An algorithm for computing the asymptotic Fisher information matrix for seasonal SISO models

by

André Klein[1], and Guy Mélard[2*]

University of Amsterdam[1] and Université Libre de Bruxelles[2]

Abstract

The paper presents an algorithm for computing the asymptotic Fisher information matrix of a possibly seasonal single input single output (SISO) time series model. That matrix is a block matrix whose elements are basically integrals over the oriented unit circle of rational functions. The procedure makes use of the autocovariance function of one or the cross-covariance function of two autoregressive processes based

[*] A. Klein is with the Department of Actuarial Sciences and Econometrics, University of Amsterdam, Roetersstraat 11, 1018 WB Amsterdam, The Netherlands (phone +31-20-5254245, fax: +31-20-5254349, e-mail:aklein@fee.uva.nl).

G. Mélard is with the Faculty of Social Sciences, Politics and Economics, ECARES and ISRO, 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).

on the same noise. The algorithm also works when the input variable is omitted, the case of a seasonal ARMA model.

AMS 1991 *subject classification.* Primary 62M10; secondary 62F15.

Key words. Fisher information matrix, SISO model.

1 Introduction

In this paper we consider the most general single input single output (SISO) model, i.e. a 6-polynomial dynamic model, relating an explanatory or input variable and a dependent or output variable. A justification for the 6-polynomial SISO model is given in Section 2. We study the SISO model it in its seasonal flavor, where each polynomial is factored into a regular polynomial, a polynomial in the lag operator L, and a seasonal polynomial, a polynomial in L^s, where s is the seasonal period, 4 for quarterly data and 12 for monthly data. That family of models is often useful with economic or management variables. In engineering, it is the nonseasonal form of the model which is the most popular. We assume either that the process is Gaussian or that the assumptions of Drost *et al.* (1997) are fulfilled. Under these assumptions, we obtain a simple and efficient algorithm for the asymptotic Fisher information matrix. The efficiency of the computational algorithm is due to replacing circular integrals by computation of the autocovariance function of a scalar AR process.

The asymptotic Fisher information matrix is an important tool for evaluating the

accuracy of a parameter estimation technique. The inverse of the Fisher information matrix yields the Cramér-Rao bound which provides asymptotically a lower bound for the covariance matrix of unbiased estimators, and even the covariance matrix itself for efficient estimation methods. Furthermore, the asymptotic Fisher information matrix has been used recently in an on-line estimation method of the parameters of ARMA models, where the estimates are updated at each time by means of recurrence equations. Zahaf (1998) has proposed such an approach but limited to nonseasonal and seasonal ARMA models because of lack of availability of a computational procedure for the asymptotic information matrix of more general models. The present work will thefore have an immediate application.

With respect to Klein and Mélard (1994a), there are four kinds of improvements: (1) the parametrisation there was limited to nonseasonal models; (2) an intercept is added to the model; (3) the details of the algorithm were just sketched; (4) the result is not restricted to Gaussian processes. The algorithm described in this paper avoids crude evaluation, i. e. computation by numerical integration or quadrature, of circular integrals of rational functions in favour of a simple symbolic representation of the integrands, the use of a generic polynomial multiplier and an Euclid algorithm. An algorithm for seasonal models has been decribed by Klein and Mélard (1990) but only for ARMA models. The algorithmic aspects detailed here were not necessary because of the simplicity of the integrands found in that case.

The paper is also related to Klein *et al.* (1998) but the exact information matrix which is the main subject of that paper is not used here, mainly because the method is too complex for the general seasonal model considered here. Generalisation to the MISO model can be considered, see Klein and Mélard (1994b). Several authors have given expressions in a more general framework, e.g. Zadrozny (1989, 1992), Zadrozny and Mittnik (1994), and Terceiro (1990). See also Klein *et al.* (2000).

The article is organized as follows. In Section 2, we formulate the model. In Section 3 a general expression for the asymptotic information matrix for SISO models is given. Examples are provided in Section 4. The algorithm is described in the Appendix.

2 The model

Let s denote the seasonal period, e.g. 12 for monthly data, 4 for quarterly data. Consider the single-input single-output (SISO) model defined by the equation

$$\frac{a(L)A(L^s)}{f(L)F(L^s)}y(t) = \mu + b_0\frac{b(L)B(L^s)}{e(L)E(L^s)}x(t) + \frac{c(L)C(L^s)}{d(L)D(L^s)}u(t), \tag{1}$$

where $y(t)$ is the output variable, $x(t)$ is the input variable, $\{u(t)\}$ is a sequence of identically and independently distributed (i. i. d.) random variables with zero mean and standard deviation σ, μ is the intercept, $a(L)$, $b(L)$, $c(L)$, $d(L)$, $e(L)$, and $f(L)$ are (regular) polynomials in the unit delay operator L, $A(L^s)$, $B(L^s)$, $C(L^s)$, $D(L^s)$, $E(L^s)$, and $F(L^s)$ are (seasonal) polynomials in L^s. For example $a(L) =$

$1 + a_1 L + a_2 L^2 + \cdots + a_{n_1} L^{n_1}$, for some n_1. It is assumed that $\{u(t)\}$ and $\{x(t)\}$ are independent for all leads and lags. A delay can be considered for the input variable so that $x(t)$ is replaced by $x(t-l)$, for some (non necessarily positive) integer l. This will have no effect on our derivations (see Example 5). The model depends on σ^2 and on d parameters which are the coefficients of the 12 polynomials plus b_0 and μ.

The input variable can be stochastic or deterministic. In the former case, statistical inference can be done conditionally on the value which is taken. See Klein *et al.* (1998) for either the deterministic case or the stochastic and conditional case. Their inconvenience is that they depend on the data, i. e. they are not suited at the experimental design stage, see Dahran (1985). Also, that approach is not practical when the information matrix is used within a recurrence estimation scheme, e. g. Zahaf (1998). Here, we study the stochastic and unconditional case. This is because we need to evaluate covariances involving $x(t)$. Although, using the approach of this paper, more modelling effort is required for the input series, it will be seen in some of the examples that the results obtained using that approach are very similar to those obtained from software that conditions upon the input values.

Let us suppose that the input variable $x(t)$ is generated by a seasonal ARMA model defined by the equation

$$x(t) = \frac{h(L)H(L^s)}{g(L)G(L^s)} v(t), \tag{2}$$

where $\{v(t)\}$ is a sequence of i. i. d. random variables with zero mean and standard

deviation τ, $g(L)$, and $h(L)$ are polynomials in L, $G(L^s)$ and $H(L^s)$ are polynomials in L^s.

For simplicity we suppose that each of the polynomials has a constant term equal to 1, except possibly $b(L)$ and $B(L^s)$. The contemporary regression coefficient b_0 is generally different from 0 or 1. All the polynomials are assumed to have their roots outside of the unit circle. Klein and Spreij (1996) have considered ARMAX models of the form

$$a^*(L)y(t) = b^*(L)x(t) + c^*(L)u(t),$$

where $a^*(L)$, $b^*(L)$, and $c^*(L)$ are polynomials. They have set forth conditions for an ARMAX model to be identifiable by studying the kernel of the corresponding Fisher information matrix and its relations with Sylvester matrices. They prove that a necessary and sufficient condition is that the polynomials $a^*(L)$, $b^*(L)$, and $c^*(L)$ have no common root. It can be shown, by inspecting the ARMAX representation of our model, see (6) below, that the following pairs of products of polynomials should have no common root: $a(L)A(L)$ and $f(L)F(L)$, $b(L)B(L)$ and $e(L)E(L)$, $c(L)C(L)$ and $d(L)D(L)$, $d(L)D(L)$ and $e(L)E(L)$, $d(L)D(L)$ and $f(L)F(L)$, $e(L)E(L)$ and $f(L)F(L)$, and also that the following triplet should have no common root: $a(L)A(L)$, $b(L)B(L)$, and $c(L)C(L)$.

Note that the 3-rational operators SISO model (1) encompasses the Box-Jenkins transfer function model, in which the operator on $y(t)$ is omitted, and the ARMAX

model in which the denominators of the operators on $y(t)$ and $x(t)$ are omitted.

