The use of Polynomial Toolbox$^{\text{TM}}$ for MATLAB$^{\text{TM}}$ for filter design

Jiří Fajt*, **Michael Šebek**+

*Department of Control Engineering, Faculty of Electrical Engineering,
Czech Technical University
+Centre for Applied Cybernetics, Czech Technical University

Abstract

Polynomial methods, originally developed for the design of control systems, are increasingly utilized in other engineering areas. Polynomial methods have found great use in signal processing and in filter design, new requirements introduced in these fields are that complex polynomial coefficients are often required.

Polynomial Toolbox for MATLAB allows to use complex coefficients in most computational tasks, especially in solving Diophantine equations and spectral factorization.

In this paper, it will be briefly outlined how polynomial methods can be used for filter design. It will be shown how Polynomial Toolbox for MATLAB can be suitably used for such filter design.

1 Introduction

Polynomial methods have been widely used in control engineering since early 70s. They are based on system description by input-output relations [1]. However, transfer functions are not treated as functions of complex variable but as algebraic objects. A system design then only requires algebraic operations with polynomials and polynomial matrices, typically one has to solve algebraic polynomial equations and spectral factorization. Polynomial approach enables to solve many design problems in an elegant way and it gives an engineer good insight into the structure of a designed system.

It was later realized that advantages of polynomial approach can be utilized in other close engineering areas such as in signal processing and in communications. An engineer designing filters for a communication system is often given its transfer function and thus input-output requirements.

There is no need to use complex polynomial coefficients in the control field. In communication systems, it is needed to obtain information about both the amplitude and the phase of signal. This leads to the use of so called complex envelope when processing baseband signal. Therefore polynomials with complex coefficients must be used.

An algorithm for a scalar signal Wiener filter design based on polynomial approach suggested in [5] will be briefly outlined in this paper. If the polynomial approach is to be used efficiently for a filter design, a powerful simulation software for performing demanding algebraic operations with polynomials is a must. Main focus of this paper is to demonstrate that the Polynomial Toolbox for MATLAB [6] is such a powerful design software tool.

Remarks on notation:

Further in the text, the following notation is used:
Backward shift operator q^{-1}, $q^{-1}s(t) = s(t-1)$, forward shift operator q, $q\ s(t) = s(t+1)$.
Given polynomial $P(q^{-1}) = p_0 + p_1 q^{-1} + \ldots + p_n q^{-n}$, conjugate polynomial is defined $P_*(q) = p_0^* + p_1^* q + \ldots + p_n^* q^n$, where p_i^* denotes complex conjugate of p_i.
Polynomial $P(q^{-1})$ is stable (Schur stable) if all zeros of $P(z^{-1})$ are in $|z| < 1$. Note that whenever $P(q^{-1})$ is stable, all zeros of $P_*(q)$ are in $|z| > 1$.

2 Filtering problem

Algorithms for design of filters, predictors and smoothers by polynomial methods are dealt with in [2], [3], [4], [5] in detail. Algorithms concern scalar signals as well as multivariate signals. There are several algorithms for optimal filter design, filters are optimal in the sense of minimizing quadratic error criteria. Polynomial approach based on the evaluation of orthogonality [5] will be shortly outlined here.

Consider a linear and discrete time system, which is stable and time invariant. The system is driven by a vector of stationary white noise $e(t)$, generates a stationary measurement vector sequence $y(t)$ and a vector sequence of stationary desired responses $f(t)$ (fig. 1) [5].

