UvA Econometrics

Discussion Paper: 2004/15

A program for computing the exact Fisher information matrix of a Gaussian VARMA model

André Klein, Guy Mélard,
Jerzy Niemczyk and Toufik Zahaf

www.fee.uva.nl/ke/UvA-Econometrics

Department of Quantitative Economics
Faculty of Economics and Econometrics
Universiteit van Amsterdam
Roetersstraat 11
1018 WB AMSTERDAM
The Netherlands

UvA Universiteit van Amsterdam

A Program for Computing the Exact Fisher Information Matrix of a Gaussian VARMA Model

André Klein*, Guy Mélard†, Jerzy Niemczyk‡and Toufik Zahaf§

Abstract

— A program in the MATLAB environment is described for computing the Fisher information matrix of the exact information matrix of a Gaussian vector autoregressive moving average (VARMA) model. A computationally efficient procedure is used on the basis of a state space representation. It relies heavily on matrix operations. An illustration of the procedure is given for simple VARMA models and an example of output from a more realistic application is discussed.

1 Introduction

Time series analysis provides a useful technology that can be used in many fields of application such as econometrics, finance, meteorology, signal processing and biology, etc. Time

*A. Klein is at the Department of Quantitative economics, University of Amsterdam, Roetersstraat 11, 1018 WB Amsterdam, The Netherlands. Phone: +31-20-5254245, fax: +31-20-5254349, e-mail: A.A.B.Klein@uva.nl.

†G. Mélard is with ISRO and ECARES, Université Libre de Bruxelles, Campus Plaine CP 210, Bd du Triomphe, B-1050 Bruxelles, Belgium. Phone: +32-2-6505890, fax: +32-2-6505899, e-mail: gmelard@ulb.ac.be.

Author has benefited from an IAP-network in Statistics, contract P5/24, financially supported by the Belgian Federal Office for Scientific, Technical and Cultural Affairs (OSTC). He is grateful to Abdessamad Saidi for his help on the Nsiri-Roy example.

‡J. Niemczyk was PhD student at ISRO, Université Libre de Bruxelles, now with University of Amsterdam, Roetersstraat 11, 1018 WB Amsterdam, The Netherlands. Phone: +31-20-5257363, fax: +31-20-5254349, e-mail: J.Niemczyk@uva.nl.

§T. Zahaf is with GlaxoSmithKline Biologicals s.a. and ISRO U.L.B. Phone: +32-2-6569256, fax: +32-2-6569132, e-mail: zahaf.toufik@gskbio.com.

series analysis enables the past of a phenomenon to be studied in order to forecast its future behavior. Among the models which can be used to represent a series, the general dynamic model includes the large class of VARMA which is considered here. VARMA models can be written in state space form which enables the use of the Kalman filter to compute the pseudo-likelihood function, as if the process were Gaussian. That approach provides good estimators of the parameters even when the process is not Gaussian.

For a time series with fixed length N, estimation by the pseudo-maximum likelihood method of the parameters of an autoregressive-moving average (ARMA) model was a pole of interest from the beginning of the 70's until the middle of the 80's. The major drawback of the method is the computation time which has been reduced over the years. Among the many algorithms which have appeared, the fastest was obtained through the use of a fast version of the Kalman filter which is also called the Chandrasekhar equations (e.g. Morf *et al.*, 1974, Pearlman, 1980). Estimation by pseudo-maximum likelihood can now be found in most statistical software packages.

Engineers have faced the problem of fitting ARMA and other time series models but with on-line data. Consequently, they have developed estimation techniques where the estimates are updated each time a new observation becomes available. Algorithms such as the *Recursive Maximum Likelihood* (RML), a special case of the *Recursion Prediction Error Method* (RPEM), provides an on-line estimator which has the same asymptotic properties as the off-line pseudo-maximum likelihood method (e.g. Ljung and Söderström, 1983). These methods are used in signal processing and automatic control, much less in statistics and econometrics, despite promising applications with stock exchange or meteorological data.

A good evaluation of an estimator is useful insofar as its statistical significance can be assessed. This requires estimating the (generally asymptotic) covariance matrix of the vector of estimators. Statistical tests on the parameters, determination of confidence intervals, the scoring method (and also a new variant of the RML method developed by one of the authors, see Zahaf, 1998) and the determination of the size needed for a sample, all require computing the Fisher information matrix. Its inverse is the asymptotic covariance matrix of the estimator and is also related to the Cramér-Rao lower bound. Most of the literature on this subject for time series models has considered the *asymptotic* Fisher information matrix, based on an approximation of the Gaussian likelihood function. Porat and Friedlander (1986) have published an algorithm for computing the *exact* information matrix but their method is time-consuming. Zadrozny (1989, 1992) and Terceiro (1990) have separately developed methods based on the Kalman filter, which is faster than the Porat and Friedlander (1986) method. Mélard and Klein (1994) and Klein *et al.* (1998), for scalar models, and Klein *et al.* (2000), for vector models, have used the Chandrasekhar equations. The method of Mélard and Klein (1994) is also more efficient in computation time than alternative procedures. Klein *et al.* (2000) have given, for a general multivariate model expressed in state space form, a system of matrix recurrence equations which enables computing the information matrix as a whole instead of element by element, as the alternative methods. Being based on the Chandrasekhar equations, the method is in principle more efficient than those of Zadrozny (1989, 1992) and Terceiro (1990), which make use of the Kalman filter. Also, the matrix representation allows easier checking of the strictly definite positive character of the information matrix.

We have implemented the algorithm of Klein *et al.* (2000), adding some modifications described in Section 3, for VARMA models in the MATLAB environment. The purpose of this paper is to describe the program and illustrate its usefulness on an example. The program set forth also allows computation of the likelihood function and its first-order derivatives.

In Section 2, we recall briefly the theory detailed by Klein *et al.* (2000). In Section 3, we describe the modification made in the algorithm in order to obtain stable results. In Section 4, we handle two examples.

