View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Research Papers in Economics

Fast estimation methods for time series models in state-space form

Alfredo G. Hiernaux
José Casals
Miguel Jerez *

July 13, 2005

Abstract

We propose two fast, stable and consistent methods to estimate time series models expressed in their equivalent state-space form. They are useful both, to obtain adequate initial conditions for a maximum-likelihood iteration, or to provide final estimates when maximum-likelihood is considered inadequate or costly. The state-space foundation of these procedures implies that they can estimate any linear fixed-coefficients model, such as ARIMA, VARMAX or structural time series models. The computational and finite-sample performance of both methods is very good, as a simulation exercise shows.

Keywords: State-space models, subspace methods, Kalman Filter, system identification

*corresponding author. Departamento de Fundamentos del Anlisis Econmico II. Facultad de Ciencias Económicas. Campus de Somosaguas. 28223 Madrid (SPAIN). email: mjerez@ccee.ucm.es, tel: (+34) 91 394 23 61, fax: 91 394 25 91.

1 Introduction

Most statistical model estimation is based either on least squares (LS) or maximum likelihood (ML) methods. The LS approach has clear advantages in terms of computational simplicity, stability and cost. However it is limited in scope, as it cannot be applied to many relevant models such as those with moving average terms or multiplicative seasonality. On the other hand ML estimation most often requires iterative optimization, so its statistical efficiency and wide scope is somewhat balanced with increased instability, complexity and computational cost. Specifically when modeling high-frequency time series, such as those generated by financial markets, it may be hard to accept the underlying assumptions and costs implied by gaussian ML.

As the pros and cons of LS and ML are complementary, many works focus in devising methods that provide the best of both worlds. For example, Spliid (1983), Koreisha and Pukkila (1989, 1990), Flores and Serrano (2002) and Dufour and Pelletier (2004) extend ordinary and generalized LS to VARMAX modeling. Also Lütkepohl and Poskitt (1996) propose a LS-based method to specify and estimate a canonical VARMA structure. Finally, Francq and Zakoïan (2000) provide weak GARCH representations that can be estimated via two-stage LS. These methods avoid iteration when there are no exclusion constraints. Imposing such restrictions requires an iterative procedure which partially offsets their computational advantage. Also, they are typically devised for specific parameterizations, such as the VARMAX representations and their implementation in other frameworks is not trivial.

In this paper we propose two fast and consistent algorithms to estimate time series models written in their equivalent state-space (SS) form. They are useful either to obtain adequate initial conditions for ML iteration, or to provide final estimates when ML is considered inadequate or too expensive. Their use of the SS formulations makes them independent of particular representations, as they can be applied to any linear fixed-coefficients model with an equivalent SS form.

Both algorithms are based on subspace methods, see Van Overschee and De Moor (1996), Ljung (1999), or Favoreel et al. (2000). The first method, SUBEST1, consists of estimating the SS model matrices by solving a weighted reduced-rank LS problem with defined over a set of subspace regressions. Iteration is required when the problem has constraints relating the parameters in the standard and state-space representations of the model. Any exclusion or fixed-value constraint is treated in the same way. Montecarlo experiments show that this method is: a) asymptotically equivalent to ML, b) very fast and stable, and c) has a perceptible bias in comparison with ML when estimating models with moving-average parameters. Such bias was to be expected, as it has been reported several times by the literature, see, e.g., Ansley and Newbold (1980).

The second and more sophisticated procedure, SUBEST2, consists of a gaussian ML solution to the subspace regressions. Simulations show, as expected, that the method is more efficient than SUBEST1 and compensates effectively the shortcomings noted above. The cost of this improved behavior is a higher computational load that, however, remains much smaller than that of ML.

These results point to a clear segmentation of problems between both algorithms. When the model does not include moving average terms, SUBEST1 is more adequate. Also it is better to compute initial estimates for ML because of its speed. On the other hand, SUBEST2 is more efficient as final estimation procedure. Its performance in estimation is comparable to ML for large gaussian samples and practically identical for non-gaussian samples, while consistently maintaining a substantial stability and computational advantage.

The structure of the paper is as follows. Section 2 defines the basic notation and results. Building on these grounds, Section 3 derives the estimation algorithms and Sections 4 explores their performance in finite samples.

The algorithms described in this article are implemented in a MATLAB toolbox for time series modelling called E^4. The source code of this toolbox is freely provided under the terms of the GNU General Public License and can be downloaded at $www.ucm.es/info/icae/e4$. This site also includes a complete user manual and other materials.

2 Notation and basic results

2.1 State-space models

Consider a vector of endogenous outputs, $\boldsymbol{z_t}$, which is related to its past and to current and past values of a vector of exogenous inputs, $\boldsymbol{u_t}$, through a standard time series model depending on a vector of constant parameters, $\boldsymbol{\theta}$, to be estimated. Assume also that this model can be written in an equivalent SS representation.

Definition 2.1 *(State-space model):*

$$\begin{aligned} \boldsymbol{x_{t+1}} &= \boldsymbol{\Phi x_t} + \boldsymbol{\Gamma u_t} + \boldsymbol{E w_t} \\ \boldsymbol{z_t} &= \boldsymbol{H x_t} + \boldsymbol{D u_t} + \boldsymbol{C v_t} \end{aligned} \tag{1}$$

where $\boldsymbol{x_t} \in \mathbb{R}^n$ is the state vector, $\boldsymbol{u_t} \in \mathbb{R}^r$ is the vector of inputs and $\boldsymbol{z_t} \in \mathbb{R}^m$ is an observable output. On the other hand, $\boldsymbol{w_t} \in \mathbb{R}^n$ and $\boldsymbol{v_t} \in \mathbb{R}^m$ are uncorrelated sequences of errors such that: $E(\boldsymbol{w_t}) = \boldsymbol{0}, E(\boldsymbol{v_t}) = \boldsymbol{0}$, $E(\boldsymbol{w_t w_t'}) = \boldsymbol{Q}, E(\boldsymbol{v_t v_t'}) = \boldsymbol{R}$ and $E(\boldsymbol{v_t w_t'}) = \boldsymbol{S}$.

When $\boldsymbol{w_t} = \boldsymbol{v_t} = \boldsymbol{e_t}$ and $\boldsymbol{C} = \boldsymbol{I}$, (1) is known as an innovations form (Aoki, 1990).

Definition 2.2 *(Innovations model):*

$$\begin{aligned} \boldsymbol{x_{t+1}} &= \boldsymbol{\Phi x_t} + \boldsymbol{\Gamma u_t} + \boldsymbol{E e_t} \\ \boldsymbol{z_t} &= \boldsymbol{H x_t} + \boldsymbol{D u_t} + \boldsymbol{e_t} \end{aligned} \tag{2}$$

where $\boldsymbol{e_t} \in \mathbb{R}^m$ is a vector of errors independent of the initial state and such that $\boldsymbol{e_t} \sim iidN(0, \boldsymbol{Q})$. Many time series models, including transfer functions, VAR and VARMAX, can be directly written in the steady-state innovations form (Aoki, 1990; Terceiro, 1990). Also, under weak assumptions any model in the general form (1) can be written in the innovations form (2) (Casals et al., 1999, Theorem 1).

