# A Superfast Algorithm for Determining the Discrete-time Stability for Matrix Polynomials

Vadim Olshevsky
Department of Mathematics and Computer Science
Georgia State University
Atlanta, GA 30303

E-mail: volshevsky@cs.gsu.edu
Web: http://www.cs.gsu.edu/˜matvro

**Abstract**

The classical *Schur-Cohn criterion* for checking the discrete-time stability of a given scalar polynomial $f(z)$ requires $O(n^2)$ arithmetic operations where $n = \deg f(z)$. Recall that the discrete-time stability means that all the roots of $f(z)$ lie in the interior of the unit circle. Though the importance of the Schur-Cohn test is difficult to overestimate, it applies only to *scalar* polynomials, i.e., its area of applications is limited to SISO (single-input-single-output) systems. In this paper we study the stability for the more general *matrix polynomials* of the form $L(z) = z^n I + z^{n-1}A_{n-1} + \ldots + zA_1 + A_0$ that occur in MIMO (multi-input-multi-output) systems. Here the coefficients $A_k$ are $m \times m$ matrices. In this case instead of the roots one usually considers the eigenvalues of $L(z)$ which are defined as complex numbers $\{z_k\}$ for which the $m \times m$ matrix $L(z_k)$ is not invertible. Unfortunately, there are no known efficient (fast or superfast) procedures to check if $L(z)$ is stable. In this extended abstract we present a first superfast $O(mn \log^2 n)$ algorithm to check the discrete-type stability for a given matrix polynomial. Thus, even in the simplest scalar case our algorithm outperforms the classical Schur-Cohn procedure. The derivation of our method is based on a nontrivial reduction of computations on matrix polynomials to computations with dense *block*-structured matrices, and then using the concept of *block*-displacement structure to design a superfast procedure. Recently the method of dispacement was used to design a fast method for list decoding of algebraic geometric codes. Here we apply it to solve an important problem in control. We expect that our method will be useful in several application in digital filtering.

## 0 Introduction

### 0.1 Some history

#### 0.1.1 Continuous-time stability

The classical Lyapunov theorem [L1892] reduces the stability of the equilibrium of a *continuos-time* dynamical system to the condition that the characteristic polynomial of the linearized system has all its roots in the left-half plane $\Pi^-$. Therefore polynomials with all roots in $\Pi^-$ are called stable in the *continuous-time sense*.

The importance of checking the root localization with respect to $\Pi^-$ was made clear before [L1892] in the work of engineers and scientists in the middle of the XIX century. At that time many attempts to build more powerful steam engines faced a problem: unfortunately faster machines showed a clear tendency to vibrate and hence to work unstably. Attempting to overcome the difficulty a famous physicist J.Maxwell [M1868] and a Russian engineer I.Vyshnegradsky [V1876] produced stability conditions for polynomials of degree at most three. This partial progress motivated Maxwell to propose, at a meeting of the London Mathematical Society in 1868, a problem of finding general stability conditions for a polynomial of arbitrary degree. Interestingly, Maxwell was not aware of the fact that even a more general problem has already been completely solved by Hermite in [H1856]. Unfortunately, the Hermite's results were not formulated in terms of efficient algorithms or compact formulas, so they remained unknown to engineers. Hence the solution to the Maxwell's challenge obtained by Routh [R1877] was totally independent of the Hermite's work. Further, it was Hurwitz who adapted in [H1895] the Hermite's work to the form suitable for numerical computations, and the resulting conditions as well as the corresponding algorithm are now called Routh-Hurwitz conditions/algorithm that can be found in many engineering textbooks, see, e.g., [KH97].

### 0.1.2 Discrete-time stability

The dual problem of zero-location with respect to the unit circle has received much less attention, no doubt because of the late interest (since 1950s) in *discrete-time systems*. Similarly to the continuous-time case, the engineers were anticipated by mathematicians. The problem was solved by Cohn [C22] who adapted a previous algorithm of Schur [S17] to check the discrete-time stability. Because of its importance in applications the Schur-Cohn algorithm is typically included into modern texts on digital signal processing, see, e.g., [P97]. It is elegant and concise enough to be briefly formulated in the introduction.

The classical Schur-Cohn algorithm

**Input:** The coefficients of

$$f_n(z) = x_0 z^n + x_1 z^{n-1} + \dots + x_{n-1} z + x_n. \tag{0.1}$$

**Output:** The reflection coefficients $\{\rho_k\}_1^n$.

**Criterion:** $f_n(z)$ is stable if and only if $|\rho_k| < 1$ for all $k = 1, 2, \dots, n$

1 ). Compute the first reflection coefficient by $\rho_1 = \frac{x_n}{x_0^*}$.

