Applying the Polynomial Toolbox for MATLAB in Mobile Communication Problems *

Martin Hromčík[‡], Michael Šebek[‡], Jan Ježek[†]

[‡] Centre for Applied Cybernetics
Czech Technical University, Prague, Czech Republic

[†] Institute of Information Theory and Automation
Prague, Czech Republic

Abstract

Quite recently the polynomial design methods found a new great field of application outside the control area: the algebraic approach has been used successfully in signal processing and mobile communications. In contrast to the control systems synthesis, polynomials and polynomial matrices with *complex* coefficients are often required when designing filters, equalizers, decouplers and other components of mobile phones for instance.

Polynomial Toolbox for MATLAB admits complex polynomials in most computations, including Diophantine equations and spectral factorizations. As a result, the toolbox appears a suitable tool for rapid prototyping whenever polynomial design routines with complex coefficients are required.

The objective of this report is twofold. First we would like to explain in a clear and popular manner how the complex coefficients arise in technical practice. Based on this motivation, we will present important numerical algorithms for complex polynomials and polynomial matrices and their implementation in the Polynomial Toolbox for MATLAB. The power of the Toolbox will be illustrated by selected numerical examples involving complex coefficients finally.

1 Introduction

Within the control community there is no need to take polynomial matrices with *complex* coefficients into account in fact. That is also why the great majority of research results on polynomial methods, including numerical algorithms, concern just the real case as a rule.

However, the ideas of the algebraic approach to control systems synthesis have recently been successfully applied also in problems out of the control area, namely in signals and communications [7, 8], and many of such problems naturally call for polynomials and polynomial matrices with complex coefficients. Speaking in broad terms, complex entries are able to carry twofold information about particular signal module and phase which is often desirable. For example, several mobile radio communication filtering algorithms rely upon complex polynomials and solutions of related equations [2]. Vibrational systems and filters are additional examples of systems whose models involve complex coefficients [1]. When applying the algebraic design methods in these cases, solutions to linear and quadratic polynomial equations are sought.

2 Complex Coefficients in Communications

In this section the structure of the mobile radio communication channel is presented along with the way the complex valued polynomials are introduced. The facts presented here are mainly adopted from [2].

Basically, there are two reasons for going into complex computations. First, it often appears convenient to code the bitstream to be transmitted into complex numbers and send their real and imaginary parts using the same frequency range. Such a way, the bandwidth reserved for the channel is more efficiently exploited. However, even if the transmitted symbols are real, the amplitude response of the overall communication channel can become asymmetrical around the carrier frequency if interference or multipath propagation occurs, giving rise to a complex transfer function in the baseband representation. These effects will be thoroughly described further.

2.1 The Radio Transmission System

The radio communication channel typically consists of three parts: the transmitter, radio frequency (RF) chan-

*The work of Martin Hromcik and Michael Sebek has been supported by the Ministry of Education of the Czech Republic under contract No. LN00B096. The work of Jan Ježek was supported by the grant agency of the Czech Republic under the contract 102/02/0709.

nel and a receiver. The transmitter accepts data - time series of bits - from a source and transforms it in a form suitable for transmitting via the RF channel, typically to a band-limited continuous-time signal. The channel adds noise to the input signal and acts as a filter on the transmitted data. Therefore, the receiver cannot be just a pure "reverse" of the transmitter but in addition it has to process the received signal to remove distortion introduced by the RF channel.

In the transmitter, the bit stream to be sent is divided into groups of bits which form digital *symbols* - real or complex numbers. By forming symbols, several bits can be transmitted at the same time.

As the signal for the radio channel must be continuous both in time and amplitude, the digital symbols are pulse-shaped first. This pulse-shaping is performed by a lowpass filter with a spectrum efficient impulse response $p(t)$. Such a way, the sequence of symbols $\{u_k\}$ is transformed in a sequence of continuous-time pulses

$$s_b(t) = \sum_{k=1}^{N} u_k p(t - kT_s)$$

where $1/T_s$ is the symbol rate. This signal is real valued whenever the symbols accepted are real and complex otherwise.

