Polynomial Methods in Time-Series Analysis

F. Aparicio-Pérez[1]

[1]National Institute of Statistics
Madrid, Spain

COMPSTAT 2010 Paris, August 22-27

Outline

Polynomial Matrices

- Polynomial matrices can be considered to be arrays formed by polynomials in a complex variable z
- Most operations that are valid with normal matrices are also valid with polynomial matrices.
- But: Some are not, for example, the inverse of a polynomial matrix exists if the matrix is not singular, but may be that it is not a polynomial matrix
- For example the polynomial matrix $a(z) = \begin{pmatrix} 1-z & z \\ z & 1-0.5z \end{pmatrix}$ has determinant $1-1.5z-0.5z^2$, and its inverse is the rational matrix $a^{-1}(z) = (1-1.5z-0.5z^2)^{-1} \cdot \begin{pmatrix} 1-0.5z & -z \\ -z & 1-z \end{pmatrix}$

Unimodular matrices

- A square polynomial matrix is called unimodular if its determinant is a non-zero scalar.
- The inverse of a unimodular matrix is a polynomial matrix.
- For example, the polynomial matrix $b(z) = \begin{pmatrix} 1 - z^2 & -2z \\ 2z & 4 \end{pmatrix}$ has determinant 4 and is thus unimodular, its inverse is the polynomial matrix $b^{-1}(z) = \begin{pmatrix} 1 & 0.5z \\ -0.5z & 0.25 - 0.25z^2 \end{pmatrix}$.
- The degree of a n x m polynomial matrix is defined as the maximum of the degrees of the *nm* polynomials that it has as elements.

Triangularization

- A basic result about n x m polynomial matrices is that they can be reduced by means of pre(post)-multiplication by a unimodular polynomial matrix to row(column) Hermite form (a triangular form).
- For example, $R(z) = U(z) \cdot a(z)$, where $U(z) = \begin{pmatrix} 1+z & z \\ -z & 1-z \end{pmatrix}$ is a unimodular matrix with determinant 1 and $R(z) = \begin{pmatrix} 1 & 2z + 0.5z^2 \\ 0 & 1 - 1.5z - 0.5z^2 \end{pmatrix}$ is an upper triangular matrix, so $det(a(z)) = (det(U_0))^{-1} \cdot det(R(z)) = (1)^{-1} \cdot (1 - 1.5z - 0.5z^2)$.

Right and Left Matrix Fraction Descriptions (1)

- A $s \times m$ rational transfer function $T(z)$ is a $s \times m$ array that has as elements polynomial quotients.
- A right coprime fraction (r.c.f) or right coprime matrix fraction description, of $T(z)$ is a pair of polynomial matrices, $(N_r(z), D_r(z))$, of orders $s \times m$ and $m \times m$ respectively such that:
  (i) $D_r(z)$ is non-singular (its determinant is not the zero polynomial).
  (ii) $T(z) = N_r(z)D_r(z)^{-1}$.
  (iii) $(N_r(z), D_r(z))$ is right-coprime, that is, all its greatest common right divisors are unimodular matrices.

Right and Left Matrix Fraction Descriptions (2)

- An important result states that given a $n \times m$ rational transfer function $T(z)$, it can always be expressed as a r.c.f. or l.c.f. $T(z) = D_l(z)^{-1} N_l(z) = N_r(z) D_r(z)^{-1}$.
- And it can be done in a numerically reliable and efficient way
- An example of a 2×1 transfer function expressed as a r.c.f and a l.c.f. is:

$$T(z) = \begin{pmatrix} z(z-1)(z+2) \\ z+1 \end{pmatrix} ((z+1)(z-1))^{-1} =$$

$$\begin{pmatrix} \frac{z(z+2)}{z+1)} \\ \frac{1}{z-1} \end{pmatrix} = \begin{pmatrix} z+1 & z-1 \\ 0 & (z-1)^2 \end{pmatrix}^{-1} \begin{pmatrix} (z+1)^2 \\ z-1 \end{pmatrix}$$

Matrix Polynomial Equations

- Several kinds of polynomial equations arise in system theory and signal processing.
- Some of them are described in Kučera (1979)
- The so-called symmetric matrix polynomial equation has the form

$$A'(z^{-1})X(z) + X'(z^{-1})A(z) = B(z) \tag{1}$$

where $A(z)$ and $B(z)$ are given polynomial matrices with real coefficients and $B(z)$ is para-Hermitian, that is $B(z) = B_l(z^{-1}) + B_r(z)$, with $B_l(z) = B_r'(z)$.

