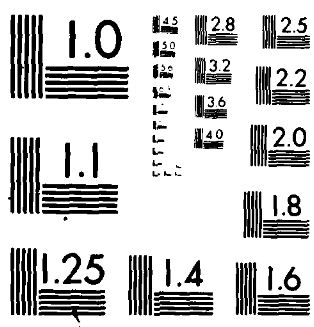
AD-A098 309 NAVAL UNDERWATER SYSTEMS CENTER NEW LONDON CT NEW LO--ETC F/G 12/1
LINEAR CHEBYSHEV COMPLEX FUNCTION APPROXIMATION.(U)
FEB 81 R L STREIT, A H NUTTALL
UNCLASSIFIED NUSC-TR-6403 NL

END
DATE
FILMED
5-81
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

NUSC Technical Report 6403
26 February 1981

Linear Chebyshev Complex Function Approximation

Roy L. Streit
Information Services Department

Albert H. Nuttall
Surface Ship Sonar Department

Naval Underwater Systems Center
Newport, Rhode Island / New London, Connecticut

Distribution unlimited; approved for public release.

Preface

The research presented in this report was conducted under NUSC Project No. A75205, Subproject No. ZR0000101, "Applications of Statistical Communications Theory to Acoustic Signal Processing," Principal Investigator, Dr. A. H. Nuttall (Code 3302), and under NUSC Project No. A70210, Subproject No. ZR000010, "Optimization of Mutually Coupled Arrays," Principal Investigator, Dr. R. L. Streit (Code 7122). The Program Manager is CAPT D. F. Parrish (MAT 08L).

The Technical Reviewer of this report was B. G. Buehler (Code 3292).

Reviewed and Approved: 26 February 1981

W. A. VonWinkle

W. A. VonWinkle
Associate Technical Director
for Technology

The authors of this report is located at the
New London Laboratory, Naval Underwater Systems
Center, New London, Connecticut 06320.

(14) NUSC-TR-6403

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: TR 6403
2. GOVT ACCESSION NO.: AD-A098309
3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): (6) LINEAR CHEBYSHEV COMPLEX FUNCTION APPROXIMATION.
5. TYPE OF REPORT & PERIOD COVERED:
6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): (10) Roy L./Streit, Albert H./Nuttall
8. CONTRACT OR GRANT NUMBER(s):

(9) Technical rept.

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Naval Underwater Systems Center
New London Laboratory
New London, CT 06320

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
A75205
A70210

11. CONTROLLING OFFICE NAME AND ADDRESS:
Chief of Naval Material
Washington, D. C. 20360

(11)

12. REPORT DATE: 26 Feb[redacted]81
13. NUMBER OF PAGES: 45

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
(12) 50

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Distribution unlimited; approved for public release.

(16) ZR00001 (17) ZR0000101

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Chebyshev Approximation	Array Application
Complex Functions	Linear Programming
Complex Basis	Minimax Approximation
Linear Approximation	Optimum Antenna Array
Magnitude of Complex Error	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A new computational technique is described for the Chebyshev, or minimax, approximation of a given complex valued function by means of linear combinations of given complex valued basis functions. The domain of definition of all functions can be any finite set whatever. Neither the basis functions nor the function approximated need satisfy any special hypotheses beyond the requirement that they be defined on a common domain. Theoretical upper and lower bounds on the accuracy of the computed Chebyshev error are derived. These bounds permit

405928

20. (Continued):

both *a priori* and *a posteriori* error assessments. Efforts to extend the method to functions whose domain of definition is a continuum are discussed. Numerical examples and a FORTRAN program listing are included.

An application is presented involving pre-shading a 50-element antenna array to minimize the effects of a 10% element failure rate, while maintaining full steering capability and mainlobe beamwidth.

B

Table of Contents

List of Figures

List of Tables

A

Linear Chebyshev Complex Function Approximation

I. Introduction

The approximation of desired or given functional behavior by finite sets of simpler or specified basis functions is a recurrent problem in many fields. For example, in the mathematical field, we might wish to approximate a (desired) complex integral by a set of (simpler) sinusoidal components. Or in an antenna array processing application, we often want to realize a (given) low sidelobe behavior by means of an array with (specified) element locations which are not under our control.

For the case where the given functional behavior and the specified basis functions are all real valued and defined on a finite discrete data set, and where the approximation is afforded by a real-weighted linear combination of these basis functions, the optimum solution for minimizing the maximum magnitude error, i.e., the Chebyshev norm, is in very good shape due to a fine algorithm given in [1]. Specifically, this algorithm solves the following mathematical problem: given real constants $\{f_i\}$, $\{h_{ik}\}$, where $1 \leq i \leq m$, $1 \leq k \leq n$, $m \geq n$, the real quantities $\{a_k\}_1^n$ are determined that minimize the maximum absolute value of the error residuals

$$e_i \equiv f_i - \sum_{k=1}^{n} a_k h_{ik} \quad \text{for } 1 \leq i \leq m. \tag{1.1}$$

This algorithm has recently been used to good advantage in an array processing application to design some real symmetric weighting functions with very good sidelobe behavior, subject to constraints on the rate of decay of the distant sidelobes [2].

Here we wish to employ the algorithm, as described above for real variables in (1.1), for the minimization of the Chebyshev norm of

$$e_n(z) \equiv f(z) - \sum_{k=1}^{n} a_k h_k(z) \tag{1.2}$$

when $f(z)$ and $\{h_k(z)\}_1^n$ are complex, and z can take values in an arbitrary finite discrete point set. The weighting coefficients $\{a_k\}_1^n$, may be complex, or alternatively, they may be restricted to be real. Applications are afforded by an antenna array with arbitrarily specified element locations, but employing weights that are restricted to be real, or alternatively by array weights that are also allowed to be phased (complex). In Section II, the basic mathematical theory and algorithm for the minimization of (1.2) is developed. Numerical examples and applications of the technique, some efforts attempted for extending the method to a continuum of values of z, and a discussion constitute the rest of the main body of the report. An Appendix presents a computer program in a form which should be useful to readers interested in applying the technique to their own particular applications.

Although the above algorithm [1] is limited to a discrete set of points, it has been used fruitfully to minimize the continuous error (1.2) over a real variable z in the interval $[z_a, z_b]$, when f and $\{h_k\}$ are real, in the following manner. First, an initial set of $m \geq n$ real points $\{z_i^{(1)}\}_1^m$ was specified and the Chebyshev norm minimized in the usual fashion, resulting in the coefficient set $\{a_k^{(1)}\}_1^n$. For this set of optimum coefficients, the locations $\{z_i^{(2)}\}_1^\ell$ of the largest peaks of $|e_n(z)|$ were located, by setting the derivative $e_n'(z)$ to zero and solving numerically for $\{z_i^{(2)}\}_1^\ell$; the number ℓ of such peaks will generally be less than m, but larger than n. (This approach presumes the availability of computable expressions for $f'(z)$ and $\{h_k'(z)\}_1^n$). Then the modified set of points $\{z_i^{(2)}\}_1^\ell$ were used for another Chebyshev minimization, resulting in coefficient set $\{a_k^{(2)}\}_1^n$. Repetition of this procedure stabilized after a few trials with a unique set of $\{z_i\}_1^\ell$ at which the maximum errors were equal and irreducible. In the examples tried in [2], the number of peaks, ℓ, at which the magnitude error $|e_n(z)|$ was largest and equal, turned out to be $n+1$. Further discussion of this recursive approach is given in Section V.

II. Mathematical Theory and Algorithm

Let f and $h_1, \ldots, h_n$ be complex valued functions defined on the finite discrete point set $Q_m = \{z_1, \ldots, z_m\}$. For a complex vector $a = (a_1, \ldots, a_n) \epsilon C^n$, define the complex error

$$f(z) - \sum_{k=1}^{n} a_k h_k(z) \equiv e_n(z;a), \quad z \epsilon Q_m . \tag{2.1}$$

The discrete linear Chebyshev approximation problem is to find a complex* vector $\tilde{a} = (\tilde{a}_1, \ldots, \tilde{a}_n) \epsilon C^n$ so that

$$E_n(f) \equiv \min_{a \epsilon C^n} \max_{z \epsilon Q_m} |e_n(z;a)| = \max_{z \epsilon Q_m} | e_n(z;\tilde{a}) | . \tag{2.2}$$

The quantity $E_n(f)$ is called the discrete Chebyshev, or minimax, error of the approximation on the point set Q_m.

We do not solve this problem exactly. An algorithm presented in [3] for its solution is erroneous; we have discovered examples (see Section IV) such that the recursive procedure described there need not converge to a solution of (2.2). We will show that problem (2.2) can be replaced by a related approximate problem solvable by available linear programming techniques. The exact solution of this related problem yields approximate solutions of (2.2). The error in these approximate solutions to (2.2) can be determined and, in fact, made arbitrarily small, using the results we prove below; see Theorems 1 and 2.

It can be shown by standard mathematical methods [4, p.1] that a vector $\tilde{a}$ satisfying (2.2) exists, although it may not be unique. Sufficient conditions are known that result in unique $\tilde{a}$, but we do not need these conditions here. Therefore, no further assumptions on f, $h_1, \ldots, h_n$ or the point set Q_m are made. In order to proceed,we need the following result.

* The restriction of $\tilde{a}$ to real values is discussed at the end of this section.

Lemma 1. If $z = x + iy$, where x and y are real, then

$$|z| = \max_{-\pi<\theta\leq\pi} (x\cos\theta + y\sin\theta) . \tag{2.3}$$

Proof. If $z=0$, the result is obvious. Suppose, then, that $z\neq 0$. By the Cauchy-Schwartz inequality, for every real θ,

$$x\cos\theta + y\sin\theta \leq (x^2+y^2)^{1/2}(\cos^2\theta + \sin^2\theta)^{1/2} = |z|$$

so that

$$\max_{-\pi<\theta\leq\pi} (x\cos\theta + y\sin\theta) \leq |z| .$$

For the particular value $\theta = \arg(z)$, it is seen that (2.3) holds. This completes the proof.

Now, let the real and imaginary parts of the complex error $e_n(z;a)$ be denoted by $R_n(z;a)$ and $I_n(z;a)$, respectively. Thus, from Lemma 1,

$$|e_n(z;a)| = \max_{-\pi<\theta\leq\pi} (R_n(z;a)\cos\theta + I_n(z;a)\sin\theta) . \tag{2.4}$$

If, in this last equation, we take the maximum over any finite subset T of angles θ in the interval $-\pi<\theta\leq\pi$, instead of all angles in the interval $-\pi<\theta\leq\pi$, we must have

$$|e_n(z;a)| \geq \max_{\theta\in T} (R_n(z;a)\cos\theta + I_n(z;a)\sin\theta) . \tag{2.5}$$

It will be seen shortly that the next result is very important and central to our problem.

Lemma 2. Let $\theta_j = \pi(j-1)/p$, $j = 1,2, \ldots, 2p$, where the integer $p\geq 2$. Let $z = x + iy$, and let

$$M = \max_{j=1,\ldots,2p} (x\cos\theta_j + y\sin\theta_j) . \tag{2.6}$$

Then

$$M \leq |z| \leq M\sec(\pi/2p) . \tag{2.7}$$

Proof. That $|z| \geq M$ is obvious, so we only have to prove $|z| \leq M\sec(\pi/2p)$. Let $P(x,y)$ be the point in the Euclidean plane corresponding to the complex number $z = x + iy \neq 0$, so that

$$x = |z|\cos(\arg z)$$
$$y = |z|\sin(\arg z).$$

Thus, for any angle φ, we must have

$$(|z| \cos(\arg z) - x)\cos\varphi + (|z| \sin(\arg z) - y)\sin\varphi = 0$$

which, after simple algebraic manipulation, can be written

$$|z| = x'(\varphi) \sec \alpha(\varphi) \tag{2.8}$$

where $x'(\varphi) = x \cos\varphi + y \sin\varphi$ and $\alpha(\varphi) = \arg(ze^{-i\varphi})$. Alternatively, (2.8) can be derived geometrically by considering x' to be the x coordinate of the point $P(x,y)$ after a rotation of the axes through the angle φ. From (2.8) we have

$$|z| = x'(\theta_j) \sec \alpha(\theta_j), \; j = 1, \ldots, 2p . \tag{2.9}$$

Let the index k be such that

$$M = x'(\theta_k) = \max_{1 \leq j \leq 2p} x'(\theta_j). \tag{2.10}$$

With the particular angles θ_j chosen here, $x'(\theta_{j+p}) = -x'(\theta_j)$ for $j = 1, \ldots, p$, so that we must have $x'(\theta_k) > 0$. Since $z \neq 0$ is fixed in (2.9), it is clear from (2.10) and the definition of the angles $\alpha(\theta_j)$ that

$$0 < \sec \alpha(\theta_k) = \min_{1 \leq j \leq 2p} |\sec \alpha(\theta_j)| \leq \sec(\pi/2p).$$

Therefore,

$$\begin{aligned} |z| &= x'(\theta_k) \sec \alpha(\theta_k) \\ &= M \sec(\pi/2p). \end{aligned}$$

This concludes the proof.

We are now in a position to describe a problem that we can solve exactly and that is related to the given discrete linear Chebyshev approximation problem (2.2). For notational convenience, we define, for any complex vector $a \varepsilon C^n$,

$$G_j(z;a) = R_n(z;a) \cos\theta_j + I_n(z;a) \sin\theta_j, \; j = 1, \ldots, 2p , \tag{2.11}$$

where $\theta_1, \ldots, \theta_{2p}$ are the angles given explicitly in Lemma 2. We seek a complex vector $\hat{a} = (\hat{a}_1, \ldots, \hat{a}_n) \varepsilon C^n$ satisfying

$$\begin{aligned} M_{np}(f) &\equiv \min_{a \varepsilon C^n} \max_{z \varepsilon Q_m} \max_{j=1,\ldots,2p} G_j(z;a) \\ &= \max_{z \varepsilon Q_m} \max_{j=1,\ldots,2p} G_j(z;\hat{a}) . \end{aligned} \tag{2.12}$$

Using standard mathematical methods, it is easy to see that at least one such vector $\hat{a} \varepsilon C^n$ exists. The connection between the problem (2.12) and the problem (2.2) is explored in the next few results.

Theorem 1. Let $p \geq 2$ be an integer, and let $\theta_j = \pi(j-1)/p$, $j = 1,2, \ldots, 2p$. Then

$$M_{np}(f) \leq E_n(f) \leq M_{np}(f) \sec(\pi/2p) . \qquad (2.13)$$

Proof. Using $\tilde{a}$ and $\hat{a}$ as in (2.2) and (2.12), respectively, we have

$$\begin{aligned}
M_{np}(f) &= \max_{z \varepsilon Q_m} \max_{1 \leq j \leq 2p} G_j(z;\hat{a}) && \text{from (2.12)} \\
&\leq \max_{z \varepsilon Q_m} \max_{1 \leq j \leq 2p} G_j(z;\tilde{a}) && \text{from (2.12)} \\
&\leq \max_{z \varepsilon Q_m} |e_n(z;\tilde{a})| && \text{implied by (2.7)} \\
&= E_n(f) && \text{from (2.2)} \\
&\leq \max_{z \varepsilon Q_m} |e_n(z;\hat{a})| && \text{from (2.2)} \\
&= \max_{z \varepsilon Q_m} \left\{ \max_{1 \leq j \leq 2p} G_j(z;\hat{a}) \right\} \sec(\pi/2p) && \text{implied by (2.7)} \\
&= M_{np}(f) \sec(\pi/2p) .
\end{aligned}$$

This concludes the proof.

