Multidimensional Systems and Signal Processing, 13, 55–77, 2002
© 2002 Kluwer Academic Publishers. Manufactured in The Netherlands.

Stability Testing of Two-Dimensional Discrete-Time Systems by a Scattering-Type Stability Table and its Telepolation

YUVAL BISTRITZ
Department of Electrical Engineering-Systems, Tel Aviv University, Tel Aviv 69978, Israel

Received May 12, 2000; Revised May 1, 2001; Accepted May 3, 2001

Abstract. Stability testing of two-dimensional (2-D) discrete-time systems requires decision on whether a 2-D (bivariate) polynomial does not vanish in the closed exterior of the unit bi-circle. The paper reformulates a tabular test advanced by Jury to solve this problem. The 2-D tabular test builds for a real 2-D polynomial of degree (n_1, n_2) a sequence of n_2 matrices or 2-D polynomials (the '2-D table'). It then examines its last polynomial - a 1-D polynomial of degree $2n_1n_2$ - for no zeros on the unit circle. A count of arithmetic operations for the tabular test is performed. It shows that the test has $O(n^6)$ complexity (assuming $n_1 = n_2 = n$)- a significant improvement compared to previous tabular tests that used to be of exponential complexity. The analysis also reveals that, even though the testing of the condition on the last polynomial requires $O(n^4)$ operations, the count of operations required for the table's construction makes the overall complexity $O(n^6)$. Next it is shown that it is possible to *tele*scope the last polynomial of the table by inter*polation* and circumvent the construction of the 2-D table. The *telepolation* of the tabular test replaces the table by $n_1n_2 + 1$ stability tests of 1-D polynomials of degree n_1 or n_2 of certain form. The resulting new 2-D stability testing procedure requires a very low $O(n^4)$ count of operations. The paper also brings extension for the tabular test and its simplification by telepolation to testing 2-D polynomials with complex valued coefficients.

Key Words: LSI (discrete-time) 2-D systems, Stability testing, Interpolation

1. Introduction

Two-dimensional and higher multidimensional linear discrete-time systems arise in modeling sampled data of images, signals from several sensors and other applications. Stability is required, like in the one dimensional case, to ensure that bounded inputs produce bounded outputs. However ensuring stability and testing it for higher dimensional systems is known as a much more difficult problem. This work will concentrate on the problem of testing the stability of two-dimensional (2-D) discrete-time (linear shift invariant) systems. The key to stability determination of 2-D discrete systems is an efficient solution for the next problem.

Problem statement. Given a 2-D (bivariate) polynomial

$$D(z_1, z_2) = [1, z_1, \ldots, z_1^{n_1}]D[1, z_2, \ldots, z_2^{n_2}]^t \tag{1}$$

of degree (n_1, n_2), where $D = (d_{i,k})$ is the coefficient matrix (superscript ^{t} denotes transpose), determine whether it does not vanish in the closed exterior of the unit bi-circle, viz.,

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

where the notations $T = \{z : |z| = 1\}$, $U = \{z : |z| < 1\}$, $V = \{z : |z| > 1\}$ will be used to denote the unit circle, its interior, and its exterior, respectively, and the bar denotes closure, $\overline{V} = V \cup T$.

A 2-D polynomial $D(z_1, z_2)$ that satisfies (2) will be called *stable*. Some other forms are also in use in the literature to describe 2-D stability. The polynomial may be presented in negative powers of the variable and/or it may be required not to vanish in $\overline{U} \times \overline{U}$. The moving between different conventions may require some minor adjustment, like reversion of the matrix D, cf. [1]. We shall also call a 1-D system polynomial stable, if

$$p(z) = [1, z, \ldots, z^n][p_0, \ldots, p_n]^t \neq 0 \qquad \forall z \in \overline{V}. \tag{3}$$

The fact that it is possible to arrange a similarity in the definition of 1-D and 2-D stable polynomials as in (2) and (3), is instrumental for the solution of the problem but it is somewhat misleading. It hides inherent difficulties that complicate the stability issue compared to the 1-D case. The main source of difficulty, that also complicates other topics in the analysis and design of higher dimensional systems is the absence of factorization for multivariate polynomials. An interesting peculiar outcome is a phenomenon called "nonessential singularity of the second kind" (NSSK) observed by Goodman [2]. Accordingly, a 2-D digital filter may be stable even though its denominator $D(z_1, z_2)$ vanishes on T^2 (i.e. it is not "stable") because its numerator 2-D polynomial (without having a common factor with $D(z_1, z_2)$) may stabilize the filter. Consequently, in a strict mathematical sense, a stable $D(z_1, z_2)$ is a sufficient but not necessary condition for stability of the system. However, it was rightly argued in [3] that in a practical engineering sense a stable 2-D polynomial is also necessary for designing a robustly stable filter. Another pertinent outcome is that the problem can not be solved by in numerical manner similar to the possibility of testing stability of $p(z)$ by determining numerically its n zeros and examining their locations. Comprehensive background on 2-D discrete systems with particular attention to the stability issue may be found in [3] [4] [5] [6].

Considerable amount of papers have been published offering solutions to the stated problem. The solutions are traditionally classified into "tabular" and "determinantal" tests. Tabular tests, first proposed in [7], attempt to extend 1-D tabular stability tests to the 2-D case. Determinantal methods, first proposed in [8], attempt to similarly generalize 1-D stability conditions posed on the determinants of "stability" matrices (like the Schur-Cohn and the inner matrices). Many early solutions are reviewed and listed in [3] [4] [5] [6]. Some more recent 2-D stability tests, [9], [10], [11], [12], [13], [1], [14], will be noted later.

The current paper considers the 2-D stability test proposed in [15]. The test evolves from Jury's so called *modified* 1-D stability test proposed by him in [16] and in several more versions and occasions in the last four decades [17, p. 104], [18], [19], [20]. The modified Jury test forms one of several types of stability tests that evolved from the Schur-Cohn test through modifications proposed by Marden and Jury. In [21] this class

of Schur-Cohn-Marden-Jury (SCMJ) tests were classified into four types. In this classification the modified Jury tests fall into the "C-type" category. The special property of tests in this category is that they create for the 1-D polynomial a tables that produces at certain locations entries equal (or with sign opposite) to the principal minors of the Schur-Cohn Bezoutian matrix for the tested polynomial. The pertinent results from [21] will be cited and used below. The relation with the Schur-Cohn minors allowed [16], [15] to adopt Siljak's simplification in [22] and obtain 2-D stability tests with just a single so called 'positivity test'.

The current paper first brings a new form to this tabular 2-D stability test. The derivation follows the 1-D stability test assigned in [21] as the prototype for the C-type stability tests. This prototype uses a uniform recursion to constructs for the 1-D tested polynomials a sequence of 1-D polynomials (the 1-D 'table') whose leading coefficients are equal to the principal minors of the Schur-Cohn matrix. Strictly speaking, this form of the 1-D stability test is not identical with any of the of the previously available versions of the modified Jury test [17] [18] [19] [20]. The 2-D stability test uses a simple to program algorithm to build for $D(z_1, z_2)$ a '2-D stability table' in the form of a sequence of 2-D polynomials, or matrices (a third adequate look on it is a 1-D stability table with entries that are polynomials instead of scalars, hence called 'polynomial array' [15]). An accompanying theorem poses necessary and sufficient conditions on the on the 2-D table for $D(z_1, z_2)$ to be stable.

The paper proceeds to analyze the computational complexity of this tabular 2-D stability test, a task that was not carried out for it before. It is shown that the test has $O(n^6)$ complexity (for $n = n_1 = n_2$) - a definite advantage over all previous 2-D tabular stability tests that used to be of exponential complexity. This analysis also paves the way to further complexity reduction.

The stability theorem associated with the 2-D table makes it apparent that the construction of the table is required only in order to obtain its last entry - a target polynomial that will be denoted by $\epsilon(s)$. This is a symmetric 1-D polynomial of degree $2n_1n_2$ that has to tested for having no zeros on the unit-circle. The task can be carried out by only $O(n^4)$ (again for $n_1 = n_2 = n$) operations. However the higher cost of the table's construction makes the overall complexity $O(n^6)$.

The paper proposes to *tele*scope (brings forth) $\epsilon(s)$ by inter*polation*. This approach, thereupon called *telepolation*, circumvents the construction of the 2-D table and instead requires only a finite number of low degree 1-D stability tests. These 1-D stability tests provide sample values for $\epsilon(s)$ that are used to recover it via a simple closed form formula. The resulting new 2-D stability testing procedure has an overall complexity of only $O(n^4)$ ($n_1 = n_2 = n$). A more detailed count and comparison with other available solutions to this problem indicates that the procedure achieves one of the lowest count of operations reported for 2-D stability testing.

The procedure derived in this paper has been reported before in a conference [23]. The brief presentation there was restricted to 2-D polynomial with real coefficients and contains no derivation details or proofs. This paper brings all the details required to establish the method and numerical illustrations. It also generalizes the 2-D tabular test and its simplification by telepolation to testing (2) for also 2-D polynomials with complex valued coefficients.