2 ). Perform the first step of the *recursive* Schur-Cohn procedure:

$$f_{n-1}(z) = \frac{1}{1-|\rho_1|^2} \cdot \frac{f_n(z) - \rho_1 [f_n(\frac{1}{z^*})]^*}{z}. \tag{0.2}$$

3 ). Proceed with $f_{n-1}(z)$ recursively.

In order to figure out the cost of the Schur-Cohn procedure we need to explain why it terminates after at most $n$ steps. Observing that $[f_n(\frac{1}{z^*})]^* = x_n^* z^n + x_{n-1}^* z^{n-1} + \dots + x_1^* z + x_0^*$ we realize that the recursion (0.2) reduces to

$$f_{n-1}(z) = \frac{1}{1-|\rho_1|^2} \cdot \frac{(x_0 - \frac{x_n}{x_0^*} x_n^*) z^n + (x_1 - \frac{x_n}{x_0^*} x_{n-1}^*) z^{n-1} + \dots + (x_{n-1} - \frac{x_n}{x_0^*} x_1^*) z + 0}{z}.$$

Hence $\deg f_{n-1}(z) = n-1$, and the Schur-Cohn recursion (0.2) has the degree-reduction property. Therefore it terminates after at most $n$ steps (one can stop earlier if a negative $\rho_k$ is obtained). Thus the cost of the standard Schur-Cohn test is $O(n^2)$ arithmetic operations. We next address the issue of speeding up the root localiziation procedure.

### 0.1.3 Main problem

In this extended abstract we consider the more general case of matrix polynomials of the form

$$L(z) = z^n I + z^{n-1} A_{n-1} + \dots + z A_1 + A_0,$$

where the coefficients $A_k$ are $m \times m$ matrices. Such polynomials appear in MIMO systems, see, e.g. [K80]. There is also a deep mathematical theory of matrix polynomials, see, e.g., [GLR82]. Instead of roots one needs to consider the eigenvalues of $L(z)$ which are defined as complex numbers $\{z_k\}$ for which the evaluated polynomial $L(z_k)$ turns into a singular matrix. There are $n \times m$ eigenvalues, counting with multiplicities. Clearly, in the simplest scalar case $m = 1$ the eigenvalues are just the roots. In this paper we propose the first superfast $O(m^2 n \log^n)$ algorithm to check if all the eigenvalues of $L(z)$ lie in the interior of the unit circle. In fact, we can do more: our algorithm computes the number of the eigenvalues inside and outside the unit circle, assuming that there are no eigenvalues symmetric with respect to the unit crcle, like $z_k z_m^* = 1$ (which is of course true if they are all inside).

#### 0.1.4 Structure of the paper

In the next section we reduce the problem of stability checking to the problem of superfast block factorization of a certain structured matrix. To this end we need to recall several deep results in operator and matrix theory associated with the work of M.G.Krein.

## 1 The scalar case: stability, Toeplitz matrices, and Gohberg-Semencul formula.

### 1.1 Inner product on the unit circle, orthogonal polynomials and their Toeplitz matrices

Discrete-time stable polynomials is the main focus of this paper. In this subsection we recall a characterization of these polynomials as of being orthogonal polynomials (we even present more generalized results, since our algorithm will apply to more general polynomials). The main technical result of this subsection is that polynomial stability is related to the inertia of the inverse of a certain Toeplitz matrix. This will be the basis for our superfast algorithm.

A nonnegative measure $W(z)dz$ on the unit circle generates an inner product by the formula

$$< p, q >= \frac{1}{2\pi} \int_{-\pi}^{\pi} p(e^{j\theta}) \cdot [q(e^{j\theta})]^* W(\theta) d\theta. \tag{1.3}$$

For polynomials $p(z) = p_0 + p_1 z + ... + p_n z^n$, $q(z) = q_0 + q_1 z + ... + q_n z^n$, it is easy to express the inner product (1.3) in terms of finite matrices with Toeplitz structure. Indeed,

$$< p, q >= \sum_{m,j=0}^{n} p_m q_j^* t_{m-j} \tag{1.4}$$

where the numbers

$$t_k = \int_{\pi}^{\pi} z^m [z^j]^* W(\theta) d\theta = \int_{\pi}^{\pi} e^{kj\theta} W(\theta) d\theta$$

are called the the moments. The formula (1.4) can be finally rewritten as

$$< p, q >= \begin{bmatrix} q_0^* & q_1^* & \cdots & q_n^* \end{bmatrix} \cdot T_n \cdot \begin{bmatrix} p_0 \\ p_1 \\ \vdots \\ p_n \end{bmatrix}. \tag{1.5}$$

where

$$T_k = [t_{m-j}]_{m,j=0}^{k} \tag{1.6}$$

has the *Toeplitz* structure (i.e., it has constant values along each diagonal, e.g., $t_{2-1} = t_{3-2} = t_{4-3} = \ldots$), and the Hermitian structure ($t_{m-j} = t_{j-m}^*$). The Toeplitz matrix in (1.6) will be the focus of the discussion in this section: its inertia contains the information about the zero location we are interested in.

