Zentrum für Technomathematik
Fachbereich 3 – Mathematik und Informatik

On Sliding Window Schemes For Discrete Least-Squares Approximation By Trigonometric Polynomials

Heike Faßbender

Report 98–02

Berichte aus der Technomathematik

Report 98–02 **November 1998**

On Sliding Window Schemes For Discrete Least-Squares Approximation By Trigonometric Polynomials

Heike Fassbender [*]

Abstract

Fast, efficient, and reliable algorithms for up- and downdating discrete least-squares approximations of a real-valued function given at arbitrary distinct nodes in $[0, 2\pi)$ by trigonometric polynomials are presented. A combination of the up- and downdating algorithms yields a sliding window scheme. The algorithms are based on schemes for the solution of (inverse) unitary eigenproblems and require only $\mathcal{O}(mn)$ arithmetic operations as compared to $\mathcal{O}(mn^2)$ operations needed for algorithms that ignore the structure of the problem. Numerical examples are presented that show that the proposed algorithms produce consistently accurate results that are often better than those obtained by general QR decomposition methods for the least-squares problem.

Key words. trigonometric approximation, unitary Hessenberg matrix, Schur parameter, Szegö polynomial, updating, downdating, sliding window scheme

1 Introduction

A problem in signal processing is the approximation of a function known only at some measured points by a trigonometric polynomial. A number of different models for representing the measured points as a finite superposition of sine- and cosine-oscillations are possible. One choice could be to compute the trigonometric interpolating function. Then several numerical algorithms are available [14]. But in general a large number of measured points are given, such that this approach leads to a trigonometric polynomial with a lot of superposed oscillations (and a large linear system to be solved). In practical applications it is often sufficient to compute a trigonometric polynomial with only a small number of superposed oscillations. A different, often chosen approach is the (fast) Fourier transform [14]. In this case the frequencies of the sine- and cosine-oscillations have to be chosen

[*]Universität Bremen, Fachbereich 3 Mathematik und Informatik, Zentrum für Technomathematik, 28334 Bremen, Germany, e-mail : heike@math.uni-bremen.de

equidistant. The following approach gives more freedom in the choice of the frequencies and the number of superposed oscillations. Given a set of m arbitrary distinct nodes $\{\theta_k\}_{k=1}^m$ in the interval $[0, 2\pi)$, a set of m positive weights $\{\omega_k^2\}_{k=1}^m$, and a real-valued function f whose values at the nodes θ_k are explicitly known. Then the trigonometric function

$$t(\theta) = a_0 + \sum_{j=1}^{\ell}(a_j \cos j\theta + b_j \sin j\theta), \qquad a_j, b_j \in \mathbb{R}, \tag{1}$$

of order at most $\ell < m/2$ is sought that minimizes the discrete least-squares error

$$||f - t||_{\mathbb{R}} := \sqrt{\sum_{k=1}^{m} |f(\theta_k) - t(\theta_k)|^2 \omega_k^2}. \tag{2}$$

In general, m (the number of measured functional values) is much larger than $n = 2\ell + 1$ (the number of coefficients to be determined).

The problem (2) can easily be reformulated as the standard least-squares problem of minimizing

$$||D\widetilde{A}\widetilde{t} - D\widetilde{f}||_2 = \min \tag{3}$$

over all coefficient vectors $\widetilde{t}$ in the Euclidean norm, where $D = diag(\omega_1, ..., \omega_m) \in \mathbb{R}^{m\times m}$ is a diagonal matrix with the given weights on the diagonal, $\widetilde{f} = (f(\theta_1), \dots, f(\theta_m))^T$ is a vector of the measured values of the function f, and

$$\widetilde{A} = \begin{pmatrix} 1 & \sin\theta_1 & \cos\theta_1 & \cdots & \sin\ell\theta_1 & \cos\ell\theta_1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 1 & \sin\theta_m & \cos\theta_m & \cdots & \sin\ell\theta_m & \cos\ell\theta_m \end{pmatrix} \in \mathbb{R}^{m\times n}.$$

A different approach is used by Reichel, Ammar, and Gragg in [15]. They noted that the problem (2) can be reformulated as the following standard least-squares problem: Minimize

$$||DAc - Dg||_2 = \min, \tag{4}$$

where A is a transposed Vandermonde matrix

$$A = \begin{pmatrix} 1 & z_1 & \cdots & z_1^{n-1} \\ 1 & z_2 & \cdots & z_2^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & z_m & \cdots & z_m^{n-1} \end{pmatrix} \in \mathbb{C}^{m\times n}$$

with $z_k = exp(\imath\theta_k), \imath = \sqrt{-1}$. $g = (g(z_1), ..., g(z_m))^T \in \mathbb{C}^m$ is a vector of the values of a complex function $g(z)$ and $c = (c_0, ..., c_{n-1})^T \in \mathbb{C}^n$ is the solution vector. With the proper choice of g ($g = \Lambda^\ell \widetilde{f}$, $\widetilde{f}$ as above, $\Lambda = diag(z_1, \dots, z_m)$) it is easy to see that the coefficients of the trigonometric polynomial (1) that minimizes the error (2) can be read off of the least-squares solution $\widehat{c}$ of (4) (see [15])

$$\begin{aligned} a_0 &= c_\ell \\ a_j &= 2Re(c_{j+\ell}) \qquad 1 \le j \le \ell. \\ b_j &= -2Im(c_{j+\ell}) \end{aligned}$$

The usual way to solve these least-squares problem is to compute the QR decomposition of DA or $D\tilde{A}$. Ignoring the special structure of DA or $D\tilde{A}$ this requires $\mathcal{O}(mn^2)$ arithmetic operations. It can be observed however, that Szegö polynomials, that is polynomials that are orthogonal with respect to an inner product on the unit circle, arise naturally as a convenient basis for solving the above standard least-squares problems. This observation can be used to develop fast, efficient and reliable algorithms for solving the approximation problem (2).

Since DA has full column rank, there is an $m \times m$ unitary matrix Q with orthonormal columns and an $m \times n$ upper triangular matrix R with positive diagonal elements such that

$$DA = QR = (Q_1|Q_2)\begin{pmatrix} R_1 \\ 0 \end{pmatrix} = Q_1R_1,$$

where $Q_1 \in \mathbb{C}^{m\times n}$ has orthonormal columns and $R_1 \in \mathbb{C}^{n\times n}$ has positive diagonal elements. The solution of (4) is given by $\hat{c} = R_1^{-1}Q_1^HDg$. The following interpretation of the elements of Q_1 and R_1 in terms of Szegö polynomials can be given [15]: Q_1 is determined by the values of the Szegö polynomials at the nodes z_k. R_1 expresses the power basis in terms of the orthonormal Szegö polynomials. Therefore, the columns of R_1^{-1} are the coefficients of the Szegö polynomials in the power basis. There exist algorithms for determining the values of the Szegö polynomials at nodes z_k which require $\mathcal{O}(mn)$ arithmetic operations [15, 9].

Observe that

$$\begin{aligned} DA &= \begin{pmatrix} \omega_1 & \omega_1 z_1 & \omega_1 z_1^2 & \cdots & \omega_1 z_1^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ \omega_m & \omega_m z_m & \omega_m z_m^2 & \cdots & \omega_m z_m^{n-1} \end{pmatrix} \\ &= (q, \Lambda q, \Lambda^2 q, ..., \Lambda^{n-1} q) \\ &= \sigma_0(q_0, \Lambda q_0, \Lambda^2 q_0, ..., \Lambda^{n-1} q_0) \end{aligned}$$

with $q = (\omega_1, ..., \omega_m)^T, \sigma_0 = ||q||_2, q_0 := \sigma_0^{-1}q$ and $\Lambda = diag(z_1, ..., z_m)$. Thus, the matrix DA is given by the first n columns of the Krylov matrix $K(\Lambda, q_0, m) = (q_0, \Lambda q_0, ..., \Lambda^{m-1}q_0)$. We may therefore use the following consequence of the Implicit Q Theorem [11] to compute the desired QR decomposition. If there exists a unitary matrix U with $Ue_1 = q_0$ such that $U^H\Lambda U = H$ is a unitary upper Hessenberg matrix with positive subdiagonal elements, then the QR decomposition of $K(\Lambda, q_0, m)$ is given by UR with $R = K(H, e_1, m)$. The construction of such a unitary Hessenberg matrix from spectral data, here contained in Λ and q_0, is an inverse eigenproblem. Hence the best trigonometric approximation to f can be computed via solving this inverse eigenproblem. Because of the uniqueness of the here given QR decomposition of $K(\Lambda, q_0, m)$, it follows from the above given interpretation of the elements of Q_1 that the elements in the first n columns of U are the values of the Szegö polynomials at the nodes z_k. Thus solving the inverse unitary Hessenberg eigenvalue problem $U^H\Lambda U = H$ is equivalent to computing Szegö polynomials.

Unitary Hessenberg matrices have special properties which allow the development of efficient algorithms for this class of matrices. Any $n \times n$ unitary Hessenberg matrix with positive subdiagonal elements can be uniquely parameterized by n complex parameters, that is

$$H = G_1(\gamma_1)G_2(\gamma_2)\cdots G_n(\gamma_n)$$

for certain complex-valued parameters $|\gamma_k| < 1, 1 \leq k < n$, and $|\gamma_n| = 1$. Here $G_k(\gamma_k)$ denotes the $n \times n$ elementary reflector in the $(k, k+1)$ plane

$$G_k = G_k(\gamma_k) = diag(I_{k-1}, \begin{pmatrix} -\gamma_k & \sigma_k \\ \sigma_k & \overline{\gamma_k} \end{pmatrix}, I_{n-k-1})$$

with $\gamma_k \in \mathbb{C}$, $\sigma_k \in \mathbb{R}^+$, $|\gamma_k|^2 + \sigma_k^2 = 1$, and

$$G_n(\gamma_n) = diag(I_{n-1}, -\gamma_n)$$

with $\gamma_n \in \mathbb{C}$, $|\gamma_n| = 1$. The nontrivial entries γ_k are called *Schur parameters* and the σ_k are called *complementary Schur parameters.* This parameterization can be used to develop an efficient and reliable algorithm for solving the inverse unitary Hessenberg eigenvalue problem. The algorithm manipulates the n complex parameters instead of the n^2 matrix elements. An adaption of this scheme to the computation of the vector $c' = Q_1^H Dg$ can be given, which requires $\mathcal{O}(mn)$ arithmetic operations. After computing the vector c', the least-squares solution $\widehat{c} = \sigma_0^{-1} R_1^{-1} c'$ of (4) can be obtained using an algorithm closely related to the Levinson algorithm. For details see Section 2.1 or [15].

Altogether, the algorithm to construct the least-squares solution $\widehat{c}$ of (4) requires $\mathcal{O}(mn + n^2)$ arithmetic operations. The coefficients of the optimal trigonometric polynomial t of (2) can be recovered from $\widehat{c}$. This representation of t is convenient if we desire to integrate or differentiate the polynomial or if we wish to evaluate it at many equidistant points on a circle with a center at the origin. If we, on the other hand, only desire to evaluate t at a few points, then we can use the representation of t in terms of Szegö polynomials.