2 Preliminaries

A Gaussian VARMA model of order (p, q) is defined by the equation

$$z_t = \alpha_1 z_{t-1} + \alpha_2 z_{t-2} + \ldots + \alpha_p z_{t-p} + w_t - \beta_1 w_{t-1} - \beta_2 w_{t-2} - \ldots - \beta_q w_{t-q}, \tag{1}$$

where $z_t \in \mathbb{R}^m$ is the vector of observations, w_t is a sequence of independently and normally distributed random vectors with mean 0 and an invertible covariance matrix Q. It is assumed that the process is stationary, invertible and uniquely specified which implies conditions on the autoregressive and moving average matrix polynomials which will not be detailed here. The model is put under state space form (for example, Shea, 1989)

$$x_{t+1} = \Phi x_t + F w_t, \tag{2}$$
$$z_t = H x_t + w_t, \tag{3}$$

where $x_t \in \mathbb{R}^n$ is the vector of the state variables and

$$\Phi = \begin{pmatrix} \alpha_1 & I_m & 0_m & \cdots & 0_m \\ \alpha_2 & 0_m & I_m & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0_m \\ \vdots & \vdots & \ddots & 0_m & I_m \\ \alpha_h & 0_m & \cdots & \cdots & 0_m \end{pmatrix}, \qquad F = \begin{pmatrix} \alpha_1 - \beta_1 \\ \alpha_2 - \beta_2 \\ \vdots \\ \alpha_h - \beta_h \end{pmatrix} \qquad \text{and} \qquad H^\top = \begin{pmatrix} I_m \\ 0_m \\ \vdots \\ 0_m \end{pmatrix}, \tag{4}$$

$h = \max(p, q)$, $\alpha_i = 0_m$ for $i > p$, $\beta_j = 0_m$ for $j > q$. We use the same notations as in [3] except q which was denoted by s.

Let $\widehat{z}_{t|t-1}$, be the prediction of the observation vector and $\hat{x}_{t|t-1}$, the linear prediction of the state vector, given the past of the process. Let $\widetilde{z}_t = z_t - \widehat{z}_{t|t-1}$ be the (one-step-ahead) prediction error of the observations and $\tilde{x}_t = x_t - \hat{x}_{t|t-1}$, the prediction error of the state vector. Let $B_t = \mathbb{E}\{\widetilde{z}_t \widetilde{z}_t^\top\}$ be the covariance matrix of the prediction error of the observations, and $P_{t|t-1} = \mathbb{E}\{\widetilde{x}_t \widetilde{x}_t^\top\}$, the covariance matrix of the prediction error of the state vector, a $n \times n$ matrix, where $n = hm$.

There are several ways to express the exact likelihood function of a time series $\{z_1, \ldots, z_N\}$ of length N. Except for the closed form expression of a normal multivariate density, the simplest representation is based on the Chandrasekhar recurrence equations, which are also the most computationally efficient, even with respect to the Kalman filter.

It is possible here because the process defined by (1) is stationary and the state space representation (2-3) is time-invariant. It consists in the following recurrence equations (Shea, 1989) to derive $\widetilde{z}_t$, and B_t:

$$\begin{aligned} B_t &= B_{t-1} + HY_{t-1}X_{t-1}Y_{t-1}^\top H^\top && (5)\\ K_t &= \left[K_{t-1}\,B_{t-1} + \Phi Y_{t-1}X_{t-1}Y_{t-1}^\top H^\top\right] B_t^{-1} && (6)\\ Y_t &= [\Phi - K_{t-1}H]Y_{t-1} && (7)\\ X_t &= X_{t-1} - X_{t-1}Y_{t-1}^\top H^\top B_t^{-1} HY_{t-1}X_{t-1} && (8)\\ \hat{z}_{t|t-1} &= H\hat{x}_{t|t-1} && (9)\\ \hat{x}_{t+1|t} &= \Phi\hat{x}_{t|t-1} + K_t\widetilde{z}_t, && (10) \end{aligned}$$

using auxiliary matrices K_t, X_t and Y_t, of dimensions $n \times m$, $m \times m$ and $n \times m$, respectively. It can be seen from (2 - 3) and (9 - 10) that prediction error is the following

$$\begin{aligned} \tilde{x}_{t+1} &= x_{t+1} - \hat{x}_{t+1|t} \\ &= \overline{\Phi}_t\tilde{x}_t + \bar{F}_t w_t, && (11) \end{aligned}$$

and

$$\begin{aligned} \tilde{z}_t &= z_{t+1} - \hat{z}_{t+1|t} \\ &= H\tilde{x}_t + w_t, && (12) \end{aligned}$$

where $\bar{F}_t = (F - K_t)$ and $\overline{\Phi}_t = (\Phi - K_t H)$. We suppose that the model (2) depends on ℓ parameters denoted by the vector $\theta = (\theta_1, \ldots, \theta_\ell)^\top$. Thus Φ, F, H and Q are functions of θ and are supposed to be two times continuously differentiable.

Given a time series of length N, the Chandrasekhar equations (5-10) are used to compute the negative logarithm of the likelihood of the system described by (2) and (3)

$$l(\theta) = -\log L(\theta) = \sum_{t=1}^{N}\left\{\frac{m}{2}\log(2\pi) + \frac{1}{2}\log|B_t| + \frac{1}{2}\tilde{z}_t^\top B_t^{-1}\tilde{z}_t\right\}. \qquad (13)$$

The Chandrasekhar recurrence equations are an alternative to the better known Kalman filter recurrences. They are restricted to stationary processes and time-invariant models, which are the case here (Morf *et al.*, 1974). The exact information matrix is given as the following $\ell \times \ell$ matrix

$$J_N(\theta) = \sum_{t=1}^{N}\left[\frac{1}{2}\left(\frac{\partial \mathrm{vec}\ B_t}{\partial\theta^\top}\right)^\top \left(B_t^{-1}\otimes B_t^{-1}\right)\left(\frac{\partial \mathrm{vec}\ B_t}{\partial\theta^\top}\right) + \mathbb{E}\left\{\left(\frac{\partial\tilde{z}_t}{\partial\theta^\top}\right)^\top B_t^{-1}\left(\frac{\partial\tilde{z}_t}{\partial\theta^\top}\right)\right\}\right]. \qquad (14)$$

That formula was proved by [6].

The differentiation rule applied in [3] is used to derive recursion equations for $B_t^\theta = \partial \mathrm{vec} B_t/\partial\theta^\top$ and $\tilde{z}_t^\theta = \partial\tilde{z}_t/\partial\theta^\top$ at the matrix level from which $J_N(\theta)$ can be deduced.