The Symmetric Matrix Polynomial Equation

- The solution of the symmetric matrix polynomial equation can be found in an efficient and numerically reliable way, as explained in Henrion and Šebek (1998).
- This equation can be used to compute the autocovariances of a VARMA process, see Söderström, Ježek and Kučera (1998).
- Given a stationary VARMA process of the form $a(B)y_t = b(B)\epsilon_t$, its autocovariance generating function is $G(z) = a^{-1}(z)b(z)\Sigma b'(z^{-1})a'^{-1}(z^{-1})$,
- We are looking for a decomposition of the form $G(z) = M(z) + M'(z^{-1})$.

Autocovariances of a VARMA process

- Pre-multiplying by $a(z)$, post-multiplying by $a'(z^{-1})$ and calling $X'(z) = a(z)M(z)$ we get, after transposition, $b(z^{-1})\Sigma b'(z) = a(z^{-1})X(z) + X'(z^{-1})a'(z)$,
- This is equation (1) with $B(z) = b(z^{-1})\Sigma b'(z)$ and $A(z) = a'(z)$.
- To find the autocovariances of the process we first solve this symmetric matrix polynomial equation for X, with the condition that X_0 be symmetric.
- Then, since $M(z) = (1/2)\Gamma_0 + z\Gamma_1 + z^2\Gamma_2 + \cdots$, ($\Gamma_i$ is the lag-i autocovariance of y_t), we solve recursively (long division) the equation $a(z)M(z) = X'(z)$ to get the first autocovariances

Spectral Factorization

- Finally, the Yule-Walker equations can be used to obtain the next autocovariances.
- This method is more efficient than the methods that are usually employed in time series analysis.
- Another application of the symmetric matrix polynomial equation is spectral factorization.

Polynomial Filters

- Given a stationary VARMA process of the form $a(B)y_t = b(B)\epsilon_t$, sometimes it is necessary to compute the model that follow some linear combination(s) of its components.
- More in general, the linear combination(s) may include delayed components.
- This problem is usually addressed in time series using ad-hoc hand computations for each case, but these computations grow quickly in complexity.
- Suppose that we want to compute the VARMA model that follows the process $z_t = F(B)y_t$, where $F(z)$ is an s x n polynomial matrix.

Right to Left Matrix Fractions Descriptions

- After solving in the VARMA model for y_t we pre-multiply by $F(B)$ and obtain $z_t = F(B)a^{-1}(B)b(B)\epsilon_t$,
- But $F(B)a^{-1}(B) = \tilde{a}^{-1}(B)\tilde{F}(B)$, that is, we transform a right matrix fraction description into a left one.
- Finally we do the spectral factorization $\tilde{F}(B)b(B)\epsilon_t = c(B)u_t$, where u_t is a new white noise with covariance matrix Σ_u
- The final model is $\tilde{a}(B)z_t = c(B)u_t$
- The method can be extended to the case of a rational filter of the form $G(B)z_t = F(B)y_t$

An Example (1)

- Let the joint model of x_t and y_t be

$$\begin{pmatrix} 1-B^4 & 0 \\ -(1-B)^2 & (1-B)^2 \end{pmatrix} \begin{pmatrix} x_t \\ y_t \end{pmatrix} =$$

$$\begin{pmatrix} 1+.44B+.5B^2+.32B^3 & -.25B-.25B^2-.34B^3 \\ .18B+.05B^2 & 1-.78B+.14B^2 \end{pmatrix} \begin{pmatrix} \epsilon_{1t} \\ \epsilon_{2t} \end{pmatrix}$$

