MATHEMATICS OF COMPUTATION
VOLUME 57, NUMBER 195
JULY 1991, PAGES 273–289

DISCRETE LEAST SQUARES APPROXIMATION BY TRIGONOMETRIC POLYNOMIALS

L. REICHEL, G. S. AMMAR, AND W. B. GRAGG

ABSTRACT. We present an efficient and reliable algorithm for discrete least squares approximation of a real-valued function given at arbitrary distinct nodes in $[0, 2\pi)$ by trigonometric polynomials. The algorithm is based on a scheme for the solution of an inverse eigenproblem for unitary Hessenberg matrices, and requires only $O(mn)$ arithmetic operations as compared with $O(mn^2)$ operations needed for algorithms that ignore the structure of the problem. Moreover, the proposed algorithm produces consistently accurate results that are often better than those obtained by general QR decomposition methods for the least squares problem. Our algorithm can also be used for discrete least squares approximation on the unit circle by algebraic polynomials.

1. INTRODUCTION

Let $\{\theta_k\}_{k=1}^m$ be a set of m distinct nodes in the interval $[0, 2\pi)$, let $\{w_k^2\}_{k=1}^m$ be a set of positive weights, and let $f(\theta)$ be a real-valued function whose values at the nodes θ_k are explicitly known. In this paper we present an efficient and reliable method for the construction of the trigonometric polynomial

$$t(\theta) = a_0 + \sum_{j=1}^{l} (a_j \cos j\theta + b_j \sin j\theta), \qquad a_j, b_j \in \mathbb{R}, \tag{1.1}$$

of order at most $l < m/2$ that minimizes the discrete least squares error

$$\|f - t\|_{\mathbb{R}} := \left(\sum_{k=1}^{m} |f(\theta_k) - t(\theta_k)|^2 w_k^2 \right)^{1/2}. \tag{1.2}$$

We consider the computation of the desired trigonometric polynomial in terms of the following closely related approximation problem. For complex-valued functions g and h defined at the nodes z_k, $z_k := \exp(i\theta_k)$, $1 \le k \le m$, introduce the discrete inner product on the unit circle,

$$\langle g, h \rangle := \sum_{k=1}^{m} \overline{g(z_k)} h(z_k) w_k^2, \tag{1.3}$$

Received March 12, 1990.
1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 65D10, 42A10, 65F99.
The authors' research was supported in part by NSF grant DMS-9002884 and IBM Bergen Scientific Centre.

where the bar denotes complex conjugation. Let $g(z)$ be an analytic function in $|z| < 1$ whose values $g(z_k)$ at the m distinct nodes z_k are explicitly known. We wish to compute the polynomial

$$p(z) = \sum_{j=0}^{n-1} c_j z^j \tag{1.4}$$

of degree less than $n \leq m$ that minimizes the discrete least squares error

$$\|g - p\|_{\mathbb{C}} := \langle g - p, g - p \rangle^{1/2}. \tag{1.5}$$

Introduce the vectors

$$\mathbf{g} := [g(z_1), g(z_2), \ldots, g(z_m)]^T \in \mathbb{C}^m,$$
$$\mathbf{c} := [c_0, c_1, \ldots, c_{n-1}]^T \in \mathbb{C}^n.$$

Then the coefficient vector $\mathbf{c}$ for the polynomial (1.4) that minimizes the error (1.5) is the least squares solution $\hat{\mathbf{c}}$ of

$$DA\mathbf{c} = D\mathbf{g}, \tag{1.6}$$

where

$$D = \operatorname{diag}[w_1, w_2, \ldots, w_m] \tag{1.7}$$

with $w_j := \sqrt{w_j^2}$, and A is the transposed Vandermonde matrix

$$A = \begin{bmatrix} 1 & z_1 & z_1^2 & \cdots & z_1^{n-1} \\ 1 & z_2 & z_2^2 & \cdots & z_2^{n-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & z_m & z_m^2 & \cdots & z_m^{n-1} \end{bmatrix}. \tag{1.8}$$

We compute $\hat{\mathbf{c}}$ by using the QR decomposition of DA. Since DA has full rank, there is a unique matrix $Q \in \mathbb{C}^{m \times n}$ with orthonormal columns and a unique $n \times n$ right triangular matrix R with *positive* diagonal elements such that $DA = QR$. The solution of (1.6) can be obtained from this QR decomposition by first evaluating the vector $Q^* D\mathbf{g}$, and then computing $\hat{\mathbf{c}} = R^{-1}(Q^* D\mathbf{g})$.

Algorithms that compute the QR decomposition of DA without using its structure require $O(mn^2)$ arithmetic operations [5, 9]. In this paper we present two algorithms for the solution of the least squares problem (1.6) that implicitly compute the QR decomposition of DA and require only $O(mn)$ arithmetic operations. Moreover, numerical results presented in §5 show that one of these efficient algorithms often yields higher accuracy than methods in which a QR decomposition of the matrix DA is computed without using its structure.

Our approach is based on computational aspects associated with the family of orthogonal polynomials for the inner product (1.3). Polynomials that are orthogonal with respect to an inner product on the unit circle, such as the discrete inner product (1.3), are known as *Szegő polynomials.*

Let $\{\phi_j\}_{j=0}^{m-1}$ denote the family of *orthonormal* Szegő polynomials with respect to the inner product (1.3), where ϕ_j is of degree j and has positive leading coefficient. Let $R = [r_{jk}]$ be the right triangular matrix whose nontrivial elements are determined by

$$z^{k-1} = \sum_{j=1}^{k} r_{jk}\phi_{j-1}(z), \qquad 1 \le k \le n. \tag{1.9}$$

Thus, R expresses the power basis in terms of the orthonormal Szegő polynomials. Also define the $m \times n$ matrix $Q = [q_{kj}]$ by

$$q_{kj} = \phi_j(z_k)w_k, \qquad j = 0, \ldots, n-1;\ k = 1, \ldots, m. \tag{1.10}$$

Observe that Q has orthonormal columns; that is, $Q^*Q = I$, where Q^* denotes the transposed complex conjugate of Q. Moreover, R has positive diagonal entries, and we have

$$DA = QR. \tag{1.11}$$

Thus, Q is determined by the values of the orthonormal Szegő polynomials ϕ_j, $0 \le j < n$, at the nodes z_k, $1 \le k \le m$, and the columns of R^{-1} are the coefficients of these Szegő polynomials in the power basis.