For solving the first term of (14) the derivatives of the Chandrasekhar equations are used, whereas the situation is different for the second one, which involves the expected value

of stochastic elements. In order to obtain an appropriate covariance structure, vectorization of $J_N(\theta)$ is recommended. Consequently we obtain

$$\begin{aligned} \mathrm{vec}J_N(\theta) = &\sum_{t=1}^{N} \frac{1}{2}\left(\frac{\partial \mathrm{vec}B_t}{\partial \theta^\top} \otimes \frac{\partial \mathrm{vec}B_t}{\partial \theta^\top}\right)^\top \mathrm{vec}(B_t^{-1} \otimes B_t^{-1}) \\ &+ \sum_{t=1}^{N} \mathbb{E}\left(\frac{\partial \tilde{z}_t}{\partial \theta^\top} \otimes \frac{\partial \tilde{z}_t}{\partial \theta^\top}\right)^\top \mathrm{vec}B_t^{-1} \end{aligned} \tag{15}$$

For developing a recurrence for the second term of (15) recurrence relations have to be derived using the differentiation rules detailed in the theory described by [3]. We have

$$\begin{aligned} \mathbb{E}\left\{\tilde{z}_t^\theta \otimes \tilde{z}_t^\theta\right\}^\top &= \left[\mathbb{E}\left\{\tilde{x}_t^\theta \otimes \tilde{x}_t^\theta\right\}^\top\right](H \otimes H)^\top + \mathbb{E}\left\{w_t^\theta \otimes w_t^\theta\right\}^\top \\ &+ \left[\mathbb{E}\left\{\tilde{x}_t^\theta \otimes w_t^\theta\right\}^\top\right](H^\top \otimes I_m) + \left[\mathbb{E}\left\{w_t^\theta \otimes \tilde{x}_t^\theta\right\}^\top\right](I_m \otimes H^\top) \end{aligned} \tag{16}$$

where $(\partial \mathrm{vec}\ A) / \left(\partial \theta^\top\right) = A^\theta$ and $(\partial b) / \left(\partial \theta^\top\right) = b^\theta$, for any matrix A and any vector b. Each term is computed by recursions with respect to time, using initial values which are also described in [3]. Third and fourth factor of (16) can be rewritten as $\mathbb{E}\left\{\tilde{x}_t^\theta H \otimes w_t^\theta\right\}^\top$, $\mathbb{E}\left\{w_t^\theta \otimes \tilde{x}_t^\theta H\right\}^\top$ and they are related by $\mathbb{E}\left\{\tilde{x}_t^\theta H \otimes w_t^\theta\right\}^\top = M_{l,l}\mathbb{E}\left\{w_t^\theta \otimes \tilde{x}_t^\theta H\right\}^\top M_{m,m}$ via the commutation matrices property, which is, for any matrices $A \in \mathbb{R}^{k\times l}$ and $B \in \mathbb{R}^{m\times n}$ we have

$$M_{m,k}(A \otimes B)M_{l,n} = (B \otimes A). \tag{17}$$

Thus wee need to compute only one of those factors.

3 On stability of the program

In order to obtain (16), at each iteration, we need to update the following expressions $\mathbb{E}\left\{\tilde{x}_{t+1}^\theta \otimes \tilde{x}_{t+1}^\theta\right\}^\top$, $\mathbb{E}\left\{x_{t+1}^\theta \otimes x_{t+1}^\theta\right\}^\top$, $\mathbb{E}\left\{x_{t+1}^\theta \otimes \tilde{x}_{t+1}^\theta\right\}^\top$, $\mathbb{E}\left\{x_{t+1}^\theta \otimes w_{t+1}^\theta\right\}^\top$, $\mathbb{E}\left\{w_{t+1}^\theta \otimes \tilde{x}_{t+1}^\theta\right\}^\top$, $\mathbb{E}\left\{w_{t+1}^\theta \otimes w_{t+1}^\theta\right\}^\top$, $\mathbb{E}\left\{\left(\tilde{x}_{t+1}^\theta\right)^T \otimes \tilde{x}_{t+1}\right\}$, $\mathbb{E}\left\{\left(\tilde{x}_{t+1}^\theta\right)^T \otimes x_{t+1}\right\}$, $\mathbb{E}\left\{\left(x_{t+1}^\theta\right)^T \otimes \tilde{x}_{t+1}\right\}$, $\mathbb{E}\left\{\left(x_{t+1}^\theta\right)^T \otimes x_{t+1}\right\}$, $\mathbb{E}\left\{\left(w_{t+1}^\theta\right)^T \otimes \tilde{x}_{t+1}\right\}$, $\mathbb{E}\left\{\left(w_{t+1}^\theta\right)^T \otimes x_{t+1}\right\}$, $\mathbb{E}\left\{\tilde{x}_{t+1} \otimes \tilde{x}_{t+1}\right\}$, $\mathbb{E}\left\{\tilde{x}_{t+1} \otimes x_{t+1}\right\}$, $\mathbb{E}\left\{x_{t+1} \otimes x_{t+1}\right\}$.
All those equations are presented in [3]. In **??**, we formulated a procedure for deriving the initial values. To illustrate the method, let us consider the example of a two-dimensional $VARMA(1,1)$ process defined with

$$\alpha_1 = \begin{pmatrix} 0.8 & -0.2 \\ 1.2 & 0.2 \end{pmatrix}, \beta_1 = \begin{pmatrix} 0 & -1 \\ 0.5 & -0.5 \end{pmatrix} \text{ and } Q = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

This is Example 2 of Klein *et al.* (2004). We might conjecture that there exists a limit Fisher information matrix for our model and that this is the asymptotic Fisher information matrix defined by

$$J_\infty(\theta) = \mathbb{E}\left\{\left(\frac{\partial w_t}{\partial \theta^\top}\right)^\top Q^{-1}\left(\frac{\partial w_t}{\partial \theta^\top}\right)\right\}.$$

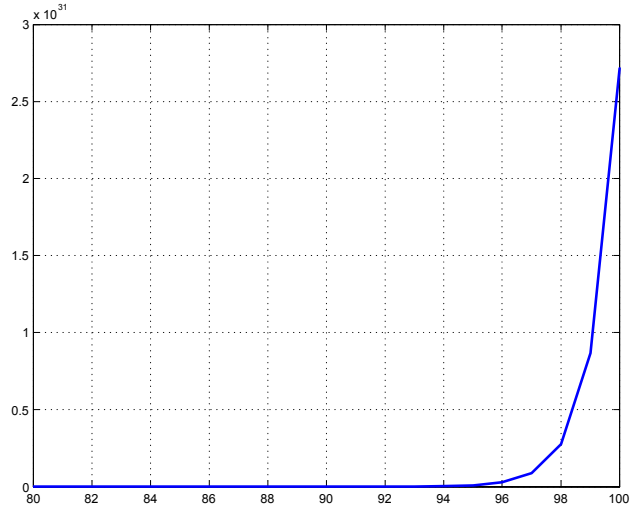

Figure 1: Graph above plots $\frac{1}{t-1}||J_{t-1}(\theta)|| - \frac{1}{t}||J_t(\theta)||$ for $t = 80, \ldots, 100$, original formulae.