As $D\widetilde{A}$ in (3) is a real $m \times n$ matrix with full column rank, there exists a unique "skinny" real QR decomposition $\widetilde{Q}_1\widetilde{R}_1$ of $D\widetilde{A}$ where $\widetilde{Q}_1 \in \mathbb{R}^{m \times n}$ has orthonormal columns and $\widetilde{R}_1 \in \mathbb{R}^{n \times n}$ is upper triangular with positive diagonal entries. Analogous to the interpretation of the QR decomposition of DA in (4), an interpretation of the elements of $\widetilde{Q}_1$ and $\widetilde{R}_1$ can be given. This leads to orthogonal Laurent polynomials and the (generalized) inverse unitary eigenproblem $\widetilde{U}^H(\Lambda - \lambda I)\widetilde{U}G_e = G_o - \lambda G_e$, where G_o and G_e are unitary block diagonal matrices with 1×1 or 2×2-blocks on the diagonal. The nonzero entries of G_o and G_e are just the Schur parameters and the complementary Schur parameters:

$$G_o = G_1(\gamma_1)G_3(\gamma_3)\cdots G_{2[(n+1)/2]-1}(\gamma_{2[(n+1)/2]-1}) =$$

$$= \begin{pmatrix} -\gamma_1 & \sigma_1 & & & \\ \sigma_1 & \overline{\gamma_1} & & & \\ & & -\gamma_3 & \sigma_3 & \\ & & \sigma_3 & \overline{\gamma_3} & \\ & & & & \ddots \end{pmatrix}$$

is the product of the odd numbered elementary reflectors and

$$G_e^H = G_2(\gamma_2)G_4(\gamma_4)\cdots G_{2[n/2]}(\gamma_{2[n/2]}) = \begin{pmatrix} 1 & & & \\ & -\gamma_2 & \sigma_2 & \\ & \sigma_2 & \overline{\gamma_2} & \\ & & & \ddots \end{pmatrix}$$

is the product of the even numbered elementary reflectors. The (generalized) inverse eigenproblem $\widetilde{U}^H(\Lambda - \lambda I)\widetilde{U}G_e = G_o - \lambda G_e$, where a *Schur parameter pencil* is constructed from spectral data, is equivalent to the inverse unitary Hessenberg eigenproblem $U^H \Lambda U = H = G_1(\gamma_1)\ldots G_n(\gamma_n)$ [1].

Observe that

$$\begin{aligned} D\widetilde{A} &= \frac{1}{2}(q, \Lambda q, \Lambda^H q, \Lambda^2 q, (\Lambda^H)^2 q, \ldots, \Lambda^\ell q, (\Lambda^H)^\ell q) \begin{pmatrix} 2 & & & & & \\ & -i & 1 & & & \\ & i & 1 & & & \\ & & & \ddots & & \\ & & & & -i & 1 \\ & & & & i & 1 \end{pmatrix} \\ &= \frac{1}{2}\sigma_0(q_0, \Lambda q_0, \Lambda^H q_0, \Lambda^2 q_0, (\Lambda^H)^2 q_0, \ldots, \Lambda^\ell q_0, (\Lambda^H)^\ell q_0)F \\ &= \frac{1}{2}\sigma_0 \kappa(\Lambda, q_0, \ell)F \end{aligned}$$

with q, σ_0, q_0 and Λ as before. A QR-like decomposition of $D\widetilde{A}$ can be obtained using the following result [8, 10] : If there exists a unitary matrix V such that $V(\Lambda - \lambda I)V^H G_e = G_o - \lambda G_e, V^H e_1 = q_0$, then the QR decomposition of $\kappa(\Lambda, q_0, \ell)$ is given by VR with $R = \kappa(G_o G_e^H, e_1, \ell)$. Hence $D\widetilde{A} = \frac{\sigma_0}{2} VRF$ and the optimal solution of (3) is given by $\widetilde{t} = 2\sigma_0^{-1}F^{-1}R^{-1}V^H D\widetilde{f}$. The construction of such a Schur parameter pencil from spectral data is a (generalized) inverse eigenproblem. Thus the best trigonometric approximation to f can be computed via solving this inverse eigenproblem. As explained in [9], the elements of V are the values of orthogonal Laurent polynomials at the nodes θ_k. Thus solving the inverse unitary eigenproblem $V(\Lambda - \lambda I)W = G_o - \lambda G_e$ is equivalent to computing orthogonal Laurent polynomials.

The special structure of the inverse eigenproblem $V(\Lambda - \lambda I)V^H G_e = G_o - \lambda G_e$ can be used to develop an efficient and reliable algorithm for computing V, γ_j and $\sigma_j, j = 1, \ldots, n$. The advantage of this approach over (4) is that here an algorithm can be given which solves the real-valued problem (2) using only real arithmetic. For details see Section 2.2 or [10].

The two algorithms sketched above are updating procedures in the sense that the least-squares fit is obtained by incorporating the nodes of the inner product one at a time. In certain applications it may be desirable to replace certain node-weight pairs (θ_k, ω_k^2). This can be carried out by successively removing a node-weight pair from the current approximation, and then adding a new node-weight pair. Downdating Szegö polynomials/orthogonal Laurent polynomials and a given least-squares fit when one node is deleted from the inner product can easily be implemented solving unitary eigenproblems.

The updating and downdating procedures, based on solving inverse unitary eigenproblem and unitary eigenproblems, can be combined to yield a sliding window scheme, in which one node is replaced by another.

Updating and downdating of polynomial approximations when all nodes z_k are real has received a lot of attention in the literature, see [16] and the references therein. A collection of algorithms for updating and downdating based on orthogonal polynomials is presented in [7]. Downdating Szegö polynomials is considered in [3], while the updating process is the topic of [15, 2, 9].

In the following we will discuss the up- and downdating process for solving the approximation problem (2) via the Vandermonde approach (4) and the sin-cos-approach (3). The connection of the presented algorithms to algorithms for computing Szegö polynomials/orthogonal Laurent polynomials will not be discussed any further, see [9] for a detailed discussion in the updating context and [3] for a discussion of downdating Szegö polynomials.

Updating schemes are discussed in Section 2, while downdating schemes are presented in Section 3. In Section 4 we compare the different algorithms based on solving (inverse) unitary eigenproblems with each other and with a general QR decomposition. We will see that the proposed algorithms produce consistently accurate results that are often better than those obtained by general QR decomposition methods for the least-squares problem.

2 Updating

Let the trigonometric polynomial $t(\theta) = a_0 + \sum_{j=1}^{\ell}(a_j \cos j\theta + b_j \sin j\theta)$ be the optimal solution of the approximation problem (2) corresponding to the data $Y_m = \{\theta_k, \omega_k^2\}_{k=1}^m$. Suppose Y_{m+1} is obtained from Y_m by augmenting a new node-weight pair $(\theta_{m+1}, \omega_{m+1}^2)$. Solving the approximation problem for Y_{m+1} assuming the knowledge of its solution for Y_m is called *updating* the least-squares fit.

As explained in the introduction, the least-squares fit can be represented by the coefficients of its expansion in terms of Szegö polynomials (orthogonal Laurent polynomials). Using matrix notation the approximation problem can be solved via an inverse unitary eigenproblem. Hence, the problem of updating the optimal trigonometric approximation t of (2) can be expressed as follows:

Given

$\sigma_0 > 0$	
H_m	unitary upper Hessenberg matrix of size $m \times m$
d_m	a vector of length m
(λ, ν^2)	a node-weight pair

(σ_0, H_m, d_m representing the solution of (2) for some data set Y_m) find $\sigma_0 > 0$, a vector d_{m+1} and a unitary upper Hessenberg matrix H_{m+1} such that

1. the eigenvalues of H_{m+1} are $e^{\imath\lambda}$ and those of H_m
2. the vector d_{m+1} contains the first components of the eigenvectors of H_{m+1}, that is if the entries of d_m are $\delta_1/\sigma_0, \ldots, \delta_m/\sigma_0$ and $\sigma_0 = (\sum_{k=1}^m \delta_k^2)^{\frac{1}{2}}$, then the new σ_0 will be $\sigma_0 = (\sigma_0^2 + \nu^2)^{\frac{1}{2}}$ and the entries of d_{m+1} are δ_k/σ_0 for $k = 1, \ldots, m$ and ν/σ_0.

A similar definition of the updating process in terms of Schur parameter pencil can be given.

The approximation problem (2) can entirely be solved by updating, starting from the trivial solution for $m = 1$.

2.1 The Vandermonde Approach

As we have seen in the introduction, the computation of the best least-squares fit f of (2) via the approach (4) involves solving an inverse unitary Hessenberg eigenvalue problem $\Lambda U = UH$. Here we will outline an algorithm for the construction of the matrices H and U from spectral data. This algorithm, called IUHQR algorithm, can be regarded as an inverse QR algorithm for unitary Hessenberg matrices and was first described by Ammar, Gragg and Reichel in [2].

The problem of constructing a unitary upper Hessenberg matrix from spectral data can be formulated more generally as follows. Given m distinct unimodular complex numbers $\{\lambda_k\}_{k=1}^m$ and associated positive weights $\{\nu_k^2\}_{k=1}^m$, we wish to construct a unitary Hessenberg matrix H with the λ_k and ν_k equal to the eigenvalues and first components of the corresponding eigenvectors of H, respectively. The following proposition is a consequence of the "Implicit Q Theorem" ([11, §7.4.5]) and gives existence and uniqueness of the unitary transformation to an unreduced unitary upper Hessenberg matrix H.

Proposition 2.1 *Given m distinct complex numbers $\{\lambda_k\}_{k=1}^m$ on the unit circle and associated positive weights $\{\nu_k^2\}_{k=1}^m$, there is a unique $m \times m$ unitary Hessenberg matrix H with positive subdiagonal elements and a unique unitary matrix U satisfying*

$$\begin{aligned} U^H e_1 &= \sigma_0^{-1}(\nu_1, ..., \nu_m)^T \\ U\Lambda U^H &= H \\ \Lambda &= diag(\lambda_1, ..., \lambda_m), \end{aligned}$$

where $\sigma_0 = (\sum_{k=1}^m \nu_k^2)^{\frac{1}{2}}$.

The required upper Hessenberg matrix H is obtained by performing a sequence of elementary unitary similarity transformations whose composition results in the $m \times m$ unitary matrix U such that

$$\begin{pmatrix} 1 & \\ & U \end{pmatrix} \begin{pmatrix} \delta & u^H \\ u & \Lambda \end{pmatrix} \begin{pmatrix} 1 & \\ & U^H \end{pmatrix} = \begin{pmatrix} \delta & e_1^H \\ e_1 & U\Lambda U^H \end{pmatrix}$$

is an upper Hessenberg matrix with positive subdiagonal elements. The number δ is arbitrary and remains unchanged during the execution of the algorithm. Then $H = U\Lambda U^H$ has the desired eigenvalues and associated eigenvectors.