It will sometimes be easier to refer to the polynomials $a(L)$, $b(L)$, ..., $f(L)$, $A(L^s)$, $B(L^s)$, ..., $F(L^s)$ in that order, as $\theta_1(L)$, $\theta_2(L)$, ..., $\theta_6(L)$, $\theta_7(L^s)$, $\theta_8(L^s)$, ..., $\theta_{12}(L^s)$. The degree of $\theta_r(L)$ is denoted by n_r. The number of coefficients of $\theta_r(L)$ or $\theta_r(L^s)$ which are not identically equal to 0 is denoted by p_r. The parameters included in $\theta_r(L)$ will be stored in the order of increasing power in L in a $p_r \times 1$ vector denoted by θ_r. We have $p_r = n_r$ for $r \leq 6$ and $sp_r = n_r$ for $r > 6$. We suppose that all the parameters are functionally independent so that the total number of parameters, including $\theta_{13} = b_0$ and $\theta_{14} = \mu$, is equal to $p = \sum_{r=1}^{12} p_r + 2$. Let θ be the $p \times 1$ vector of these parameters in the same order as before and with b_0 and μ in the last two positions. θ will be considered as a 14×1 block vector. Of course, the model doesn't need to be used with all the polynomials.

Let us consider the framework of Drost *et al.* (1997), which provides a generalisation to the time series context of the efficient estimation theory for semiparametric models due to Bickel *et al.* (1993). We suppose that the assumptions needed for their results hold. For the sake of simplicity, we omit the input variable $x(t)$ at this stage. Let $y^N = (y(1), ..., y(N))^T$ be an observed series of length N assumed to satisfy the stochastic difference equation (1), where $\{u(t)\}, t = 0, \pm 1, \pm 2, ...$ is a i. i. d. sequence of random variables with mean zero and probability density f. The latter is specified up to a scale factor σ. Let f_1 be the standard version of f: $f(u) = f_1(u/\sigma)/\sigma$. Given

the assumptions which have been made, there is a stationary solution to (1) and we suppose that the observed series is such a stationary solution (this is not necessary, see Hallin and Werker, 1998, for a discussion). The parameters θ are the coefficients in (1).

Let $\mathfrak{L}(\theta; y^N)$ be the likelihood function of the sample, the density of the vector y^N of observations considered as a function of the parameters, and $\mathbb{E}$ denotes the expectation. The information bound which is the generalisation of the Cramér-Rao (lower) bound (CRB) is the matrix

$$J(\theta) = -\mathbb{E}\left(\frac{\partial^2 \log \mathfrak{L}(\theta; y^N)}{\partial\theta\partial\theta^T}\right) \tag{3}$$

where ^{T} denotes transposition, where the expectation is evaluated at the true unknown value of $\theta = \theta_0$ and with respect to the true law. If the estimation method yields asymptotically efficient estimators, for instance the case of a maximum likelihood estimator $\hat{\theta}$ of the parameter θ under the appropriate assumptions, a good estimate of its asymptotic covariance matrix is $J(\hat{\theta})^{-1}$.

Omitting y^N from the notation, the log-likelihood $\ell(\theta) = \log \mathfrak{L}(\theta)$ is written in the form

$$\ell(\theta) = \sum_{t=1}^{N} \ell_t(\theta),$$

where $\ell_t(\theta) = \log(f(u(t;\theta)))$ and $u(t;\theta)$ is the residual obtained from solving (1) for $u(t)$. We ignore the effect of the initial conditions (see again the discussion in Hallin

and Werker, 1998; the initial conditions are handled in the so-called exact information matrix of Klein *et al.,* 1998, in the Gaussian case). Let us compute the first and second derivatives of $\ell_t(\theta)$:

$$\begin{aligned}\frac{\partial \log f(u(t;\theta))}{\partial \theta_j} &= \frac{\partial \log f(u(t;\theta))}{\partial u(t;\theta)}\frac{\partial u(t;\theta)}{\partial \theta_j},\\ \frac{\partial^2 \log f(u(t;\theta))}{\partial \theta_i \partial \theta_j} &= \frac{\partial^2 \log f(u(t;\theta))}{\partial (u(t;\theta))^2}\frac{\partial u(t;\theta)}{\partial \theta_i}\frac{\partial u(t;\theta)}{\partial \theta_j} + \frac{\partial \log f(u(t;\theta))}{\partial u(t;\theta)}\frac{\partial^2 u(t;\theta)}{\partial \theta_j \partial \theta_j}.\end{aligned}$$

Hence

$$J_{ij}(\theta) = -\sum_{t=1}^{N} \mathbb{E}\left(\frac{\partial^2 \log f(u(t;\theta))}{\partial (u(t;\theta))^2}\frac{\partial u(t;\theta)}{\partial \theta_i}\frac{\partial u(t;\theta)}{\partial \theta_j}\right). \tag{4}$$

for all i and j, since, in the case of our model, $\partial^2 u(t;\theta)/\partial\theta_j\partial\theta_j$ is measurable with respect to the σ-algebra spanned by $u(t-1;\theta)$, $u(t-2;\theta)$, ..., and

$$\mathbb{E}\left(\frac{\partial \log f(u(t;\theta))}{\partial u(t;\theta)}\right) = 0.$$

Let us denote $\phi_f = -\partial \log f(u)/\partial u = -(\partial f(u)/\partial u)/f(u)$, the score function of density f, and remark that

$$I_f \stackrel{\text{def}}{=} -\mathbb{E}\frac{\partial^2 \log f(u)}{\partial u^2} = -\int_{-\infty}^{\infty} \frac{\partial^2 \log f(u)}{\partial u^2} f(u)\mathrm{d}u = \mathbb{E}\left(\frac{\partial \log f(u)}{\partial u}\right)^2 = \frac{I_{f_1}}{\sigma^2},$$

where I_{f_1} is the Fisher information quantity associated with μ in the location family $\{f_1(u-\mu); \mu \in \mathbb{R}\}$. For standard normal, double exponential, and logistic laws, we have respectively $I_{f_1} = 1$, $I_{f_1} = 2$, and $I_{f_1} = \pi^2/9$. Hence, we have

$$J_{ij}(\theta) = I_f \mathbb{E}\left(\sum_{t=1}^{N} \frac{\partial u(t;\theta)}{\partial \theta_i}\frac{\partial u(t;\theta)}{\partial \theta_j}\right) = I_{f_1}\frac{N}{\sigma^2}\mathbb{E}\left(u_{\theta_i}(t)u_{\theta_j}(t)\right), \tag{5}$$

where $u_\theta(t) = \partial u(t)/\partial\theta$. In the time series context, quasi-maximum likelihood estimation is often used, generally by using the Gaussian likelihood or one of its approximations (the least squares method, for example). In that case, if the true innovation law f is known, the factor I_f determines the possible gain in efficiency provided by an adaptative estimation method, see Drost *et al.* (1997). In the sequel we focus on the calculation of the third factor which corresponds to the (standard, one-observation) information bound or Cramér-Rao lower bound for the model in (1) in the Gaussian case, and its inverse will provide the asymptotic covariance matrix of the Gaussian quasi-maximum likelihood estimator or a least-squares estimator of θ for that model. See e.g. Godolphin and Unwin (1983) and Friedlander (1984). Let $u_r(t)$ be the sub-vector of $u_\theta(t)$ associated to θ_r. $J(\theta)$ is computed block by block in the next section.

3 The asymptotic information matrix

Write (1) under the form

$$\begin{aligned} a(L)A(L^s)d(L)D(L^s)e(L)E(L^s)y(t) &= d(L)D(L^s)e(L)E(L^s)f(L)F(L^s)\mu \\ &\quad + b_0 b(L)B(L^s)d(L)D(L^s)f(L)F(L^s)x(t) \\ &\quad + c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u(t). \end{aligned} \tag{6}$$

Differentiating (1), for equations 10-12, 16-18 and 20 below, or (6), for equations 7-9, 13-15 and 19 below, successively with respect to θ_r, $r = 1, ..., 14$ gives :

$$c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u_1(t) = \tilde{a}(L)A(L^s)d(L)D(L^s)e(L)E(L^s)y(t) \tag{7}$$

$$c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u_2(t) = -b_0\tilde{b}(L)B(L^s)d(L)D(L^s)f(L)F(L^s)x(t) \tag{8}$$

$$c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u_3(t) = -\tilde{c}(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u(t) \tag{9}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_4(t) = \frac{\tilde{d}(L)c(L)C(L^s)}{d^2(L)D(L^s)}u(t) \tag{10}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_5(t) = b_0\frac{\tilde{e}(L)b(L)B(L^s)}{e^2(L)E(L^s)}x(t) \tag{11}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_6(t) = -\frac{\tilde{f}(L)a(L)A(L^s)}{f^2(L)F(L^s)}y(t) \tag{12}$$

$$c(L)\tilde{C}(L^s)e(L)E(L^s)f(L)F(L^s)u_7(t) = a(L)\tilde{A}(L^s)d(L)D(L^s)e(L)E(L^s)y(t) \tag{13}$$

$$c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u_8(t) = -b_0b(L)\tilde{B}(L^s)d(L)D(L^s)f(L)F(L^s)x(t) \tag{14}$$