Theorem 2. Let $p \geq 2$ be an integer, and let $\theta_j = \pi(j-1)/p$, $j = 1,2, \ldots, 2p$. Let

$$\mathscr{E}_{np}(f) = \max_{z \varepsilon Q_m} |e_n(z;\hat{a})| \qquad (2.14)$$

where the complex vector $\hat{a} \varepsilon C^n$ is any vector satisfying (2.12). Then

$$E_n(f) \leq \mathscr{E}_{np}(f) \leq E_n(f) \sec(\pi/2p) . \qquad (2.15)$$

Proof. Using $\tilde{a}$ and $\hat{a}$ as before, we have

$$\begin{aligned}
E_n(f) &\leq \mathscr{E}_{np}(f) && \text{from (2.2)} \\
&= \max_{z \varepsilon Q_m} |e_n(z;\hat{a})| && \text{from (2.14)} \\
&\leq \max_{z \varepsilon Q_m} \left\{ \max_{1 \leq j \leq 2p} G_j(z;\hat{a}) \right\} \sec(\pi/2p) && \text{implied by (2.7)} \\
&\leq \max_{z \varepsilon Q_m} \max_{1 \leq j \leq 2p} G_j(z;\tilde{a}) \sec(\pi/2p) && \text{from (2.12)} \\
&\leq \max_{z \varepsilon Q_m} |e_n(z;\tilde{a})| \sec(\pi/2p) && \text{implied by (2.7)} \\
&= E_n(f) \sec(\pi/2p).
\end{aligned}$$

This concludes the proof.

Corollary 2.1. Under the conditions of Theorem 2,

$$M_{np}(f) \leq E_n(f) \leq \mathscr{E}_{np}(f) . \qquad (2.16)$$

Proof. Immediate.

The preceding corollary evidently gives excellent upper and lower bounds on the discrete linear Chebyshev approximation error $E_n(f)$, and these bounds are readily available after the numerical computation of $\hat{a}\varepsilon \mathbf{C}^n$ and $M_{np}(f)$ has been completed. We point out that the above two theorems substantially generalize results in [3, p.854].

Using the Maclaurin series for sec x in (2.15) gives the relative discrepancy

$$0 \leq \frac{\mathscr{E}_{np}(f) - E_n(f)}{E_n(f)} \leq \sec(\pi/2p) - 1 = \frac{\pi^2}{8p^2} + o\left(\frac{1}{p^4}\right), \; p \to \infty . \qquad (2.17)$$

Note that this upper bound on the relative error is independent of f, the point set Q_m, the basis functions $\{h_k\}$, and n.

We will now explicitly formulate an overdetermined system of real linear equations to be solved in the Chebyshev norm (to be defined) which is equivalent to solving the problem (2.12). Referring to the choice of θ_j's in Lemma 2, we observe that $\theta_{p+j} = \pi + \theta_j$, $j = 1, \ldots, p$, and, and so from (2.11), we have

$$G_{p+j}(z;a) = -G_j(z;a) \; , \; j = 1, \ldots, p \; .$$

Therefore, we may rewrite (2.12) as

$$M_{np}(f) = \min_{a\varepsilon \mathbf{C}^n} \; \max_{\substack{1 \leq t \leq m \\ 1 \leq j \leq p}} |G_j(z_t;a)| \; . \qquad (2.18)$$

Now, breaking the following quantities into their real and imaginary components

$$\begin{aligned} f(z) &= u(z) + iv(z) \\ h_k(z) &= r_k(z) + is_k(z) \; , \; k = 1, \ldots, n, \\ a_k &= b_k + ic_k, \; k = 1, \ldots, n, \end{aligned} \qquad (2.19)$$

we may write

$$\begin{aligned} R_n(z;a) &= u(z) - \sum_{k=1}^{n} b_k r_k(z) + \sum_{k=1}^{n} c_k s_k(z) \\ I_n(z;a) &= v(z) - \sum_{k=1}^{n} b_k s_k(z) - \sum_{k=1}^{n} c_k r_k(z) \; . \end{aligned} \qquad (2.20)$$

Using (2.11) and (2.20) gives

$$G_j(z_t;a) = u(z_t)\cos\theta_j + v(z_t)\sin\theta_j$$

$$-\sum_{k=1}^{n} b_k\,[r_k(z_t)\cos\theta_j + s_k(z_t)\sin\theta_j]$$

$$-\sum_{k=1}^{n} c_k\,[r_k(z_t)\sin\theta_j - s_k(z_t)\cos\theta_j]. \tag{2.21}$$

Note that $G_j(z_t;\ a)$ is a real linear equation in the 2n variables $\{b_k\}$ and $\{c_k\}$, $k = 1, \ldots, n$, and that all the coefficients of this equation are computable directly from known data.

Define the $mp \times 2n$ real matrix B in the partitioned form

$$B = \begin{bmatrix} B_1 & D_1 \\ B_2 & D_2 \\ \vdots & \vdots \\ B_m & D_m \end{bmatrix}$$

with the $p \times n$ submatrices

$$B_t = [b_{jk}^{(t)}] \quad \text{and} \quad D_t = [d_{jk}^{(t)}]\ ,\ t = 1, \ldots, m$$

whose general real entries are

$$b_{jk}^{(t)} = r_k(z_t)\cos\theta_j + s_k(z_t)\sin\theta_j$$

$$j = 1, \ldots, p;\ k = 1, \ldots, n.$$

$$d_{jk}^{(t)} = r_k(z_t)\sin\theta_j - s_k(z_t)\cos\theta_j \tag{2.22}$$

Also, define the real vector

$$g \equiv [g_{11}, \ldots, g_{1p}, g_{21}, \ldots, g_{2p}, \ldots, g_{m1}, \ldots, g_{mp}]^T \tag{2.23}$$

of length mp, where

$$g_{tj} = u(z_t)\cos\theta_j + v(z_t)\sin\theta_j,\ t = 1, \ldots, m;\ \ j = 1, \ldots, p.$$

Finally, define the real vector

$$x = [b_1, \ldots, b_n, c_1, \ldots, c_n]^T \tag{2.24}$$

of length 2n. With this notation in hand, it is easily seen that the overdetermined system of mp equations in 2n unknowns

$$Bx = g \tag{2.25}$$

has a residual error vector defined by

$$g - Bx$$

whose mp components are precisely the mp real numbers $G_j(z_i;\ a)$ arranged in a special order. Therefore, the problem (2.18) can be solved by computing a solution to the overdetermined linear system (2.25) in the Chebyshev norm; i.e., the largest magnitude component of the residual vector g–Bx is minimized over all choices of the vector x.

This equivalent problem in linear algebra can, in principle, be solved exactly and in a finite number of steps using linear programming methods [1], [3]. The proof of this fact is the content of the following self-contained mathematical result.

Theorem 3. Let $A = [a_{jk}]$ be a real $r \times s$ matrix with $r \geqslant s \geqslant 1$, and let $b = (b_1, \ldots, b_r)$ be a real vector of length r. Let $\alpha_1^*, \ldots, \alpha_{s+1}^*, \omega^*$ denote a solution of the following primal linear program in the $s+2$ real variables $\alpha_1, \ldots, \alpha_{s+1}, \omega$ with 2r linear constraints:

Minimize: ω
subject to: $\alpha_1 \geqslant 0, \ldots, \alpha_{s+1} \geqslant 0,\ \omega \geqslant 0$,

$$\begin{aligned} &\sum_{k=1}^{s} \alpha_k a_{jk} + \alpha_{s+1} A_j + \omega \geqslant b_j, \quad j = 1, \ldots, r , \\ -&\sum_{k=1}^{s} \alpha_k a_{jk} - \alpha_{s+1} A_j + \omega \geqslant -b_j, \quad j = 1, \ldots, r , \end{aligned} \tag{2.26}$$

where

$$A_j = -\sum_{k=1}^{s} a_{jk}, \quad j = 1, \ldots, r . \tag{2.27}$$

Define

$$x_k = \alpha_k^* - \alpha_{s+1}^* , \quad k = 1, \ldots, s . \tag{2.28}$$

If

$$M = \min \max_{1 \leqslant j \leqslant r} |b_j - \sum_{k=1}^{s} a_{jk} y_k| \tag{2.29}$$

with the minimum taken over all real $y_1, \ldots, y_s$, then

$$M = \omega^* = \max_{1 \leqslant j \leqslant r} |b_j - \sum_{k=1}^{s} a_{jk} x_k| \tag{2.30}$$

where $x_1, \ldots, x_s$ are given by (2.28).

Proof. We first prove that $M \leq \omega^*$. From (2.28), we have $\alpha_k^* = \alpha_{s+1}^* + x_k$ which, substituted into the constraints (2.26) and using (2.27), gives

$$\sum_{k=1}^{s} a_{jk} x_k + \omega^* \geq b_j \ , j = 1, \ldots, r \ ,$$

and

$$-\sum_{k=1}^{s} a_{jk} x_k + \omega^* \geq -b_j \ , j = 1, \ldots, r \ .$$

Clearly, these last sets of inequalities together imply

$$\max_{1 \leq j \leq r} \ |b_j - \sum_{k=1}^{s} a_{jk} x_k| \leq \omega^* . \tag{2.31}$$

Hence, from (2.29), $M \leq \omega^*$.

We next prove that $\omega^* \leq M$. Let $x_1^*, \ldots, x_s^*$ denote any solution of (2.29). Then we may write

$$|b_j - \sum_{k=1}^{s} a_{jk} x_k| \leq M \ , \quad j = , \ldots, r$$

or, without absolute values,

$$\left. \begin{aligned} b_j - \sum_{k=1}^{s} a_{jk} x_k^* &\leq M \\ -b_j + \sum_{k=1}^{s} a_{jk} x_k^* &\leq M . \end{aligned} \right\} \quad , \quad j = , \ldots, r \tag{2.32}$$

Now, define

$$\beta_{s+1} = \max \{0, -\min_{1 \leq k \leq s} x_k^*\} \tag{2.33}$$

$$\beta_k = x_k^* + \beta_{s+1} \ , \quad k = 1, \ldots, s \ .$$

Clearly, the $s+2$ real numbers $\beta_1, \ldots, \beta_{s+1}$, M are non-negative by construction. Furthermore, substituting $x_k^* = \beta_k - \beta_{s+1}$, $k = 1, \ldots, s$, into (2.32), and using (2.27), gives

$$b_j - \sum_{k=1}^{s} a_{jk} \beta_k - \beta_{s+1} A_j \leq M$$

$$-b_j + \sum_{k=1}^{s} a_{jk} \beta_k + \beta_{s+1} A_j \leq M \ .$$

Clearly these inequalities show that the numbers $\beta_1, \ldots, \beta_{s+1}$, M satisfy all the constraints (2.26). Hence, it must be that $\omega^* \leq M$. Since we have already established that $M \leq \omega^*$, we conclude that $M = \omega^*$. Hence the inequality (2.31) must actually be an equality in light of the definition (2.29). This completes the proof.

Theorem 3 does not require that the solutions of either the linear program or of (2.29) be unique. Theorem 3 states only that from a given solution of the linear program, we may construct a solution of (2.29) using (2.28). Conversely, it is easy to see that any solution of (2.29) can be used to construct a solution of the linear program using (2.33).

An excellent algorithm, which we will refer to as ACM 495, is available in the literature [1] for solving the linear program of Theorem 3. It requires as input only the overdetermined system of equations $Ax = b$. The linear program is then set up and solved by the algorithm, so that knowledge of linear programming techniques is not necessary to use the algorithm in practice. The computational procedure, internal to the algorithm, actually solves the dual of the above primal linear program using a modification of the simplex method. The dual formulation of this problem is available in [5, p. 296]. We will not discuss the details of the linear programming technique further in this report.

A very simple modification [3, p. 863] of ACM 495 yields an algorithm for solving any real overdetermined system of linear equations in the Chebyshev norm subject to the additional constraints that all the residuals be non-negative. For A and b as in Theorem 3, this problem takes the form

$$\underset{x_1,\ldots,x_s}{\text{minimize}} \; \max_{1 \leq j \leq r} \left(b_j - \sum_{k=1}^{s} a_{jk} x_k \right) \tag{2.34}$$

subject to the r constraints

$$b_j - \sum_{k=1}^{s} a_{jk} x_k \geq 0 \, , \quad j = 1, \ldots, r \, . \tag{2.35}$$

The solution $x_1, \ldots, x_s$ returned by this modified algorithm is correct, even though the residuals returned may be in error. The correct residuals, if desired, must be calculated directly from the solution. Alternatively, if the residuals are required to be non-positive, then the same modified algorithm will work with A and b replaced by -A and -b, respectively.

Requiring non-negative residuals in the overdetermined system (2.25) has interesting geometrical interpretations. For example, if we take $p = 2$ in Lemma 2, then $\theta_1 = 0$ and $\theta_2 = \pi/2$. Thus, from (2.11), $G_1(z;a)$ and $G_2(z;a)$ are merely the real and imaginary parts of the complex error $e_n(z;a)$. Thus, the 2m components of the residual vector g-Bx are precisely the real and imaginary parts of $e_n(z;a)$ evaluated at all m data points. Therefore, if the system (2.25) is required to have non-negative residuals, we have forced the error curve to lie entirely in the first quadrant of the complex plane. More generally, we may always constrain $e_n(z;a)$ to lie in a given convex wedge shaped sector $\mathscr{W}$ of the complex plane with vertex at the origin, by making different, but appropriate, choices of the angles θ_1 and θ_2. Further exploration of this idea shows that upper and lower bounds for the error $W_n(f)$, defined by

$$W_n(f) \equiv \min_{a \epsilon C^n} \max_{z \epsilon Q_m} |e_n(z;a)|$$

$$\text{subject to: } e_n(z;a) \,\epsilon\, \mathscr{W}, \qquad z \epsilon Q_m$$

where $e_n(z;a)$ is given by (2.1), can be obtained in terms of the error $W_{np}(f)$, defined by

$$W_{np}(f) \equiv \min_{a \epsilon C^n} \max_{z \epsilon Q_m} \max_{j=1,\ldots,p} G_j(z;a)$$

$$\text{subject to: } G_j(z;a) \geq 0, \qquad z \epsilon Q_m, j = 1, \ldots, p \; ,$$

where $G_j(z;a)$ is given by (2.21), but for a different set of angles $\theta_1, \ldots, \theta_p$. The quantity $W_{np}(f)$ may be computed by solving the linear system analogous to (2.25) in the Chebyshev norm with the constraint of non-negative residuals. This approach is especially effective when the vertex angle of the wedge does not exceed $\pi/2$. This topic is not pursued further in this report.

Suppose, finally, that the complex solution vector $a \epsilon C^n$ of problem (2.12) is required to be strictly real, while f and $\{h_k\}$ are complex. Then, in the vector x of (2.24), $c_1 = \ldots = c_n = 0$. Thus, the overdetermined system $Bx = g$ of mp equations in 2n unknowns can be replaced by a smaller system $\hat{B}\hat{x} = g$ of mp equations in only n unknowns, where the mp $\times$ n real matrix $\hat{B}$ is defined in partitioned form by

$$\hat{B} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_m \end{bmatrix} \tag{2.36}$$

where the $p \times n$ submatrices $B_1, \ldots, B_m$ are unchanged from (2.22), and the real vector $\hat{x} = [b_1, \ldots, b_n]^T$. A solution of $\hat{B}\hat{x} = g$ in the Chebyshev norm can be computed using linear programming and algorithm ACM 495 as before.