- We want to compute the marginal model of y_t
- That is, we compute the model of the filter $y_t = F(B) \begin{pmatrix} x_t \\ y_t \end{pmatrix}$ with $F(z) = (0\ 1)$

An Example (2)

- We obtain

$$\tilde{a}(z) = 1 - 0.5z^2, \quad \tilde{F}(z) = (0.5z, \ \ 1),$$

$$c(z) = 1 + 0.032502z, \quad \sigma_u^2 = 1.5384$$

- So, the marginal model of y_t is

$$(1 - 0.5B^2)y_t = (1 + 0.032502B)u_t, \quad \sigma_u^2 = 1.5384$$

- The result is automatically obtained by the computer

Introduction

- An exact method for the computation of a univariate Wiener-Kolmogorov filter based on a finite sample can be found in Burman(1980).
- The exact multivariate case based on a finite sample has been addressed in the literature before using state-space methods and, for some particular cases, like the signal plus noise model or the deconvolution problem, using polynomial methods (e.g. Ahlén and Sternad (1991)).
- We will provide a brief description of a new polynomial method that solves the general multivariate case,

Filter Equations (1)

- Assume that two multivariate processes, s_t and y_t, follow jointly a stationary, invertible and left coprime VARMA model with VAR part $a(B)$ and MA part $b(B)$ that we consider partitioned as in

$$\begin{pmatrix} a_{11}(B) & a_{12}(B) \\ a_{21}(B) & a_{22}(B) \end{pmatrix} \begin{pmatrix} s_t \\ y_t \end{pmatrix} = \begin{pmatrix} b_{11}(B) & b_{12}(B) \\ b_{21}(B) & b_{22}(B) \end{pmatrix} \begin{pmatrix} \epsilon_{1t} \\ \epsilon_{2t} \end{pmatrix} \quad (2)$$

- Assume also that a finite sample of y_t is available, but no observations from s_t are available. We are interested in estimating the values of s_t.

Filter Equations (2)

- First, we transform the model into another one that has a diagonal AR part, this is accomplished by pre-multiplying (2) by $Adj(a(B))$, the adjoint of $a(B)$, the result is

$$det(a(B))I_n y_t = \begin{pmatrix} d_{11}(B) & d_{12}(B) \\ d_{21}(B) & d_{22}(B) \end{pmatrix} \begin{pmatrix} \hat{\epsilon}_{1t} \\ \hat{\epsilon}_{2t} \end{pmatrix} \qquad (3)$$

- where $d(z) = Adj(a(z))b(z)L'$, with $L'L = \Sigma_\epsilon$ (Cholesky decomposition), $\hat{\epsilon}_t = (L^{-1})'\epsilon_t$ is a standardized white noise process and I_n is the identity matrix of dimension n.
- Now we will use the Wiener-Kolmogorov formula that assumes that we have a doubly infinite realization of y_t, see Caines(1988) p. 139.

Filter Equations (3)

- The key points are (i) the diagonal AR part cancels out and (2) since we actually have a finite sample, we use the exact finite sample forecasts and backcasts of y_t as needed (we only need a few of them).
- Because of the properties of conditional expectations this procedure will provide the exact Wiener-Kolmogorov filter based on the finite sample.
- The joint covariance generating function of s_t and y_t is $G(z) = (det(a(z))^{-1} d(z) d'(z^{-1})(det(a(z^{-1}))^{-1}$
- and the optimal filter is $\hat{s}_t = G_{12}(B) \cdot G_{22}^{-1}(B)\hat{y}_t =$ $[d_{11}(B) d_{12}(B)] \begin{bmatrix} d'_{21}(F) \\ d'_{22}(F) \end{bmatrix} \Theta'^{-1}(F)\Theta^{-1}(B)\hat{y}_t,$

Filter Equations (4)