Figure 1 shows the differences $\frac{1}{t-1}||J_{t-1}(\theta)|| - \frac{1}{t}||J_t(\theta)||$ for $t = 80, \ldots, 100$. Note that the scale of the vertical axis goes from 0 to 3.10^{31}. This is far from our expectations.

A closer look at the steps in the computation revealed that the matrices that were intended to be symmetric were no longer symmetric. As a matter of fact, even after a few dozen iterations, the program provides EFIM which is not symmetric and is not semipositive definite. We have noticed that, due to numerical imprecision, relationship (17) does not hold exactly in our case. The differences between two expressions are very small (some elements of the matrices differ after the fourteenth to sixteenth decimal place) but they accumulate fast enough to distort the final result. To solve this numerical problem we make use of the commutation matrices property described at the end of previous section. For each expression above we compute its 'reflection' derived from (17) and we take their average. For example for $\mathbb{E}\{w_{t+1}^{\theta} \otimes \tilde{x}_{t+1}^{\theta}\}^{\top}$ we compute its 'reflection' $\mathbb{E}\{\tilde{x}_{t+1}^{\theta} \otimes w_{t+1}^{\theta}\}^{\top}$ and then we take the average $\mathbb{E}\{w_{t+1}^{\theta} \otimes \tilde{x}_{t+1}^{\theta}\}^{\top} = \frac{1}{2}\big[\mathbb{E}\{w_{t+1}^{\theta} \otimes \tilde{x}_{t+1}^{\theta}\}^{\top} + M_{l,l}\mathbb{E}\{\tilde{x}_{t+1}^{\theta} \otimes w_{t+1}^{\theta}\}^{\top} M_{n,m}\big]$. We do the same to all expressions, including (16), even though some of them do not require computing the 'reflection' since they are of the form $(A \otimes A)$

Let us check that our refinement produces a satisfactory solution. Figure 2 shows the same difference as Figure 1 but this time for $t = 2 \cdot 10^5, \ldots, 10^6$. The scale of the vertical axis goes from 0 to 4.10^{-6}, which is nice. It looks that without applying these refinements the program will not provide stable results.

For the number of observations, $N = 10^6$, we get the following EFIM

$$\frac{1}{N}J_N(\theta) = \begin{pmatrix} 3.110805 & -1.082425 & 1.307974 & -0.095109 & -1.279889 & 1.168475 & 0.470106 & 0.668476 \\ -1.082425 & 3.783811 & -1.127718 & 0.340585 & -0.364131 & -1.902170 & -0.864129 & 1.097822 \\ 1.307974 & -1.127718 & 5.037135 & -1.861411 & 0.573368 & -0.027173 & -1.176628 & 0.472826 \\ -0.095109 & 0.340585 & -1.861411 & 5.257241 & -0.288043 & 1.032604 & 0.211956 & -1.967387 \\ -1.279889 & -0.364131 & 0.573368 & -0.288043 & 1.749995 & -0.499996 & -0.000000 & -0.000001 \\ 1.168475 & -1.902170 & -0.027173 & 1.032604 & -0.499996 & 2.999987 & 0.000003 & -0.000000 \\ 0.470106 & -0.864129 & -1.176628 & 0.211956 & -0.000000 & 0.000003 & 1.749995 & -0.499999 \\ 0.668476 & 1.097822 & 0.472826 & -1.967387 & -0.000001 & -0.000000 & -0.499999 & 2.999991 \end{pmatrix}$$

with eigenvalues equal to

$(8.20921811 \quad 6.85508984 \quad 0.10662288 \quad 0.29046018 \quad 1.37934008 \quad 2.27652174 \quad 4.05188028 \quad 3.51981513)$.

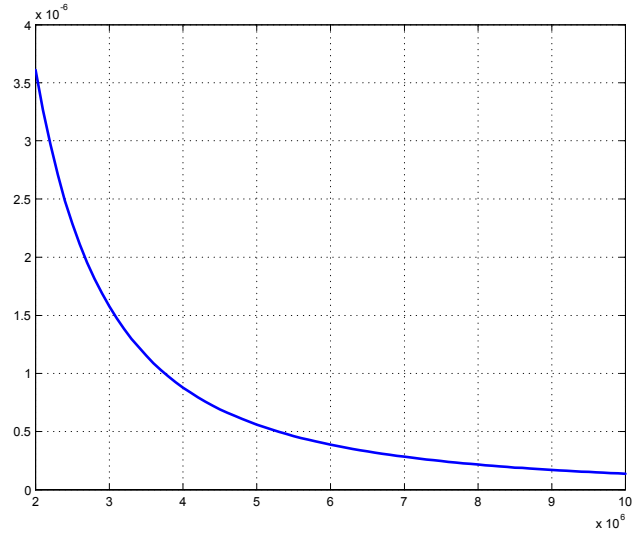

Figure 2: Graph above plots the differences $\frac{1}{t-1}||J_{t-1}(\theta)|| - \frac{1}{t}||J_t(\theta)||$ for $t = 2 \cdot 10^5, \ldots, 10^6$, applying the refinements described above.

It can be shown (Klein *et al.*, 2004) that the asymptotic FIM is equal to

$$J_\infty(\theta) = \begin{pmatrix} 3.110809 & -1.082428 & 1.307971 & -0.095109 & -1.279891 & 1.168478 & 0.470109 & 0.668478 \\ -1.082428 & 3.783816 & -1.127717 & 0.340580 & -0.364130 & -1.902174 & -0.864130 & 1.097826 \\ 1.307971 & -1.127717 & 5.037138 & -1.861413 & 0.573370 & -0.027174 & -1.176630 & 0.472826 \\ -0.095109 & 0.340580 & -1.861413 & 5.257246 & -0.288043 & 1.032609 & 0.211957 & -1.967391 \\ -1.279891 & -0.364130 & 0.573370 & -0.288043 & 1.750000 & -0.500000 & 0 & 0 \\ 1.168478 & -1.902174 & -0.027174 & 1.032609 & -0.500000 & 3 & 0 & 0 \\ 0.470109 & -0.864130 & -1.176630 & 0.211957 & 0. & 0 & 1.750000 & -0.500000 \\ 0.668478 & 1.097826 & 0.4728261 & -1.967391 & 0. & 0 & -0.500000 & 3 \end{pmatrix}$$

with eigenvalues equal to

(8.20923183 6.85510786 0.10662309 0.29046074 1.37934653 2.27653031 4.05189158 3.51981769)

We see that elements of those two matrices starting differ between each other after fifth decimal place. Below, Standard Deviations derived from the Exact and Asymptotic Fisher Information Matrices for different number of observations are presented.

