Faster ARMA Maximum Likelihood Estimation

A.I. McLeod[a] and Y. Zhang[b]

[a] *Department of Statistical and Actuarial Sciences,*
The University of Western Ontario,
London, Ontario Canada N6A 5B7

[b] *Department of Mathematics and Statistics,*
Acadia University,
Wolfville, Nova Scotia, Canada B4P 2R6

July 5, 2007

Abstract

A new likelihood based AR approximation is given for ARMA models. The usual algorithms for the computation of the likelihood of an ARMA model require $O(n)$ flops per function evaluation. Using our new approximation, an algorithm is developed which requires only $O(1)$ flops in repeated likelihood evaluations. In most cases, the new algorithm gives results identical to or very close to the exact maximum likelihood estimate (MLE). This algorithm is easily implemented in high level Quantitative Programming Environments (QPEs) such as *Mathematica*, MatLab and R. In order to obtain reasonable speed, previous ARMA maximum likelihood algorithms are usually implemented in C or some other machine efficient language. With our algorithm it is easy to do maximum likelihood estimation for long time series directly in the QPE of your choice. The new algorithm is extended to obtain the MLE for the mean parameter. Simulation experiments which illustrate the effectiveness of the new algorithm are discussed. *Mathematica* and R packages which implement the algorithm discussed in this paper are available (McLeod and Zhang, 2007). Based on these package implementations, it is expected that the interested researcher would be able to implement this algorithm in other QPE's.

Keywords: Autoregressive approximation; Efficiency of the sample mean; Maximum likelihood estimator; High-order autoregression; Long time series and massive datasets; Quantitative programming environments

1. Introduction

The ARMA(p,q) model may be written in operator notation as $\phi(\mathcal{B})(z_t-\mu)=\theta(\mathcal{B})a_t$, where $\mathcal{B}$ is the backshift operator on t, $\phi(\mathcal{B})=1-\phi_1\mathcal{B}-...-\phi_p\mathcal{B}^p$, $\theta(\mathcal{B})=1-\theta_1\mathcal{B}-...-\theta_p\mathcal{B}^q$, μ is the mean of z_t and a_t is assumed to be Gaussian white noise with mean zero and variance σ_a^2. It is assumed that z_t is causal-stationary and invertible so that all roots of $\phi(\mathcal{B})\theta(\mathcal{B})=0$ are outside the unit circle. For model identifiability it is assumed that $\phi(\mathcal{B})$ and $\theta(\mathcal{B})$ have no common factors. Given n consecutive observations from this time series model, $z_1,\ldots,z_n$, the log-likelihood function was discussed by Box, Jenkins and Reinsel (1994), as well as many other authors. Other asymptotically first-order efficient methods are available, such as the *HR* algorithm (Hannan and Rissanen, 1982) but many researchers prefer methods of estimation and inference based on the likelihood function (Barnard, Jenkins and Winsten, 1962; Fisher, 1973; Box and Luceño, 1997, §12B) and Taniguchi (1983) has shown that MLE is second-order efficient. Some of the widely used algorithms for ARMA likelihood evaluation are listed in Box and Luceño (1997, §12B). All of these algorithms require $O(n)$ flops per likelihood evaluation. The algorithm presented in §3 requires only $O(1)$ flops per evaluation and so is much more efficient for longer time series. This is especially important when implementing the algorithm in a high level QPE. For example, one may be interested in forecasting long time series in biomedical signal processing using MatLab (Baura, 2002, §7.1). In §2 we discuss the AR(p) case and in §3 the extension to the ARMA(p,q) case.

2. AR(p) Case

2.1. Exact Likelihood Function

It follows from Champernowne (1948, eq. 3.5) and Box, Jenkins and Reinsel (1994, eqn. A7.4.10) that the log-likelihood function may be written

$$L(\phi, \mu, \sigma_a^2) = -\frac{n}{2}\log(\sigma_a^2) - \frac{1}{2}\log(g_p) - S(\phi, \mu)/(2\sigma_a^2), \tag{1}$$

where $\phi = (\phi_1, \ldots, \phi_p)$, $g_p = \det(\Gamma_n \sigma_a^{-2}) = \det(\Gamma_p \sigma_a^{-2})$, Γ_n is the covariance matrix of n successive observations,

$$S(\phi, \mu) = \beta' D \beta, \tag{2}$$

where D, the Champernowne matrix, is the $(p+1) \times (p+1)$ matrix with (i, j)-entry,

$$D_{i,j} = D_{j,i} = (z_i - \mu)(z_j - \mu) + \ldots + (z_{n+1-j} - \mu)(z_{n+1-i} - \mu) \tag{3}$$

and $\beta = (-1, \phi)$. It should be pointed out that Champernowne (1948, p.206) assumes $n > 2p$. However, it may be shown (McLeod and Zhang, 2007) that eqn. (2) is valid if and only if $n \geq 2p$.

Maximizing over σ_a^2, the concentrated log-likelihood may be written

$$L_c(\phi, \mu) = -\frac{n}{2}\log(S(\phi, \mu)/n) - \frac{1}{2}\log(g_p). \tag{4}$$

As in Jones (1980), the parametrization using partial autocorrelations (Barndorff-Nielsen and Schou, 1973),

$$(\phi_1, \ldots, \phi_p) \longleftrightarrow (\zeta_1, \ldots, \zeta_p) \tag{5}$$

may be used to constrain the optimization. In the reparameterized model,

$$g_p = \prod_{j=1}^{p} (1 - \zeta_j^2)^{-j}. \tag{6}$$

The Burg estimators are used as initial estimates since they are more accurate than the Yule-Walker estimates in many situations (Percival and Walden, 1993, p.414; Zhang and McLeod, 2006b). Like the Yule-Walker estimates, the Burg estimates are always inside the admissible region and may be efficiently computed using the Durbin-Levinsion recursion (Percival and Walden, 1993, p.452). Modern QPEs provide various built-in algorithms for nonlinear function optimization which may be used to obtain the MLE of ϕ. Since the sample mean, $\bar{z} = (z_1 + \cdots + z_n)/n$, is an asymptotically fully efficient estimate of μ, it is often used in place of the MLE. This algorithm using the sample mean to estimate μ and then MLE for the other parameters will be denoted by *SampleMean* in the following sections.