Ammar, Gragg and Reichel describe in [2] an inverse unitary QR algorithm to solve this problem (analogous to the method by Gragg and Harrod [13] to solve the inverse eigenvalue problem for symmetric tridiagonal matrices, see also [5]). The idea is to build up the Hessenberg matrix successively by adding node-weight pairs (λ_k, ν_k^2) one at a time.

Suppose we have constructed an upper Hessenberg matrix H_p corresponding to the p node-weight pairs $\{(\lambda_k, \nu_k^2)\}_{k=m-p+1}^m$. The eigenvalues of H_p are $\{\lambda_k\}_{k=m-p+1}^m$ and the first components of its normalized eigenvectors are $\{(\nu_k/\sigma_p)\}_{k=m-p+1}^m$ where $\sigma_p = (\sum_{k=m-p+1}^m \nu_k^2)^{\frac{1}{2}}$. That is, an $p \times p$ unitary matrix Q_p is constructed such that

$$\begin{pmatrix} I_{m-p} & \\ & Q_p \end{pmatrix} \begin{pmatrix} \delta & u^H \\ u & \Lambda \end{pmatrix} \begin{pmatrix} I_{m-p} & \\ & Q_p^H \end{pmatrix} =$$

$$= \left(\begin{array}{c|ccc|c|cccc} \delta & \overline{\nu}_1 & \cdots & \overline{\nu}_{m-p-1} & \overline{\nu}_{m-p} & \sigma_p & 0 & \cdots & 0 \\ \hline \nu_1 & \lambda_1 & & & & & & & \\ \vdots & & \ddots & & & & & & \\ \nu_{m-p-1} & & & \lambda_{m-p-1} & & & & & \\ \hline \nu_{m-p} & & & & \lambda_{m-p} & & & & \\ \hline \sigma_p & & & & & & & & \\ 0 & & & & & & & & \\ \vdots & & & & & & H_p & & \\ 0 & & & & & & & & \end{array}\right) = \widetilde{H}_p$$

and $Q_p^H e_1 = \sigma_p^{-1}(\nu_{m-p+1}, ..., \nu_m)^T$.

Now we perform a sequence of unitary similarity transformations in order to add the next node-weight pair $(\lambda_{m-p}, \nu_{m-p})$ and to construct the corresponding $(p+1) \times (p+1)$ Hessenberg matrix. Let $\sigma_{p+1} = (\sigma_p^2 + \nu_{m-p}^2)^{\frac{1}{2}}$ and $\alpha_{p+1} = -\nu_{m-p}/\sigma_{p+1}$. Then for $G_{m-p}^H(\alpha_{p+1})\widetilde{H}_p =: \widehat{H}_{p+1}$,

$$\widehat{H}_{p+1} = \left(\begin{array}{c|ccc|ccccc} \delta & \overline{\nu}_1 & \cdots & \overline{\nu}_{m-p-1} & \overline{\nu}_{m-p} & \sigma_p & 0 & \cdots & 0 \\ \hline \nu_1 & \lambda_1 & & & & & & & \\ \vdots & & \ddots & & & & & & \\ \nu_{m-p-1} & & & \lambda_{m-p-1} & & & & & \\ \hline \sigma_{p+1} & & & & x & x & x & \cdots & x \\ 0 & & & & x & x & x & \cdots & x \\ 0 & & & & & x & x & \cdots & x \\ \vdots & & & & & & \ddots & \ddots & \vdots \\ 0 & & & & & & & x & x \end{array}\right),$$

the trailing principal $(p+1) \times (p+1)$ submatrix is a unitary upper Hessenberg matrix. On the completion of the similarity transformation we obtain

$$\widehat{H}_{p+1} G_{m-p}(\alpha_{p+1}) = \left(\begin{array}{c|ccc|ccccc} \delta & \overline{\nu}_1 & \cdots & \overline{\nu}_{m-p-1} & \sigma_{p+1} & 0 & 0 & \cdots & 0 \\ \hline \nu_1 & \lambda_1 & & & & & & & \\ \vdots & & \ddots & & & & & & \\ \nu_{m-p-1} & & & \lambda_{m-p-1} & & & & & \\ \hline \sigma_{p+1} & & & & x & x & x & \cdots & x \\ 0 & & & & x & x & x & \cdots & x \\ 0 & & & & \otimes & x & x & \cdots & x \\ \vdots & & & & & & \ddots & \ddots & \vdots \\ 0 & & & & & & & x & x \end{array}\right).$$

The $\otimes$ element is chased down along the subdiagonal by unitary similarity transformations with a sequence of matrices $G_{m-p+1}, ..., G_m$ until the $(p+1) \times (p+1)$ trailing principal submatrix H'_{p+1} has upper Hessenberg form. Proposition 2.1 gives that H'_{p+1} is unitarily similar to a unitary upper Hessenberg matrix H_{p+1} with positive subdiagonal elements. The latter matrix is the desired Hessenberg matrix. Note that in $G^H_{m-p+j} ... G^H_{m-p} \widetilde{H}_l G_{m-p} ... G_{m-p+j-1}$ the $(p+1) \times (p+1)$ trailing principal submatrix is a unitary upper Hessenberg matrix with positive subdiagonal elements. Hence we can carry out the similarity transformation by manipulating the Schur parameters. Ammar, Gragg and Reichel present in [2] such an algorithm, the inverse unitary QR algorithm.

From this we get the following algorithm to compute a $m \times m$ unitary Hessenberg matrix $H = H(\gamma_1, ..., \gamma_m)$ from given eigenvalues and given first components of the eigenvectors.

IUHQR algorithm to solve the inverse unitary Hessenberg eigenvalue problem

```
input : $\lambda_1, ..., \lambda_m, \nu_1, ..., \nu_m$ (eigenvalues and first components of the normalized eigenvectors)
output : $\gamma_1, ..., \gamma_m, \sigma_1, ..., \sigma_m$ (Schur parameters and complementary Schur parameters)
        $\sigma_0 = \nu_1$
        $\gamma_1 = -\lambda_1$
        $\sigma_1 = 0.0$
        for i = 2, ..., m
            $\sigma_{old} = \sigma_0$
            $\sigma_0 = \sqrt{\sigma_{old}^2 + \nu_i^2}$
            $\beta = \sigma_{old}/\sigma_0$
            $\alpha = -\nu_i/\sigma_0$
            $\gamma_i = -\lambda_i \gamma_{i-1}$
            for k = 1, ..., i-1
                $\sigma_{old} = \sigma_k$
                $\gamma_{old} = \gamma_k$
                $\sigma_k = \beta\sqrt{\sigma_{old}^2 + |\alpha + \gamma_k \overline{\alpha} \lambda_i^{k-2}|^2}$
                $\gamma_k = \beta^2 \gamma_k - \overline{\lambda}_i^{k-2} \alpha^2$
                $\alpha = \beta \lambda_i (\alpha + \gamma_{old} \lambda_i^{k-2} \overline{\alpha})/\sigma_k$
                $\beta = \beta \sigma_{old}/\sigma_k$
            end for k
        end for i
```

An efficient implementation of this algorithm requires $\approx 12m^2 - 5m$ arithmetic operations. A mathematically equivalent algorithm which manipulates matrix elements (instead of Schur parameters) requires $\mathcal{O}(m^3)$ arithmetic operations.

The IUHQR algorithm is an updating procedure because it incorporates node-weight pairs one at a time. After the jth step of the algorithm, the $j \times j$ unitary Hessenberg matrix corresponding to the first j nodes and weights has been obtained. The order in which the node-weight pairs are incorporated is optional.

As explained in the introduction, the IUHQR algorithm can be used to construct the trigonometric approximation polynomial t via the Vandermonde approach (4). The IUHQR algorithm can easily be modified to compute the unitary matrix Q which transforms Λ to the upper Hessenberg matrix $H = G_1(\gamma_1) \ldots G_m(\gamma_m)$. In order to solve the trigonometric approximation problem we only need the matrix Q_1, the first n columns of Q. That is, the IUHQR algorithm can be curtailed so that only $\mathcal{O}(mn)$ arithmetic operations are required for the computation of the parameters $\{\gamma_j\}_{j=1}^n, \{\sigma_j\}_{j=0}^n$ and the vector $c' = Q_1^H Dg$. There is no need to form Q_1 explicitly, the algorithm can be modified such that c' is updated successively. See [15] for details.

In order to compute the least-squares solution $\widehat{c}$ from (4), the vector $\widehat{c} = R_1^{-1} c'$ has to be build, where $R_1 = \sigma_0 K(G_1(\gamma_1) \cdots G_n(\gamma_n), e_1, n)$. As $R_1^H R_1 = T$ is a Toeplitz matrix, R_1^{-1} can be computed via the Levinson algorithm. Since the Schur parameters γ_j and the complementary Schur parameters σ_j are already known, the Levinson algorithm can be simplified. Let $S = R_1^{-1} = (s_1, \ldots, s_n)$. Then the s_j are given by the following recursion

$$\begin{aligned} s_1 &= \sigma_0^{-1} e_1 \\ s_{j+1} &= \sigma_j^{-1}(J s_j + \gamma_j \widetilde{I}_j \overline{s_j}) \qquad j = 1, 2, \ldots, n \end{aligned}$$

where $J = (e_2, e_3, \ldots, e_n, 0)$ and $\widetilde{I}_k = (e_k, \ldots, e_1, e_{k+1}, \ldots, e_n)$. Reichel, Ammar and Gragg present in [15] an $\mathcal{O}(n^2)$ algorithm for computing $R_1^{-1} b$ for any vector $b \in \mathbb{C}^n$ using the above recursion. This is also an updating process, as the Schur parameters are incorporated successively.

Altogether, the algorithm to construct the least-squares solution $\widehat{c}$ of (4) requires $\mathcal{O}(mn + n^2)$ arithmetic operations. The coefficients of the optimal trigonometric polynomial t of (2) can be recovered from $\widehat{c}$.

2.2 The sin-cos-approach

As explained in the introduction, the discrete trigonometric approximation problem (2) can be reformulated in terms of (3). This lead to the problem of solving a (generalized) inverse eigenproblem: Given m distinct unimodular complex numbers $\{\lambda_k\}_{k=1}^m$ and associated positive weights $\{\nu_k^2\}_{k=1}^m$, we wish to construct a Schur parameter pencil $G_o - \lambda G_e$ with the λ_k and ν_k equal to the eigenvalues and first components of the corresponding eigenvectors of $G_o G_e^H$, respectively. The following proposition is a consequence of the Proposition 2.1 and the fact that H is similar to $G_o G_e^H$. It gives existence and uniqueness of the transformation to Schur parameter pencil form.

