Testing Stability of 2-D Discrete Systems by a Set of Real 1-D Stability Tests

Yuval Bistritz, *Fellow, IEEE*

***Abstract*—Stability of a two-dimensional (2-D) discrete system depends on whether a bivariate polynomial does not vanish in the closed exterior of the unit bi-circle. The paper shows a procedure that tests this 2-D stability condition by testing the stability of a finite collection of real univariate polynomials by a certain modified form of the author's one-dimensional (1-D) stability test. The new procedure is obtained by telepolation (interpolation) of a 2-D tabular test whose derivation was confined to using a real form of the underlying 1-D stability test. Consequently, unlike previous telepolation-based tests, the procedure requires the testing of *real* instead of *complex* univariate polynomials. The proposed test is the least-cost procedure to test 2-D stability with real polynomial 1-D stability tests and real arithmetic only.**

***Index Terms*—Discrete-time systems, immittance algorithms, multidimensional digital filters, polynomials, stability tests, two-dimensional (2-D) systems.**

I. Introduction

THIS PAPER will present a new procedure that tests whether a bivariate polynomial

$$D(z_1, z_2) = \sum_{i=0}^{n_1}\sum_{k=0}^{n_2} d_{i,k} z_1^i z_2^k \tag{1}$$

has no zeros in the closed exterior of the unit bi-circle, viz.,

$$D(z_1, z_2) \neq 0 \qquad \forall (z_1, z_2) \in \bar{V} \times \bar{V} \tag{2}$$

where $\bar{V} = V \cup T$ with $T = \{z : |z| = 1\}$ and $V = \{z : |z| > 1\}$. A polynomial like (1) for which the condition (2) holds will be called **stable in the sense of two-dimensional discrete system** (2-D stable). The relation of this definition to stability of 2-D discrete systems and more background on the topic can be found in various texts, e.g., [1]–[4].

A univariate polynomial $p_n(z)$ that has all its zeros in U, or equivalently

$$p_n(z) = \sum_{i=0}^{n} p_i z^i \neq 0 \quad \forall z \in \bar{V} \tag{3}$$

will **stable in the sense of one-dimensional discrete system** (1-D stable) stable for its well known relation to stability conditions of a 1-D discrete systems.

The new procedure will test the condition (2) by testing stability of a collection of real univariate polynomials using a certain modified form of the author's 1-D stability test [5], [6] plus testing for no zeros on $[-1, 1]$ a real polynomial (or, and preferably, a related real symmetric polynomial for no zeros on T) obtained by interpolation of values that the collection of 1-D stability tests provide. The procedure will be derived by invoking a recently introduced approach called telepolation [7] to the tabular test in [8]. The new procedure is apparently the most efficient stability test for 2-D discrete systems that uses only *real* arithmetic and the manipulation of only *real* polynomials.

Virtually all the techniques that were proposed to test the condition (2) use (or rely in a less direct manner on) its equivalence to

$$D(s, z) \neq 0 \qquad \forall (s, z) \in T \times \bar{V} \tag{4}$$

plus a 1-D stability condition posed on $D(z, a)$ for some fixed $a, \ , |a| > 1$. This simplification was introduced by Huang [9] (with $a = \infty$) and variations on it were obtained afterwards also by several other researchers, see accounts in [1] and [3]. The condition (4) may be tested by a 1-D stability test for complex coefficient univariate polynomial that regards $D(s, z)$ as a univariate polynomial in z with coefficients that are polynomials of $s \in T$. This approach was used in the first 2-D stability test proposed in [10] and in many subsequent tests including [11]–[14].

In the early years when 2-D stability tests started to emerge, 1-D stability conditions for complex-coefficient polynomials were not as widely known to engineers as real polynomial tests. Bose suggested in [15] (see also [2]) a technique that allows development of 2-D stability testing procedures from stability conditions posed on real univariate polynomial. Accordingly, $D(s, z)$ is multiplied by $D(s^{-1}, z)$, that is its complex-conjugated coefficient polynomial when regraded as a univariate polynomial in z with coefficients dependent on $s \in T$. The product $D(s, z)D(s^{-1}, z)$ can then be converted, using the transformation

$$x = \frac{s + s^{-1}}{2} \tag{5}$$

into another polynomial, say $H(x, z)$ whose degree is $(n_1, 2n_2)$ (instructions on how to do this will appear in Section II). Since (5) maps the unit-circle T to the real interval $[-1, 1]$, condition (4) is replaced by

$$H(x, z) \neq 0 \qquad \forall (x, z) \in [-1, 1] \times \bar{V}. \tag{6}$$

The condition (6) can be tested by a 1-D stability test for a real polynomial, regarding $H(x, z)$ as a univariate polynomial in z with real coefficients that are polynomials in $x \in [-1, 1]$. It is notable that this technique doubles the degree in z of the polynomial that has to be tested. Hence, we refer to it as the "dou-

Manuscript received April 10, 2002; revised January 13, 2003. This paper was recommended by Associate Editor O. Au.

The author is with the Department of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel (e-mail: bistritz@eng.tau.ac.il).

Digital Object Identifier 10.1109/TCSI.2004.830679

bling-degree" technique. The doubling-degree technique underlies the method in this paper. It was used in the past to develop several 2-D stability tests including [16]–[18] and, more pertinent to the work here [8], [19]–[21]. The technique can be shown to always lead to a more costly 2-D stability tests than attainable by using instead (4) with a corresponding complex 1-D stability test. As a consequence, it produced so far 2-D stability tests with a futile increase in cost of computation. The reason is that, contrary to what might be assumed, tabular tests that were developed from a complex 1-D stability test are too carried out by manipulation of only real arrays and with only real arithmetic for a real $D(z_1, z_2)$. This paper reveals, for the first time, that the use of the doubling-degree technique may acquire the produced 2-D stability test with an added value that can not be obtained without using it. It will present a 2-D stability test that is carried by testing of a set of real univariate polynomials and uses throughout only real arithmetic.

The testing of the condition (4) or (6) by a 1-D stability test leads to "tabular" 2-D stability tests. The term "tabular" reflects a historical tradition to present recursive 1-D stability tests by an array of numbers known as "stability table." A tabular 2-D stability test is therefore any procedure that uses a recursion of bivariate polynomials where the "2-D table" is representable by either the produced sequence of bivariate polynomial, or by the sequence of matrices that are the coefficients of the bivariate polynomials, or by a hybrid of the above two—an array of univariate polynomials. The first generation of 2-D tabular tests used to have a cost of computation that increases exponentially with n (for say $n_1 = n_2 = n$, the cost figures can be shown to contain terms as severe as $n^2 4^n$), limiting their usefulness to only very low degree polynomials. Tests that belong to this category include the first 2-D stability test of Maria–Fahmy [10] and tests that were proposed for many years afterwards, e.g., [11], [16], [17], [19], [20]. The second generation of tabular tests, those in [12]–[14], stepped down from the exponential complexity of the first generation tests to moderate polynomial $O(n^6)$ complexity (again for $n_1 = n_2 = n$), where the fact that [12] has too $O(n^6)$ complexity was established for it in [22]. The second valuable property shared by the second generation of tabular tests is that essentially (that is, aside from a couple of preliminary simple 1-D stability tests of polynomials of degree n), upon completing the "2-D table," there remains to test only the last polynomial of the table—a univariate (symmetric) polynomial—for zeros on T. The latter condition is at times also called 'positivity' test, because the condition of no zeros on T can also be expressed as positivity on T of the 'balanced' polynomial that corresponds to the symmetric polynomial.