2.2 The subspace structure

Subspace identification starts from the transformation of a SS model, given by (1) or (2), into a single linear equation. This equation requires the following matrices related to the data:

Definition 2.3 *(Block-Hankel Matrix):* The Block-Hankel Matrix (BHM) is partitioned in two blocks called past (p) and future (f). Different choices for both parameters, p and f, are discussed by Viberg (1995), Peternell et al. (1996) or Chui (1997). For convenience and simplicity we assume that $p = f = i$. Under

these conditions, the output BHM would be given by:

$$\boldsymbol{Z_{0:2i-1}} = \left(\frac{\boldsymbol{Z_p}}{\boldsymbol{Z_f}}\right) = \left(\frac{\boldsymbol{Z_{0:i-1}}}{\boldsymbol{Z_{i:2i-1}}}\right) = \left(\begin{array}{cccc} \boldsymbol{z_0} & \boldsymbol{z_1} & \cdots & \boldsymbol{z_{T-2i}} \\ \boldsymbol{z_1} & \boldsymbol{z_2} & \cdots & \boldsymbol{z_{T-2i+1}} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{z_{i-1}} & \boldsymbol{z_i} & \cdots & \boldsymbol{z_{T-i-1}} \\ \hline \boldsymbol{z_i} & \boldsymbol{z_{i+1}} & \cdots & \boldsymbol{z_{T-i}} \\ \boldsymbol{z_{i+1}} & \boldsymbol{z_{i+2}} & \cdots & \boldsymbol{z_{T-i+1}} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{z_{2i-1}} & \boldsymbol{z_{2i}} & \cdots & \boldsymbol{z_{T-1}} \end{array}\right) \tag{3}$$

For short hand notation we denote, for any BHM $\boldsymbol{A}$: $\boldsymbol{A_p} = \boldsymbol{A_{0:i-1}}$, $\boldsymbol{A_{pr}} = \boldsymbol{A_{i:i}}$, $\boldsymbol{A_f} = \boldsymbol{A_{i:2i-1}}$ and $\boldsymbol{A_{f^+}} = \boldsymbol{A_{i+1:2i-1}}$. The different subscripts *p, pr* and *f* denote, respectively, the past, the present and the future blocks.

Definition 2.4 *(State sequences):* The state sequences are defined as,

$$\boldsymbol{X_t} = (\boldsymbol{x_t} \quad \boldsymbol{x_{t+1}} \quad \boldsymbol{x_{t+2}} \quad \cdots \quad \boldsymbol{x_{t+T-2i}}) \tag{4}$$

Taking this expression as starting point, past and future state sequences which begin, respectively, with $t = 0$ and $t = i$, can be written as $\boldsymbol{X_p} = \boldsymbol{X_0}$ and $\boldsymbol{X_f} = \boldsymbol{X_i}$.

On the other hand, the following matrices are related to the model parameters:

Definition 2.5 *(Extended observability matrix):*

$$\boldsymbol{O_i} = \begin{pmatrix} \boldsymbol{H} \\ \boldsymbol{H\Phi} \\ \boldsymbol{H\Phi^2} \\ \vdots \\ \boldsymbol{H\Phi^{i-1}} \end{pmatrix} \in \mathbb{R}^{im \times n} \tag{5}$$

Definition 2.6 *(Deterministic lower block triangular Toeplitz matrix):*

$$\boldsymbol{H_i^d} = \begin{pmatrix} \boldsymbol{D} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{H\Gamma} & \boldsymbol{D} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{H\Phi\Gamma} & \boldsymbol{H\Gamma} & \boldsymbol{D} & \dots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{H\Phi^{i-2}\Gamma} & \boldsymbol{H\Phi^{i-3}\Gamma} & \boldsymbol{H\Phi^{i-4}\Gamma} & \dots & \boldsymbol{D} \end{pmatrix} \in \mathbb{R}^{im \times r} \tag{6}$$

Definition 2.7 *(Stochastic lower block triangular Toeplitz matrices):*

$$\boldsymbol{H_i^{sw}} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{HE} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{H\Phi E} & \boldsymbol{HE} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{H\Phi^{i-2}E} & \boldsymbol{H\Phi^{i-3}E} & \boldsymbol{H\Phi^{i-4}E} & \dots & \boldsymbol{0} \end{pmatrix} \in \mathbb{R}^{im \times im} \tag{7}$$

$$\boldsymbol{H_i^{sv}} = \begin{pmatrix} \boldsymbol{C} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C} & \boldsymbol{0} & \dots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{C} & \dots & \boldsymbol{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \dots & \boldsymbol{C} \end{pmatrix} \in \mathbb{R}^{im \times im} \tag{8}$$

2.3 Subspace regression models

Given the matrices defined above, the basic subspace equation, which derives from the model in Definition 2.1., is

$$\boldsymbol{Z_f} = \boldsymbol{O_i X_i} + \boldsymbol{H_i^d U_f} + \boldsymbol{H_i^{sv} V_f} + \boldsymbol{H_i^{sw} W_f} \tag{9}$$

When a model can be written in innovations form, equation (9) collapses to the simpler form:

$$\boldsymbol{Z_f} = \boldsymbol{O_i X_i} + \boldsymbol{H_i^d U_f} + \boldsymbol{H_i^s E_f} \tag{10}$$

where $\boldsymbol{H_i^s}$ is just as (7) but with the identity matrix, $\boldsymbol{I} \in \mathbb{R}^{m \times m}$, in the main diagonal.

Finally, an important algebraic tool to estimate the parameters in (9) or (10) are the orthogonal projections, defined as:

Definition 2.8 *(Ortogonal projections):* Given $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, the orthogonal projector into the row space of $\boldsymbol{A}$ is denoted by $\boldsymbol{\pi_A}$ and defined as: $\boldsymbol{\pi_A} = \boldsymbol{A}^+\boldsymbol{A}$, where $\boldsymbol{A}^+$ is the Moore-Penrose pseudo-inverse of $\boldsymbol{A}$.

In the same way, $\boldsymbol{\pi}_{\boldsymbol{A}}^{\perp} = \boldsymbol{I} - \boldsymbol{A}^{+}\boldsymbol{A}$ is the matrix of the projection into the ortogonal complement of the row space of $\boldsymbol{A}$.

3 Main results

The basic problem consists of estimating all the parameters in the matrices $\boldsymbol{\Phi} \in \mathbb{R}^{n\times n}, \boldsymbol{\Gamma} \in \mathbb{R}^{n\times r}, \boldsymbol{H} \in \mathbb{R}^{m\times n}, \boldsymbol{D} \in \mathbb{R}^{m\times r}, \boldsymbol{E} \in \mathbb{R}^{n\times n}, \boldsymbol{C} \in \mathbb{R}^{m\times m}, \boldsymbol{Q} \in \mathbb{R}^{n\times n}, \boldsymbol{S} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{R} \in \mathbb{R}^{m\times m}$ of (1) and (2), using the time series of the input and output variables. In general, these matrices will be nonlinear functions of the parameters $\boldsymbol{\theta}$ in the standard representation.