$$c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u_9(t) = -c(L)\tilde{C}(L^s)e(L)E(L^s)f(L)F(L^s)u(t) \tag{15}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_{10}(t) = \frac{\tilde{D}(L^s)c(L)C(L^s)}{d(L)D^2(L^s)}u(t) \tag{16}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_{11}(t) = b_0\frac{\tilde{E}(L^s)b(L)B(L^s)}{e(L)E^2(L^s)}x(t) \tag{17}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_{12}(t) = -\frac{\widetilde{F}(L^s)a(L)A(L^s)}{f(L)F^2(L^s)}y(t) \tag{18}$$

$$\begin{aligned} c(L)C(L^s)e(L)E(L^s)f(L)F(L^s)u_{13}(t) \quad &= \\ -b(L)B(L^s)d(L)D(L^s)f(L)F(L^s)x(t) \end{aligned} \tag{19}$$

$$\frac{c(L)C(L^s)}{d(L)D(L^s)}u_{14}(t) = 1, \tag{20}$$

where $\tilde{\theta}_r(L) = (L, ..., L^{p_r})^T$ for $r = 1$ to 12. If we omit the arguments L and L^s, equations (7-20) can be written in block matrix form

$$
\begin{aligned}
u_\theta(t) \;=\; & \\
(21) \qquad & \begin{bmatrix}
\tilde{a}AdD/cCfF & 0 & 0 & 0 \\
0 & -b_0\tilde{b}BdD/cCeE & 0 & 0 \\
0 & 0 & -\tilde{c}/c & 0 \\
0 & 0 & \tilde{d}/d & 0 \\
0 & b_0\tilde{e}bBdD/cCe^2E & 0 & 0 \\
-\tilde{f}aAdD/cCf^2F & 0 & 0 & 0 \\
\tilde{A}adD/cCfF & 0 & 0 & 0 \\
0 & -b_0\tilde{B}bdD/cCeE & 0 & 0 \\
0 & 0 & -\tilde{C}/C & 0 \\
0 & 0 & \tilde{D}/D & 0 \\
0 & b_0\tilde{E}bBdD/cCeE^2 & 0 & 0 \\
-\tilde{F}aAdD/cCfF^2 & 0 & 0 & 0 \\
0 & -bBdD/cCeE & 0 & 0 \\
0 & 0 & 0 & -dD/cC
\end{bmatrix} \cdot \begin{bmatrix} y(t) \\ x(t) \\ u(t) \\ 1 \end{bmatrix}
\end{aligned}
$$

Using (1), we have

$$u_\theta(t) = \begin{bmatrix} b_0\tilde{a}bBdD/acCeE & \tilde{a}/a & 0 \\ -b_0\tilde{b}BdD/cCeE & 0 & 0 \\ 0 & -\tilde{c}/c & 0 \\ 0 & \tilde{d}/d & 0 \\ b_0\tilde{e}bBdD/cCe^2E & 0 & 0 \\ -b_0\tilde{f}bBdD/cCeEf & -\widetilde{f}/f & 0 \\ b_0\tilde{A}bBdD/AcCeE & \widetilde{A}/A & 0 \\ -b_0\tilde{B}bdD/cCeE & 0 & 0 \\ 0 & -\tilde{C}/C & 0 \\ 0 & \tilde{D}/D & 0 \\ b_0\tilde{E}bBdD/cCeE^2 & 0 & 0 \\ -b_0\tilde{F}bBdD/cCeEF & -\widetilde{F}/F & 0 \\ -bBdD/cCeE & 0 & 0 \\ 0 & 0 & -dD/cC \end{bmatrix} \cdot \begin{bmatrix} x(t) \\ u(t) \\ 1 \end{bmatrix} . \tag{22}$$

Replacing (2) in (22), we have

$$
(23)\qquad u_\theta(t) = \begin{bmatrix}
b_0\tilde{a}bBdDhH/acCeEgG & \tilde{a}/a & 0 \\
-b_0\tilde{b}BdDhH/cCeEgG & 0 & 0 \\
0 & -\tilde{c}/c & 0 \\
0 & \tilde{d}/d & 0 \\
b_0\tilde{e}bBdDhH/cCe^2EgG & 0 & 0 \\
-b_0\tilde{f}bBdDhH/cCeEfgG & -\widetilde{f}/f & 0 \\
b_0\tilde{A}bBdDhH/AcCeEgG & \widetilde{A}/A & 0 \\
-b_0\tilde{B}bdDhH/cCeEgG & 0 & 0 \\
0 & -\tilde{C}/C & 0 \\
0 & \tilde{D}/D & 0 \\
b_0\tilde{E}bBdDhH/cCeE^2gG & 0 & 0 \\
-b_0\tilde{F}bBdDhH/cCeEFgG & -\widetilde{F}/F & 0 \\
-bBdDhH/cCeEgG & 0 & 0 \\
0 & 0 & -dD/cC
\end{bmatrix} \cdot \begin{bmatrix} v(t) \\ u(t) \\ 1 \end{bmatrix}.
$$

Denote $\lambda^{(m)}(L)$ be the m-th column of the matrix in (23), $m = 1, 2, 3$. It is composed of block columns $\lambda_r^{(m)}(L)$, $r = 1, ..., 14$, with elements $\lambda_{ir}^{(m)}(L)$, $i = 1, ..., p_r$. Then, omitting argument θ,

$$
(24)\qquad (N\sigma^2)^{-1}J = (N\sigma^2)^{-1}(J^{(1)} + J^{(2)} + J^{(3)}),
$$

where

$$
\begin{aligned}
(N\sigma^2)^{-1} J^{(1)} &= E[\{\lambda^{(1)}(L)v(t)\}\{\lambda^{(1)}(L)v(t)\}^T] \\
(N\sigma^2)^{-1} J^{(2)} &= E[\{\lambda^{(2)}(L)u(t)\}\{\lambda^{(2)}(L)u(t)\}^T] \\
(N\sigma^2)^{-1} J^{(3)} &= \lambda^{(3)}(1)\left\{\lambda^{(3)}(1)\right\}^T . \qquad (25)
\end{aligned}
$$

Note that $(J^{(1)} + J^{(2)})$ has zeros on the last row and the last column and that $J^{(3)}$ has a single non zero element located on the last row and the last column. Hence the maximum likelihood estimator of the intercept μ will be asymptotically independent on the estimators of the other parameters. Let us introduce the matrices α and β with the polynomials, respectively, on the numerator and denominator of the first

two columns of the matrix in (23):

$$(26) \qquad \alpha = \begin{bmatrix} b_0bBdDhH & 1 \\ -b_0BdDhH & 0 \\ 0 & -1 \\ 0 & 1 \\ b_0bBdDhH & 0 \\ -b_0bBdDhH & -1 \\ b_0bBdDhH & 1 \\ -b_0bdDhH & 0 \\ 0 & -1 \\ 0 & 1 \\ b_0bBdDhH & 0 \\ -b_0bBdDhH & -1 \\ -bBdDhH & 0 \end{bmatrix}, \qquad \beta = \begin{bmatrix} acCeEgG & a \\ cCeEgG & 1 \\ 1 & c \\ 1 & d \\ cCe^2EgG & 1 \\ cCeEfgG & f \\ AcCeEgG & A \\ cCeEgG & 1 \\ 1 & C \\ 1 & D \\ cCeE^2gG & 1 \\ cCeEFgG & F \\ cCeEgG & 1 \end{bmatrix}.$$

For the computation of the expectations in (25), the traditional way (Åström, 1970) was to evaluate a circular integral. Motivated by the comparisons of Klein and Mélard (1989),we have prefered to follow the idea suggested by Pham(1989) to rely on the computation of a covariance of two autoregressive processes built on the same white noise process. Klein and Mélard (1990) have used the Tunnicliffe Wilson (1979) algorithm. Here, we refer to the Euclid algorithm of Demeure and Mullis (1989) as

follows.