After pulse shaping, the modulation follows: the spectrum of $s_b(t)$ (the *baseband signal*) is shifted to a high frequency band suitable for transmission. For both real and complex valued symbols, the modulation can be accomplished by multiplying the baseband signal $s_b(t)$ by a complex-valued carrier $e^{j\omega_c t}$ and transmitting the real part of corresponding signal:

$$s_p(t) = Re\{s_b(t)\}\cos\omega_c t - Im\{s_b(t)\}\sin\omega_c t \qquad (1)$$

The RF channel is a link between the transmitter and the receiver. The passband signal is carried by electromagnetic waves. However, the space between the transmitter and receiver station often contains obstacles such as buildings, mountains, etc. In addition, the waves are reflected and scattered when touching grounds or other objects, namely if the the carrier frequency is high. As a result, signals which traverse different paths reach the receiver at different times. This property is called the delay spread. These effects cause that the transfer function describing the radio channel will in general not be flat within the spectrum of the transmitted passband signal $s_p(t)$ and will distort it accordingly, see Figure 2. Hence the baseband representation of the received signal becomes nonsymmetric around $\omega = 0$ even if the transmitted sequence is real-valued. This implies that the transfer function of the baseband representation of the channel features *complex coefficients*.

It is clear now that the role of the receiver is twofold. First of all, reversed versions of all operations performed by the transmitter are to be applied. In case the channel were ideally perfect, the desired submitted symbols would be exactly recovered.

However, as we tried to explain above, in reality the received and transmitted symbols differ. Hence some additional manipulations with the set of received symbols have to follow to eliminate the imperfections of radio transmission. This problem can be mathematically formulated as an optimization task. For computational reasons, the cost criterion is often chosen as a quadratic form of involved signals. In that case the whole theory of Wiener and Kalman filtering can be directly applied. In addition the overall communication channel is usually described by its transfer function as a ratio of two polynomials. Then the polynomial methods for control systems design can be easily adopted. The main difference compared to the control theory is that *complex* polynomials are often addressed.

2.2 Designing Filters for Mobile Communications

Many results on applying polynomial design methods in filters and equalizers for mobile communications have been achieved by the Signals and Systems Group at the University of Uppsala which also cooperates with the PolyX Ltd. closely. Their algorithms based on polynomial approach for LQ optimal feedforward filters and LQ optimal decision feedback equalizers have been applied by the Ericsson company in their phones for instance. An interested reader is referred to [3], [4], [5] (http://www.signal.uu.se/Publications/pbookch.html) for detailed description of particular procedures.

As one can check, the crucial computational parts of all the papers cited above are the Diophantine equations, often two-sided and symmetric, and polynomial spectral factorizations. And not only scalar complex polynomials but also complex polynomial matrices are of interest to tackle more complex problems of mobile communications involving multiple-antenna arrays for instance [6].

In the next sections we will demonstrate that the Polynomial Toolbox is capable to resolve such advanced tasks. At first the algorithms for complex linear two-sided symmetric equations and complex spectral factorizations are presented in the next section which are implemented in the Polynomial Toolbox for MATLAB.

3 Advanced Algorithms for Complex Polynomials

3.1 Two-sided Symmetric Equations

The problem of the matrix discrete time symmetric equations with complex coefficients has been deeply

studied in [1]. In the same report also a reliable algorithm based on Sylvester matrices was proposed. A reader more interested in this problematic is strongly recommended to read this paper.

In the sequel we will briefly outline the main ideas leading to a reliable numerical method for the continuous time complex polynomials. We will start with the simpler scalar case.

For $a(s)$ and $b(s)$ scalar the concerned equation $a^\star(s)x(s) + a(s)x^\star(s) = b(s)$ reads

$$(\bar{a}_0 - \bar{a}_1 s + \bar{a}_2 s^2 - \cdots + (-1)^{\delta a}\bar{a}_{\delta a} s^{\delta a})(x_0 + x_1 s + \cdots + x_{\delta x} s^{\delta x}) +$$
$$+ (a_0 + \cdots + a_{\delta a} s^{\delta a})(\bar{x}_0 - \bar{x}_1 s + \bar{x}_2 s^2 - \cdots + (-1)^{\delta x}\bar{x}_{\delta x} s^{\delta x}) =$$
$$b_0 + b_1 s + \cdots + b_{\delta b} s^{\delta b}.$$