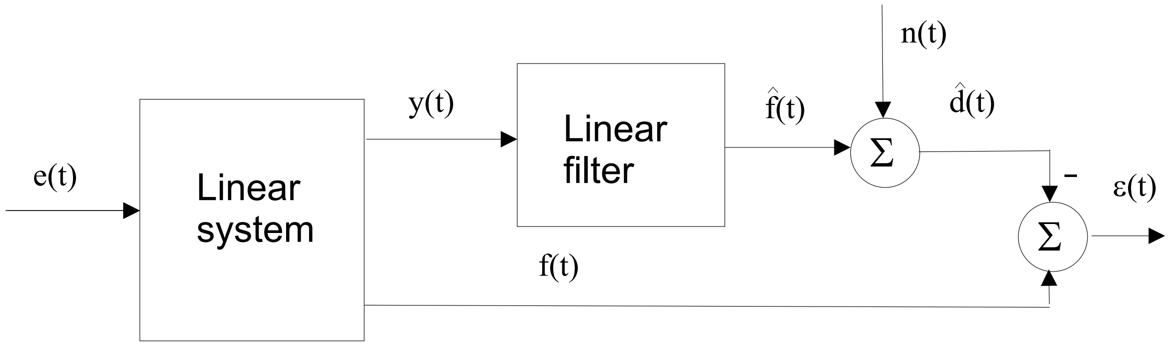

Figure 1: System description, $\hat{f}(t)$ is the estimate of $f(t)$, $n(t)$ is a variational term.

Estimator is defined

$$\hat{d}(t|t+m) = \frac{Q(q^{-1})}{R(q^{-1})} y(t+m) + n(t), \tag{1}$$

stationary signal $n(t)$ represents modification of the estimate.

Optimal filter minimizes error criterion

$$\begin{aligned} \bar{J} &= tr\ E\{f(t) - \hat{d}(t|t+m)\}\{f^*(t) - \hat{d}^*(t|t+m)\} \\ &= tr\{E\epsilon(t)\epsilon^*(t) - E\epsilon(t)n^*(t)\ - En(t)\epsilon^*(t) + En(t)n^*(t)\} \end{aligned} \tag{2}$$

Optimality (minimum $\bar{J}$) is guaranteed when the error $\epsilon(t)$ and any admissible linear function of the measurements $n(t)$ are orthogonal [5].

Algorithm based on orthogonality can be divided into three steps [5].

1. Find rational transfer functions of the system represented by polynomial (matrix) fractions. From the spectral density of $y(t)$ perform spectral factorization.
2. Define estimation error $\epsilon(t)$ and variation of the estimate $n(t)$. Express $E\epsilon(t)n^*(t)$ in frequency domain using Parseval's formula. Simplify it using spectral factorization.
3. Fulfill the orthogonality requirement $E\epsilon(t)n^*(t)$. We get a polynomial equation whose solution gives us the estimator.

3 A scalar signal filter

A system is described by the following ARMA models.

$$\begin{aligned} y(t) &= \frac{C(q^{-1})}{D(q^{-1})}e(t) + w(t) \\ w(t) &= \frac{M(q^{-1})}{N(q^{-1})}v(t) \end{aligned} \tag{3}$$

where $w(t)$ is measurement noise which does not necessarily have to be white, $e(t), v(t)$ are white and have zero mean, $C(q^{-1}), D(q^{-1}), M(q^{-1}), N(q^{-1})$ are stable.

The measurements $\{y(t)\}$ can be described by the innovation model

$$y(t) = \frac{\beta(q^{-1})}{D(q^{-1})N(q^{-1})}(\sqrt{\sigma_n}\eta(t)) \tag{4}$$

where σ_n is the variance of innovation sequence, $\beta(q^{-1})$ is a monic and stable polynomial.

A spectral factorization equation for β is obtained by setting spectral densities from (3) and (4) equal.

$$r\beta(q^{-1})\beta^*(q) = C(q^{-1})C^*(q)N(q^{-1})N^*(q) + \rho M(q^{-1})M^*(q)D(q^{-1})D^*(q) \tag{5}$$

where $r = \frac{\sigma_\eta}{\sigma_e}, \rho = \frac{\sigma_v}{\sigma_e}$ and $\beta(z^{-1})$ is stable.

It is derived in [5] that $R(q^{-1}) = \beta(q^{-1})$ and for $Q(q^{-1}) = Q_1(q^{-1})N(q^{-1})$ and any polynomial $L_*(q)$, a unique solution of the linear polynomial equation [5]

$$q^{-m}C(q^{-1})C^*(q)N^*(q) = r\beta_*(q)Q_1(q^{-1}) + qD(q^{-1})L_*(q) \tag{6}$$

determines the solution for the searched estimator

$$\hat{d}(t|t+m) = \frac{Q_1(q^{-1})N(q^{-1})}{\beta(q^{-1})}y(t|t+m) \tag{7}$$

Solution of (6) is unique because $Q(q^{-1})$ must be causal, ie. it may not contain any terms with positive powers of q, and $L_*(q)$ should not contain any terms with powers of q^{-1}. Solvability conditions of Diophantine equation (6) are thoroughly examined in [2], [3], [4], [5].