More precisely, denoting $v_1(t) = v(t)$ and $v_2(t) = u(t)$, for the element (i, j) of the block (r, k) of the m-th term of (25), $m = 1, 2$, $r, k = 1, ..., 13$, $i = 1, ..., p_r$, $j = 1, ..., p_k$, we have to compute the expectation

$$\begin{aligned}(N\sigma^2)^{-1}(J_{rk})_{ij}^{(m)} &= \mathbb{E}[\{\lambda_{ir}^{(m)}(L)v_m(t)\}\{\lambda_{jk}^{(m)}(L)v_m(t)\}] \\ &= \mathbb{E}\left[\left\{\frac{L^{is_r}\alpha_r^{(m)}(L)}{\beta_r^{(m)}(L)}v_m(t)\right\}\left\{\frac{L^{js_k}\alpha_k^{(m)}(L)}{\beta_k^{(m)}(L)}v_m(t)\right\}\right], \qquad (27)\end{aligned}$$

where $s_r = 1$ if $r \leq 6$ or $r = 13$, and $s_r = s$ if $r > 6$, taking the chosen ordering of the nonseasonal and seasonal polynomials. This is the covariance between two ARMA processes $\{z_{ir}^{(m)}(t)\}$ and $\{z_{jk}^{(m)}(t)\}$ built using the same white noise process $\{v_m(t)\}$ and defined by

$$\beta_r^{(m)}(L)z_{ir}^{(m)}(t) = L^{is_r}\alpha_r^{(m)}(L)v_m(t), \qquad \beta_k^{(m)}(L)z_{jk}^{(m)}(t) = L^{js_k}\alpha_k^{(m)}(L)v_m(t).$$

The Åström circular integral approach consists in computing (27) by

$$\frac{\sigma_m^2}{2\pi i}\oint_\gamma z^{-is_r+js_k}\frac{\alpha_r^{(m)}(z^{-1})\alpha_k^{(m)}(z)}{\beta_r^{(m)}(z^{-1})\beta_k^{(m)}(z)}\frac{dz}{z},$$

where σ_m^2 is the variance of $\{v_m(t)\}$, and γ is the positively oriented unit circle.

Given that the polynomials in α and β are often products of similar polynomials among the $\theta_r(L)$ or $\theta_r(L^s)$, and in order to avoid unnecessary computations, let $\beta_{rk}^{*(m)}(L) = \beta_{kr}^{*(m)}(L)$ be the least common multiple (LCM) of the polynomials $\beta_r^{(m)}(L)$ and $\beta_k^{(m)}(L)$, a polynomial of degree $\delta_{rk}^{*(m)}$, say. Define the polynomials $\beta_{r(k)}^{*(m)}(L) =$

$\beta^{*(m)}_{rk}(L)/\beta^{(m)}_{r}(L)$, and $\beta^{*(m)}_{r(k)}(L) = \beta^{*(m)}_{rk}(L)/\beta^{(m)}_{k}(L)$, and let the polynomials

$$\alpha^{*(m)}_{r(k)}(L) = \alpha^{(m)}_{r}(L)\beta^{*(m)}_{r(k)}(L), \qquad \alpha^{*(m)}_{(r)k}(L) = \alpha^{(m)}_{k}(L)\beta^{*(m)}_{(r)k}(L),$$

of respective degrees $\delta^{*(m)}_{r(k)}$ and $\delta^{*(m)}_{(r)k}$. We shall use the coefficients of these polynomials defined by

$$\alpha^{*(m)}_{r(k)}(L) = \sum_{l=0}^{\delta^{*(m)}_{r(k)}} \alpha^{*(m)}_{r(k),l} L^l, \qquad \alpha^{*(m)}_{(r)k}(L) = \sum_{l=0}^{\delta^{*(m)}_{k(r)}} \alpha^{*(m)}_{(r)k,l} L^l.$$

Then, let us consider an autoregressive process $\{z^{*(m)}_{rk}(t)\}$ (or $\{z^{*(m)}_{kr}(t)\}$) of order $\delta^{*(m)}_{rk}$ built using $\{v_m(t)\}$ again and satisfying

$$\beta^{*(m)}_{rk}(L) z^{*(m)}_{rk}(t) = v_m(t), \tag{28}$$

and denote $R^{*(m)}_{rk}(l)$ its lag l autocovariance. Then

$$z^{(m)}_{ir}(t) = L^{is_r}\alpha^{*(m)}_{r(k)}(L) z^{*(m)}_{rk}(t), \qquad z^{(m)}_{jk}(t) = L^{js_k}\alpha^{*(m)}_{k(r)}(L) z^{*(m)}_{rk}(t),$$

and (27) can be written

$$
\begin{aligned}
(N\sigma^2)^{-1}(J_{rk})_{ij}^{(m)} &= \mathbb{E}\left[\left\{L^{is_r}\alpha_{r(k)}^{*(m)}(L)z_{rk}^{*(m)}(t)\right\}\left\{L^{js_k}\alpha_{(r)k}^{*(m)}(L)z_{rk}^{*(m)}(t)\right\}\right] \\
&= \mathbb{E}\left[\left\{\sum_{l=0}^{\delta_{r(k)}^{*(m)}}\alpha_{r(k),l}^{*(m)}z_{rk}^{*(m)}(t-l-is_r)\right\}\left\{\sum_{l=0}^{\delta_{(r)k}^{*(m)}}\alpha_{(r)k,l}^{*(m)}z_{rk}^{*(m)}(t-l-js_k)\right\}\right] \\
&= \sum_{l=0}^{\delta_{r(k)}^{*(m)}}\sum_{q=0}^{\delta_{(r)k}^{*(m)}}\alpha_{r(k),l}^{*(m)}\alpha_{(r)k,q}^{*(m)}\mathbb{E}\left[z_{rk}^{*(m)}(t-l-is_r)z_{rk}^{*(m)}(t-q-js_k)\right] \\
&= \sum_{l=0}^{\delta_{r(k)}^{*(m)}}\sum_{q=0}^{\delta_{(r)k}^{*(m)}}\alpha_{r(k),l}^{*(m)}\alpha_{(r)k,q}^{*(m)}R_{rk}^{*(m)}(l+is_r-q-js_k) \\
&= \sum_{p=-\delta_{(r)k}^{*(m)}}^{\delta_{r(k)}^{*(m)}}\left(\sum_{q=\max(0,-p)}^{\min(\delta_{(r)k}^{*(m)},\delta_{r(k)}^{*(m)}-p)}\alpha_{r(k),p+q}^{*(m)}\alpha_{(r)k,q}^{*(m)}\right)R_{rk}^{*(m)}(p+is_r-js_k) \\
&= \sum_{p=-\delta_{(r)k}^{*(m)}}^{\delta_{r(k)}^{*(m)}}\varepsilon_{rkp}^{(m)}R_{rk}^{*(m)}(p+is_r-js_k), \qquad (29)
\end{aligned}
$$

where

$$
\varepsilon_{rkp}^{(m)} = \sum_{q=\max(0,-p)}^{\min(\delta_{(r)k}^{*(m)},\delta_{r(k)}^{*(m)}-p)}\alpha_{r(k),p+q}^{*(m)}\alpha_{(r)k,q}^{*(m)}. \qquad (30)
$$

A large part of the procedure described above still holds in the case of multiple input variables, the MISO (multiple input single output) model, with q variables, say. The notation should be modified by using a superscript $l = 1, ..., q$ to $x(t)$, $v(t)$, b_0 and polynomials $b(L)$, $B(L)$, $e(L)$, $E(L)$. Then (24) can easily be adapted to reflect that there are more than one input variable. Let us assume that $(x^{(1)}, ..., x^{(q)})'$ is a VARMA process with autoregressive matrix polynomial $g(L)$ and moving average

matrix polynomial $h(L)$, and innovations $(u^{(1)}, ..., u^{(q)})$ with covariance matrix Σ. If the variables $x^{(1)}, ..., x^{(q)}$ are uncorrelated, then the $q \times q$ matrices $g(L)$, $h(L)$ and Σ are diagonal and the method described above can be used. Otherwise, the blocks of the information matrix which correpond to pairs $(x^{(l_1)}, x^{(l_2)})$ with $l_1 \neq l_2$ should be computed using an algorithm for evaluating the autocovariance function of a VARMA process, e. g. Ansley(1980), Kohn and Ansley(1982). In the following algorithm only SISO models are treated.

4 Numerical examples

4.1 Example 1

The univariate model for the machine-tool shipments (U.S., from January 1968 to December 1974) is taken from Pankratz (1983, p. 502):

$$\nabla\nabla_{12}y(t) = (1 - c_1L - c_2L^2 - c_6L^6)(1 - C_1L^{12})u(t),$$

using the Box-Jenkins notation, and where ∇ and ∇_{12} denote the regular differencing and the seasonal differencing operators, respectively. Note that the regular moving average polynomial is lacunary, which is easily handled by our program. Using the estimates provided, we obtain the following table, where the last column is ours.