3.1 Algorithm Subest1

3.1.1 Models in innovations form

As we have seen, a model in the innovations form (2) can be written in subspace representation as equation (10). By projecting this equation into the row space of $\boldsymbol{U_{0:2i-1}}$ (past and future information) and the row space of $\boldsymbol{Z_p}$ (only past information), we obtain,

$$\boldsymbol{Z_f \pi_{U,Z_p}} = \boldsymbol{O_i X_i \pi_{U,Z_p}} + \boldsymbol{H_i^d U_f \pi_{U,Z_p}} \tag{11}$$

where it is assumed that the noise is incorrelated with the inputs, i.e.,

$$\boldsymbol{E_f \pi_{U,Z_p}} = \boldsymbol{0} \tag{12}$$

From (11) we can obtain estimates of the states conditional to the inputs and the past values of the outputs as,

$$\hat{\boldsymbol{X}}_{\boldsymbol{i}} = \boldsymbol{O}_{\boldsymbol{i}}^{+}(\boldsymbol{Z}_{\boldsymbol{f}}\boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z}_{\boldsymbol{p}}} - \boldsymbol{H}_{\boldsymbol{i}}^{\boldsymbol{d}}\boldsymbol{U}_{\boldsymbol{f}}\boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z}_{\boldsymbol{p}}}) \tag{13}$$

Also from (10) we obtain (see Appendix A):

$$\boldsymbol{Z}_{\boldsymbol{f}^{+}} = \boldsymbol{O}_{\boldsymbol{i}-\boldsymbol{1}}\big[(\boldsymbol{\Phi} - \boldsymbol{E}\boldsymbol{H})\boldsymbol{X}_{\boldsymbol{i}} + \boldsymbol{E}\boldsymbol{Z}_{\boldsymbol{pr}} + (\boldsymbol{\Gamma} - \boldsymbol{E}\boldsymbol{D})\boldsymbol{U}_{\boldsymbol{pr}}\big] + \boldsymbol{H}_{\boldsymbol{i}-\boldsymbol{1}}^{\boldsymbol{d}}\boldsymbol{U}_{\boldsymbol{f}^{+}} + \boldsymbol{H}_{\boldsymbol{i}-\boldsymbol{1}}^{\boldsymbol{s}}\boldsymbol{E}_{\boldsymbol{f}^{+}} \tag{14}$$

Substituting the unknown states by its estimates (13), we can define the expected value of $\boldsymbol{Z}_{\boldsymbol{f}^{+}}$ conditioned to matrices $\hat{\boldsymbol{X}}_{\boldsymbol{i}}, \boldsymbol{Z}_{\boldsymbol{pr}}, \boldsymbol{U}_{\boldsymbol{pr}}$ and $\boldsymbol{U}_{\boldsymbol{f}^{+}}$ as,

$$\hat{\boldsymbol{Z}}_{\boldsymbol{f}^{+}} = E\big[\boldsymbol{Z}_{\boldsymbol{f}^{+}}/\hat{\boldsymbol{X}}_{\boldsymbol{i}}, \boldsymbol{Z}_{\boldsymbol{pr}}, \boldsymbol{U}_{\boldsymbol{pr}}, \boldsymbol{U}_{\boldsymbol{f}^{+}}\big] \tag{15}$$

Then, projecting (14) and $\hat{\boldsymbol{Z}}_{\boldsymbol{f}^{+}}$ into the row space of $\boldsymbol{U}$ and $\boldsymbol{Z}_{\boldsymbol{p}}$, we can determine $\boldsymbol{\Phi}, \boldsymbol{\Gamma}, \boldsymbol{E}, \boldsymbol{H}$ and $\boldsymbol{D}$, by solving the weighted least-squares problem:

$$\min_{\boldsymbol{\Phi},\boldsymbol{\Gamma},\boldsymbol{H},\boldsymbol{D},\boldsymbol{E}} \left\| \boldsymbol{\Omega}^{-\frac{\boldsymbol{1}}{\boldsymbol{2}}} \Big[\boldsymbol{Z}_{\boldsymbol{f}^{+}}\boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z}_{\boldsymbol{p}^{+}}} - \hat{\boldsymbol{Z}}_{\boldsymbol{f}^{+}}\boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z}_{\boldsymbol{p}^{+}}} \Big] \right\|_{F}^{2} \tag{16}$$

where $\|\cdot\|_F$ denotes de Frobenius norm; $\boldsymbol{\Omega}$ is the covariance matrix of the prediction errors which can be expressed as $\boldsymbol{\Omega} = \boldsymbol{Z}_{\boldsymbol{f}^{+}}\boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z}_{\boldsymbol{p}^{+}}}^{\perp}\boldsymbol{Z}_{\boldsymbol{f}^{+}}{}'$ and where we have used the fact that $\boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z}_{\boldsymbol{p}^{+}}}^{\perp}$ is idempotent.

Note that, $\boldsymbol{\Omega}$ contains the future information of the output which is uncorrelated with its pasts and the input. Consequently, matrices of the SS model in (9) are generated by the parameters $\boldsymbol{\theta}$ of the original model. In the same way, the

matrices of the subspace equations (5)-(8) are generated by those of the SS representation. Then, solving (16) requires iterative numerical techniques. Finally, the error variance, which in innovations form is $\boldsymbol{Q} = \boldsymbol{R} = \boldsymbol{S}$ can be written as:

$$\boldsymbol{Q} = \boldsymbol{H_i^s} \boldsymbol{E_{pr}} \boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z_{p^+}}} \boldsymbol{E_{pr}}' \boldsymbol{H_i^{s\prime}} \tag{17}$$

with

$$\boldsymbol{H_i^s} \boldsymbol{E_{pr}} \boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z_{p^+}}} = \boldsymbol{Z_{pr}} \boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z_{p^+}}} - \boldsymbol{O_i} \hat{\boldsymbol{X}}_{\boldsymbol{i}} - \boldsymbol{H_i^d} \boldsymbol{U_{pr}} \boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z_{p^+}}} \tag{18}$$

where in both equations $i = 1$, which is the row space of $\boldsymbol{Z_{pr}}$. Hence, we just need to compute the first row of each matrix, $\boldsymbol{H_i^s}$, $\boldsymbol{O_i}$ and $\boldsymbol{H_i^d}$. Note that (18) is the observation equation of the innovations model (2) formulated in subspace form.

3.1.2 General SS models

The procedure described above has a basic limitation, as the SS representation of some relevant models does not directly conform with (10). Some of these are Structural Time Series Models (STSM) (see Harvey, 1989; Harvey and Shepard, 1993), VARMAX models with observation errors (Terceiro, 1990) or Stochastic Volatility Models (Harvey et al., 1994). To apply the previous algorithm to these models, we devise a two-stage variant of the previous method. Firstly, we estimate parameters in matrices $\boldsymbol{\Phi}, \boldsymbol{\Gamma}, \boldsymbol{H}$ and $\boldsymbol{D}$ solving the problem,

$$\min_{\boldsymbol{\Phi},\boldsymbol{\Gamma},\boldsymbol{H},\boldsymbol{D}} \left\| \boldsymbol{\Omega}^{-\frac{1}{2}} \left[\boldsymbol{Z_f} \boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z_p}} - \boldsymbol{O_i} \hat{\boldsymbol{X}}_{\boldsymbol{i}} - \boldsymbol{H_i^d} \boldsymbol{U_f} \boldsymbol{\pi}_{\boldsymbol{U},\boldsymbol{Z_p}} \right] \right\|_F^2 \tag{19}$$