The orthonormal polynomials ϕ_j satisfy the *Szegő recurrence relations*

$$\begin{aligned} \phi_0(z) &= \tilde{\phi}_0(z) = 1/\sigma_0, \\ \sigma_{j+1}\phi_{j+1}(z) &= z\phi_j(z) + \gamma_{j+1}\tilde{\phi}_j(z), \\ \sigma_{j+1}\tilde{\phi}_{j+1}(z) &= z\overline{\gamma}_{j+1}\phi_j(z) + \tilde{\phi}_j(z), \end{aligned} \qquad j = 0, 1, \ldots, m-2, \tag{1.12}$$

where the recurrence coefficients $\gamma_{j+1} \in \mathbb{C}$ and $\sigma_{j+1} > 0$ are determined by

$$\begin{aligned} \sigma_0 = \alpha_0 &= \left(\sum_{k=1}^{m} w_k^2\right)^{1/2}, \\ \gamma_{j+1} &= -\langle 1, z\phi_j(z)\rangle/\alpha_j, \\ \sigma_{j+1} &= (1 - |\gamma_{j+1}|^2)^{1/2}, \qquad j = 0, 1, \ldots, m-2 \\ \alpha_{j+1} &= \alpha_j\sigma_{j+1}, \end{aligned} \tag{1.13}$$

(see, for example, Grenander and Szegő [12, Chapter 12]). It follows from (1.12) that

$$\tilde{\phi}_j(z) = z^j\overline{\phi}_j(1/z), \qquad j = 0, 1, \ldots, m-1. \tag{1.14}$$

Moreover, since the measure that defines (1.3) has m points of increase, we have $|\gamma_j| < 1$ for $1 \le j < m$ (and $|\gamma_m| = 1$). The coefficients γ_j are known as *Schur parameters*, and we refer to the σ_j as the associated *complementary parameters*. Although the complementary parameters $\sigma_1, \sigma_1, \ldots, \sigma_n$ are

mathematically redundant, we retain them during calculations to avoid numerical instabilities. We take σ_0^2 to be the total weight of the measure that defines (1.3).

In §2 we describe an $O(mn)$ method for determining Q that is based on the Szegő recursions (1.12) and (1.13), in which the coefficients γ_j, σ_j, and α_j, and the values $\phi_j(z_k)$, $1 \leq k \leq m$, are computed for increasing values of j. We refer to this procedure as the *Stieltjes procedure for Szegő polynomials* because it is analogous to the Stieltjes procedure that is often used to compute the recurrence coefficients and values of orthogonal polynomials that satisfy a three-term recurrence relation (see Gautschi [7, 8] and Forsythe [6]).

The Stieltjes procedure is attractive because it is efficient and easily implemented. Numerical results show that it often provides accurate answers if n is small relative to m. However, for many distributions of nodes z_k the Stieltjes procedure is very sensitive to roundoff errors for larger values of n. This sensitivity is illustrated by some computed examples in §5.

Consider for the moment the determination of polynomials that are orthogonal with respect to some discrete inner product on a finite interval, and thus satisfy a three-term recurrence relation. In matrix-theoretic terms, the construction of three-term recurrence coefficients using the classical Stieltjes procedure can be viewed as the Lanczos process for transforming a real diagonal matrix by orthogonal similarity to tridiagonal form. In fact, this is a manifestation of an inverse eigenvalue problem for tridiagonal matrices. One can therefore use other numerical methods for this inverse eigenvalue problem to generate the recurrence coefficients and, implicitly, the values of the orthogonal polynomials. In particular, an efficient method for this problem that uses elementary orthogonal similarity transformations, due to Rutishauser, is described in [11]. This method is applied to the generation of orthogonal polynomials that satisfy a three-term recurrence relation in [15], and it is observed that this method is numerically more reliable than the classical Stieltjes procedure.

The relationship between Szegő polynomials and unitary Hessenberg matrices is analogous to that between orthogonal polynomials on a finite interval and tridiagonal matrices. In particular, the Schur parameters determine a unitary Hessenberg matrix such that the characteristic polynomial of the $k \times k$ leading principal submatrix of H is the monic Szegő polynomial. In this context, the Stieltjes procedure for Szegő polynomials can be viewed as a solution method for an inverse eigenvalue problem for unitary Hessenberg matrices.

An algorithm for constructing a unitary Hessenberg matrix from spectral data using elementary unitary similarity transformations is presented in [1]. This algorithm can be regarded as an inverse QR algorithm for unitary Hessenberg matrices, and is analogous to the algorithm of [11] for tridiagonal matrices. In §3 we use this inverse eigenvalue algorithm to compute $Q^* D\mathbf{g}$ by applying a sequence of elementary unitary transformations on the vector $D\mathbf{g}$. This sequence of unitary matrices implicitly determines Q. We will see that this approach also requires only $O(mn)$ operations. Moreover, numerical results

show that the inverse unitary QR algorithm produces accurate results, which are often more accurate than general $O(mn^2)$ algorithms for computing Q.

In §4 we show that the solution $\hat{\mathbf{c}} = R^{-1}Q^*D\mathbf{g}$ of (1.6) can be computed from $Q^*D\mathbf{g}$ and the recurrence coefficients γ_j, σ_j using $O(n^2)$ arithmetic operations. This algorithm relies on the Szegő recursions (1.12) and is closely related to the Levinson algorithm. We also show how the optimal polynomial can be evaluated from $Q^*D\mathbf{g}$ and the recurrence coefficients without the explicit computation of $\hat{\mathbf{c}}$.

We have outlined an algorithm for the computation of the polynomial (1.4) that minimizes the error (1.5). It is easy to modify this algorithm in order to obtain a trigonometric polynomial (1.1) that minimizes the error (1.2).

Proposition 1.1. *Let*

$$\mathbf{f} := [f(\theta_1), f(\theta_2), \ldots, f(\theta_m)]^T \in \mathbb{R}^m, \tag{1.15}$$

and let the matrices A, D, Q, and R be defined by (1.7)–(1.10) *with $n = 2l + 1$, and let $\Lambda = \operatorname{diag}[z_k]_{k=1}^m$, where $z_k = \exp(i\theta_k)$. Denote by $\hat{\mathbf{c}} = [\hat{c}_0, \hat{c}_1, \ldots, \hat{c}_{2l}]^T$ the least squares solution of*

$$DA\mathbf{c} = D\Lambda^l\mathbf{f}. \tag{1.16}$$

Then the coefficients of the trigonometric polynomial (1.1) *that minimizes the error* (1.2) *are given by*

$$\begin{cases} a_0 = \hat{c}_l, \\ a_j = 2\operatorname{Re}(\hat{c}_{j+1}), \qquad 1 \le j \le l, \\ b_j = -2\operatorname{Im}(\hat{c}_{j+1}). \end{cases} \tag{1.17}$$