## 1.2 Inertia of Toeplitz matrices and the zero location of the Szego polynomials

The classical Szego case is when the measure $W(z)$ is positive on the unit circle, so that (1.3) is a true definite inner product: $< p, p > \geq 0$ . In this case $T_n$ in (1.5) must be positive definite as well. Hence by applying the standard Gram-Schmidt process to $\{z^k\}$ one obtains the orthogonal polynomials $< f_k(z), f_m(z) >= 0$ for $k \neq m$. Polynomials $\{f_k\}$ are called the Szego polynomials, and the coefficients of $f_n(z) = x_0 z^n + x_1 z^{n-1} + \ldots + x_n$ can be determined by the Yule-Walker equations

$$T_n \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

The classical result of Szego claims that all the zeros of $f_n$ lie inside the unit circle, i.e., they are discrete-time stable polynomials. The converse statement is also true: any discrete-time stable polynomial can be seen as the Szego polynomial (a more general result appears in theorem 1.2 below).

The above classical results concern a positive measure. In [K66] M.G.Krein considered the case of indefinite measure $W(z)dz$ (in this case $T_n$ in (1.5) is no longer positive definite), and obtained the following two theorems.

**Theorem 1.1** *Let $t_{-n}, ...t_{-1}, t_0, t_1, ..., t_n$ be complex numbers such that the matrices $T_k = [t_{p-q}]_{p,q=0}^k$ $(k = 0, 1, ..., n)$ are Hermitian and nonsingular, and let $\beta$ ( respectively, $\gamma$ ) stand for the number of constances ( respectively, alternations ) of sign in the sequence*

$$1, D_1, D_2, ..., D_{n-1} \qquad (D_k := \det T_k). \tag{1.7}$$

*Let $f_n(z) = x_0 z^n + x_1 z^{n-1} + ... + x_n$ be a polynomial whose coefficients are found from the equation*

$$T_n \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{1.8}$$

*If $D_n \cdot D_{n-1} > 0$ (respectively, $d_n \cdot d_{n-1} < 0$ ), then polynomial $f_n$ has $\beta$ ( respectively, $\gamma$ ) eigenvalues inside the unit circle and $\gamma$ ( respectively, $\beta$)eigenvalues outside the unit circle ( the zeros are counted with multiplicities ).*

Clearly, if there are no sign alternations (so that $T_n$ is positive definite) then all the zeros of $f_n$ lie in the interior of the unit circle.

Theorem 1.1 gives a hope to exploit the properties of $T_n$ to design superfast algorithms for $f_n$. However, in our application we are given a polynomial $f_n$, and we are not given a Toeplitz matrix $T_n$. The next theorem tells us that under mild conditions (that always hold in our application) such $T_n$ always exists.

**Theorem 1.2** *For a polynomial $f_n(z) = x_0 z^n + x_1 z^{n-1} + ... + x_n$ there exists a Hermitian invertible (and strongly nonsingular, i.e., all $d_k \neq 0$) Toeplitz matrix $T_n = [t_{p-q}]_{p,q=0}^k$ such that (0.1) holds true if and only if $x_0$ is real and $f_n$ has no zeros on the unit circle and no pair of zeros that are symmetric with respect to the unit circle.*

Clearly if $T_n$ is invertible, that $T_n$ and its inverse $T_n^{-1}$ have the same number of eigenvalues inside and outside the unit circle. This simple observation has important algorithmic consequences, because as we shall see in a moment, $T_n^{-1}$ is given directly in terms of the coefficients of $f_n$.

## 1.3 Gohberg-Semencul inversion formula

The following result was obtained in [GS72] (see also [GF74]).