The rest of the paper is constructed as follows. The next section brings a revised form for the 2-D tabular stability test in [15] and examines its computational complexity. Section 3 derives a simplified stability test obtained by telepolation of the tabular test. Section 4 generalizes the test to the complex case and discusses various ways to examine $\epsilon(s)$. Section 5 evaluates the efficiency of the test and highlights issues that deserve further investigation.

2. The 2-D Tabular and its Companion 1-D Stability Tests

2.1. Notation

We shall follow the notation convention that we used in some previous related works, e.g. in [1]. Briefly, arrays and polynomials may be used interchangeable. For example, a matrix D may be associated with a 2-D polynomial by $D(z_1, z_2) = \mathbf{z}_1{}^t D\mathbf{z}_2$ as in (1) where $\mathbf{z} \coloneqq [1, z, \ldots, z^i, \ldots]^t$ and its length is depending on context. Vectors and 1-D polynomials are linked similarly, e.g. p is associated with the 1-D polynomial in (3) $p(z) = \mathbf{z}^t p$. The letter **s** is reserved for $s \in T$ (or it infers intention to interpret it at some point in such a way). We shall also use $\tilde{s}$ to denote the variable for a 'balanced polynomial' - a polynomial that extend to equal degree in s and s^{-1} or in $s^{1/2}$ and $s^{-1/2}$, depending on parity, e.g., $p(\tilde{s}) = [s^{-n/2}, \ldots, s^{n/2}]\, p$. $D^\sharp \coloneqq JD^\star J$ and $e_k^\sharp \coloneqq Je_k^\star$ will denote matrix and vector (conjugate) reversion, respectively, where J is the reversion matrix (a matrix with 1's on the main anti-diagonal and 0's elsewhere), and $\star$ denotes complex conjugate.

2.2. Companion 1-D Stability Test

Here we bring the 1-D stability that will be used to derive the 2-D tabular test. It will later also become part of the proposed simpler procedure. In a classification of the Schur-Cohn and Marden-Jury (SCMJ) class of tests into four types in [21], the next cited test is referred as the basic form for C-type tests, where all the C-type tests encompass all versions of the so called Modified Jury tests [17, p. 104] [18] [19] [20]. All the results cited in this subsection are proved in [21].

Algorithm 1 [A scattering-type 1-D stability 'table']. Assign to a complex 1-D polynomial $p(z)$ (3) a sequence of polynomials $\{c_m(z) = \sum_{i=0}^{m} c_{m,i} z^i,\ m = n-1, \ldots, 0\}$, as follows.

$$zc_{n-1}(z) = p_n^\star p(z) - p_0 p^\sharp(z)\,;\ q_{n-1} = 1 \tag{4a}$$

For $m = n-1, \ldots, 1$ do:

$$zc_{m-1}(z) = \frac{c_{m,m}c_m(z) - c_{m,0}c_m^\sharp(z)}{q_m}\,;\ q_{m-1} = c_{m,m} \tag{4b}$$

THEOREM 1: **[Stability conditions for Algorithm 1]**. *If Algorithm 1 becomes singular (a $c_{m,m} = 0$ occurs) then p(z) is not stable. Else, p(z) is stable if, and only if, $c_{m,m} > 0$ for all $m = 1, \ldots, n$.*

Algorithm 1 has some further interesting properties related to the Schur-Cohn (SC) matrix for $p(z)$. The SC matrix, say R, is an $n \times n$ matrix associated with $p(z)$ by the expression,

$$R = L(p^{\star}_{n:1})L^{t}(p_{n:1}) - L(p_{0:n-1})L^{t}(p^{\star}_{0:n-1}) \tag{5}$$

where $p_{0:n-1} = [p_0, \ldots, p_{n-1}]^t$, $p_{n:1} = [p_n, \ldots, p_1]^t$ and $L(a)$ denotes the lower triangular Toeplitz matrix whose first column is the vector a. It forms the Bezoutian with respect to the unit circle for $p(z)$ and may also be expressed by a generating 2-D function, e.g. [24].

THEOREM 1^c: **[Complementing properties for Algorithm 1]**

(a) Algorithm 1 does not become singular if, and only if, its SC matrix R is strongly regular (all the leading principal minors of R are non-zero).

(b) If R is strongly regular then the principal minors of R are given by the leading coefficients of the polynomials created by Algorithm 1, det $\{R_k\} = c_{n-k,n-k}$ $k = 1, \ldots, n$ (where R_k denotes the $k \times k$ upper-left submatrix of R).

(c) If Algorithm 1 does not become singular then p(z) has ν zeros in V and $n-\nu$ zeros in U where ν is given by

$$\nu = Var\{1, c_{n-1,n-1}, \ldots, c_{0,0}\} \tag{6}$$

and Var denotes the number of sign variations in the indicated sequence. (If algorithm 1 is not singular then its $c_{m,m}$ are well defined and non-zero for all $m = n-1, \ldots 0$.)

Remark 1. Theorem 1^c or parts of it were often quoted in the literature but till [21] no proof for it was widely available. In additions often some of the details were not always stated correctly. Algorithm 1 plays several functions in this paper. First, it is an instrument to derive the tabular 2-D stability test. Then, it will become an essential algorithm repeatedly used in the 2-D stability test of reduced complexity. In addition, both the tabular test and its simplification require decision on whether a polynomial symmetric $\epsilon(z)$ of degree $2M$ ($\epsilon^{\sharp} = \epsilon$) has no zeros on the unit circle. Algorithm 1 in conjunction with zero location rule in Theorem 1^c may be used (most often) for this task as well. For this latter task notice the following. A symmetric polynomial has reciprocal zeros, i.e., $\epsilon(z_i) = 0 \rightarrow \epsilon(\frac{1}{z_i^{\star}})$. Therefore, $\epsilon(z) \neq 0$ $\forall z \in T$ if, and only if, it has M zeros in U and M zeros in V (their reciprocal). Setting $\epsilon(z)$ into Algorithm 1 causes an immediate singularity ($c_{n-1}(z) \equiv 0$, a similar singularity would occur in also all other algebraic stability). The remedy is to use the fact (attributed to Cohn, see e.g. [25]) that $\epsilon(z)$ and its derivative $\epsilon(z)'$ have the same number of zeros in V. Thus one can apply Algorithm 1 to $\epsilon(z)'$ and use condition (c) of Theorem 1 to determine whether it has M zeros in V. This scheme works well as long as Algorithm 1 does not become singular for $\epsilon(z)'$. We shall bring alternative complete solutions to this problem later in section 4.3.

2.3. 2-D Tabular Stability Test

The next lemma is a simplification for the condition (2) that was introduced to the field by Huang [26] (with $a = \infty$) and has become the starting point of most of the methods for testing 2-D stability.

LEMMA 1: $\boldsymbol{D}(z_1, z_2)$ *is stable if, and only if,*
(i)

$$D(z, a) \neq 0 \qquad \text{for all} \quad z \in \overline{V} \text{ and some } a \in \overline{V} \tag{7}$$

(ii)

$$D(s, z) \neq 0 \qquad \text{for all} \quad (s, z) \in T \times \overline{V}. \tag{8}$$

The above form is due to Strintzis [27]. A simple proof for this Lemma has recently appeared in [28]. Related forms of simplifying conditions are also available (see a summary in [6]). The lemma shows that the main task of 2-D stability testing concentrates on condition (8), which indeed was taken as the key problem in [15]. This paper will regard the problem as stated in (8) and will obtain solutions for it using the above simplification with $a = 1$. This specific choice will allow a pleasant invariance in the form of the stability conditions when the two variables in the tested polynomial are interchanged. (Swapping of variables in the tested polynomial is clearly always allowed and amounts to $D \to D^t$ in (1). It will turn out that when $n_1 \neq n_2$ this operation may reduce the amount of computation.)

The condition (8) is clearly equivalent to the condition

$$D(\tilde{s}, z) \neq 0 \qquad \text{for all} \quad (s, z) \in T \times \overline{V} \tag{8'}$$

where $D(\tilde{s}, z) \coloneqq s^{-n_1/2} D(s, z)$. Regarding $D(\tilde{s}, z)$ as a polynomial in z with coefficients that are balanced polynomials, it is possible to examine the latter condition by applying Algorithm 1 to $p_{\tilde{s}}(z) = D(\tilde{s}, z)$. The approach gives rise to the next algorithm.