with $\boldsymbol{\Omega} = \boldsymbol{Z_f} \boldsymbol{\pi}^{\perp}_{\boldsymbol{U},\boldsymbol{Z_p}} \boldsymbol{Z_f}'$ and where the state estimates are the same as in the steady state innovations case. Notice that, whereas in equation (16) $\boldsymbol{E}$ is a decision

variable, here parameters in $\boldsymbol{E}$ are to be estimated in the second stage. Taking into account equation (13), we use state estimates as an approximation of the Kalman states (Ho and Kalman, 1966), to obtain initial conditions of the Kalman filter gain, $\tilde{\boldsymbol{K}}$ and the prediction error covariance, $\tilde{\boldsymbol{B}}$ (see Appendix B). Later, we compute the estimates for $\boldsymbol{E}, \boldsymbol{C}, \boldsymbol{Q}, \boldsymbol{S}$ and $\boldsymbol{R}$, by solving the constrained optimization problem:

$$\min_{\boldsymbol{E},\boldsymbol{C},\boldsymbol{Q},\boldsymbol{R},\boldsymbol{S}} \left\| \begin{bmatrix} \tilde{\boldsymbol{K}} \\ \tilde{\boldsymbol{B}} \end{bmatrix} - \begin{bmatrix} \hat{\boldsymbol{K}}_i \\ \hat{\boldsymbol{B}}_i \end{bmatrix} \right\|_F^2 \tag{20}$$

where $\hat{\boldsymbol{K}}_i$ and $\hat{\boldsymbol{B}}_i$ must satisfy the covariance equations of the Kalman Filter propagated i times:

$$\begin{aligned} \hat{\boldsymbol{K}}_i &= (\boldsymbol{\Phi} \boldsymbol{P}_{i|i-1} \boldsymbol{H}' + \boldsymbol{E}\boldsymbol{S}\boldsymbol{C}')\hat{\boldsymbol{B}}_i' \\ \boldsymbol{P}_{i+1|i} &= \boldsymbol{\Phi} \boldsymbol{P}_{i|i-1} \boldsymbol{\Phi}' + \boldsymbol{E}\boldsymbol{Q}\boldsymbol{E}' - \hat{\boldsymbol{K}}_i \hat{\boldsymbol{B}}_i \hat{\boldsymbol{K}}_i' \\ \hat{\boldsymbol{B}}_i &= \boldsymbol{H} \boldsymbol{P}_{i|i-1} \boldsymbol{H}' + \boldsymbol{C}\boldsymbol{R}\boldsymbol{C}' \end{aligned} \tag{21}$$

Loosely speaking, we estimate matrices $\boldsymbol{E}, \boldsymbol{C}, \boldsymbol{Q}, \boldsymbol{R}$ and $\boldsymbol{S}$ such that, (i) they return $\hat{\boldsymbol{K}}_i$ and $\hat{\boldsymbol{B}}_i$ propagating i times equations (21), and (ii) closely resembles the values $\tilde{\boldsymbol{K}}$ and $\tilde{\boldsymbol{B}}$ resulting from the previous stage.

3.2 Algorithm Subest2

This second procedure works exclusively with models in innovations form. First, as before, we start determining the states as in equation (13). Then we obtain the matrix $\tilde{\boldsymbol{Z}}_{\boldsymbol{f}}$ as the residual resulting from,

$$\tilde{\boldsymbol{Z}}_{\boldsymbol{f}} = \boldsymbol{Z}_{\boldsymbol{f}} - \boldsymbol{O}_i \hat{\boldsymbol{X}}_i - \boldsymbol{H}_i^d \boldsymbol{U}_{\boldsymbol{f}} \tag{22}$$

This residual matrix has a particular structure, with one-step-ahead errors in the first row, two-step-ahead errors in the second row and so on. Then, the gaussian loglikelihood function can be written as,

$$\log l(\theta) = -\frac{im}{2}\log(2\pi) - \frac{i}{2}\log(|\boldsymbol{\Sigma}|) - \frac{1}{2}tr(\tilde{\boldsymbol{Z}}_{\boldsymbol{f}}'\boldsymbol{\Sigma}^{-1}\tilde{\boldsymbol{Z}}_{\boldsymbol{f}}) \tag{23}$$

where, $tr(.)$ is the trace operator and $\boldsymbol{\Sigma}$ is the Prediction Error (PE) covariance. Forecasting errors $\tilde{\boldsymbol{Z}}_{\boldsymbol{f}}$ are obviously autocorrelated, so $\boldsymbol{\Sigma}$ has the following structure,

$$\boldsymbol{\Sigma} = \boldsymbol{O_i P_i O_i'} + \boldsymbol{H_i^s}(\boldsymbol{I_i} \otimes \boldsymbol{Q})\boldsymbol{H_i^{s\prime}} \tag{24}$$

where $\otimes$ is the Kronecker product, $\boldsymbol{P_i}$ is the covariance matrix of the states and $\boldsymbol{I_i}$ is an $i \times i$ identity matrix. There are two components in expression (24): the first addend in the right-hand-side refers to the error covariance of the states, while the second addend corresponds to the future output error variance conditional to the estimated states. Due to the structure of $\tilde{\boldsymbol{Z}}_{\boldsymbol{f}}$, the PE covariance matrix can be defined as,

$$\boldsymbol{\Sigma} = \begin{cases} \Sigma_{jk} \equiv PE_j \text{ covariance matrix} & \text{when } j = k \\ \Sigma_{jk} \equiv \text{cov}(PE_j, PE_k) & \text{when } j \neq k \end{cases} \qquad j,k = 1,2,...,i \tag{25}$$

where PE_l denotes the l-step-ahead prediction errors. To calculate expression (24), we must propagate i times the Kalman Filter covariance equations (21) as in 3.1.2, to compute $\boldsymbol{P_i}$.

The algorithm uses only models in innovations form but this is not a limita-

tion. To accommodate non-innovations models, we transform any general SS noise structure into its innovations representation by using the procedure by (Casals et al., 1999, Theorem 1). This is possible because a likelihood function is maximized and then, contrary to Subest1, the innovations variance is jointly estimated with the other parameters of the model.

4 Montecarlo experiments

We will now analyze the behavior of SUBEST1 and SUBEST2 in finite samples for several common time series models. Their performance is compared in terms of precision and computational cost with that of a state-space based ML algorithm, initialized with the true parameter values.

Tables 1-8 summarize the main results. The simulations in Tables 1-6 refer to homoskedastic models. They have been computed with 1,000 replications of each data generating processes. In each replication we obtain a sample of $T = 50$ and $T = 300$ observations, after discarding the first 50 values to improve randomization. Tables 7-8 show the results obtained for two common conditional heteroskedastic models. In this case the sample sizes are increased to $T = 500$ and $T = 3,000$, so they are representative of the high frequency financial time series to which these models are typically applied.

Tables 1-3 show the results obtained for three univariate nonseasonal models: gaussian AR(2), gaussian ARMA(2,1) and an ARMA(2,1) with errors drawn from a Student-t distribution, and a bivariate VARMA(2,1) model. The AR parameters

in all cases have been chosen so that the roots are complex and far from the unit circle. This avoids ill-conditioning situations due to approximate cancellation of real AR and MA roots. As could be expected:

1. ML estimates have a clear advantage in precision for gaussian models with MA terms. However, if the model is pure AR or has nongaussian errors, the root mean-squared errors (RMSEs) of SUBEST2 estimates are comparable to those of ML and consistently better than those of SUBEST1.