If the sample mean is used, μ may be replaced by $\bar{z}$ in (3) and so after the initial evaluation, repeated evaluations of (4) require $O(1)$ flops, which explains why the new algorithm is efficient for long time series. Since it practice p is considered fixed, it is not included in the asymptotic flop count.

2.2. Exact MLE for the Mean Parameter

The exact MLE for the mean may be obtained by simply optimizing the log-likelihood function given in (4). However, this would then require $O(n)$ flops per function evaluation. A more efficient approach is now presented.

Assuming that ϕ is known, the exact MLE is given by,

$$\hat{\mu} = \frac{1_n' \Gamma_n^{-1} z}{1_n' \Gamma_n^{-1} 1_n}, \tag{7}$$

where 1_n denotes the n dimensional column vector with all entries equal to 1, $1_n'$ denotes its transpose and $z = (z_1, \ldots, z_n)$. Since $\hat{\mu}$ does not depend on σ_a^2, we may assume without loss of generality that $\sigma_a^2 = 1$. Direct evaluation of (7) using the exact inverse matrix derived by Siddiqui (1958) would require $O(n^2)$ flops. A more efficient approach may be developed using the inverse matrix result of Zinde-Walsh (1988). Zinde-Walsh (1988, eqn. 3.2) showed that

$$\Gamma_n^{-1} = \dot{\Gamma}_n - \Omega, \tag{8}$$

where $\dot{\Gamma}_n$ denotes the $n \times n$ matrix with (i,j)-entry given by $\gamma_{i-j}^{(u)}$, where $\gamma_k^{(u)} = \text{ Cov}\,(u_t, u_{t-k})$, $u_t = \phi(\mathcal{B})a_t$ and Ω is a zero matrix except for $p \times p$ submatrices in the upper-left and lower-right corners. The (i,j)-entry of the submatrix of Ω in the upper-left corner is

$$\Omega_{i,j} = \sum_{k=\min(i,j)}^{p-|i-j|} \phi_k \phi_{k+|i-j|}. \tag{9}$$

The matrix in the lower-right corner is just the transpose of the upper-left corner submatrix. Using the above results it was found that,

$$\begin{aligned} 1_n' \Gamma_n^{-1} &= 1_n' \phi^2(1) - (\epsilon_1, \ldots, \epsilon_p, 0, \ldots, 0, \epsilon_p, \ldots, \epsilon_1) \\ &\quad -(\kappa_1, \ldots, \kappa_p, 0, \ldots, 0, \kappa_p, \ldots, \kappa_1), \end{aligned} \tag{10}$$

where $\phi(1) = 1 - \phi_1 - \ldots - \phi_p$, $\epsilon = 1_n' \Omega$,

$$\epsilon = (\epsilon_1, \ldots, \epsilon_p, 0, \ldots, 0, \epsilon_p, \ldots, \epsilon_1) \tag{11}$$

and

$$\kappa_i = \sum_{k=1}^{i} \gamma_k^{(u)}. \tag{12}$$

Using (10), $\hat{\mu}$ can now be evaluated in $O(n)$ flops. Note that this evaluation will only typically be two or three times in the full MLE algorithm outlined below.

An iterative algorithm, *MeanMLE*, is used for the simultaneous joint MLE of $(\phi_1, \ldots, \phi_p, \mu)$,

Step 0 Set the maximum number of iterations, $M \leftarrow 5$. Set the iteration counter, $i \leftarrow 0$. Set $\hat{\mu}^{(0)} \leftarrow \bar{z}$, where $\bar{z}$ is the sample mean. Obtain initial parameter values $\hat{\phi}_k^{(0)},\ k = 1, \ldots, p$ using the Burg algorithm or set $\hat{\phi}_k^{(0)} = 0,\ k = 1, \ldots, p$. Set $\ell_0 = L_c(\hat{\phi}^{(0)}, \hat{\mu}^{(0)})$.

Step 1 Obtain $\hat{\phi}_k^{(i+1)},\ k = 1, \ldots, p$ by numerically maximizing $L_c(\phi, \hat{\mu}^{(i)})$ over ϕ. Set $\ell_{i+1} = L_c(\hat{\phi}^{(i+1)}, \hat{\mu}^{(i)})$.

Step 2 Using $\hat{\phi}^{(i+1)}$ evaluate $\hat{\mu}^{(i+1)}$.

Step 3 Terminate when ℓ_{i+1} has converged or $i > M$. Otherwise set $i \leftarrow i + 1$ and return to Step 1 to perform the next iteration.

Convergence usually occurs in two or three iterations.

2.3. Champernowne Matrix Computation

$D_{i,j}$ has $n - (i + 1) - (j + 1)$ terms so each term requires $O(n)$ flops. If the sample mean is used, this computation only has to be done once, but if the exact MLE for the mean is used, D must be computed several times. It

may be shown that $D = C - E$, where the (i, j)-entry of the matrix C may be written, $C_{|i-j|}$, where $C_k = z_1 z_k + \ldots + z_{n-k} z_n$. The (i, j)-entry for the matrix E may be computed sequentially $E_{i+1,j+1} =$
$E_{i,j} + z_i z_j + z_{n+1-i} z_{n+1-j}$, $i < j$. Using the above results reduces the flop count for the matrix D slightly.