- So, we can compute the exact finite Wiener Kolmogorov filter running three cascaded filters.
- First filter: $\hat{x}_t = \Theta^{-1}(B)\hat{y}_t$, with time running forwards
- Second filter: $\hat{v}_t = \tilde{\Theta}'^{-1}(F)\tilde{e}(F)\hat{x}_t$, with time running backwards, where $e'(z) = [d_{21}(z)d_{22}(z)]$ and $e(z^{-1})\Theta'^{-1}(z^{-1}) = \tilde{\Theta}'^{-1}(z^{-1})\tilde{e}(z^{-1})$ (we transform a right fraction into a l.c.f)
- Third filter: $\hat{s}_t = [d_{11}(B)d_{12}(B)]\hat{v}_t$ with time running forwards.
- Making some more polynomial computations the number of filters can be further reduced to two, one running backwards and the other forwards in time

Initial Conditions (1)

- No matter what filters we use, we must compute the initial and final conditions of the processes involved.
- For example, using the above filter we need the initial and final conditions of the y_t, s_t, x_t, and v_t processes.
- The final conditions of y_t are simply the exact forecasts of y_t, and can be obtained from its marginal model.
- The initial conditions of y_t are the exact forecasts of the marginal time-reversed process of y_t, that can be obtained as an echelon realization of a process that has as autocovariances the transposed autocovariances of y_t
- But, to obtain the joint MSE's of forecasts and backcasts it is better to use an extended innovations algorithm.

Initial Conditions (2)

- The initial and final conditions of the other processes can be computed in three different forms.
- The first form is to use the extended innovations algorithm.
- For s_t this can be done since the joint model of y_t with s_t is known, and from this model the cross-covariances can be computed.
- But for v_t (or other processes) the joint model of y_t and v_t can also be computed from the filter equations. It will be a singular joint model, but its cross-covariances can still be computed.

Initial Conditions (3)

- The second form consists of summing something that we could call a left-right matrix geometric series. This form can even be extended to non-stationary processes.
- The third form may be seen as a generalization of the univariate procedure in Burman (1980). It solves a system of linear equations formed by the filter equations and the backwards in time model of the process that the filter transforms.

Running the Filters

- Once we know how to compute the initial and final conditions, running the filters is easy.
- Moreover, the time of computation grows only linearly with T.
- We have seen that fixed interval smoothing can be efficiently done.
- Fixed point smoothing can also be efficiently done, using the extended innovations algorithm.
- Fixed lag smoothing can also be done, but not so efficiently, because of the non-recursive nature of Wiener-Kolmogorov filters.

Some applications and an Example (1)

- Exact filtering is of central importance in engineering, statistics, physics,...
- In time series analysis, the methodology that we have proposed can be used to compute the classical univariate filters, e.g. the Hodrick-Prescott filter
- But new univariate or multivariate filters can also be computed
- For example, consider two quarterly economic indicators y_{1t} and y_{2t}, that follow the structural models $y_{jt} = T_{jt} + S_t + e_{jT}$.
- This is a structural decomposition as trend, seasonal and irregular components with the peculiarity that both seasonal components are assumed to be equal.
- There are many possible specifications of the components,

Some Applications and an Example (2)

$$\begin{pmatrix} 1+B+B^2+B^3 & 0 & 0 & 0 & 0 \\ 0 & 1-B & 0 & 0 & 0 \\ 0 & 0 & (1-B)^2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} S_t \\ T_{1t} \\ T_{2t} \\ e_{1t} \\ e_{2t} \end{pmatrix} = \tag{4}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1+B & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a_{1t} \\ a_{2t} \\ a_{3t} \\ a_{4t} \\ a_{5t} \end{pmatrix} \tag{5}$$

Some applications and an Example (3)

- We are interested in extracting the two trends and the common seasonal component from the two observed indicators.
- To do so, first we have to compute the joint model of the five processes S_t, T_{1t}, T_{2t}, y_{1t} and y_{2t}.
- Then we have to compute the filter equations.
- Next, the initial conditions have to be calculated.
- Finally, the forward and backward filters have to be run.