III. Numerical Examples and Efficiency of Approach

We illustrate the procedure of the preceding section by approximating the complex function $f(x) = \exp(i3x)$ by a weighted sum of the basis functions 1, $\exp(ix)$, $\exp(i2x)$. That is, we seek to minimize the magnitude of the complex error curve

$$e_3(x) \equiv \exp(i3x) - \sum_{k=1}^{3} a_k \exp(i(k-1)x) \tag{3.1}$$

over interval $[0, \pi/4]$, by choice of a_1, a_2, a_3, by solving the problem $M_{np}(f)$ of (2.12). Two cases are of interest; in the first, the coefficients $\{a_k\}_1^3$ are restricted to be real, whereas in the second, these coefficients can be complex. The number m, of equispaced x-values at which (3.1) is sampled, is taken to be either 11, 101, or 1001, thereby ensuring that the smaller sample sizes are subsets of the larger sizes. The value of p, which is half the number of phase-shifted values of (3.1) employed in the error minimization, is taken to be 2, 6, 18, 54, again ensuring the subset behavior of the smaller size cases. Note that p and the phase shifts, $\{\theta_i\}$, are as given in Theorem 1 in Section II.

The optimum real coefficients in (3.1) for the problem $M_{np}(f)$ are given in table 1 for these choices of m and p, and a plot of the magnitude of the error for several representative cases is given in figure 1. The best approximation of all cases considered is afforded by m = 1001, p = 54* and its error curve is plotted as a solid line; its maximum error is .1078, which is realized at two points in the interval $[0, \pi/4]$. The cases for smaller m (less sampling of the abscissa) and smaller p (less sampling of the phase of the complex error) are poorer; for example, the maximum error for m = 11, p = 2 is .1184, realized at only one point, namely $x = \pi/4$.

Table 1. Coefficients for the Real Weight Case

m	p	a_1	a_2	a_3
11	2	.936738	-2.443144	2.518388
	6	.828404	-2.280319	2.396455
	18	.858547	-2.321885	2.425096
	54	.844146	-2.301461	2.410611
101	2	.936781	-2.443223	2.518458
	6	.831314	-2.284548	2.399525
	18	.865131	-2.331446	2.432033
	54	.853823	-2.315301	2.420506
1001	2	.936785	-2.443232	2.518466
	6	.831237	-2.284448	2.399461
	18	.865213	-2.331571	2.432127
	54	.853443	-2.314772	2.420138

We have not plotted the other error curves with real coefficients for m = 101 and 1001, because they are indistinguishable from figure 1, as a perusal of table 1 shows. For example, the coefficients for m = 11, p = 2 are very close to those for m = 101, p = 2 and m = 1001, p = 2. Thus, our sampling in x is already "fine enough" at m = 11. However, there is a significant change in the coefficients as p is varied, for a fixed value of m; that is, p = 2 yields very coarse phase-sampling of the error curve and should definitely be made larger.

The Chebyshev error curve (m = 1001, p = 54) in figure 1 realizes its maximum value at only n-1 points, rather than at n + 1 points, where n = 3 is the number of coefficients for this example. This is probably related to the fact that we have

*In this case, we observe that $a_1 - \sqrt{2}\,a_2 + a_3 \cong 0$.

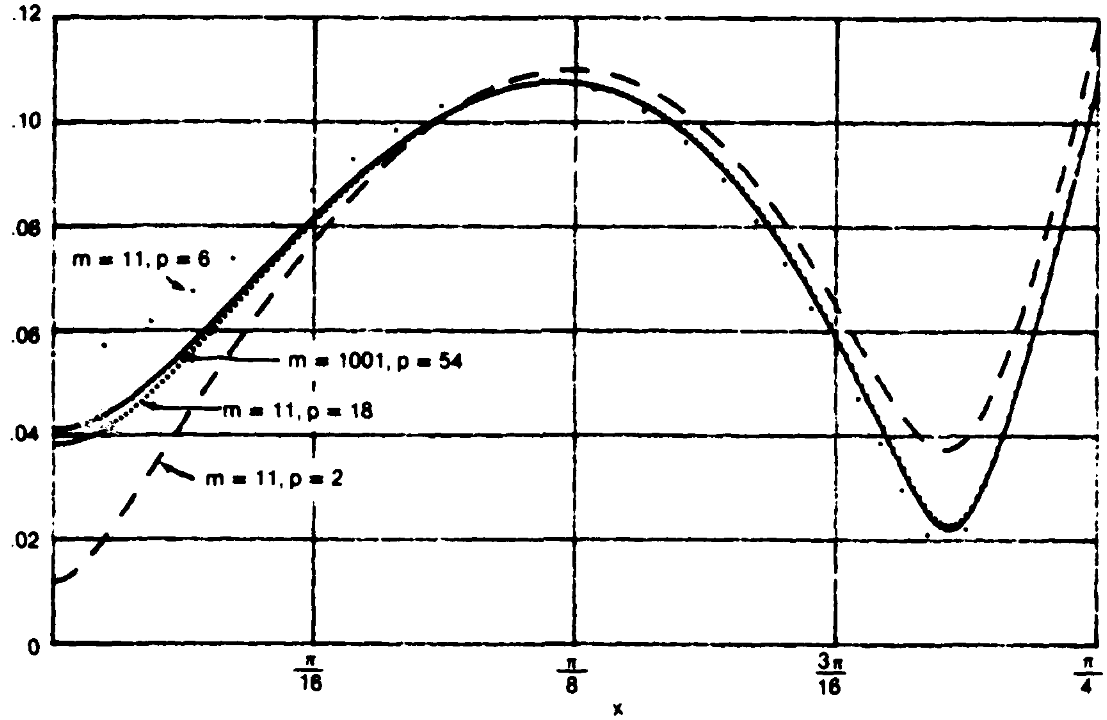


Figure 1. Error Curves for Real Coefficients; m = 11

minimized both the real and imaginary parts of the complex error, but have allowed ourselves to use only real coefficients.

The solution of the problem $M_{np}(f)$ for complex weights is given in table 2 for the same choices of m and p as above. Again, the change in coefficient values is more marked with p than with m. Magnitude-error curves for m = 11 and 101 are given in figures 2 and 3, respectively; the curves for m = 1001 are indistinguishable from those for m = 101 and are not presented.

Table 2. Coefficients for the Complex Weight Case

m	p	$Re(a_1)$	$Im(a_1)$	$Re(a_2)$	$Im(a_2)$	$Re(a_3)$	$Im(a_3)$
11	2	.364737	.954343	-2.021670	-2.119639	2.669023	1.153207
	6	.378045	.907888	-2.016657	-2.018598	2.648834	1.100488
	18	.373079	.898715	-2.003032	-2.003205	2.639992	1.094451
	54	.371586	.896504	-1.999352	-1.999473	2.637788	1.092947
101	2	.362962	.953469	-2.018255	-2.119960	2.667544	1.154238
	6	.376532	.904026	-2.012095	-2.014055	2.646131	1.099461
	18	.370549	.893500	-1.995913	-1.997062	2.635782	1.093144
	54	.368950	.890017	-1.991172	-1.991196	2.632622	1.090777
1001	2	.362947	.953499	-2.018253	-2.120028	2.667560	1.154275
	6	.376502	.903926	-2.011979	-2.013914	2.646047	1.099417
	18	.370711	.893848	-1.996440	-1.997545	2.636145	1.093278
	54	.369179	.890566	-1.991954	-1.991974	2.633175	1.091006

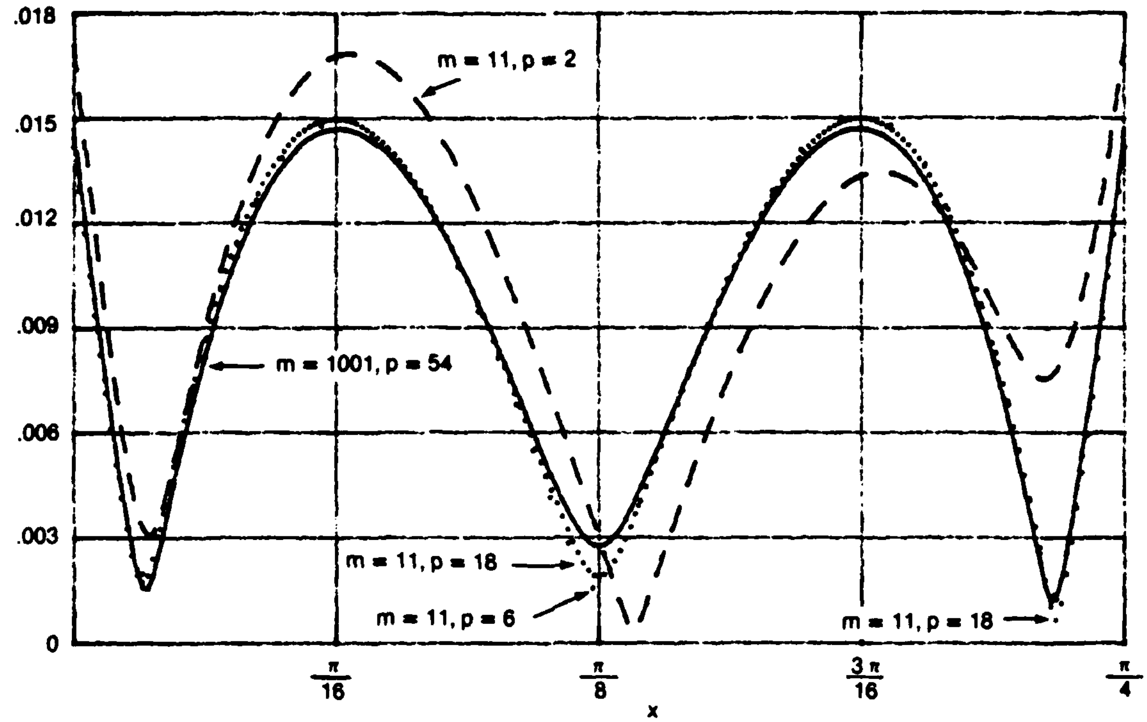


Figure 2. Error Curves for Complex Coefficients; m = 11

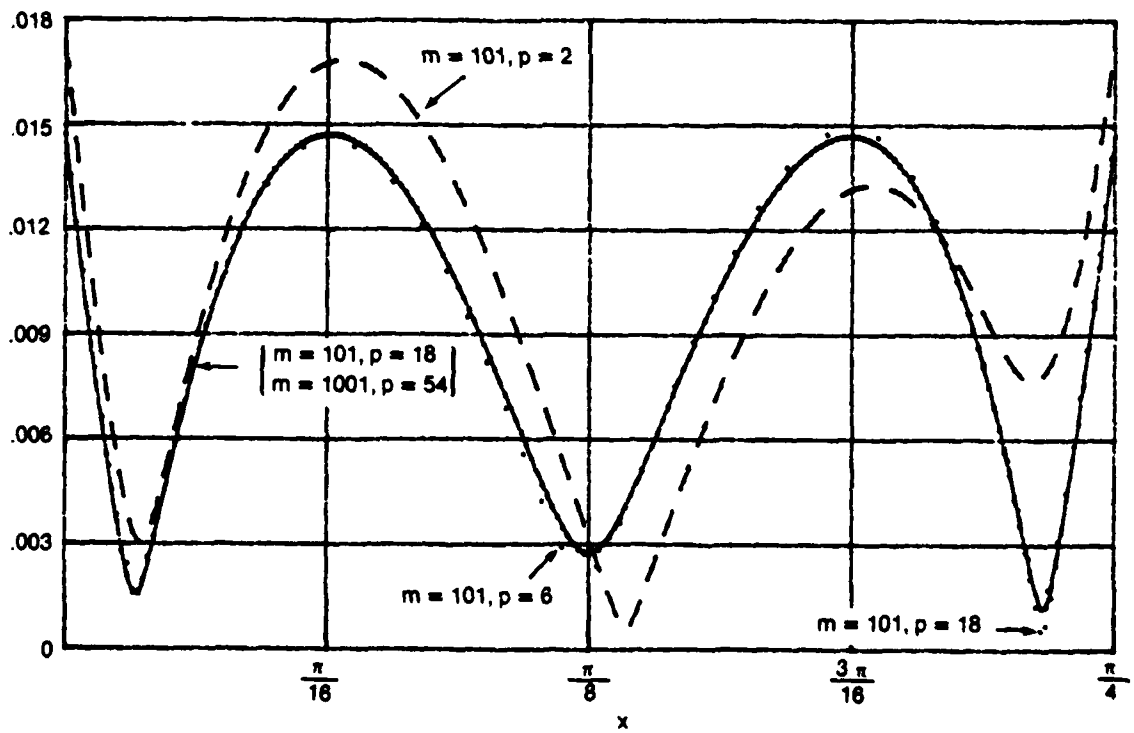


Figure 3. Error Curves for Complex Coefficients: m = 101

The Chebyshev error curve ($m = 1001$, $p = 54$) is now symmetric about the midpoint of the interval of interest and has four equal error peaks of value .0147. This error is 7.3 times smaller than that for the real coefficient case. Also the number of equal error peaks now equals 1 plus the number of coefficients; whether this property holds generally is not known.

Upper and lower bounds on the discrete Chebyshev error $E_n(f)$ for the real and complex coefficient cases are given in table 3. These bounds are precisely those presented in (2.16). They correspond to sampling the complex error (3.1) both in the abscissa x and in the phase of $e_3(x)$. The lower bounds monotonically increase with increasing m or p. The upper bounds decrease with increasing p, but increase with increasing m. All these trends follow from the fact that smaller sample sizes are subsets of the larger sizes.

Table 3. Bounds on the Discrete Chebyshev Error $E_n(f)$

m	p	Real Coefficients Lower Bound	Real Coefficients Upper Bound	Complex Coefficients Lower Bound	Complex Coefficients Upper Bound
11	2	.083718	.118396	.012089	.017097
	6	.105074	.108780	.013963	.014456
	18	.107307	.107717	.014143	.014197
	54	.107612	.107658	.014168	.014174
101	2	.083731	.118414	.012252	.017328
	6	.105192	.108893	.014436	.014946
	18	.107556	.107967	.014677	.014733
	54	.107767	.107813	.014703	.014709
1001	2	.083734	.113418	.012255	.017331
	6	.105191	.108901	.014440	.014950
	18	.107565	.107976	.014679	.014735
	54	.107775	.107821	.014704	.014712

However, the maximum magnitude error, evaluated over the continuum of x-values in the interval $[0, \pi/4]$ (actually computed on a dense discrete sampling space), obeys none of these monotonic relations, as table 4 demonstrates. For example, the maximum error in the real case for $m = 11$, $p = 18$ is less than that for $m = 11$, $p = 54$. Also, the maximum error in the complex case for $m = 11$, $p = 6$ is greater than that for $m = 101$, $p = 6$. The reason for this behavior is that we have minimized a discrete approximation to our problem of interest, sampling both in the abscissa x and in the phase values of the complex error. However, the numerical discrepancies are small, as they must be for reasonably fine sampling in both variables. (A recursive gradient procedure could be used with any of these coefficient sets to improve the final maximum magnitude-error if desired.)

The FORTRAN program listing in the Appendix is the exact code used to generate the complex weights in example (3.1) for $m = 101$ and $p = 6$. The imbedded comments should enable anyone seeking to use and understand the code to do so. Further remarks are given in the discussion in Section VI.