2. Computational cost of ML is 3.5/4 and 8/10 times the cost of SUBEST1 for $T = 50$ and $T = 300$, respectively. On the other hand, the cost of SUBEST2 consistently doubles the cost of SUBEST1 for $T = 50$. This overhead decreases when the sample size grows.

[INSERT TABLES 1-3]

Table 4 shows the results corresponding to a gaussian ARMA(1,1)$\times(0,1)_s$ process with $s = 4$ and $s = 12$, and some redundancy between the regular AR and MA roots. Note that ML keeps the advantage in precision but its computational overhead increases the cost of SUBEST1 to 4 and 6 times in the quarterly model and 9 and 14 times in the monthly model. This is due to the increased dimension of the state vector, which also affects SUBEST2 comparative performance, and also to the degradation in conditioning, which typically requires more iterations to achieve convergence. With $T = 300$, ML and SUBEST2 provide very similar results in terms of accuracy.

[INSERT TABLES 4.1-4.2]

Tables 5 and 6 show the results obtained with two typical formulations in the state-space literature: an STSM with a low signal-to-noise ratio and an AR(2) model with observation errors, respectively. Note in Table 5 that SUBEST1 is the slower method and its estimates are very imprecise in comparison with those of SUBEST2 and ML. This is mainly due to the fact that the loss function considered in SUBEST1 does not depend on the errors covariance matrix. Therefore, its estimates do not take into account the restriction of a null covariance between the errors in the state and observation equation. The low signal-to-noise ratio makes the matter worst, as it deteriorates the identificability of the parameters. On the other hand, SUBEST2 and ML estimates are more precise, as they take into account this critical independence restriction. Their computational overhead is also similar, due to the very small number of parameters to be estimated. Results provided by the Table 6 are rather similar, the main difference being the improvement of SUBEST1 owing to a higher signal-to-noise ratio.

[INSERT TABLES 5-6]

Finally, Tables 7 and 8 present results that had been obtained for two common high frequency financial models: Autorregresive Stochastic Volatility (ARSV) and Generalized Autorregresive Conditional Heteroskedasticity (GARCH) Models.

Results in Table 7 are close to those of an AR model with observation errors in precision and efficiency. This was expected, because the SS representation of both models is very similar. However, when the sample size increases, the computational cost of ML becomes between 5 and 20 times the cost of SUBEST1 and, between 3 and 12 times the cost of SUBEST2, for $T = 500$ and $T = 3,000$.

Table 8 shows that computational disadvantage of ML in the GARCH model is larger than in the previous case. In particular, computational load of ML becomes 49 times the cost of SUBEST1 and more than 16 times the cost of SUBEST2, for $T = 500$. When the sample size increases to $T = 3,000$, SUBEST2 becomes the fastest method. In fact, ML converges 90 and 79 times slower than SUBEST2 and SUBEST1, respectively, while the three methods remain very similar in terms of precision.

[INSERT TABLES 7-8]

Both algorithms are very useful with high frequency financial data. Furthermore, usual problems in the ML estimation of multivariate conditional heteroskedasticity models, as convergence problems due to ill-conditioning or doubts about gaussian distribution, strongly suggest the use of these alternative estimation methods.

References

Anderson, B. D. O. and Moore, J. B. (1979). *Optimal Filtering*. Englewood Cliffs (N.J.): Prentice Hall.

Ansley, C. F. and Newbold, P. (1980). Finite sample properties of estimators for autorregresion moving average models. *Journal of Econometrics*, 13:159–183.

Aoki, M. (1990). *State Space Modelling of Time Series*. Springer Verlag, New York.

Casals, J., Sotoca, S., and Jerez, M. (1999). A fast stable method to compute the likelihood of time invariant state space models. *Economics Letters*, 65(3):329–337.

Chui, N. L. C. (1997). *Subspace Methods and Informative Experiments for System Identification*. PhD thesis, Pembroke College Cambridge.

Dufour, J. M. and Pelletier, D. (2004). Linear estimation of weak varma models with macroeconomic applications. Technical report, Centre Interuniversitaire de Recherche en Economie Quantitative (CIREQ), Université de Montréal, CANADA.

Favoreel, W., De Moor, B., and Van Overschee, P. (2000). Subspace state space system identification for industrial processes. *Journal of Process Control*, 10:149–155.

Flores, R. and Serrano, G. (2002). A generalized least squares estimation method for varma models. *Statistics*, 36(4):303–316.

Francq, C. and Zakoïan, J. M. (2000). Estimating weak garch representations. *Econometric Theory*, 16:692–728.

Harvey, A. C. (1989). *Forecasting, structural time series models and th Kalman Filter*. Cambridge University Press.

Harvey, A. C., Ruiz, E., and Shepard, N. (1994). Multivariate stochastic variance models. *Review of Economic Studies*, 61:247–264.

Harvey, A. C. and Shepard, N. (1993). *Structural Time Series Models*, volume 11 of *Handbook of Statistics*. Elsevier Science Publishers B.V.

Ho, B. and Kalman, R. (1966). Effective construction of linear state-variable models from input-output functions. *Regelungstechnik*, 14:545–548.

Koreisha, S. G. and Pukkila, T. H. (1989). Fast linear estimation methods for vector autorregresive moving average models. *Journal of Time Series Analysis*, 10:325–339.

Koreisha, S. G. and Pukkila, T. H. (1990). A generalized least-squares approach for estimation of autorregresive moving-average models. *Journal of Time Series Analysis*, 11:139–151.

Ljung, L. (1999). *System Identification, Theory for the User*. PTR Prentice Hall.

Lütkepohl, H. and Poskitt, D. S. (1996). Specification of echelon form varma models. *Journal of Business and Economic Statistics*, 14(1):69–79.

Peternell, K., Sherrer, W., and Deistler, M. (1996). Statistical analysis of novel subspace identification methods. *Signal Processing*, 52:161–177.

Spliid, H. (1983). A fast estimation method for the vector autoregressive moving average model with exogenous variables. *Journal of the American Statistical Association*, 78(384):843–849.

Terceiro, J. (1990). *Estimation of Dynamics Econometric Models with Errors in Variables*. Springer-Verlag, Berlin.

Van Overschee, P. and De Moor, B. (1996). *Subspace Identification for Linear Systems: Theory, Implementation, Applications*. Kluwer Academic, Dordrecht, The Netherlands.

Viberg, M. (1995). Subspace-based methods for the identification of the linear time-invariant systems. *Automatica*, 31(12):1835–1852.