Algorithm 2 [A scattering-type 2-D stability 'table']. Denoting $\boldsymbol{D}(\tilde{s}, z) = \sum_{k=0}^{n} d_k(\tilde{s}) z^k$, $n \coloneqq n_2$, assign to D a sequence of polynomials $\{C_m(\tilde{s}, z) = \sum_{k=0}^{m} c_{[m]k}(\tilde{s})\, z^k,\ m = n-1, \ldots, 0\}$, using the following recursions.

$$zC_{n-1}(\tilde{s}, z) = d_n^{\sharp}(\tilde{s}) D(\tilde{s}, z) - d_0(s) D^{\sharp}(\tilde{s}, z) \qquad , \qquad q_{n-1}(\tilde{s}) = 1 \tag{9a}$$

For $m = n-1, \ldots, 1$ do:

$$zC_{m-1}(\tilde{s}, z) = \frac{c_{[m]m}(\tilde{s}) C_m(\tilde{s}, z) - c_{[m]0}(s) C_m^{\sharp}(\tilde{s}, z)}{q_m(\tilde{s})} \qquad q_{m-1}(\tilde{s}) = c_{[m]m}(\tilde{s}) \tag{9b}$$

The $\tilde{s}$ interpretation is constructive to also obtain stability conditions for this algorithm and later to derive its simplification. Otherwise, it is possible to equally regard Algorithm 1 as creating for $D(s, z)$ a sequence of bivariate polynomials $\{C_m(s, z),\ m = n-1, \ldots, 0\}$ (i.e with normal rather than balanced first variables). It can be shown easily that replacing everywhere in the Algorithm 2 the $\tilde{s}$ by s leads to an equivalent algorithm in the sense of producing the same sequence of coefficient matrices $\{C_m,\ m = n-1 \ldots 0\}$. Maybe the most transparent presentation of the algorithm for its programming (especially by an array oriented language like Matlab) is one that drops altogether polynomial notation and replace (9a) and (9b) by, respectively,

$$[0\, C_{n-1}] = d_n^\sharp * D - d_0 * D^\sharp \quad ; \quad q_{n-1} = 1 \tag{9'a}$$

$$[0\, C_{m-1}] = (c_{[m]m} * C_m - c_{[m]0} * C_m^\sharp)/q_m \quad ; \quad q_{m-1} = c_{[m]m} \tag{9'b}$$

where $*$ and / denote convolution and deconvolution (respectively) of the vector by each column of the matrix columns (the padded zero in the left hand sides represents a column of zeros), cf. [1]. It is important to realize that the division by $q_m(s)$ is exact [15]. Namely, $q_m(s)$ is a factor of the numerator 2-D polynomial that it divides and consequently the recursion creates 2-D polynomials and not 1-D polynomials with coefficients that are rational functions of s. The 2-D polynomial $C_m(s, z)$ has degrees $(2(n_2-m)\ n_1,\ m)$, $m = n_2-1, \ldots, 0$. It is also noted that the polynomials $c_{[m],m}(s)$ are symmetric, i.e. $c_{[m],m} = c_{[m],m}^\sharp$. (We already used this property to write $c_{[m]m}(\tilde{s})$ in (9b) instead of the anticipated $c_{[m]m}^\sharp(s)$). Therefore $c_{[m],m}(\tilde{s})$ is real for $s \in T$. Since Algorithm 2 implements for each $s \in T$ Algorithm 1 for $p_{\tilde{s}}(z) = D(\tilde{s}, z) = s^{-n_1/2} D(s, z)$ and since condition (8) in Lemma 1 is equivalent to (8′), it is concluded, via Theorem 1, that condition (8) holds if and only if all the $c_{[m],m}(\tilde{s})$ polynomials created by Algorithm 2 satisfy

$$c_{[m]m}(\tilde{s}) > 0 \quad \forall s \in T \quad , \quad m = n-1, \ldots, 0\,. \tag{10}$$

The $c_{[m]m}(\tilde{s})$ are, according to Theorem 1^c, the principal minors of the Schur-Cohn matrix of $p_{\tilde{s}}(z) = D(\tilde{s}, z)$ where now $R = R(\tilde{s})$ becomes a matrix with entries dependent on $\tilde{s}$. Therefore, following Siljak [22], the necessary and sufficient conditions for positive definiteness of $R(\tilde{s})$, given by (10) (positivity of all principal minors), are also given by its positive definiteness at a single point on T plus positivity of its determinant for all $s \in T$. Namely, conditions (10) are replaceable by the two conditions (a:=) stability of $D(1, z)$ (because it is equivalent, according to Theorems 1 & 1c, to positive definiteness of $R(\tilde{s})$ at $s = 1$) and (b:=) $c_{[0]0}(\tilde{s}) > 0$, $\forall \in T$. Clearly, (b) implies (b':=) $c_{[0]0}(\tilde{s}) \neq 0$, $\forall\ s \in T$. However in combination with (a) it follows from Theorem 1 that the converse, (a) & (b') $\rightarrow$ (a) & (b), is also true. Therefore (a) and (b) are equivalent to (a) and (b'). Finally the

balanced polynomial in (b') may be dropped because (b') is equivalent to $c_{[0]0}(s) = C_0(s, z) \neq 0$, $\forall s \in T$ (it is recalled that the last polynomial of Algorithm 2 has degree 0 in z). Combining these results with Lemma 1, we have proved the following 2-D conditions for Algorithm 2.

THEOREM 2: **[Stability conditions for Algorithm 2]**. *Assume algorithm 2 is applied to $D(z_1, z_2)$ and denote by $\epsilon(s) \coloneqq C_0(s, z)$ the last polynomial that it produces. $D(z_1, z_2)$ is stable if, and only if, the following three conditions hold.*

$$\begin{aligned}
&(\mathbf{i}) \quad D(z,1) \neq 0 \quad \forall z \in \overline{V} \\
&(\mathbf{ii}) \quad D(1,z) \neq 0 \quad \forall z \in \overline{V} \\
&(\mathbf{iii}) \quad \epsilon(s) \neq 0 \quad \forall s \in T (or \ \epsilon(\tilde{s}) > 0 \ \forall s \in T)
\end{aligned} \tag{11}$$

Note that $\epsilon(s)$ is a symmetric polynomial in s of degree $2n_1n_2$. The examination of condition (11) will sometimes referred as the positivity test. The tabular 2-D stability test consists of Algorithm 2 plus Theorem 2. A numerical illustration for this 2-D test follows.

2.4. First Example (Part 1: Tabular Test)

The current form of the tabular test will be illustrated by a numerical example that was used also in [15]. This numerical example will be used later to illustrate also the simplification of the test. Consider the real 2-D polynomial,

$$D(z_1, z_2) = \begin{bmatrix} 1 & z_1^1 & z_1^2 & z_1^3 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 4 \\ 1 & 2 & 4 & 8 \end{bmatrix} \begin{bmatrix} 1 \\ z_2^1 \\ z_2^2 \\ z_2^3 \end{bmatrix} \tag{12}$$

(For the sake of comparison with [15] we remind that we stated the problem with a slightly different convention. In addition we also scaled the polynomial there to have integer coefficients.) We may test conditions (i) and (ii) of Theorem 2, by the underlying 1-D stability test. Applying Algorithm 1 to $D(z, 1) = D(1, z) = [1, 3, 7, 15]\mathbf{z}$ yields the sequence of polynomials $c_2(z) = [38, 102, 224]\mathbf{z}$, $\boldsymbol{c}_1(z) = [18972, 48732]\mathbf{z}$, $c_0(z) = 8994960$. Setting the leading coefficients in (6) gives $Var\{1, 224, 48732, 8994960\} = 0$. Therefore $D(z, 1) = D(1, z)$ are stable. Next, Algorithm 2 generates the following sequence of matrices.

$$C_2 = \begin{bmatrix} 0 & 0 & 8 \\ 0 & 8 & 20 \\ 8 & 20 & 42 \\ 16 & 40 & 84 \\ 8 & 20 & 42 \\ 4 & 10 & 20 \\ 2 & 4 & 8 \end{bmatrix}, C_1 = \begin{bmatrix} 0 & 64 \\ 64 & 320 \\ 288 & 1056 \\ 912 & 2960 \\ 2120 & 5652 \\ 3440 & 8760 \\ 4424 & 11108 \\ 3520 & 8760 \\ 2320 & 5652 \\ 1252 & 2960 \\ 456 & 1056 \\ 144 & 320 \\ 32 & 64 \end{bmatrix}, C_0 = \begin{bmatrix} 512 \\ 3584 \\ 15744 \\ 55808 \\ 152800 \\ 345824 \\ 666120 \\ 1054544 \\ 1411144 \\ 1582800 \\ 1411144 \\ 1054544 \\ 666120 \\ 345824 \\ 152800 \\ 55808 \\ 15744 \\ 3584 \\ 512 \end{bmatrix}$$

It remains to determine whether the condition $\epsilon(s) \coloneqq C_0^t \mathrm{s} \neq 0 \ \forall s \in T$ holds for this polynomial of degree $2n_1n_2$=18. This can be done using Algorithm 1 and the zero location rule in Theorem 1^c as described in the remark that follows them. Applying Algorithm 1 to the derivative of $\epsilon(s)$ yields a sequence of polynomials (omitted for brevity) whose leading coefficients arranged into (6) gives $Var\ \{7.2090 \cdot 10^7,$

$3.7727 \cdot 10^{13}$, $-2.6888 \cdot 10^{24}$, $2.9103 \cdot 10^{33}$, $-1.8855 \cdot 10^{42}$, $-1.4340 \cdot 10^{50}$, $-3.3009 \cdot 10^{57}$, $2.2488 \cdot 10^{67}$, $-1.1427 \cdot 10^{76}$, $4.3455 \cdot 10^{84}$, $2.8899 \cdot 10^{94}$, $1.6337 \cdot 10^{103}$, $9.1679 \cdot 10^{111}$, $1.2443 \cdot 10^{120}$, $-1.3979 \cdot 10^{130}$, $7.9626 \cdot 10^{139}$, $-1.7445 \cdot 10^{149}\} = 9$. Therefore $\epsilon(s)$ has 9 zeros in V 9 zeros in U and no zeros on T. Therefore the tested 2-D polynomials is stable.