Table 4. Maximum Magnitude Error, Computed Over 2001 Equispaced Points in [0, π/4]

m	p	Real Coefficients	Complex Coefficients
11	2	.118396	.017097
	6	.108780	.015142
	18	.107890	.015004
	54	.107983	.015005
101	2	.118415	.017329
	6	.108893	.014946
	18	.107967	.014733
	54	.107813	.014711
1001	2	.118417	.017331
	6	.108902	.014950
	18	.107976	.014735
	54	.107821	.014712

Efficiency and timing estimates for actual calculation of complex Chebyshev approximations by the method of this report is an important consideration in some applications. If we define an operation as consisting of a multiplication followed by an addition, then it is known [6] that the number of operations per simplex iteration required by algorithm ACM 495 [1] is exactly the number of equations times the number of unknowns. In our case, the number of equations is mp, and the number of unknowns is 2n if the coefficients are complex, or n if the coefficients are required to be real. Thus, the operation count per iteration is either 2nmp or nmp. The number of iterations required is difficult to estimate, since it depends on the particular problem. However, in randomly generated problems, it has been observed [6] that the number of iterations, I, is approximately the number of unknowns times some small constant c, where usually $1 \leq c \leq 3$. (Similar estimates have been observed [7,p.160], [8] in more general linear programs as well.) Thus, in our case, $I = 2cn$ if the coefficients are complex and $I = cn$ if they are real.

The CPU time should be proportional to the total operation count, which equals the product of the number of iterations and the number of operations per iteration. That is, we expect the CPU time to be proportional to n^2mp. For the particular example here, however, we obtain an excellent fit to the limited data in table 5 with

$$\text{CPU time (msec)} = .128\, n^{1.13} m^{1.18} p^{1.18},$$

where $n = 6$ if the coefficients are complex, and $n = 3$ if they are real. This fit was obtained by letting the exponents of n, m, and p vary separately. Other examples, however, lead us to anticipate that, more generally,

$$\text{CPU time} \propto n^2 (mp)^{1.2},$$

with a proportionality factor of the order of .01-.03 msec, where n is either twice the number of approximation coefficients if the coefficients are complex, or exactly the number of coefficients if they are required to be real.

Table 5. Number of Simplex Iterations and CPU Time

m	p	Real Coefficients Simplex	Real Coefficients CPU (s)	Complex Coefficients Simplex	Complex Coefficients CPU (s)
11	2	6	.02	10	.05
	6	8	.08	15	.16
	18	11	.23	21	.58
	54	13	.81	27	2.25
101	2	7	.25	10	.40
	6	9	.73	17	1.60
	18	13	2.65	21	5.78
	54	15	11.39	28	24.27
1001	2	9	3.05	13	5.00
	6	10	10.34	17	19.38
	18	13	48.16	24	105.47
	54	16	170.52	28	359.20

The CPU time estimates apply, of course, only to the DEC VAX 11/780 computer on which the calculations were performed. The virtual memory feature of this system allows very large problems to be solved; however, for sufficiently large problems, the system overhead incurred (page faulting, and so on) may significantly and adversely affect these estimates.

IV. Application to Array Design with a Constraint

Consider a linear antenna array with N elements located at arbitrary fixed positions $\{x_k\}_1^N$, receiving a plane wave arrival of wavelength λ from direction θ_a, $-\frac{\pi}{2} \leq \theta_a \leq \frac{\pi}{2}$, relative to a normal to the array. If the array is steered to look in direction θ_ℓ, $-\frac{\pi}{2} \leq \theta_\ell \leq \frac{\pi}{2}$, then the complex transfer function of the beamformer is given by

$$T(u) = \sum_{k=1}^{N} w_k \exp(-id_k u) , \tag{4.1}$$

where $\{w_k\}_1^N$ are the element weights, and

$$d_k = 2\pi x_k/\lambda \quad \text{for } 1 \leq k \leq N ,$$

$$u = \sin\theta_a - \sin\theta_\ell .$$

Observe that the total range of u depends on the look direction θ_ℓ; for example, if $\theta_\ell = 0$, then the range of u is the closed interval [-1,1]. The peak response of T(u) should occur at u = 0, so we normalize (without loss of generality) according to

$$T(0) = 1 = \sum_{k=1}^{N} w_k .$$

To realize small sidelobes, we must minimize $|T(u)|$ for all u values in some subset U of the total range of u. For example, if $\theta_f = 0$, the total range of u is $[-1,1]$, and U could be the union of intervals $[-1,-u_0]$ and $[u_0,1]$, where $u_0>0$ is chosen small relative to 1. For the special case of real weights $\{w_k\}$, since from (4.1), $T(-u) = T^*(u)$, we can confine attention to $U = [u_0\ ,\ 1]$. The normalization constraint is most easily accounted for by solving for w_N and eliminating it; we obtain then

$$T(u) = \exp(-id_N u) - \sum_{k=1}^{N-1} w_k\,(\exp(-id_N u) - \exp(-id_k u))\ . \tag{4.2}$$

This problem now fits the framework of (2.1) if we identify

$$\begin{aligned}
z &= u \\
n &= N-1 \\
e_n(z) &= T(u) \\
f(z) &= \exp(-id_N u) \\
a_k &= w_k \\
h_k(z) &= \exp(-id_N u) - \exp(-id_k u) \\
Q_m &= \text{finite subset of } U\ .
\end{aligned} \tag{4.3}$$

There has been no statement, thus far, as to the real or complex nature of the weights $\{w_k\}$. This distinction depends upon the application and the capability of the beamformer. Both cases fit the above framework; the only difference is that the number of unknowns to be solved for will be twice as large for the complex weights as for the real weights.

If the array is half-wavelength equispaced, then the computed element weights will be identical to the classical Dolph-Chebyshev weights and can, in this instance, be computed analytically. The general case of arbitrary spacings, however, cannot be computed analytically, yet the algorithm presented in this report can always be applied.

In the remainder of this section, we presume that the elements are equispaced at half-wavelength. Then $x_k = k\lambda/2$ and (4.1) becomes

$$T(u) = \sum_{k=1}^{N} w_k \exp(-i\pi k u)\ . \tag{4.4}$$

Observe now that T(u) in (4.4) has period 2 in u, regardless of whether the weights $\{w_k\}$ are real or complex, or whether some elements have failed, i.e., zero weight values. This means that we can study and control T(u) in (4.4) over any convenient u-interval of length 2, and need not confine our investigation to $[-1,1]$. In particular, we concentrate on the u-interval $[0,2]$ in the following.

As an illustration of the capability of the minimization technique of this report, a 50-element half-wavelength equispaced linear array was initially designed for peak sidelobes of -30 dB relative to the main peak. This is of course a standard Dolph-Chebyshev case, and gives -30 dB sidelobes throughout the u-range $[u_0, 2-u_0]$, where u_0 = .0538117.* Then 10% of the elements were randomly eliminated from the array, but the remaining weights were unchanged; this corresponds to 5 elements failing in the array. The relative response of this particular array, with elements 7, 22, 40, 43, 50 failed, is illustrated in figure 4. The peak sidelobe has increased from -30 dB to -21.58 dB, a degradation of 8.4 dB, and there is a large variety of different size peaks.

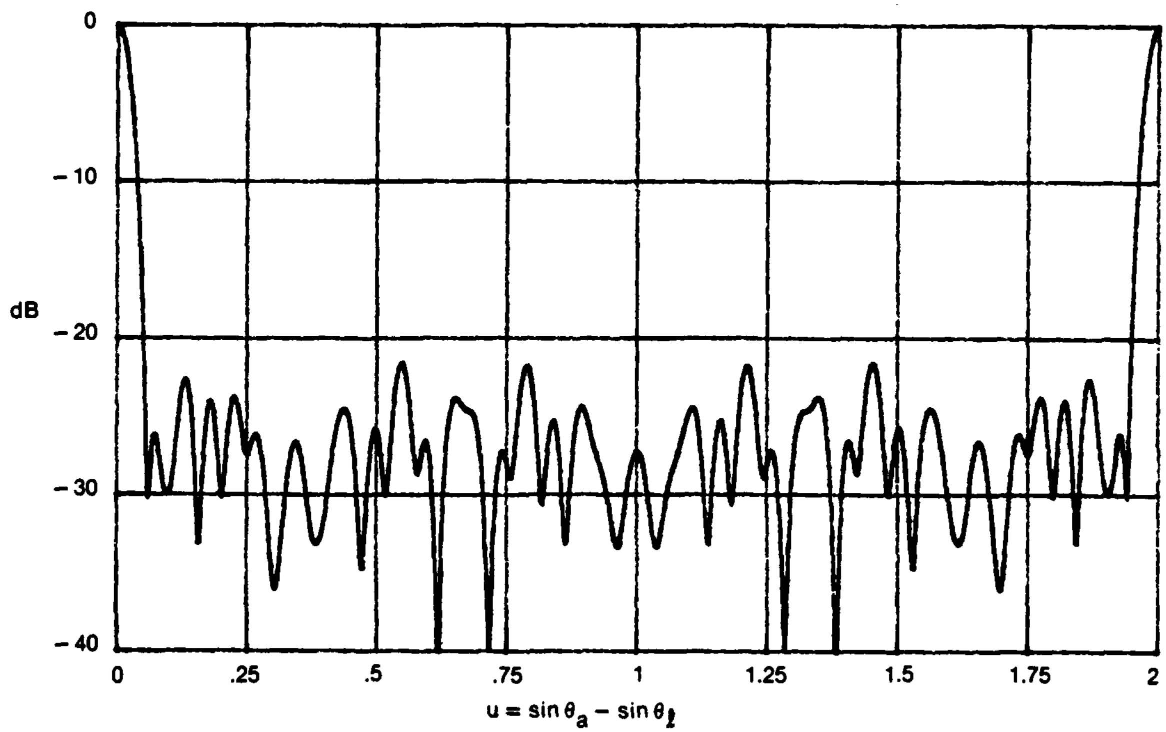


Figure 4. Relative Pattern for 5 Elements Failed

When our method with p = 2 and m = 251 equispaced points in $[u_0, 2-u_0]$ is applied to this defective array, and the remaining 45 elements are weighted with real coefficients, subject to the constraints that the mainlobe width be the *same* as the ideal 50-element array, and that the steering range in u be the same, the resultant array pattern is displayed in figure 5. The peak sidelobe is now -23.62 dB, an improvement of 2.04 dB over figure 4; however, there is still a significant variation in the values of the sidelobes, due to an insufficient number of phase controls, namely only p = 2.

When we increase the parameter values to p = 8, m = 501, the resultant best real weights are displayed graphically in figure 6 and the corresponding array pattern is given in figure 7. The gaps in figure 6 at locations 7, 22, 40, 43, 50 correspond to zero weighting at the failed elements. The general character of the weights is a bell-

*For an N-element array and -t dB peak sidelobes, we have $u_0 = (2/\pi)\ \mathrm{arc\ cos}(1/z_0)$ where $2z_0 = [r + \sqrt{r^2-1}\,]^{1/M} + [r - \sqrt{r^2-1}\,]^{1/M}$, $r = 10^{t/20}$, and $M = N-1$.