Proof. The algebraic polynomial $p(z) = \sum_{j=0}^{2l} \hat{c}_j z^j$ minimizes the error (1.5) over all algebraic polynomials of degree at most $2l$, and the associated trigonometric polynomial

$$t(\theta) = z^{-l}p(z), \qquad z = \exp(i\theta), \tag{1.18}$$

minimizes the error (1.2) over all trigonometric polynomials with complex-valued coefficients of order at most l. It remains to be shown that if $\mathbf{f} \in \mathbb{R}^m$ then the trigonometric polynomial $t(\theta)$ is real-valued, i.e., that its coefficients a_j and b_j in (1.1) are real-valued. The latter is equivalent to $J_n\bar{\hat{\mathbf{c}}} = \hat{\mathbf{c}}$, where J_n denotes the $n \times n$ reversal matrix, i.e., the columns of J_n are the columns of the identity matrix in reverse order. Since $\hat{\mathbf{c}}$ is the unique solution of the normal equations

$$A^*D^2A\hat{\mathbf{c}} = A^*D^2\Lambda^l\mathbf{f}, \tag{1.19}$$

we have

$$J_nA^*D^2AJ_nJ_n\hat{\mathbf{c}} = J_nA^*D^2\Lambda^l\mathbf{f}, \tag{1.20}$$

and from $\Lambda^{-l}AJ_n = \Lambda^l\overline{A}$ it follows that

$$A^T D^2 \overline{A}(J_n \hat{\mathbf{c}}) = A^T D^2 \overline{\Lambda}^l \mathbf{f}. \tag{1.21}$$

Complex conjugation of the linear system of equations (1.21) now shows that $J_n\bar{\hat{\mathbf{c}}}$ is also a solution of the normal equations (1.19), and therefore $J_n\bar{\hat{\mathbf{c}}} = \hat{\mathbf{c}}$. Formulas (1.17) now follow from (1.18). □

A scheme closely related to the Stieltjes procedure for Szegő polynomials for least squares approximation by trigonometric polynomials has been described by Newbery [14]. Without explicitly introducing Szegő polynomials, Newbery derives recurrence relations for $\mathrm{Re}(\phi_j(e^{i\theta}))$ and $\mathrm{Im}(\phi_j(e^{i\theta}))$, and expresses the optimal trigonometric polynomial in terms of these basis functions. Another scheme closely related to the Stieltjes procedure has recently been presented by Demeure [4, §7]. This scheme, in contrast with Algorithm 2.1 below, explicitly uses A^*A to determine the recurrence coefficients.

Finally we note that if we seek a trigonometric polynomial (1.1) that *interpolates* $f(\theta)$ at the nodes θ_k, $1 \le k \le m$, or a polynomial (1.4) that interpolates $g(z)$ at the nodes z_k, $1 \le k \le m$, then several numerical schemes are available. If the nodes z_k are *equidistant* on the unit circle, then the fast Fourier transform can be applied (see, e.g., Henrici [13, Chapter 13]). If the nodes are not equidistant, then interpolating trigonometric polynomials can be computed by the Björck-Pereyra algorithm [3] or by the use of barycentric formulas (see Berrut [2] and Henrici [13, Chapter 13]).

2. The Stieltjes procedure for Szegő polynomials

In this section we consider the computation of $\mathbf{c}' := Q^*D\mathbf{g} \in \mathbb{C}^n$, where Q is defined by (1.10) and $\mathbf{g}$ is an arbitrary vector in $\mathbb{C}^m$. The numerical method used is analogous to the Stieltjes procedure for the generation of polynomials orthogonal with respect to a measure on an interval, which has been studied by Gautschi [7, 8].

In the Stieltjes procedure for Szegő polynomials, the Szegő recursions (1.12) and (1.13) are used to successively generate the m-vectors $\mathbf{x}_k$ and $\mathbf{y}_k$, $0 \le k < n$, that contain the values of ϕ_k and $\tilde{\phi}_k$, respectively, at the nodes. The columns of Q, which are given by $D\mathbf{x}_k$ $(0 \le k < n)$, can be stored if desired; however, if we only desire the vector $\mathbf{c}' = Q^*D\mathbf{g}$, the following algorithm can be performed, in which only the current $\mathbf{x}_k$ and $\mathbf{y}_k$ vectors are stored. In the following, $\mathbf{e} = [1, 1, \ldots, 1]^T$ denotes the m-vector containing just ones, D is the matrix defined by (1.7), and $\Lambda := \mathrm{diag}[z_1, z_2, \ldots, z_m]$.

Algorithm 2.1. Stieltjes procedure for Szegő polynomials.

Input: m, n $(n \le m)$, $\{w_k\}_{k=1}^m$, $\{z_k\}_{k=1}^m$, and $\mathbf{g} = [g_k]_{k=1}^m \in \mathbb{C}^m$;

Output: vector $\mathbf{c}' = [c_j']_{j=0}^{n-1} := Q^*D\mathbf{g}$, where Q is defined by (1.10), and parameters $\{\gamma_j\}_{j=1}^{n-1}$ and $\{\sigma_j\}_{j=0}^{n-1}$;

$$\sigma_0 := \alpha := \left(\sum_{k=1}^{m} w_k^2\right)^{1/2};$$
$$\mathbf{x} := \mathbf{y} := \sigma_0^{-1}\mathbf{e}; \qquad c_0' := \mathbf{x}^* D^2 \mathbf{g};$$
for $j = 1, 2, \ldots, n-1$ **do**
$$\gamma_j := -\alpha^{-1}\mathbf{e}^T \Lambda D^2 \mathbf{x};$$
$$\sigma_j := [(1 - |\gamma_j|)(1 + |\gamma_j|)]^{1/2};$$
$$\alpha := \alpha\sigma_j;$$
$$\mathbf{t} := \sigma_j^{-1}(\Lambda\mathbf{x} + \gamma_j \mathbf{y});$$
$$\mathbf{y} := \sigma_j^{-1}(\bar{\gamma}_j \Lambda\mathbf{x} + \mathbf{y});$$
$$\mathbf{x} := \mathbf{t};$$
$$c_j' := \mathbf{x}^* D^2 \mathbf{g};$$

Numerical examples in §5 illustrate that for many distributions of nodes the Stieltjes procedure is very sensitive to roundoff errors. This parallels the behavior of the Stieltjes procedure for the generation of polynomials orthogonal on a point set in a real interval [15, 11, 7].

The classical Stieltjes procedure for generating three-term recurrence coefficients is a special case of the Lanczos process, in which a diagonal matrix is transformed to tridiagonal form by an orthogonal similarity transformation. Similarly, the Stieltjes procedure for Szegő polynomials can be regarded as an application of the Arnoldi process to transform a unitary diagonal matrix by unitary similarity to a Hessenberg matrix. More precisely, the Stieltjes procedure implicitly generates the $n \times n$ leading principal submatrix of an $m \times m$ unitary Hessenberg matrix. The following lemma is a consequence of the "Implicit Q Theorem" [9, Chapter 7.4.5]. Throughout this paper, $\mathbf{e}_1 = [1, 0, \ldots, 0]^T$ denotes the first axis vector of appropriate dimension.