Proposition 2.2 *Given m distinct complex numbers $\{\lambda_k\}_{k=1}^m$ on the unit circle and associated positive weights $\{\nu_k^2\}_{k=1}^m$, there is a unique unreduced $m \times m$ Schur parameter pencil $G_o - \lambda G_e$ (with positive complementary Schur parameters) and unique unitary matrices Q and P such that*

$$\begin{aligned} Q^H e_1 &= \sigma_0^{-1}(\nu_1, ..., \nu_m)^T \\ Q(\Lambda - \lambda I)P &= G_o - \lambda G_e \\ \Lambda &= diag(\lambda_1, ..., \lambda_m), \end{aligned}$$

where $\sigma_0 = (\sum_{k=1}^m \nu_k^2)^{\frac{1}{2}}$.

In the following a matrix pair (Q, P) of unitary matrices Q and P is constructed such that $Q^H e_1 = u = \sigma_0^{-1}(\nu_1, ..., \nu_m)^T$ and $Q(\Lambda - \lambda I)P = G_o - \lambda G_e$, that is

$$\begin{pmatrix} 1 & \\ & Q \end{pmatrix} \left[\begin{pmatrix} \delta & 0 \\ u & \Lambda \end{pmatrix} - \lambda \begin{pmatrix} \delta & \\ & I \end{pmatrix} \right] \begin{pmatrix} 1 & \\ & P \end{pmatrix} =$$

$$= \begin{pmatrix} \delta & 0 \\ e_1 & Q\Lambda P \end{pmatrix} - \lambda \begin{pmatrix} \delta & \\ & QP \end{pmatrix}.$$

$(Q\Lambda P, QP)$ is a pair of unitary matrices with the eigenvalues $\{\lambda_k\}_{k=1}^m$ and the first components of the eigenvectors of $(Q\Lambda P)(QP)^H = Q\Lambda Q^H$ are the ν_k/σ_0. Q and P can be chosen such that $G_o = Q\Lambda P$ and $G_e = QP$ have the desired form G_o, G_e respectively.

Analogous to the idea of the IUHQR-algorithm we build up a Schur parameter pencil by adding a node-weight-pair (λ, ν^2) one at a time. Suppose we have constructed a unitary Schur parameter pencil $G_o^p - \lambda G_e^p$ with the desired properties from the node-weight-pairs $\{(\lambda_k, \nu_k^2)\}_{k=m-p+1}^m$. Now a Schur parameter pencil $G_o^{p+1} - \lambda G_e^{p+1}$ has to be computed by adding the pair $(\lambda_{m-p}, \nu_{m-p}^2)$. Thus we want to reduce the unitary matrix pencil

$$\begin{pmatrix} \lambda_{m-p} & \\ & G_o^p \end{pmatrix} - \lambda \begin{pmatrix} 1 & \\ & G_e^p \end{pmatrix} \tag{5}$$

to a Schur parameter pencil such that the eigenvectors of $G_o^{p+1}(G_e^{p+1})^H$ have as the first components (ν_k/σ_{p+1}) with $\sigma_{p+1} = (\sum_{k=m-p}^m \nu^2)^{\frac{1}{2}}$.

Bunse-Gerstner and Elsner present in [4] an algorithm to reduce a unitary matrix pencil to a Schur parameter pencil. It is shown that the algorithm is backward stable. In the following this algorithm is briefly sketched for the special structure of the unitary matrix pencil (5) given here. Let

$$\hat{G}_j(\alpha) = diag(I_{j-1}, \begin{pmatrix} -\alpha & \overline{\beta} \\ \beta & \overline{\alpha} \end{pmatrix}, I_{n-j})$$

with $|\alpha|^2 + |\beta|^2 = 1$.

Given

$$\left(\begin{array}{cc|ccc} * & 0 & 0 & \cdots & 0 \\ \nu_{m-p} & \lambda_{m-p} & 0 & \cdots & 0 \\ \hline \sigma_p & 0 & & & \\ 0 & 0 & & & \\ \vdots & \vdots & & G_o^p & \\ 0 & 0 & & & \end{array}\right) - \lambda \left(\begin{array}{cc|ccc} * & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \hline 0 & 0 & & & \\ \vdots & \vdots & & G_e^p & \\ 0 & 0 & & & \end{array}\right) = \tilde{G}_o^p - \lambda \tilde{G}_e^p,$$

we perform a sequence of unitary equivalence transformations in order to add the next node-weight pair $(\lambda_{m-p}, \nu_{m-p}^2)$ and to construct the corresponding $(p+1) \times (p+1)$ Schur parameter pencil. Let $\sigma_{p+1} = (\sigma_p^2 + \nu_{m-p}^2)^{\frac{1}{2}}$ and $\alpha_0 = -\nu_{m-p}/\sigma_{p+1}$. Then

$$G_2(\alpha_0)(\tilde{G}_o^p - \lambda \tilde{G}_e^p)G_2^H(\alpha_0) =: G_{o1}^{p+1} - \lambda G_{e1}^{p+1} =$$

$$\begin{pmatrix} * & & & & & & \\ \sigma_{p+1} & x & x & x & & & \\ & x & x & x & & & \\ & x & x & x & & & \\ & & & & -\gamma_3 & \sigma_3 & \\ & & & & \sigma_3 & \overline{\gamma}_3 & \\ & & & & & & \ddots \end{pmatrix} - \lambda \begin{pmatrix} * & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & -\overline{\gamma}_2 & \sigma_2 & & & \\ & & & \sigma_2 & \gamma_2 & & & \\ & & & & & -\overline{\gamma}_4 & \sigma_4 & \\ & & & & & \sigma_4 & \gamma_4 & \\ & & & & & & & \ddots \end{pmatrix}.$$

Here x denotes any changed matrix element. Now α_2 is determined such that in $G_3(\alpha_2)G_{o1}^{p+1}$ the (4,2)-element is eliminated. This gives

$$G_3(\alpha_2)(G_{o1}^{p+1} - \lambda G_{e1}^{p+1}) =$$

$$\begin{pmatrix} * & & & & & & \\ \sigma_{p+1} & -\tilde{\gamma}_1 & x & x & & & \\ & \tilde{\sigma}_1 & x & x & & & \\ & & x & x & & & \\ & & & & -\gamma_3 & \sigma_3 & \\ & & & & \sigma_3 & \overline{\gamma}_3 & \\ & & & & & & \ddots \end{pmatrix} - \lambda \begin{pmatrix} * & & & & & & & \\ & 1 & & & & & & \\ & & x & x & x & & & \\ & & x & x & x & & & \\ & & & \sigma_2 & \gamma_2 & & & \\ & & & & & -\overline{\gamma}_4 & \sigma_4 & \\ & & & & & \sigma_4 & \gamma_4 & \\ & & & & & & & \ddots \end{pmatrix}.$$

By multiplication from the right with a properly chosen $\widehat{G}_3(\zeta_2)$ the (2,4) or (3,4)-element can be eliminated. In either case we obtain

$$G_3(\alpha_2)(G_{o1}^{p+1} - \lambda G_{e1}^{p+1})\widehat{G}_3(\zeta_2) =$$

$$\begin{pmatrix} * & & & & & & \\ \sigma_{p+1} & -\tilde{\gamma}_1 & a & & & & \\ & \tilde{\sigma}_1 & b & & & & \\ & & & x & & & \\ & & & & -\gamma_3 & \sigma_3 & \\ & & & & \sigma_3 & \overline{\gamma}_3 & \\ & & & & & & \ddots \end{pmatrix} - \lambda \begin{pmatrix} * & & & & & & & \\ & 1 & & & & & & \\ & & x & x & x & & & \\ & & x & x & x & & & \\ & & x & x & x & & & \\ & & & & & -\overline{\gamma}_4 & \sigma_4 & \\ & & & & & \sigma_4 & \gamma_4 & \\ & & & & & & & \ddots \end{pmatrix}.$$

The additional zeros which are not enforced by the elimination with $\widehat{G}_3(\zeta_2)$ must occur because $G_3(\alpha_2)G_{o1}^{p+1}\widehat{G}_3(\zeta_2)$ is still unitary. Normalization with a matrix of the form $diag(1, 1, \frac{\overline{a}}{|a|}, 1, ..., 1)$ gives

$$G_{o2}^{p+1} - \lambda G_{e2}^{p+1} = G_3(\alpha_2)(G_{o1}^{p+1} - \lambda G_{e1}^{p+1})\widehat{G}_3(\zeta_2)diag(1, 1, \frac{\overline{a}}{|a|}, 1, ..., 1) =$$

$$\begin{pmatrix} * & & & & & & \\ \sigma_{p+1} & -\tilde{\gamma}_1 & \overline{\tilde{\sigma}_1} & & & & \\ & \tilde{\sigma}_1 & \overline{\tilde{\gamma}_1} & & & & \\ & & & x & & & \\ & & & & -\gamma_3 & \sigma_3 & \\ & & & & \sigma_3 & \overline{\gamma}_3 & \\ & & & & & & \ddots \end{pmatrix} - \lambda \begin{pmatrix} * & & & & & & & \\ & 1 & & & & & & \\ & & x & x & x & & & \\ & & x & x & x & & & \\ & & x & x & x & & & \\ & & & & & -\overline{\gamma}_4 & \sigma_4 & \\ & & & & & \sigma_4 & \gamma_4 & \\ & & & & & & & \ddots \end{pmatrix}.$$

The 3×3 block in G_{e2}^{p+1} can be transformed to

$$\begin{pmatrix} -\overline{\tilde{\gamma}_2} & \tilde{\sigma}_2 & \\ \tilde{\sigma}_2 & \tilde{\gamma}_2 & \\ & & x \end{pmatrix}$$

by an equivalence transformation with properly chosen matrices $diag(1,1,1,x,1,...,1)$, $\widehat{G}_4(\alpha_3)$ and $\widehat{G}_4(\zeta_3)$.

Moreover, in $\widehat{G}_3(\alpha_4)G_{o2}^{p+1}\widehat{G}_4(\zeta_3)diag(1,1,1,x,1,...,1)$ a new 3×3 block appears, starting at the element (4,4). In an obvious way one proceeds with the reduction to Schur parameter pencil form; at each step an additional 3×3 block is generated alternating in $G_{o,2j-1}$ and $G_{e,2j}$. After $\mathcal{O}(m)$ steps the unitary matrix pencil is reduced to a Schur parameter pencil.

In summary, a backward stable algorithm to solve the inverse eigenvalue problem for Schur parameter pencils results; due to its length we omit it here, the algorithm is given in [9]. An efficient implementation of the algorithm requires $\approx 32m^2 - 35m$ arithmetic operations. A mathematically equivalent algorithm which manipulates matrix elements (instead of Schur parameters) requires $\mathcal{O}(m^3)$ arithmetic operations.

The algorithm is an updating procedure because it incorporates node-weight pairs one at a time. After the jth step of the algorithm, the $j \times j$ Schur parameter pencil corresponding to the first j nodes and weights has been obtained. The order in which the node-weight pairs are incorporated is optional.