A Appendix

From equation (10), displacing time subscripts, we obtain:

$$\boldsymbol{Z_{f^+} = O_{i-1} X_{i+1} + H^d_{i-1} U_{f^+} + H^s_{i-1} E_{f^+}} \tag{26}$$

Note that $\boldsymbol{Z_{f^+}}$ includes only the future block information while $\boldsymbol{Z_f}$ integrates the present and the future. On the other hand, from the state equation in (2), we can express:

$$\boldsymbol{x_t} = (\boldsymbol{\Phi - EH})^i \boldsymbol{x_{t-i}} + \sum_{j=0}^{i-1} (\boldsymbol{\Phi - EH})^j [\boldsymbol{E z_{t-1-j}} + (\boldsymbol{\Gamma - ED}) \boldsymbol{u_{t-1-j}}] \tag{27}$$

which can be written in subspace form as,

$$\boldsymbol{X_i} = (\boldsymbol{\Phi - EH})^i \boldsymbol{X_p} + \boldsymbol{\Psi_i}(\boldsymbol{\Phi - EH, E}) \boldsymbol{Z_p} + \boldsymbol{\Psi_i}(\boldsymbol{\Phi - EH, \Gamma - ED}) \boldsymbol{U_p} \tag{28}$$

where $\boldsymbol{\Psi_i}(\boldsymbol{A, B}) = [\boldsymbol{A^{i-1} B} \quad \boldsymbol{A^{i-2} B} \quad ... \quad \boldsymbol{AB} \quad \boldsymbol{B}]$. Again, displacing time indices, we write:

$$\boldsymbol{X_{i+1}} = (\boldsymbol{\Phi - EH})^{i+1} \boldsymbol{X_{p^+}} + \boldsymbol{\Psi_{i+1}}(\boldsymbol{\Phi - EH, E}) \boldsymbol{Z_p^+} + \boldsymbol{\Psi_{i+1}}(\boldsymbol{\Phi - EH, \Gamma - ED}) \boldsymbol{U_p^+} \tag{29}$$

or, in the same way,

$$\boldsymbol{X_{i+1}} = (\boldsymbol{\Phi - EH}) \boldsymbol{X_i} + \boldsymbol{E Z_{pr}} + (\boldsymbol{\Gamma - ED}) \boldsymbol{U_{pr}} \tag{30}$$

Finally, substituting (30) into (26), we obtain:

$$\boldsymbol{Z_{f^+}} = \boldsymbol{O_{i-1}}[(\boldsymbol{\Phi} - \boldsymbol{EH})\boldsymbol{X_i} + \boldsymbol{EZ_{pr}} + (\boldsymbol{\Gamma} - \boldsymbol{ED})\boldsymbol{U_{pr}}] + \boldsymbol{H^d_{i-1}}\boldsymbol{U_{f^+}} + \boldsymbol{H^s_{i-1}}\boldsymbol{E_{f^+}} \quad (31)$$

B Appendix

To obtain the equations of the Kalman Filter (Anderson and Moore, 1979) in subspace form, we must propagate i times equations:

$$\begin{aligned}
\boldsymbol{\hat{Z}_{pr}} &= \boldsymbol{H}\boldsymbol{\hat{X}_{i|i-1}} + \boldsymbol{DU_{pr}} && (32)\\
\boldsymbol{\hat{X}_{i+1|i}} &= \boldsymbol{\Phi}\boldsymbol{\hat{X}_{i|i-1}} + \boldsymbol{\Gamma U_{pr}} + \boldsymbol{K_i}(\boldsymbol{Z_{pr}} - \boldsymbol{\hat{Z}_{pr}}) && (33)\\
\boldsymbol{K_i} &= (\boldsymbol{\Phi P_{i|i-1}H'} + \boldsymbol{ESC'})\boldsymbol{B'_i} && (34)\\
\boldsymbol{P_{i+1|i}} &= \boldsymbol{\Phi P_{i|i-1}\Phi'} + \boldsymbol{EQE'} - \boldsymbol{K_i B_i K'_i} && (35)\\
\boldsymbol{B_i} &= \boldsymbol{H P_{i|i-1} H'} + \boldsymbol{CRC'} && (36)
\end{aligned}$$

From (33) we get $\boldsymbol{\tilde{K}_i}$, an approximation for $\boldsymbol{K_i}$ as,

$$\boldsymbol{\tilde{K}_i} = \frac{(\boldsymbol{\hat{X}_{i+1}} - \boldsymbol{\Gamma U_{pr}}\boldsymbol{\pi_{U,Z_{p^+}}})}{\boldsymbol{\tilde{Z}_{pr}}} \quad (37)$$

where the state sequence is computed as,

$$\boldsymbol{\hat{X}_{i+1}} = \boldsymbol{O^+_{i-1}}(\boldsymbol{Z_{f^+}}\boldsymbol{\pi_{U,Z_{p^+}}} - \boldsymbol{H^d_{i-1}}\boldsymbol{U_{f^+}}\boldsymbol{\pi_{U,Z_{p^+}}}) \quad (38)$$

and $\boldsymbol{\tilde{Z}_{pr}}$ as,

$$\boldsymbol{\tilde{Z}_{pr}} = \boldsymbol{Z_{pr}}\boldsymbol{\pi_{U,Z_p}} - \boldsymbol{O_i}\boldsymbol{\hat{X}_i} - \boldsymbol{H^d_i}\boldsymbol{U_{pr}}\boldsymbol{\pi_{U,Z_p}} \quad (39)$$

Finally, an approximation of $\boldsymbol{B_i}$ is obtained as,

$$\tilde{\boldsymbol{B}}_{\boldsymbol{i}} = \tilde{\boldsymbol{Z}}_{\boldsymbol{pr}} \tilde{\boldsymbol{Z}}'_{\boldsymbol{pr}}$$

C Appendix

Table 1: Univariate nonseasonal models with gaussian errors.

Table 1.1 AR model (1-.4B+.3B^2)$Z_t = a_t$; $a_t \sim iidN(0,1)$.

	Method	SUBEST1			SUBEST2			ML		
T	True values	-.4	.3	1.0	-.4	.3	1.0	-.4	.3	1.0
	Average	-.384	.305	-.931	-.367	.284	.943	-.386	.304	.946
50	Std. Dev	.144	.144	.203	.136	.127	.198	.137	.135	.196
	RMSE[1]	.145	.144	.203	.136	.127	.198	.138	.135	.196
	Time	100%			216%			416%		
	Average	-.397	.306	-.985	-.391	.296	.992	-.395	.300	.994
300	Std. Dev	.059	.066	.083	.053	.055	.082	.053	.056	.082
	RMSE[1]	.059	.066	.083	.054	.055	.083	.054	.065	.083
	Time	100%			219%			1040%		

Table 1.2 ARMA model (1-.4B+.3B^2)$Z_t = (1 - .8B)a_t$; $a_t \sim iidN(0,1)$.

	Method	SUBEST1				SUBEST2				ML			
T	True values	-.4	.3	-.8	1.0	-.4	.3	-.8	1.0	-.4	.3	-.8	1.0
	Average	-.360	.289	-.671	1.020	-.307	.297	-.680	1.000	-.354	.313	-.800	.927
50	Std. Dev	.305	.175	.291	.214	.222	.141	.254	.206	.196	.141	.197	.194
	RMSE[1]	.308	.175	.318	.215	.240	.141	.281	.206	.201	.141	.197	.207
	Time	100%				244%				359%			
	Average	-.398	.309	-.763	1.014	-.389	.304	-.805	1.001	-.395	.304	-.797	.993
300	Std. Dev	.085	.071	.061	.085	.066	.061	.074	.083	.066	.061	.050	.081
	RMSE[1]	.085	.072	.072	.086	.067	.062	.074	.083	.066	.062	.050	.081
	Time	100%				230%				881%			

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.