Lemma 2.1. *Given distinct nodes z_k on the unit circle and associated positive weights w_k^2, $k = 1, \ldots, m$, there is a unique $m \times m$ upper Hessenberg matrix H with positive subdiagonal elements and a unique unitary matrix U satisfying*

$$\begin{aligned} &U\mathbf{e}_1 = \mathbf{q}_0 = \sigma_0^{-1}[w_1, w_2, \ldots, w_m]^T, \\ &U^*\Lambda U = H, \\ &\Lambda = \operatorname{diag}[z_1, z_2, \ldots, z_m], \end{aligned} \tag{2.1}$$

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

Observe that the matrix DA in (1.6) is the first n columns of the Krylov matrix $[\mathbf{q}_0, \Lambda\mathbf{q}_0, \ldots, \Lambda^{m-1}\mathbf{q}_0]$. Consequently, the matrix Q in (1.10) is the first n columns of the unitary matrix U defined in (2.1). We therefore propose another method for computing $\mathbf{c}' = Q^* D\mathbf{g}$. This method generates the Schur parameters γ_j, σ_j, and implicitly the Szegő polynomials $\phi_j(z)$ and the

matrix Q, using the algorithm for the inverse eigenvalue problem for unitary Hessenberg matrices presented in [1].

3. An inverse eigenvalue problem

We outline an algorithm for the construction of the matrices H and U from Λ and $\mathbf{q}_0 = U\mathbf{e}_1$. This algorithm will allow us to obtain $Q^*D\mathbf{g}$ and $R^{-1}Q^*D\mathbf{g}$ without storing Q, R, or R^{-1}.

Any $m \times m$ unitary upper Hessenberg matrix H with positive subdiagonal elements can be uniquely expressed as a product of m elementary unitary matrices,

$$H = G_1(\gamma_1)G_2(\gamma_2)\cdots G_{m-1}(\gamma_{m-1})G_m'(\gamma_m), \tag{3.1}$$

for certain complex-valued parameters $|\gamma_j| < 1$, $1 \le j < m$, and $|\gamma_m| = 1$. Here, $G_j(\gamma_j)$ denotes the $m \times m$ Givens reflector in the $(j, j+1)$ plane,

$$G_j(\gamma_j) := \begin{bmatrix} I_{j-1} & & & \\ & -\gamma_j & \sigma_j & \\ & \sigma_j & \overline{\gamma}_j & \\ & & & I_{m-j-1} \end{bmatrix},$$

where $\sigma_j = (1 - |\gamma_j|^2)^{1/2}$, and I_k denotes the $k \times k$ identity matrix. Since $|\gamma_m| = 1$, the matrix $G_m'(\gamma_m) := \operatorname{diag}[1, 1, \ldots, 1, -\gamma_m]$ is also unitary. It can be verified that if the parameters γ_j in (3.1) are identical with the Schur parameters in (1.12)–(1.13), then the characteristic polynomial of the leading principal $j \times j$ submatrix of H is the monic Szegő polynomial $\sigma_0\sigma_1\cdots\sigma_j\phi_j(z)$, where $\phi_j(z)$ is defined by (1.12)–(1.13) (see [1] for details). We therefore call the representation (3.1) the *Schur parametric form* of H.

We solve the inverse eigenvalue problem (2.1) by transforming Λ to a unitary upper Hessenberg matrix H in Schur parametric form by applying a judiciously chosen sequence of Givens reflectors. These reflectors are chosen so that intermediate matrices generated during the algorithm also can be represented in Schur parametric form, and the algorithm actually manipulates Schur parameters and associated complementary parameters instead of matrix elements. We refer to this algorithm, which is described in [1], as the *inverse unitary QR (IUQR) algorithm* because of its relationship with the unitary QR algorithm presented in [10]. The IUQR algorithm requires only $O(m^2)$ arithmetic operations to solve the inverse eigenvalue problem, i.e., to compute the set of parameters $\{\gamma_j, \sigma_j\}_{j=1}^m$ that defines H in (2.1). A mathematically equivalent algorithm that manipulates matrix elements would require $O(m^3)$ arithmetic operations.

The required upper Hessenberg matrix H is obtained by performing a sequence of elementary unitary similarity transformations whose composition re-

sults in the $m \times m$ unitary matrix U such that

$$\begin{bmatrix} 1 & \mathbf{0}^* \\ \mathbf{0} & U^* \end{bmatrix} \begin{bmatrix} \delta & \mathbf{u}^* \\ \mathbf{u} & \Lambda \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^* \\ \mathbf{0} & U \end{bmatrix} = \begin{bmatrix} \delta & \sigma_0 \mathbf{e}_1^* \\ \sigma_0 \mathbf{e}_1 & H \end{bmatrix} \tag{3.2}$$

is an upper Hessenberg matrix with positive subdiagonal elements. (The number δ is arbitrary and remains unchanged during the execution of the algorithm.) Then $H = H(\gamma_1, \ldots, \gamma_{m-1}, \gamma_m)$ has the desired eigenvalues and associated eigenvectors.

The IUQR algorithm builds the Hessenberg matrix by incorporating node-weight pairs one at a time. For each $j = 1, 2, \ldots, m$, let

$$H_j := H(\gamma_1^{(j)}, \ldots, \gamma_{j-1}^{(j)}, \gamma_j^{(j)}) \tag{3.3}$$

denote the unitary upper Hessenberg matrix corresponding with the first j node-weight pairs $\{(z_k, w_k^2)\}_{k=1}^j$, and let $\sigma_0^{(j)} := (\sum_{k=1}^j w_k^2)^{1/2}$. (The eigenvalues of H_j are $\{z_k\}_{k=1}^j$ and the first components of its normalized eigenvectors are $\{(w_k/\sigma_0)\}_{k=1}^j$.) If $j < m$, the matrix H_{j+1} is obtained by performing a sequence of elementary unitary similarity transformations to transform

$$\begin{bmatrix} \delta & \sigma_0^{(j)} & 0 \cdots 0 & w_{j+1} & w_{j+2} \cdots w_m \\ \sigma_0^{(j)} & & & & \\ 0 & & & & \\ \vdots & & H_j & & 0 \\ 0 & & & & \\ w_{j+1} & & & z_{j+1} & \\ w_{j+2} & & & & z_{j+2} \\ \vdots & & 0 & & \ddots \\ w_m & & & & & z_m \end{bmatrix} \tag{3.4}$$

to the matrix

$$\begin{bmatrix} \delta & \sigma_0^{(j+1)} & 0 \cdots 0 & w_{j+2} \cdots w_m \\ \sigma_0^{(j+1)} & & & \\ 0 & & & \\ \vdots & & H_{j+1} & 0 \\ 0 & & & \\ w_{j+2} & & & z_{j+2} \\ \vdots & & 0 & \ddots \\ w_m & & & & z_m \end{bmatrix}. \tag{3.5}$$

The first step of the transformation from (3.4) to (3.5) is to carry out a similarity transformation to permute the coordinate planes $(2, 3, \ldots, j, j+1)$ to $(j+1, 2, 3, \ldots, j)$. The leading principal submatrix of order $j+2$ of the

matrix so obtained from (3.4) is of the form

$$\widetilde{H}^{(1)} := \begin{bmatrix} \delta & w_{j+1} & \sigma_0^{(j)}\mathbf{e}_1^* \\ w_{j+1} & z_{j+1} & \mathbf{0}^* \\ \sigma_0^{(j)}\mathbf{e}_1 & \mathbf{0} & H_j \end{bmatrix}$$

$$= \begin{bmatrix} \delta & w_{j+1} & \sigma_0^{(j)} & & & & & \\ w_{j+1} & z_{j+1} & & & & & & \\ \sigma_0^{(j)} & & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times & \times \\ & & & \times & \times & \times & \times \\ & & & & \times & \times & \times \\ & & & & & \times & \times \end{bmatrix}.$$