As explained in the introduction, the algorithm can be used to construct the trigonometric approximation polynomial t via the sin-cos-approach (3). The algorithm can easily be modified to compute the unitary matrix $\widetilde{Q}$ which transforms $\Lambda - \lambda I$ to the Schur parameter pencil $G_o - \lambda G_e$. In order to solve the trigonometric approximation problem we only need the matrix $\widetilde{Q}_1$, the first n columns of $\widetilde{Q}$. That is, the algorithm can be curtailed so that only $\mathrm{O}(mn)$ arithmetic operations are required for the computation of the parameters $\{\gamma_j\}_{j=1}^n, \{\sigma_j\}_{j=0}^n$ and the vector $\widetilde{t}' = \widetilde{Q}_1^H D\widetilde{f}$. There is no need to form $\widetilde{Q}_1$ explicitly, the algorithm can be modified such that $\widetilde{t}'$ is updated successively.

In order to compute the least-squares solution $\widetilde{t}$ from (3), the vector $\widetilde{t} = F^{-1}\widetilde{R}_1^{-1}\widetilde{t}'$ has to be build, where $\widetilde{R}_1 = \sigma_0\kappa(G_oG_e^H, e_1, \ell), G_o, G_e \in \mathbb{C}^{n\times n}$. An algorithm for inverting $\kappa(G_oG_e^H, e_1, \ell)$ can be obtained by the following observation [8, 10]. Let $S = (s_1, \ldots, s_n)$ be the inverse of $\kappa(G_oG_e^H, e_1, \ell)$. Then s_{2k} is a permutation of the $2k$th column of the inverse of the Krylov matrix $K(H, e_1, 2\ell)$ and s_{2k+1} is a permutation of the $(2k+1)$st column of the inverse of the Krylov matrix $K(\overline{H}, e_1, 2\ell) = \overline{K(H, e_1, 2\ell)}$:

$$\begin{array}{rcl} (e_1, He_1, H^2e_1, ..., H^{2\ell}e_1)\widehat{I}_{2k}s_{2k} & = & e_{2k}, \\ (e_1, \overline{H}e_1, \overline{H}^2e_1, ..., \overline{H}^{2\ell}e_1)\widehat{I}_{2k+1}s_{2k+1} & = & e_{2k+1}. \end{array}$$

Since $(K(H, e_1, 2\ell))^H K(H, e_1, 2\ell)$ is a Toeplitz matrix, the inverse $T = (t_1, ..., t_n)$ of $K(H, e_1, 2\ell)$ can be computed by the simple modification of the Levinson algorithm given in the previous section. Thus, an algorithm to invert $\kappa(G_oG_e^H, e_1, k)$ is given by

```
$s_1 = e_1$
for $j = 1, 2, ..., n-1$
        $t_{j+1} = \sigma_j^{-1}(Jt_j + \gamma_j \widetilde{I}_j \overline{t}_j)$
        if $j+1$ even
        then $s_{j+1} = \widehat{I}_{j+1}^{-1} t_{j+1}$
        else $s_{j+1} = \widehat{I}_{j+1}^{-1} \overline{t}_{j+1}$
        end if
end for
```

where $J, \widetilde{I}_j$ as before and

$$\begin{aligned}\widehat{I}_{2j+1}^{-1} &= (e_{2j}, e_{2j-2}, e_{2j-4}, ..., e_4, e_2, e_1, e_3, ..., e_{2j-1}, e_{2j+1}, e_{2j+2}, ..., e_N),\\ \widehat{I}_{2j}^{-1} &= (e_{2j-1}, e_{2j-3}, e_{2j-5}, ..., e_3, e_1, e_2, e_4, ..., e_{2j-2}, e_{2j}, e_{2j+1}, ..., e_N).\end{aligned}$$

An $\mathcal{O}(n^2)$ algorithm for computing $\widetilde{R}_1^{-1}b$ for any vector $b \in \mathbb{C}^n$ using the above recursion can be given [8, 10]. Using this, the solution vector $\widetilde{t} = F^{-1}\widetilde{R}_1^{-1}\widetilde{t}'$ can be computed. This is also an updating process, as the Schur parameters are incorporated successively.

Altogether, the algorithm to construct the least-squares solution $\widetilde{t}$ of (3) requires $\mathcal{O}(mn + n^2)$ arithmetic operations.

Using the algorithms discussed in this section, the least-squares problem (3) is solved via factoring $D\widetilde{A} = \frac{\sigma_0}{2}\widetilde{Q}\widetilde{R}F$ where $\widetilde{R} = \kappa(G_o G_e^H, e_1, \ell)$ and $\widetilde{Q}(\Lambda - \lambda I)\widetilde{Q}^H G_e = G_o - \lambda G_e$. As $D\widetilde{A}$ is a real $m \times n$ matrix there exists a unique, skinny real-valued QR decomposition of $D\widetilde{A}$

$$D\widetilde{A} = \widehat{Q}_1 \widehat{R}_1 \quad \text{with} \quad \widehat{Q}_1 \in \mathbb{R}^{m\times n}, \widehat{R}_1 \in \mathbb{R}^{n\times n}$$

where the diagonal elements of $\widehat{R}_1$ are positive. This can easily be obtained from the factorization $\widetilde{Q}\widetilde{R}F$ by taking a closer look at the structure of $\widetilde{R}F$ [10, 8]. This reveals that there exists a unitary blockdiagonal matrix $\mathcal{C}$ such that $\widehat{R} = \mathcal{C}\widetilde{R}F$ is a real-valued, upper triangular matrix with positive diagonal elements. Let $\widehat{Q} = \widetilde{Q}\mathcal{C}$. Then a QR decomposition of $D\widetilde{A}$ is given by

$$D\widetilde{A} = \frac{\sigma_0}{2}\widehat{Q}\widehat{R} = \frac{\sigma_0}{2}\widehat{Q}_1\widehat{R}_1$$

where $\widehat{Q}_1$ corresponds to the first n columns of $\widehat{Q}$ and $\widehat{R}_1$ is the upper $n \times n$ block of $\widehat{R}$. Since $\widehat{R}$ has positive diagonal elements, this QR decomposition has to be unique, and $\widehat{Q}_1$ and $\widehat{R}_1$ are real-valued matrices with $\widehat{Q}_1 \in \mathbb{R}^{m\times n}$ and $\widehat{R}_1 \in \mathbb{R}^{n\times n}$. The minimum norm solution $\widetilde{t}$ of the least-squares problem (3) is obtained by

$$\widetilde{t} = 2\sigma_0^{-1}\widehat{R}_1^{-1}\widehat{Q}_1^H D\widetilde{f}.$$

Fast and efficient algorithms to compute $\widehat{Q}_1$ and $\widehat{R}_1$ using only real arithmetic (all algorithms discussed so far use complex arithmetic!) can be developed. In order to keep this paper at a reasonable length, we omit details of this approach here. Observing the effect of the transformation $\widehat{Q}$ on the real and imaginary part of $\Lambda = diag(z_1, \ldots, z_m) = diag(\cos\theta_1, \ldots, \cos\theta_m) + \imath\ diag(\sin\theta_1, \ldots, \sin\theta_m)$, it can be shown that essentially only the real part of Λ is needed for computing $\widehat{Q}$ and $\widehat{R}$. The algorithms are updating schemes as they incorporate two node-weight pairs at a time.The algorithm to construct the least-squares solution $\widetilde{t}$ of (3) using only real arithmetic requires $\mathcal{O}(mn+n^2)$ arithmetic operations. For a detailed discussion see [10, 8].

3 Downdating

Let the trigonometric polynomial $t(\theta) = a_0 + \sum_{j=1}^{\ell}(a_j \cos j\theta + b_j \sin j\theta)$ be the optimal solution of the approximation problem (2) corresponding to the data $Y_m = \{\theta_k, \omega_k^2\}_{k=1}^m$. Suppose Y_{m+1} is obtained from Y_m by augmenting a new node-weight pair $(\theta_{m+1}, \omega_{m+1}^2)$. Solving the approximation problem for Y_m assuming the knowledge of its solution for Y_{m+1} is called *downdating* the least-squares fit.

Assume the solution t corresponding to Y_{m+1} is obtained by the updating method discussed in the Section 2.1. The problem of downdating the optimal trigonometric approximation t of (2) can then be expressed as follows:

Given

$\sigma_0 > 0$	
H_{m+1}	unitary upper Hessenberg matrix of size $(m+1) \times (m+1)$
d_{m+1}	a vector of length $m+1$
(λ, ν^2)	a node-weight pair from Y_{m+1}

$(\sigma_0, H_{m+1}, d_{m+1}$ representing the solution of (2) for some data set $Y_{m+1} = \{\theta_k, \omega_k^2\}_{k=1}^{m+1})$ find $\sigma_0 > 0$, a vector d_m and a unitary upper Hessenberg matrix H_m such that

1. the eigenvalues of H_m are $\{e^{\imath\theta_k}\}_{k=1}^{m+1} \setminus e^{\imath\lambda}$
2. the vector d_m contains the first components of the eigenvectors of H_m, that is if the entries of d_{m+1} are $\delta_1/\sigma_0, \ldots, \delta_{m+1}/\sigma_0$ and $\sigma_0 = (\sum_{k=1}^{m+1} \delta_k^2)^{\frac{1}{2}}$, then the new σ_0 will be $\sigma_0 = (\sigma_0^2 - \nu^2)^{\frac{1}{2}}$ and the entries of d_m are $\{\delta_k\}_{k=1}^{m+1} \setminus \nu$ normalized by σ_0.

A similar definition of the downdating process in terms of Schur parameter pencil can be given.