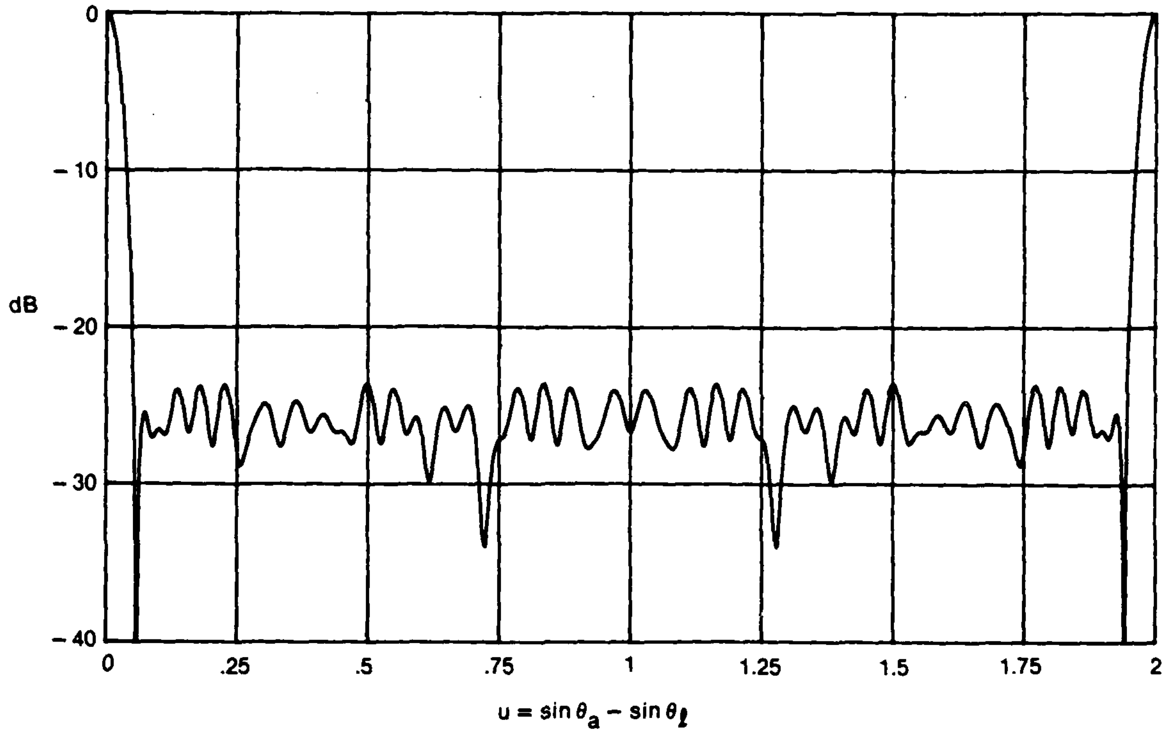


Figure 5. Relative Pattern for p = 2, m = 251, Real Weights

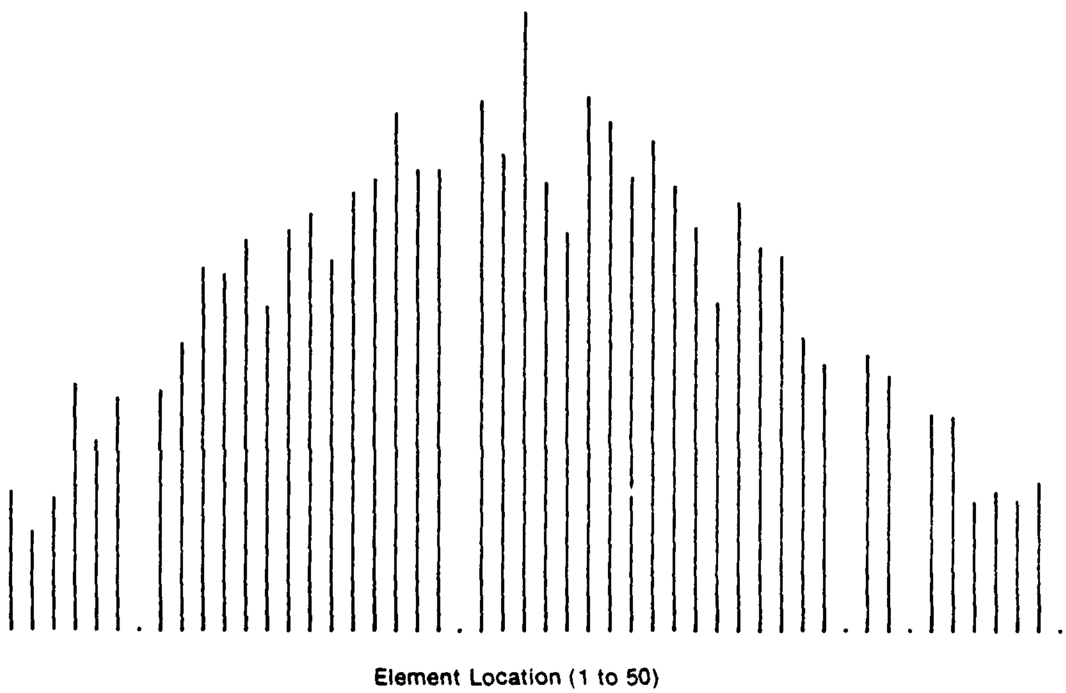


Figure 6. Best Real Weights for p = 8, m = 501

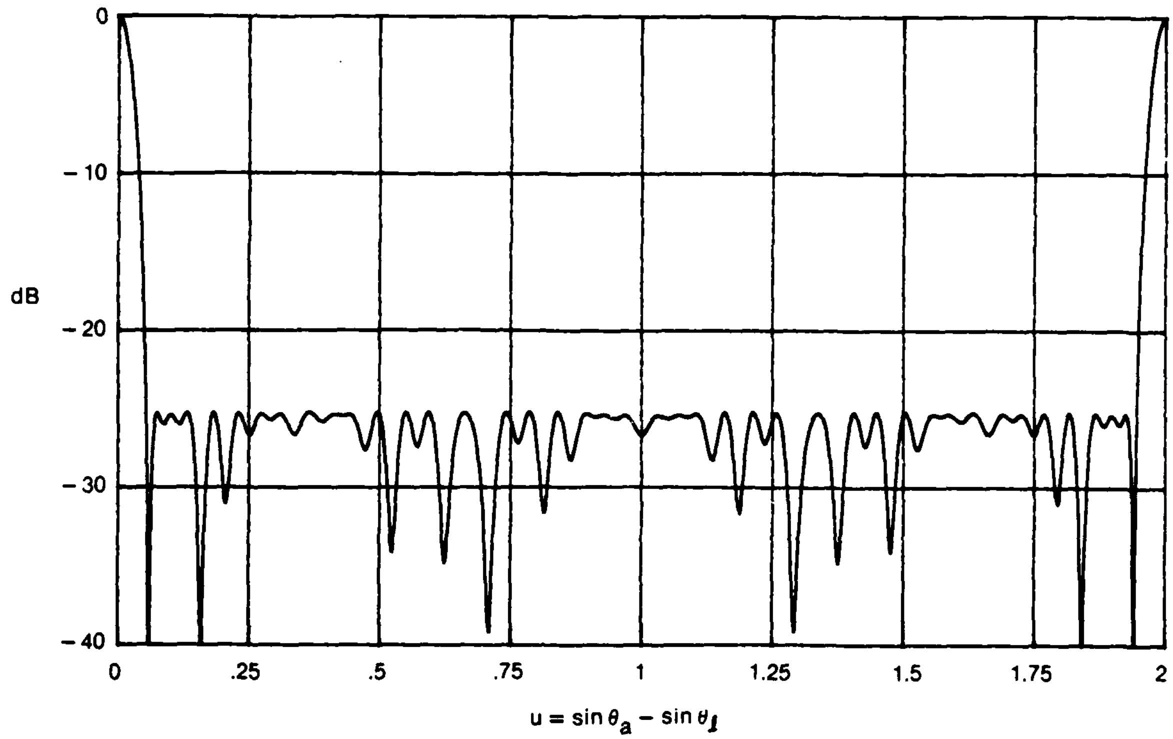


Figure 7. Relative Pattern for p = 8, m = 501, Real Weights

shaped one of all positive numbers, but there is significant fluctuation in the actual weight values, of the order of 10%. The pattern in figure 7 has a peak sidelobe of -25.20 dB, an improvement of 3.62 dB over figure 4, but still 4.80 dB poorer than the ideal 50-element array.

When the weights were allowed to be complex, and the maximum sidelobe minimized in the *same* steering range $[u_0, 2-u_0]$ for $p=2$ and $m=501$ equispaced points in $[u_0, 2-u_0]$, the best complex weights turned out to be virtually pure real, and the corresponding pattern was almost identical to figure 5. A much improved pattern for complex weights was achieved when we took $p=8$, $m=501$; in fact, the best complex weights were real (within 10^{-6} relative error) and the pattern was the same as figure 7. Although we had anticipated a better pattern for the complex weight case than for the real weights, that did not materialize; the best complex weights for this equispaced linear array with 5 missing elements were real. The reason for this behavior is unknown, but it is an encouraging result from the array design viewpoint, for it indicates that there is no need to allow phasing at the individual elements; gain alone will achieve all the sidelobe reduction that can be achieved. This conclusion is drawn only for the half-wavelength equi-spaced line array with omnidirectional element response.

The use of linear programming to design antenna arrays is not entirely new. In [9] and [10], linear programming was used to synthesize desired complex transfer functions to within 3 dB of the best possible sidelobe level. Their method corresponds to taking $p=2$ in the method presented in this report.

The computation of the real weights of figure 6 (where $p = 2$, $m = 251$, and $n = 44$) and of figure 7 (where $p = 8$, $m = 501$, and $n = 44$) required 1.2 minutes/205 iterations and 38.4 minutes/402 iterations, respectively. On the other hand, when the weights were allowed to be complex (replacing $n = 44$ by $n = 88$, but leaving p and m unchanged in both cases), the computations required 7.0 minutes/657 iterations and 179 minutes/1262 iterations, respectively. The two of these four cases requiring the smallest CPU times encountered almost no system overhead due to program size. However, the two cases requiring the largest CPU times encountered very significant system overhead because their large memory requirements caused significant usage of the virtual memory feature of the DEC VAX 11/780. The 38.4 minute case required over 3.6 million page faults, while the 179 minute case required over 11 million page faults. It is important to bear in mind that the DEC VAX 11/780 is essentially a mini-computer, and that without virtual memory, only the largest mainframe computers could have solved either of these two problems.

V. Efforts to Extend the Method

Our basic problem is to minimize the maximum magnitude of complex error

$$e_n(z) = f(z) - \sum_{k=1}^{n} a_k h_k(z) \tag{5.1}$$

over a continuum of values of z, when f, $\{h_k\}$, and $\{a_k\}$ are complex. We immediately approximate this desired problem by discretizing the z variable to a finite number of values, in order to make the problem computable. Furthermore, at any z value of interest, we additionally discretize the number of phase errors we are willing to consider. To be specific, since the algorithm in [1] applies only to real quantities, we consider the "projection" of a rotated version of the complex error:

$$P(z,\Psi) = \mathrm{Re}\{\exp(i\Psi)\, e_n(z)\} \ . \tag{5.2}$$

Then, since the argument of complex error (5.1) is unknown *a priori*, we let Ψ take on a finite set of values spaced over any π radian interval, and minimize the magnitude of projection (5.2) over all these selected Ψ values. This is equivalent to the method of Section II.

In an effort to eliminate this second discretization process in Ψ, a perturbation method was put forth in [3] that claimed guaranteed convergence to the optimum weights, for any given finite discrete set of z-values. When applied to the examples in [3], the proposed perturbation technique did indeed converge. However, when applied to the following example, of approximation of exp(i3x) by the three basis functions 1, exp(ix), exp(i2x), over 100 equispaced points in the domain $[0, \pi/4]$ in x, it sometimes failed to converge, depending on the initial weights employed. The reason for this failure is that the "direction of the minimum" furnished by the perturbation is often totally irrelevant, and the best scale factor to apply to this perturbation is very small. Thus there occurs a small random meander in the coefficient space, and occasional convergence to a non-optimum point.

A modification of this technique was attempted wherein the magnitude of the perturbation was bounded. Although this improved the situation somewhat, convergence to the optimum was not always obtained.

It was thought that this meander in coefficient space might be eliminated by tracking the exact z-values at which (5.1) is a maximum. Recall that in the real case discussed in the Introduction, convergence to the absolute optimum over a continuum of real z-values was achieved in a practical example by re-evaluating the z-points of maximum error and using these in a recursive approach. When this idea was extended to the two continuous variables z, Ψ in (5.2), and only the $2n+1$ largest error points were retained, convergence was not obtained. When however, the single "point" of a maximum, i.e., a pair of values (z_k, Ψ_k), was replaced by a "patch", i.e., a set of values $\{(z_{kp}, \Psi_{kp})\}$ covering the maximum point (z_k, Ψ_k), the convergence to the absolute optimum for the examples considered was apparently achieved. The patch width in Ψ was of the order of a degree in most cases. The problem with this latter modification is that a large number of computations of the error function and its derivative must be evaluated, and the improvement over the method of Section II is insignificant when p there is large.

If the final error in (5.2), after application of the method of Section II, is inadequate, due to inadequate sampling in z and/or Ψ, it is possible, for a given coefficient set $\{a_k\}$, to locate the point (z_m, Ψ_m) at which (5.2) is largest, and then use a gradient approach to decrease this maximum error at (z_m, Ψ_m). Of course, the particular point of maximum will jump around as the set $\{a_k\}$ is perturbed; nevertheless, the technique does converge (although slowly) and does lead to smaller errors at the maximum of (5.2) in a continuum for z and Ψ.

VI. Discussion and Summary

It has been observed that two of the locations of maximum magnitude error often occur at the endpoints, if the specified domain in (1.2) is a real interval; for example, see figures 2 and 3. (The example of real coefficients in figure 1 had one of the maximum error points at an endpoint, but not the other. However, if we had specified domain $[-\pi/4, \pi/4]$ in that example, we would have observed four peak-error points, two of which would have been at endpoints, due to the conjugate property of the desired function and the basis functions.) Since the endpoints may be the only ones we can anticipate *a priori* and specify as locations of maximum error, an obviously useful procedure is to use more values of phase shift Ψ in (5.2) (alternatively, the angles $\{\theta_j\}$ in Lemma 2) at the endpoints than in the interior, so as to better control these very-likely locations of maximum error. For example, we might use $p = 6$ in the interior of a specified real interval domain of z and use $p = 12$ or 20 at the two endpoints. This does not add greatly to the total computation, since there are generally far more interior points than (two) endpoints. The program in the Appendix may be readily used with different values of p at different data points by exploiting the INDEX array in the user-supplied subroutine named ZPHASE.

The p different phase shifts Ψ selected in (5.2) have been chosen here to be equally spaced over a 180° span (along with their 180° mates). This is the most reasonable selection in the absence of *a priori* knowledge of the complex error, its magnitude, and phase because it gives the best upper bound (2.7) in Lemma 2 of any set of phases. However, one could select any value of Ψ to investigate the error; for example, different sets of values of Ψ could be used at various values of abscissa z. The program in the Appendix may be used with any desired set of phases at any, or all, of the data points simply by altering the user-supplied subroutine named ZPHASE.

The potential for significant round-off error accumulation is always present in linear Chebyshev complex function approximation. For example, in approximating $f(x) = \cos(12x) + i \sin(3x)$ by a complex linear combination of the 12 basis functions 1, exp(ix), . . . , exp (i11x) on the interval $[0, \pi/4]$, the complex coefficients of best approximation were observed to be large in magnitude and lie in all quadrants of the complex plane; therefore, significant numerical round-off error occurred during computation of the residuals within algorithm ACM 495[1]. Even if the coefficients of best approximation had happened to be better behaved, serious cancellation error may still occur in some problems because of the very nature of complex arithmetic. It might, therefore, be wise to use a double precision version of algorithm ACM 495 routinely in complex Chebyshev approximation problems to alleviate such cancellation errors.

One method of detecting the presence of significant round-off errors is supplied by the nature of the approximation problem itself. That is, Theorem 1 and the third step of the proof of Theorem 2 together imply that

$$M_{np}(f) \leqslant \mathscr{E}_{np}(f) \leqslant M_{np}(f) \sec(\pi/2p) \ . \qquad (6.1)$$

Once $M_{np}(f)$ and the coefficients have been computed in algorithm ACM 495, these bounds may be checked to see if significant numerical round-off error has occurred. In the example presented in the Appendix, rounding to 5 significant digits gives

$$.014436 = M_{np}(f) < \mathscr{E}_{np}(f) = M_{np}(f) \sec(\pi/2p) = .014946 \ .$$

However, if we round to 6 significant digits instead, it is seen that the second inequality in (6.1) does not quite hold. We conclude that the effects of round-off errors, although visible in the results, are not significant in this example. (Single precision numbers on the DEC VAX 11/780 have approximately 7 significant decimal digits.)

A sensitivity analysis on the optimum coefficients may be in order in some applications to determine their utility. This consideration is completely independent of their numerical accuracy. For example, in an antenna array design problem where some elements are spaced significantly less than a half-wavelength apart, it might well turn out that the optimum coefficients need to be specified with a relative error of better than 10^{-6}. Then, although the mathematical results may be correct and accurate, practical usage is precluded. This sensitivity can be determined by perturbing the optimum weights a few percent and observing if a drastic change occurs on the desired sidelobe behavior. (Such arrays are referred to as super-directive arrays.)

References

1. I. Barrodale and C. Phillips, "Solution of an Overdetermined System of Linear Equations in the Chebyshev Norm," Algorithm 495, *ACM Transactions on Mathematical Software*, vol. 1, no. 3, September 1975, pp 264-270.

2. A. H. Nuttall, "Some Windows with Very Good Sidelobe Behavior," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, no. 1, February 1981. (Also in NUSC Technical Report 6239, 9 April 1980.)

3. I. Barrodale, L. M. Delves, and J. C. Mason, "Linear Chebyshev Approximation of Complex-Valued Functions," *Mathematics of Computation*, vol. 32, no. 143, July 1978, pp. 853-863. MR 58 #3313.

4. G. Meinardus, *Approximation of Functions: Theory and Numerical Methods*, Springer-Verlag, 1967.

5. I. Barrodale and A. Young, "Algorithms for Best L_1 and L_∞ Linear Approximations on a Discrete Set," *Numerische Mathematik*, vol. 8, 1966, pp. 295-306.

6. I. Barrodale, private communication, 18 December 1980.

7. G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, Princeton, N.J., 1963.

8. E. H. McCall, *Performance Results of the Simplex Algorithm for a Set of Real-World Linear Programming Models*, Technical Report 80-4, Computer Science Department, University of Minnesota, Minneapolis, Minnesota, January 1980.

9. G. W. McMahon, B. Hubley, and A. Mohammed, "Design of Optimum Directional Arrays Using Linear Programming Techniques", *Journal of the Acoustical Society of America*, vol. 51, no. 1, part 2, 1972, pp. 304-309.

10. G. L. Wilson, "Computer Optimization of Transducer-Array Patterns," *Journal of the Acoustical Society of America*, vol. 59, no. 1, Jan. 1976, pp. 195-203.