By inspection, the considered polynomial equation is equivalent to the following set of constant linear matrices for coefficients of $x(s)$:

$$\underbrace{\begin{bmatrix} \bar{a}_0 & & & 0 \\ -\bar{a}_1 & \bar{a}_0 & & \\ \vdots & -\bar{a}_1 & \ddots & \bar{a}_0 \\ \bar{a}_\delta(-1)^\delta & & \ddots & -\bar{a}_1 \\ & \bar{a}_\delta(-1)^\delta & & \vdots \\ 0 & & \ddots & \bar{a}_\delta(-1)^\delta \end{bmatrix}}_{\overline{A_1}} \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_\delta \end{bmatrix}}_{X} +$$

$$+ \underbrace{\begin{bmatrix} a_0 & & & 0 \\ a_1 & -a_0 & & \\ \vdots & -a_1 & \ddots & a_0(-1)^\delta \\ a_\delta & & \ddots & a_1(-1)^\delta \\ & -a_\delta & & \vdots \\ 0 & & \ddots & a_\delta(-1)^\delta \end{bmatrix}}_{A_2} \underbrace{\begin{bmatrix} \bar{x}_0 \\ \bar{x}_1 \\ \vdots \\ \bar{x}_\delta \end{bmatrix}}_{\overline{X}} = \underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_\delta \end{bmatrix}}_{B}$$

Here δ is an integer such that $\delta \geq \max(\delta a, \delta b, \delta x)$. The terms a_i, b_i, and x_i respectively are zeros for $i \geq \delta a$, resp. $i \geq \delta b$, resp. $i \geq \delta x$.

According to the cited paper [1], this set can be rearranged as

$$\underbrace{\left(\begin{bmatrix} Re[A_1] & Im[A_1] \\ -Im[A_1] & Re[A_1] \end{bmatrix} + \begin{bmatrix} Re[A_2] & Im[A_2] \\ Im[A_2] & -Re[A_2] \end{bmatrix}\right)}_{A} \times$$

$$\times \underbrace{\begin{bmatrix} Re[X] \\ Im[X] \end{bmatrix}}_{X} = \underbrace{\begin{bmatrix} Re[B] \\ Im[B] \end{bmatrix}}_{B} \qquad (2)$$

If δ satisfies $\delta \geq \delta b$ then a solution to the above constant matrix equation exists. Particular coefficients of $x(s)$ can be directly distilled from the vector $\boldsymbol{X}$.

The set (2) is obviously redundant. However, linear transformations can be explicitly prescribed which reduce the dimension of the set by employing the fact that $Im[x_i] = 0$ for i even and $Re[x_i] = 0$ if i is odd.

For polynomial matrices the problem of symmetric equation becomes far more complicated. Details can be found in [11] for real case and in [1] in case of discrete time symmetry and complex numbers.

Nevertheless, if we apply additional restrictions on the shape of involved matrices and on their leading or coefficient terms, the above considerations can be formulated in the matrix case too in principle. Namely, if we require $X(s)$ having triangular constant coefficient matrix and A_0 having nonzero leading minors in addition to the stability of A, the solution to the equation (2) is unique and yields the desired polynomial solution $X(s)$. Note that these restrictions fit the conditions on uniqueness of a matrix spectral factor.

A Sylvester algorithm equivalent to that of the previous section can also be derived in similar way. One more preliminary step consists of rearranging the original polynomial equation to a matrix-vector form. Of course the subsequent reductions of the resulting set $\boldsymbol{AX} = \boldsymbol{B}$ are also much more involving. Nevertheless, the reduction process as it has been suggested in [11] in the real case and in [1] for complex discrete time polynomials can be adopted for the complex continuous time case as well.

3.2 Spectral Factorization

We will present two procedures for computing the polynomial spectral factor in this section.

Quite recently a new numerical algorithm addressing the scalar discrete time spectral factorization based on fast Fourier transform (FFT) has been developed by the Polynomial Toolbox authors [10]. Compared to its predcessors, the new method performs better both in terms of numerical accuracy and namely computational speed [10]. Although the algorithm itself was originally proposed for real polynomials, it is based on the theory of complex valued functions and remains valid for complex valued coefficients as well. The FFT algorithm and its application in a practical signal processing problem is the subject of another report downloadable from this WWW site called *FFT Based Algorithm for Spectral Factorization*, see also [19]. An interested reader is advised to see this document to understand the routine and read about upgrading loudspeakers' characteristics by LQ feedforward optimization via this algorithm.

Another approach to spectral factorization is based on the theory of Newton-Raphson iterations and symmetric linear polynomial equations.

We will first illustrate the idea of the Newton-Raphson iterations by a simple example of finding function roots.

Given a real function $f(x)$ of one real variable, the solution to the equation $f(x) = 0$ can be reached under some assumptions by the Newton's iterational process. The recursive formula reads $f(x_{i+1}) = f'(x_i)(x_i - x_{i+1}) = df(x_i, x_i - x_{i+1})$ where $f'(x_i$ and $df(x_i, .)$ mean respectively the derivative and differential of f at the point x_i. The formula features a clear geometrical meaning - the next iterations is computed as the intersection of the tangent to the curve $y = f(x)$ at the point $[x_i, f(x_i)]$ and the $x-$axis.