STD	parameters	α_{11}	α_{21}	α_{12}	α_{22}	β_{11}	β_{21}	β_{12}	β_{22}
N	by method	0.8	1.2	-0.2	0.2	0	0.5	-1	-0.5
10	exact	0.5638	0.4841	0.3418	0.3213	0.6599	0.4816	0.4654	0.4680
	asymptotic	0.5150	0.4344	0.3139	0.2931	0.5630	0.3729	0.4361	0.4403
30	exact	0.3037	0.2577	0.1852	0.1734	0.3385	0.2283	0.2558	0.2582
	asymptotic	0.2973	0.2508	0.1812	0.1692	0.3251	0.2153	0.2518	0.2542
100	exact	0.1638	0.1384	0.0999	0.0933	0.1801	0.1198	0.1385	0.1398
	asymptotic	0.1628	0.1374	0.0993	0.0927	0.1780	0.1179	0.1379	0.1392
10^3	exact	0.0515	0.0435	0.0314	0.0293	0.0564	0.0374	0.0436	0.0440
	asymptotic	0.0515	0.0434	0.0314	0.0293	0.0563	0.0373	0.0436	0.0440
10^4	exact	0.0163	0.0137	0.0099	0.0093	0.0178	0.0118	0.0138	0.0139
	asymptotic	0.0162	0.0137	0.0099	0.0093	0.0178	0.0118	0.0137	0.0139
10^5	exact	0.0051	0.0043	0.0031	0.0029	0.0056	0.0037	0.0044	0.0044
	asymptotic	0.0052	0.0043	0.0031	0.0029	0.0056	0.0037	0.0044	0.0044
10^6	exact	0.0016	0.0014	0.0010	0.0009	0.0018	0.0012	0.0014	0.0014
	asymptotic	0.0016	0.0014	0.0010	0.0009	0.0018	0.0012	0.0014	0.0014

It appear that asymptotic approximation of STD based on 1000 observations is already

very good estimate of the exact STD and that with 10^6 observations the difference between the two come out after eight decimal place. Whereas, for small samples the difference between asymptotic and exact STD is not negligible anymore.

4 Examples

The first example will serve to illustrate the procedure. The second example will show an application of the method.

4.1 Illustration of the procedure

In order to clarify the procedure given above, we illustrate with an example a VMA(1) where $m = 2,\ N = 5,\ p = 0,\ s = 1,\ h = 1$ and $n = 2$.

1. The parameters of the model are the following:
   $\beta_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ \beta_1 = \begin{bmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{bmatrix}$, where $\beta_{11} = 0.8,\ \beta_{21} = 0.4,\ \beta_{12} = 0.2,\ \beta_{22} = 0.3,$
   $\theta = [\beta_{11},\ \beta_{21},\ \beta_{12},\ \beta_{22}]^\top$ and $Q = \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}$.

2. The cross-covariances (δ) are given by
   $\delta(0) = Q = \begin{bmatrix} 4 & 1 \\ 1 & 2 \end{bmatrix}$ and

   $$\delta(1) = -\beta_1 Q = \begin{bmatrix} -4\beta_{11} - \beta_{12} & -\beta_{11} - 2\beta_{12} \\ -4\beta_{21} - \beta_{22} & -\beta_{21} - 2\beta_{22} \end{bmatrix} = \begin{bmatrix} -3.4 & -1.2 \\ -1.9 & -1.0 \end{bmatrix}.$$

   It is easy to show that the derivatives are

   $$\frac{\partial\, \mathrm{vec}\delta(0)}{\partial\theta^\top} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \frac{\partial\, \mathrm{vec}\delta(1)}{\partial\theta^\top} = \begin{bmatrix} -4 & 0 & -1 & 0 \\ 0 & -4 & 0 & -1 \\ -1 & 0 & -2 & 0 \\ 0 & -1 & 0 & -2 \end{bmatrix}.$$

   For a moving average model, we do not need the autocovariances (Γ), so their computations have been omitted.

3. The matrices $\Phi,\ F$ and H are

   $$\Phi = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},\ F = \begin{bmatrix} -0.8 & -0.2 \\ -0.4 & -0.3 \end{bmatrix} \text{ and } H^\top = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

4. Computation of $P_{1|0}H^\top$ and $\left(\partial\, \mathrm{vec}P_{1|0}H^\top / \partial\, \theta^\top\right)$ yields

$$P_{1|0}H^\top = \left[-\beta_1\delta^\top(1)\right] = -\begin{bmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{bmatrix}\begin{bmatrix} -4\beta_{11}-\beta_{12} & -4\beta_{21}-\beta_{22} \\ -\beta_{11}-2\beta_{12} & -\beta_{21}-2\beta_{22} \end{bmatrix} =$$

$$\begin{bmatrix} 4\beta_{11}^2+2\beta_{12}\beta_{11}+2\beta_{12}^2 & 4\beta_{21}\beta_{11}+\beta_{21}\beta_{12}+\beta_{22}\beta_{11}+2\beta_{22}\beta_{12} \\ 4\beta_{21}\beta_{11}+\beta_{21}\beta_{12}+\beta_{22}\beta_{11}+2\beta_{22}\beta_{12} & 4\beta_{21}^2+2\beta_{21}\beta_{22}+2\beta_{22}^2 \end{bmatrix},$$

$$\frac{\partial\,\mathrm{vec}P_{1|0}H^\top}{\partial\theta^\top} = \begin{bmatrix} 8\beta_{11}+2\beta_{12} & 0 & 2\beta_{11}+4\beta_{12} & 0 \\ 4\beta_{21}+\beta_{22} & 4\beta_{11}+\beta_{12} & \beta_{21}+2\beta_{22} & \beta_{11}+2\beta_{12} \\ 4\beta_{21}+\beta_{22} & 4\beta_{11}+\beta_{12} & \beta_{21}+2\beta_{22} & \beta_{11}+2\beta_{12} \\ 0 & 8\beta_{21}+2\beta_{22} & 0 & 2\beta_{21}+4\beta_{22} \end{bmatrix}$$

which gives

$$P_{1|0}H^\top = \begin{bmatrix} 2.96 & 1.72 \\ 1.72 & 1.06 \end{bmatrix} \text{ and } \frac{\partial\,\mathrm{vec}P_{1|0}H^\top}{\partial\theta^\top} = \begin{bmatrix} 6.8 & 0.0 & 2.4 & 0.0 \\ 1.9 & 3.4 & 1.0 & 1.2 \\ 1.9 & 3.4 & 1.0 & 1.2 \\ 0.0 & 3.8 & 0.0 & 2.0 \end{bmatrix}.$$