Appendix

The Computer Program

```
$ RUN ZACM

CPU TIME IN SEC       =        1.61
PAGE FAULTS           =           1
NUMBER OF ITERATIONS =          17

NUMBER OF DATA POINTS       M   =          101
NUMBER OF BASIS FUNCTIONS   N   =            3
NUMBER OF PHASES PER POINT  P   =            6

LOWER BOUND FOR BEST CHEBYSHEV ERROR     =   0.14436293E-01
UPPER BOUND FOR BEST CHEBYSHEV ERROR     =   0.14946010D-01
LOWER BOUND * SECANT( PI/(2*P) )         =   0.14945551E-01

CALCULATED RANK =           6 ,AS EXPECTED.

UNIQUE SOLUTION:

              REAL PART        IMAG PART
   1         0.37653193       0.90402567
   2        -2.01209474      -2.01405549
   3         2.64613128       1.09946144
```

```
$ TYPE ZMAIN.FOR


C        ***********************************************************
C        THIS MAIN ROUTINE SOLVES A LINEAR COMPLEX FUNCTION
C        APPROXIMATION PROBLEM. THE COMPUTED APPROXIMATION CAN
C        BE MADE AS CLOSE TO THE BEST CHEBYSHEV, OR MINIMAX,
C        APPROXIMATION AS DESIRED. THE APPROXIMATION IS CONSTRUCTED
C        ON A FINITE DATA SET FROM ARBITRARY BASIS FUNCTIONS.
C        ***********************************************************
C
C        *************************************************************
C        REFERENCE:
C             ROY L. STREIT AND ALBERT H. NUTTALL, "LINEAR CHEBYSHEV
C        COMPLEX FUNCTION APPROXIMATION," NUSC TECHNICAL REPORT 6403,
C        NAVAL UNDERWATER SYSTEMS CENTER, NEW LONDON, CT,06320.
C        *************************************************************
C
C        **************************************************************
C        THIS APPROACH SOLVES AT MOST M*P LINEAR EQUATIONS IN 2*N
C        UNKNOWNS IN THE CHEBYSHEV NORM; THAT IS, THE MAXIMUM MAGNITUDE
C        RESIDUAL IS MINIMIZED. ALL EQUATIONS AND UNKNOWNS ARE REAL.
C        THE SOLUTION IS COMPUTED USING LINEAR PROGRAMMING.
C        SINGLE PRECISION IS USED TO SOLVE THE SYSTEM OF EQUATIONS;
C        HOWEVER, DOUBLE PRECISION IS USED TO SET UP THE SYSTEM
C        ITSELF IN ORDER TO MINIMIZE POSSIBLE ROUND-OFF ERRORS.
C        **************************************************************
C
C        ***********************************************************
C        THE N COMPLEX COEFFICIENTS OF APPROXIMATION ARE GIVEN BY:
C
C                COEF(K) + I * COEF(N+K) ,  K=1,2,...,N.
C
C        THE COEF ARRAY IS COMPUTED IN SUBROUTINE ACM495.
C        ***********************************************************
C
C        *************************************************************
C        USERS MUST SPECIFY THE FOLLOWING SIX NUMBERS:
C
C        THE NUMBER OF BASIS FUNCTIONS:
         PARAMETER N=3
C
C        THE NUMBER OF DATA POINTS:
         PARAMETER M=101
C
C        THE NUMBER OF PHASES PER DATA POINT: [P .GE. 2]
         PARAMETER P=6
C
C        MUST THE FINAL COEFFICIENTS BE REAL? [IREAL=1 IFF YES]
         PARAMETER IREAL=0
C
C        MUST THE RESIDUALS BE NON-NEGATIVE? [ISIDES=1 IFF YES]
         PARAMETER ISIDES=0
C
C        RELATIVE ERROR CRITERION: [RELERR=0.0 IFF CHEBYSHEV SOLUTION]
         DATA RELERR/0.0/
```

```
C
C
C       USERS NEED CHANGE NOTHING MORE IN THIS ROUTINE, BUT
C       THEY MUST MAKE APPROPRIATE CHANGES IN THE FOLLOWING
C       SUBROUTINES CALLED BY THIS MAIN ROUTINE:
C
C               ZFUNCT   -   DEFINES THE FUNCTION APPROXIMATED
C               ZBASIS   -   DEFINES THE BASIS FUNCTIONS
C               ZABSCS   -   DEFINES THE FINITE DATA SET
C               ZPHASE   -   DEFINES THE PHASES AT ALL DATA POINTS
C
C       ************************************************************
C
        PARAMETER MIP=M*P,MDIM=MIP+1,N2=2*N,NDIM=N2+3
        DIMENSION BDATA(NDIM,MDIM),FDATA(MDIM),COEF(NDIM),
     1            INDEX(M)
        DOUBLE PRECISION COSDTA(MIP),SINDTA(MIP),ARG(MIP),ZRDATA(M),
     1                   ZIDATA(M),RESIDR(M),RESIDI(M),CHEBER,DZERO,PI
        INTEGER OCODE,RANK
        DATA ZERO/0.0/
        DATA DZERO/0.D0/
        DATA PI/3.14159265358979323 8D0/
C
C       SET THE UNIT ROUND-OFF ERROR (FOR VAX 11/780)
        DATA TOL/.596E-7/
C
C       **********
C       INITIALIZE
C       **********
C
        IF(MIP.GE.N)GO TO 50
        PRINT 51,MIP,N
   51   FORMAT(/,' ***** INITIALIZATION  ERROR:  M*P MUST EXCEED N',
     1         /,' ***** BUT: M*P = ',I10,', AND N = ',I10,
     2         /,' *****',
     3         /,' *****              EXECUTION TERMINATED')
        GO TO 9999
   50   CONTINUE
C
        DO 89 I=1,N2
        COEF(I)=ZERO
   89   CONTINUE
        DO 91 I=1,M
        RESIDR(I)=DZERO
        RESIDI(I)=DZERO
   91   CONTINUE
C
C       ******************
C       DEFINE THE PROBLEM
C       ******************
C
C       DETERMINE THE DATA POINTS
        CALL ZABSCS(ZRDATA,ZIDATA,M)
C
C       DETERMINE THE PHASES AT ALL DATA POINTS
        IP=P
        CALL ZPHASE(INDEX,ARG,M,IP,MIPSUM,RESIDR,RESIDI)
C
C       COMPUTE THE NECESSARY SINES AND COSINES
```

```
      CALL ZTRIGD(ARG,COSDTA,SINDTA,MIPSUM,INDEX)
C
C     SET UP THE OVER-DETERMINED SYSTEM OF REAL EQUATIONS
      CALL ZFNSET(BDATA,FDATA,COSDTA,SINDTA,INDEX,NDIM,N,M,
     1            ZRDATA,ZIDATA,IREAL)
C
C     SET CONSTRAINT IF COEFFICIENTS MUST BE REAL
      NSET=N2
      IF(IREAL.EQ.1)NSET=N
C
C     SET OPTION FOR ONE SIDED SOLUTION OF OVER-DETERMINED SYSTEM
      NSIDES=2
      IF(ISIDES.EQ.1)NSIDES=1
C
C     GET INITIAL TIMING AND PAGING INFORMATION (FOR THE VAX 11/780)
      CALL GETJPI(NCPU1,NPFS1)
C
C     ******************************************************
C     SOLVE THE OVER-DETERMINED SYSYTEM OF MIPSUM EQUATIONS
C     IN NSET UNKNOWNS. ALL EQUATIONS AND UNKNOWNS ARE REAL.
C     ******************************************************
C
      CALL ACM495(MIPSUM,NSET,MDIM,NDIM,BDATA,FDATA,TOL,RELERR,COEF,
     1            RANK,RESMAX,ITER,OCODE,NSIDES)
C
C     COMPUTE THE RESIDUALS DIRECTLY FOR GREATER ACCURACY
      CALL ZRESID(RESIDR,RESIDI,N,M,COEF,ZRDATA,ZIDATA,CHEBER)
C
C     GET FINAL TIMING AND PAGING INFORMATION (FOR VAX 11/780)
      CALL GETJPI(NCPU2,NPFS2)
C
C     ******************
C     PRINT SUMMARY DATA
C     ******************
C
C     PRINT ELAPSED TIMING AND PAGING INFORMATION (FOR VAX 11/780)
      DCPU=(NCPU2-NCPU1)/100.
      IDPF=NPFS2-NPFS1
      PRINT 110,DCPU
  110 FORMAT(/,' CPU TIME IN SEC       = ',1F10.2)
      PRINT 114,IDPF
  114 FORMAT(' PAGE FAULTS           = ',I10)
C
      PRINT 100,ITER
  100 FORMAT(' NUMBER OF ITERATIONS = ',I10)
      PRINT 111,M
  111 FORMAT(/,' NUMBER OF DATA POINTS       M  = ',I10)
      PRINT 112,N
  112 FORMAT(' NUMBER OF BASIS FUNCTIONS  N  = ',I10)
      PRINT 115,P
  115 FORMAT(' NUMBER OF PHASES PER POINT P  = ',I10)
      IF (IREAL.EQ.1)PRINT 134
  134 FORMAT(/,' THE COEFFICIENTS ARE REQUIRED TO BE REAL.')
C
      PRINT 103,RESMAX
  103 FORMAT(/,' LOWER BOUND FOR BEST CHEBYSHEV ERROR   = ',1E16.8)
      PRINT 107,CHEBER
  107 FORMAT(' UPPER BOUND FOR BEST CHEBYSHEV ERROR   = ',1D16.8)
      SEC=RESMAX/COS(PI/(2.D0*P))
```

```
      PRINT 109,SEC
  109 FORMAT(' LOWER BOUND * SECANT( PI/(2*P) )       = ',1E16.8)
C
      IF (RANK.EQ.NSET)GO TO 119
      PRINT 171,RANK
  171 FORMAT(/,' CALCULATED RANK = ',I10,' .')
      PRINT 118,NSET
  118 FORMAT(/,53('*'),/,53('*'),/,' THE RANK SHOULD EQUAL ',I10,'.',
     1 /,' CHECK FOR POSSIBLE ERRORS IN PROGRAM AND/OR PROBLEM.',
     2 /,53('*'),/,53('*'))
      GO TO 181
  119 CONTINUE
      PRINT 117,RANK
  117 FORMAT(/,' CALCULATED RANK = ',I10,' ,AS EXPECTED.')
  181 CONTINUE
C
C     PRINT RELATIVE ERROR OF MAXIMUM RESIDUAL IF NECESSARY
      IF(RELERR.GT.ZERO)PRINT 124,RELERR
  124 FORMAT(//,' RELATIVE ERROR IN THE MAXIMUM RESIDUAL INCURRED',
     1 /,' BY THIS APPROXIMATE SOLUTION = ',1E16.8,/)
C
      IF(OCODE.EQ.0)PRINT 121
      IF(OCODE.EQ.1)PRINT 122
      IF(OCODE.EQ.2)PRINT 123
  121 FORMAT(/,' THE FOLLOWING SOLUTION IS PROBAPLY NON-UNIQUE:',/)
  122 FORMAT(/,' UNIQUE SOLUTION:',/)
  123 FORMAT(/,' PREMATURE TERMINATION DUE TO ROUND-OFF ERRORS.',
     1 /,' BEST COMPUTED SOLUTION:',/)
C
      PRINT 133
  133 FORMAT(15X,'REAL PART', 7X,'IMAG PART')
C
C     ***************************************
C     PRINT THE COEFFICIENTS OF APPROXIMATION
C     ***************************************
C
      PRINT 102,(I,COEF(I),COEF(N+I),I=1,N)
  102 FORMAT(I5,3X,2F16.8)
C
C     ******************
C     END OF MAIN PROGRAM
C     ******************
C
 9999 END


$ TYPE ZFUNCT.FOR


C     ************************************************
C     SUBROUTINE ZFUNCT EVALUATES THE APPROXIMATED
C     COMPLEX FUNCTION AT A SPECIFIED DATA POINT.
C     ************************************************
C
C     ************************************************
C     ZFNR = REAL PART OF THE FUNCTION      (OUTPUT)
C     ZFNI = IMAG PART OF THE FUNCTION      (OUTPUT)
```

```
C       ZR   = REAL PART OF THE DATA POINT   (INPUT)
C       ZI   = IMAG PART OF THE DATA POINT   (INPUT)
C       **********************************************
C
        SUBROUTINE ZFUNCT(ZFNR,ZFNI,ZR,ZI)
C
        DOUBLE PRECISION ZR,ZI,ZFNR,ZFNI
C
C       ************************************
C       BEGIN USER CODE FOR SPECIFIC PROBLEM
C       ************************************
C
        ZFNR=COS(3.D0*ZR)
        ZFNI=SIN(3.D0*ZR)
C
C       **********************************
C       END USER CODE FOR SPECIFIC PROBLEM
C       **********************************
C
C
C       ************************
C       END OF SUBROUTINE ZFUNCT
C       ************************
C
        RETURN
        END


$ TYPE ZBASIS.FOR


C       **********************************************
C       SUBROUTINE ZBASIS EVALUATES THE IB-TH COMPLEX
C       BASIS FUNCTION AT A SPECIFIED DATA POINT.
C       **********************************************
C
C       *****************************************************
C       IB   = THE BASIS FUNCTION INDEX, WHERE
C                IB = 1, 2, ...                     (INPUT)
C       ZFNR = REAL PART OF THE IB-TH BASIS FUNCTION (OUTPUT)
C       ZFNI = IMAG PART OF THE IB-TH BASIS FUNCTION (OUTPUT)
C       ZR   = REAL PART OF THE SPECIFIED DATA POINT (INPUT)
C       ZI   = IMAG PART OF THE SPECIFIED DATA POINT (INPUT)
C       *****************************************************
C
        SUBROUTINE ZBASIS(IB,ZFNR,ZFNI,ZR,ZI)
C
        DOUBLE PRECISION ZR,ZI,ZFNR,ZFNI
C
C       ************************************
C       BEGIN USER CODE FOR SPECIFIC PROBLEM
C       ************************************
C
        ZFNR=COS((IB-1.D0)*ZR)
        ZFNI=SIN((IB-1.D0)*ZR)
C
C       **********************************
```

```
C       END USER CODE FOR SPECIFIC PROBLEM
C       ***********************************
C
C
C       ***********************
C       END OF SUBROUTINE ZBASIS
C       ***********************
C
        RETURN
        END


$ TYPE ZABSCS.FOR


C       ********************************************
C       SUBROUTINE ZABSCS DEFINES THE DATA POINTS ON
C       WHICH THE APPROXIMATION IS CONSTRUCTED.
C       ********************************************
C
C       *****************************************************
C       ZRDATA(I) = REAL PART OF THE I-TH DATA POINT (OUTPUT)
C       ZIDATA(I) = IMAG PART OF THE I-TH DATA POINT (OUTPUT)
C       M         = TOTAL NUMBER OF DATA POINTS      (INPUT)
C       *****************************************************
C
        SUBROUTINE ZABSCS(ZRDATA,ZIDATA,M)
C
        DOUBLE PRECISION ZRDATA(1),ZIDATA(1)
C
C       ***********************************
C       BEGIN USER CODE FOR SPECIFIC PROBLEM
C       ***********************************
C
        DOUBLE PRECISION X,PI
        DATA PI/3.14159265358979323800/
C
        DO 10 I=1,M
        X=PI*(I-1.D0)/(4.D0*(M-1.D0))
        ZRDATA(I)=X
        ZIDATA(I)=0.D0
   10   CONTINUE
C
C       ***********************************
C       END USER CODE FOR SPECIFIC PROBLEM
C       ***********************************
C
C
C       ***********************
C       END OF SUBROUTINE ZABSCS
C       ***********************
C
        RETURN
        END
```

```
$ TYPE ZPHASE.FOR


C       ************************************************
C       SUBROUTINE ZPHASE DEFINES THE PHASES AT EACH DATA
C       POINT, AS WELL AS THE TOTAL NUMBER OF PHASES.
C       NOTE THAT THE TOTAL NUMBER OF PHASES EQUALS THE
C       TOTAL NUMBER OF EQUATIONS SOLVED BY ACM495.
C       ************************************************
C
C       *** CAUTION *** THE TOTAL NUMBER OF PHASES MUST NEVER
C       *************** EXCEED THE PRODUCT M*P. HOWEVER, THE
C       *************** NUMBER OF PHASES MAY BE VARIED FROM DATA
C       *************** POINT TO DATA POINT. SEE NUSC REPORT 6043.
C
C       *************************************************************
C       INDEX(I) = NUMBER OF PHASES AT THE I-TH DATA POINT    (OUTPUT)
C       ARG      = AGGREGATE ARRAY OF ALL PHASES AT ALL DATA
C                  POINTS IN LEXICOGRAPHICAL ORDER             (OUTPUT)
C       M        = NUMBER OF DATA POINTS                       (INPUT)
C       P        = THE INTEGER PARAMETER P OF MAIN ROUTINE     (INPUT)
C       MIPSUM   = TOTAL NUMBER OF PHASES IN THE ARRAY ARG     (OUTPUT)
C
C       THE FOLLOWING TWO ARRAYS ARE ZERO FILLED, UNLESS AN
C       INITIAL APPROXIMATION HAS BEEN PROVIDED IN THE MAIN
C       ROUTINE. USE ONLY IF NEEDED, OR ELSE IGNORE THEM.
C
C       RESIDR(I)= REAL PART OF COMPLEX RESIDUAL AT THE I-TH
C                  DATA POINT                                  (INPUT)
C       RESIDI(I)= IMAG PART OF COMPLEX RESIDUAL AT THE I-TH
C                  DATA POINT                                  (INPUT)
C
C       ************************************************************
C
        SUBROUTINE ZPHASE(INDEX,ARG,M,P,MIPSUM,RESIDR,RESIDI)
C
        INTEGER P
        DIMENSION INDEX(1)
        DOUBLE PRECISION ARG(1),RESIDR(1),RESIDI(1)
C
C       ************************************
C       BEGIN USER CODE FOR SPECIFIC PROBLEM
C       ************************************
C
C       THIS CODE DEFINES THE PHASES
C         PI*(J-1)/P ,J=1,2,...,P,
C       AT EACH OF THE M DATA POINTS.
C
        DOUBLE PRECISION PI,X
        DATA PI/3.14159265358979323800/
C
C       DEFINE THE NUMBER OF PHASES AT EACH DATA POINT
C
        DO 10 I=1,M
        INDEX(I)=P
   10   CONTINUE
C
C       DEFINE ALL THE PHASES
```

```
C
      MLOOP=0
      DO 20 I=1,M
      LOOP=INDEX(I)
      X=LOOP
      DO 30 J=1,LOOP
      MLOOP=MLOOP+1
      ARG(MLOOP)=PI*(J-1)/X
   30 CONTINUE
   20 CONTINUE
C
C     TOTAL NUMBER OF PHASES
C
      MIPSUM=MLOOP
C
C     **********************************
C     END USER CODE FOR SPECIFIC PROBLEM
C     **********************************
C
C
C     ************************
C     END OF SUBROUTINE ZPHASE
C     ************************
C
      RETURN