Next we set $\sigma_0^{(j+1)} := ((\sigma_0^{(j)})^2 + w_{j+1}^2)^{1/2}$ and $\alpha_0 := -w_{j+1}/\sigma_0^{(j+1)}$. Then

$$\widetilde{H}^{(2)} = \begin{bmatrix} \delta & w_{j+1} & \sigma_0^{(j)} & & & & & \\ \sigma_0^{(j+1)} & \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times & \times \\ & & & \times & \times & \times & \times \\ & & & & \times & \times & \times \\ & & & & & \times & \times \end{bmatrix} := G_2(\alpha_0)\widetilde{H}^{(1)}$$

is a Hessenberg matrix whose trailing principal $(j+1)\times(j+1)$ submatrix is unitary. On the completion of the similarity transformation of $\widetilde{H}^{(1)}$, we obtain

$$\widetilde{H}^{(2)}G_2^*(\alpha_0) = \begin{bmatrix} \delta & \sigma_0^{(j+1)} & & & & & \\ \sigma_0^{(j+1)} & \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times & \times \\ & \otimes & \times & \times & \times & \times & \times \\ & & & \times & \times & \times & \times \\ & & & & \times & \times & \times \\ & & & & & \times & \times \end{bmatrix}. \tag{3.6}$$

The circled element in (3.6) forms a "bulge" which is to be chased down along the subdiagonal in order to obtain a matrix of Hessenberg form. Define $G_3(\alpha_1)$ so that $\widetilde{H}^{(3)} := G_3(\alpha_1)\widetilde{H}^{(2)}G_2^*(\alpha_0)$ is a Hessenberg matrix. Then $\widetilde{H}^{(3)}G_3^*(\alpha_1)$ has a bulge in position $(5, 3)$, which we annihilate by multiplying from the left with $G_4(\alpha_2)$. Proceeding in this manner, we obtain the upper Hessenberg matrix $\widetilde{H}^{(j+1)}G_{j+1}^*(\alpha_{j-1})$, which is unitarily similar to $\widetilde{H}^{(1)}$. The trailing principal $(j+1)\times(j+1)$ submatrix of $\widetilde{H}^{(j+1)}G_{j+1}^*(\alpha_{j-1})$ is unitarily similar to a unitary upper Hessenberg matrix with positive subdiagonal elements. The latter matrix is the desired Hessenberg matrix $H_{j+1} = H(\gamma_1^{(j+1)}, \gamma_2^{(j+1)}, \ldots, \gamma_{j+1}^{(j+1)})$.

This procedure for adding a node-weight pair to H_j, if implemented by directly manipulating the elements of the matrices $\widetilde{H}^{(k)}$, for $1 \le k < j$, would

require $O(j^2)$ arithmetic operations. However, note that for each k the trailing $(j+1)\times(j+1)$ principal submatrix of $\widetilde{H}^{(k)}$ is unitary and of Hessenberg form; consequently, it is unitarily similar to a unitary Hessenberg matrix with positive subdiagonal elements, which we denote by $\widehat{H}_{j+1}^{(k)}$. Hence, we can implicitly carry out the similarity transformations by manipulating the Schur parameters of the intermediate matrices. In particular, one can show that

$$\widehat{H}_{j+1}^{(k)} = H(\gamma_1^{(j+1)}, \gamma_2^{(j+1)}, \ldots, \gamma_{k-2}^{(j+1)}, \overline{z}_{j+1}^{k-2}\alpha_{k-2}, \gamma_{k-1}^{(j)}, \ldots, \gamma_j^{(j)})$$

for $k = 2, 3, \ldots, j+1$. We can therefore add a node-weight pair to H_j using only $O(j)$ arithmetic operations.

Thus, the inverse unitary QR algorithm can be used to construct the unitary $m \times m$ Hessenberg matrix $H = H(\gamma_1, \ldots, \gamma_{m-1}, \gamma_m)$ from its eigenvalues and the first components of its normalized eigenvectors in $O(m^2)$ arithmetic operations. If the matrix U in (2.1) is desired, it can be accumulated as the product of the elementary unitary transformation performed during the IUQR algorithm. Moreover, for any vector $\mathbf{x} \in \mathbb{C}^m$, the vector $U^*\mathbf{x}$ can be obtained by performing row operations on $\mathbf{x}$ during the algorithm. See [1] for more details on the IUQR algorithm.

Recall that the matrix Q in (1.10) consists of the first n columns of the unitary matrix U. These columns correspond to the first $n-1$ Schur parameters of $H = H_m$. Choosing $\mathbf{x} = D\mathbf{g}$ shows that we can compute the first $n-1$ Schur parameters and $Q^*D\mathbf{g}$ without solving the inverse eigenvalue problem in its entirety. In effect, we *curtail* the IUQR algorithm so that only the first $n-1$ Schur parameters of H_j are computed for each $j \le m$. This adaptation of the IUQR scheme to the solution of the discrete least squares problem is given in the following algorithm, which requires $O(mn)$ arithmetic operations.

Algorithm 3.1. Computation of $Q^*D\mathbf{g}$, Schur parameters γ_j and associated complementary parameters σ_j.

Input: m, n $(n \le m)$, distinct nodes $\{z_k\}_{k=1}^m$ on $|z| = 1$, associated positive weights $\{w_k\}_{k=1}^m$, and vector $\mathbf{g} \in \mathbb{C}^m$;

Output: vector $\mathbf{c}' = [c_j']_{j=0}^{n-1} := Q^*D\mathbf{g}$ and parameters $\{\gamma_j\}_{j=1}^{n-1}$, $\{\sigma_j\}_{j=0}^{n-1}$. The vector $\mathbf{c}'$ below is assumed to be of length $n+1$; its last component, c_n', is used for temporary storage.