**Theorem 1.3** *[ Gohberg-Semencul formula ] Let* $T_n = \begin{bmatrix} t_{p-q} \end{bmatrix}_{p,q=0}^{n}$ *be a Hermitian nonsingular matrix with nonsingular leading submatrix* $T_{n-1}$*. Let*

$$T_n \cdot \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{1.9}$$

*Then* $x_0 \neq 0$ *and*

$$T_n^{-1} = \frac{1}{x_0} \cdot \left\{ \begin{bmatrix} x_0 & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ x_n & \cdots & \cdots & x_0 \end{bmatrix} \cdot \begin{bmatrix} x_0^* & \cdots & \cdots & x_n^* \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & x_0^* \end{bmatrix} - \begin{bmatrix} 0 & 0 & \cdots & & 0 \\ x_n^* & 0 & \cdots & & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ x_1^* & \cdots & \cdots & x_n^* & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & x_n & \cdots & \cdots & x_1 \\ 0 & 0 & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \ddots & x_n \\ 0 & \cdots & & \cdots & 0 \end{bmatrix} \right\}. \tag{1.10}$$

$$T_{n-1}^{-1} = \frac{1}{x_0} \cdot \left\{ \begin{bmatrix} x_0 & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ x_{n-1} & \cdots & \cdots & x_0 \end{bmatrix} \cdot \begin{bmatrix} x_0^* & \cdots & \cdots & x_{n-1}^* \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & x_0^* \end{bmatrix} - \begin{bmatrix} x_n^* & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ x_1^* & \cdots & \cdots & x_n^* \end{bmatrix} \cdot \begin{bmatrix} x_n & \cdots & \cdots & x_1 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & 0 & x_n \end{bmatrix} \right\}. \tag{1.11}$$

## 1.4 Conclusion on how to design a superfast stability test

Let us now return to the problem of zero-location for a given $f_n$. Theorem 1.2 tells us that under mild conditions (we shall discuss these mild limitations in a moment) we can associate a Toeplitz matrix $T_n$ with it. We could run (we will not do it in our algorithm) the symmetric Gaussian elimination algorithm to compute the triangular factorization

$$T_n = L_T D_T L_T^*, \qquad \text{where} \qquad D_T = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & d_n \end{bmatrix}, \tag{1.12}$$

where $d_k$ are exactly those discussed in theorem 1.1, and $L_T$ is lower triangular. Denoting by $\widetilde{I}$ the reverse permutation matrix (all-zero matrix with ones on the main anti-diagonal) and using the obvious property of Toeplitz matrices $\widetilde{I} T \widetilde{I} = \bar{T}$ (where bar denotes taking entriwise conjugates) we obtain

$$T_n^{-1} = L_i D_i L_i^*, \qquad \text{where} \qquad D_i = \begin{bmatrix} \frac{1}{d_n} & 0 & \cdots & 0 \\ 0 & \frac{1}{d_{n-1}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{1}{d_1} \end{bmatrix}, \tag{1.13}$$

(where $L_i$ is a lower triangular matrix easily connected to $L_T$ in (1.12), but the exact form of this connection is not important to us). Now the properties of determinants imply $D_k = d_1 \cdot \ldots \cdot d_k$ (1.7). Hence any algorithm (slow or fast or superfast) computing the factorization (1.13) can be used for checking the zero location of $f_n$. If all $d_k$ are positive we conclude that $f_n$ is discrete-time stable. If we have $\beta$ strictly positive and $\gamma$ strictly negative $d_k$'s we conclude that $f_n$ has $\beta$ zeros inside the unit circle and $\gamma$ zeros outside. If our factorization algorithm applied to the matrix $T_n^{-1}$ given by the formula (1.10) does not run to completion we conclude that $f_n$ either has a zero on the unit circle or it has a pair of zeros symmetric with respect to the unit circle (recall the mild limitations in theorem 1.2). In any case, we are always able to say if $f_n$ is discrete-time stable or not.

In the rest of the paper we design a superfast algorithm to check the stability, and then generalize it to matrix polynomials.

## 2 Displacement structure and superfast algorithm

Many important problems in pure and applied mathematics and engineering can be reduced to matrix computations. Unfortunately, practical circumstances often impose limitations on the use of available standard matrix methods. For example, in many applications the size of the associated matrices is prohibitively large, so the available methods often require an extremely large amount of time. This is one reason why one seeks in various applications to identify special/characteristic structures that can be exploited in order to speed-up computations. Such additional assumptions are often provided by particular physical properties leading to various structured matrices, such as Toeplitz, Hankel, Vandermonde, Cauchy, Pick matrices, and others.