      END


$ TYPE ZTRIGD.FOR


C     *********************************************************
C     SUBROUTINE ZTRIGD COMPUTES THE REQUIRED SINES AND COSINES
C     *********************************************************
C
      SUBROUTINE ZTRIGD(ARG,COSDTA,SINDTA,MIPSUM,INDEX)
C
      DOUBLE PRECISION X,ARG(1),COSDTA(1),SINDTA(1)
      DIMENSION INDEX(1)
C
      DO 10 I=1,MIPSUM
      X=ARG(I)
      COSDTA(I)=COS(X)
      SINDTA(I)=SIN(X)
   10 CONTINUE
C
C     ************************
C     END OF SUBROUTINE ZTRIGD
C     ************************
C
      RETURN
      END


$ TYPE ZFNSET.FOR
```

```
C       ************************************************************
C       SUBROUTINE ZFNSET SETS UP THE TRANSPOSE OF THE COEFFICIENT
C       MATRIX OF THE REQUIRED OVER-DETERMINED SYSTEM OF EQUATIONS
C       IN THE ARRAY NAMED BDATA. SPECIFICALLY,
C
C               TRANSPOSE(BDATA) * X  =  FDATA
C
C       WHERE FDATA IS THE CONSTANT VECTOR AND X IS THE UNKNOWN
C       SOLUTION VECTOR. THE FDATA ARRAY IS ALSO FILLED BY THIS
C       SUBROUTINE. IF THE X VECTOR IS REQUIRED TO BE REAL, THEN
C       THE "LAST" HALF OF EACH OF THE REQUIRED EQUATIONS ARE
C       IGNORED IN THE FINAL COMPUTATIONS. HENCE, THIS PART OF THE
C       BDATA ARRAY IS NOT COMPUTED IF AND ONLY IF IREAL = 1 .
C       ************************************************************
C
        SUBROUTINE ZFNSET(BDATA,FDATA,COSDTA,SINDTA,INDEX,NDIM,N,M,
     1                     ZRDATA,ZIDATA,IREAL)
C
        DIMENSION INDEX(1),BDATA(NDIM,1),FDATA(1)
        DOUBLE PRECISION ZR,ZI,ZRDATA(1),ZIDATA(1),COSDTA(1),SINDTA(1),
     1                    ZFNR,ZFNI
C
C       FILL THE FDATA ARRAY
C
        MLOOP=0
        DO 10 I=1,M
        ZR=ZRDATA(I)
        ZI=ZIDATA(I)
        CALL ZFUNCT(ZFNR,ZFNI,ZR,ZI)
        LOOP=INDEX(I)
        DO 20 J=1,LOOP
        MLOOP=MLOOP+1
        FDATA(MLOOP)=ZFNR*COSDTA(MLOOP)+ZFNI*SINDTA(MLOOP)
   20   CONTINUE
   10   CONTINUE
C
C       FILL THE BDATA ARRAY
C
        DO 30 K=1,N
        IB=K
        MLOOP=0
        DO 40 I=1,M
        ZR=ZRDATA(I)
        ZI=ZIDATA(I)
        CALL ZBASIS(IB,ZFNR,ZFNI,ZR,ZI)
        LOOP=INDEX(I)
        DO 50 J=1,LOOP
        MLOOP=MLOOP+1
        BDATA(K,MLOOP)=ZFNR*COSDTA(MLOOP)+ZFNI*SINDTA(MLOOP)
C
        IF (IREAL.EQ.1)GO TO 50
        BDATA(K+N,MLOOP)=ZFNR*SINDTA(MLOOP)-ZFNI*COSDTA(MLOOP)
C
   50   CONTINUE
   40   CONTINUE
   30   CONTINUE
C
```

```
C       ************************
C       END OF SUBROUTINE ZFNSET
C       ************************
C
        RETURN
        END


$ TYPE ZRESID.FOR


C       ***********************************************************
C       SUBROUTINE ZRESID COMPUTES THE COMPLEX RESIDUAL AT EACH
C       OF THE DATA POINTS DIRECTLY FROM THE DEFINITION.
C       ***********************************************************
C
C       *************************************************************
C       RESIDR(I) = REAL PART OF RESIDUAL AT THE I-TH DATA POINT
C       RESIDI(I) = IMAG PART OF RESIDUAL AT THE I-TH DATA POINT
C       N         = NUMBER OF BASIS FUNCTIONS
C       M         = NUMBER OF DATA POINTS
C       COEF(I)   = REAL PART OF THE I-TH COEFFICIENT, I=1,2,...,N
C       COEF(N+I) = IMAG PART OF THE I-TH COEFFICIENT, I=1,2,...,N
C       ZRDATA(I) = REAL PART OF THE I-TH DATA POINT , I=1,2,...,M
C       ZIDATA(I) = IMAG PART OF THE I-TH DATA POINT , I=1,2,...,M
C       CHEBER    = MAXIMUM MAGNITUDE RESIDUAL (CHEBYSHEV ERROR)
C       *************************************************************
C
        SUBROUTINE ZRESID(RESIDR,RESIDI,N,M,COEF,ZRDATA,ZIDATA,
     1                    CHEBER)
C
        DOUBLE PRECISION RESIDR(1),RESIDI(1),ZRDATA(1),ZIDATA(1),
     1                   ZR,ZI,ZFNR,ZFNI,CHEBER,RESR,RESI
        DIMENSION COEF(1)
C
        CHEBER=-1.D0
        DO 10 I=1,M
        ZR=ZRDATA(I)
        ZI=ZIDATA(I)
C
        CALL ZFUNCT(ZFNR,ZFNI,ZR,ZI)
C
        RESR=ZFNR
        RESI=ZFNI
        ZR=ZRDATA(I)
        ZI=ZIDATA(I)
        DO 20 J=1,N
        IB=J
C
        CALL ZBASIS(IB,ZFNR,ZFNI,ZR,ZI)
C
        COEFR=COEF(J)
        COEFI=COEF(N+J)
        RESR=RESR-(COEFR*ZFNR-COEFI*ZFNI)
        RESI=RESI-(COEFR*ZFNI+COEFI*ZFNR)
   20   CONTINUE
        RESIDR(I)=RESR
```

```
          RESIDI(I)=RESI
C
          ZR=SQRT(RESR*RESR+RESI*RESI)
          IF(ZR.GT.CHEBER)CHEBER=ZR
   10     CONTINUE
C
C
C         ************************
C         END OF SUBROUTINE ZRESID
C         ************************
C
          RETURN
          END


$ TYPE ACM495.FOR


C         **********************************************************
C         SUBROUTINE ACM495 SOLVES THE REAL OVER-DETERMINED SYSTEM OF
C         M LINEAR EQUATIONS IN N UNKNOWNS
C
C                 TRANSPOSE(A) * X = B
C
C         IN THE CHEBYSHEV NORM; THAT IS, THE MAXIMUM MAGNITUDE
C         COMPONENT OF THE RESIDUAL VECTOR [  = B-TRANSPOSE(A)*X    ]
C         IS MINIMIZED OVER ALL CHOICES OF THE UNKNOWN VECTOR X.
C
C         ONE SIDED CHEBYSHEV SOLUTIONS CAN ALSO BE COMPUTED; THAT IS,
C         THE MAXIMUM COMPONENT OF THE RESIDUAL VECTOR IS MINIMIZED
C         SUBJECT TO THE REQUIREMENT THAT THE RESIDUAL VECTOR BE
C         NON-NEGATIVE. [SEE REF. 2,PAGE 863.]
C         **********************************************************
C
C         ************************************************************
C         REFERENCE:
C         1. I. BARRODALE AND C. PHILLIPS, "SOLUTION OF AN OVERDETERMINED
C            SYSTEM OF LINEAR EQUATIONS IN THE CHEBYSHEV NORM," ACM
C            TRANS. ON MATH. SOFTWARE, VOL. 1, NO. 3, SEPTEMBER 1975,
C            PP. 264-270.
C
C         2. I. BARRODALE, L. M. DELVES, AND J. C. MASON, "LINEAR
C            CHEBYSHEV APPROXIMATION OF COMPLEX-VALUED FUNCTIONS,"
C            MATH. COMP., VOL. 32, NO. 143, JULY 1978, PP. 853-863.
C         ************************************************************
C
          SUBROUTINE ACM495(M,N,MDIM,NDIM,A,B,TOL,RELERR,X,RANK,
     1                      RESMAX,ITER,OCODE,NSIDES)
C
C         **********************************************************
C         M = NUMBER OF EQUATIONS.
C         N = NUMBER OF UNKNOWNS. (N MUST NOT EXCEED M)
C         MDIM = NUMBER OF COLUMNS OF A. (MDIM.GE.M+1)
C         NDIM = NUMBER OF ROWS    OF A. (NDIM.GE.N+3)
C         A = TWO DIMENSIONAL REAL ARRAY DIMENSIONED A(NDIM,MDIM).
C             ON ENTRY, THE TRANSPOSE OF THE MATRIX OF COEFFICIENTS
C             OF THE OVER-DETERMINED SYSTEM MUST BE STORED IN THE
```

```
C            FIRST M COLUMNS AND N ROWS OF A. THESE VALUES ARE
C            DESTROYED BY THIS SUBROUTINE.
C        B = ONE DIMENSIONAL REAL ARRAY OF LENGTH MDIM.
C            ON ENTRY, B CONTAINS THE RIGHT HAND SIDES OF THE
C            M EQUATIONS OF THE OVER-DETERMINED SYSTEM IN ITS FIRST
C            M LOCATIONS. ON EXIT, B CONTAINS THE RESIDUALS FOR THE
C            EQUATIONS IN ITS FIRST M LOCATIONS. (SEE NSIDES.) THESE
C            RESIDUALS ARE NOT COMPUTED DIRECTLY FROM THE DEFINITION.
C        TOL = A SMALL POSITIVE TOLERANCE, TYPICALLY THE UNIT
C              ROUND-OFF ERROR OF THE COMPUTER.
C        RELERR = A REAL VARIABLE WHICH ON ENTRY MUST EQUAL 0.
C                 IF A CHEBYSHEV SOLUTION IS REQUIRED. IF RELERR
C                 IS POSITIVE, THIS SUBROUTINE CALCULATES AN
C                 APPROXIMATE SOLUTION WITH RELERR AS AN UPPER
C                 BOUND ON THE RELATIVE ERROR ON ITS LARGEST
C                 RESIDUAL. ON EXIT, RELERR GIVES A SMALLER UPPER
C                 BOUND FOR THIS RELATIVE ERROR. [SEE REF. 1.]
C        X = ONE DIMENSIONAL REAL ARRAY OF LENGTH NDIM.
C            ON EXIT, X CONTAINS A SOLUTION TO THE PROBLEM IN THE
C            FIRST N LOCATIONS. THE CONTENTS OF X(N+1),....,X(NDIM)
C            ARE UNCHANGED.
C        RANK = AN INTEGER WHICH GIVES ON EXIT THE RANK OF THE
C               COEFFICIENT MATRIX. (WILL DEPEND ON TOL.)
C        RESMAX = ON EXIT, THE LARGEST MAGNITUDE RESIDUAL.
C        OCODE = 0 IF OPTIMAL SOLUTION IS PROBABLY NONUNIQUE. A DEFINITE
C                     STATEMENT REQUIRES FURTHER COMPUTATION WHICH IS NOT
C                     DEEMED TO BE COST EFFECTIVE.
C              = 1 IF UNIQUE OPTIMAL SOLUTION.
C              = 2 IF CALCULATIONS TERMINATED PREMATURELY DUE TO
C                     ACCUMULATED ROUND-OFF ERRORS.
C        NSIDES = 1 IF ONE SIDED CHEBYSHEV SOLUTION IS COMPUTED.
C                      IN THIS CASE THE RESIDUALS RETURNED FROM THIS
C                      PROGRAM ARE ERRONEOUS AND MUST BE COMPUTED IN
C                      THE CALLING ROUTINE. [SEE REF. 2.]
C               = 2 IF TWO SIDED CHEBYSHEV APPROXIMATION IS COMPUTED.
C                      THIS IS THE STANDARD FORM. THE RESIDUALS RETURNED
C                      FROM THIS PROGRAM ARE CORRECT IN THIS CASE. MORE
C                      NUMERICAL ACCURACY IN THE RESIDUALS MAY RESULT
C                      FROM DIRECT CALCULATION IN THE CALLING PROGRAM.
C        ****************************************************************
C
         DIMENSION A(NDIM,MDIM),B(MDIM),X(NDIM)
         INTEGER PROW,PCOL,RANK,RANKP1,OCODE
C
C        THE FOLLOWING NUMBER IS MACHINE DEPENDENT.
         DATA BIG/1.E+38/
C
C        **************
C        INITIALIZATION
C        **************
C
         IF(NSIDES.LE.1)SIDES=1.
         IF(NSIDES.GE.2)SIDES=2.
         MP1=M+1
         NP1=N+1
         NP2=N+2
         NP3=N+3
         NP1MR=1
         RANK=N
```

```
      RELTMP=RELERR
      RELERR=0.
      DO 10 J=1,M
      A(NP1,J)=1.
      A(NP2,J)=-B(J)
      A(NP3,J)=N+J
   10 CONTINUE
      A(NP1,MP1)=0.
      ITER=0
      OCODE=1
      DO 20 I=1,N
      X(I)=0.
      A(I,MP1)=I
   20 CONTINUE
C
C     *******
C     LEVEL 1
C     *******
C
      LEV=1
      K=0
   30 K=K+1
C
C     THE NEXT VARIABLE IS NOT USED, BUT IS IN THE PUBLISHED CODE.
      KP1=K+1
C
      NP1MK=NP1-K
      MODE=0
      DO 40 J=K,M
      B(J)=1.
   40 CONTINUE
C
C     DETERMINE THE VECTOR TO ENTER THE BASIS
C
   50 D=-BIG
      DO 60 J=K,M
      IF(B(J).EQ.0.)GO TO 60
      DD=ABS(A(NP2,J))
      IF(DD.LE.D)GO TO 60
      PCOL=J
      D=DD
   60 CONTINUE
      IF(K.GT.1)GO TO 70
C
C     TEST FOR ZERO RIGHT HAND SIDE
C
      IF(D.GT.TOL)GO TO 70
      RESMAX=0.
      MODE=2
      GO TO 380
C
C     DETERMINE THE VECTOR TO LEAVE THE BASIS
C
   70 D=TOL
      DO 80 I=1,NP1MK
      DD=ABS(A(I,PCOL))
      IF(DD.LE.D)GO TO 80
      PROW=I
      D=DD
```

```
   80   CONTINUE
        IF(D.GT.TOL)GO TO 330
C
C       CHECK FOR LINEAR DEPENDENCE IN LEVEL 1
C
        B(PCOL)=0.
        IF(MODE.EQ.1)GO TO 50
        DO 100 J=K,M
        IF(B(J).EQ.0.)GO TO 100
        DO 90 I=1,NP1MK
        IF(ABS(A(I,J)).LE.TOL)GO TO 90
        MODE=1
        GO TO 50
   90   CONTINUE
  100   CONTINUE
        RANK=K-1
        NP1MR=NP1-RANK
        OCODE=0
        GO TO 160
  110   IF(PCOL.EQ.K)GO TO 130
C
C       INTERCHANGE COLUMNS IN LEVEL 1
C
        DO 120 I=1,NP3
        D=A(I,PCOL)
        A(I,PCOL)=A(I,K)
        A(I,K)=D
  120   CONTINUE
  130   IF(PROW.EQ.NP1MK)GO TO 150
C
C       INTERCHANGE ROWS IN LEVEL 1
C
        DO 140 J=1,MP1
        D=A(PROW,J)
        A(PROW,J)=A(NP1MK,J)
        A(NP1MK,J)=D
  140   CONTINUE
  150   IF(K.LT.N)GO TO 30
  160   IF(RANK.EQ.M)GO TO 380
        RANKP1=RANK+1
C
C       *******
C       LEVEL 2
C       *******
C
        LEV=2
C
C       DETERMINE THE VECTOR TO ENTER THE BASIS
C
        D=TOL
        DO 170 J=RANKP1,M
        DD=ABS(A(NP2,J))
        IF(DD.LE.D)GO TO 170
        PCOL=J
        D=DD
  170   CONTINUE
C
C       COMPARE CHEBYSHEV ERROR WITH TOL
C
```

```
      IF(D.GT.TOL)GO TO 180
      RESMAX=0.
      MODE=3
      GO TO 380
  180 IF(A(NP2,PCOL).LT.-TOL)GO TO 200
      A(NP1,PCOL)=SIDES-A(NP1,PCOL)
      DO 190 I=NP1MR,NP3
      IF(I.EQ.NP1)GO TO 190
      A(I,PCOL)=-A(I,PCOL)
  190 CONTINUE
C
C     ARRANGE FOR ALL ENTRIES IN PIVOT COLUMN
C     (EXCEPT PIVOT) TO BE NEGATIVE
C
  200 DO 220 I=NP1MR,N
      IF(A(I,PCOL).LT.TOL)GO TO 220
      DO 210 J=1,M
      A(NP1,J)=A(NP1,J)+SIDES*A(I,J)
      A(I,J)=-A(I,J)
  210 CONTINUE
      A(I,MP1)=-A(I,MP1)
  220 CONTINUE
      PROW=NP1
      GO TO 330
  230 IF(RANKP1.EQ.M)GO TO 380
      IF(PCOL.EQ.M)GO TO 250
C
C     INTERCHANGE COLUMNS IN LEVEL 2
C
      DO 240 I=NP1MR,NP3
      D=A(I,PCOL)
      A(I,PCOL)=A(I,M)
      A(I,M)=D
  240 CONTINUE
  250 MM1=M-1
C
C     *******
C     LEVEL 3
C     *******
C
      LEV=3
C
C     DETERMINE THE VECTOR TO ENTER THE BASIS
C
  260 D=-TOL
      VAL=SIDES*A(NP2,M)
      DO 280 J=RANKP1,MM1
      IF(A(NP2,J).GE.D)GO TO 270
      PCOL=J
      D=A(NP2,J)
      MODE=0
      GO TO 280
  270 DD=VAL-A(NP2,J)
      IF(DD.GE.D)GO TO 280
      MODE=1
      PCOL=J
      D=DD
  280 CONTINUE
      IF(D.GE.-TOL)GO TO 380
```

```
      DD=-D/A(NP2,M)
      IF(DD.GE.RELTMP)GO TO 290
      RELERR=DD
      MODE=4
      GO TO 380
  290 IF(MODE.EQ.0)GO TO 310
      DO 300 I=NP1MR,NP1
      A(I,PCOL)=SIDES*A(I,M)-A(I,PCOL)
  300 CONTINUE
      A(NP2,PCOL)=D
      A(NP3,PCOL)=-A(NP3,PCOL)
C
C     DETERMINE THE VECTOR TO LEAVE THE BASIS
C
  310 D=BIG
      DO 320 I=NP1MR,NP1
      IF(A(I,PCOL).LE.TOL)GO TO 320
      DD=A(I,M)/A(I,PCOL)
      IF(DD.GE.D)GO TO 320
      PROW=I
      D=DD
  320 CONTINUE
      IF(D.LT.BIG)GO TO 330
      OCODE=2
      GO TO 380
C
C     PIVOT ON A(PROW,PCOL)
C
  330 PIVOT=A(PROW,PCOL)
      DO 340 J=1,M
      A(PROW,J)=A(PROW,J)/PIVOT
  340 CONTINUE
      DO 360 J=1,M
      IF(J.EQ.PCOL)GO TO 360
      D=A(PROW,J)
      DO 350 I=NP1MR,NP2
      IF(I.EQ.PROW)GO TO 350
      A(I,J)=A(I,J)-D*A(I,PCOL)
  350 CONTINUE
  360 CONTINUE
      TPIVOT=-PIVOT
      DO 370 I=NP1MR,NP2
      A(I,PCOL)=A(I,PCOL)/TPIVOT
  370 CONTINUE
      A(PROW,PCOL)=1./PIVOT
      D=A(PROW,MP1)
      A(PROW,MP1)=A(NP3,PCOL)
      A(NP3,PCOL)=D
      ITER=ITER+1
      GO TO (110,230,260),LEV
C
C     **************
C     PREPARE OUTPUT
C     **************
C
  380 DO 390 J=1,M
      B(J)=0.
  390 CONTINUE
      IF(MODE.EQ.2)GO TO 450
```

```
      DO 400 J=1,RANK
      K=A(NP3,J)
      X(K)=A(NP2,J)
  400 CONTINUE
      IF(MODE.EQ.3.OR.RANK.EQ.M)GO TO 450
      DO 410 I=NP1MR,NP1
      K=ABS(A(I,MP1))-FLOAT(N)
      B(K)=A(NP2,M)*SIGN(1.,A(I,MP1))
  410 CONTINUE
      IF(RANKP1.EQ.M)GO TO 430
      DO 420 J=RANKP1,MM1
      K=ABS(A(NP3,J))-FLOAT(N)
      B(K)=(A(NP2,M)-A(NP2,J))*SIGN(1.,A(NP3,J))
  420 CONTINUE
C
C     TEST FOR NON-UNIQUE SOLUTION
C
  430 DO 440 I=NP1MR,NP1
      IF(ABS(A(I,M)).GT.TOL)GO TO 440
      OCODE=0
      GO TO 450
  440 CONTINUE
  450 IF(MODE.NE.2.AND.MODE.NE.3)RESMAX=A(NP2,M)
      IF(RANK.EQ.M)RESMAX=0.
      IF(MODE.EQ.4)RESMAX=RESMAX-D
C
C     ***********************
C     END OF SUBROUTINE ACM495
C     ***********************
C
      RETURN
      END
```

