

$$u_j = u_{j-1} + d_j \tag{6.3}$$

$$d_j = \sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2} \tag{6.4}$$

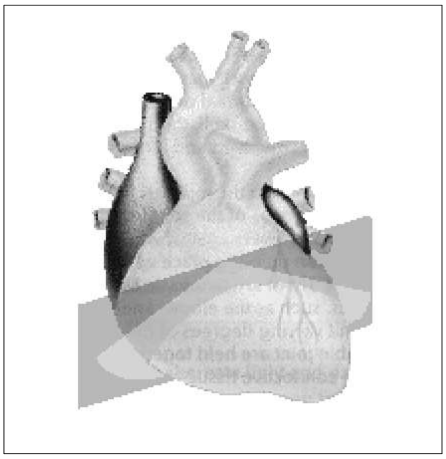
for $j = 2, \ldots, r$. Via the normalization $t_j = t_j/L$ with $L = u_r + d_N$ we force all sampling points to be in $[0, 1)$. Other choices for d_j in (6.1) can be found in [8] in conjunction with curve approximation using splines.

Having carried out the transformations (6.1)–(6.4), we can solve the problem of recovering the curve f from its perturbed points s_j by Algorithm 2.

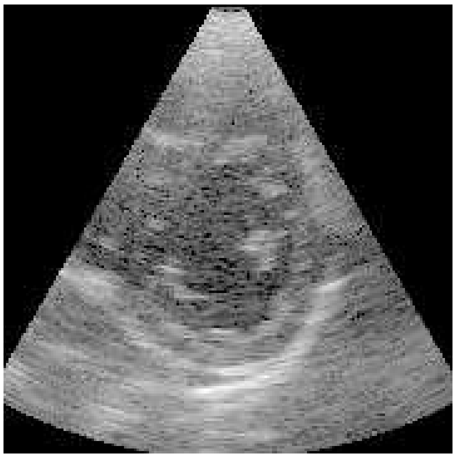
6.1. Object boundary recovery in Echocardiography. Trigonometric polynomials are certainly not suitable to model the shape of arbitrary objects. However they are often useful in cases where an underlying (stationary) physical process implies smoothness conditions of the object. Typical examples arise in medical imaging, for instance in clinical cardiac studies, where the evaluation of cardiac function using parameters of left ventricular contractibility is an important constituent of an echocardiographic examination [30]. These parameters are derived using boundary tracing of endocardial borders of the Left Ventricle (LV). The extraction of the boundary of the LV comprises two steps, once the ultrasound image of a cross section of the LV is given, see Figure 6.1(a)–(d). First an edge detection is applied to the ultrasound image to detect the boundary of the LV, cf. Figure 6.1(c). However this procedure may be hampered by the presence of interfering biological structures (such as papillar muscles), the unevenness of boundary contrast, and various kinds of noise [29]. Thus edge detection often provides only a set of nonuniformly spaced, perturbed boundary points rather than a connected boundary. Therefore a second step is required, to recover the original boundary from the detected edge points, cf. Figure 6.1(d). Since the shape of the Left Ventricle is definitely smooth, trigonometric polynomials are particularly well suited to model its boundary.

After having transformed the detected boundary points as described in (6.1)–(6.4) we can use Algorithm 2 to recover the boundary. The noise level δ depends on the technical equipment under use, it can be determined from experimental experience. Figure 6.2(a)–(b) demonstrate the importance of determining a proper degree for the approximating polynomial. The approximation displayed in Figure 6.2(a) has been computed by solving (1.1) where M has been chosen too small, we obviously have underfitted the data. The overfitted approximation obtained by solving (1.1) using a too large M is shown in Figure 6.2(b). The approximation shown in Figure 6.1(d) has been computed by Algorithm 2, it provides the optimal balance between fitting the data and smoothness of the solution.