The procedures in [7], [22], [23] relies on "telepolation" of the tabular tests in [12]–[14], respectively. Telepolation presents a new approach that takes advantage of the above mentioned special features of second generation tabular tests. A finite set of 1-D stability tests are used to bring forth (to "**tele**scope") by inter**polation** the last polynomial of the 2-D table. The results are 2-D stability tests that are simple (involve repetitive use of a same form 1-D stability test) and attain a lower complexity of $O(n^4)$. The reduction from $O(n^6)$ to $O(n^4)$ occurs because the telepolation eliminates the computation of the 2-D table that turns out to be responsible for the $O(n^6)$ cost tag of the second generation tabular tests. These telepolation based tests constitute apparently the most efficient procedures (i.e. with least counts of real arithmetic operations) available to test 2-D stability (topped with the immittance-type tests [7], [23] that exceed by a factor of approximately 2 the scattering-type test in [22]).

The fact that it is possible to test 2-D stability by a finite number of 1-D stability tests is in itself a new observation that was discovered via the telepolation approach and demonstrated so far in [7], [22], [23] (its debut was in [24]). It is important to distinguish the above statement from similarly sounding statements that occur in some of the previous papers on 2-D stability tests. Previously, declaration on that the 2-D stability test that is being proposed requires the examination of a small number of univariate polynomials used to refer to the number of univariate polynomials in the set of necessary and sufficient conditions for 2-D stability (as stability or positivity conditions). It was obviously assumed that additional means (like construction of a 2-D table) are required to produce this set of polynomials before the final set of conditions can be examined. (Incidentally, it turns out that a higher order of complexity of these additional means diminishes the difference between testing one or several positivity conditions.) In contrast, the 1-D stability conditions in telepolation-based procedures are not just necessary *conditions* for 2-D stability but their testing is, to the most part, *the algorithm* to determine 2-D stability. It is worth mentioning that papers proposing 2-D stability tests used to appear without evaluation of the overall computational cost of the proposed technique they propose, for at least two decades after the first paper [10] on this topic. This situation hindered the development of efficient 2-D stability tests because it hid the fact that the bulk part of the computation lied not in testing the univariate polynomials that are present in the set of necessary and sufficient conditions for stability, but in the algorithm required to produce these univariate polynomials.

The new procedure in this paper stems from the 2-D stability tabular test in [8] via telepolation. The tabular test in [8] is too a second generation tabular stability test. Namely, it features a single positivity condition and $O(n^6)$ complexity. It differs from the other tabular tests to which telepolation was applied so far in that it was confined to use the real form of the Bistritz test (BT) [5]. The 2-D stability test in [7] and in [23] stem from 2-D tabular tests that were developed from the complex 1-D BT in [25] and in [26], respectively, while the one in [22] telepolates a 2-D tabular test that stems from the 1-D modified Jury test (MJT) in [27] for complex univariate polynomials. As a consequence, the telepolation of the test in [8] leads, as will be seen, to a collection of *real*-coefficient polynomial 1-D stability tests instead of a collection of *complex*-coefficient polynomial 1-D stability tests in [7], [22], [23]. Computational cost assessment for the proposed test will show that it requires four times more computation than the immittance-type tests in [7] or [23] (that are the least cost available 2-D stability tests) and two times more than the scattering-type test in [22]. With no other known test that competes with the efficiency of the test to be presented here, the test in this paper is considered to be the most efficient stability test for 2-D discrete systems that uses only *real* arithmetic and the manipulation of only *real* polynomials.

The tabular test in [8] has been preceded by some other works that too used the doubling-degree technique in conjunction with the real BT to obtain 2-D stability tests. Karan and Srivastava in [19], [20] proposed a 2-D stability test with several positivity tests (they actually used not [5] but the form in [28] that has some interesting properties in other contexts [29] but offers a less pleasant form for developing 2-D stability tests). Premaratne in [21] (switching to [5]) showed for the scheme the single positivity testing condition and some other improvements. However, both procedures are first generation (i.e. exponential order of complexity) tabular tests, as was shown in [8].

This paper is constructed as follows. Section II summarizes the tabular stability test in [8]. Section III first brings the companion 1-D stability test for real polynomials that can be used to sample this tabular 2-D table on $[-1,1]$. Next, an efficient algorithm is derived to recover the polynomial, whose positivity over $[-1,1]$ implies 2-D stability, from a certain choice of sample values. The derivation of the recovering algorithm reveals that further saving in the overall cost of computation is possible by testing instead an intermediate polynomial that it produces for no zeros on T. Section IV summarizes the new procedure, illustrates it by a numerical example and evaluates its cost of computation. Some concluding remarks end the paper.

II. Real Immittance-Type Tabular 2-D Stability Test

Following a brief introduction of notation, this section brings a summary of the tabular 2-D stability test in [8].

A. Notation

All polynomials in the rest of the paper are assumed to have real (matrix or vector) coefficients. The variable z may take any complex number while the variables x and s will usually assume values on the real interval $[-1,1]$ and on the unit-circle T, respectively. A polynomial like (1) is said to have degree (n_1, n_2). It may also be written as $D(z_1,z_2) = \mathbf{z_1}^t D \mathbf{z_2}$, where $\mathbf{z} := [1, z, \ldots, z^i, \ldots]^t$ of length depending on context, and D is also used to denote its coefficient matrix. Similarly, for a univariate polynomial $p(z)$, p is also used to denote its coefficient vector, and it may be written also as $p(z) = \mathbf{z}^t p$. Reversion is defined for a column vector by $p^\sharp := Jp$ where J is the reversion matrix (defined as a square matrix with 1's on its main anti-diagonal and zeros elsewhere), and correspondingly $p^\sharp(z) = \mathbf{z}^t p^\sharp (= z^n p(z^{-1}))$. A vector and a polynomial are called symmetric if $p = p^\sharp$ and $p(s) = p^\sharp(s)$. Reversion of columns (only!) for a matrix will be denoted by $D^\flat = DJ$. A matrix F and its corresponding bivariate polynomial $F(x,z)$ are said to posses column-symmetry if $F^\flat = F$. Indexes are attached to matrices and vectors to sign their position in a sequence, e.g., $F_m(x,z) = \mathbf{x}^t F_m \mathbf{z}$, $m = 0,\ldots,n$. Reference to columns of an indexed matrix carries its identifying index in brackets,, e.g., $F_m = [f_{[m]\ 0}, f_{[m]\ 1}, \ldots]$. Indexes are also used to denote order of 1-D polynomials in a sequence, $f_m(z) = \mathbf{z}^t f_m = \sum_{i=0}^{n-m} f_{m,i} z^i$, $m = 0,\ldots,n$ (in this case they are related to their degrees as indicated). Finally, the "balanced polynomial" is defined for a polynomial $p(s) = [1, s, \ldots, s^n]p$ by $p(\tilde{s}) = s^{-n/2} p(s)$ and it may also be written as $\tilde{\mathbf{s}}^t p$, where $\tilde{\mathbf{s}} = [s^{-n/2}, s^{-(n-1)/2}, \ldots, s^{(n-1)/2}, s^{n/2}]^t$.