3. ARMA Maximum Likelihood Estimation

Previous AR-approximation methods for fitting MA(q) and ARMA(p, q) were based on first fitting a suitable high-order autoregressive approximation (Durbin, 1959; Parzen, 1969; Hannan and Rissanen, 1982; Wahlberg, 1989; Choi, 1992 §4.1). The next step is to use the fitted AR model to estimate an MA(q) or ARMA(p, q) model. As noted by McClave (1973), this approach can lead to biased estimates which have larger mean-square error than the MLE.

Instead of directly fitting an autoregressive model to the time series, our new method is based on approximating the exact likelihood function for the ARMA(p, q) model by the likelihood function for a suitable high-order autoregression. The approximating autoregression of order r is determined as the minimum mean-square error (MMSE) linear predictor of order r for the ARMA(p, q) model, $\varphi(B)(z_t - \mu) = a_t$, where
$\varphi(\mathcal{B}) = 1 - \varphi_1 \mathcal{B} - \ldots - \varphi_r \mathcal{B}^r$. By taking r sufficiently large, an accurate approximation to the exact ARMA(p, q) likelihood may be obtained. In practice $r = 30$ is sufficient for many ARMA models as we will now show.

The Kullback-Leibler discrepancy may be used to choose a suitable r. Letting $\Sigma_{\phi,\theta}$ and Σ_{φ} denote the covariance matrices for the ARMA(p, q)

and its AR(r) approximation, the Kullback-Leibler discrepancy may be written (Ullah, 2002, eqn. 5),

$$\mathcal{I} = \frac{1}{2}(\ \mathrm{tr}\,\Sigma_{\phi,\theta}\Sigma_{\varphi}^{-1} - \log|\Sigma_{\phi,\theta}|/|\Sigma_{\varphi}| - n). \tag{13}$$

Figure 1 displays a plot of $\mathcal{I}$ in the case of an MA (1) model with $\theta_1 = 0.9$ and $n = 200$. It is seen that $r = 30$ works well even for this model with a parameter near the non-invertible boundary. It appears that $r = 30$ is adequate for many sorts of models occurring in applications although as the parameters move very close to the non-invertible boundary, our approximates requires larger r and fails entirely when the boundary is reached. A *Mathematica* notebook to compute and plot the Kullback-Leibler discrepancy for the ARMA(p, q) and its AR(r) approximation is available (McLeod and Zhang, 2007).

[Figure 1 here]

In practice, as shown by simulation in §4.2, our method with $r = 30$ can still be used even when there is a root on the boundary but the statistical efficiency relative to existing exact MLE algorithms is reduced. Models with a root on the non-invertible boundary usually indicate over-differencing and may be avoided by refitting with an alternative model specification (Zhang and McLeod, 2006a).

After a suitable r has been chosen, the ARMA likelihood may be obtained from (4),

$$L_c(\phi, \theta, \mu) = L_c(\varphi, \mu), \tag{14}$$

where $\varphi = (\varphi_1, \ldots, \varphi_r)$. Then $L_c(\phi, \theta, \mu)$ may be maximized using a built-in optimization function. The algorithm given in §2.2 may be used to compute

the exact MLE for the mean by using this AR(r) approximation. As shown in §4.3, this algorithm works as well as existing exact MLE algorithms for the mean in ARMA$(1,1)$ models.

In *Mathematica*, MatLab and in R, nonlinear optimization functions which can handle box constraints are available. In this case it is useful to reparametrize the ARMA model as suggested by Monahan (1984) using the transformation of Barndorff-Nielsen and Schou (1973). Alternatively, if only an unconstrained optimization function is available then a penalty function approach may be used to constrain the parameters to the admissible region. This penalty function approach has been used for many years with the Powell (1964) algorithm in our MHTS Time Series Package (McLeod and Hipel, 2007) for a wide variety of MLE problems in time series analysis (Hipel and McLeod, 1994).

Usually it is most expedient to set the initial parameter estimates to zero. In case of difficulty with convergence, initial estimates may be obtained (Hannan and Rissanen, 1982) by fitting a high order autoregression to provide estimates of the innovations and then using linear regression to estimate the parameters ϕ and θ. Experience suggests, as is illustrated in §4.1, computing initial parameter estimates in the ARMA case usually does not significantly increase the speed and, in practice, convergence is rarely an issue. In particular, convergence was obtained for all models fitted in §4 without difficulty.

A simple alternative to the MMSE linear predictor approximation is to just use the truncated inverted form of model (Box, Jenkins and Reinsel, 1994, §4.2.3), $\pi(B)(z_t - \mu) = a_t$, where $\pi(B) = 1 - \pi_1 B - \ldots - \pi_r B^r$. The

coefficients π_k, $k = 1, \ldots, r$ are obtained from
$\pi_k = \phi_k + \theta_1 \pi_{k-1} - \ldots - \theta_q \pi_q - \phi_k$ using boundary conditions
$\pi_0 = 1; \pi_k = 0$ if $k < 0$ and $\phi_k = 0$ if $k > p$. When r is chosen large enough, this approximates the MMSE predictor (Brockwell and Davis, 1991, §5). However, for fixed r there will always be parameter values in the admissible ARMA(p, q) region for which $\varphi(B) = 0$ has roots outside the admissible region for a causal-stationary AR(r). As shown in Table 1, the MMSE predictor provides a much more accurate approximation in terms of the Kullback-Leibler discrepancy. For these reasons the MMSE linear predictor approximation is used.

[Table 1 here]

4. Illustrative Examples

The primary purpose of the illustrative examples presented in this section is to demonstrate the usefulness of our algorithm and correctness of our implementations in `R` and *Mathematica*. For this purpose, our algorithm is also compared with existing MLE algorithms.