Table 4.1. Comparison between the standard errors obtained using the algorithm of the paper and those obtained as a by-product of an optimisation procedure.

		standard error	
parameter	estimate	optimisation	algorithm
c_1	0.812	0.104	0.105
c_2	-0.224	0.103	0.091
c_6	-0.401	0.075	0.059
C_1	-0.808	0.107	0.071

4.2 Example 2

Our final univariate example is based on Enders (1995, Model 3, p. 115) and will emphasize performance results. Three of the univariate models for tourism in Spain (from January 1970 to March 1989, 218 observations) were estimated as follows:

$$\nabla\nabla_{12}y(t) = \left(1 - 0.640L - 0.306L^{12}\right) u(t), \tag{31}$$

$$\nabla\nabla_{12}y(t) = (1 - 0.740L)\left(1 - 0.671L^{12}\right) u(t), \tag{32}$$

$$\nabla\nabla_{12}y(t) = \left[1 - 0.640L + \varepsilon(L^2 + ... + L^{11}) - 0.306L^{12}\right] u(t), \tag{33}$$

where ε is a small number so that the corresponding parameter is different from zero but the model is nearly equivalent to the first one. The latter was not considered by Enders but will be used to show the performance of the algorithm for dealing with a 12×12 information matrix. All the computing time evaluations were done on a computer with a 133 MHz Pentium processor, as a mean over 10000 replications. In

each case, we have computed the variance by dividing the sum of squares of residuals SSR by the number of degrees of freedom, for example, for the first model 3.367 is divided by 216.

Table 4.2. Computation times required for computing the information matrix and the covariance matrix (see text for details)

	Computing time	
Model	information matrix	covariance matrix
(31)	0.109	0.005
(32)	0.249	0.005
(33)	0.164	0.312

4.3 Example 3

This is the open loop model of Gevers and Ljung (1986). It is defined by the equation

$$(34) \qquad (1 - a_1 L - a_2 L^2) y(t) = (b_0' - b_1' L) x(t) + (1 - c_1 L - c_2 L^2) u(t),$$

where $x(t)$ is a Gaussian white noise. The value of the parameters are $a_1 = 1.5$, $a_2 = -0.7$, $b_0' = 1$, $b_1' = -0.5$, $c_1 = 1$, $c_2 = -0.2$, $\sigma_1 = 1$, and $\sigma_2 = 10$ or 1. Note that the alternative polynomial expansion (a) $b'(L)B(L^s)$, discussed in the Appendix, is used. The simulation results of Gevers and Ljung (1986) performed over 10 runs of 500 data each have shown that the accuracy is good when the signal to noise ratio is

equal to 1, but is worse when the noise dominates the signal, except for the moving average coefficients which do not seem to be influenced.

Table 4.3. Comparison between the standard errors (SE) of the estimated values obtained by Gevers and Ljung, over 10 runs of 500 data each, with the asymptotic values.

		Parameters					
σ_2	Method	a_1	a_2	b'_0	b'_1	c_1	c_2
10	Gevers-Ljung	0.001		0.005		0.035	
	asymptotic SE	0.0044	0.0037	0.0140	0.0186	0.0441	0.0440
1	Gevers-Ljung	0.009		0.058		0.031	
	asymptotic SE	0.0136	0.0114	0.0443	0.0584	0.0463	0.0453

4.4 Example 4

Solbrand *et al.* (1985) have considered simulations for another simple ARMAX models of the form (34), where $a_1 = -0.9$, $a_2 = -0.95$, $b'_0 = 1$, $b'_1 = 0$, $c_1 = -1.5$, $c_2 = -0.75$, and have compared several on line recursive methods with an off-line maximum likelihood method. Using $N = 100$, 200 and 500, like the authors, we obtain the following standard errors.

Table 4.4. Comparison between the standard errors obtained using the algorithm of the paper for several lengths.

N	a_1	a_2	b'_0	b'_1	c_1	c_2
100	0.02633	0.02596	0.13496	0.13748	0.06799	0.06794
200	0.01862	0.01836	0.09543	0.09721	0.04808	0.04804
500	0.01177	0.01161	0.06036	0.06148	0.03041	0.03039

4.5 Example 5

Box *et al.* (1994, p. 410) have analysed the differences of the sales data ($y(t)$) in terms of the differences of a leading indicator ($x(t)$) using the following model

$$\begin{aligned} y(t) &= 0.035 + 4.82\frac{1}{1-0.72L}x(t-3) + (1-0.54L)u(t), \\ x(t) &= (1-0.32L)v(t). \end{aligned}$$

fitted on their data (Series M). Since the standard errors of the estimates were not provided (and also some ambiguity on the sample really used, see the footnote of p. 410, confirmed by the fact that the obtained estimates are different), we have fitted the model using SCA (Liu and Hudak, 1994). The estimates are as follows, in our ordering, $c_1 = -0.6284$, $e_1 = -0.7256$, $b_0 = 4.7024$, $\mu = 0.0341$, and $\sigma^2 = 0.046468$. The information matrix is therefore of dimension 4. After reestimation we have also $h_1 = -0.440$, and $\tau^2 = 0.080962$.

Here are the vector of the standard errors and the covariance matrix of the estimators obtained by inverting the Fisher information matrix computed using our

method.

$$\begin{bmatrix} 0.0635 \\ 0.0034 \\ 0.0459 \\ 0.0065 \end{bmatrix}, \quad \begin{bmatrix} 1.00 & & & \\ 0.00 & 1.00 & & \\ 0.00 & -0.69 & 1.00 & \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$

and here are the equivalent values obtained using the optimisation algorithm in SCA:

$$\begin{bmatrix} 0.0658 \\ 0.0034 \\ 0.0496 \\ 0.0078 \end{bmatrix}, \quad \begin{bmatrix} 1.00 & & & \\ 0.04 & 1.00 & & \\ -0.07 & -0.74 & 1.00 & \\ 0.01 & -0.28 & -0.04 & 1.00 \end{bmatrix}.$$

The results are very similar. We have fitted the model using Autobox 4.0 (Automatic Forecasting Systems, 1998). Their estimates are $c_1 = -0.588$, $e_1 = -0.725$, $b_0 = 4.71$, $\mu = 0.0107$, and $\sigma^2 = 0.047471$. The vector of the standard errors and the covariance matrix of the estimators obtained by inverting the Fisher information matrix computed using our method are

$$\begin{bmatrix} 0.06604 \\ 0.00364 \\ 0.04807 \\ 0.00733 \end{bmatrix}, \quad \begin{bmatrix} 1.00 & & & \\ 0.00 & 1.00 & & \\ 0.00 & -0.67 & 1.00 & \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$

whereas equivalent values obtained using the optimisation algorithm in AFS Autobox

are

$$\begin{bmatrix} 0.06600 \\ 0.00368 \\ 0.05051 \\ 0.00235 \end{bmatrix}, \quad \begin{bmatrix} 1.00 & & & \\ 0.00 & 1.00 & & \\ -0.04 & 0.00 & 1.00 & \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}.$$

The results are not longer similar because two problems arise: the constant and its standard error are different, and the large correlation between the estimates of e_1 and b_0 doesn't appear. We suspect a program error in Autobox 4.0 that was confirmed. We have heard that it was resolved in the current release of FreeFore and Autobox 5.0 but we were unable to check it by ourselves.

4.6 Example 6

De Gooijer and Klein (1989) have analysed the relation between the incoming and the outgoing maritime steel traffic at the port of Antwerp. In a related work, Klein (1986, p. 6.15-6.16) has used the following transfer function model for explaining the incoming traffic, between January 1971 and March 1981. Let $y(t)$ be the difference of the 0.325 power of the incoming steel traffic and $x(t)$ the difference of natural logarithm of the outgoing maritime steel traffic. The model used was

$$y(t) = b_0 \frac{1}{1 - e_1 L} x(t-1) + \frac{1}{(1 - d_1 L - d_2 L^2 - d_3 L^3)(1 - D_1 L^{12})} u(t)$$

The vector of parameters is $(d_1, d_2, d_3, e_1, D_1, b_0)'$. The standard deviation of $u(t)$ is estimated as 3.806. The model for the input $x(t)$ (Klein, 1986, p. D2) was as follows

$$(1 - g_1L - g_2L^2 - g_3L^3)x(t) = (1 - h_1L)v(t),$$

with the following estimates $g_1 = -1.09$, $g_2 = -0.74$, $g_3 = -0.48$, $h_1 = -0.54$, and $\tau = 0.235$ (an intervention had been applied in February 1975, but it will be ignored here).

The following table shows the standard errors obtained (i) during the course of parameter estimation using an optimisation algorithm and a least squares criterion, (ii) using the algorithm described above.

Table 4.5. Comparison between the standard errors obtained using the algorithm of the paper and those obtained as a by-product of an optimisation procedure.

		standard error	
parameter	estimate	optimisation	algorithm
d_1	-0.78	0.09	0.08
d_2	-0.48	0.11	0.10
d_3	-0.34	0.09	0.08
e_1	0.91	0.05	0.04
D_1	0.39	0.10	0.08
b_0	-2.16	0.69	0.64

It is clear that the algorithm produces adequate results.