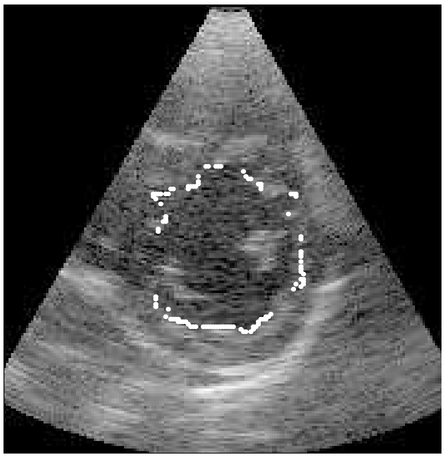
6.2. Boundary recovery from a sequence of images. In cardiac clinical studies one is more interested in the behavior of the Left Ventricle over a period of time rather than in a single "snapshot". Thus for a fixed cross section we are given a sequence of ultrasound images (usually regularly spaced in time) describing the variation of the shape of the LV with time. One cycle from diastole (the state of maximal contraction of the LV), passing systole (the state of maximal expansion) to the next diastole consists typically of about 30 image frames. Since the behavior of the LV is (at least for a short period of time) almost periodic, one can model the varying shape of a fixed cross section of the LV as distorted two-dimensional torus, which in turn can be interpreted as 2-D trigonometric polynomial. Clearly we have to use a different degree for the time coordinate τ and for the spatial coordinate u.



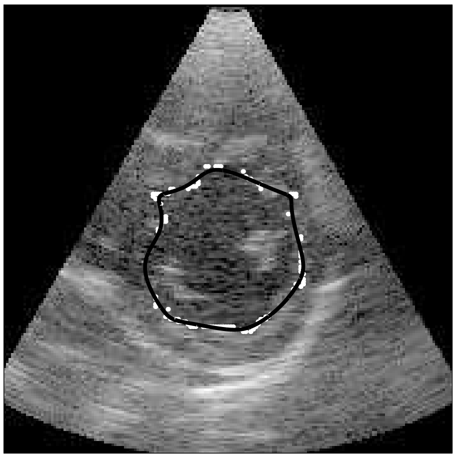
(a) 2-D echocardiography



(b) Cross section of Left Ventricle



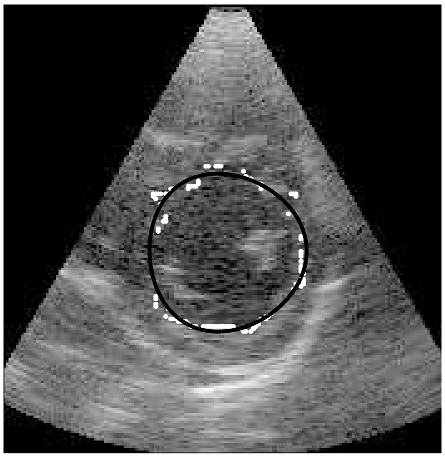
(c) Detected boundary points



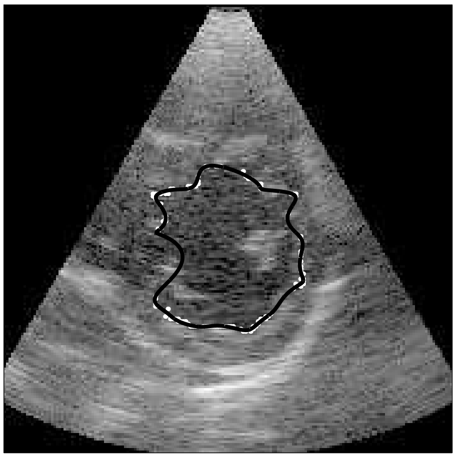
(d) Recovered boundary of LV computed by Algorithm 2

FIG. 6.1. *The recovery of the boundary of the Left Ventricle from 2-D ultrasound images is a basic step in echocardiography to extract relevant parameters of cardiac function.*

Due to interfering biological structures and other distortions it sometimes happens that some of the image frames cannot be used to extract any reliable boundary information. Thus we have to approximate these missing boundaries from the information of the other image frames. To be more precise, assume that an echocardiographic examination provides a sequence of ultrasound images I_τ taken at time points $\tau = 0, 1, \ldots, T-1$, where T is approximately the length of one diastolic cycle (the



(a) Underfitted solution



(b) Overfitted solution

FIG. 6.2. *The approximation in the left image results from using a too small polynomial degree, the approximation in the right image from a too large degree for the trigonometric approximation.*

time points could also be nonuniformly spaced). Assume that some of the images I_τ provide no useful information, so that we can only detect boundary points $\{s_{j,k}\}_{k=1}^{r_j}$ from the images I_{τ_j}, where $\{\tau_j\}_{j=1}^{r}$ is a subset of $0, 1, \dots T-1$. In order to get a complete description of the LV for the time interval $[0, T]$, we have not only to approximate the boundaries f_j from each I_j, but we also have to recover the boundaries corresponding to the missing images. In other words we look for a 2-D trigonometric polynomial $p_* \in \boldsymbol{P}_M^2$ of appropriate degree M that satisfies $p(\tau_j, u_{j,k}) \equiv (x_{j,k}, y_{j,k})$ where the parameter u is related to $s_{j,k} = x_{j,k} + iy_{j,k}$ by formulas (6.2)–(6.4). This approximation can be computed by the 2-D version of Algorithm 2, as indicated in the beginning of Section 5.