5. The initial values of Chandrasekhar relations yield :

$$B_1 = H\ P_{1|0}H^\top + Q = \begin{bmatrix} 6.96 & 2.72 \\ 2.72 & 3.06 \end{bmatrix},$$

$$Y_1 = F\ Q = \begin{bmatrix} -3.4 & -1.2 \\ -1.9 & -1.0 \end{bmatrix},$$

$$K_1 = Y_1B_1^{-1} = \begin{bmatrix} -0.5137 & 0.0645 \\ -0.2226 & -0.1289 \end{bmatrix},$$

$$X_1 = -B_1^{-1} = \begin{bmatrix} -0.2202 & 0.1957 \\ 0.1957 & -0.5007 \end{bmatrix}.$$

The derivatives of the initial values of Chandrasekhar :

$$\frac{\partial\ \mathrm{vec}B_1}{\partial\theta^\top} = (I_2\otimes H)\ \frac{\partial\ \mathrm{vec}P_{1|0}H^\top}{\partial\theta^\top} = \begin{bmatrix} 6.8 & 0.0 & 2.4 & 0.0 \\ 1.9 & 3.4 & 1.0 & 1.2 \\ 1.9 & 3.4 & 1.0 & 1.2 \\ 0.0 & 3.8 & 0.0 & 2.0 \end{bmatrix},$$

$$\frac{\partial\ \mathrm{vec}Y_1}{\partial\theta^\top} = (Q\otimes I_2)\,\frac{\partial\ \mathrm{vec}F}{\partial\theta^\top} = \begin{bmatrix} -4 & 0 & -1 & 0 \\ 0 & -4 & 0 & -1 \\ -1 & 0 & -2 & 0 \\ 0 & -1 & 0 & -2 \end{bmatrix},$$

$$\begin{aligned} \frac{\partial\ \mathrm{vec}K_1}{\partial\theta^\top} &= \left(B_1^{-1}\otimes I_2\right)\frac{\partial\ \mathrm{vec}Y_1}{\partial\theta^\top} - (I_2\otimes Y_1)\left(B_1^{-1}\otimes B_1^{-1}\right)\frac{\partial\ \mathrm{vec}B_1}{\partial\theta^\top} \\ &= \begin{bmatrix} -0.1339 & -0.3421 & 0.3279 & -0.1124 \\ 0.3044 & -0.8324 & 0.1024 & 0.1026 \\ 0.1112 & 0.7948 & -0.7772 & 0.2593 \\ -0.1324 & 0.8206 & -0.0183 & -0.5732 \end{bmatrix}, \end{aligned}$$

$$\begin{aligned}\frac{\partial\ \mathrm{vec}X_1}{\partial\theta^\top} &= \left(B_1^{-1}\otimes B_1^{-1}\right)\frac{\partial\ \mathrm{vec}B_1}{\partial\theta^\top}\\ &= \begin{bmatrix} 0.1659 & -0.1474 & 0.0302 & -0.0268\\ -0.0107 & 0.1327 & 0.0451 & -0.0177\\ -0.0107 & 0.1327 & 0.0451 & -0.0177\\ -0.1120 & 0.2865 & -0.1041 & 0.2663\end{bmatrix}.\end{aligned}$$

6. We also give an example of the *commutation* and *permutation* matrices. The *block-permutation* matrix $M_{2,2}^b$ and the commutation matrix $M_{4,2}$ are given by :

$$M_{2,2}^b = \begin{bmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1\end{bmatrix},\quad M_{4,2} = \begin{bmatrix} 1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&1\end{bmatrix}.$$

7. Some of the initial values of the components of EFIM are

$$\begin{aligned}E\left\{\frac{\partial\widetilde{x}_1}{\partial\theta^\top}\otimes\frac{\partial\widetilde{x}_1}{\partial\theta^\top}\right\}^\top &= \left\{\frac{\partial\,\mathrm{vec}F}{\partial\theta^\top}\otimes\frac{\partial\,\mathrm{vec}F}{\partial\theta^\top}\right\}^\top\left\{\left(M_{4,2}\left[\mathrm{vec}Q\otimes I_2\right]M_{1,2}\right)\otimes I_2\right\}\\ &= \begin{bmatrix} 4&0&1&0&0&0&0&0&1&0&2&0&0&0&0&0\\ 0&4&0&1&0&0&0&0&0&1&0&2&0&0&0&0\\ 0&0&0&0&4&0&1&0&0&0&0&0&1&0&2&0\\ 0&0&0&0&0&4&0&1&0&0&0&0&0&1&0&2\end{bmatrix}^\top,\end{aligned}$$

$$\begin{aligned}E\left\{\left(\frac{\partial\widetilde{x}_1}{\partial\theta^\top}\right)^\top\otimes\widetilde{x}_1\right\} &= \left\{\left(\frac{\partial\,\mathrm{vec}F}{\partial\theta^\top}\right)^\top\otimes F\right\}\left(M_{4,2}\left[\mathrm{vec}Q\otimes I_2\right]M_{1,2}\right)\\ &= \begin{bmatrix} 3.4&1.9&0.0&0.0&1.2&1.0&0.0&0.0\\ 0.0&0.0&3.4&1.9&0.0&0.0&1.2&1.0\end{bmatrix}^\top,\end{aligned}$$

$$E\left\{\widetilde{x}_1\otimes\widetilde{x}_1\right\} = \left(F\ \otimes F\right)\mathrm{vec}Q = \begin{bmatrix} 2.96\\ 1.72\\ 1.72\\ 1.06\end{bmatrix}.$$

The initial value of $\mathrm{vec}J_1(\theta)$ is

$$\mathrm{vec}J_1(\theta) = \frac{1}{2}\left[\left(\frac{\partial\ \mathrm{vec}B_1}{\partial\theta^\top}\right)\otimes\left(\frac{\partial\ \mathrm{vec}B_1}{\partial\theta^\top}\right)\right]^\top\left[I_2\otimes B_t^{-1}\otimes B_t^{-1}\otimes I_2\right]\mathrm{vec}I_4,$$

so we deduce

$$J_1(\theta) = \begin{bmatrix} 0.5436 & -0.2493 & 0.1883 & -0.1249\\ -0.2493 & 0.9954 & -0.0443 & 0.4457\\ 0.1883 & -0.0443 & 0.0813 & -0.0499\\ -0.1249 & 0.4457 & -0.0499 & 0.2450\end{bmatrix}.$$