B. 2-D Tabular Stability Test

The tabular 2-D stability test in [8] begins with the conversion of $D(z_1,z_2)$ into $H(x,z)$ (4)–(6). First one forms

$$\tilde{\mathbf{s}}^t Q \mathbf{z} := D(s^{-1}, z) D(s,z) = \sum_{k=0}^{2n_2} q_k(\tilde{s}) z^k. \tag{7}$$

Denote the columns of D and Q by

$$D = [d_1, d_2, \ldots, d_{n_1}] \quad Q = [q_0, \ldots, q_{2n_2}].$$

The conversion (7) involves the following convolutions:

$$q_k = \sum_{i=0}^{n_2} d_i * d_{k-i}^\sharp = \sum_{i=\max(0,k-n_2)}^{\min(k,n_2)} d_i * d_{k-i}^\sharp, \quad k = 0,\ldots,2n_2. \tag{8}$$

$Q(s,z)$ is a polynomial of degree $(2n_1, 2n_2)$ that may also be regarded as a 1-D polynomial in z with coefficients $q_k(s)$ that are symmetric polynomials for all $m = 0,\ldots,2n_2$. Next, Q is converted into H such that

$$H(x,z) = Q(\tilde{s}, z)_{|x=\frac{1}{2}(s+s^{-1})}. \tag{9}$$

As shown in [8], this can be done by pre-multiplying the last n_1+1 rows of Q, denoted by $Q_{(n_1:2n_1, 0:2n_2)}$, as follows:

$$H = B_{n_1} \acute{I}_{n_1} Q_{(n_1:2n_1, 0:2n_2)} \tag{10}$$

where $\acute{I}_{n_1}$ is an (n_1+1)th sized diagonal matrix $\mathrm{diag}[1,2,\ldots,2]$ and B_{n_1} is an (n_1+1)th sized square "Chebyshev matrix" defined as an upper triangular matrix

$$B_M = \left[\begin{array}{c|c|c|c|c} b_0 & b_1 & \cdots & b_{M-1} & b_M \\ 0_M & 0_{M-1} & & 0 & \end{array}\right] \tag{11}$$

whose columns consist of the coefficient vectors b_m of the (first kind) Chebyshev polynomials $T_m(x) = \mathbf{x}^t b_m$, $m = 0,\ldots M$, augmented to length $M+1$ by 0_m, a vector of zeros of length m. The b_m's obey the recursion

$$b_{m+1} = 2\begin{bmatrix} 0 \\ b_m \end{bmatrix} - \begin{bmatrix} b_{m-1} \\ 0 \end{bmatrix} \tag{12}$$

that can be run for $m = 1,\ldots,M-1$ with the initiation $b_0 = 1$ and $b_1 = [0,1]^t$ to obtain them.

Next, the construction of the 2-D stability table follows the next algorithm.

Algorithm 1 : **(The 2-D Real Table)** Given $D(z_1,z_2)$, convert it to $H(x,z)$ using (7)–(12) and then assign to it a sequence

$$R_m(x,z) = \sum_{k=0}^{n_r-m} r_{[m]\ k}(x) z^k, \qquad m = 0,\ldots,n_r := 2n_2 \tag{13}$$

of column-symmetric 2-D polynomials $R_m = R_m^\flat$ (i.e. $r_{[m]\ k}(x) = r_{[m]\ n_r-m-k}(x)$ $k = 0,\ldots,n_r-m$), as follows.

Initiation.

$$R_0(x,z) = H(x,z) + H^\flat(x,z) \tag{14a}$$

$$R_1(x,z) = \frac{H(x,z) - H^\flat(x,z)}{z-1}. \tag{14b}$$

Body. For $m = 0, 1, \ldots, n_r - 2$

$$zR_{m+2}(x,z) = \frac{r_{[m]\ 0}(x)(z+1)R_{m+1}(x,z) - r_{[m+1]\ 0}(x)R_m(x,z)}{\eta_{m-1}(x)} \tag{15}$$

where $\eta_0(x) = 2$, $\eta_1(x)$ and

$$\eta_m(x) = r_{[m-1]\ 0}(x) \quad \text{for} \quad m \geq 2. \tag{16}$$

The degrees of the polynomials $R_m(x,z)$ are $(n_1, 2n_2)$ for $m = 0$ and $(mn_1, 2n_2 - m)$ for $m = 1, \ldots, 2n_2$. Finally, the stability conditions in the following theorem are shown in [8] for the above table.

Theorem 1 (Stability Conditions for Algorithm 1.): Assume Algorithm 1 produces for $D(z_1, z_2)$ the sequence $\{R_m(x,z)\}_0^n$. The following conditions (i), (ii) and (iii) are necessary and sufficient for $D(z_1, z_2)$ to be stable.

$$(i) \quad D(z_1, 1) \neq 0 \qquad \forall_{z_1} \in \bar{V} \tag{17}$$

$$(ii) \quad D(1, z_2) \neq 0 \qquad \forall_{z_2} \in \bar{V} \tag{18}$$

$$(iii) \quad \rho(x) := R_{n_r}(x,z) \neq 0 \qquad \forall x \in [-1, 1] \tag{19}$$

where $R_{n_r}(x,z)$ is the last polynomial in the sequence that Algorithm 1 produces, a polynomial of degree $(M, 0)$, $M := 2n_1n_2$.

The condition (19) may be tested by Sturm's method as was suggested in [30]. It can be stated also as a positivity condition. Namely, condition (iii) in the theorem may be replaced by

$$\rho(x) > 0 \qquad \forall x \in [-1, 1] \tag{20}$$

because, if $\rho(x)$ has no zeros in $[-1, 1]$ and the other conditions in the theorem hold, then, $\rho(x)$ cannot be negative on this interval [8]. The condition (19) is stronger than (20) because it disregards the sign (this becomes in particular handy when the testing (19) goes through singularities).

The cost evaluation carried out in [8] shows that this 2-D stability test requires $(5/3)n_1^2n_2^4 + O(n_{1,2}^5)$ real multiplications, where $O(n_{1,2}^k)$ denotes a bivariate polynomial in n_1 and n_2 with power terms $n_1^{k_1}n_2^{k_2}$ such that $k_1 + k_2 \leq k$ (dropped for brevity). This cost is about two times the cost of the immittance-type tabular tests in [13], [14] (that too propagates matrices with symmetry), an increase that is caused by the doubling of the sequence length that the conversion (7)) introduces. The cost is comparable to the cost of the scattering-type tabular test in [12] as calculated in [22]. The reason for equality in this latter case is that the symmetry of the matrices R_m makes up for the doubling in length of the sequence when the comparison is held with the table in [12] that produces matrices with no symmetry.

III. Telepolation of the 2-D Tabular Test

From the point of view of the stability conditions in Theorem 1, the only role that Algorithm 1 serves is to obtain $\rho(x)$ at its end. Now, the conversion of $D(z_1, z_2)$ to $H(x,z)$ requires $O(n^4)$ operations. The testing of conditions (i) and (ii) require $O(n^2)$ operations. The testing of (iii) has too $O(n^4)$ of complexity. Thus, Algorithm 1 is solely responsible for the $O(n^6)$ overall complexity shown for the test in [8].

Being a polynomial of degree $M = 2n_1n_2$, $\rho(x)$ can be determined by a set of $M + 1$ values $b_m = \rho(x_m)$ for any set of different points x_m. Such sample values can be produced by an algorithm that presents the behavior of Algorithm 1 at a fixed real x that would have $O(n^2)$ complexity per sample. Thus, the complete set of sample values can be acquired in $O(n^4)$ operations. The recovery of $\rho(x)$ from sample values presents a standard interpolation problem for a polynomial of degree M known to have $O(M^2)$ solutions [31]. These considerations indicate that an $O(n^4)$ complexity solution can be worked out. This section provides the two key components to implement the anticipated scheme in a very neat and efficient manner. First, it brings a 1-D stability test that can sample $\rho(x)$ in the interval $[-1, 1]$. Then, a simple closed form interpolation formula that recovers the polynomial $\rho(x)$ from a specific choice of sampling points is derived.