There is a nice approach to design fast algorithms for these matrices based on the concept of *displacement*. The point is that many structured matrices $R$ satisfy the displacement equation of the form

$$\overbrace{\boxed{R}}^{n} - \boxed{A_\pi}\,\boxed{R}\,\boxed{A_\pi} = -\overbrace{\boxed{B_\zeta}}^{\alpha}\,\boxed{C_\pi} \qquad (2.1)$$

or a slightly different equation $A_\zeta R - RA_\pi = -B_\zeta C_\pi$ (used, e.g., in [OP98] in a different application). Here the two matrices $\{A_\pi, A_\zeta\}$ are auxiliary and usually sparse (do not contain information $R$). The two *rectangular* $\alpha \times n$ and $n \times \alpha$ matrices $\{C_\pi, B_\zeta\}$ are called a *generator* of $R$.

If the displacement equation (2.1) (with given $\{C_\pi, A_\pi, A_\zeta, B_\zeta\}$) has the unique solution $R$, then the entire information on $n^2$ entries of $R$ is conveniently compressed into only $2\alpha n$ entries of its generator $\{C_\pi, B_\zeta\}$. Avoiding operations on matrix entries and operating directly on generator allows us to design fast and superfast algorithms. We refer an interested reader, e.g., to [BP94] and to a more recent survey [O97] (see also many references therein). Here mention only a few recent publications not covered in [O97]. In [OP98] the displacement approach allowed us to design a first superfast algorithm for the classical Nevanlinna-Pick problem. In [OS99] we gesigned a superfast algorithm to multiply a confluent Cauchy-like matrix with vectors (a generalization of several algorithms including FFT, convolution and fast multipoint evaluation). This allowed to design a superfast algorithm for tangential confluent rational interpolation problem. In [OS00] we used displacement to design a fast algorithm for list decoding of algebraic-geometric codes (solving a problem raised in [GS98]). There are other applications, e.g., to matrix-vector product [GO94], preconditiong [KO97] among others. In this section we apply the approach to solve the root location problem. The key observation that the matrix $S := T_n^{-1}$ in (1.11) satisfy the equation:

$$S - Z \cdot S \cdot Z^* = \begin{bmatrix} x_0 & x_n^* \\ x_1 & x_{n-1}^* \\ \vdots & \vdots \\ x_{n-1} & x_1^* \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_0^* & x_1^* & \cdots & x_{n-1}^* \\ x_n & x_{n-1} & \cdots & x_1 \end{bmatrix} = G^{(1)} J [G^{(1)}]^*, \qquad (2.2)$$

where $Z$ is the lower shift matrix (the all-zero matrix with 1's on the main subdiagonal).

Here is the description of our recursive divide-and-conquer algorithm for factorization of matrix $S = T_N^{-1}$ of (1.10) and (2.2). The idea is to apply a divide-and conquer technique to the standard Schur complement formula

$$S = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} = \begin{bmatrix} I & 0 \\ S_{21}S_{11}^{-1} & I \end{bmatrix} \begin{bmatrix} S_{11} & 0 \\ 0 & S_2 \end{bmatrix} \begin{bmatrix} I & S_{11}^{-1}S_{21} \\ 0 & I \end{bmatrix}, \qquad \text{where} \qquad S_2 = S_{22} - S_{21}S_{11}^{-1}S_{12}. \qquad (2.3)$$

and to proceed the factorization procedure for $S_{11}$ and for the Schur complement $S_2$. Then after $n \log n$ recursive divide-and-conquer steps we shall obtain the diagonal factor in (1.13) and will be able to make a conclusion about root distribution. If this procedure would be implemented in terms of computations with the entries of $S$ then it would be too expensive. The idea is to avoid computations with matrices and to work with their generators. Thus we are given a generator of $S$ in (2.3) and in order to proceed the recursion we need the generators of $S_{11}$ and of $S_2$. The generator of $S_{11}$ is easily obtained from the one of the entire $S$ by just extraction of the first half of the entries. The generator of $S_{22}$ is given by the formula

$$G^{(2)} = G_{21} - ((I - Z_{22}) \cdot R_{S1} \cdot S_{11}^{-1} - Z_{21}) \cdot (I - Z_{11})^{-1} \cdot G_{11}, \qquad (2.4)$$

see, e.g., lemma 4.2 in [KO97], and the references therein. In order to compute $G^{(2)}$ we need a generator of $S_{11}^{-1}$. This can be done if we have $S_{11}$ factored. For convenience we explain this fact not for $S_{11}$ but for the entire $S$. If we have $S$ factored as in (2.3), then we just invert this factorized representation to obtain.

$$S^{-1} = \begin{bmatrix} I & -S_{11}^{-1}S_{21} \\ 0 & I \end{bmatrix} \begin{bmatrix} S_{11}^{-1} & 0 \\ 0 & S_2^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -S_{21}S_{11}^{-1} & I \end{bmatrix} \qquad (2.5)$$