3.1 The Vandermonde Approach

Assume the optimal least-squares solution t of (2) corresponding to the data $Y_{m+1} = \{\theta_k, \omega_k^2\}_{k=1}^{m+1}$ has been computed via the algorithms discussed in Section 2.1. Then a unitary matrix U and a unitary upper Hessenberg matrix H_{m+1} is known such that

$$U^H \Lambda U = H_{m+1}, \quad Ue_1 = \sigma_0^{-1}(\omega_1, \ldots, \omega_{m+1})^T$$

where

$$\sigma_0 = \sqrt{\sum_{k=1}^{m+1} \omega_k^2}, \qquad \Lambda = diag(\lambda_1, \ldots, \lambda_{m+1}), \qquad \lambda_k = e^{\imath\theta_k}.$$

Let (θ_j, ω_j^2) be the node-weight pair to be deleted from the solution. Using the knowledge of the above solution, we wish to construct a unitary upper Hessenberg matrix H_m such that

$$W^H \widehat{\Lambda} W = H_m,$$

where

$$\begin{aligned}
\widehat{\Lambda} &= diag(\lambda_1, \ldots, \lambda_{j-1}, \lambda_{j+1}, \ldots, \lambda_{m+1}),\\
We_1 &= \widehat{\sigma}_0^{-1}(\omega_1, \ldots, \omega_{j-1}, \omega_{j+1}, \ldots, \omega_{m+1})^T,\\
\widehat{\sigma}_0 &= (\sigma_0^2 - \omega_j^2)^{\frac{1}{2}}
\end{aligned}$$

or some suitable permutation of $\widehat{\Lambda}$ and We_1. In other words, we wish to determine a unitary matrix V such that

$$\begin{pmatrix} 1 & 0 \\ 0 & V^H \end{pmatrix} \begin{pmatrix} \delta & \sigma_0 e_1^T \\ \sigma_0 e_1 & H_{m+1} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & V \end{pmatrix} = \begin{pmatrix} \delta & \widehat{\sigma}_0 \widehat{e}_1^T & \omega_j \\ \widehat{\sigma}_0 \widehat{e}_1 & H_m & 0 \\ \omega_j & 0 & \lambda_j \end{pmatrix}.$$

Then, as $\begin{pmatrix} \widehat{\Lambda} & \\ & \lambda_j \end{pmatrix} = P_j^H \Lambda P_j$ where $P_j = (e_1, \ldots, e_{j-1}, e_{j+1}, \ldots, e_{m+1}, e_j)$

$$\begin{pmatrix} W & \\ & 1 \end{pmatrix} = P_j^H U V.$$

Applying one step of the standard QR algorithm with the exact shift λ_j to the matrix H_{m+1} determines a unitary matrix $\widetilde{V}$ such that

$$\widetilde{V}^H H_{m+1} \widetilde{V} = \begin{pmatrix} \widetilde{H} & \\ & \lambda_j \end{pmatrix}$$

because λ_j is an eigenvalue of H_{m+1}. $\widetilde{H}$ is a unitary upper Hessenberg matrix. As H_{m+1} is an upper Hessenberg matrix, $\widetilde{V}$ has to be of upper Hessenberg form as well. Hence, the vector $\sigma_0 e_1$ will not be transformed as required as $\widetilde{V}^H e_1$ is a full vector.

Applying one step of an RQ algorithm with the exact shift λ_j to H_{m+1} determines an upper triangular matrix R and a unitary upper Hessenberg matrix Q such that $H_{m+1} - \lambda_j I = RQ$ and

$$Q H_{m+1} Q^H = QR + \lambda_j I = \begin{pmatrix} \widehat{H} & \\ & \lambda_j \end{pmatrix}.$$

$\widehat{H}$ is a unitary upper Hessenberg matrix. The vector $\sigma_0 e_1$ is transformed such that only the first two entries are nonzero

$$\sigma_0 Q e_1 = (x, x, 0, \ldots, 0)^T.$$

Observe that with the reversal matrix $J = (e_{m+1}, e_m, \ldots, e_1)$

$$\begin{aligned}
H_{m+1} = RQ \quad &\Leftrightarrow \quad J H_{m+1} J = (JRJ)(JQJ)\\
&\Leftrightarrow \quad \underbrace{J H_{m+1}^T J}_{\nabla} = \underbrace{(JQ^T J)}_{\nabla} \underbrace{(JR^T J)}_{\nabla}
\end{aligned}$$

and that if $H_{m+1} = G_1(\gamma_1) G_2(\gamma_2) \cdots G_{m+1}(\gamma_{m+1})$ then

$$H^P := J H_{m+1}^T J = G_1(\tilde{\gamma}_1) G_2(\tilde{\gamma}_2) \cdots G_{m+1}(\tilde{\gamma}_{m+1}).$$

Hence applying the RQ algorithm to H_{m+1} is equivalent to applying the QR algorithm to H^P. One iteration of the QR algorithm with the exact shift λ_j applied to H^P generates a unitary upper Hessenberg matrix

$$V = G_1(\beta_1)G_2(\beta_2)\cdots G(\beta_{m+1})$$

such that

$$V^H H^P V = \begin{pmatrix} H' & 0 \\ 0 & \lambda_j \end{pmatrix}. \tag{6}$$

Moreover, β_{m+1} can be taken to be an arbitrary unimodular number, because deflation has taken place [12].

Only the last two components of $\sigma_0 V^H e_{m+1}$ are nonzero; they are given by

$$\begin{pmatrix} 1 & 0 \\ 0 & -\overline{\beta_{m+1}} \end{pmatrix}\begin{pmatrix} -\overline{\beta_m} & (1-|\beta_m|^2)^{\frac{1}{2}} \\ (1-|\beta_m|^2)^{\frac{1}{2}} & \beta_m \end{pmatrix}\begin{pmatrix} 0 \\ \sigma_0 \end{pmatrix} = \begin{pmatrix} (1-|\beta_m|^2)^{\frac{1}{2}}\sigma_0 \\ -\overline{\beta_{m+1}}\beta_m\sigma_0 \end{pmatrix}.$$

We can choose $\beta_{m+1} = -\beta_m/|\beta_m|$ to obtain

$$\begin{pmatrix} (1-|\beta_m|^2)^{\frac{1}{2}}\sigma_0 \\ |\beta_m|\sigma_0 \end{pmatrix}.$$

Transforming (6) by similarity using $\hat{J} = (e_m, e_{m-1}, \ldots, e_1, e_{m+1})$ and transposing the result, we obtain

$$W^H H_{m+1} W = \begin{pmatrix} H'' & 0 \\ 0 & \lambda_j \end{pmatrix}, \quad \text{where} \quad W = J\overline{V}\hat{J}.$$

Moreover,

$$\sigma_0 W^H e_1 = \begin{pmatrix} (1-|\beta_m|^2)^{\frac{1}{2}}\sigma_0\hat{e}_1 \\ |\beta_m|\sigma_0 \end{pmatrix},$$

and by the uniqueness of the reduction, $H'' = H_m$, $\sigma_0(1-|\beta_m|^2)^{\frac{1}{2}} = \hat{\sigma}_0$, and $|\beta_m|\sigma_0 = \omega_j$.

Note that the downdating process requires knowledge of the node θ_j to be deleted, but not of the corresponding weight ω_j^2. In an implementation of the process the computed value $|\beta_m|\sigma_0$ can therefore be used to assess the accuracy of the computation.

If the optimal least-squares solution t of (2) corresponding to the data Y_{m+1} has been computed via the algorithms discussed in Section 2.1, the least-squares solution $\hat{c} = R_1^{-1}Q_1^H Dg \in \mathbb{C}^{m+1}$ of (4) is known. The optimal least-squares solution t of (2) corresponding to the data Y_m is then obtained by applying W^H to $c' = Q_1^H Dg$ incrementally

$$W^H c' = \begin{pmatrix} c'' \\ \omega_j g(z_j) \end{pmatrix}.$$

In a second step a new $\hat{c} \in \mathbb{C}^m$ has to be computed from c'' using the Schur parameters of H_m via the simplified Levinson algorithm discussed in Section 2.1.

This downdating procedure was first described by Ammar, Gragg and Reichel in [3]. They present an $\mathcal{O}(m)$ algorithm which is based on the unitary QR algorithm introduced by Gragg in [12]. The transition from a unitary $n \times n$ upper Hessenberg matrix $H = G_1(\gamma_1) \cdots G_n(\gamma_n)$ to $H_+ = RQ + \mu I = Q^H HQ$ is computed implicitly without explicitly forming $H - \mu I$. It is well known that as H is upper Hessenberg Q is a unitary upper Hessenberg matrix. Hence Q has a factorization of the form

$$Q = Q_1 Q_2 \cdots Q_n$$

with

$$Q_k = G_k(\alpha_k), \qquad |\alpha_k|^2 + \beta_k^2 = 1, \beta_n = 0.$$

In a preparatory step, a unitary matrix $Q_1 = G_1(\alpha_1)$ is determined such that

$$(H - \mu I)e_1 = \rho_1 Q_1 e_1.$$

Let

$$H_k = Q_k^H H_{k-1} Q_{k-1}, \quad H_0 = H, \quad Q_0 = I.$$

The matrix H_1 is a unitary upper Hessenberg matrix. The matrix $H_1 Q_1$ is a matrix of the form

$$\begin{pmatrix} x & x & x & x & \cdots & x \\ x & x & x & x & \cdots & x \\ \otimes & x & x & x & \cdots & x \\ & & x & x & \cdots & x \\ & & & \ddots & \ddots & \vdots \\ & & & & x & x \end{pmatrix}.$$

Hence, $H_1 Q_1$ is upper Hessenberg, apart from an additional entry in the position $(3, 1)$. The remaining matrices $Q_2, \ldots, Q_n$ are chosen in order to chase this additional element down the subdiagonal. That is, suppose H_k is a unitary upper Hessenberg matrix. Then $H_k Q_k$ is upper Hessenberg, apart from an additional entry in the position $(k+2, k)$. Premultiplication of $H_k Q_k$ by Q_{k+1}^H to form H_{k+1} must create a zero in position $(k+2, k)$. Making use of the factorization of H and Q, and the fact that the resulting matrix H_+ has to have a factorization as well, an efficient unitary QR-step is developed by Gragg in [12].

3.2 The sin-cos-approach

Assume the optimal least-squares solution t of (2) corresponding to the data $Y_{m+1} = \{\theta_k, \omega_k^2\}_{k=1}^{m+1}$ has been computed via the algorithms discussed in Section 2.2. Then a unitary matrix Q and a Schur parameter pencil $G_o^{m+1} - \lambda G_e^{m+1}$ is known such that

$$Q(\Lambda - \lambda I)Q^H G_e^{m+1} = G_o^{m+1} - \lambda G_e^{m+1}, \quad Q^H e_1 = \sigma_0^{-1}(\omega_1, \ldots, \omega_{m+1})^T$$

where

$$\sigma_0 = \sqrt{\sum_{k=1}^{m+1} \omega_k^2}, \qquad \Lambda = diag(\lambda_1, \ldots, \lambda_{m+1}), \qquad \lambda_k = e^{\imath\theta_k}.$$

Instead of the above equation, we can just as well consider the equivalent equation

$$Q\Lambda Q^H = G_o^{m+1}(G_e^{m+1})^H.$$

Let (θ_j, ω_j^2) be the node-weight pair to be deleted from the solution. Using the knowledge of the above solution, we wish to construct a Schur parameter pencil $G_o^m - \lambda G_e^m$ or equivalently a matrix $G_o^m(G_e^m)^H$ such that

$$\widehat{W}\widehat{\Lambda}\widehat{W}^H = G_o^m(G_e^m)^H,$$

where

$$\begin{aligned}
\widehat{\Lambda} &= diag(\lambda_1, \ldots, \lambda_{j-1}, \lambda_{j+1}, \ldots, \lambda_{m+1}),\\
\widehat{W}^H e_1 &= \widehat{\sigma}_0^{-1}(\omega_1, \ldots, \omega_{j-1}, \omega_{j+1}, \ldots, \omega_{m+1})^T,\\
\widehat{\sigma}_0 &= (\sigma_0^2 - \omega_j^2)^{\frac{1}{2}}
\end{aligned}$$

or some suitable permutation of $\widehat{\Lambda}$ and $\widehat{W}^H e_1$. In other words, we wish to determine a unitary matrix V such that

$$\begin{pmatrix} 1 & 0 \\ 0 & V \end{pmatrix} \begin{pmatrix} \delta & \sigma_0 e_1^T \\ \sigma_0 e_1 & G_o^{m+1}(G_e^{m+1})^H \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & V^H \end{pmatrix}$$
$$= \begin{pmatrix} \delta & \widehat{\sigma}_0 \widehat{e}_1^T & \omega_j \\ \widehat{\sigma}_0 \widehat{e}_1 & G_o^m(G_e^m)^H & 0 \\ \omega_j & 0 & \lambda_j \end{pmatrix}.$$