A. *Companion 1-D Stability Test*

The following Algorithm 2 and Theorem 2 form a companion 1-D stability test usable the telepolation of the real 2-D tabular test in the previous section. Algorithm 2 and Theorem 2 are derived and proved in [32] (where they appear as Algorithm 3 and Theorem 7), a paper that presents further properties and applications for this 1-D stability test.

Algorithm 2: (**A Companion 1-D Algorithm**) Consider a real polynomial of degree n

$$p(z) = \sum_{k=0}^{n} p_k z^k \quad p(1) > 0, \qquad p_n \neq 0. \tag{21}$$

Assign to it a sequence of symmetric polynomials $r_m(z) = \sum_{k=0}^{n-m} r_{m,k}z^k$, $r_m(z)$ of degree $n - m$, and $r_m^\sharp(z) = r_m(z)$, $m = 0, \ldots, n$, as follows:

$$r_0(z) = p(z) + p^\sharp(z) \quad r_1(z) = \frac{p(z) - p^\sharp(z)}{(z-1)}. \tag{22}$$

Let $\eta_0 = 2$, $\eta_1 = 1$ and for $m = 1, \ldots, n-1$ do

$$zr_{m+1}(z) = \frac{r_{m-1,0}(z+1)r_m(z) - r_{m,0}r_{m-1}(z)}{\eta_m}, \qquad \eta_m = r_{m-2,0}. \tag{23}$$

*Theorem 2 (***Companion Stability Conditions***.):* The real polynomial $p_n(z)$ (21) is stable if, and only if, $r_{1,0} = p_n - p_0 > 0$ and the sequence that Algorithm 2 produces for it satisfies

$$r_m(1) > 0 \qquad m = 0, 1, \ldots, n. \tag{24}$$

A stable $p_n(z)$ implies that the next further conditions also hold

$$r_{m,0} > 0 \qquad m = 0, 1, \ldots, n. \tag{25}$$

The fact that conditions (25) are necessary for 1-D stability means that for a stable polynomial, all the symmetric polynomial $r_m(z)$ are normal, i.e., with no vanishing leading coefficient, $r_{m,n-m} = r_{m,0} \neq 0$. The normal conditions (25) provide a broad enough setting for the Algorithm 1 to be well defined for stability testing. Because, when an $r_{m,0} = 0$ occurs the algorithm can terminated at once (three steps before this division by this zero would arise) and the tested polynomial declared as not stable.

Our intention is to use this test for polynomials $p_{x_m}(z) := H(x_m, z)$ with a fixed $x_m \in [-1, 1]$ in the context of the conditions stated by Theorem 1 but instead of obtaining $\rho(x)$ by

Algorithm 1 to obtain it by interpolation. Let $s_m \in T$ correspond to x_m by the mapping (5). Then, using (9), (7), $p_{x_m}(z) = D(\tilde{s}_m^{-1}, z)D(\tilde{s}_m, z)$. Therefore, if $D(z_1, z_2)$ is 2-D stable then $D(s_m, z)$ is 1-D stable and hence $p_{x_m}(z)$ is 1-D stable. In other words, 1-D stability of $p_{x_m}(z)$ is a necessary condition for 2-D stability of $D(z_1, z_2)$. Note that for $z = 1$ one has $p_{x_m}(1) = D(\tilde{s}_m^{-1}, 1)D(\tilde{s}_m, 1) = |D(\tilde{s}_m, 1)|^2 \geq 0$. We intend to carry a collection of stability tests $p_{x_m}(z)$ only after $D(z, 1)$ is determined to be 1-D stable (i.e. after checking condition 1 in Theorem 1). Therefore, $D(s, 1) \neq 0 \, \forall s \in T$ and the requirement for $p_{x_m}(1) > 1$ in (21) will hold and need not be examined. Next, comparison of the two algorithms reveals that Algorithm 2 acts on $p_{x_m}(z) := H(x_m, z)$ in a manner that presents the projection of Algorithm 1 on $H(x, z)$ at any fixed real x_m. With these clarifications, the viability of the next mode of operation follows.

Theorem 3 (Sampling Properties of the Companion 1-D Stability Test.): Assume $H(x, z)$ is related to $D(z_1, z_2)$ via the conversion (7)–(9). Let $x_m \in [-1, 1]$ and consider $p_{x_m}(z) = H(x_m, z)$ (a univariate polynomial of degree $n_r := 2n_2$). Apply Algorithm 2 to $p_{x_m}(z)$ and use Theorem 2 to examine its stability. If $p_{x_m}(z)$ is not stable then $D(z_1, z_2)$ is not stable. If $p_{x_m}(z)$ is stable then Algorithm 2 ends normally and its last element $\gamma_m := r_{n_r}(z)$ (a univariate polynomial of degree 0) is equal to $\gamma_m = \rho(x_m)$, where $\rho(x) := R_{n_r}(x, z)$, a bivariate polynomial of degree $(2n_1n_2, 0)$, is the last polynomial that Algorithm 1 would produce for $D(z_1, z_2)$.

B. Interpolation Problem

The polynomial

$$\rho(x) = [1, x, \ldots, x^M][\rho_0, \ldots, \rho_M]^t \tag{26}$$

that appears in Theorem 1, like any polynomial of degree M, can be determined from knowing its values $\gamma_i = \rho(x_i)$ at any set of distinct $M+1$ points $X_{0:M} = \{x_0, \ldots, x_M\}$ by solving the set of equations,

$$\begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_M \end{bmatrix} = \begin{bmatrix} 1 & x_0 & \ldots & x_0^M \\ 1 & x_1 & \ldots & x_1^M \\ \vdots & \vdots & & \vdots \\ 1 & x_M & \ldots & x_M^M \end{bmatrix} \begin{bmatrix} \rho_0 \\ \rho_1 \\ \vdots \\ \rho_M \end{bmatrix}. \tag{27}$$

Furthermore, this is a standard polynomial interpolation can be solved for arbitrary points by several efficient algorithms in $O(M^2)$ operations (where the least cost general algorithm to solve it is by the Newton interpolation algorithm) see, e.g., [31].

Instead of using a standard general purpose solution, we propose in the following a solution more suitable for the problem at hand. It assumes a specific choice of sample points but in turn obtains a competing closed expression for the vector ρ. An important special feature of the following solution is that it produces, at an intermediate step, the coefficient vector for the symmetric polynomial $g(s)$ that corresponds to the solution $\rho(x)$ via the mapping (5), viz.,

$$\rho(x)_{|x=\frac{1}{2}(s+s^{-1})} := g(\tilde{s}) = s^{-M}g(s). \tag{28}$$

It follows that the positivity condition (20) can be replaced by

$$g(\tilde{s}) > 0 \qquad \forall s \in T \tag{29}$$

and the no zero condition (19) in Theorem 1 by

$$g(s) \neq 0 \qquad \forall s \in T. \tag{30}$$

Indeed, it will be argued that there is no actual need to recover $\rho(z)$ and that it is possible, and as a matter of fact preferable, to halt at the intermediate step, right after $g(s)$ has been obtained, and proceed to examine the condition (30) instead of the condition (iii) in Theorem 1.