We next explain how to use the formula (2.5) in order to compute the generator of the inverse of $S$. Let $S$ satisfy (2.2). Then, in accordance to Lemma 3.1 in [KO97]

$$AS + SA^* = G_A J G_A^*, \qquad A = (I + Z)(I - Z)^{-1}, \qquad G_A = \sqrt{2}(I - Z)^{-1} G^{(1)}, \qquad (2.6)$$

which implies

$$S^{-1}A + A^*S^{-1} = (S^{-1}G_A)J(S^{-1}G_A)^*, \qquad (2.7)$$

Let $\widetilde{I}$ denotes the reverse-identity-matrix. From the easily verified identity $A = \widetilde{I}A^*\widetilde{I}$ and (2.7) here we obtain

$$A(\widetilde{I}S^{-1}\widetilde{I}) + (\widetilde{I}S^{-1}\widetilde{I})A^* = (\widetilde{I}S^{-1}G_A)J(\widetilde{I}S^{-1}G_A)^*$$

Finally, using Lemma 3.1 of [KO97] again we obtain

$$(\widetilde{I}S^{-1}\widetilde{I}) - Z(\widetilde{I}S^{-1}\widetilde{I})Z^* = G_{inv} J G_{inv}^*,$$

where

$$G_{inv} = (I - Z)(\widetilde{I}S^{-1}\widetilde{I})\widetilde{I}(I - Z)^{-1}G^{(1)}. \qquad (2.8)$$

The formulas (2.4) and (2.8) are the basis for the following algorithm that performs all manipulations on on the entries of $S$ but on the entries of its generator.

Procedure discrete-time D&C-Factor

**Input:** A generator $G^{(1)}$ of $S$ in (2.2).

**Output:** 1 ). A generator for the Schur complement $S_2 = S_{22} - S_{21}S_{11}^{-1}S_{12}$.

2 ). A generator for the inverse $S^{-1}$.

**Steps:** 1 ). Consider a partition into four matrices of nearly equal sizes. Extract by inspection from (2.2) generators for submatrices $S_{11}, S_{12}, S_{21}$.

2 ). Use a generator for $S_{11}$ and apply the algorithm D&C-Factor to $S_{11}$ to obtain the generator for its inverse $S_{11}^{-1}$.

3 ). Compute a generator for the Schur complement using formulas

$$G^{(2)} = G_{21} - ((I - Z_{22}) \cdot R_{S1} \cdot S_{11}^{-1} - Z_{21}) \cdot (I - Z_{11})^{-1} \cdot G_{11}, \tag{2.9}$$

4 ). Use this generator and apply the algorithm D&C-Factor to $S_2$ to obtain the generator for its inverse $S_2^{-1}$.

5 ). Use generators for $S_{11}^{-1}, S_2^{-1}, S_{12}, S_{21}$ and the formula (2.5) to compute the generator for $S^{-1}$ as

$$G_{inv} = (I - Z)(\widetilde{I}S^{-1}\widetilde{I})\widetilde{I}(I - Z)^{-1}G^{(1)}.$$

It is clear that once we arrive at the level of $1 \times 1$ matrices, we essentially have the required entries of the middle factor $D_i$ of (1.13) at hand.

We finally address the issue of the computational cost of our algorithm. Let us denote by $C(n)$ the arithmetic complexity of the above algorithm for $n \times n$ matrix $S$ and by $M(n)$ complexity of multiplication by a vector for a class of structured matrices including $S$, its submatrices, Schur complements and inverses. Then the analysis of the algorithm D&C-Factor results in the operations count

$$C(n) = [2C(\frac{n}{2}) + 8M(\frac{n}{2}) + O(n)].$$

Therefore the arithmetic complexity of the D&C-Factor algorithm depends upon the estimate $M(n)$. It was shown in [GO94] $M(n) = O(n \log n)$, so that $C(n) = O(n \log^2 n)$.

## 3 Matrix polynomials

We have chosen above to explain our algorithm for the simplest scalar case. Even in this case our $O(n \log^2)$ algorithm is superfast, whereas the classical $O(n^2)$ Schur-Cohn test is just fast. However, all the results are valid for the case of matrix polynomials, but the notations and formulas look more cumbersome. Here we just highlight the main ideas. All the results in Sec. 1 have been carried over to matrix polynomials. Moreover, there is a collection of papers [G88] devoted to different approaches to theorems 1.1 and 1.2. The results can be used to connect the root location of matrix polynomials to *block*-Toeplitz matrices. Further, there is a Gohberg-Heinig formula for inversion of block-Toeplitz matrices. This matrix has the block-displacement structure. Our superfast algorithm can handle this matrix as well, and the corresponding cost is $C(n) = O(m^2 n \log^2 n)$ operations.