Under certain conditions we can use the 1-D version of Algorithm 2 instead of its 2-D version. As long as the assumptions of Corollary 5.1 are satisfied, we can compute $p_* \in \boldsymbol{P}_M^2$ by a successive application of Algorithm 2. We first approximate the boundaries f_j for each j separately from its samples $\{s_{j,k}\}_{k=1}^{r_j}$, which yields j different polynomials $p^{(M_j)} \in \boldsymbol{P}_{M_j}$. Having done this, the next step is to recover the missing boundaries at those time points where no information is available. We proceed by approximating successively the missing information "line by line". We choose $u = 0$, say, and approximate the missing information from the samples $p^{(M_j)}(u)$ taken at the time points $\tau_j, j = 1, \dots, r$.

Note that the Toeplitz matrices of the systems $(T_M)_u c_u^{(M)} = b_u^{(M)}$ coincide for all u, since the sampling geometry is constant along the u-coordinate (because we have recovered all samples at each τ_j). Thus we have to solve multiple Toeplitz systems with the same system matrix but different right hand side. It is well-known that this can be done efficiently by exploiting the Gohberg-Semencul representation of the inverse of the Toeplitz matrix [14]. In our context this reads as follows. We solve

$$(T_M)_u c_u^{(M)} = b_u^{(M)} \tag{6.5}$$

for one u by Algorithm 2. We can solve now all other systems efficiently by establishing $(T_M)^{-1}$ in the Gohberg-Semencul form

$$(T_M)^{-1} = \left([L^{(M)}]^* L^{(M)} - U^{(M)}[U^{(M)}]^*\right)/z_0 \tag{6.6}$$

where $L^{(M)}$ is a lower triangular Toeplitz matrix with $z = [z_0, z_1, \dots, z_{2M}]^T$ as its first column, $U^{(M)}$ is an upper triangular Toeplitz matrix with $[0, z_1, \dots, z_{2M}]^T$ as its last column, z being the first column of $(T_M)^{-1}$. The matrix vector multiplications to compute $c_u^{(M)} = (T_M)_u^{-1} b_u^{(M)}$ can now be carried out quickly using the Fast Fourier transform by embedding $L^{(M)}$ and $U^{(M)}$ into circulant matrices.

7. Miscellaneous remarks. For sampling sets with large gaps it can happen that the system $T_M c^{(M)} = b^{(M)}$ gets ill-conditioned with increasing degree M and therefore Algorithm 2 may become instable [6]. In this case one can use a different, more robust approach, which however comes at higher computational costs [27]. We solve the system $T_M c^{(M)} = b^{(M)}$ iteratively, e.g. by the conjugate gradient method until a certain stopping criterion is satisfied at iteration k, say, yielding the solution $c_k^{(M)}$. We use this solution as initial guess at the next level $M+1$ by setting $c_0^{(M+1)} = [0\ (c_k^{(M)})^T\ 0]^T$. The crucial point in this procedure is to find a stopping criterion that guarantees convergence of the iterates, see [27, 24] for more details.

The computation of the entries of the Toeplitz matrix in Section 6 involves the nodes u_j which in this particular case depend on the (perturbed) samples s_j. Therefore not only the right hand side $b^{(M)}$, but also T_M is subject to perturbations. Hence in principle one might use the concept of total least squares (see [12]) instead of a least squares approach. A detailed discussion of this modification is beyond the scope of this paper.

Acknowledgments. The major part of this work was completed during my stay at Stanford University. I want to thank Prof. David Donoho and the Department of Statistics for their hospitality.

REFERENCES

[1] H. Akaike, *Block Toeplitz matrix inversion*, SIAM J. Appl. Math., 24 (1973), pp. 234–241.

[2] G. Ammar and W. Gragg, *Superfast solution of real positive definite Toeplitz systems*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 61–76.

[3] G. Beylkin, *On the fast Fourier transform of functions with singularities*, Appl. Comp. Harm. Anal., 2 (1995), pp. 363–381.