The relation between the coefficient vector of $\rho(x)$ and the coefficients of the symmetric polynomial $g(s) = \mathbf{s}^t[g_0, \ldots, g_{M-1}, g_M, g_{M-1}, \ldots, g_0]^t$ are

$$\begin{bmatrix} \rho_0 \\ \rho_1 \\ \vdots \\ \rho_M \end{bmatrix} = B_M \acute{I}_M \begin{bmatrix} g_M \\ g_{M-1} \\ \vdots \\ g_0 \end{bmatrix} \tag{31}$$

where B_M is the Chebyshev matrix and $\acute{I}_M$ the diagonal matrix defined before for (10). This relation was derived in (an appendix of) [8] to deduce there (10). Similar relations were obtained before in [30] in order to convert a symmetric polynomial like $g(s)$ into a $\rho(x)$ in order to test (19) by a Sturm sequence instead of testing the condition (29). The above transformation also underlies the conversion in [15] of $Q(\tilde{s}, z)$ into $H(x, z)$ via corresponding trigonometric relations. It is noteworthy that, unlike the derivations in [15] and [30], the relation (19) was obtained in [8] without assuming that $x \in [-1, 1]$ or $s \in T$ (allowing some additional flexibilities like choosing sampling points also outside this interval).

In order to use Theorem 3 we choose the sample points x_i in $[-1, 1]$. So that sample values $\gamma_i = \rho(x_i)$ can be obtained by the companion 1-D stability test of the last subsection and stability of each $p_{x_i}(z) = R(x_i, z)$ is a necessary condition for the stability of $D(z_1, z_2)$.

For $x \in [-1, 1]$, the vectors b_m in the columns of B_M are the coefficients of the Chebyshev polynomials $T_m(x) = \mathbf{x}^t b_m$, viz.,

$$[1, x, \ldots, x^M]B_M = [T_0(x), T_1(x), \ldots, T_M(x)]. \tag{32}$$

Note that (12) represents vectorially the familiar recursion

$$T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x) \tag{33}$$

known to generate the (first kind) Chebyshev polynomials, when started with $T_0(x) = 1$ $T_1(x) = x$.

It follows that it is possible to write the set of (27) as

$$\begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_M \end{bmatrix} = \begin{bmatrix} 1 & T_1(x_0) & \ldots & T_M(x_0) \\ 1 & T_1(x_1) & \ldots & T_M(x_1) \\ \vdots & \vdots & & \vdots \\ 1 & T_1(x_M) & \ldots & T_M(x_M) \end{bmatrix} \acute{I}_M \begin{bmatrix} g_M \\ g_{M-1} \\ \vdots \\ g_0 \end{bmatrix}. \tag{34}$$

The equation can be used to find a corresponding $g(s)$ for values of $\rho(x)$ at any distinct set of x_i's. A simple and closed form solution for (34) (and hence for (27)) results when the sample points are chosen in a discrete cosine transform (DCT)-like manner as follows:

$$\theta := \frac{2\pi}{2M+1}; \quad x_m := \cos(m\theta), \qquad m = 0, \ldots, M. \tag{35}$$

Incidentally, due to familiar relations between the Chebyshev polynomial and the above cosine sequence, the above sequence

$X_{0:M} = \{x_0, \ldots, x_M\}$ of points can in itself be obtained by recursion similar to (33), viz.,

$$x_0 = 1 \quad x_1 = a \quad x_{m+1} = 2ax_m - x_{m-1} \tag{36}$$

for $m = 0, \ldots, M-1$, where $a := \cos\theta$.

For the above choice of sample points, the solution to (34) can be adopted from the solution that was designed for the interpolation problem that arises in complex 1-D stability test based procedures in [22] and [7]. In order to deduce the solution usable here from the solution derived for [22] and [7], let us evaluate each row (34) for sample value γ_i $i = 1, \ldots, M$ in (35)

$$\begin{aligned}\gamma_i &= [T_0(x_i), \ldots, T_M(x_i)]\,[g_M, 2g_{M-1} \ldots, 2g_0]^t \\ &= \left[1, \frac{s_i + s_i^{-1}}{2}, \ldots, \frac{s_i^M + s_i^{-M}}{2}\right][g_M, 2g_{M-1}, \ldots, 2g_0]^t \\ &= \left[s_i^{-M}, \ldots, 1, \ldots, s_i^M\right][g_0, \ldots, g_M, \ldots, g_0]^t.\end{aligned} \tag{37}$$

Next, identify the polynomial $\epsilon(s)$ in [7], [22] (restricting interest to the case there when ϵ is real) with $g(s)$. Also, identify the points and values $\epsilon(s_m) = b_m$ for $m = 0, \ldots M$ there with $x_m = \mathrm{Real}\{s_{M-m}\}$ and $\gamma_m = b_{M-m}$ for $m = 0, \ldots, M$ here. With these correspondences, it follows from the solution proved there that the solution for (34) here is given by

$$\begin{bmatrix} g_M \\ g_{M-1} \\ \vdots \\ g_0 \end{bmatrix} = \frac{1}{2M+1} C_M \acute{I}_M \begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_M \end{bmatrix} \tag{38}$$

where $C_M = (c_{i,\ell})$ is a square $M+1$ sized matrix whose entries are

$$c_{i,\ell} = \cos(i\ell\theta), \qquad i = 0, \ldots, M;\ \ell = 0, \ldots, M. \tag{39}$$

The entries of C_M consist of values $c_{i,\ell} \in X_{0:M}$, where each row (and column) forms a certain permutation of the set $X_{0:M}$. As a matter of fact, C_M can be constructed from the already available entries of $X_{0:M}$ by the following procedure.

For $i = 0, \ldots, M$ and $\ell = 0, \ldots, M$

$$\begin{aligned} &k(i,\ell) \equiv i\ell \bmod (2M+1) \\ \text{if} \quad &k(i,\ell) > M \quad \text{then} \quad k(i,\ell) \leftarrow 2M+1-k(i,\ell) \\ &c_{i,\ell} = x_{k(i,\ell)} \end{aligned} \tag{40}$$

where, for nonnegative integers a, b, c, the notation $a \equiv b \bmod c$ means that a is the least integer equivalent to b modulo c (i.e., a is $0 \le a < c$ such that c divides $b-a$). The proof of the above procedure follows from the periodicity of the sample points and properties of the function $\cos(k\theta)$.

Finally, the coefficients for $\rho(x)$ can be determined from $[g_M, g_{M-1}, \ldots, g_0]^t$ by one extra matrix multiplication (31). Or putting things together, the solution for (27) for sample values chosen as in (35) is expressed by

$$\begin{bmatrix} \rho_0 \\ \rho_1 \\ \vdots \\ \rho_M \end{bmatrix} = \frac{1}{2M+1} B_M \acute{I}_M C_M \acute{I}_M \begin{bmatrix} \gamma_0 \\ \gamma_1 \\ \vdots \\ \gamma_M \end{bmatrix}. \tag{41}$$

Once the solution ρ has been obtained, it remains to examine whether the condition (19) holds for the polynomial $\rho(x)$ of degree M. This condition can be tested normally by creating a Sturm sequence of length $M+1$ (see, e.g., [30]).

The above derivation reveals however a way to lower even further the overall computation. As shown, the sample values of $\rho(x)$ create at an intermediate stage (38), the symmetric polynomial $g(s)$ that relates to $\rho(x)$ via (28). As a consequence, it is possible to test for $g(s)$ the condition (30) instead of testing (19). The testing of the condition (28) by the author's unit-circle zero location procedure [7] takes normally a number of operations comparable to the cost of testing (19) by the Sturm method in [30] and it also handles neatly not normal situations (that may occur and coexist with validity of (19) and (30)!). Avoiding the transformation of $g(s)$ into $\rho(x)$ (31), that has an $O(n^4)$ cost, contributes noticeable to reducing the overall $O(n^4)$ cost of the 2-D stability test.