4.1. Timings

Timings for the algorithms described in §3 were obtained in *Mathematica* and R on a Windows XP PC Pentium 4. The ARMA$(1, 1)$ model with $\phi_1 = 0.9$ and $\theta = 0.5$ was selected as typical of order $(1, 1)$ models which might occur in practice. This model was simulated 25 times for series of length $n = 10^k,\ k = 2, 3, \ldots, 6$ and the average time needed for fitting the model was determined. Timings were also compared to *HR*

(Hannan and Rissanen, 1982). The *HR* algorithm does not require non-linear optimization and only requires linear least squares and residual computation. The built-in least squares algorithms in *Mathematica* and `R` were used. The effects of initial values and MLE estimation of the mean were also examined. The initial value options also examined were *Origin*, *XInit* and *HRInit* corresponding respectively to initializing the nonlinear optimization algorithm at 0.0 for all parameter values except the mean, using exact known parameter values or using the Hannan-Rissanen estimates as initial parameter settings. The algorithms for estimating the mean, *SampleMean* and *MeanMLE*, are also compared. The *MeanMLE* refers to the algorithm in §2.2 and *SampleMean* to just using $\bar{z}$ as in §2.1. In the R timings we also compared our algorithms with the built-in `R` algorithms `arima` and `arima0`. These algorithms implement the state-space Kalman filter algorithm given in Durbin and Koopman (2001). Further details of this implementation (Ripley, 2002) indicate that this algorithm is coded in C and then interfaced to `R`.

[Table 2 about here]

Comparing *Origin* with *HR*, our algorithm is much faster for larger n. Although *HR* is faster than *Origin* for small n this is probably not important since both algorithms are very fast and *Origin* which uses the MLE method is preferred anyway – especially for small n. Since the computing time required by *HRInit* does not include the initialization times needed by *HR* itself, it is clear from Table 2, that if these are added to *HRInit*, the initialization is normally not worthwhile in terms of reducing computer time. Even with *XInit* when the exact initial values are

used, this only results in a modest improvement in speed. It is seen that in terms of speed *Mathematica* outperforms `R` except when n is very large. These timings also demonstrate that the *Mathematica* and `R` implementations of our algorithms are suitable for even very large n. Given the high-overhead imposed by the interpretive `R` language, the performance of our algorithms is not unreasonable in practice even though in most cases it is slower than `arima` and `arima0`.

4.2. Comparison with Durbin's Algorithm

The statistical efficiency of *Durbin*, the algorithm of Durbin (1959) for MA(q) estimation, is compared with *SampleMean* and exact MLE as implemented in `R` in `arima`. For each parameter value $\theta_1 = 0, \pm 0.3, \pm 0.5, \pm 0.9, \pm 1$, and for each series length $n = 50, 100, 200, 400$ one thousand time series were simulated. The empirical statistical efficiency may be taken as the empirical MSE of the exact MLE algorithm divided by the empirical MSE of *SampleMean*. Similarly, for the efficiency for the *Durbin* algorithm. The variance of the estimated efficiency may be derived using a Taylor series linearization. Details of this derivation as well as a comparison with the bootstrap variance estimate are given in our online supplement (McLeod and Zhang, 2007). In Figure 2, a trellis plot compares these efficiencies. In each plot, the vertical line running through the plotted point indicates a 95% confidence interval for that efficiency. From this plot, we see that *SampleMean* has efficiency very close to 1 except when the parameter $\theta_1 = \pm 1$ when it is less efficient and when $\theta_1 = 0.9$ it is super-efficient. In the super-efficiency cases, the efficiency approaches 1 as

n increases. The efficiency of *Durbin* is generally much less than *SampleMean* but it approaches 1 as n gets larger provided the parameter is not on the boundary.

The results shown in Figure 2 were replicated using our *Mathematica* implementation of *SampleMean* and the exact MLE algorithm for the MA (1) given in McLeod and Quenneville (2001).

[Figure 2 here]

4.3. Finite Sample Efficiency of the Sample Mean

If the parameters $\phi_1, ..., \phi_p, \theta_1, \ldots, \theta_q$ are known, the exact MLE for the mean is given by eqn. (7). It is also the best linear unbiased estimate *BLUE*. Another estimate of μ is simply the sample mean, $\bar{z} = (z_1 + \ldots + z_n)/n$. The exact efficiency for $\bar{z}$ vs. the *BLUE* for a series of length n may be written,

$$\mathcal{E} = n^2/((1_n'\Gamma 1_n)(1_n'\Gamma^{-1}1_n)). \tag{15}$$

In actual applications, the ARMA parameters are not known. In our simulation study, we compare two MLE methods for estimating the mean. The MLE methods are the *MeanMLE* algorithm of §2.2 and the `R` function `arima`. With each of these MLE methods, the empirical efficiency of $\bar{z}$ vs. the MLE estimate of μ based on 10^3 simulations for series of lengths $n = 50, 100, 200$ for the ARMA$(1, 1)$ model at each parameter setting. These empirical efficiencies are compared with the exact efficiency of $\bar{z}$ vs. *BLUE* given in eqn. (15) and all results are displayed in Table 3. Both *MeanMLE* and `arima` are closely efficient and there is general agreement

with the *BLUE* except when $\phi_1 = 0$ and $\theta_1 = 0.9, 0.95$. The simulation experiment confirms that `MeanMLE` is working correctly as expected and this was its main purpose.

Since the sample mean is asymptotically efficient in ARMA(p, q) models (Brockwell and Davis, 1991, §7.1) it would be expected the efficiencies would get closer to 1 as n increases and it is seen that in many cases this holds. However it is surprising that even for $n = 200$, some sample efficiencies are quite low for both *MeanMLE* and `arima`. This fact does not previously appear to have been observed in the ARMA case although Samarov and Taqqu (1988) found asymptotic inefficiency in a situation which we will now discuss briefly.