Some applications and an Example (4)

- The joint model is the model of the following filter:

$$\begin{pmatrix} S_t \\ T_{1t} \\ T_{2t} \\ y_{1t} \\ y_{2t} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} S_t \\ T_{1t} \\ T_{2t} \\ e_{1t} \\ e_{2t} \end{pmatrix} \qquad (6)$$

- And the marginal model of y_{1t} and y_{2t} is obtained doing:

$$\begin{pmatrix} y_{1t} \\ y_{2t} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} S_t \\ T_{1t} \\ T_{2t} \\ e_{1t} \\ e_{2t} \end{pmatrix} \qquad (7)$$

Some applications and an Example (5)

$$\begin{pmatrix} 1+B+B^2+B^3 & 0 & 0 & 0 & 0 \\ 0 & 1-B & 0 & 0 & 0 \\ 0 & 0 & (1-B)^2 & 0 & 0 \\ -1 & -1 & 0 & 1 & 0 \\ -1 & 0 & -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} S_t \\ T_{1t} \\ T_{2t} \\ y_{1t} \\ y_{2t} \end{pmatrix} = \tag{8}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1+B & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_{1t} \\ u_{2t} \\ u_{3t} \\ u_{4t} \\ u_{5t} \end{pmatrix} \tag{9}$$

Some applications and an Example (6)

And the marginal model is (with two decimal places):

$$\begin{pmatrix} 1-B^4 & 0 \\ -(1-B)^2 & (1-B)^2 \end{pmatrix} \begin{pmatrix} y_{1t} \\ y_{2t} \end{pmatrix} =$$

$$\begin{pmatrix} 1+.44B+.5B^2+.32B^3 & -.25B-.25B^2-.34B^3 \\ .18B+.05B^2 & 1-.78B+.14B^2 \end{pmatrix} \begin{pmatrix} w_{1t} \\ w_{2t} \end{pmatrix}$$

And the three cascaded filters are (with one decimal place):

Some applications and an Example (7)

First filter, with time running forwards

$$\begin{pmatrix} 3 - 4.3B + 1.5B^2 - .6B^3 - .9B^4 + 1.4B^5 & \cdots \\ 1.8 - 2.9B + 1.5B^2 - .6B^3 + .2B^4 - .1B^5 & \cdots \end{pmatrix} \begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} =$$

$$\begin{pmatrix} y_{1t} \\ y_{2t} \end{pmatrix}$$

Some applications and an Example (8)

Second filter, with time running backwards

$$\begin{pmatrix} 1 - .1F + .2F^2 + .2F^3 & \cdots & \cdots & \cdots & \cdots \\ -12.8F + 5.1F^2 - 3.7F^3 & \cdots & \cdots & \cdots & \cdots \\ 11.8F - 5.9F^2 + 2.9F^3 & \cdots & \cdots & \cdots & \cdots \\ -5.9F^2 & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & \cdots & \cdots \end{pmatrix} \begin{pmatrix} v_{1s} \\ v_{2s} \\ v_{3s} \\ v_{4s} \\ v_{5s} \end{pmatrix} =$$

$$\begin{pmatrix} .3 - .6F + .1F^2 + .1F^3 & .1 - .3F + .1F^2 + .1F^3 \\ .3 - 5.2F + 9.3F^2 - 4.5F^3 & -.2 - 2.2F + 5.1F^2 - 2.7F^3 \\ 4.6F - 9.2F^2 + 4.6F^3 & .4 + 1.9F - 5F^2 + 2.8F^3 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_{1s} \\ x_{2s} \end{pmatrix}$$

Some applications and an Example (9)

Third filter, with time running forwards

$$\begin{pmatrix} S_t \\ T_{1t} \\ T_{2t} \end{pmatrix} =$$

$$\begin{pmatrix} 1 - 3B + 3B^2 - B^3 & 0 & 0 & 0 & 0 \\ 0 & 1 - B^2 - B^4 + B^6 & 0 & 0 & 0 \\ 0 & 0 & 1 - B^4 & 0 & 0 \end{pmatrix} \begin{pmatrix} v_{1t} \\ v_{2t} \\ v_{3t} \\ v_{4t} \\ v_{5t} \end{pmatrix}$$

Summary

- The results suggest that the polynomial methods can solve efficiently many problems in time series analysis that could only be solved before using another kind of techniques.
- The advantage is that in many cases they are more direct, faster and provide more intuition to the researcher.