IV. Proposed Procedure

This section starts with summarizing the proposed 2-D stability test, then illustrates it by a numerical example and finally evaluates its cost of computation.

A. *Procedure for Testing Stability of $D(z_1, z_2)$*

The new 2-D stability test is presented by the next five steps that puts together the components presented in the previous sections. In this outline, 'exit' notes points at which the procedure terminates with a "not (2-D) stable" decision (points where a necessary condition for 2-D stability is already violated).

- Step 1) Determine whether $D(z, 1)$ is 1-D stable. If not stable—"exit." Optionally (because, strictly speaking, this second 1-D stability test can be skipped, as explained below), test 1-D stability of $D(1, z)$ and if not stable "exit."
- Step 2) Convert D to H using (7)–(12).
- Step 3) Set $M = n_1 n_2$, $\theta = (2\pi)/(2M+1)$, $a = \cos(\theta)$ and use (36) to create the set of sample points $X_{0:M}$. For $m = 0, 1, \ldots, M$ apply the companion 1-D stability (Algorithm 2 + Theorem 2) to $p_{x_m}(z) = R(x_m, z)$. If $p_{x_m}(z)$ is not 1-D stable (as soon as a $r_{[i]0} \le 0$, or a $r_i(1) \le 0$ is observed)—'exit'. Otherwise, retain $r_n(z) (= r_{n,0} > 0)$ as γ_m.
- Step 4) Use (38) to obtain $\{g_0, \ldots, g_M\}$ from $\{\gamma_0, \gamma_1, \ldots, \gamma_M\}$. Note that the entries of C, defined by (39), may be obtained from the already available entries of $X_{0:M}$ by (40). It is also notable that division by $2M+1$ may be skipped (a resulting up scaled $g(s)$ is admissible). Next, use $\{g_0, \ldots g_M\}$ to form the symmetric matrix $g(s) = \mathbf{s}^t[g_0, g_1, \ldots, g_M, \ldots, g_1, g_0]^t$. (It is also possible to use (31) to obtain $\rho(x)$, but as aforementioned, this alternative implies an unnecessary increase in the computation cost.)
- Step 5) Examine the condition "$g(s) \ne 0\ \forall s \in T$" or equivalently (29). (Alternatively, if $\rho(x)$ has been recovered, examine for $\rho(x)$ the condition (19) or (20).) $D(z_1, z_2)$ is stable if, and only if, this condition is true and the current step has been reached without an earlier "exit."

The reason testing the stability of $D(1,z)$, the second condition in Theorem 1, is said to be only optional at Step 1), is because it is actually tested later in Step 3), where the sampling point $x_0 = 1$ is acquired by testing the stability of $p_{x_0}(z) = D(1,z)^2$. Also, as already explained it is preferable to obtain only $g(s)$ in Step 4) and examine it in Step 5) for no zeros on T with the refined procedure [6]. When a fully algebraic decision is not crucial, a reasonably reliable decision for Step 5) may also be reached by plotting the balanced polynomial $g(\tilde{s})$ for values along (the upper half of) the unit circle and inspect whether it is positive for all $s \in T$ (alternatively, plotting and inspecting positivity of $\rho(x)$ for all $x \in [-1,1]$).

B. Numerical Example

For illustration, we consider the polynomial used in [8] as well as in many papers before [1, p. 129], [9], [10], [27]

$$D(z_1,z_2) = \begin{bmatrix} 1 & z_1^1 & z_1^2 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ z_1^2 \\ z_2^2 \end{bmatrix}.$$

Step 1) Here, $D(z,1) = D(z,1) = 1 + 3z + 7z^2$ is stable.

Step 2) Convert D to H using (7)–(10)

$$H = \begin{bmatrix} 1 & 4 & 11 & 16 & 13 \\ 0 & 2 & 8 & 18 & 20 \\ 0 & 0 & 4 & 8 & 16 \end{bmatrix}.$$

Step 3) Here, $n_1 = n_2 = 2$, $M = 2n_1n_2 = 8$. Here $\theta = (2\pi)/(17)$. (The remaining of this example uses approximate rational numbers that were produced by MATLAB running this example in rational numeric format. Due to some further properties of Algorithm 2 [32], this provides a better compromise between compactness and accuracy than using decimal numbers). Use (35) to obtain the sample points $X_{0:8} = \{1, 939/1007, 1059/1433, 842/1889, 253/2742, -1223/4469, -549/911, -1175/1382, -635/646\}$. Next obtain samples values $\gamma_i = \rho(x_i)$ for $i = 0,\dots,9$ by application of Algorithm 2 to $p_{x_i}(z) = [1, x_i, x_i^2]H[1, z, z^2, z^3, z^4]^t$.

For $m = 0$, $x_0 = 1$ $p_{x_0}(z) = [1, 6, 23, 42, 49][1, z, z^2, z^3, z^4]^t$. Algorithm 4 produces the table shown at the bottom of the page (entries in parentheses become available by symmetry). The polynomial is stable. Retain $\gamma_0 = 1782000$.

For $m = 1$, $x_1 = 939/1007$ $p_{x_1}(z) = [1, 2041/348, 4585/209, 7352/185, 5923/130]$ $[1, z, z^2, z^3, z^4]^t$. Algorithm 4 produces the second table shown at the bottom of the page.

The polynomial is stable. Retain $\gamma_1 = 1295834$. Repeating the process for $m = 2,\dots 8$ (the details are skipped), all subsequent polynomials $p_{x_m}(z)$ are too stable and provide sample values as follows: $\gamma_2 = 487828$, $\gamma_3 = 90265$, $\gamma_4 = 125437/15$, $\gamma_5 = 17468/25$, $\gamma_6 = 21\,930/79$, $\gamma_7 = 23\,215/31$ and $\gamma_8 = 138\,817/73$.

Step 4) Form C by (39) or using (40) set the sampling points $X_{0:8}$ into

$$C_s = \begin{bmatrix} x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 & x_0 \\ x_0 & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\ x_0 & x_2 & x_4 & x_6 & x_8 & x_7 & x_5 & x_3 & x_1 \\ x_0 & x_3 & x_6 & x_8 & x_5 & x_2 & x_1 & x_4 & x_7 \\ x_0 & x_4 & x_8 & x_5 & x_1 & x_3 & x_7 & x_6 & x_2 \\ x_0 & x_5 & x_7 & x_2 & x_3 & x_8 & x_4 & x_1 & x_6 \\ x_0 & x_6 & x_5 & x_1 & x_7 & x_4 & x_2 & x_8 & x_3 \\ x_0 & x_7 & x_3 & x_4 & x_6 & x_1 & x_8 & x_2 & x_5 \\ x_0 & x_8 & x_1 & x_7 & x_2 & x_6 & x_3 & x_5 & x_4 \end{bmatrix}$$

and $\gamma_0 \dots \gamma_8$ into (38) to obtain $g = [g_0, \dots, g_M, \dots, g_0]^t = [144, 1224, 6356, 22464, 60608, 127392, 215584, 293880, 326696, 293880, 215584, 127392, 60608, 22464, 6356, 1224, 144]^t$.

It is also possible (but not required) to proceed and obtain $\rho(x)$. Setting the above g into (31), for which B_8 of (11) is created by (12), gives $\rho(x) = \mathbf{x}^t[4320, 30912, 112208, 257664, 405632, 444672, 333056, 156672, 36864]^t$ which is indeed the last polynomial $R_4(x,z)$ that Algorithm 1 produces for this example, see [8].