It should be noted that the ARMA models where the sample mean efficiency is low have an extremely high frequency spectrum. The spectral density and autocorrelation plots of the models in Table 3 are given in McLeod and Zhang (2007). The models for which the sample mean is inefficient all have strong negative autocorrelation but are better characterized in terms of the spectral density function. All models for which the sample mean efficiency is less than 10% efficient are all characterized by a high frequency spectrum in which the high frequencies are more than one hundred times the power of the low frequencies, that is, the ratio of the spectral density evaluated at the Nyquist frequency divided by the spectral density evaluated at the origin is larger than 100. This situation may be called, infrared-catastrophe since it seems unrealistic in any time series applications with actual scientific data.

Previously Samarov and Taqqu (1988) showed that asymptotically the

sample mean can be very inefficient for hyperbolic decay time series (McLeod, 1988) in the antipersistent case which corresponds to the infrared-catastrophe case for these models. In all other hyperbolic-decay cases, including the fractional ARMA case in eqn. (16), the asymptotic efficiency is above 98% (Samarov and Taqqu, 1988, Table 1).

This simulation experiment was repeated using the *SampleMean* algorithm implemented in *Mathematica* and similar results were obtained (McLeod and Zhang, 2007).

[Table 3 here]

5. Conclusion

Mathematica and `R` packages that implement the ARMA maximum likelihood algorithms described in this paper are available (McLeod and Zhang, 2007). In addition simulation scripts to obtain the results reported in this article are also available so the interested can easily reproduce and/or extend our simulation results using either *Mathematica* or `R`.

Our algorithms are suitable for use with long time series. But the principal advantage of our algorithms for maximum likelihood estimation of ARMA models is that they may easily be implemented directly in high-level QPEs. Using the `R` and *Mathematica* packages, it is relatively straightforward to implement ARMA maximum likelihood in other high level QPEs. QPEs such as MatLab and Strata as well as `R` and *Mathematica* are becoming important in teaching statistical methods so it is expected our algorithm will be useful teaching time series analysis in such computing environments.

The AR-likelihood approximation technique of this paper could be used for other types of linear time series models. It would be relatively straightforward to extend the methods of this paper to multiplicative seasonal and subset ARMA models. It may also be possible to develop an extension to the vector ARMA models case. Another interesting family of linear time series models are the fractional ARMA time series (Hipel and McLeod, 1994, Ch. 11; Brockwell and Davis, §13.2) defined by

$$\phi(\mathcal{B})\nabla^d(z_t - \mu) = \theta(\mathcal{B})a_t, \tag{16}$$

where $d \in (-0.5, 0.5)$. Figure 3 shows the Kullback-Leibler discrepancy, $\mathcal{I}$ for the case of fractionally differenced white noise, $p = 0$ and $q = 0$, with long-memory parameter $d = 0.1, 0.2, 0.3, 0.4$. When $d \in (0, 0.2)$, $r = 30$ is adequate but much higher orders may be needed for more strongly persistent time series such as when $d \geq 0.4$. In the case such strongly persistent time series our suggested AR approximation may not be useful.

[Figure 3 here]

Acknowledgements Both authors were supported by NSERC Discovery Grants. The authors would like to thank the referees for helpful comments and their careful reading of our work.

Referemces

Barnard, G.A., Jenkins, G.M. and Winston, C.B. 1962. Likelihood inference and time series. *Journal of the Royal Statistical Society, B* 125, 321–372.

Barndorff-Nielsen, O. and Schou, G., 1973. On the parametrization of autoregressive models by partial autocorrelations. *Journal of Multivariate Analysis* 3, 408–419.

Baura, G.D., 2002. *System Theory and Practical Applications of Biomedical Signals.* Wiley, New York.

Box, G.E.P., Jenkins, G.M. and Reinsel, G.C., 1994. *Time Series Analysis: Forecasting and Control*. 3rd Ed., Holden-Day, San Francisco.

Box, G.E.P. and Luceño, A., 1997. *Statistical Control by Monitoring and Feedback Adjustment*. Wiley, New York.

Brockwell, P.J. and Davis, R.A., 1991. *Time Series: Theory and Methods*. (2nd edn.) Springer-Verlag, New York.

Champernowne, D.G., 1948. Sampling theory applied to autoregressive sequences. *Journal of the Royal Statistical Society B* 10, 204–242.

Choi, B., 1992. *ARMA Model Identification*. Springer-Verlag, New York.

Durbin, J., 1959. Efficient estimation of parameters in moving-average models. *Biometrika* 46, 306–316.

Durbin, J. and Koopman, S.J., 2001. *Time Series Analysis by State Space Methods.* Oxford University Press, Oxford.

Fisher, R.A. 1973. *Statistical methods and scientific inference.* Hafner Press, New York.

Hannan, E.J. and Rissanen, J., 1982. Recursive estimation of mixed

autoregressive-moving average order. *Biometrika* 69, 81–94.

Hipel, K.W. and McLeod, A.I., 1994. *Time Series Modelling of Water Resources and Environmental Systems.* Elsevier, Amesterdam. Reprint, www `http://www.stats.uwo.ca/faculty/aim/1994Book/`.

Jones, R.H., 1980. Maximum likelihood fitting of ARMA models to time series with missing observations. *Technometrics* 22, 389–395.

McClave, E.J., 1973. On the bias of AR approximation to moving averages. *Biometrika* 60, 599-605.

McLeod, A.I., 1998. Hyperbolic decay time series. *Journal of Time Series Analysis* 19, 473–484.