2.5. Count of Operations

We want next to evaluate the computational complexity of the above tabular test. For simplicity, we shall obtain only an approximate count of multiplications by keeping track of only leading terms in polynomial expressions for the precise count (all counts are of polynomial order). The notation $O(n_{1,2}^k)$ will be used to indicate polynomial order with powers $n_1^{\alpha_1} n_2^{\alpha_2}$, $\alpha_1 + \alpha_2 \leq k$. Multiplication of two polynomials of degrees ℓ_1 and ℓ_2 (convolution of their coefficients) requires approximately $(\ell_1 + 1) \times (\ell_2 + 1)$ operations. The division (deconvolution) of a polynomial of degree $\ell_1 + \ell_2$ by a polynomial of degree ℓ_2 (known to be its factor) requires $\ell_1(\ell_1 + 1)/2$ multiplications (counting into it also ℓ_1 divisions). The calculation of $C_{n_2}{-}k$ at step k of Algorithm 1 ($k = 2, \ldots n_2{-}1$) requires, for each of its $(n_2 - k + 1)$ columns, two convolutions of degrees $2kn_1$ by $2kn_1$ followed by deconvolution of degree $4kn_1$ by degree $2(k - 1)n_1$. Summation of these counts over all the steps of the algorithm gives $\frac{5}{2} n_1^2 n_2^4 + O(n_{1,2}^5)$. This remains approximately also the overall complexity for the whole tabular stability test because the arithmetic costs associated with the examination of conditions (i) (ii) (iii) in Theorem 2 (given respectively by $O({n_1}^2)$, $O({n_2}^2)$ and $O(n_1^2 n_2^2)$) are, by comparison, negligible.

The paper [15] motivated the significance of the test in reaching a symmetric polynomial of degree $2n_1n_2$ compared to symmetric polynomial of much higher degree in previously proposed tabular tests. Indeed, all tabular stability tests preceding [15], since the first proposed tabular test of Maria and Fahmy [7] till and including several publications by Hu and coauthors e.g [10] (and references listed therein and in [15]), end with a (symmetric) polynomial of degree $n_1 2^{n_2}$. A more significant achievement of this tabular test is in the reduction of the overall complexity to $O(n^6)$ from previous severe exponential complexity that featured previous tabular tests. This aspect of the achievement has been overlooked because most often tabular tests were presented without a count of the amount of operations they require. A representative estimate for the level of complexity of past tabular tests may be obtained by performing a count of operations for an algorithm that is similar to Algorithm 2 but omits the divisions by the $q_m(s)$ factors. In such a case, the degree of $C_m(s, z)$ would become $(n_1 2^{n_2 - m}, m)$, $m = n_2{-}1, \ldots, 0$. The overall cost can be obtained by summation over all steps of the summation over all columns for a step of two convolutions per column, using the above guideline adjusted to the different column sizes (i.e. with $n_1 2^k$ replacing $2kn_1$). The result is an exponential count with $n_1^2\, 4^{n_2}$ as its domineering term.

More recently, the author has extended his 1-D stability test in [29] and in [30] [31] to 2-D tabular tests in [1], and [14], respectively. These tabular tests use instead of two-term recursion of matrices with no specific structure, three-term recursion of matrices with

certain (centro-)symmetry that allows the computation of only half of their entries. They may be regarded as the *immittance* counterpart of the *scattering* type tabular test here and in [15]. (The immittance formulation stems from the formulation in [30] [31] and was found to produce better algorithms for also several other classical signal processing algorithms related to the Schur-Cohn algorithm, see [32] and references there in). The immittance tabular tests in [1] [14] have too $O(n^6)$ complexity but taking advantage of the symmetries make them several times more efficient than the tabular test considered here. The idea of using the tests [30] [31] for 2-D stability was explored before also in [33] [11]. However the 2-D "immittance" stability test presented in the latter work is inferior the immittance tabular tests in [1] [14] and even to the tabular test here as it can be shown to be of an exponential order of complexity typical to early generation of 2-D tabular stability tests.

3. Simplification By Telepolation

The simplification proposed makes constructive use of two observations regarding the tabular test presented so far. One is that the count analysis revealed that the $O(n^6)$ complexity of the tabular test is determined by the cost of the table's construction. The second is that according to Theorem 2 the construction of the table has to be carried out only in order to get its last polynomial, distinguished by the notation $\epsilon(s)$.

This section presents a 2-D stability test of $O(n^4)$ overall complexity that is obtained by "telepolation" of the tabular table of the previous section. The idea is to (*tele*scope $\epsilon(s)$ by inter*polation*) of the table, without its full construction. As has been already said the testing of the testing of $\epsilon(s)$ requires only $O(n^4)$ operations. The telepolation involves two more components: the sampling of $\epsilon(s)$ and its reconstruction from these samples. Each of the two components has $O(n^4)$ complexity. Judicious choice of sampling points and an efficient formula to recover $\epsilon(s)$ from them, are used to attain an overall $O(n^4)$ complexity procedure of quite low count of operations.

3.1. Sampling by the Companion 1-D Stability Test

Sample values for $\epsilon(\tilde{s})$ (i.e. the balanced polynomial) for $s \in T$ can be obtained by using Algorithm 1. Recall that the tabular test stems from using Algorithm 1 to implement the testing of the condition $D(\tilde{s}, z) \neq 0 \ \forall \ (s, z) \in T \times \overline{V}$, it follows that application of Algorithm 1 to $p_{\tilde{s}_i}(z) = D(\tilde{s}_i, z)$ may be used to produce $\epsilon(\tilde{s}_i)$. This next corollary describes this important component for the method.

COROLLARY 1: ***Let*** $s_i \in T$ ***and apply the companion 1-D stability test to*** $p_{s_i}(z) = D(s_i, z)$ ***(or*** $q_{s_i}(z) := D(\tilde{s}_i, z) = s^{-n_1/2} p_{s_i}(z)$***). If the 1-D polynomial is not stable then*** $D(z_1, z_2)$ ***is not stable. Else, Algorithm 1 produces at its end*** $\epsilon(\tilde{s}_i) = c_{0,0} = C_0(z)$.

It is clear that applying Algorithm 1 to $qs_i(z)$ produces $\epsilon(\tilde{s}_i)$ because it is exactly how Algorithm 2 was obtained from Algorithm 1. The reason $p_{s_i}(z)$ may be equally used to

get $\epsilon(\tilde{s}_i)$ follows from the fact that (4a) produces for both $q_{s_i}(z)$ and $p_{s_i}(z)$ a same $c_{n-1}(z)$. (This is the argument that also explains why in Algorithm 2 all $\tilde{s}$ can be replaced by s.) Therefore both choices produces the same sequence $c_m(z)$ for $m = n-1, \ldots, 0$. In particular they both reach the same $c_{0,0} = \epsilon(\tilde{s}_i)$. Statement (b) is obvious from (2). It will provides a very useful enhancement for testing 2-D stability by telepolation because it implies that all necessary conditions for 1-D stability (e.g., $c_{m,m} > 0$ $m = n-1, \ldots, 0$) encountered in the process of acquiring sample values are necessary conditions for 2-D stability. Consequently, the procedure will be bundled with frequently occurring necessary conditions for 2-D stability that may be used for earlier identification of an unstable 2-D polynomial.

3.2. The Interpolation Problem

We also need an algorithm to recover $\epsilon(s)$ from sample values of $\epsilon(\tilde{s})$. Denote the entries of the coefficient vector of $\epsilon(s)$ by $\epsilon = [\epsilon_0, \ldots, \epsilon_{2M}]^t$. Let

$$\theta = \frac{2\pi}{2M+1} \quad , \quad w = e^{j\theta}$$

where $j = \sqrt{-1}$. If $\epsilon(\tilde{s})$ is known at $M+1$ values $s_i \in T$ given by

$$b_i = \epsilon(\tilde{s}_i) \quad , \quad s_i = w^{-M+i} \quad , \quad i = 0, 1, \ldots, M,$$

then it can be determined from these values as follows.

$$\epsilon_{M-m} = \frac{b_M + 2\sum_{k=1}^{M} b_{M-k}\cos(mk\theta)}{(2M+1)}, \; m = 0, \ldots, M$$

$$\epsilon_{M+m} = \epsilon_{M-m} \qquad , \quad m = 1, \ldots, M \tag{13}$$

The expression is obtained by a DFT-like formula specialized to the current problem, see the appendix. The cost of determining ϵ from $n_1 n_2 + 1$ values of $b_i = \epsilon(\tilde{s}_i)$ is $n_1^2 n_2^2$ real operations. Since ϵ is required only up to a scaling factor, the division by $2M+1$, that provides an exact reproduction of (5), may be dropped.