4 Solution using Polynomial Toolbox

It will be demonstrated in this section how a filter for scalar system can be designed using Polynomial Toolbox for MATLAB[1]. Procedure in Matlab exactly follows the algorithm described in previous section. However, there are several problems to watch out for and they will be stressed herein.

Variable z^{-1} is used as a function variable instead of q^{-1} because q^{-1} is not fully supported in the toolbox[2]. ARMA models for (3) and r, ρ, m are assumed to be known[3]. Chosen models contain polynomials with complex coefficients.

```
>> r = 1/2; rho = 1/10; m = -1;
>> C = 1 +  (-1-0.5i)*z^-1  + (0.3+0.5i)*z^-2
>> D = 1 + (-1.3+0.2i)*z^-1 + (0.5+0.2i)*z^-2 + (-0.1 - 0.2i)*z^-3
>> M = 1 + (-0.5+0.1i)*z^-1 + (0.1-0.1i)*z^-2
>> N = 1 + (-0.4+0.2i)*z^-1 + (0.2+0.1i)*z^-2
```

Function `spf` computing spectral factorization returns its result as a function of z but we need it as a function of z^{-1}. Therefore $\beta(z^{-1})$ can be calculated as follows.

```
>> beta = spf( 1/r*C*C'*N*N'+rho/r*M*M'*D*D' )
beta =
 -0.0039-0.0013i + (0.032+0.18i)z + (-0.56-0.46i)z^2 +
          + (1.5+0.76i)z^3 + (-2.2-0.32i)z^4 + (1.5+0i)z^5
>> beta = beta * z^(-deg(beta))
beta =
(-0.0039-0.0013i)z^-5 + (0.032+0.18i)z^-4 + (-0.56-0.46i)z^-3 +
          + (1.5+0.76i)z^-2 + (-2.2-0.32i)z^-1 + 1.5+0i
>> isstable(beta)
ans = 1
```

The next step is to solve Diophantine equation (6). Unfortunately the Polynomial Toolbox cannot solve polynomial equations with double sided polynomials. Solution of equation (6) can be easily converted to an equation of single sided polynomials, both sides get multiplied by $z^{-max(n)}$ where $max(n)$ is the highest positive power of z. Polynomial equation $AX + BY = C$ has generally an infinite number of solutions given by a particular solution X_0, Y_0 and polynomials R, S such that for any polynomial T holds that $X = X_0 + RT$ and $Y = Y_0 + ST$ are solutions of the equation. The unique solution[4] fulfilling conditions of causality and optimality [5] is found further.

```
>> eqC = z^-m*C*C'*N';
>> eqA = r*beta';
```

[1] This example was created using a pre-release of Polynomial Toolbox version 3. Some amendments would have to be made in order to use the example in the commercially available version 2.5. Versions 2.5 and older do not support double sided polynomials. The solution using version 3 was chosen because it is more illustrative.

[2] It is not needed as z^{-1} can be used instead. However, it was formally better to use q^{-1} in the algorithm description above.

[3] These models were arbitrarily chosen for the sake of this simulation. They consist of stable polynomials with complex coefficients. These models may not have any real-world counterpart.