McLeod, A.I. and Quenneville, B., 2001. Mean likelihood estimators. *Statistics and Computing* 11, 57–65.

McLeod, A.I. and Hipel, K.W., 2007. McLeod-Hipel Time Series Package, (www `http://www.stats.uwo.ca/faculty/aim/epubs/mhts/`).

McLeod, A.I. and Zhang, Y., 2007. Online supplements to "Faster ARMA Maximum Likelihood Estimation", www (`http://www.stats.uwo.ca/faculty/aim/2007/faster/`).

Monahan, J.F., 1984. A note on enforcing stationarity in autoregressive-moving average models. *Biometrika* 71, 403–404.

Parzen, E., 1969. Multiple time series modeling. In *Multivariate Analysis II* ed. P. Krishnaiah, 389-409. Academic Press, New York.

Percival, D.B. and Walden, A.T., 1993. *Spectral Analysis for Physical Applications*. Cambridge University Press, Cambridge.

Powell, M.J.D., 1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer*

Journal 7, 155–162.

Ripley, B.D., 2002. Time Series in R. *R News* 2, 2–7.

Samarov, A. and Taqqu, M., 1988. On the efficiency of the sample mean in long memory noise. *Journal of Time Series Analysis* 9, 191–200.

Siddiqui, M.M., 1958. On the inversion of the sample covariance matrix in a stationary autoregressive process. *Annals of Mathematical Statistics* 29, 585–588.

Taniguchi, M., 1983. On the second order asymptotic efficiency of estimators of gaussian ARMA processes. *The Annals of Statistics* 11, 157–169.

Ullah, A., 2002. Use of entropy and divergence measures for evaluating econometric approximations and inference. *Journal of Econometrics* 107, 313–326.

Wahlberg, B., 1989. Estimation of autoregressive moving-average models via high-order autoregressive approximations. *Journal of Time Series Analysis* 10, 283–299.

Zhang, Y. and McLeod, A.I., 2006a. Fitting MA(q) models in the closed invertible region. *Statistics and Probablity Letters* 76, 1331–1334.

Zhang, Y. and McLeod, A.I., 2006b. Computer algebra derivation of the bias of Burg estimators. *Journal of Time Series Analysis* 27, 157–165.

Zinde-Walsh, V., 1988. Some exact formulae for autoregressive moving average processes. *Econometric Theory* 4, 384–402.

Table 1: Kullback-Leibler discrepancy for AR(r) approximation to a MA (1) with $\theta_1 = 0.95$ and $n = 200$ using the MMSE approximation and the approximation based on truncating the inverted form of the model.

n	MMSE	Truncated
10	4.99	31.20
20	1.23	10.70
30	0.38	3.77
40	0.12	1.42
50	0.04	0.62

Table 2: Average CPU time in seconds with `R` and *Mathematica* for fitting the ARMA$(1,1)$ model with $\phi_1 = 0.9$ and $\theta = 0.5$ using *SampleMean*, *MeanMLE* and the Hannan-Rissanen estimator. In the `R` case, built-in functions `arima` and `arima0` are also used. Twenty-five replications for series of length $n = 10^k,\ k = 2, 3, \ldots, 6$ were done. The case where the mean is estimated by the sample average is compared with the MLE for each algorithm. The effect of initial parameter settings is also examined. The settings *Origin*, *XInit* and *HRInit* correspond to setting (ϕ_1, θ_1) equal to $(0, 0)$, $(0.9, 0.5)$ or using the estimator of Hannan-Rissanen respectively.

	n				
method	10^2	10^3	10^4	10^5	10^6
	Timings in `R`				
	SampleMean				
Origin	0.47	0.47	0.70	2.13	8.63
HR	0.24	1.05	10.2	92.0	902.
XInit	0.32	0.27	0.46	1.40	6.81
HRInit	0.29	0.27	0.57	1.70	7.78
arima	0.02	0.04	0.24	1.85	13.6
arima0	0.01	0.01	0.02	0.18	1.78
	MeanMLE				
Origin	1.00	0.85	1.03	3.14	16.70
XInit	0.90	0.68	0.82	2.80	17.15
HRInit	0.80	0.58	0.89	2.91	16.59
arima	0.05	0.19	0.74	3.94	32.54
arima0	0.03	0.03	0.07	0.88	13.62
	Timings in *Mathematica*				
	SampleMean				
Origin	0.26	0.29	0.36	0.90	4.97
HR	0.01	0.05	0.54	5.00	47.4
XInit	0.27	0.30	0.36	0.89	4.96
HRInit	0.27	0.29	0.35	0.88	4.99
	MeanMLE				
Origin	0.75	0.89	1.14	4.10	30.46
XInit	0.78	0.90	1.14	4.11	30.71
HRInit	0.75	0.89	1.10	4.10	30.49

Table 3: Empirical efficiency of the sample mean vs. three other methods: BLUE, *MeanMLE* and `arima`. Each empirical efficiency is based on 1000 simulations for ARMA$(1,1)$ models with $n = 50, 100, 200$.