$\sigma_0 := w_1$; $\gamma_1 := -z_1$; $\sigma_1 := 0$; $c_0' := w_1 g_1$; $c_1' := 0$;
for $j = 1, 2, \ldots, m-1$ **do**
  $j' := \min(j, n-1)$;
  for $k = j'+1, j', \ldots, 1$ **do** $c_k' := c_{k-1}'$;
  $c_0' := w_{j+1} g_{j+1}$; $\beta_0 := \sigma_0$; $\sigma_0 := (\sigma_0^2 + w_{j+1}^2)^{1/2}$;
  $\beta_0 := \beta_0/\sigma_0$; $\alpha_0 := -w_{j+1}/\sigma_0$;

$$\begin{bmatrix} c_0' \\ c_1' \end{bmatrix} := \begin{bmatrix} -\overline{\alpha}_0 & \beta_0 \\ \beta_0 & \alpha_0 \end{bmatrix} \begin{bmatrix} c_0' \\ c_1' \end{bmatrix};$$

if $j+1 < n$ **then**
$\gamma_{j+1} := -\gamma_j z_{j+1}$; $\sigma_{j+1} := 0$;
for $k = 1, 2, \ldots, j'$ **do**
$\tau := \alpha_{k-1} + \gamma_k z_{j+1}^{k-2} \overline{\alpha}_{k-1}$; $\rho := \beta_{k-1}(\sigma_k^2 + |\tau|^2)^{1/2}$;
$\alpha_k := \tau \beta_{k-1} z_{j+1}/\rho$; $\beta_k := \beta_{k-1}\sigma_k/\rho$;
$\gamma_k := \beta_{k-1}^2 \gamma_k - \overline{z}_{j+1}^{k-2} \alpha_{k-1}^2$; $\sigma_k := \rho$;
$\begin{bmatrix} c_k' \\ c_{k+1}' \end{bmatrix} := \begin{bmatrix} -\overline{\alpha}_k & \beta_k \\ \beta_k & \alpha_k \end{bmatrix} \begin{bmatrix} c_k' \\ c_{k+1}' \end{bmatrix}$;

4. Computation of the optimal polynomial

After computing the vector $\mathbf{c}' = Q^* D\mathbf{g}$ and the parameters $\{\gamma_j\}_{j=1}^{n-1}$ and $\{\sigma_j\}_{j=0}^{n-1}$ using either Algorithm 3.1 or Algorithm 2.1, we can obtain the least squares solution $\hat{\mathbf{c}} = R^{-1}\mathbf{c}'$ of (1.6) in $O(n^2)$ arithmetic operations using the Szegő recursions in the following manner. The vector $\mathbf{r}_j \in \mathbb{C}^j$ containing the first j components of the jth column of R^{-1} contains the coefficients of the orthonormal Szegő polynomial ϕ_{j-1} in the power basis. (This follows from the fact that the columns of R contain the coefficients of the power basis in terms of the basis of orthonormal Szegő polynomials.) We can therefore recursively generate the vectors $\mathbf{r}_j \in \mathbb{C}^j$ by writing the Szegő recursions in terms of these coefficients. This leads to the algorithm below. In the following, $\tilde{\mathbf{r}}_j := J_j \overline{\mathbf{r}}_j$ denotes the conjugate reversal of the vector $\mathbf{r}_j$.

Algorithm 4.1. Compute $R^{-1}\mathbf{b}$ for an arbitrary vector $\mathbf{b} \in \mathbb{C}^n$.
Input: parameters $\{\gamma_j\}_{j=1}^{n-1}$, $\{\sigma_j\}_{j=0}^{n-1}$ and vector $\mathbf{b} = [b_j]_{j=1}^n$;
Output: vector $\mathbf{a} = [a_j]_{j=1}^n := R^{-1}\mathbf{b}$;
$\mathbf{a} := \mathbf{0}$; $\mathbf{r}_1 := 1/\sigma_0$; $a_1 := \mathbf{r}_1 b_1$;
for $j = 1, \ldots, n-1$ **do**
$\mathbf{r}_{j+1} := \sigma_j^{-1} \left(\begin{bmatrix} 0 \\ \mathbf{r}_j \end{bmatrix} + \gamma_j \begin{bmatrix} \tilde{\mathbf{r}}_j \\ 0 \end{bmatrix} \right)$;
$\mathbf{a} := \mathbf{a} + b_{j+1} \begin{bmatrix} r_{j+1} \\ \mathbf{0}_{n-j-1} \end{bmatrix}$;

Hence, Algorithm 4.1 allows us to obtain the solution $\hat{\mathbf{c}}$ of (1.6) from $\mathbf{c}' = Q^* D\mathbf{g}$, the Schur parameters γ_j, and the associated complementary parameters σ_j. The vector $\hat{\mathbf{c}}$ contains the coefficients of the optimal algebraic polynomial p in power form (1.4). By Proposition 1.1, the coefficients of the optimal trigonometric polynomial t in the form (1.1) can be recovered from $\hat{\mathbf{c}}$ if $\mathbf{g} = \Lambda^l \mathbf{f}$, where $\mathbf{f} \in \mathbb{R}^m$. These representations of p and t are convenient if we desire to integrate or differentiate these polynomials, or if we wish to evaluate them at many equidistant points on a circle with center at the origin. The latter can be carried out efficiently by the fast Fourier transform method [13, Chapter 13].

If, on the other hand, we only desire to evaluate p or t at a few points, then we can use the representation of p in terms of orthonormal Szegő polynomials. The vector $\mathbf{c}' = Q^* D\mathbf{g}$ contains the coefficients of $p(z)$ when expressed in this basis. The following algorithm uses the Szegő recursions to evaluate $p(z)$ using $\mathbf{c}' = Q^* D\mathbf{g}$. The quantities q and r in the algorithm contain the values of the polynomials ϕ_j and $\tilde{\phi}_j$ at z.

Algorithm 4.2. Evaluation of p at z.

Input: vector $\mathbf{c}' = [c_j']_{j=0}^{n-1}$, parameters $\{\gamma_j\}_{j=1}^{n-1}$ and $\{\sigma_j\}_{j=0}^{n-1}$, and $z \in \mathbb{C}$;

Output: $p = p(z) := \sum_{j=0}^{n-1} c_j' \phi_j(z)$, where ϕ_j is the orthonormal Szegő polynomial of degree j defined by (1.12)–(1.13);

$q := r := 1/\sigma_0$; $p := c_0' q$;

for $j = 1, 2, \ldots, n-1$ **do**

$$\begin{bmatrix} q \\ r \end{bmatrix} := \sigma_j^{-1} \begin{bmatrix} z & \gamma_j \\ z\overline{\gamma}_j & 1 \end{bmatrix} \begin{bmatrix} q \\ r \end{bmatrix};$$

$p := p + c_j' q$;

5. Computed examples

We present some numerical examples that compare accuracy and speed of the Stieltjes procedure, the IUQR algorithm, and the QR decomposition method for general matrices as implemented in LINPACK [5] for the solution of the least squares problem (1.6). The examples were executed on a VAXstation 2000 using single-precision and double-precision arithmetic (approximately 7 and 16 significant decimal digits, respectively).

For any vector $\mathbf{c} = [c_j]_{j=0}^{m-1} \in \mathbb{C}^m$, define the vector $|\mathbf{c}| = [|c_j|]_{j=0}^{m-1}$. The subroutines CQRDC and CQRSL of LINPACK are implementations of the QR decomposition method for the solution of a general overdetermined linear system of equations. When applied to the system (1.6), CQRDC produces a vector $\mathbf{c}_s'$ such that, in exact arithmetic, $|\mathbf{c}_s'| = |\mathbf{c}'|$. The moduli are equal because the matrix with orthonormal columns that is implicitly generated by the subroutine CQRDC is equal to Q up to a unimodular scaling of its columns. The subroutine CQRSL then computes the (mathematically unique) least squares solution $\hat{\mathbf{c}}$.