This approach has been proved successful in much more general situations. For f being a function of many variables, or even if f is a functional, the method stays valid provided some properties of f and of the starting point x_0 are fulfilled. The process features quadratic convergence if successful.

Solving the spectral factorization problem is equivalent to finding a solution to the equation $f(A) = A^\star A - P = 0$ under the constraint of stability. Applying the Newton's scheme, considering $df(A) = A^\star dA + (dA)^\star A$, and replacing dA by $A_i - A_{i+1}$, we come to the formula

$$A_i^\star A_{i+1} + A_{i+1}^\star A_i = P + A_i^\star A_i \qquad (3)$$

for the succeeding iteration A_{i+1}. Moreover, stability and uniqueness of successive A_i's is guaranteed provided the initial A_0 is stable and triangularity of either leading or constant coefficient of all A_i's is required. The proof of this crucial statement was the subject of the reports [14, 15] and [17].

Although only real polynomials and polynomial matrices have been considered in the mentioned papers, the approach can be applied to complex coefficients directly. Hence the only remaining issue is the solution to the symmetric (matrix) polynomial equation with complex entries. Related algorithms are fortunately available now and were discussed in the previous section.

4 Polynomial Toolbox and Complex Coefficients

In the previous sections, the importance of polynomials with complex coefficients in communications was enlightened and numerical algorithms for most advanced computations were presented. Now some particular linear equations and spectral factorization problems involving complex polynomials and polynomial matrices of various degree and size will be resolved using the Polynomial Toolbox for MATLAB.

4.1 Diophantine Equations

The Polynomial Toolbox provides nine solvers for various kinds of linear (matrix) polynomial equations. We will concentrate on two in practice most important types: the symmetric equation $A^*X + X^*A = B$ and one-sided Diophantine equation $AX + BY = C$.

First let us create a 3-by-3 polynomial matrix A in variable 'z' of degree 4 with complex coefficients using the Polynomial Toolbox `prand` command:

```
>> A = prand(5,5,'z') + prand(5,5,'z')*j
Polynomial matrix in z: 5-by-5,  degree: 5 A =
  Column 1
     1.6-0.046i + (0.28-0.3i)z + (1.5+0.42i)z^2 + ...
     0.45-2.5i + (1.3-0.19i)z + ...
     0.21+0.22i + (-0.082-0.23i)z + ...
     1-0.39i + (-0.1-1i)z + (0.22+0.085i)z^2 + ...
     0.27-0.24i + (-0.88-0.63i)z + ...
  Column 2
    -0.43-0.22i + (2-0.69i)z + (-1.1+0.27i)z^2 + ...
     0.83+0.88i + (-0.058-0.81i)z + ...
    -0.45-0.24i + (0.66+0.25i)z + (0.6+1.1i)z^2 + ...
     0.76-0.081i + (-0.001+0.36i)z + ...
     0.29+0.72i + (-0.66+0.23i)z + (1.8+0.71i)z^2 + ...
  Column 3
    -1.5+0.36i + (0.84-1.4i)z + (-2+2.4i)z^2 + ...
     2.4+0.77i + (0.17-2.5i)z + (-0.18-1.5i)z^2 + ...
    -0.026-1.5i + (0.96+1.6i)z + (-1.4-0.36i)z^2 + ...
     0.34-0.85i + (0.066+0.98i)z + ...
    -1.9-0.37i + (0.61-0.069i)z + (0.4+1.1i)z^2 + ...
  Column 4
    -0.011-0.9i + (-1.1-2.1i)z + (0.12+0.77i)z^2 + ...
     0.1+1.6i + (0.65+0.38i)z + (-0.026+0.39i)z^2 + ...
     0.2+0.19i + (0.41-2.9i)z + (-0.12+0.28i)z^2 + ...
    -2.5+0.88i + (-0.29-0.63i)z + (-0.5+1i)z^2 + ...
     1.4-1.5i + (0.53+2.3i)z + (0.16+0.15i)z^2 + ...
  Column 5
    -0.81-0.14i + (0.32-0.98i)z + ...
     0.39+1.4i + (-2-1.3i)z + (1.2-0.16i)z^2 + ...
     1+0.059i + (1.5+0.67i)z + (-1.1-0.92i)z^2 + ...
    -0.68-0.0068i + (0.085+0.37i)z + ...
     0.5+1.5i + (0.37+0.74i)z + (0.41-1.2i)z^2 + ...
```