			θ_1						
ϕ_1	algorithm	n	-0.95	-0.9	-0.5	0.	0.5	0.9	0.95
-0.95	*BLUE*	50	1.00	1.00	0.97	0.75	0.25	0.01	0.01
-0.95	*MeanMLE*	50	1.00	1.00	0.99	0.74	0.39	0.02	0.01
-0.95	`arima`	50	1.02	1.02	1.00	0.74	0.40	0.02	0.01
-0.95	*BLUE*	100	1.00	1.00	0.98	0.85	0.38	0.02	0.01
-0.95	*MeanMLE*	100	1.00	1.00	1.00	0.87	0.58	0.04	0.01
-0.95	`arima`	100	1.01	1.01	1.00	0.87	0.58	0.04	0.01
-0.95	*BLUE*	200	1.00	1.00	0.99	0.92	0.54	0.03	0.01
-0.95	*MeanMLE*	200	1.00	1.00	0.99	0.91	0.67	0.06	0.02
-0.95	`arima`	200	1.01	1.00	0.99	0.91	0.67	0.06	0.02
-0.9	*BLUE*	50	1.00	1.00	0.99	0.86	0.39	0.02	0.01
-0.9	*MeanMLE*	50	1.00	1.01	1.00	0.85	0.59	0.05	0.02
-0.9	`arima`	50	1.02	1.01	1.01	0.86	0.59	0.05	0.02
-0.9	*BLUE*	100	1.00	1.00	0.99	0.92	0.56	0.03	0.01
-0.9	*MeanMLE*	100	1.00	1.00	1.00	0.95	0.74	0.07	0.02
-0.9	`arima`	100	1.01	1.01	1.00	0.95	0.74	0.07	0.02
-0.9	*BLUE*	200	1.00	1.00	1.00	0.96	0.71	0.06	0.02
-0.9	*MeanMLE*	200	1.00	1.00	1.00	0.95	0.81	0.11	0.03
-0.9	`arima`	200	1.01	1.01	1.00	0.95	0.81	0.12	0.03
-0.5	*BLUE*	50	1.00	1.00	1.00	0.99	0.83	0.13	0.06
-0.5	*MeanMLE*	50	1.00	1.00	1.01	1.00	0.96	0.26	0.13
-0.5	`arima`	50	1.00	1.00	1.03	1.00	0.96	0.26	0.13
-0.5	*BLUE*	100	1.00	1.00	1.00	0.99	0.91	0.19	0.07
-0.5	*MeanMLE*	100	1.00	1.00	1.00	1.00	0.95	0.31	0.13
-0.5	`arima`	100	1.00	1.00	1.01	1.00	0.95	0.32	0.13
-0.5	*BLUE*	200	1.00	1.00	1.00	1.00	0.95	0.30	0.10
-0.5	*MeanMLE*	200	1.00	1.00	1.00	1.00	0.96	0.43	0.17
-0.5	`arima`	200	1.00	1.00	1.00	1.00	0.96	0.44	0.17

			θ_1						
ϕ_1	algorithm	n	-0.95	-0.9	-0.5	0.	0.5	0.9	0.95
0.	*BLUE*	50	0.99	0.99	1.00	1.00	0.96	0.34	0.18
0.	*MeanMLE*	50	1.02	1.02	1.02	1.03	1.03	1.03	1.03
0.	`arima`	50	1.03	1.03	1.03	1.02	1.03	1.03	1.03
0.	*BLUE*	100	1.00	1.00	1.00	1.00	0.98	0.44	0.19
0.	*MeanMLE*	100	1.01	1.01	1.01	1.00	1.01	1.01	1.01
0.	`arima`	100	1.01	1.01	1.01	1.01	1.01	1.01	1.01
0.	*BLUE*	200	1.00	1.00	1.00	1.00	0.99	0.58	0.26
0.	*MeanMLE*	200	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.	`arima`	200	1.00	1.00	1.00	1.00	1.00	1.00	1.00
0.5	*BLUE*	50	0.97	0.97	0.98	0.99	1.00	0.67	0.44
0.5	*MeanMLE*	50	0.94	0.94	0.96	1.00	1.03	0.71	0.50
0.5	`arima`	50	0.94	0.94	0.96	1.00	1.03	0.74	0.55
0.5	*BLUE*	100	0.99	0.99	0.99	0.99	1.00	0.75	0.44
0.5	*MeanMLE*	100	0.96	0.96	0.97	0.99	1.02	0.72	0.43
0.5	`arima`	100	0.96	0.96	0.97	0.99	1.02	0.74	0.44
0.5	*BLUE*	200	0.99	0.99	0.99	1.00	1.00	0.84	0.54
0.5	*MeanMLE*	200	0.98	0.98	0.99	1.00	1.01	0.82	0.55
0.5	`arima`	200	0.98	0.98	0.99	1.00	1.01	0.84	0.56
0.9	*BLUE*	50	0.89	0.89	0.90	0.91	0.93	1.00	0.97
0.9	*MeanMLE*	50	0.81	0.81	0.82	0.94	0.91	1.05	1.03
0.9	`arima`	50	0.80	0.80	0.81	1.58	0.91	1.08	1.02
0.9	*BLUE*	100	0.93	0.93	0.93	0.94	0.95	1.00	0.96
0.9	*MeanMLE*	100	0.85	0.85	0.86	0.93	0.93	1.10	1.06
0.9	`arima`	100	0.85	0.85	0.86	0.93	0.93	1.09	1.04
0.9	*BLUE*	200	0.96	0.96	0.96	0.96	0.97	1.00	0.96
0.9	*MeanMLE*	200	0.92	0.92	0.92	0.95	0.96	1.06	0.96
0.9	`arima`	200	0.92	0.92	0.92	0.95	0.96	1.04	0.97
0.95	*BLUE*	50	0.88	0.88	0.88	0.89	0.91	0.99	1.00
0.95	*MeanMLE*	50	0.78	0.78	0.79	0.91	0.86	1.02	1.03
0.95	`arima`	50	0.77	0.77	0.79	0.91	0.91	1.01	1.01
0.95	*BLUE*	100	0.89	0.89	0.89	0.90	0.91	0.98	1.00
0.95	*MeanMLE*	100	0.80	0.80	0.81	0.89	0.87	1.08	1.18
0.95	`arima`	100	0.80	0.80	0.81	0.89	0.87	1.08	1.14
0.95	*BLUE*	200	0.93	0.93	0.93	0.93	0.94	0.98	1.00
0.95	*MeanMLE*	200	0.87	0.87	0.87	0.92	0.91	1.11	1.32
0.95	`arima`	200	0.87	0.87	0.87	0.92	0.91	1.13	1.29