Then, as $\begin{pmatrix} \widehat{\Lambda} & \\ & \lambda_j \end{pmatrix} = P_j^H \Lambda P_j$ where $P_j = (e_1, \ldots, e_{j-1}, e_{j+1}, \ldots, e_{m+1}, e_j)$

$$\begin{pmatrix} \widehat{W} & \\ & 1 \end{pmatrix} = P_j^H Q^H V^H.$$

Analogous to the ideas of Ammar, Gragg and Reichel in [3] we want to compute W via a QR-type step. In [4] a QR-like algorithm for Schur parameter pencils is introduced. The method is based on the standard QR algorithm applied to a matrix of the form $G_o G_e^H$. No initial reduction to Hessenberg form is performed. It is shown that each iterate is then of the same form as $G_o G_e^H$ again. Hence, applying one QR-step with the exact shift λ_j to the matrix $G_o^{m+1}(G_e^{m+1})^H$ determines a unitary matrix $\widetilde{V}$ such that

$$\widetilde{V} G_o^{m+1}(G_e^{m+1})^H \widetilde{V}^H = \begin{pmatrix} X & \\ & \lambda_j \end{pmatrix}$$

because λ_j is an eigenvalue of $G_o^{m+1}(G_e^{m+1})^H$. X is a unitary matrix of the same form as $G_o^m(G_e^m)^H$. $\widetilde{V}$ is a matrix of the form $G_1(G_3G_2G_3)(G_5G_4G_5)\cdots$. Hence, the vector $\sigma_0 e_1$ will not be transformed as required as $\widetilde{V}e_1$ is a full vector.

Applying one step of an RQ algorithm with the exact shift λ_j to $G_o^{m+1}(G_e^{m+1})^H$ determines an upper triangular matrix R and a unitary U of the form

$$G_1(G_3G_2G_3)(G_5G_4G_5)\cdots = G_1G_3G_5\cdots G_2G_3G_4G_5\cdots$$

that is, $U = G_o H$ for some upper Hessenberg matrix H and a matrix of the form G_o. R and U are determined such that $G_o^{m+1}(G_e^{m+1})^H - \lambda_j I = RU$ and

$$UG_o^{m+1}(G_e^{m+1})^H U^H = UR + \lambda_j I = \begin{pmatrix} \widehat{X} & \\ & \lambda_j \end{pmatrix}.$$

The vector $\sigma_0 e_1$ is transformed such that only the first two entries are nonzero

$$\sigma_0 U e_1 = (x, x, 0, \ldots, 0)^T.$$

Observe that with the reversal matrix $J = (e_{m+1}, e_m, \ldots, e_1)$

$$\begin{aligned} G_o^{m+1}(G_e^{m+1})^H = RU \quad &\Leftrightarrow \quad JG_o^{m+1}(G_e^{m+1})^H J = (JRJ)(JUJ) \\ &\Leftrightarrow \quad J(G_o^{m+1}(G_e^{m+1})^H)^T J = (JU^T J)(JR^T J). \end{aligned}$$

Let $G^P := J(G_o^{m+1}(G_e^{m+1})^H)^T J$ and let us assume for simplicity that $m+1$ is odd (a similar argumentation can be given in the case that $m+1$ is even). Then

$$G_o^{m+1}(G_e^{m+1})^H = G_1(\gamma_1)G_3(\gamma_3)\cdots G_{m+1}(\gamma_{m+1})G_2^H(\gamma_2)G_4^H(\gamma_4)\cdots G_m^H(\gamma_m)$$

and

$$G^P = G_1(\overline{\gamma_m})G_3(\overline{\gamma_{m-2}})\cdots G_{m-1}(\overline{\gamma_2})G_0(\gamma_{m+1})G_2^H(\gamma_{m-1})G_4^H(\gamma_{m-3})\cdots G_m^H(\gamma_1)$$

where

$$G_0(\gamma) = diag(-\gamma, I).$$

That is, $G_o^{m+1}(G_e^{m+1})^H$ and G^P are of the same form. Further, as U is a matrix of the form $G_o H$ for some upper Hessenberg matrix H and a matrix of the form G_o, $JU^T J$ is a matrix of the same form. Hence, as $JR^T J$ is upper triangular again, applying the RQ algorithm to $G_o^{m+1}(G_e^{m+1})^H$ is equivalent to applying the QR algorithm to G^P. One iteration of the QR algorithm with the exact shift λ_j applied to G^P generates a unitary matrix V

$$G_1(\beta_1)G_3(\beta_3)\cdots G_{m-1}(\beta_{m-1})G_2(\delta_2)G_3(\delta_3)\cdots G_{m+1}(\delta_{m+1})$$

such that

$$V^H G^P V = \begin{pmatrix} X' & 0 \\ 0 & \lambda_j \end{pmatrix}. \tag{7}$$

Moreover, δ_{m+1} can be taken to be an arbitrary unimodular number, because deflation has taken place.

Only the last two components of $\sigma_0 V^H e_{m+1}$ are nonzero; they are given by

$$\begin{aligned} &\begin{pmatrix} 1 & 0 \\ 0 & -\overline{\delta_{m+1}} \end{pmatrix} \begin{pmatrix} -\overline{\delta_m} & (1-|\delta_m|^2)^{\frac{1}{2}} \\ (1-|\delta_m|^2)^{\frac{1}{2}} & \delta_m \end{pmatrix} \begin{pmatrix} 0 \\ \sigma_0 \end{pmatrix} \\ &= \begin{pmatrix} (1-|\delta_m|^2)^{\frac{1}{2}}\,\sigma_0 \\ -\overline{\delta_{m+1}}\,\delta_m\,\sigma_0 \end{pmatrix}. \end{aligned}$$

We can choose $\delta_{m+1} = -\delta_m/|\delta_m|$ to obtain

$$\begin{pmatrix} (1-|\delta_m|^2)^{\frac{1}{2}}\sigma_0 \\ |\delta_m|\sigma_0 \end{pmatrix}.$$

Transforming (7) by similarity using $\hat{J} = (e_m, e_{m-1}, \ldots, e_1, e_{m+1})$ and transposing the result, we obtain

$$WG_o^{m+1}(G_e^{m+1})^H W^H = \begin{pmatrix} X'' & 0 \\ 0 & \lambda_j \end{pmatrix}, \quad \text{where} \quad W = J\overline{V}\hat{J}.$$

Moreover,

$$\sigma_0 W^H e_1 = \begin{pmatrix} (1-|\delta_m|^2)^{\frac{1}{2}}\sigma_0 \widehat{e}_1 \\ |\delta_m|\sigma_0 \end{pmatrix},$$

and by the uniqueness of the reduction, $X'' = G_o^m(G_e^m)^H$, $\sigma_0(1-|\delta_m|^2)^{\frac{1}{2}} = \widehat{\sigma}_0$, and $|\delta_m|\sigma_0 = \omega_j$.

Note that the downdating process requires knowledge of the node θ_j to be deleted, but not of the corresponding weight ω_j^2. In an implementation of the process the computed weight can therefore be used to assess the accuracy of the computation.

If the optimal least-squares solution t of (2) corresponding to the data Y_{m+1} has been computed via the algorithm based on the generalized inverse unitary eigenproblem discussed in Section 2, the least-squares solution $\widetilde{t} = \widetilde{R}_1^{-1}\widetilde{Q}_1^H D\widetilde{f} \in \mathbb{C}^{m+1}$ of (3) is known. The optimal least-squares solution t of (2) corresponding to the data Y_m is then obtained by applying W^H to $t' = \widetilde{Q}_1^H D\widetilde{f}$ incrementally

$$W^H t' = \begin{pmatrix} t'' \\ \omega_j f(z_j) \end{pmatrix}.$$

In a second step a new $\widetilde{t} \in \mathbb{C}^m$ has to be computed from t'' using the Schur parameters of $G_o^m(G_e^m)^H$ via a simplified Levinson algorithm. This is analogous to the second step of the updating procedure. There first $\widetilde{Q}_1^H D\widetilde{f}$ is computed via solving the generalized inverse unitary eigenproblem. Then $\widetilde{R}_1^{-1}$ is computed using the Schur parameters of G_o and G_e via a simplified Levinson algorithm. Details are given in [10].

A QR step has to be applied to a matrix X of the form $G_o G_e^H$. We get a unitary matrix $X' = V^H X V$, where X' can be written as $G_o'(G_e')^H$ again. If X corresponds to an unreduced Schur parameter pencil $G_o - \lambda G_e$, then X' will correspond to an unreduced Schur parameter pencil $G_o' - \lambda G_e'$. The transformation to an unreduced Schur parameter pencil is uniquely determined, up to unitary scaling, if the first column of the transformation matrix is given. Therefore one can derive $G_o' - \lambda G_e'$, up to scaling from $G_o - \lambda G_e$ by any unitary transformation $Q^H(G_o - \lambda G_e)P$ to Schur parameter pencils, for which the first column of Q coincides with a scalar multiple of the first column of V. This was used by Bunse-Gerstner and Elsner in [4] to derive an implicit single shifted QR step.

In a preparatory step, a matrix $V_1 = G_1(\alpha_1)$ is determined such that

$$V_1^H(G_o - \lambda_j G_e)e_1 = \rho e_1.$$

The pencil $V_1^H(G_o - \lambda G_e)$ differs from a Schur parameter pencil only by three additional entries

$$\begin{pmatrix} x & x & & & \\ x & x & & & \\ & & x & x & \\ & & x & x & \\ & & & & \ddots \end{pmatrix} - \lambda \begin{pmatrix} x & + & + & & \\ + & x & x & & \\ & x & x & & \\ & & & x & x \\ & & & x & x \end{pmatrix}.$$

This "bulge" is then chased down along the diagonal to restore the Schur parameter pencil form again. This is an $\mathcal{O}(m)$ process.