[4] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*, SIAM, Philadelphia, PA, 1994.

[5] R. Chan and M. Ng, *Conjugate gradient methods for Toeplitz systems*, SIAM Review, 38 (1996), pp. 427–482.

[6] G. Cybenko, *The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations*, SIAM J. Sci. Statist. Comp., 1 (1980), pp. 303–3190.

[7] C. Demeure, *Fast QR factorization of Vandermonde matrices*, Linear Algebra Appl., 122–124 (1989), pp. 165–194.

[8] P. Dierckx, *Curve and surface fitting with splines*, Monographs on Numerical Analysis, Oxford University Press, 1993.

[9] A. Dutt and V. Rokhlin, *Fast Fourier transforms for nonequispaced data*, SIAM J. Sci. Comp., 14 (1993), pp. 1368–1394.

[10] H. Fassbender, *On numerical methods for discrete least-squares approximation by trigonometric polynomials*, Math. Comp., 66 (1997), pp. 719–741.

[11] H. G. Feichtinger, K. Gröchenig, and T. Strohmer, *Efficient numerical methods in non-uniform sampling theory*, Numerische Mathematik, 69 (1995), pp. 423–440.

[12] R. Fierro, G. Golub, P. Hansen, and D. O'Leary, *Regularization by truncated total least squares*, SIAM J. Sci. Comp., 18 (1997), pp. 1223–1241.

[13] W. Gautschi, *How (un)stable are Vandermonde systems?*, in Asymptotic and computational analysis (Winnipeg, MB, 1989), Dekker, New York, 1990, pp. 193–210.

[14] I. Gohberg and A. Semencul, *On the inversion of finite Toeplitz matrices and their continuous analogs*, Mat. Issled., 2 (1972), pp. 201–233.

[15] G. Golub, M. Heath, and G. Wahba, *Generalized cross-validation as a method for choosing a good ridge parameter*, Technometrics, 21 (1979), pp. 215–223.

[16] G. Golub and C. van Loan, *Matrix Computations*, Johns Hopkins, Baltimore, third ed., 1996.

[17] K. Gröchenig, *personal communication.*

[18] ———, *A discrete theory of irregular sampling*, Linear Algebra Appl., 193 (1993), pp. 129–150.

[19] ———, *Finite and infinite-dimensional models for non-uniform sampling*, in SampTA - Sampling Theory and Applications, Aveiro, Portugal, 1997, pp. 285–290.

[20] P. Hansen, *Analysis of discrete ill-posed problems by means of the L-curve*, SIAM Review, 34 (1992), pp. 561–680.

[21] N. Levinson, *The Wiener RMS (root-mean square) error criterion in filter design and prediction*, J. Math. Phys., 25 (1947), pp. 261–278.

[22] A. Newbery, *Trigonometric interpolation and curve-fitting*, Math. Comp., 24 (1970), pp. 869–876.

[23] A. Okabe, B. Boots, and K. Sugihara, *Spatial tessellations: concepts and applications of Voronoĭ diagrams*, John Wiley & Sons Ltd., Chichester, 1992. With a foreword by D. G. Kendall.

[24] M. Rauth and T. Strohmer, *Smooth approximation of potential fields from noisy scattered data*, Geophysics, 63 (1998), pp. 85–94.

[25] L. Reichel, G. Ammar, and W. Gragg, *Discrete least squares approximation by trigonometric polynomials*, Math. Comp., 57 (1991), pp. 273–289.

[26] W. Rudin, *Fourier Analysis on Groups*, Wiley Interscience, New York, 1976.

[27] O. Scherzer and T. Strohmer, *A multi–level algorithm for the solution of moment problems*, Num.Funct.Anal.Opt., 19 (1998), pp. 353–375.

[28] T. Strohmer, *Computationally attractive reconstruction of band-limited images from irregular samples*, IEEE Trans. Image Proc., 6 (1997), pp. 540–548.

[29] M. Suessner, M. Budil, T. Strohmer, M. Greher, G. Porenta, and T. Binder, *Contour detection using artifical neuronal network presegmention*, in Proc. Computers in Cardiology, Vienna, 1995.

[30] D. Wilson, E. Geiser, and J. Li, *Feature extraction in 2-dimensional short-axis echocardiographic images*, J. Math. Imag. Vision, 3 (1993), pp. 285–298.