8. Finally, after 5 recurrences, we obtain the following exact information matrix

$$J_5(\theta) = \begin{bmatrix} 3.8699 & -2.8925 & 0.6952 & -0.4612 \\ -2.8925 & 9.4501 & -1.9187 & 1.3993 \\ 0.6952 & -1.9187 & 2.6913 & -1.5903 \\ -0.4612 & 1.3993 & -1.5903 & 3.4271 \end{bmatrix}.$$

4.2 Application

The series were simulated (Nsiri and Roy, 1996) using a VARMA(1,2) model

$$z_t = \begin{pmatrix} 0.6 & 0 \\ 0.5 & -0.5 \end{pmatrix} z_{t-1} + w_t - \begin{pmatrix} 0.8 & -0.2 \\ 0 & 0 \end{pmatrix} w_{t-1} - \begin{pmatrix} -0.85 & -0.8 \\ 0 & 0 \end{pmatrix} w_{t-2},$$

with the number of observations $N = 120$, and covariance matrix $Q = I_2$. Contrarily with the intention of that paper, which was to investigate refined echelon forms in order to reduce the number of parameters, we have estimated all the coefficients of the VARMA(1,2) model. The column "estimate" contains the exact maximum likelihood estimators obtained by merosa (Mélard et al., 2004) which makes use of the NAG library [As far as we know, no standard software package fits VARMA models using the exact maximum likelihood estimator ; the authors rejected the idea to compute the exact information matrix on the basis of a non-exact estimator]. The column "merosa/StdErr" are the standard errors provided by the software as a by-product of the optimisation procedure. The column "KMNZ/StdErr" are the square roots of the elements of the matrix obtained by inverting the exact information matrix following the method of this paper. The two columns "t-stat" are the Student statistics (ratio of the estimate by the standard error, distributed as a standard normal distribution if the corresponding coefficient is equal to 0) associated to the merosa estimates using the respective standard errors. Here are the results.

Parameter value in the simulation	merosa estimate	merosa StdErr	merosa t-stat	KMNZ StdErr	KMNZ t-stat
$\alpha^1_{11} = 0.6$	0.4804	0.0948	5.0675	0.1134	4.2363
$\alpha^1_{21} = 0.5$	0.5299	0.0421	12.5867	0.0559	9.4794
$\alpha^1_{12} = 0$	0.2200	0.1888	1.1653	0.2417	0.9102
$\alpha^1_{22} = -0.5$	−0.5434	0.0681	−7.9794	0.1056	−5.1458
$\beta^1_{11} = 0.8$	0.6199	0.0956	6.4843	0.1028	6.0302
$\beta^1_{21} = 0$	0.0409	0.0834	0.4904	0.1038	0.3940
$\beta^1_{12} = -0.2$	0.0775	0.2091	0.3706	0.2472	0.3135
$\beta^1_{22} = 0$	−0.0779	0.1077	−0.7233	0.1348	−0.5778
$\beta^2_{11} = -0.85$	−0.7635	0.0805	−9.4845	0.1074	−7.1089
$\beta^2_{21} = 0$	0.1039	0.0745	1.3946	0.0860	1.2081
$\beta^2_{12} = -0.8$	−1.0160	0.1328	−7.6506	0.1645	−6.1763
$\beta^2_{22} = 0$	0.0710	0.0816	0.8701	0.0951	0.7465

It should be noted that all the coefficients having a true zero value are not significant at 5% level.

A Program for a VARMA Model

In this appendix we describe a Matlab program for computing the exact Fisher information matrix (EFIM) in the case of a VARMA model. The choice of Matlab is motivated by the fact that the complex matrix structure of the several relations is relatively easy to implement in this environment. The main program `EXFIM` is subdivided in a certain number of procedures and we will give the purpose of each of them.

A.1 Framework

1. In the program **EXFIM,** for computing the exact Fisher Information Matrix the following parameters are read:

   the length N of time series,

   the value of the AR and MA matrix parameters, $(\alpha_i;\ i = 1, ..., p)$ and $\left(\beta_j;\ j = 1, ..., q\right)$,

   the covariance matrix Q of the white noise w_t.

   The orders of the AR and MA polynomials, p and q, are deduced from the size of the parameters. **EXFIM** calls the following procedures: **autocovd**, **primod** and **FIM**.

2. In procedure **autocovd**, see Niemczyk (2004), the autocovariances of the process z_t $(E\left\{z_t\, z_{t-k}^{\top}\right\})$, cross-covariances $(E\left\{z_t\, w_{t-k}^{\top}\right\})$ and their derivatives with respect to θ are computed.

3. In procedure **primod,** the matrices H, F, Φ are created and their derivatives computed. **Primod** also checks that the model is stationary and invertible.

4. In procedure **DP10H,** the product $P_{1|0}\, H^T$ and its derivatives are computed.

5. Procedure **FIM** consists in a loop over time, $t = 1, ..., N$, where, for $t = 1$, the Chandrasekhar recursions and their derivatives plus the components of EFIM, are initialized. For $t > 1$, **EFIM** procedure is used to update the expression (16) and then the elements of the Chandrasekhar recursions (5-10) so that the components of EFIM are updated.

A.2 Matlab program :

The algorithm programmed in the MATLAB environment is available from the authors. It can be translated in any other matrix environment where the Kronecker product is defined. The program is subdivided into 6 procedures described below. Let $n = mh$, $\ell = pm^2 + qm^2$ and $\theta = \mathrm{vec}[\alpha_1, \ldots, \alpha_p, \beta_1, \ldots, \beta_q]$.

A.2.1 EXFIM

Parameters needed for the computation are read and the exact Fisher Information Matrix is computed.

`[FM]=EXFIM(AR,MA,S,N)`

variable	type		description
AR	`real` array $(p\,m \times m)$	input:	the matrix of the AR coefficient $AR = [\alpha_1, \ldots, \alpha_p]$
MA	`real` array $(q\,m \times m)$	input:	the matrix of the MA coefficient $MA = [\beta_1, \ldots, \beta_q]$
S	`real` array $(m \times m)$	input:	Q the covariance matrix of w_t
N	`integer`	input:	the length of the times series
FM	`real` array $(\ell \times \ell)$	output:	the exact Fisher Information Matrix

A.2.2 autocovd:

The autocovariances of the process z and the cross-covariances between z and w are computed. Details for the computations can be found in [13].