4 Numerical examples

We present some numerical examples that compare the accuracy of the following methods for solving the trigonometric approximation problem (2):

- AGR : the algorithm proposed in [15] as sketched in the introduction. The least-squares problem (4) $||DAc - Dg||_2 = min$ is solved via QR decomposition of DA, where the desired Q-factor of the QR decomposition is computed by an inverse unitary Hessenberg eigenvalue problem and the inverse of the upper square subblock of R is computed by an algorithm closely related to the Levinson algorithm (as explained in Section 2.1)

- pencil : the algorithm proposed in [8] as sketched in the introduction. The least-squares problem (3) $||D\tilde{f} - D\tilde{A}\tilde{t}||_2 = min$ is solved via QR decomposition of $D\tilde{A}$, where the desired Q-factor of the QR decomposition is computed by an inverse eigenvalue problem for Schur parameter pencils and the inverse of the upper square subblock of R is computed by an algorithm closely related to the Levinson algorithm (as explained in Section 2.2)

- linpack : The least-squares problem (3) $||D\tilde{f} - D\tilde{A}\tilde{t}||_2 = min$ is solved via the explicit formation of the matrix $D\tilde{A}$ and the use of the LINPACK [6] routines sqrdc and sqrsl (with real arithmetic)

For comparison of accuracy we compute the solution $\tilde{t}_d$ of the system $min||D\tilde{A}\tilde{t} - D\tilde{f}||_2$ in double precision using the NAG routine F04AMF. The figures display the relative error $||\tilde{t} - \tilde{t}_d||_2/||\tilde{t}_d||_2$ where $\tilde{t}$ is the coefficient vector computed in single precision by the method under consideration. Each graph displays the errors for $m = 50$ and increasing values of n. The arguments of the nodes are either equispaced in the interval $[0, \pi)$, $[0, 3/2\pi)$ or $[0, 2\pi)$ or the arguments are randomly generated uniformly distributed numbers in $[0, 2\pi)$. The weights are all equal to one, the elements of the real vector $\tilde{f}$ are randomly generated uniformly distributed numbers in $[-5, 5]$.

A comparison of the methods AGR, linpack and pencil is given in Figure 1. The graphs at the top of Figure 1 display the relative errors in the coefficient vectors for equispaced nodes in intervals smaller than 2π. As n increases, and the problem becomes more ill conditioned, the LINPACK routines are the first to produce inaccurate results. The method pencil produces errors that are somewhat smaller than AGR. The graphs at the bottom of Figure 1 display the relative error when the arguments are equispaced in $[0, 2\pi)$ and when the arguments are randomly generated uniformly distributed numbers in $[0, 2\pi)$. In the first case, the LINPACK routines and pencil produce smaller errors than AGR. Note that in this case we are computing the Fourier transform and thus the FFT is a better method for solving this problem. When the arguments are randomly generated uniformly distributed points in $[0, 2\pi)$ the least-squares problem is relatively well conditioned and the algorithms AGR, and pencil yield roughly the same accuracy as n gets close to m.

We obtained similar results to those in Figure 1 with other choices for the nodes and the weights. The numerical experiments have shown that generally the method pencil produces more accurate results than the method AGR. The method linpack produces inaccurate results first. AGR, and pencil are algorithms to solve the trigonometric approximation problem in $\mathcal{O}(mn)$ arithmetic operations, while the method linpack requires $\mathcal{O}(mn^2)$ operations. For more numerical examples and a more detailed discussion on the updating procedures see [8].

——	AGR
— · —	pencil
· · · · · ·	linpack

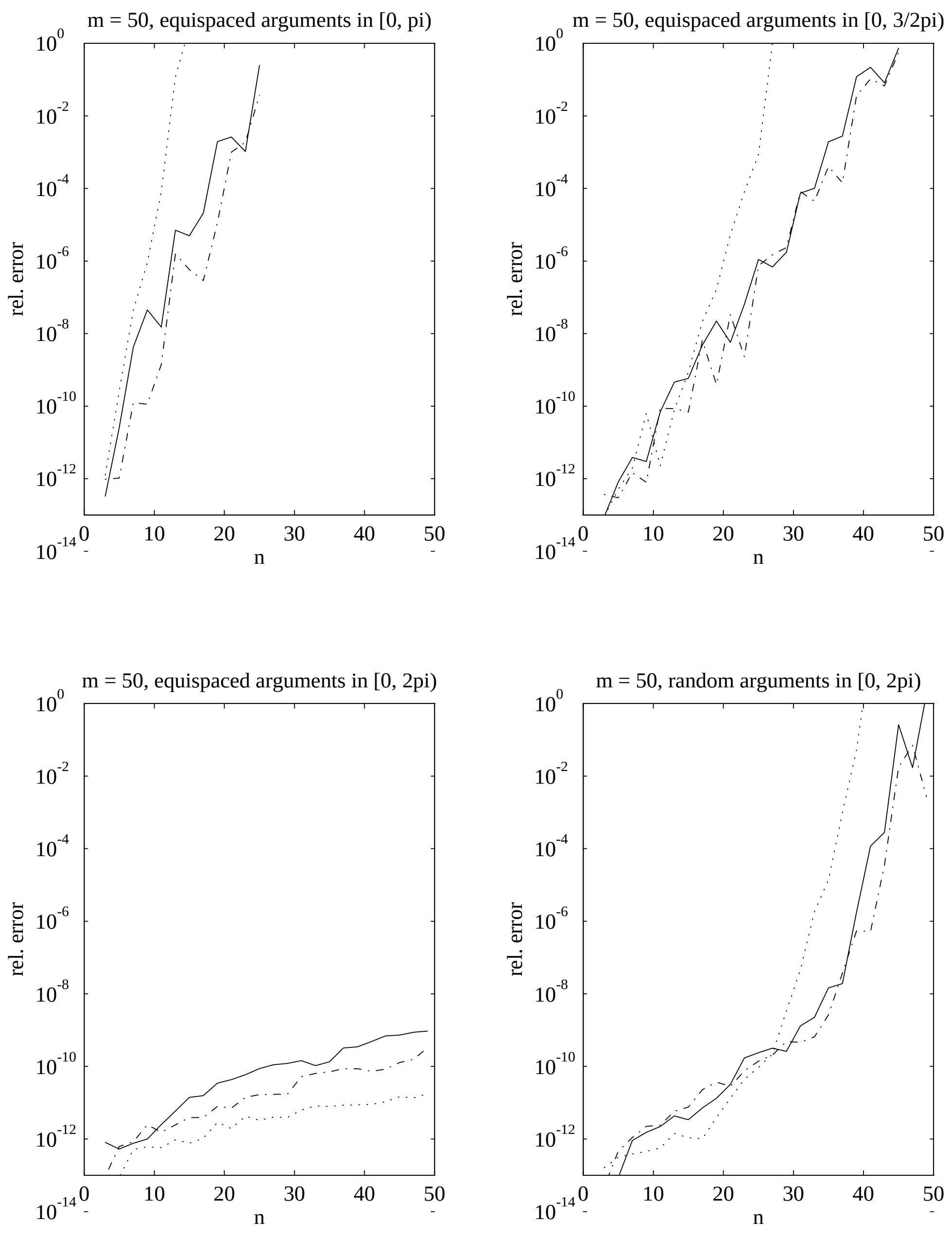

Figure 1:

We will refrain from giving numerical examples for the downdating procedures. For numerical examples and a detailed discussion on the downdating procedure as described in Section 3.1 see [3].

References

[1] G. S. Ammar, W. B. Gragg, and L. Reichel. On the Eigenproblem for Orthogonal Matrices. In *Proc. 25th IEEE Conference on Decision and Control*, pages 1963 – 1966, 1986.

[2] G. S. Ammar, W. B. Gragg, and L. Reichel. Constructing a Unitary Hessenberg Matrix from Spectral Data. In G.H. Golub and P. Van Dooren, editors, *Numerical Linear Algebra, Digital Signal Processing, and Parallel Algorithms*, pages 385–396. Springer-Verlag, Berlin, 1991.

[3] G.S. Ammar, W.B. Gragg, and L. Reichel. Downdating of Szegö Polynomials and Data Fitting Applications. *Lin. Alg. and its Applic.*, 172:315 – 336, 1992.

[4] A. Bunse-Gerstner and L. Elsner. Schur Parameter Pencils for the Solution of the Unitary Eigenproblem. *Lin. Alg. and its Appl.*, 154 - 156:741 – 778, 1991.

[5] C. deBoor and G. H. Golub. The Numerically Stable Reconstruction of a Jacobi Matrix from Spectral Data. *Lin. Alg. and its Appl.*, 21:245 – 260, 1978.

[6] J. Dongarra, J.R. Bunch, C. Moler, and G.W. Stewart. *Linpack User's Guide.* SIAM, Philadelphia, PA, 1979.

[7] S. Elhay, G.H. Golub, and J. Kautsky. Updating and downdating of orthogonal polynomials with data fitting applications. *SIAM J. Matrix Anal. Appl.*, 12:327 – 353, 1991.

[8] H. Faßbender. *Numerische Verfahren zur diskreten trigonometrischen Polynomapproximation.* Dissertation Universität Bremen, 1993.

[9] H. Faßbender. Inverse unitary eigenproblems and related orthogonal functions. *Numerische Mathematik*, to appear.

[10] H. Faßbender. On Numerical Methods for Discrete Least-Squares Approximation by Trigonometric Polynomials. *Math. Comp.*, to appear.

[11] G. H. Golub and C. F. Van Loan. *Matrix Computation.* The John Hopkins University Press, second edition, 1989.

[12] W. B. Gragg. The QR algorithm for unitary Hessenberg matrices. *J. Comp. Appl. Math.*, 16:1 – 8, 1986.

[13] W. B. Gragg and W. J. Harrod. The Numerically Stable Reconstruction of Jacobi Matrices from Spectral Data. *Numer. Math.*, 44:317 – 335, 1984.

[14] P. Henrici. *Applied and Computational Analysis*, volume 3. Wiley, 1986.

[15] L. Reichel, G. S. Ammar, and W. B. Gragg. Discrete Least Squares Approximation by Trigonometric Polynomials. *Math. Comp.*, 57:273 – 289, 1991.

[16] L.B. Scott and L.R. Scott. Efficient methods for data smoothing. *SIAM J. Numer. Anal.*, 26:681 – 692, 1989.

Berichte aus der Technomathematik ISSN 1435-7968

`http://www.math.uni-bremen.de/zetem/berichte`

— Vertrieb durch den Autor —

Reports **Stand: 10. November 1998**

98–01. Peter Benner, Heike Faßbender:
An Implicitly Restarted Symplectic Lanczos Method for the Symplectic Eigenvalue Problem, Juli 1998.

98–02. Heike Faßbender:
Sliding Window Schemes for Discrete Least-Squares Approximation by Trigonometric Polynomials, Juli 1998.

98–03. Peter Benner, Maribel Castillo, Enrique S. Quintana-Ortí:
Parallel Partial Stabilizing Algorithms for Large Linear Control Systems, Juli 1998.

98–04. Peter Benner:
Computational Methods for Linear–Quadratic Optimization, August 1998.

98–05. Peter Benner, Ralph Byers, Enrique S. Quintana-Ortí, Gregorio Quintana-Ortí:
Solving Algebraic Riccati Equations on Parallel Computers Using Newton's Method with Exact Line Search, August 1998.

98–06. Lars Grüne, Fabian Wirth:
On the rate of convergence of infinite horizon discounted optimal value functions, November 1998.