In the computed examples, we input m distinct arguments $\theta_j \in [0, 2\pi)$ and corresponding positive weights as described below. We then solve the least squares problem (1.16), where the elements of the real vector $\mathbf{f}$ are randomly generated uniformly distributed numbers in $[-5, 5]$. The labels on the following graphs refer to the following three procedures, which were all performed in single-precision arithmetic.

IUQR: Algorithm 3.1 is used to compute $\mathbf{c}'$. Then $\hat{\mathbf{c}}$ is calculated using Algorithm 4.1.

Stieltjes: Algorithm 2.1 is used to compute $\mathbf{c}'$. Then $\hat{\mathbf{c}}$ is calculated using Algorithm 4.1.

LINPACK: The matrix DA is explicitly formed, and the subroutine CQRDC is used to compute $\mathbf{c}_s'$. The subroutine CQRSL is then used to calculate $\hat{\mathbf{c}}$.

For comparison of accuracy we also solve the system (1.16) in double-precision arithmetic using the QR decomposition method for general matrices as implemented in the double precision LINPACK subroutines ZQRDC and ZQRSL. Let $\mathbf{c}_d'$ denote the coefficient vector determined by ZQRDC and $\hat{\mathbf{c}}_d$ denote the vector obtained by ZQRSL.

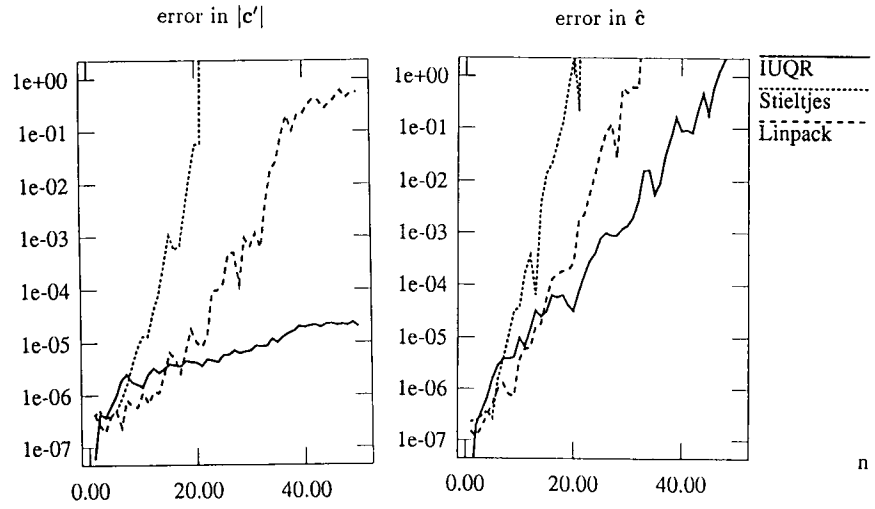

FIGURE 1

$m = 50$, equispaced arguments in $[0, 3\pi/2)$, weights equal to one

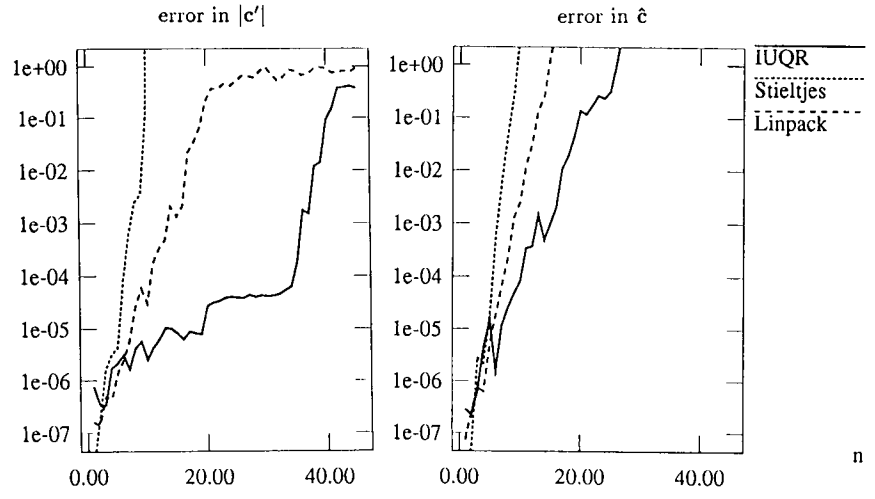

FIGURE 2

$m = 50$, equispaced arguments in $[0, \pi)$, weights equal to one

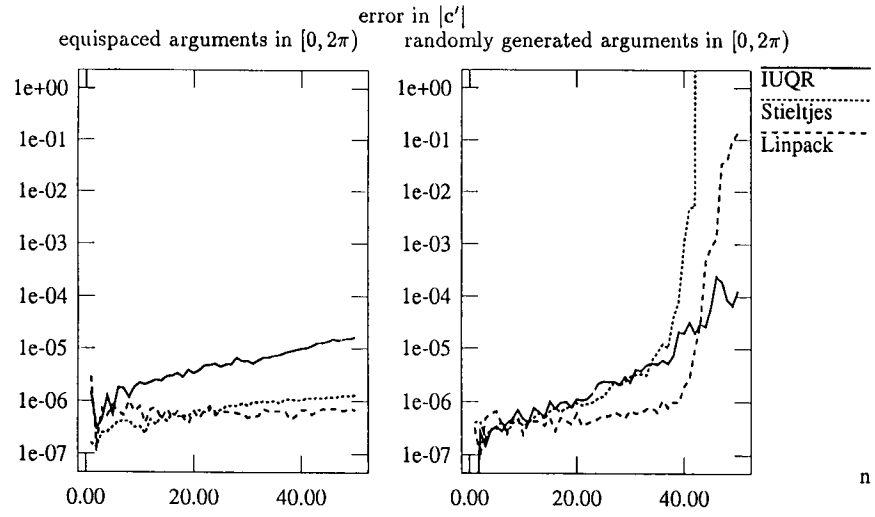

FIGURE 3
$m = 50$, weights equal to one

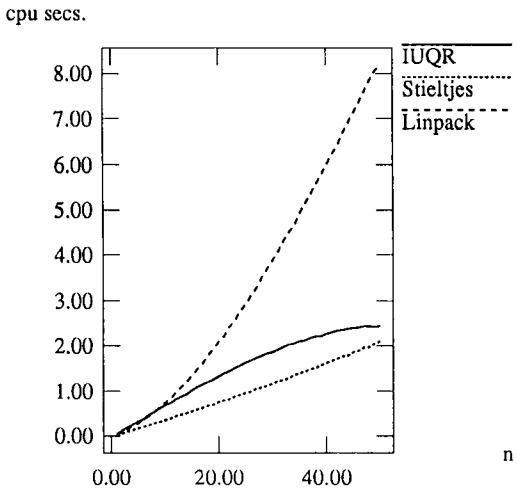

FIGURE 4
Average CPU time; $m = 50$

Let $\|\mathbf{c}\|_2 := (\sum_{j=0}^{n-1} |c_j|^2)^{1/2}$ denote the 2-norm of a vector $\mathbf{c} \in \mathbb{C}^n$. The first four figures display the relative error $\| |\mathbf{c}'| - |\mathbf{c}'_d| \|_2 / \|\mathbf{c}'_d\|_2$, denoted "error in $|\mathbf{c}'|$," and the relative error $\|\hat{\mathbf{c}} - \hat{\mathbf{c}}_d\|_2 / \|\hat{\mathbf{c}}_d\|_2$, denoted "error in $\hat{\mathbf{c}}$." Each graph displays the errors for $m = 50$ and increasing values of n.