3.3. 2-D Stability Testing By Telepolation

The new proposed 2-D stability test consists of putting together the results derived up to this point. It is presented as a 4 step procedure below. In this summary, 'exit' is used to

mark points that admit early termination of the procedure because an indication that "$D(z_1, z_2)$ is not stable" has already been found.

A procedure for testing stability of $D(z_1, z_2)$ (D is real).
Step 1$^{(r)}$. Determine whether $D(z, 1)$ is 1-D stable. If not stable - 'exit'.
Step 2$^{(r)}$. Set $M = n_1 n_2$, $\theta = \frac{2\pi}{2M+1}$, $w = e^{j\theta}$.

For $i = 0,1,\ldots, M$ do: Set $s_i = w^{-M+i}$. Apply the companion 1-D stability (Algorithm 1 + Theorem 1) to $p_{s_i}(z) = D(s_i, z)$. If a $c_{m,m} \leq 0$ $m = 1, 2, \ldots$ is detected (implying $p_{s_i}(z)$ is not 1-D stable) - 'exit'. Otherwise, retain the last element as $b_i := c_{[0],0}$ (> 0).

Step 3$^{(r)}$. Use (13) to obtain $\epsilon(s) = \sum_{i=0}^{2M+1} \epsilon_i s^i$ from the values b_i $i = 0,\ldots, M$,
Step 4$^{(r)}$. Examine the condition "$\epsilon(s) \neq 0$ $\forall s \in T$". $D(z_1, z_2)$ is stable if and only if this condition is true and the current step has been reached without an earlier 'exit'.

Note that condition (ii) of Theorem 1 is checked at step 2 because $s_i = 1$ is one of the interpolation points.

3.4. First Example (Part 2: Telepolation)

We now use the previous numerical example (12) to illustrate the testing of $D(z_1, z_2)$ by telepolation.

Step 1: $D(z, 1)$ is stable. Its testing using the companion 1-D stability test was demonstrated in the first part of this example. Step 2: For this example, $M = n_1 n_2 = 9$, $\theta = 2\pi/19$, $w = e^{j\theta} = 0.9458 + j0.3247$. We need samples of $\epsilon(\tilde{s})$ at values $s_i = w^{-9+i}$ for $i = 0,\ldots, 9$. Each value is obtained by application of Algorithm 1 to $D(s_i, z)$. For $i = 0$, $s_0 = w^{-9} = -0.98636 - j0.1646$, Algorithm 1 is applied to $p_{s_0}(z) = [1, s_0, s_0^2, s_0^3]D\mathbf{z} = [-0.87947 - j0.47595, \ -0.81313 - j0.62720, \ -2.6126 - j1.4190, \ -4.2252 - j2.8380]\mathbf{z}$. The algorithm gives the sequence of polynomials $c_2(z) = [2.2425 + j0.3469, \ 14.052 - j1.2544, \ 24.907]\mathbf{z}$, $c_1(z) = [318.92 - j38.93, \ 615.20]\mathbf{z}$, $c_0(z) = 11050.875$. Thus $p_{s_0}(z)$ is stable according to Theorem 1 and, by Corollary 1, $c_0(z)$ provides $\epsilon(\tilde{s}_0) = b_0 = 11050.875$. Performing the companion 1-D tests $D(s_i, z)$ for also $i = 1,\ldots, 9$ determines all these polynomials as 1-D stable and provides the remaining required sample values as follows: $b_1 = 113162/5$, $b_2 = 276373/7$, $b_3 = 132379/3$, $b_4 = 363276/7$, $b_5 = 143432$, $b_6 = 700668$, $b_7 = 2809553$, $b_8 = 6716279$, $b_9 = 8994960$. (Note that the acquisition of b_9 coincides with the stability test of $D(1, z)$ detailed in testing (12) by the tabular test in Part 1 of this example.) Step 3: Obtain the vector ϵ from the above values of b_i $i = 0,\ldots, 9$ using (13). The resulting ϵ is identical to that obtained in Part 1 of this numerical example. For example, the central value ϵ_9 of the vector ϵ may be easily checked to be given indeed by $\epsilon_9 = (b_9 + 2\sum_1^8 b_i)/19 = 1582800$. Step 4: It remains to determine whether the condition $\epsilon(s) \neq 0$ $\forall s \in T$ holds. This condition was already determined to be true in

Part 1 of this example. Therefore (12) is found to be stable also by the telepolation of the table.

4. Complex 2-D Polynomials

This section brings generalizations for the methods of the last two sections to test the condition (1) for a 2-D polynomial with complex coefficients followed by some comments on carrying out the test for $\epsilon(s)$.

4.1. The Complex Tabular Test

The derivation of the 2-D tabular test relies on the 1-D companion stability test and the simplifying conditions of Lemma 1. They both are not limited to real polynomials. Subsequently, the derivation has not used any property that is restricted to the real case. In other words, Algorithm 2 and Theorem 2 were already presented in a manner that holds equally for also a D with complex entries. All that is needed is to read the statements with attention to the fact that our definition of the $\sharp$ operation presents not just reversion but also complex conjugation.

4.2. Telepolation of the Complex 2-D Table

We consider now the simplification of the tabular test by telepolation in the complex case. On transition to the complex case all $c_{[m]m}(s)$ become symmetric in the Hermitian sense. In particular $\epsilon(s)$ becomes complex valued with $\epsilon = J\epsilon^{\star}$. This means that $\epsilon(\tilde{s})$ is still real valued for all $s \in T$ but it gets different values for s_i and $s_i^{\star}$. Two adjustments of the telepolation procedure are required for the complex case: (i) Algorithm 2 has to be run $2M + 1$ rather then just $M + 1$ times ($M := n_1 n_2$) (values of $b_i = \epsilon(\tilde{s}_i)$ for $i = M + 1, \ldots, 2M$ are also required) and (ii) The previous formula (13) to recover ϵ that assumed real vector ϵ has to be replaced by a complex version given by (14) below. The derivation of this formula appears too in the appendix. (Actually, the appendix obtains first (14) then shows that it deduces to (13) in the real case.) The resulting complex version of the procedure may be summarized in 4 steps that correspond the previous steps for the real case.

A procedure for testing stability of $D(z_1, z_2)$ (D is complex).
Step $1^{(c)}$. Determine whether $D(z, 1)$ is 1-D stable. If not stable - 'exit'.
Step $2^{(c)}$. Set $M = n_1 n_2$, $\theta = \frac{2\pi}{2M+1}$, $w = e^{j\theta}$.

For $i = 0,1,\ldots, 2M$ do: Set $s_i = w^{-M+i}$. Apply the companion 1-D stability (Algorithm 1 + Theorem 1) to $p_{s_i}(z) = D(s_i, z)$. If a $c_{m,m} \leq 0$ $m = 1, 2, \ldots$ is detected (implying $p_{s_i}(z)$ is not 1-D stable) - 'exit'. Otherwise, retain the last element as $b_i := c_{[0],0}(>0)$.

Step 3$^{(c)}$. Use b_i $i = 0, \ldots, 2M$ to obtain $\epsilon(s) = \Sigma_{i=0}^{2M+1} \epsilon_i \, s^i$ by the next formula

$$\epsilon_{M-m} = \frac{b_M + \sum_{k=1}^{M} [(b_{M+k} + b_{M-k})\cos(mk\theta) + j(b_{M+k} - b_{M-k})\sin(mk\theta)]}{2M+1},$$

$$m = 0, \ldots, M$$

$$\epsilon_{M+m} = \epsilon^{\star}_{M-m} \quad m = 1, \ldots, M \tag{14}$$

Step 4$^{(c)}$. Examine the condition "$\epsilon(s) \neq 0 \ \forall s \in T$". $D(z_1, z_2)$ is stable if and only if this condition is true and the current step has been reached without an earlier 'exit'.

The following changes occur in the transition to the complex case. Step 1$^{(c)}$ deals now with complex instead of real polynomials. Step 2$^{(c)}$ requires M more repetitions of Algorithm 2. In step 3$^{(c)}$ the solution of the interpolation uses a complex version of the recovery formula. Finally, in step 4$^{(c)}$ the tested polynomial becomes complex and possesses the symmetry $\epsilon = Je^{\star}$.