Step 5) The polynomial $g(s) = [1, s \dots, s^{16}]g$ with the above g can be shown to have no zeros on T (or the above $\rho(x)$ to have no zeros on $[-1,1]$). Thus, the tested polynomial is 2-D stable.

$$\begin{array}{ccccccccc} 50 & & 48 & & 46 & & (48) & & (50) \\ & 48 & & 84 & & (84) & & (48) & \\ & & 2148 & & 3096 & & (2148) & & \\ & & & 71\,280 & & (71280) & & & \\ & & & & 1\,782\,000 & & & & \end{array}$$

$$\begin{array}{ccccccccc} 6053/130 & & 4971/109 & & 9170/209 & & (4971/109) & & (6053/130) \\ & 5793/130 & & 13\,099/167 & & (13\,099/167) & & (5793/130) & \\ & & 14\,779/8 & & 18\,722/7 & & (14\,779/8) & & \\ & & & 509\,417/9 & & (509\,417/9) & & & \\ & & & & 1\,295\,834 & & & & \end{array}$$

C. Cost of Computation

An approximate count of operations for the procedure will now be carried out. The procedure involves only real polynomials so that all counts are in terms of real arithmetic operations (multiplication and additions; the term 'flop' when used stands for one addition plus one multiplication). The notation $O(n_{1,2}^k)$ denotes a bivariate polynomial in n_1 and n_2 with powers $n_1^{k_1} n_2^{k_2}$ such that $k_1 + k_2 \leq k$.

Step 1) is a 1-D stability test for a polynomial of degree $n = n_1$. Its $O(n^2)$ complexity is negligible compared to the overall anticipated $O(n^4)$ count. In Step 2), the conversion of D into Q involves $(n_2+1)^2$ multiplications of pairs of polynomials of degree n_1. Each multiplication requires $(n_1+1)^2$ flops. The subsequent conversion of Q to H requires $n_1^2 n_2$ flops. Thus, Step 2) requires $n_1^2 n_2^2 + O(n_{1,2}^3)$ flops. Step 3) involves $M+1$ repetitions of Algorithm 2 each with a polynomial of degree $2n_2$, where $M = 2n_1 n_2$. Algorithm 2 requires for a polynomial of degree n, $(1/2)n^2 + O(n)$ flops (assuming it is carried out as a recursion with two multipliers). Thus, Step 3) requires $4n_1 n_2^3$ flops. Step 4) is carried out by (38) and requires M^2, i.e. $4n_1^2 n_2^2$ flops. Step 5) can be carried out by $4n_1^2 n_2^2$ multiplications and $8n_1^2 n_2^2$ using the zero location method in [5] or its bettered version [6]. The resulting overall complexity of the test is therefore $9n_1^2 n_2^2 + 4n_1 n_2^3 + O(n_{1,2}^3)$ real multiplications and $13n_1^2 n_2^2 + 4n_1 n_2^3 + O(n_{1,2}^3)$ real additions.

The above count of operations does not position this test as the least cost 2-D stability test that is available at this time because it requires approximately four times the cost of the immittance-type telepolation-based tests in [7], [23] and two times the cost of the scattering-type telepolation-based test in [22]. These tests involve a collection of (only) $n_1 n_2 + 1$ 1-D stability tests (but) of polynomials with complex coefficients (complex arithmetic operations are translation to real arithmetic counts in the conventional manner) plus one positivity test of a symmetric polynomial of degree $2n_1 n_2$. The doubling-degree technique used here doubles the number of 1-D polynomials and double their degrees. However, it also enables a 2-D stability test manipulates only *real* coefficient univariate polynomials. Since the only known more efficient stability tests are the above mentioned tests that employ a collection of complex univariate polynomial stability tests (a comparison with the costs of other available 2-D stability tests is detailed in [22] and [7]), the procedure in this paper appears to be the least cost stability test for 2-D discrete system that relies on testing of real univariate polynomial and involves real arithmetic operations only.

As one might anticipate at this point, it is also possible to produce a scattering counterpart for the procedure presented in this paper. It would consist of telepolation of a tabular test that is obtained by combination of the tabular test in [12] with the doubling-degree technique. After putting the details together properly, the outcome would be a 2-D stability test that is carried out by $2n_1 n_2 + 1$ MJT 1-D stability tests applied to real polynomials each of degree $2n_2$ (instead of $n_1 n_2 + 1$ MJT tests applied to complex polynomials each of degree n_2 in [22]). However the cost of such a procedure will be higher than the cost of the current procedure because the cost of each MJT test is higher than the cost of the test [32] that is used here as the companion 1-D stability test.

Beyond its unique feature of using real polynomial stability test, the test shares some additional niceties that stem from being carried out by a collection of 1-D stability tests and that are common to also the other telepolation-based 2-D stability tests. One is simplicity of programming. The second follows from the fact that the stability of each 1-D polynomials is a necessary condition for 2-D stability. Therefore, all necessary conditions for 1-D stability encountered en route are also necessary condition for 2-D stability. This property may speed the spotting of an unstable bivariate polynomial and save the remaining cost of computation.

V. Concluding Remarks

The paper has presented a new procedure to test stability of a 2-D discrete system. The new procedure carries out the task by a finite number of *real* univariate polynomial 1-D stability tests (plus one unit-circle zero location test) and it is apparently the most efficient algebraic 2-D stability testing procedure that involves real univariate polynomials and real-arithmetic only.

It is interesting to examine the contribution in this paper from the perspective of the roles that the doubling-degree technique has played in the development of 2-D stability tests. The doubling-degree technique was introduced by Bose [15] to help researchers to develop 2-D stability tests from real 1-D stability tests. It has been used since then in numerous papers (we cited only a few of the more pertinent papers). The doubling-degree technique increases the cost of computation compared to developing tests starting with stability conditions posed on 1-D polynomials with complex coefficients. This property is true in general even though it has been demonstrated explicitly only for some of the cases (because many papers were published without cost assessment and not every instance was treated both with and without the doubling-degree technique).

Influenced by its use in [19] to develop a 2-D stability test from the BT, we considered the doubling-degree technique at an early stage in a conference paper [33]. As a matter of fact, this conference paper was the first to announce $O(n^6)$ complexity for 2-D tabular test. However, we soon afterward realized that developing tabular 2-D stability from 1-D stability tests for complex univariate polynomials achieve lower cost. As a consequence, our interest shifted to complex polynomial 1-D stability test based tabular tests [13], [14]. It is important to realize that a tabular 2-D stability test that is derived from a 1-D stability for complex polynomials produces a 2-D stability test that manipulates *real* matrices and vectors (or bivariate and univariate polynomials) when testing a real bivariate polynomial. For instance, irrespective of the fact that the derivation of the tests in [12]–[14] rely on complex polynomial 1-D stability test, the resulting 2-D tabular tests are efficient procedures that propagate real matrices and involve only real arithmetic. It is equally hard to justify the use of the doubling-degree technique for the derivation of tabular 2-D stability test by saying that "engineers are more familiar and will more readily apply a real 1-D stability test" [16] because the benefit of using real 1-D stability test is not passed to the user. The outcome is a tabular 2-D stability that, like a corresponding tabular test derived from a complex polynomial 1-D stability test, manipulates real bivariate polynomial (or matrices, or an array of univariate polynomial) but has a higher computational cost.

The use of telepolation changes the above situation. Telepolation carries out a 2-D stability test by a collection of 1-D stability tests of certain qualifying forms that are related to the 1-D stability test that underlies the development of the tabular 2-D stability test that is being interpolated. A tabular test that stems from a (scattering/immittance type) 1-D stability test of complex or real polynomials gives rise to a collection of 1-D stability tests of complex or real polynomials, respectively (of scattering/immittance types, respectively). This perception, renewed our interest in the study reported earlier in [33] and led us to expand it into a journal paper [8] and to write the current paper that shows that, in conjunction with telepolation, the doubling-degree technique admits testing 2-D stability by a collection of real 1-D stability tests.