In our first two examples, the arguments of the nodes are *equispaced* in the interval $[a, b)$ (i.e., $\theta_j := a + (j-1)(b-a)/m$, $j = 1, \ldots, m$) and the weights are all equal to one. Figures 1–2 display errors in the coefficient vectors for equispaced nodes in intervals smaller than 2π. In these examples the errors in the coefficient vectors computed by the IUQR algorithm are sometimes slightly larger than those determined by the Stieltjes procedure and the

LINPACK subroutines for small values of n. However, as n increases, and the problem becomes more ill-conditioned, the Stieltjes procedure is the first to produce inaccurate results. The LINPACK routines produce inaccurate results for somewhat larger values of n than the Stieltjes procedure, while the error in the coefficient vector $\mathbf{c}'$ computed by the IUQR algorithm usually becomes substantial only when n is very close to m. In this sense the IUQR algorithm displays the most robust behavior among these three algorithms. The error in the coefficient vector $\hat{\mathbf{c}}$ is generally larger than in the vector $\mathbf{c}'$ because $\hat{\mathbf{c}} = R^{-1}\mathbf{c}'$, and the matrix R can be quite ill-conditioned. We obtained results similar to those in Figures 1 and 2 with other choices for the nodes and weights. We found that the choice of weights w_j^2 and of vector $\mathbf{g}$ in (1.6) has little effect on the accuracy of these algorithms.

Figure 3 displays the computed errors in $\mathbf{c}'$ when the arguments are equispaced in $[0, 2\pi)$ and when the arguments are randomly generated uniformly distributed numbers in $[0, 2\pi)$. All weights in both of these examples are equal to one. In the first case we are computing the discrete Fourier transform, and the matrix DA has orthonormal columns. Both the Stieltjes procedure and the LINPACK subroutines produce somewhat smaller errors than the IUQR algorithm in this example, but the rate of growth of the errors is roughly the same for all three methods compared. Note that the fast Fourier transform method (FFT) [13] is a better method for solving this example. When the arguments are randomly generated uniformly distributed points in $[0, 2\pi)$, the least squares problem is relatively well-conditioned, and the IUQR algorithm and the Stieltjes procedure yield roughly the same accuracy until n gets close to m.

Figure 4 displays the average CPU time (in seconds) used by the three algorithms in the computation of $\hat{\mathbf{c}}$ for $1 \leq n \leq m = 50$. Thus, our experiments show that the IUQR algorithm is faster than the QR decomposition method for general matrices of LINPACK already for fairly small values of m and n.

6. Conclusion

A new method for least squares approximation by trigonometric polynomials is presented that is based on the solution of an inverse eigenvalue problem for a unitary Hessenberg matrix. This method (Algorithms 3.1 and 4.1) is never much less accurate but often much more accurate than competing schemes. Timings show the new method to be faster than schemes based on the QR decomposition of a general matrix already for fairly small problems. Unless m is much larger than n, or the nodes θ_j are nearly uniformly distributed in the intervals $[0, 2\pi)$, the new algorithm is more accurate than the Stieltjes procedure.

Acknowledgment

Work of L. R. was carried out during a visit at the Computer Science Department, Stanford University. He would like to thank Gene Golub

and Bernd Fischer for their hospitality. G. A. would like to thank the Department of Mathematics at the University of Kentucky for their hospitality during his visit for the 1989–1990 academic year.

BIBLIOGRAPHY

1. G. S. Ammar, W. B. Gragg, and L. Reichel, *Constructing a unitary matrix from spectral data*, Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms (G. H. Golub and P. Van Dooren, eds.), Springer-Verlag, New York, 1990, pp. 385–396.
2. J.-P. Berrut, *Baryzentrische Formeln zur trigonometrischen Interpolation*. I, II, J. Appl. Math. Phys. (ZAMP) **35** (1984), 91–105, 193–205.
3. Å. Björck and V. Pereyra, *Solution of Vandermonde systems of equations*, Math. Comp. **24** (1970), 893–903.
4. C. J. Demeure, *Fast QR factorization of Vandermonde matrices*, Linear Algebra Appl. **122–124** (1989), 165–194.
5. J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart, *LINPACK users' guide*, SIAM, Philadelphia, PA, 1979.
6. G. E. Forsythe, *Generation and use of orthogonal polynomials for data-fitting with a digital computer*, J. Soc. Indust. Appl. Math. **5** (1957), 74–88.
7. W. Gautschi, *On generating orthogonal polynomials*, SIAM J. Sci. Statist. Comput. **3** (1982), 289–317.
8. ——, *Orthogonal polynomials—constructive theory and applications*, J. Comput. Appl. Math. **12 & 13** (1985), 61–76.
9. G. H. Golub and C. F. Van Loan, *Matrix computations*, 2nd ed., Johns Hopkins Univ. Press, 1989.
10. W. B. Gragg, *The QR algorithm for unitary Hessenberg matrices*, J. Comput. Appl. Math. **16** (1986), 1–8.
11. W. B. Gragg and W. J. Harrod, *The numerically stable reconstruction of Jacobi matrices from spectral data*, Numer. Math. **44** (1984), 317–335.
12. U. Grenander and G. Szegő, *Toeplitz forms and their applications*, Chelsea, New York, 1984.
13. P. Henrici, *Applied and computational complex analysis*, vol. 3, Wiley, New York, 1986.
14. A. C. R. Newbery, *Trigonometric interpolation and curve-fitting*, Math. Comp. **24** (1970), 869–876.
15. L. Reichel, *Fast QR decomposition of Vandermonde-like matrices and polynomial least squares approximation*, Numerical Analysis Report 89-5, Department of Mathematics, M.I.T., 1989.

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF KENTUCKY, LEXINGTON, KENTUCKY 40506

DEPARTMENT OF MATHEMATICAL SCIENCES, NORTHERN ILLINOIS UNIVERSITY, DEKALB, ILLINOIS 60115

DEPARTMENT OF MATHEMATICS, NAVAL POSTGRADUATE SCHOOL, MONTEREY, CALIFORNIA 93943