4.3. Second Example

The testing of a complex coefficient 2-D polynomial by the the tabular test and by the telepolation procedure is illustrated in this second example by considering the next 2-D polynomial.

$$D(z_1, z_2) = \begin{bmatrix} 1 & z_1^1 & z_1^2 & z_1^3 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 2+1j \\ 0 & 1 & 2+1j & 4+2j \\ 1 & 2+1j & 4+2j & 8+j4 \end{bmatrix} \begin{bmatrix} 1 \\ z_2^1 \\ z_2^2 \\ z_2^3. \end{bmatrix} \tag{15}$$

Tabular test. The first two condition of Theorem 2, require stability of $D(z, 1) = D(1, z) = [1, 3 + 1j, 7 + 3j, 15 + 7j]\mathbf{z}$. Testing it with the companion 1-D stability test, Algorithm 1 yields the sequence $c_2(z) = [45-3j, 123 - 3j, 273]\mathbf{z}$, $c_1(z) = [28035 - 585j, 72495]\mathbf{z}$, $c_0(z) = 16370775$. Setting the leading coefficients into (6) gives $Var\{1, 273, 72495, 16370775\} = 0$. Therefore the tested 1-D polynomials are stable. Next, the 2-D table is constructed using Algorithm 2,

$$C_2 = \begin{bmatrix} 0 & 0 & 8-4j \\ 0 & 8-4j & 24-2j \\ 8-4j & 24-2j & 52-1j \\ 20 & 50 & 105 \\ 10 & 25 & 52+1j \\ 5 & 12+1j & 24+2j \\ 2+1j & 4+2j & 8+4j \end{bmatrix}, C_1 = \begin{bmatrix} 0 & 48-64j \\ 48-64j & 368-224j \\ 344-192j & 1384-512j \\ 1212-416j & 4092-1056j \\ 3006-608j & 8383-874j \\ 5200-330j & 13400-445j \\ 6810-100j & 17145 \\ 5400+80j & 13400+445j \\ 3470+285j & 8383+874j \\ 1743+424j & 4092+1056j \\ 606+208j & 1384+512j \\ 172+96j & 368+224j \\ 24+32j & 48+64j \end{bmatrix}, C_0 = \begin{bmatrix} 128-704j \\ 3008-4544j \\ 18336-16848j \\ 75504-50272j \\ 236744-106692j \\ 580028-177404j \\ 1174762-237791j \\ 1939795-208560j \\ 2656280-118245j \\ 3001605 \\ 2656280+118245j \\ 1939795+208560j \\ 1174762+237791j \\ 580028+177404j \\ 236744+106692j \\ 75504+50272j \\ 18336+16848j \\ 3008+4544j \\ 128+704j \end{bmatrix}$$

The condition $\epsilon(s) = C_0^t \mathbf{s} \neq 0 \forall\, s \in T$ may again be examined by applying Algorithm 1 to the derivative of $\epsilon(s)$. The algorithm exhibits no further singularity for also this example and at its end, the rule (6) shows 9 sign variations. It follows that $\epsilon(s)$ (a symmetric polynomial of degree 18) has no zeros on the unit circle. Therefore (15) is stable.

Telepolation. Step 1: $D(z, 1) = [1, 3 + 1j, 7 + 3j, 15 + 7j]\mathbf{z}$ is stable. Its stability was checked by the companion 1-D test above. Step 2: We have $M = n_1 n_2 = 9$. Therefore, like in the real polynomial examples, $\theta = 2\pi/19$ and $w = e^{j\theta} = 0.9458 + j0.3247$. This time, $\epsilon(\tilde{s})$ has to be sampled at $s_i = w^{-9+i}$ for $i = 0, \ldots, 18$ (unlike the same degree real case example that required just the first 10 values). Each value is obtained by application of the companion 1-D test to $D(s_i, z)$. All the 19 1-D polynomials are found to be 1-D stable and yield the vector of sampled values $b = [b_0, \ldots, b_{18}] = [220813/6, 1018964/15, 144631/2, 958279/20, 176972/3, 280637, 1721129, 6616777, 13936722, 16370775, 11118402, 4610246, 1344871, 387246, 329303/2, 581047/6, 472019/9, 76075/3, 20439]$. Testing

the stability of $D(1, z)$ by the companion 1-D test in the first part of this example illustrates also the acquisition of the value the value $b_9 = 16370775$. The details on the remaining sample values, that involve 1-D stability tests of complex polynomials, are omitted. Step 3: The vector ϵ is obtained from the above values of b_i $i = 0, \ldots, 18$ using (14). The vector ϵ so obtained is identical with the above C_0. Step 4: As already said in the first part of this example, checking the condition $\epsilon(s) \neq 0 \forall s \in T$ using Algorithm 1 as a zero location procedure works well for this example and shows that this condition holds. Therefore the (15) is stable.

4.4. Testing a Balanced Symmetric Polynomial for Unit Circle Zeros

The testing of the condition $\epsilon(s) \neq 0 \; \forall \; s \in T$ where $\epsilon(s) = [\epsilon_0, \epsilon_1, \ldots \epsilon_{2M}]\mathbf{s}$ is a symmetric polynomial, i.e., $\epsilon = \epsilon^\sharp$ is a central task common to both the tabular test and its telepolation. Here is a brief account on some possible ways to examine this condition.

Since $\epsilon(s)$ is symmetric, $\epsilon(\tilde{s})$ is real for $s \in T$. In addition it is known that at $s = 1$ it is positive (the latter follows from condition (ii) of Theorem 2 via Theorem 1). So the condition amounts to positivity of $\epsilon(\tilde{s})$ on T. When D is real, then $\epsilon = J\epsilon$ is a real vector. Setting $s = e^{j\theta}$, the requested condition may be expressed by

$$\epsilon(\tilde{s}) = \epsilon_M + 2\sum_{k=0}^{M-1} \epsilon_k \cos((M-k)\theta) > 0 \quad \forall \theta \in [0, \pi] \tag{16}$$

This expression may be used to test the positivity condition numerically by sketching the implied graph. When D is complex, $\epsilon = J\epsilon^\star$ is complex the expression (16) has to be replaced by

$$\epsilon(\tilde{s}) = \epsilon_M + 2\sum_{k=0}^{M-1} \epsilon_k^R \cos((M-k)\theta) + \epsilon_k^I \sin((M-k)\theta) > 0 \quad \forall \theta \in [0, 2\pi]. \tag{17}$$

where $\epsilon_k = \epsilon_k^R + j\epsilon_k^I$ and can again be used to examine the condition graphically.

Algorithm 1 and Theorem 1^c may be used to test $\epsilon(s)$ in a manner described in Remark 1 as long as the algorithms does not encounter a singularity. Other scattering-type algorithms may similarly be used where [21] brings for them generalization to complex polynomials and to the zero distribution determination to the extent that they do not encounter subsequent singularities. In particular B-type tests (that feature a single multiplier per recursion step require only $0.5n^2$, real multiplications, for testing a real polynomial of degree n. Methods to overcome singularities were presented for B-type tests in [25] [34] (real polynomials) and [35] (also complex polynomials) but not yet for the other types of tests. In particular amendment of the type C method that consist of Algorithm 1 and zero location rules to handle also the not strongly regular case will enhance the nicety of the telepolation procedure here having all parts of the procedure be handled by a single algorithm.

The method that was suggested for the positivity test in [15], and one that has been often mentioned for this task, is a technique proposed in [18]. It transforms $\epsilon(s)$ to the variable to $x = (s + s^{-1})/2$. When $\epsilon(s)$ is real, this transformation corresponds to the substitution $\cos(m\theta) = T_m(x)$ in (16) where $T_m(x)$ is the $m-th$ degree Chebyshev polynomials. After expanding this Chebyshev series into a power series, $\epsilon(s)$ is transformed into a polynomial of degree M in x. This polynomial has then to be tested for no zeros in the real interval $[-1, 1]$. This task can be done by construction of a Sturm sequence and applying Sturm's theorem. The creation of the Sturm sequence for a polynomial of degree M requires approximately M^2 operations. The transformation from the polynomial in s to a polynomial in x requires additional computation and may decrease numerical accuracy. The generalization of this technique for the complex case introduces second-kind Chebyshev polynomials which are implied by the $sin(m\theta)$ terms in the expression (17). Therefore a transformation to a polynomial in x is not possible.

An attractive alternative way is to test the condition $\epsilon(s) \neq 0 \ \forall s \in T$ by the zero location method in [30] [31] or [29]. This procedure requires less computation than transformation to a new variable, or than any test in the SCMJ class of methods. It is applicable for both real and complex polynomials and it offers means to overcome all the possible singularities.

5. Computation Issues

An approximate count of operations for the reduced complexity procedure will now be carried out (for both the real and the complex case). The count is in terms of real multiplications (with real times complex and complex times complex counted as two and four operations, respectively). 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. Step 2 involves $n_1n_2 + 1$ repetitions of Algorithm 2 in the real case and $2\ n_1n_2 + 1$ in the complex case. Algorithm 2 requires for a polynomial of degree n, $n^2 + O(n)$ operations when it is real and $3n^2 + O(n)$ real operations when it is complex. The polynomials in step 2 are complex (with one exception: $p_{s_0}(s)$ is real when D is real) and of degree n_2. Therefore step 2^r requires $3n_1n_2^3 + O(n_{1,2}^3)$ operations and step 2^c $6n_1n_2^3 + O(n_{1,2}^3)$ operations. Step 3^r and 3^c require $n_1^2n_2^2$ and $2n_1^2n_2^2 + O(n_{1,2}^3)$ operations, respectively. If step 4 is carried out too by Algorithm 1 then it requires $4n_1^2n_2^2 + O(n_{1,2}^3)$ and $12n_1^2n_2^2 + O(n_{1,2}^3)$ for the real and complex case, respectively. The resulting overall complexity of the test is $5n_1^2n_2^2 + 3n_1n_2^3 + O(n_{1,2}^3)$ real operations for real D and $12n_1^2n_2^2 + 6n_1n_2^3 + O(n_{1,2}^3)$ real operations for a complex D. Note that the count suggests that when $n_1 \neq n_2$ it is worthy to arrange n_2 to be the lower degree (i.e. replace D by D^t when $n_2 > n_1$).