Similarly, a discrete-time symmetric 3-by-3 matrix B of degree 8 is created:

```
>> B = prand(5,5,'z') + prand(5,5,'z')*j;
>> B = B*B';
```

Solution of related discrete-time symmetric polynomial equation $A^*X + X^*A = B$ can be achieved by calling the Polynomial Toolbox `axxab` command:

```
>> X = axxab(A,B)
Polynomial matrix in z: 5-by-5,  degree: 5
X =
Column 1
 2.8e+002+0i + ...
```

The following check proves accuracy of obtained result:

```
 isequal(A'*X + X'*A, B)
ans =
     1
```

Besides the symmetric equations, also all one-sided Diophantine equation solvers can be addressed by complex polynomial matrices as well. For inastance, having defined complex polynomial matrices A, B, C of degree 5 and size 5-by-5, the solution X, Y of the equation $AX+BY = C$ is reached easily by the following Polynomial Toolbox command (along with a standard check):

```
>> A = prand(5,5,'z') + prand(5,5,'z')*j;
>> B = prand(5,5,'z') + prand(5,5,'z')*j;
>> C = prand(5,5,'z') + prand(5,5,'z')*j;
>> [X,Y] = axbyc(A,B, C);
>> isequal(A*X + B*Y, C)
ans =
     1
```

All linear polynomial equation solvers of the Polynomial Toolbox are based on powerful linear *constant* matrix solvers built into MATLAB. As a result, the Polynomial Toolbox functions for polynomial equations are not only very accurate as it was shown above, but also pretty fast. For instance, each of presented examples did not consume more than 1 second on a PC notebook with Celeron 500 MHz, 64 MB RAM and MATLAB 6.0.

4.2 Polynomial Spectral Factorization

Scalar complex polynomials are considered first. In this case, the FFT based algorithm can be highly recommended for its speed and reliability.

A symmetric polynomial of degree 500 to be factored, positive on the unit circle, is created by the following commands:

```
>> p = prand(250,'z') + j*prand(250,'z');
>> p = p*p';
```

Applying the FFT based routine with 2^{12} Fourier points the spectral factor is received in a fraction of second:

```
>> tic, f = fftspf(p,2^14); toc;
elapsed_time =
    0.2200
```

Correctness checks follow now. Note that some additional Polynomial Toolbox functions and operators are applied to complex polynomials (actually, all operators and standard functions work for complex polynomials and polynomial matrices automatically):

```
>> isstable(f)
ans =
     1
>> norm(f*f'-p)/norm(p)
ans =
  1.2460e-005
```

As one can see, both the stability requirement is met and the equation $ff^\star = p$ is fulfilled with good accuracy. Should the residue be too high for a specific application, there is no problem to take more Fourier points to improve the result:

```
>>  f = fftspf(p,2^17);
>> norm(f*f'-p)/norm(p)
ans =
  3.6538e-009

>>  f = fftspf(p,2^19);
>> norm(f*f'-p)/norm(p) ans =
     0
```

In the latter case, an exact (up to working precision) solution was found.

Let us switch to complex polynomial *matrices* now. We proceed similarly to the scalar case, starting with construction of a 5-by-5 symmetric complex matrix of degree 5:

```
>> P = prand(5,5,'z') + j*prand(5,5,'z');
>> P = P*P'
Polynomial matrix in z: 5-by-5,  degree: 10
Column 1
 2.4e+001+0i + ...
```

For the spectral factorization the `spf` function of the Polynomial Toolbox is used, implementing the Newton-Raphson iterative procedure.

```
>> tic, F = spf(P); toc;
elapsed_time =
    9.8300
```

Standard checks follow:

```
>> isstable(F)
ans =
     1
>> norm(F'*F-P)/norm(P)
ans =
  3.6995e-007
```

Also in this case the solution can be refined. By prescribing a higher desired accuracy (default is 10^{-8}), a more precise solutions are found:

```
>> F = spf(P,1e-10);
>> norm(F'*F-P)/norm(P)
ans =
  1.3141e-009

>> F = spf(P,1e-12);
>> norm(F'*F-P)/norm(P)
ans =
     0
```

Although the spectral factorization as a quadratic problem is much more involving than linear polynomial equations presented in the previous subsection, we can conclude that the Polynomial Toolbox succeeded in factoring relatively large complex polynomials and polynomial matrices at acceptable computational time.