4.7 Example 7

Our next example is a transfer function model for housing starts as a function of housing sales (U.S., from January 1965 to December 1975). It is based on Pankratz (1991, p. 518). This time we needed to change the specifications for two reasons. First, the model fitted by Pankratz for the input variable is not invertible (the seasonal moving average has all its roots inside the unit circle). Second, the transfer function model relates the series in seasonal differences whereas the univariate model for the housing sales also makes use of a regular difference in order to obtain stationarity. This is not acceptable in our approach which is unconditional with respect to the observations of the input variable. We have fitted an alternative model described by

$$\nabla_{12} x(t) = \frac{1 - 0.99963 L^{12}}{1 - 0.93338 L} v(t),$$

with 11.49786 as variance of the input innovation. The transfer function model is defined by equation (C4.3.4) in Pankratz (1991), more precisely

$$\nabla_{12} y(t) = (b'_0 + b'_1 L)\, \nabla_{12} x(t) + \frac{(1 + c_1 L)(1 + C_1 L^{12})}{1 + d_1 L} u(t),$$

and the results are presented in the following table, this time with t Student statistics instead of standard errors. Note that we have used the engineering notation, contrarily to Pankratz.

Table 4.6. Comparison between the Student t-statistics obtained using the algorithm of the paper and those obtained as a by-product of an optimisation procedure.

		|Student t statistic|	
parameter	estimate	optimisation	algorithm
b'_0	0.6639	4.59	4.65
b'_1	0.8720	6.02	6.10
c_1	-0.6208	5.32	5.31
d_1	-0.9046	14.26	14.22
C_1	-0.9540	15.94	32.93

The disagreement for C_1 is not surprising since we are close to the invertibility borderline. Because Pankratz (1991) has also given the asymptotic correlation matrix,

$$\begin{bmatrix} 1.00 & & & & \\ -0.59 & 1.00 & & & \\ 0.00 & 0.00 & 1.00 & & \\ 0.00 & 0.00 & 0.74 & 1.00 & \\ 0.00 & 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}$$

we provide our own estimate

$$\begin{bmatrix} 1.00 & & & & \\ -0.66 & 1.00 & & & \\ 0.00 & 0.00 & 1.00 & & \\ 0.00 & 0.00 & 0.76 & 1.00 & \\ 0.00 & 0.00 & 0.07 & 0.09 & 1.00 \end{bmatrix}$$

which is very close indeed.

Appendix : the algorithm

The algorithm for computing the (asymptotic) Fisher information matrix of the parameters of the 12-polynomial seasonal SISO model is described now. The Fortran 90 code (tested using the free Essential Lahey Fortran 90 compiler) can be obtained at the site mentioned with the author's addresses, as well as detailed instructions. We mention here the additional capabilities and some aspects of program organization.

First, the program has been organised with several applications in mind. The most demanding application is probably recursive estimation using the Zahaf (1998) method, currently implemented only for ARMA models. That method is an improvement with respect to the classic Recursive Maximum Likelihood (RML) method. Let us denote $y^t = \{y(1), \ldots, y(t)\}$, the time series available at time t. Suppose that it can be represented by an ARMA(p_3, p_4) defined by (1), by taking all other p_r's equal to zero. Then the recursive estimator $\widehat{\theta}(t)$ at time t, of the vector of parameters θ, is a function of y^t defined by

$$\begin{aligned}
\widehat{u}(t) &= y(t) - \widehat{\theta}^T(t-1)\varphi(t), \\
\widehat{\theta}(t) &= \widehat{\theta}(t-1) + \overline{R}^{-1}(t)\psi(t)\widehat{u}(t), \\
\psi(t+1) &= \sum_{k=1}^{p_4} \hat{\theta}_{p_3+k}(t)\psi(t-k+1) + \varphi(t+1), \\
\overline{R}(t+1) &= \overline{R}(t) + \psi(t+1)\psi^T(t+1),
\end{aligned}$$

where $\varphi(t) = (y(t), \cdots, y(t-p_3+1), -\widehat{u}(t), \cdots, -\widehat{u}(t-p_4+1))$, $\psi(t)$ estimates $-u_\theta(t)$ and $\sigma^2 \overline{R}^{-1}(t)$ represents an approximation of the asymptotic covariance matrix of the estimator. For more details, see e.g. Ljung and Söderström (1983). The improvement introduced by Zahaf (1998) consists in replacing $\overline{R}^{-1}(t)$ by the inverse of the Gaussian Fisher information matrix $J^{-1}(\widehat{\theta}(t-1))$ evaluated at $\theta = \widehat{\theta}(t-1)$, so that line 2 of the recursion is replaced by:

$$\widehat{\theta}(t) = \widehat{\theta}(t-1) + \widehat{\sigma}_t^2 \, J^{-1}(\widehat{\theta}(t-1))\psi(t)\widehat{u}(t),$$

where $\widehat{\sigma}_t^2$ is the updated estimate of the innovation variance. In this case, the algorithm described here is needed at each time t. Furthermore, it is often useful to run several models in parallel. We have taken care of that fact in the computer program `GFIMS_SISO`. Four main subroutines are delivered. All of them are written for running several models. One, `GFIMS_INIT_READ` performs data structure initialisations after having read the parameter set for a given model. Another one, `GFIMS_INIT_STOR`, stores a parameter set into the data structures. `GFIMS_ADJ` adjusts the size of the working storage needed for the computations. Finally, `GFIMS_COMP` performs the computations. Note that `GFIMS_INIT_READ` and `GFIMS_INIT_STOR` should be called once for each model at the beginning. In the generalisation of the Zahaf (1998) method, `GFIMS_ADJ` and `GFIMS_COMP` should be called at each time t, and for each model. If timing of the computations is requested, only `GFIMS_ADJ` and `GFIMS_COMP` needs to be replicated a large number of time. In a simpler context, the four subprograms are

simply called one after the other, in the given order.

For each model, the input information consists in the seasonal period s (which can be equal to 1), the number of observations, the model orders (the degrees of the 16 polynomials, including the 4 polynomials of the input process), the value of the parameters (the coefficients of these 16 polynomials plus the value of the regression constant b_0 and that of μ (although it is not used in the computations), and the variances σ^2 and τ^2. ARMA models are covered as a special case where $\tau^2 = 0$. Some coefficients can be omitted in the polynomials. For example, let us consider the polynomial $1 - 0.6L - 0.8L^4 + 0.48L^5$, where the coefficients of L^2 and L^3 are omitted. Their value is considered to be 0 and they are not considered as parameters. This leaves the coefficients of L, L^4, and L^5 as parameters, whose values are -0.6, -0.8, and 0.48, respectively. In that example, as well as in the preceding sections, the usual notation in engineering has been used for polynomial coefficients. However, the altenative notation often used by statisticians and econometricians is also available in the program so that the value of the parameters can be 0.6, 0.8, and -0.48, respectively.

The parametrization chosen for the numerator of the input part of the model is $b_0 b(L) B(L^s)$, where $b(L) = (1 + b_1 L + ... + b_{p_2} L^{p_2})$ and $B(L^s) = (1 + B_1 L^s + ... + B_{p_8} L^{sp_8})$ have 1 as a constant term. This is not the mostly used notation but we have preferred it because it gives symmetric roles to $b(L)$ and $B(L^s)$. Two other notations are

however available in the program, using the alternative polynomial expansions as follows: (a) $b'(L)B(L^s)$ where $b'(L) = b'_0 + b'_1 L + ... + b'_{p_2} L^{p_2}$; (b) $b(L)B'(L)$ where $B'(L) = B'_0 + B'_1 L^s + ... + B'_{p_8} L^{sp_8}$. Of course, b_0 cannot be used as a parameter when theses alternative notations are used and its value should be taken as 1. Let us consider the changes for case (a). They are limited to the use of $\widetilde{b}'(L) = (1, L, ..., L^{p_2})^T$ instead of $\widetilde{b}(L) = (L, ..., L^{p_2})^T$. Note that p_2+1 is the number of parameters associated with $b'(L)$, which equals the number of parameters associated with $b_0 b(L)$. The other case (b) can be treated in a similar way.

The instructions which allow timing of the algorithm are provided. We don't discuss this in the following description. Let us just mention that a number of repetitions of each computation can be set so that timings are evaluated as averages over a loop, in order that times be accurate enough.