Table 2: Univariate nonseasonal models with t-student errors.

Table 2.1: ARMA model (1-.4B+.3B^2)$Z_t = (1 - .8B)a_t$; $a_t \sim iid$ t with 400 d.f.

	Method	SUBEST1				SUBEST2				ML			
T	True values	-.4	.3	-.8	1.0	-.4	.3	-.8	1.0	-.4	.3	-.8	1.0
	Average	-.379	.294	-.688	1.037	-.293	.300	-.691	1.016	-.364	.316	-.808	.942
50	Std. Dev	.292	.174	.280	.218	.207	.145	.238	.207	.183	.143	.182	.195
	RMSE[1]	.292	.174	.302	.221	.233	.145	.262	.208	<u>.186</u>	<u>.144</u>	<u>.182</u>	<u>.203</u>
	Time	100%				246%				357%			
	Average	-.399	.309	-.764	1.019	-.390	.303	-.808	1.006	-.398	.302	-.802	.998
300	Std. Dev	.087	.070	.061	.086	.064	.060	.074	.085	.067	.060	.053	.084
	RMSE[1]	.087	.071	.071	.088	<u>.065</u>	<u>.060</u>	.075	.085	.067	<u>.060</u>	<u>.053</u>	<u>.084</u>
	Time	100%				230%				866%			

Table 1.2: ARMA model (1-.4B+.3B^2)$Z_t = (1 - .8B)a_t$; $a_t \sim iid$ t with 8 d.f.

	Method	SUBEST1				SUBEST2				ML			
T	True values	-.4	.3	-.8	1.3	-.4	.3	-.8	1.3	-.4	.3	-.8	1.3
	Average	-.383	.282	-.691	1.009	-.282	.294	-.678	.991	-.354	.308	-.802	.919
50	Std. Dev	.315	.193	.314	.256	.211	.161	.255	.247	.193	.155	.207	.238
	RMSE[1]	.315	.194	.332	<u>.411</u>	.242	.161	.283	.419	<u>.199</u>	<u>.155</u>	<u>.207</u>	.475
	Time	100%				245%				368%			
	Average	-.401	.307	-.765	1.007	-.385	.302	-.808	.994	-.399	.300	-.803	.986
300	Std. Dev	.088	.072	.065	.059	.066	.061	.076	.057	.068	.061	.053	.055
	RMSE[1]	.088	.072	.074	<u>.328</u>	<u>.068</u>	<u>.061</u>	.077	.341	<u>.068</u>	<u>.061</u>	<u>.053</u>	.349
	Time	100%				230%				862%			

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.

Table 3: Bivariate VARMA model with gaussian errors.

$$\begin{pmatrix} 1-.7B+.6B^2 & 0 \\ 0 & 1-1.3B+.5B^2 \end{pmatrix} \begin{pmatrix} z_{1t} \\ z_{2t} \end{pmatrix} = \begin{pmatrix} 1-.3B & -.9B \\ .6B & 1-.8B \end{pmatrix} \begin{pmatrix} a_{1t} \\ a_{2t} \end{pmatrix}$$

$$\begin{pmatrix} a_{1t} \\ a_{2t} \end{pmatrix} \sim iidN \left[\begin{pmatrix} .0 \\ .0 \end{pmatrix}, \begin{pmatrix} .07 & .02 \\ - & .05 \end{pmatrix} \right]$$

Table 3.1: Results with sample size $T = 50$

True values	-0.70	-1.30	0.60	0.50	-0.30	0.60	-0.90	-0.80	0.07	0.02	0.05
Method	SUBEST1										
Average	-.671	-1.359	.545	.557	-.267	.431	-.743	-.736	.079	.019	.056
Std. Dev	.137	.233	.104	.177	.200	.104	.177	.246	.017	.010	.014
RMSE[1]	.140	.240	.118	.186	.203	.199	.236	.254	.019	<u>.010</u>	.015
Time	100%										
Method	SUBEST2										
Average	-.620	-1.297	.519	.500	-.331	.453	-.832	-.776	.073	.017	.056
Std. Dev	.133	.202	.100	.151	.208	.121	.195	.251	.016	.010	.014
RMSE[1]	.155	.202	.129	<u>.151</u>	.210	.190	<u>.207</u>	.252	<u>.017</u>	<u>.010</u>	.015
Time	348%										
Method	MVE										
Average	.686	-1.279	.589	.492	-.344	.654	-1.024	-.872	.060	.017	.044
Std. Dev	.087	.188	.072	.155	.175	.122	.179	.220	.014	.010	.010
RMSE[1]	<u>.088</u>	<u>.189</u>	<u>.073</u>	.155	<u>.181</u>	<u>.134</u>	.218	<u>.231</u>	<u>.017</u>	<u>.010</u>	<u>.012</u>
Time	398%										

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.

Table 3.2: Results with sample size $T = 300$

True values	-.70	-1.30	.60	.50	-.30	.60	-.90	-.80	.07	.02	.05
Method	SUBEST1										
Average	-.693	-1.318	.580	.515	-.276	.536	-.829	-.769	.074	.021	.054
Std. Dev	.044	.090	.038	.072	.084	.044	.063	.099	.006	.004	.005
RMSE[1]	.045	.092	.043	.074	.087	.078	.094	.104	.008	<u>.004</u>	.006
Time	100%										
Method	SUBEST2										
Average	-.686	-1.311	.584	.506	-.289	.585	-.902	-.831	.072	.019	.053
Std. Dev	.039	.076	.034	.062	.070	.043	.072	.089	.006	.004	.005
RMSE[1]	.042	.076	.037	.062	.070	.046	.072	.095	<u>.006</u>	<u>.004</u>	.006
Time	342%										
Method	MVE										
Average	-.699	-1.296	.599	.500	-.305	.605	-.913	-.804	.069	.019	.049
Std. Dev	.030	.055	.023	.047	.062	.035	.052	.066	.006	.004	.004
RMSE[1]	<u>.030</u>	<u>.055</u>	<u>.023</u>	<u>.047</u>	<u>.062</u>	<u>.035</u>	<u>.054</u>	<u>.066</u>	<u>.006</u>	<u>.004</u>	<u>.004</u>
Time	744%										

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.

Table 4: Univariate seasonal ARMA models.