`[E,dE,DGG,GG]=autocovd(AR,MA,S,h)`

variable	type		description
AR	`real` array $(p\,m \times m)$	input:	the matrix of the AR coefficient
MA	`real` array $(q\,m \times m)$	input:	the matrix of the MA coefficient
S	`real` array $(m \times m)$	input:	Q the covariance matrix of w_t
h	`integer`	input:	$h = \max\{p, q\}$ the maximum lag at which we compute the autocovariance
E	`real` array $(n \times m)$	output:	matrix of cross-covariances $E = [(Ez_t w_t^\top)^\top, \ldots, (Ez_t w_{t-h}^\top)^\top]^\top$
dE	`real` array $(m\,n \times \ell)$	output:	matrix of the derivatives of E $dE = \partial \mathrm{vec}[Ez_t w_t^\top, \ldots, Ez_t w_{t-h}^\top]/\partial \mathrm{vec}\theta$
GG	`real` array $(n \times m)$	output:	matrix of the autocovariances $GG = [(Ez_t w_t^\top)^\top, \ldots, (Ez_t w_{t-h}^\top)^\top]^\top$
DGG	`real` array $(m\,n \times \ell)$	output:	matrix of derivatives of GG $DGG = \partial \mathrm{vec}[Ez_t w_t^\top, \ldots, Ez_t w_{t-h}^\top]/\partial \mathrm{vec}\theta$

A.2.3 primod:

Initialization and computation of : H, F, Φ, F^θ and Φ^θ.

`[P,PD,F,FD,H]=primod(AR,MA,m)`

variable	type		description
AR	`real` array $(p\,m \times m)$	input:	the matrix of the AR coefficient
MA	`real` array $(q\,m \times m)$	input:	the matrix of the MA coefficient
m	`integer`	input:	m
P	`real` array $(n \times n)$	output:	Φ
PD	`real` array $(n^2 \times \ell)$	output:	$\Phi^\theta = \partial \mathrm{vec}\Phi/\partial\theta^\top$
F	`real` array $(n \times m)$	output:	F
FD	`real` array $(m\,n \times \ell)$	output:	$F^\theta = \partial \mathrm{vec}F/\partial\theta^\top$
H	`real` array $(m \times n)$	output:	H

A.2.4 DP10HT :

Computation of $P_{1|0}H^\top$ and $\partial\ (\mathrm{vec}P_{1|0}H^\top)/\partial\theta^\top$.

```
[P10H,DP10H]=dp01ht(E,dE,DGG,GG,AR,MA,m)
```

variable	type		description
E, dE	`real`	input:	output of `autocovd` procedure
GG, DGG	`real`	input:	output of `autocovd` procedure
AR, MA	`real`	input:	AR and MA coefficients
m	`integer`	input:	m
$P10H$	`real` array $(n \times m)$	output:	$P_{1\|0}H^\top$
$DP10H$	`real` array $(m\,n \times \ell)$	output:	$\partial \mathrm{vec}P_{1\|0}H^\top/\partial\theta^\top$.

A.2.5 FIM:

This main procedure computes the Exact Fisher Information Matrix. It calls for another procedure, named **EFIM**, which updates expressions (51)-(61) described in [2] needed to accomplish the computation of EFIM. Inputs of this procedure are: S, P, PD, F, FD, H, $P10H$, $DP10H$ which were derived using previous procedures together with p, q, m, and N. Output of the procedure is FM which is $\ell \times \ell$ Fisher Information Matrix of our model.

```
[FM]=FIM(S,P,PD,F,FD,H,P10H,DP10H,p,q,m,N)
```

A.2.6 Additional functions:

function	description
MXX	computes the block-permutation matrix $M^b_{h,r}$ described in [3],
MMN	computes the commutation matrix $M_{m\,n}$ introduced in [3],
vec	vectorization of a matrix,
devec	de-vectorization of a matrix.

References

[1]

[2] KLEIN, A., MELARD, G, and ZAHAF, T. 1998. Computation of the exact information matrix of Gaussian dynamic regression time series models, Ann. Statist. 26, 1636-1650.

[3] KLEIN, A., MELARD, G, and ZAHAF, T. 2000. Construction of the exact Fisher information matrix of Gaussian time series models by means of matrix differential rules, *Linear Algebra and its Applications.* **321:** 209-232.

[4] KLEIN, A., MELARD, G, and SPREIJ, P. 2004.

[5] KLEIN, A., MELARD, G, and NIEMCZYK, J. 2004. Correction of "Construction of the exact Fisher information matrix of Gaussian time series models by means of matrix differential rules", *Linear Algebra and its Applications,* submitted.

[6] KLEIN, A. and NEUDECKER, H. 2000. A direct derivation of the exact Fisher information matrix of Gaussian vector state space models, *Linear Algebra and its Applications.* **321:** 233-238.

[7] KLEIN, A. and SPREIJ, P. 1996. On Fisher's information matrix of an ARMAX process and Sylvester's resultant matrices, *Linear Algebra and its Applications* **237/238:** 579-590.

[8] LJUNG SÖDERSTRÖM

[9] MATLAB

[10] MELARD, G. and KLEIN, A. 1994. On a fast algorithm for the exact information matrix of a Gaussian ARMA time series, *IEEE Trans. Signal Processing* **42:** 2201-2203.

[11] MORF, M., SIDHU, G. S. and KAILATH, T. 1974. Some new algorithms for recursive estimation on constant, linear, discrete-time systems, *IEEE Trans. Automat. Contr.* **19:** 315-323.

[12] NAG

[13] NIEMCZYK, J. 2004. Computing the derivatives of the autocovariances of a VARMA process, COMPSTAT'2004 16th Symposium held in Prague, *Proceedings in Computational Statistics,* Jaromir Antoch (Ed.), Physica-Verlag, Heidelberg, pp. 1593-1600.

[14] NSIRI ROY

[15] PEARLMAN

[16] PORAT, B. and FRIEDLANDER, B. 1986.Computation of the exact information matrix of Gaussian time series with stationary random components, *IEEE Trans. Acoust., Speech, Signal Processing* **14:** 118-130.

[17] SHEA, B. L. 1989. The exact likelihood of a vector autoregressive moving average model. *J. Royal Statist. Soc. Ser. C* **38:** 161-184.

[18] TERCEIRO LOMBA, J. 1990. *Estimation of Dynamic Econometric Models with Errors in Variables,* Springer-Verlag, Berlin.

[19] ZADROZNY, P. A. 1989. Analytical derivatives for estimation of linear dynamic models, *Computers and Mathematics with Applications* **18:** 539-553.

[20] ZADROZNY, P. A. 1992. Errata to 'Analytical derivatives for estimation of linear dynamic models'. *Computers and Mathematics with Applications* **24:** 289-290.

[21] ZAHAF