## 4 Some difficulties in extending this to the matrix case

Let us consider the more general case of matrix polynomials of the form

$$L(z) = z^n I + z^{n-1} A_{n-1} + \ldots + z A_1 + A_0,$$

where the coefficients $A_k$ are $m \times m$ matrices. Instead of roots one needs to consider the eigenvalues of $L(z)$ which are defined as complex numbers $\{z_k\}$ for which the evaluated polynomial $L(z_k)$ turns into a singular matrix. There are $n \times m$ eigenvalues, counting with multiplicities.

One problem that complicates the issue is that it could be not easy to obtain the analog of matrix $S$ in (2.2). To explain this point we mention that the the analogue of this matrix satisfies a similar equation

$$S - Z \cdot S \cdot Z^* = \begin{bmatrix} A_0 & B_0 \\ A_1 & B_1 \\ \vdots & \vdots \\ A_{n-1} & B_{n-1} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} A_0^* & A_1^* & \cdots & A_{n-1}^* \\ B_0^* & B_1^* & \cdots & B_{n-1}^* \end{bmatrix} = G^{(1)} J [G^{(1)}]^*, \qquad (4.10)$$

where the polynomial

$$L^{\#}(\lambda) = z^n B_n + z^{n-1} B_{n-1} + \ldots + z B_1 + B_0,$$

is obtained from

$$)[L(\lambda)]^* L(\lambda) = L^{\#}(\lambda)[L^{\#}(\lambda)]^*, \qquad (|\lambda| = 1), \qquad (4.11)$$

such that the polynomials $L(\lambda)$ and $L^{\#}(\lambda)$ are relatively coprime. If the polynomial $L(\lambda)$ is stable the relative coprimeness means that $L^{\#}(\lambda)$ is antistable, and the task is to find $L^{\#}(\lambda)$ from $L(\lambda)$ using (4.11). In the scalar case it was an easy task:

$$L^{\#}(\lambda) = z^n [L(\frac{1}{\lambda^*})]^* = x_0^*$$

## 5 Further work

The output of the classical Schur-Cohn algorithm are the reflection coefficients. These numbers are of practical interest by itself in several applications. They are called reflection coefficients in the context of wave propagation though the layered media,a nd in the context of linear estimation they are called partial correlation coefficients. There is a corresponding numerical analysis literature, where the task is to solve the so-called unitary Hessenberg inverse eigenvalue problem. The problem is solved once we have computed $\{\rho_k\}$. We refer to [AGR91] for a fast $O(n^2)$ algorithm, and to [AGR87] for an application to Pisarenko frequency estimation. Another area where one needs $\{\rho_k\}$ is digital filter design, we refer to [ML80] and [R95] where the computed reflection coefficients are used to build lattice-filters. Such filters have became increasingly popular in signal modeling, spectrum estimation, speech processing, adaptive filtering, and other applications. They have many favorable properties, including inherent stability, low noise accumulation in the state vector loop, possibility to suppress quantization limit cycles under simple arithmetic conventions, etc. In fact, our algorithm can be adapted to solve the above problems faster that earlier methods (i.e., $(n \log^2)$ vs. $O(n^2)$). We expect that our algorithm will be used as a building block in many applications in digital filter design and digital signal processing. There are other potential areas of application in mathematics/numerical analysis, e.g., moment problems and Gaussian quadrature.

## References

[AGR87] G.S.Ammar, W.B.Gragg and L.Reichel, *Determination of Pisarenko frequency estimates as eigenvalues of orthonormal matrices*, SPIE vol. **826**, Proc. of the 1987 SPIE conference on Advanced Algorithms and Architectures for Signa Processing II, 143-145 (1987).

[AGR91] G.S.Ammar, W.B.Gragg and L.Reichel, *Constructing a unitary Hessenberg matrix from spectral data*, in Numerical Linear Algebra, Digital Signal Processing and Parallel Algorithms (G.Golub and P.Van Doores, Eds.), 385-595, Springer Verlag, Berlin, 1991.

[BP94] D.Bini and V.Pan, *Polynomial and Matrix Computations*, Volume 1, Birkhauser, Boston, 1994.

[C22] A.Cohn, Ü*ber die Anzahl der Wurzeln einer algebraischen Gleichung in einem Kreise*, Math. Zeit, **14** (1922), 110-148.

[G60] F.R. Gantmacher, The Theory of Matrices Chelsea 1960.

[G88] Orthogonal matrix-valued polynomials and Applications. (edited by I.C.Gohberg), OT34, Birkhauser Verlag, 1988.