Table 4.1 ARMA model $(1-.4B)Z_t = (1-.7B)(1-.8B^4)a_t$; $a_t \sim iidN(0, 1.0)$

	Method	SUBEST1				SUBEST2				ML			
T	True Values	-.4	-.7	-.8	1.0	-.4	-.7	-.8	1.0	-.4	-.7	-.8	1.0
	Average	-.208	-.390	-.456	1.174	-.294	-.632	-.742	.924	-.325	-.683	-.835	.886
50	Std. Dev	.557	.585	.168	.336	.486	.482	.153	.210	.325	.329	.169	.204
	RMSE[1]	.589	.662	.383	.378	.497	.486	<u>.163</u>	<u>.223</u>	<u>.333</u>	<u>.330</u>	.173	<u>.233</u>
	Time	100%				217%				441%			
	Average	-.302	-.591	-.575	1.228	-.365	-.665	-.826	.998	-.388	-.695	-.800	.990
300	Std. Dev	.250	.267	.088	.153	.147	.132	.064	.084	.146	.122	.042	.080
	RMSE[1]	.269	.288	.242	.275	.151	.136	.069	.084	<u>.146</u>	<u>.122</u>	<u>.042</u>	<u>.081</u>
	Time	100%				227%				577%			

Table 4.2 ARMA model $(1-.4B)Z_t = (1-.7B)(1-.8B^{12})a_t$; $a_t \sim iidN(0, 1.0)$

	Method	SUBEST1				SUBEST2				ML			
T	True values	-.4	-.7	-.8	1.0	-.4	-.7	-.8	1.0	-.4	-.7	-.8	1.0
	Average	-.240	-.493	-.368	1.133	-.244	-.592	-.494	1.073	-.324	-.680	-.807	.902
50	Std. Dev	.525	.534	.128	.261	.438	.454	.042	.237	.331	.323	.224	.225
	RMSE[1]	.549	.572	.450	.293	.465	.467	.309	.248	<u>.340</u>	<u>.323</u>	<u>.224</u>	<u>.245</u>
	Time	100%				436%				907%			
	Average	-.374	-.661	-.586	1.221	-.379	-.691	-.812	.970	-.386	-.694	-.805	.984
300	Std. Dev	.221	.201	.055	.118	.143	.128	.073	.084	.135	.112	.048	.083
	RMSE[1]	.222	.205	.221	.250	.145	.128	.074	.089	<u>.136</u>	<u>.112</u>	<u>.048</u>	<u>.085</u>
	Time	100%				371%				1415%			

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.

Table 5: Structural time series model.

$$\begin{pmatrix} T_{t+1} \\ \Delta_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} T_t \\ \Delta_t \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \eta_t$$

$$y_t = (1 \quad 0) \begin{pmatrix} T_t \\ \Delta_t \end{pmatrix} + \epsilon_t; \quad \begin{pmatrix} \eta_t \\ \epsilon_t \end{pmatrix} \sim iidN \left[\begin{pmatrix} .0 \\ .0 \end{pmatrix}, \begin{pmatrix} .01 & 0 \\ 0 & 10 \end{pmatrix} \right]$$

	Method	SUBEST1		SUBEST2		MVE	
T	True values	.01	10	.01	10	.01	10
	Average	1.475	8.473	.022	9.726	.035	9.911
50	Std. Dev	3.275	3.693	.040	1.406	.069	1.005
	RMSE[1]	3.588	3.996	<u>.042</u>	1.432	.073	<u>1.009</u>
	Time	395%		125%		100%	
	Average	.461	10.137	.019	9.977	.008	10.005
300	Std. Dev	.146	.476	.014	.443	.006	.409
	RMSE[1]	.474	.495	.017	.444	<u>.007</u>	<u>.409</u>
	Time	273%		100%		171%	

Table 6: AR model with observation errors.

$$(1 - 1.5B + .8B^2)z_t^* = a_t; \quad z_t = z_t^* + v_t;$$

$$\begin{pmatrix} a_t \\ v_t \end{pmatrix} \sim iidN \left[\begin{pmatrix} .0 \\ .0 \end{pmatrix}, \begin{pmatrix} 1.0 & 0 \\ 0 & .5 \end{pmatrix} \right]$$

	Method	SUBEST1				SUBEST2				ML			
T	True values	-1.5	.8	1.0	.5	-1.5	.8	1.0	.5	-1.5	.8	1.0	.5
	Average	-1.425	.731	1.030	.558	-1.429	.737	1.045	.558	-1.480	.781	.954	.469
50	Std. Dev	.268	.187	.219	.202	.133	.121	.195	.177	.128	.119	.185	.169
	RMSE[1]	.278	.199	.221	.210	.150	.136	.200	.186	<u>.129</u>	<u>.121</u>	<u>.190</u>	<u>.172</u>
	Time	100%				204%				142%			
	Average	-1.491	.798	.990	.530	-1.487	.788	1.010	.513	-1.492	.793	.997	.493
300	Std. Dev	.057	.046	.081	.069	.047	.044	.075	.057	.044	.043	.072	.055
	RMSE[1]	.057	.047	.081	.076	.048	.046	.076	.059	<u>.045</u>	<u>.043</u>	<u>.072</u>	<u>.056</u>
	Time	100%				193%				253%			

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.

Table 7: ARSV(1) model in State Space form.

$$x_{t+1} = .95x_t + w_t$$
$$z_t = x_t + 1.5u_t + v_t$$
$$E(w_t) = E(v_t) = 0, \quad E(w_t w_t') = .5, \quad E(v_t v_t') = 2, \quad E(w_t v_t') = 0$$

	Method	SUBEST1				SUBEST2				ML			
T	True values	.95	1.50	.50	2.0	.95	1.50	.50	2.0	.95	1.50	.50	2.0
	Average	.932	1.414	.469	1.999	.945	1.453	.462	2.010	.933	1.471	.517	1.990
500	Std. Dev	.164	1.105	.146	.087	.041	1.027	.146	.085	.034	.460	.104	.077
	RMSE[1]	.165	1.108	.149	.087	.041	1.028	.151	.085	<u>.038</u>	<u>.461</u>	<u>.106</u>	<u>.078</u>
	Time	100%				168%				476%			
	Average	.949	1.485	.497	2.001	.949	1.484	.494	2.004	.948	1.490	.501	2.000
3000	Std. Dev	.010	.202	.044	.032	.009	.198	.044	.033	.009	.183	.038	.031
	RMSE[1]	.010	.202	.044	.032	<u>.009</u>	.198	.044	.033	<u>.009</u>	<u>.184</u>	<u>.039</u>	<u>.031</u>
	Time	100%				166%				2059%			

Table 8: GARCH(1,1) model in ARMA form.

$$y_t = \epsilon_t \text{ with } \epsilon_t \sim iid(0, 1.0) \text{ and } \epsilon_t|\Omega_{t-1} \sim iidN(0, \sigma_t^2) \text{ where}$$
$$\epsilon_t^2 = 1.0 + \tfrac{1-.80B}{1-.97B} v_t \quad \text{with } v_t = \epsilon_t^2 - \sigma_t^2$$

	Method	SUBEST1			SUBEST2			ML		
T	True values	1.0	-.97	-.80	1.0	-.97	-.80	1.0	-.97	-.80
	Average	1.005	-.972	-.798	1.002	-.969	-.814	1.209	-.952	-.784
500	Std. Dev	.766	.025	.076	.760	.031	.106	1.420	.037	.056
	RMSE[1]	.766	<u>.025</u>	.076	<u>.760</u>	.031	.107	1.436	.041	<u>.058</u>
	Time	100%			298%			4916%		
	Average	.999	-.974	-.810	.998	-.972	-.809	1.095	-.967	-.797
3000	Std. Dev	.314	.015	.052	.313	.019	.068	.880	.011	.018
	RMSE[1]	.314	.016	.053	<u>.313</u>	.019	.069	.885	<u>.011</u>	<u>.018</u>
	Time	114%			100%			9047%		

[1]RMSE is the root mean-squared error. The smallest RMSEs are underlined. ML algorithm is initialized with the true parameter values.