For comparison of the count of operations with other available solutions assume $n_1 = n_2 = n$ for simplicity. As already pointed, tests preceding [15] used to be of severe $O(n^2 4^n)$ complexity. The tabular test [15], as shown here, and the immittance tabular tests in [14] [1] have only $O(n^6)$ complexity. Gu and Lee proposed in [9] to interpolate the Schur-Cohn polynomial matrix by available efficient matrix factorization routines. However they do

not detail a full specific algorithm from which a count of operations could be estimated. Kurosawa, Yamada and Yokokawa proposed in [13] to obtain the determinant of the polynomial matrix using DFT and assess its complexity as $O(n^5)$. Barret and Benidir suggested in [12] a solution that uses a generalized Levinson algorithm to interpolate the resultant matrix and showed that it requires approximately $23n^4 + O(n^3)$ real multiplications and $23.5n^4 + O(n^3)$ real additions. These solutions were presented only for the case of real 2-D polynomial and have higher complexity than the solution for the real case by the current procedure.

Competing alternatives in terms of cost of computation and availability for also the complex case are the simplification by telepolation of the immittance 2-D tabular test [36] [37]. The terms *immittance* and *scattering* are used to distinguish algorithms that stem from the zero location test formulation of Bistritz in [30] [31] from algorithms that stem from the Schur-Cohn test and several other related classical algorithms (see [32] and other references therein). Telepolation of the immittance tabular tests [1] [14] lead too to $O(n^4)$ complexity but, like with the comparison of tabular tests in the two formulations, achieve a lower count of operations because they carry out the task with symmetric polynomials. It is possible to reduce the overall count of operations reported above by using the author's zero location methods of [30] and [31] or [29] to carry out step 4 in ${n_1}^2{n_2}^2$ multiplications and $2n_1^2n_2^2$ additions $+O(n_{1,2}^3)$ for real D and in $4n_1^2n_2^2$ multiplications and $8n_1^2n_2^2$ additions $+O(n_{1,2}^3)$ for the complex D case.

A 2-D stability tests should be evaluated not only by its cost of computation but also in terms of programming complexity and numerical accuracy. The above count ranks the procedure proposed here highly in terms of computational complexity. Its programming is also quite simple, a single module - Algorithm 1 - is used repeatedly in step 2 and it may be used also for steps 1 and 4. These features may be good enough for symbolic computation or for testing polynomials of relatively low degrees. However, in floating point environment, round-off errors may hinder the potential ability of any algebraic stability procedure to give a decisive answer. The early generation of 2-D stability tests of exponential complexity were criticized for this reason in [3], as impractical for all but testing the simplest filters. A recent experimental study on the numerical accuracy of several 2-D stability tests that has been carried out in [38] shows that accuracy tends to improve as the complexity goes down by orders of magnitudes. It is wrong to assume that "squeezing" the count of operations, leads necessarily to better accuracy. Different forms of algorithm of similar levels of complexity may have different numerical behavior depending on design and implementation details that may not manifest itself in count of operations. Nevertheless, for reasons explained in [14], immittance tabular tests might have better numerical behavior than the current scattering tabular. Numerical accuracy becomes a concern also in testing the target polynomial $e(s)$ for no zeros on the unit circle because in this instance a relatively high degree polynomials with typically widely ranged coefficient values has to be processed. Its testing raises a need for also 1-D zero location tests of high numerical accuracy. Thus, numerical accuracy provides a good incentive for further research for better stability tests for both univariate and multivariate polynomials.

6. Conclusion

The paper considered the problem of deciding whether a two-variable polynomial of degree (n_1, n_2), with real or complex coefficients, has no zeros in the closed exterior of the unit bi-circle (is '2-D stable'). First a new form for the tabular test proposed in [15] was obtained. This test consists of a '2-D table' (a sequence of matrices or 2-D polynomials) and testing the last 1-D polynomial of this table for having no zeros on the unit circle. While the testing of this last polynomial has only $O(n^4)$ ($n_1 = n_2 = n$) complexity, the overall complexity of the test was shown to be $O(n^6)$ caused by the computational cost of the construction of the 2-D table. Next a 2-D stability test of overall $O(n^4)$ was obtained by telepolation of the tabular test. This approach uses a set of 1-D stability tests to sample the last polynomial and then recovers it by an interpolation formula. Finally, generalizations of the tabular test and its telepolation to the case of complex 2-D polynomials were also presented.

The new approach shows that testing the stability of a polynomial of degree (n_1, n_2) can be carried out by a finite number (of $n_1n_2 + 1$ for the real case, and $2n_1n_2 + 1$ for the complex case) 1-D stability tests of designated form of degrees n_1 or n_2 plus one zero location test of a 1-D polynomial of degree $2n_1n_2$. This new observation is highlighted in an interesting manner when it is contrasted with an early numerical method to test 2-D stability called the mapping method [3]. The mapping method suggests to test the condition (8) by performing many 1-D stability tests (by numerical calculation of their zeros) using a "dense enough" grid of sample values $s \in T$. Instead, the telepolation approach shows that a definite decision on whether the 2-D polynomial is stable can be reached by using only a well defined *finite* number of 1-D stability tests of some *specific* form.

The fact that the 2-D stability test is spanned by a finite series of 1-D stability tests simplifies considerably the programming of the procedure. It also implies a procedure that is densely filled with necessary conditions (for 1-D stability that are also necessary conditions) for 2-D stability. The latter property reduces even further the amount of computation that is required to determine a 2-D polynomial as not stable.

Appendix: The Interpolation Formulas

The purpose of this appendix is to derive the formulas (13) and (14) for recovering ϵ. As a polynomial of degree $2M$ ($M := n_1\, n_2$), $\epsilon(s) = s^t[\epsilon_0, \ldots, \epsilon_{2M}]$ can be determined from $2M + 1$ values $b_i = \epsilon(\tilde{s}_i)$ at distinct points $s_i \in T$. Therefore, the required $\epsilon = [\epsilon_0, \ldots, \epsilon_{2M}]^t$ can be determined from the solution of the next set of equations that present the collection of $2M + 1$ values of $\epsilon(\tilde{s})$ at distinct points s_i,

$$[s_i^{-M}, s_i^{-M+1}, \ldots, s_i^{-1}, 1, s_i, \ldots, s_i^{M-1}, s_i^M]\epsilon = b_i \quad , \quad i = 0, 1, \ldots, 2M. \tag{A.1}$$

When D is real it is of advantage, as will become apparent in a moment, to choose $2M$ interpolation points in complex conjugate pairs. In addition, choosing the points equally

spaced along T the set with a DFT-like orthogonality property that simplifies its solution. An adequate choice of interpolation points that satisfies both requirements is given by

$$s_i = w^{-M+i}, \qquad i = 0, 1, \ldots, 2M \quad \text{where} \quad \theta = \frac{2\pi}{2M+1} \quad , \quad w = e^{j\theta} \tag{A.2}$$

($j = \sqrt{-1}$). For this choice, the set of equations (A.1) takes the form $Q\epsilon = b$, where $b = [b_0, \ldots, b_{2M}]^t$ is the vector of known interpolation values and Q is given by

$$Q = \begin{bmatrix} w^{MM} & w^{M(M-1)} & \ldots & w^{M} & 1 & w^{-M} & \ldots & w^{-(M-1)M} & w^{-MM} \\ w^{(M-1)M} & w^{(M-1)(M-1)} & \ldots & w^{M-1} & 1 & w^{-(M-1)} & \ldots & w^{-(M-1)(M-1)} & w^{-(M-1)M} \\ \vdots & & & & \vdots & & & & \vdots \\ 1 & 1 & \ldots & 1 & 1 & 1 & \ldots & 1 & 1 \\ \vdots & & & & \vdots & & & & \vdots \\ w^{-(M-1)M} & w^{-(M-1)(M-1)} & \ldots & w^{-(M-1)} & 1 & w^{M-1} & \ldots & w^{(M-1)(M-1)} & w^{(M-1)M} \\ w^{-MM} & w^{-M(M-1)} & \ldots & w^{-M} & 1 & w^{M} & \ldots & w^{(M-1)M} & w^{MM} \end{bmatrix} \tag{A.3}$$

The matrix Q is symmetric, $Q^t = Q$, and also centro-symmetric, $JQJ = Q$. The columns of $Q = [v_{-M}, \ldots, v{-}1, v_0, v_1, \ldots, v_M]$ are given by the vectors

$$v_k := [w^{-Mk}, w^{-(M-1)k}, \ldots, w^{-k}, 1, w^{k}, \ldots, w^{(M-1)k}, w^{Mk}]^t \quad , k = 0, \pm 1, \ldots, \pm M$$

The inner product of two vectors in this set reveal the following orthogonality.

$$v_{-i}^t v_k = w^{-M(k-i)} \frac{1 - w^{(2M+1)(k-i)}}{1 - w^{k-i}} = \begin{cases} 2M+1 & k = i \\ 0 & k \neq i \end{cases}$$

It follows that

$$Q^{-1} = \frac{1}{2M+1} QJ\,. \tag{A.4}$$

Therefore, an explicit solution to $Q\epsilon = b$ is given by

$$\epsilon = \frac{1}{2M+1} QJb \tag{A.5}$$

Using the symmetry $\epsilon = J\epsilon^{\star}$, it suffices to read only the upper half rows of the this solution. The result is the expression in equation (14). If, in addition D is real, then ϵ is real. Setting $J\epsilon = \epsilon$ into (A.1) makes it clear that the value b_i of $\epsilon(\tilde{s})$ at s_i is equal to value b_{2M-i} at $s_{2M-i} = s_i^{-1}$. The resulting relations $b_{M+k} = b_{M-k}$, $i = 0, \ldots, M-1$ simplifies (14) to (13).