The 2-D stability test presented in this paper thus brings a positive twist to the use of the doubling-degree technique for developing 2-D stability tests so far. The doubling-degree technique still takes a toll in producing a test whose cost is higher than the alternatives in [7], [23] that stem from complex 1-D stability tests. However, for the first time it also enhances the product with a property that without using it is not attainable—it admits the testing of 2-D stability by a collection of *real* instead of *complex* 1-D stability tests. The use of real 1-D stability tests may have indeed (as suggests also in the above quote from [16]) a better appeal to engineers than using complex 1-D stability tests.

References

[1] B. T. O'Connor and T. S. Huang, "Stability of general two-dimensional recursive filters," in *Two-Dimensional Digital Signal Processing I: Linear Filters*, T. S. Huang, Ed. Berlin, Germany: Springer-Verlag, 1981, ch. 4.

[2] N. K. Bose, *Applied Multidimensional System Theory*. New York: Van Nostrand Reinhold, 1982, ch. 3.

[3] E. I. Jury, "Stability of multidimensional systems and related problems," in *Multidimensional Systems: Techniques and Applications*, S. G. Tzafestas, Ed. New York: Marcel Dekker, 1986, ch. 3.

[4] W.-S. Lu and A. Antoniou, *Two-Dimensional Digital Filters*. New York: Marcel Decker, 1992, ch. 5.

[5] Y. Bistritz, "Zero location with respect to the unit circle of discrete-time linear system polynomials," *Proc. IEEE*, vol. 72, pp. 1131–1142, Sept. 1984.

[6] ——, "Zero location of polynomials with respect to the unit-circle unhampered by nonessential singularities," *IEEE Trans. Circuits Syst. I*, vol. 49, pp. 305–314, Mar. 2002.

[7] ——, "Stability testing of 2-D discrete linear systems by telepolation of an immittance-type tabular test," *IEEE Trans. Circuits Syst. I*, vol. 48, pp. 840–846, July 2001.

[8] ——, "Real polynomial based immittance-type tabular stability test for 2-D discrete systems," *Circuits Syst. Signal Process.*, vol. 22, no. 3, pp. 255–276, 2003.

[9] T. S. Huang, "Stability of two-dimensional recursive filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, pp. 158–163, June 1972.

[10] G. A. Maria and M. M. Fahmy, "On the stability of two-dimensional digital filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, pp. 470–472, 1973.

[11] X. Hu, "Further simplification to 2-D filter stability test," *IEEE Trans. Circuits Syst. II*, vol. 44, pp. 330–332, Mar. 1997.

[12] X. Hu and E. I. Jury, "On two-dimensional filter stability test," *IEEE Trans. Circuits Syst.*, vol. 41, pp. 457–462, July 1994.

[13] Y. Bistritz, "Stability testing of two-dimensional discrete linear system polynomials by a two-dimensional tabular form," *IEEE Trans. Circuits Syst. I*, vol. 46, pp. 666–676, June 1999.

[14] ——, "Immittance-type tabular stability test for 2-D LSI systems based on a zero location test for 1-D complex polynomials," *Circuits, Syst. Signal Process.*, vol. 19, pp. 245–265, 2000.

[15] N. K. Bose, "Implementation of a new stability test for two-dimensional filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 117–120, Apr. 1977.

[16] X. Hu and T. S. Ng, "2-D filter stability using polynomial array," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 550–554, Apr. 1990.

[17] X. Hu, "2-D filter stability tests using polynomial array for $F(z_1, z_2)$ on $|z_1| = 1$," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 1092–1095, Sept. 1991.

[18] M. Barret and M. Benidir, "A new algorithm to test stability of 2-D digital recursive filters," *Signal Process.*, vol. 37, pp. 255–264, 1994.

[19] B. M. Karan and M. C. Srivastava, "A new test for 2-D filters," *IEEE Trans. Circuits Syst.*, vol. CAS-33, pp. 807–809, Aug. 1986.

[20] ——, "Correction to 'a new test for 2-D filters'," *IEEE Trans. Circuits Syst.*, vol. 36, p. 167, Jan. 1989.

[21] K. Premaratne, "Stability determination of two-dimensional discrete-time systems," *Multidim. Syst. Signal Process.*, vol. 4, no. 4, pp. 331–354, 1993.

[22] Y. Bistritz, "Stability testing of two-dimensional discrete-time systems by a scattering-type tabular form and its telepolation," *Multidim. Syst. Signal Process.*, vol. 13, pp. 55–77, 2002.

[23] ——, "On testing stability of 2-D discrete systems by a finite collection of 1-D stability tests," *IEEE Trans. Circuits Syst. I*, vol. 49, pp. 1634–1638, Nov. 2002.

[24] ——, "A stability test of reduced complexity for 2-D digital system polynomials," in *Proc. IEEE Int. Symp. Circuits Systems*, vol. 5, Monterey, CA, May/June 1998, pp. 94–97.

[25] ——, "A modified unit-circle zero location test," *IEEE Trans. Circuits Syst. I*, vol. 43, pp. 472–475, June 1996.

[26] ——, "A circular stability test for general polynomials," *Syst. Control Lett.*, vol. 7, no. 2, pp. 89–97, 1986.

[27] E. I. Jury, "Modified stability table for 2-D digital filter," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 116–119, Jan. 1988.

[28] Y. Bistritz, "A new stability test for linear discrete systems in table form," *IEEE Trans. Circuits Syst.*, vol. CAS-30, pp. 917–919, Dec. 1983.

[29] ——, "Losslessness and stability of LDI ladders and filters," *Circuits, Syst. Signal Process.*, vol. 11, pp. 325–352, 1992.

[30] B. D. O. Anderson and E. I. Jury, "Stability test for two-dimensional recursive filters," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, pp. 366–372, Aug. 1973.

[31] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: John Hopkins Univ. Press, 1983.

[32] Y. Bistritz, "An efficient integer-preserving stability test for discrete-time systems," *Circuits, Syst. Signal Process.*, vol. 23, pp. 195–213, 2004.

[33] ——, "An immittance-type stability test for two-dimensional digital filters," in *Proc. 28th Annu. Asilomar Conf. Signals Systems Computers*, Oct. 1994, pp. 918–922.

Yuval Bistritz (S'79–M'79–SM'87–F'03) received the B.Sc. degree in physics and the M.Sc. and Ph.D. degrees in electrical engineering from Tel Aviv University, Tel Aviv, Israel, in 1973, 1978, and 1983, respectively.

From 1979 to 1984, he held various assistant and teaching positions in the Department of Electrical Engineering, Tel Aviv University, and in 1987, he joined the department as a Faculty Member. From 1984 to 1986, he was a Research Scholar in the Information System Laboratory, Stanford University, Stanford, CA, working on fast signal processing algorithms. From 1986 to 1987, he was with AT&T Bell Laboratories, Murray Hill, NJ, and from 1994 to 1996 with the DSP Group, Santa Clara, CA, working on speech processing algorithms. His research interests are in the area of digital signal processing and system theory with current focus on stability of multidimensional systems and speaker recognition.

Dr. Bistritz received the Chaim Weizmann Fellowship award for Postdoctoral Research for 1984–1986 and the Distinguished Researcher Award from the Isreali Technological Committee in 1992.