The organisational aspects are worth to be mentioned. Indeed, straightforward application of the formulae (29) and (30) would result in lengthy code prone to errors. It suffices to mention that, for the computation of some elements of the information matrix, products of up to 12 polynomials are involved. We have adopted the following strategy, already sketched by Klein and Mélard (1994a) but detailed here and suitably generalized for the case of seasonal models. The coefficients of all the polynomials are entered in a long vector. For each polynomial, a pointer to the first coefficient is maintained as well as the degree and an indication for the power (1 for 1, for a regular

polynomial, 2 for s, for a seasonal polynomial). Strings of characters specifying the elements of α and β are hard-coded in the program. For instance, the element $(12, 1)$ of α which is $-b_0 bBdDhH$ is represented by "-rbBdDhH". The only notational change with respect to (26) is that the regression constant b_0 is represented by "r". Note that "+1", "-1" and "0" are represented as such. Note that only the upper left 13×13 blocks are considered now.

The algorithm can then be written as follows.

J $\leftarrow$ 0

For $r = 1$ to 13

For $k = 1$ to 13

If $p_r p_k > 0$ then

If $r \leq k$ then

For $m = 1$ to 2

[LCM, $R1$, $R2$] $\leftarrow$REDPSI($\alpha_r^{(m)}(L)$, $\alpha_k^{(m)}(L)$, $\beta_r^{(m)}(L)$, $\beta_k^{(m)}(L)$)

If $R1 \neq 0$ or $R2 \neq 0$ then

$\beta_{rk}^{*(m)}(L) \leftarrow$EVAPSI($LCM$)

$\alpha_{r(k)}^{*(m)}(L) \leftarrow$EVAPSI($R1$)

$\alpha_{(r)k}^{*(m)}(L) \leftarrow$EVAPSI($R2$)

$R_{rk}^{*(m)}(l) \leftarrow$DMACF($\beta_{rk}^{*(m)}(L)$)

$J_{rk} \leftarrow J_{rk}+$FIBKSI($\alpha_{r(k)}^{*(m)}(L)$, $\alpha_{(r)k}^{*(m)}(L)$, $R_{rk}^{*(m)}(l)$)

```
                endif
            endfor
        else
        J_rk ← J_kr
        endif
    endif
endfor
endfor
```

Here are some explanations. The information matrix is initially set to 0. Each block of the information matrix is computed separately, except by taking care that the matrix is symmetric and that some blocks can be empty because of omitted polynomials in the model. It is also taken into account that some blocks are identically equal to zero.

In the algorithm, `REDPSI` determines the factors $\theta_h(L)$, $h = 1, ..., 17$, of $\beta_r^{(m)}(L)$ and $\beta_k^{(m)}(L)$ in order to determine the lowest common multiple (LCM) polynomial $\beta_{rk}^{*(m)}(L) = \beta_{kr}^{*(m)}(L)$ of $\beta_r^{(m)}(L)$ and $\beta_k^{(m)}(L)$. This is only valid under the assumption that all the roots of all the polynomials $\theta_h(L)$ are distinct, an assumption that has been done. This avoids us to consider the possible factorisation of these polynomials. The factors are represented in a 1×18 vector of positive integers, with the sign S (0, -1 or $+1$) as first entry, and the power of the factor $\theta_h(L)$, $h = 1, ..., 17$,

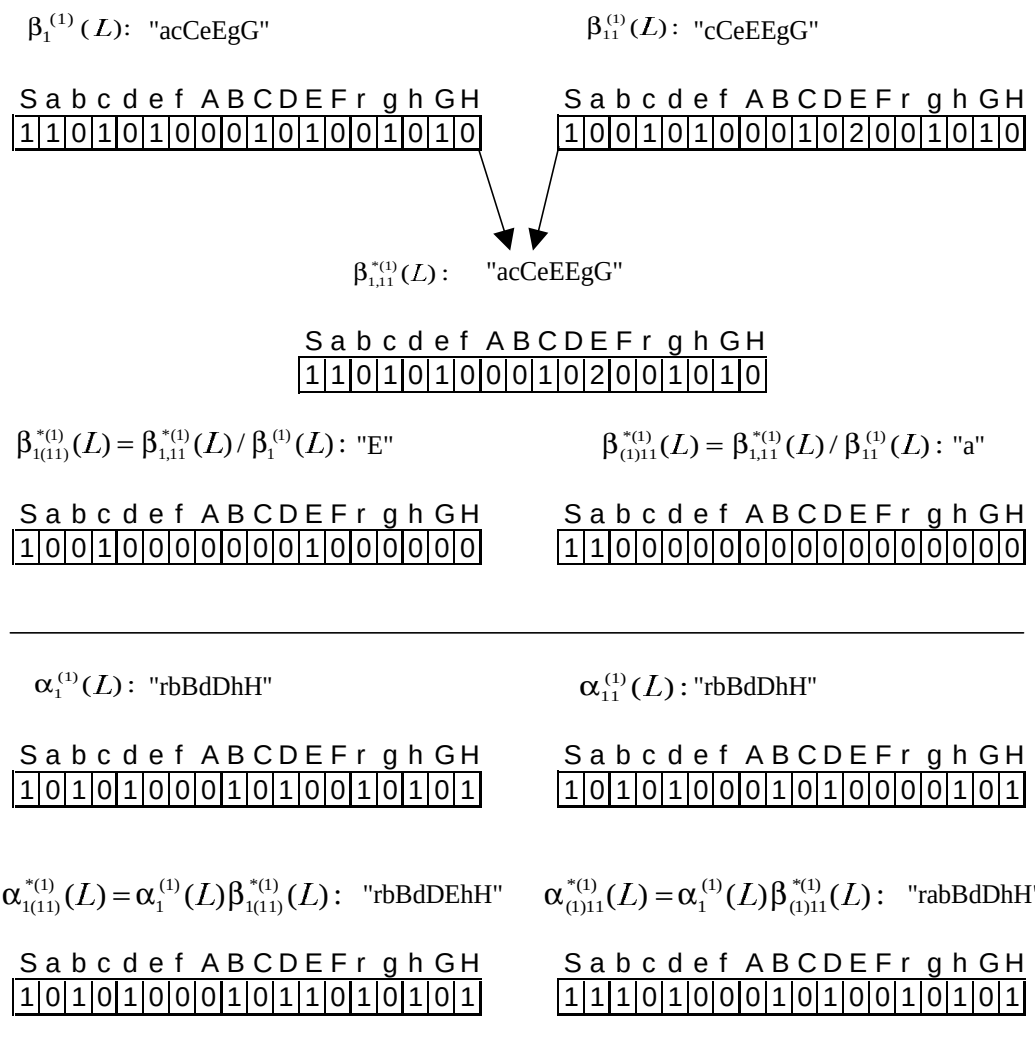
according to the order $(S; a, b, c, d, e, f; A, B, C, D, E, F; r; g, h; G, H)$. For example, for $m = 1$, $r = 1$ and $k = 11$, the polynomials are $\beta_1^{(1)}(L) = acCeEgG$, hence "acCeEgG", and $\beta_{11}^{(1)}(L) = cCeE^2gG$ or "cCeEEgG", so that the LCM $\beta_{1,11}^{*(1)}(L) = acCeE^2gG$ is "acCeEEgG". Figure 1 shows how that example is handled in terms of vector representations. For example, the vector representation of the LCM is (1;1,0,1,0,1,0;0,0,1,0,2,0;0;1,0;1,0). Hence $\beta_{1(11)}^{*(1)}(L)$: "E" and $\beta_{(1)11}^{*(1)}$: "a". `REDPSI` also determines the polynomials $\alpha_{r(k)}^{*(m)}(L)$ and $\alpha_{(r)k}^{*(m)}(L)$ using the same vector representation. In the same example as before, we have $\alpha_1^{(1)}(L) = \alpha_{11}^{(1)}(L)$ are "rbBdDhH" so that $\alpha_{1(11)}^{*(1)}(L)$ and $\alpha_{(1)11}^{*(1)}(L)$ are respectively "rbBdDEhH" and "rabBdDhH" and are therefore represented by the vectors (1;0,1,0,1,0,0;0,1,0,1,1,0;1;0,1;0,1) and (1;1,1,0,1,0,0;0,1,0,1,0,0;1;0,1;0,1).

Three calls of procedure `EVAPSI` are used to evaluate numerically the coefficients of the polynomials $\beta_{rk}^{*(m)}(L)$, $\alpha_{r(k)}^{*(m)}(L)$ and $\alpha_{(r)k}^{*(m)}(L)$. This is done using a general purpose product of polynomials routine `MLTPOLS` which, given n polynomials $\theta_h(L)$, $h = 1, ..., n$, and a $1 \times (n+1)$ vector $(i_0, i_1, ..., i_n)$ of integers (positive except the first entry which can be 0, -1 or $+1$), evaluates the coefficients of

$$i_0 \prod_{h=1}^{n} [\theta_h(L)]^{i_h} .$$

More precisely, this is done in two steps in order to take advantage of the fact that most coefficients of the seasonal polynomials (e.g. A, B, ...) are equal to zero. First the nonseasonal factors are multiplied together giving a polynomial $p(L)$ in L and

Figure 1: Diagram showing how the polynomials are handled.



the seasonal factors are multiplied together giving a polynomial $P(L^s)$ in L^s. Then, $P(L^s)$ is converted into a lacunary polynomial in L, which is multiplied by $p(L)$. Only the last operation possibly involves a high degree polynomial.