[4] A user defined function `uniqsol` was used, it is not a part of the toolbox.

```
>> eqB = z*D;
>> maxn = max([deg(eqA), deg(eqB), deg(eqC)]);

>> eqC = z^-maxn*eqC; eqB = z^-maxn*eqB; eqA = z^-maxn*eqA;

>> [Q0, L0, R, S] = axbyc(eqA, eqB, eqC)
Q0 =
(0+9.2e-009i)z^-1 + (-25+58i)z^-2 + (2e+003+2e+003i)z^-3 +
    (-3.1e+003-2.3e+003i)z^-4 + (5.9e+002+1.5e+003i)z^-5 + (2.3e+002-6.1e+002i)z^-6
L0 =
-2.2e+002+1.7e+002i + (1.1e+003+78i)z^-1 + (-2.3e+003-7.7e+002i)z^-2 +
    (2.5e+003+1.9e+003i)z^-3 + (-1.5e+003-1.6e+003i)z^-4
R =
(-0.19+0.37i)z^-4 + (0.17-0.52i)z^-5 + (-0.17+0.15i)z^-6 + (0.094+0i)z^-7
S =
-0.00012+0.00085i + (-0.031-0.023i)z^-1 + (0.034+0.15i)z^-2 +
    (-0.00085-0.35i)z^-3 + (-0.14+0.44i)z^-4 + (0.14-0.29i)z^-5

>> uniqsol
T =
(-2.4e+003+6.6e+003i)z + (-1.5e+002+9.9i)z^2 + (-1.5-3i)z^3 + (0.089-0.15i)z^4
>> Q1 = Q0 + R*T
Q1 =
(0.39+0.65i)z^-3 + (-1.4-0.59i)z^-2 + (1.3-0.096i)z^-1 + 0.041+0.062i
>> L = L0 + S*T
L =
0.041+0.0071i + (-0.048-0.014i)z + (0.016+0.0018i)z^2 +
    (-0.0035+0.0017i)z^3 + (0.00012+9.5e-005i)z^4
```

A final result for the estimator is obtained by substituting intermediate results into (7).

```
>> estimator = Q1 * N / beta;
estimator.numerator =
  (0.013+0.17i)z^-5 + (-0.5-0.44i)z^-4 + (1.3+0.73i)z^-3 + (-1.9-0.27i)z^-2 +
  + (1.3-0.11i)z^-1 + 0.041+0.062i
estimator.denominator =
  (-0.0039-0.0013i)z^-5 + (0.032+0.18i)z^-4 + (-0.56-0.46i)z^-3 +
  + (1.5+0.76i)z^-2 + (-2.2-0.32i)z^-1 + 1.5+0i
```

5 Conclusion

Polynomial methods have been employed in signal processing and in designing filters. These applications require complex polynomial coefficients. An algorithm for a design of a Wiener filter for a scalar signal was outlined here. It was demonstrated that the Polynomial Toolbox for Matlab is a good simulation package which can handle computations with complex polynomials. A numerical example of a Wiener filter design was performed in this article. It is obvious that once ARMA system models are known, filter design becomes very straightforward.

Acknowledgments

This work was supported by the Ministry of Education of the Czech Republic under Project LN00B096.

References

[1] Kučera V.: *Analysis and Design of Discrete Linear Control Systems*, Academia, Prague, 1991.

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

[3] Ahlen A., Sternad M.: *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.

[4] Ahlen A., Sternad M.: *Optimal Filtering Problems.* Chapter 5 in K Hunt, ed: Polynomial Methods in Optimal Control and Filtering, pp 120-161, Control Engineering Series, Peter Peregrinus, London, 1993.

[5] Ahlen A., Sternad M.: *Wiener Filter Design Using Polynomial Equations.* IEEE Transactions on Signal Processing, vol 39, pp 2387-2399, November 1991.

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

[7] Hromčík M., Ježek J., Šebek M.: *New Algorithm for Spectral Factorization and Its Practical Application.* Proceedings of the European Control Conference 2001, Porto, Portugal.

[8] Hromčík M., Ježek J., Šebek M.: *Complex Polynomial in Communications: Motivation, Algorithms, Software.* Konference Řízení procesů 2002, Dlouhé Stráně, červen 2002.

[9] Hromčík M., Ježek J., Šebek M., Hurák Z.: *Polynomial Spectral Factorization with Complex Coefficients.* Conference IFAC World Congress, Barcelona, July 2002.

[10] Hromčík M., Šebek M., Ježek J.: *Polynomial Toolbox for MATLAB and Complex Polynomials in Communications.* Sborník konference MATLAB 2001, pp. 94-102, Praha, říjen 2001.