[GLR82] I.Gohberg, P.Lancaster and L.Rodman, *Matrix Polynomials*, Academic Press, New York, 1982.

[GF74] I.Gohberg and I.Feldman, *Convolution equations and projection methods for their solutions*, Translations of Mathematical Monographs, **41**, Amer. Math. Soc., 1974.

[GO94] I.Gohberg and V.Olshevsky, *Complexity of multiplication with vectors for structured matrices*, *Linear Algebra and Its Applications*, **202** (1994), 163-192.

[GO94a] I.Gohberg and V.Olshevsky, *Fast state space algorithms for matrix Nehari and Nehari-Takagi interpolation problems*, Integral Equations and Operator Theory, **20**, No. 1 (1994), 44 – 83.

[GS72] I.Gohberg and A.Semencul, *On the inversion of finite Toeplitz matrices and their continuous analogs (in Russian)*, Mat. Issled., **7(2)** (1972), 201-233.

[GS98] V. Guruswami and M. Sudan, *Improved decoding of Reed-Solomon and algebraic-geometric codes*, Proceedings of the 39th IEEE Symposium on Foundations of Computer Science, 28–37, 1998.

[H1856] C.Hermite, *Sur le nombre des racines d'une* é*quatio agebrique comrise entre des limites donn*é*es*, J. reine angew. Math., **52** (1858), 39-51.

[H1895] A.Hurwitz, Ü*ber die Beningungen, unter welchen eine Gleichung nur Wurzeln mit negativen reelen Teilen besitzt*, Math. Ann. **46** (1895), 533-545.

[K66] M.G.Krein, *Distribution of roots of polynomials orthogonal on the unit circle with respect to a sign-alternating weight*, Theor. Funczii Funczion. Anal. i prolozhen., **2** (1966), 131-137, in Russian.

[K80] T.Kailath, *Linear Systems*, Prenice Hall, Englewood Cliffs, 1980.

[KH97] E.Kamen and B.Heck, *Fundamentals of Signals and Systems*, Prentice Hall, Upper Saddle River, 1997.

[KO97] T. Kailth and V. Olshevsky, *Displacement structure approach to discrete transform based preconditioners of G.Strang type and of T.Chan type,* Calcolo, **33**(1996), 191-208.

[L1892] A.Lyapunov, *Probl*é*me g*é*n*é*ral de la stabilit*é *du mouvement*, Russ. Trad. en Francais ????

[M1868] J.C.Maxwell, *On Governors*, Proc. Roy. Soc. London **16** (1868), 270-283.

[ML80] M.Morf and D.T.Lee, *State-space structure of ladder canonical forms*, Proc. 18th Conf. on Control and Design, Dec. 1980, 1221-1224.

[O97] V.Olshevsky, *Pivoting for structured matrices, with applications*, to appear in Linear Algebra and Its Applications, 53 p.
www.cs.gsu.edu/~matvro

[OP98] V.Olshevsky and V.Pan, *A unified superfast algorithm for boundary rational tangential interpolation problems and for inversion and factorization of dense structured matrices*, *Proc. of 39th Annual Symposium on Foundations of Computer Science (FOCS'98)*, IEEE Computer Society, Los Alamitos, CA, 1998, 192-201.

[OS00] V.Olshevsky and A.Shokrollahi, *Fast matrix-vector multiplication algorithms for confluent Cauchy-like matrices with applications*, to appear in STOC'00.

[OS99] V.Olshevsky and A. Shokrollahi, *A displacement approach to efficient decoding of algebraic-geometric codes*, *Proceedings of the 31st Annual ACM Symposium on Theory of Computing (STOC'99)*, 1999, 235–244.

[P97] B.Porat, *A Course in Digital Signal Processing*, Wiley, New York, 1997.

[R1877] E.J.Routh, *Stability of a Given State of Motion*, London, 1877.

[R89] L.Rodman, An Introduction to Operator Polynomials, Birkhauser Verlag, New York, 1989.

[R95] P.A.Regalia, *Adaptive IIR filtering in signal processing and control*, Marcel Dekker, New York, 1995.

[S17] I.Schur, Ü*ber Potenzreihen die im Inneren des Einheitskreises beschränkt sind*, Journal für die Reine und Angewandte Mathematik, **147**: 205-232 (1917). English translation in *Operator Theory : Advances and Applications* (I.Gohberg. ed.), vol. **18**: 31-88 (1986), Birkhäuser, Boston, 1986.

[V1876] J.Vyshnegradsky, *Sur la théorie générale des régulateurs*, Compt. Rend. Acad. Sci Paris, **83** (1876), 318-321.