`DMACF` makes use of the Demeure and Mullis (1989, 1990) algorithm for computing the lag-l autocovariances $R_{rk}^{*(m)}(l)$, $l = 0, ..., \max(\delta_{r(k)}^{*(m)} + p_r s_r - s_k, \delta_{(r)k}^{*(m)} + p_k s_k - s_r)$ of the autoregressive process of order $\delta_{rk}^{*(m)}$ defined by (28). This is the most computationally-intensive part of the algorithm since the number of operations is $O(5\left[\delta_{rk}^{*(m)}\right]^2/2)$. Of course, we could have implemented these computations in another way by avoiding LCM of pairs of polynomials, and considering instead, for each m, an autoregressive polynomial being the LCM of all the $\beta_r^{(m)}(L)$, $r = 1, ..., 12$. In that case, `DMACF` would have been called once for each m. We have already tried that alternative computational approach in the pure ARMA seasonal case (see Klein and Mélard, 1990). The results were not favorable but that doesn't mean it cannot be better for some models. A finer evaluation of operation numbers should be needed to substanciate a conclusion.

Thereafter, each element $(J_{rk})_{ij}^{(m)}$ of $(J_{rk})^{(m)}$ is computed using (29). This is done in procedure `FIBKSI`. First the $\varepsilon_{rkp}^{(m)}$ defined by (30) are computed. Then the block $(J_{rk})^{(m)}$ is treated, in order to take care of the possible Toeplitz behaviour of that matrix. For two nonseasonal (respectively seasonal) polynomials, provided the two polynomials are not lacunary as polynomials in L (respectively L^s), the block is

Toeplitz. Otherwise it is not Toeplitz but that doesn't mean that every element needs to be computed. In fact, according to (29) all elements (i,j) of the (r,k) block for which $is_r - js_k$ is the same will have the same value for $(J_{rk})_{ij}^{(m)}$. From the algorithmic point of view, this is done using a lookup table with two entries, $is_r - js_k$ and $(J_{rk})_{ij}^{(m)}$ which is initialized, loaded and searched within `FIBKSI`. For each (i,j), whatever the element is computed or taken from the table, $(J_{rk})_{ij}^{(m)}$ is accumulated in order to form $(J_{rk})_{ij}$.

Acknowledgement

This paper has benefited from a grant "Action de Recherche Concertée" 96/01/205 of the "Communauté Française Wallonie-Bruxelles" and IAP-network in Statistics, contract P5/24, financially supported by the Belgian Federal Office for Scientific, Technical and Cultural Affairs (OSTC). The authors wish also to thank Abdessamad Saidi for his computational help for Example 5. We thank an anonymous referee for his/her valuable suggestions which improved the quality of the paper.

References

C. F. ANSLEY, Computation of the theoretical autocovariance function for a vector ARMA process, *J. Statistical Computation and Simulation*, **12** (1980) 15-24.

K. J. ÅSTRÖM, *Introduction to Stochastic Control,* Academic Press, New York (1970).

AUTOMATIC FORECASTING SYSTEMS Inc., *Autobox for Windows 4.0 User's Guide,* P.O. Box 563, Hatboro, PA 19040, U.S.A (1998).

P. J. BICKEL, C. A. J. KLAASSEN, Y. RITOV, and J. A. WELLNER, *Efficient and Adaptative Estimation for Semiparametric Models,* Springer, New York (1998) (2nd edition).

G. E. P. BOX, G. M. JENKINS, and G. C. REINSEL, *Time Series Analysis Forecasting and Control*, Prentice Hall Press, Englewood Cliffs, N.J. (1994) (3rd edition).

J. DE GOOIJER, and A. KLEIN, Forecasting the Antwerp Maritime Steel Traffic Flow: A case study. *J. Forecasting* **8** (1989) 381-398.

C. J. DEMEURE, and C. T. MULLIS, The Euclid algorithm and the fast computation of cross-covariance and autocovariance sequences. *IEEE Trans. Acoust., Speech, Signal Processing*. **37** (1989) 545-552.

C. J. DEMEURE, and C. T. MULLIS, A Newton-Raphson method for moving-average spectral factorization using the Euclid algorithm. *IEEE Trans. Acoust., Speech, Signal Processing.* **38** (1990) 1697-1709.

B. G. DHARAN, A priori sample size evaluation and information matrix computation for time series models. *J. Statist. Comput. Simul.* **21** (1985) 171-177.

F. C. DROST, C. A. J. KLAASSEN, and B. J. M. WERKER, Adaptative estimation in time series models, *Ann. Statist.* **25** (1997) 786-817.

W. ENDERS, *Applied Econometric Time Series,* Wiley, New York (1995).

B. FRIEDLANDER, On the computation of the Cramér-Rao bound for ARMA parameter estimation, *IEEE Trans. Acoust., Speech, Signal Processing* **32** (1984) 721-727.

M. GEVERS, and L. LJUNG, Optimal experiment designs with respect to the intended model application, *Automatica* **22** (1986) 543-544.

E. J. GODOLPHIN, and J. M. UNWIN, Evaluation of the covariance matrix for the maximum likelihood estimator of a Gaussian autoregressive-moving average process. *Biometrika* **70** (1983) 279-284.

M. HALLIN, and B. J. M. WERKER, Optimal testing for semi-parametric autoregressive models: from Gaussian Lagrange multipliers to regression rank scores and adaptive tests, in S. Ghosh, Ed., *Asymptotics, Nonparametrics and Time Series*, M. Dekker, New York (1998) pp. 295-358.

R. KOHN, and C. F. ANSLEY, A note on obtaining the theoretical autocovariances of an ARMA process, *J. Statist. Computation and Simulation*, **15** (1982) 273-283.

A. KLEIN, Quantitative forecasts of Antwerp's maritime traffic by means of time series analysis. Doctoral dissertation, Vrije Universiteit Brussel (1986).

A. KLEIN, and G. MELARD, On algorithms for computing the covariance matrix of estimates in autoregressive-moving average models. *Computational Statistics Quarterly*. **5** (1989) 1-9.

A. KLEIN, and G. MELARD, Fisher's information matrix for seasonal autoregressive-moving average models. *J. Time Series Analysis*. **11** (1990) 231-237.

A. KLEIN, and G. MELARD, Computation of the Fisher information matrix for SISO models. *IEEE Trans. Signal Processing* **42** (1994a) 684-688.

A. KLEIN, and G. MELARD, The information matrix of multiple input single output time series models. *J. Computational and Applied Mathematics* **51** (1994b) 349-356.

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

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

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

R. KOHN, and C. F. ANSLEY, Computing the likelihood and its derivatives for a Gaussian ARMA model. *J. Statist. Comput. Simul.* **22** (1985) 229-263.

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

L. M. LIU and G. B. HUDAK, *Forecasting and Time Series Analysis using the SCA Statistical System, Volume 1, ,* Scientific Computing Associates Corp. P.O. Box 4692, Oak Brook, IL 60522-4692, U.S.A (1994).

L. LJUNG, and T. SÖDERSTRÖM, *Theory and Practice of recursive identification,* M.I.T. Press, Cambridge MA (1983).

S. MITTNIK, and P. A. ZADROZNY, Asymptotic distributions of impulse responses, step responses, and variance decompositions of estimated linear dynamic model. *Econometrica.* **61** (1993) 857-870.

A. PANKRATZ, *Forecasting with Univariate Box-Jenkins Models,* Wiley, New York (1983).

A. PANKRATZ, *Forecasting with Dynamic Regression Models,* Wiley, New York (1991).

D. T. PHAM, Cramer-Rao bounds for AR parameter and reflection coefficient estimators. *IEEE Trans. Acoust. Speech Signal Processing*. **37** (1989) 769-772.

G. SOLBRAND, A. AHLEN, and L. LJUNG, Recursive methods for off-line identification, *Int. J. Control*, **41** (1985) 177-191.

G. T. TUNNICLIFFE WILSON, Some efficient computational procedures for high order ARMA models. *J. Statist. Comput. Simul.* **8** (1979) 303-309.

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

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

T. ZAHAF, Contribution à l'estimation des paramètres de modèles de séries chronologiques, Ph. D. Thesis, Institut de Statistique et de Recherche Opérationnelle, Université Libre de Bruxelles (1998).