INITIAL DISTRIBUTION LIST

Addressee	No. of Copies
ASN (RE&S)(D. E. Mann)	1
OUSDR&E (Res & Adv Tech)(W. J. Perry)	2
Deputy USDR&E (Res & Adv Tech)(R. M. Davis)	1
Deputy USDR&E (Dir Elec & Phys Sc)(L. Wiseberg)	1
OASN, Spec Dep for Adv Concept, Dep Assist Secretary (Res & Adv Tech)(Dr. R. Hoglund)	2
ONR, ONR-100, -200, -212, -222	4
CNO, OP-090, -098	2
CNM, MAT-08T2, SP-20	2
DEFENSE INTELLIGENCE AGENCY, DT-2C	1
NAV SURFACE WEAPONS CENTER, White Oak Laboratory	1
NAV SURFACE WEAPONS CENTER, Dahlgren, Code CF42 (Joseph Halberstein)	1
NAV SURFACE WEAPONS CENTER, Silver Spring, Code 432-4 (Egbert H. Jackson)	1
DWTNSRDC ANNA	1
DWTNSRDC CARD	1
NRL, Code 5330 (Dr. Robert J. Adams), 5209 (Russell M. Brown) 7944F (Frederick Fine), USRD, AESD	6
NORDA, Code 110 (Dr. R. Goodman), Code 340 (S. W. Marshall)	2
USOC, Code 240, 241	2
NAVOCEANO, Code 02, 6200	2
NAVOCEANO, SAN DIEGO, Code 712 (J. Brown, F. J. Harris)	2
NAVELECSYSCOM, ELEX 03, 310, PME-117	3
NAVSEASYSCOM, SEA-003, -63R-1, -92R, -996	4
NASC, AIR-610	1
NAVAIRDEVCEN, Code 2052, 2041 (Herbert Heffner)	3
NOSC, Dr. George Bentien, John M. Horn, Library, Code 6565	4
NAVWPNSCEN, China Lake, Code 3556 (John A. Downs), 5013 (Gaylon E. Ryno)	3
DTNSRDC	1
NAVCOASTSYSLAB	1
CIVENGRLAB	1
NAVSURFWPNCEN	1
NUWES	1
NISC	1
NAVPGSCOL, Code 752AB (Dr. Richard W. Adler)	2
NAVTRAEQUIPCENT, Technical Library	1
INTER-SERVICE ANTENNA GROUP	1
APL/UW, SEATTLE	1
ARL/PENN STATE, STATE COLLEGE (Dr. Geoffrey Wilson, Dr. William Thompson)	3
DTIC	1
DARPA	1
NOAA/ERL	1
NATIONAL RESEARCH COUNCIL	1
WOODS HOLE OCEANOGRAPHIC INSTITUTION	1

INITIAL DISTRIBUTION LIST (Cont'd)

Addressee	No. of Copies
ENGINEERING SOCIETIES LIBRARY, UNITED ENGRING CTR	1
NATIONAL INSTITUTE OF HEALTH	1
ARL, UNIV OF TEXAS	1
MARINE PHYSICAL LAB, SCRIPPS	1
NASA Goddard Space Flight Center, Code 811 (Paul A. Lanz), 110 (Abe Kampinsky)	2
ONR, Eastern/Central Regional Office (Dr. F. W. Quelle)	1
Naval Ocean R&D Activity (Dr. D. J. Ramsdale)	1
U.S. Army Missile Command, Redstone Arsenal (Fran King)	1
Naval Air Test Center (James R. Searle)	1
Naval Ammunition Depot, Code 3083 (Joseph M. Smiddle)	1
National Security Agency, Code 3083 (A. A. Strejeck)	1
U.S. Army Electronic Command, AMSEL-WL-S (Gottried Vogt), NL-H7 (Felix Schwering), VL-G (Sol Perlman), CT-R (Boaz Gelernter)	4
U.S. Department of Commerce, OT/ITS, (Dr. M. T. Ma, Richard G. Fitzgerrel)	2
USA Satellite Comm. Agency (George T. Gobeaud)	1
Pacific Missile Test Center, Code 1171 (Cyril M. Kaloi)	1
USA Foreign Science & Technology Center, AMXST-SR-Z (James P. Muro)	1
Federal Aviation Agency (Martin Natchipolsky)	1
Rome Air Development Center, Code OCTS (C. L. Pankiewicz)	1
U.S. Information Agency, Code IBS/ET (Julius Ross)	1
Syracuse University, Electrical & Computer Eng. Dept. (Dr. David K. Cheng)	1
The Cleveland State University, Computer & Information Science (Dr. Allan D. Waren)	1
Old Dominion University, Math Dept. (Prof. Harold Ladd, Jr.)	1
Westinghouse Oceanic Division, Annapolis (Dr. Bill Kesner)	1
Arthur D. Little, Inc., Cambridge (Dr. Gordon Raisbeck)	1
Gould Oceans Systems Division-761, Cleveland (Steven C. Thompson)	1
East Tennessee State Univ., Math. Dept. (Dr. James C. Pleasant)	1
University of Maryland, Dept. Physics and Astronomy (Prof. Douglas Currie)	1
University of Victoria, Computer Science Dept., Canada (Dr. Ian Barrodale)	1
URI, Kingston, Math. Dept. (Dr. J. T. Lewis), Dept. of Eng. (Dr. D. W. Tufts)	2
University of Pennsylvania, Moore School of Electrical Engineering (Dr. Bernard Steinberg)	1
General Motors Corporation, Sea Operations Dept. Goleta (Dr. Daniel Suchman)	1
Dr. R. C. Hansen, Box 215, Tarzana, CA	1
Southern Illinois University, Math. Dept. (Dr. David W. Kammler)	1
Brooks Air Force Base, School of Aerospace Medicine, TX (Stewart J. Allen)	1
National Bureau of Standards (Dr. R. C. Baird)	1
Harry Diamond Laboratories, Branch 150 (Whilden G. Heinard)	1
Transportation Systems Center, Cambridge (Dr. Rudy Kalafus)	1

LMED

8