References

1. Y. Bistritz, "Stability Testing of Two-Dimensional Discrete Linear System Polynomials by a Two-Dimensional Tabular Form," *IEEE Trans. Circuits and Syst., part I*, vol. 46, June 1999, pp. 666–676.
2. D. M. Goodman, "Some Stability Properties of Two-Dimensional Shift-Invariant Digital Filters," *IEEE Trans. Circuits and Syst.*, vol. 24, Apr. 1977, pp. 201–208.
3. B. T. O'Connor and T. S. Huang, "Stability of General Two-Dimensional Recursive Digital Filters," Ch. 4 in T. S. Huang (Ed.), *Two-dimensional Digital Signal Processing I: Linear Filters*, Springer-Verlag, Berlin 1981.
4. N. K. Bose, *Applied Multidimensional System Theory*, Van Nostrand Reinhold, 1982.
5. D. Dudgeon and R. Mersereau, *Multidimensional Digital Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1984.
6. E. I. Jury, "Stability of Multidimensional Systems and Related Problems" in S. G. Tzafestas (Ed.), *Multidimensional Systems: Techniques and Applications*, New York: Marcel Dekker, 1986.
7. A. Maria and M. M. Fahmy, "On the Stability of Two-Dimensional Digital Filters," *IEEE Trans. Audio and Electroacoust.*, vol. 21, Aug. 1973, pp. 470–472.
8. B. D. O. Anderson and E. I. Jury, "Stability Test for Two-Dimensional Recursive Filters," *IEEE Trans. Audio and Electroacoust.*, vol. AU-21, Aug. 1973, pp. 366–372.
9. G. Gu and E. B. Lee "A Numerical algorithm for stability testing of 2-D recursive digital filter," *IEEE Trans. Circuits and Syst.*, vol. 37, Jan. 1990, pp. 135–138.
10. X. Hu and H. Yee, "Polynomial Array for $F(z_1, z_2)$ on $|z_1| = 1$ and 2-D filter Stability Tests," *IEEE Trans. Signal Process.*, vol. 40, June 1992, pp. 1579–1581.
11. K. Premaratne, "Stability Determination of Two-Dimensional Discrete-Time Systems," *Multidimensional Systems and Signal Processing*, vol. 4, no. 4, 1993, pp. 331–354.
12. M. Barret and M. Benidir, "A New Algorithm to Test Stability of 2-D Digital Recursive Filters," *Signal Processing*, vol. 37, 1994, pp. 255–264.
13. K. Kurosawa, I. Yamada and T. Yokokawa "A Fast 2-D Stability Test Procedure Based on FFT and its Computation Complexity," IEEE Trans. *Circuits and Syst.-II*, vol. 42, Oct. 1995, pp. 676–978.
14. Y. Bistritz, "Immittance-Type Tabular Stability Test for 2-D LSI Systems Based on a Zero Location Test for 1-D Complex Polynomials," *Circuits Systems and Signal Processing*, vol. 19, no 3, 2000, pp. 245–265.
15. X. Hu and E. I. Jury, "On Two-Dimensional Filter Stability Test," *IEEE Trans. Circuits and Syst.*, vol. 41, July 1994, pp. 457–462.
16. E. I. Jury, "Modified Stability Table for 2-D Digital Filter," *IEEE Trans. Circuits and Syst.*, vol. 35, Jan. 1988, 116–119.
17. E. I. Jury, *Theory and Application of the Z-Transform Method*, New York: Wiley, 1964.
18. E. I. Jury, "A Modified Stability Table for Linear Discrete Systems," *Proc. IEEE*, vol. 53, Feb. 1965, pp. 184–185.
19. E. I. Jury, *Inners and the Stability of Linear Systems*, New York: Wiley, 1982.
20. E. I. Jury, "A Note on the Modified Stability Table for Linear Discrete Time System," *IEEE Trans. Circuits and Syst.*, vol. 38, 1991, pp. 221–223.
21. Y. Bistritz, "Reflection on Schur-Cohn Matrices and Jury-Marden Tables and Classification of Related Unit Circle Zero Location Criteria," *Circuits Systems Signal Process.*, vol. 15, no. 1, 1996, pp, 111–136.
22. D. D. Siljak, "Stability Criteria for Two-Variable Polynomials," *IEEE Trans. Circuits and Syst.*, vol. 22, Mar. 1975, pp. 185–189.

23. Y. Bistritz, "On Jury's Test for 2-D Stability of Discrete-Time Systems and its Simplification by Telepolation," *Proc of the IEEE Int. Symp. Circuits and Systems*, 28–31 May 2000, Geneva.
24. H. Lev-Ari, Y. Bistritz, and T. Kailath, "Generalized Bezoutians and Families of Efficient Zero-Location Procedures," *IEEE Trans. on Circ. Syst.*, vol. 38, Feb. 1991, pp. 170–186.
25. R. H. Raible, "A Simplification of Jury's Tabular Form," *IEEE Trans. Automat. Control*, vol. AC-19, June 1974, pp. 248–250.
26. T. S. Huang, "Stability of Two-Dimensional Recursive Filters," *IEEE Trans. Audio and Electroacoust.*, vol. 20, June 1972, pp. 158–163.
27. M. G. Strintzis, "Test of Stability of Multidimensional Filters," *IEEE Trans. Circuits and Syst.*, vol. 24, Aug. 1977, pp. 432–437.
28. E. Curtin and S. Saba, "An Eementary Approach to the Algebraic Stability Theorems," *IEEE Trans. Circuits Syst. -I*, vol. 43, Apr. 1996, pp. 348–349.
29. Y. Bistritz, "A Modified Unit-Circle Zero Location Test," *IEEE Trans. Circuits and Syst.-I*, vol. 43, June 1996, pp. 472–475.
30. Y. Bistritz, "Zero Location with Respect to the Unit Circle of Discrete-Time Linear System Polynomials," *Proc. IEEE*, vol. 72, Sep. 1984, pp. 1131–1142.
31. Y. Bistritz, "A Circular Stability Test for General Polynomials," *Syst. Control Lett.*, vol. 7, no. 2, 1986, pp. 89–97.
32. Y. Bistritz, H. Lev-Ari, and T. Kailath, "Immittance Domain Three-Term Levinson and Schur Recursions for Quasi-Toeplitz Complex Hermitian Matrices," *SIAM J. on Matrix Analysis and Application.*, vol. 12, July 1991, pp. 497–520.
33. K. Premaratne and E. I. Jury, "On the Bistritz Tabular Form and its Relationship with the Schur-Cohn Minors and Inner Determinants," *J. of the Franklin Institute*, vol. 330, no. 1, 1993, pp. 165–182.
34. P. Stoica and R. M. Moses, "On the Unit Circle Problem: The Schur-Cohn procedure revisited," *Signal Processing*, vol. 26, 1992, pp. 95–118.
35. D. Pal and T. K. Kailath, "Displacement structure approach to singular distribution problems: The Unit Circle Case," *IEEE Trans. Automat. Control*, vol. 39, Jan. 1994, pp. 238–245.
36. Y. Bistritz, "A Stability Test of Reduced Complexity for 2-D Digital System Polynomials," *Proc. IEEE Int. Symp. Circuits Syst.*, Monterey, CA, May 1998.
37. Y. Bistritz, "Scattering and Immittance Type Tabular Stability Tests for 2-D Discrete-Time Systems and Their Simplification by Telepolation," in *Proc. IEEE Int. Symp. Circuits Syst.*, Australia, May 2001
38. M. Barret and M. Benidir, "Behaviour of Stability Tests for Two-Dimensional Digital Recursive Filters when Faced with Rounding Errors," *IEEE Trans. Circuits Syst. -II*, vol. 44, Apr. 1997, pp. 319–323.