5 Acknowledgements

The authors would like to thank Mikael Sternad from the University of Uppsala, Signals and Systems Department, for most valuable discussions concerning namely the role of complex polynomials, complex polynomial matrices and algebraic design methods in communications.

The work of Martin Hromcik and Michael Sebek has been supported by the Ministry of Education of the Czech Republic under contract No. LN00B096

6 Conclusion

Performance of polynomial equation and spectral factorization solvers included in the Polynomial Toolbox for MATLAB was manifested in this report for complex valued coefficients. Motivation for developing such tools for complex (matrix) polynomials comes from the field of signals and communications as it is shown in the report by an example of a radio communication channel and its optimization. In addition to particular numerical examples, the algorithms used by respective solvers are also described.

References

[1] Henrion D., Ježek J. and Šebek M., *Efficient Algorithms for Discrete-Time Symmetric Polynomial Equations with Complex Coefficients*, Proceedings of the IFAC World Congress, Beijing, China, Vol. D, p. 159-164, July 1999.

[2] Lindbom L., *A Wiener Filtering Approach to the Design of Tracking Algorithms with Applications to Mobile Radio Communication*, PhD. Thesis, Signal Processing Group, Department of Technology, Uppsala University, Sweden, 1995.

[3] M. Sternad and A. Ahlen, *H-2 Design of Model-Based Nominal and Robust Discrete Time Filters*. Chapter 5 in M Grimble and V Kucera, eds: A Polynomial Approach to H-2 and H-infinity Robust Control Design. pp 171-222, Springer-Verlag, London, 1996.

[4] A. Ahlen and M. Sternad, *Derivation and Design of Wiener Filters using Polynomial Equations*. In C T Leondes, ed: Control and Dynamic Systems, Vol 64:Stochastic Techniques in Digital Signal Processing Systems. pp 353-418, Academic Press, New York, NY, 1994.

[5] A. Ahlen and M. Sternad, *Optimal Filtering Problems*. Chapter 5 in K Hunt, ed: Polynomial Methods in Optimal Control and Filtering, pp120-161, Control Engineering Series, Peter Peregrinus, London, 1993.

[6] Claes Tidestav, *The Multivariable Decision Feedback Equalizer, Multiuser Detection and Interference Rejection*. PhD Thesis, Uppsala University, 197p, ISBN 91-506-1371-5, December 1999. http://www.signal.uu.se/Publications/ptheses.html

[7] MacChi O., Adaptive Processing: The Least Mean Squares Approach and Applications in Transmission, John Willey and Sons,Chichester, Great Britain 1995.

[8] Proakis J.G. and Salehi M., *Communication Systems Engineering*, Prentice Hall, Englewood Cliffs, New Jersey, 1994.

[9] Youla D.C., On the Fctorization of Rational Matrices, IEEE Transactions on Information Theory, IT-17, 172-189, 1961.

[10] Ježek J., Hromčík and Šebek M., *Spectral Factorization by Means of Discrete Fourier Transform*, submitted for the 8th IEEE Mediterranean Conference on Control and Education MED 2000, July 17-19, Patras, Greece.

[11] Henrion D., *Reliable Algorithms for Polynomial Matrices*, Ph.D. Thesis, Institute of Information Theory and Automation, Prague, 1998.

[12] Kailath T., *Linear Systems*, Prentice Hall, New Jersey (1980).

[13] Kučera V., *Analysis and Design of Discrete Linear Control Systems*, Academia Prague (1991).

[14] Vostrý Z., *New Algorithm for Polynomial Spectral Factorization with Quadratic Convergence I*, Kybernetika, 11, pp. 415-422, 1975.

[15] Vostrý Z., *New Algorithm for Polynomial Spectral Factorization with Quadratic Convergence II*, Kybernetika, 12, pp. 248-259, 1976.

[16] Ježek J. and Kučera V., *Efficient Algorithm for Matrix Spectral Factorization*,Automatica, vol. 29, pp. 663-669, 1985.

[17] Wilson G.T., *Factorization of the Covariance Generation Function of a Pure Moving Average Process*, SIAM Journal on Numerical Analysis, 6, pp. 1-7, 1969.

[18] Kwakernaak H., Šebek M., *PolyX Home Page*, `http://www.polyx.cz/, http://www.polyx.com/`.

[19] Hromčík M, Ježek J and Šebek M., *New Algorithm for Spectral Factorization Based on FFT and Its Practical Applications*, Proceedings of the European Control Conference 2001, Porto, Portugal, September 4-7, 2001.