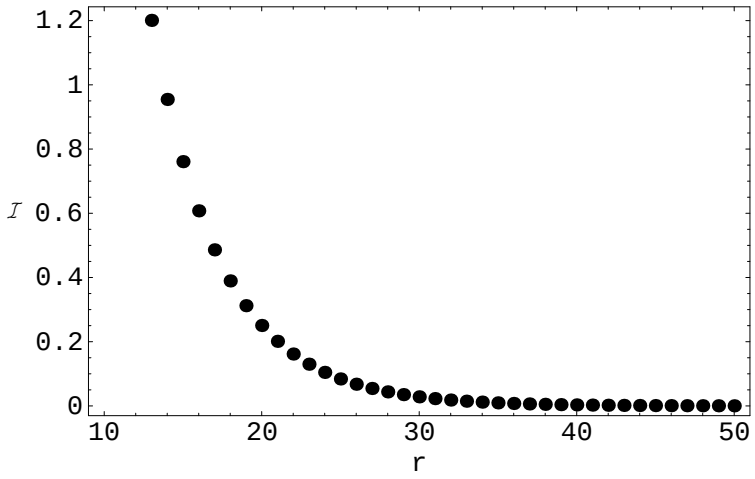

Figure 1: Kullback-Leibler discrepancy for AR(r) approximation to a MA (1) with $\theta_1 = 0.9$ and $n = 200$.

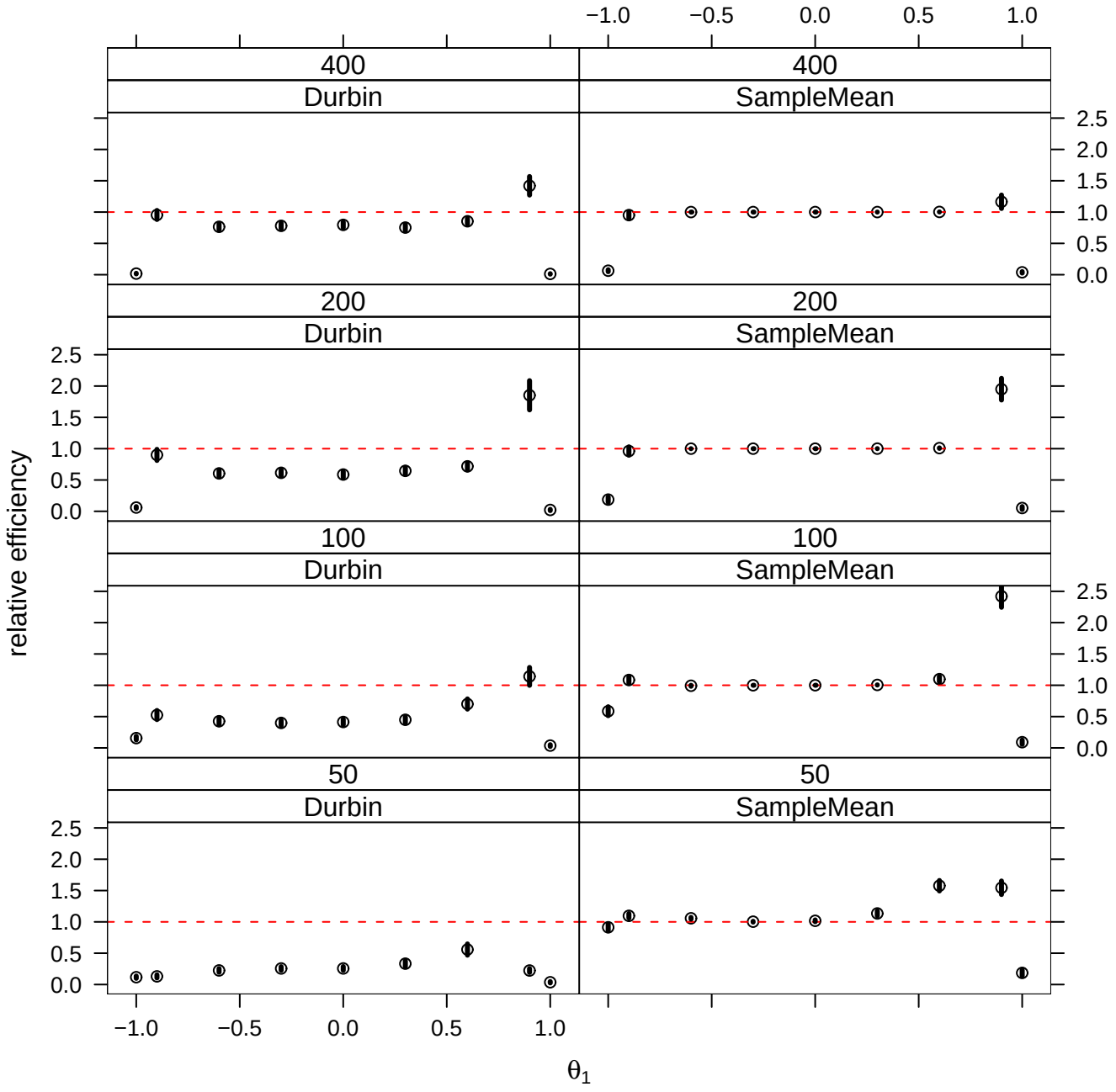

Figure 2: The vertical lines show the length of the 95% confidence interval for the statistical efficiency of *SampleMean* and *Durbin* vs. the MLE based on 10^3 simulations.

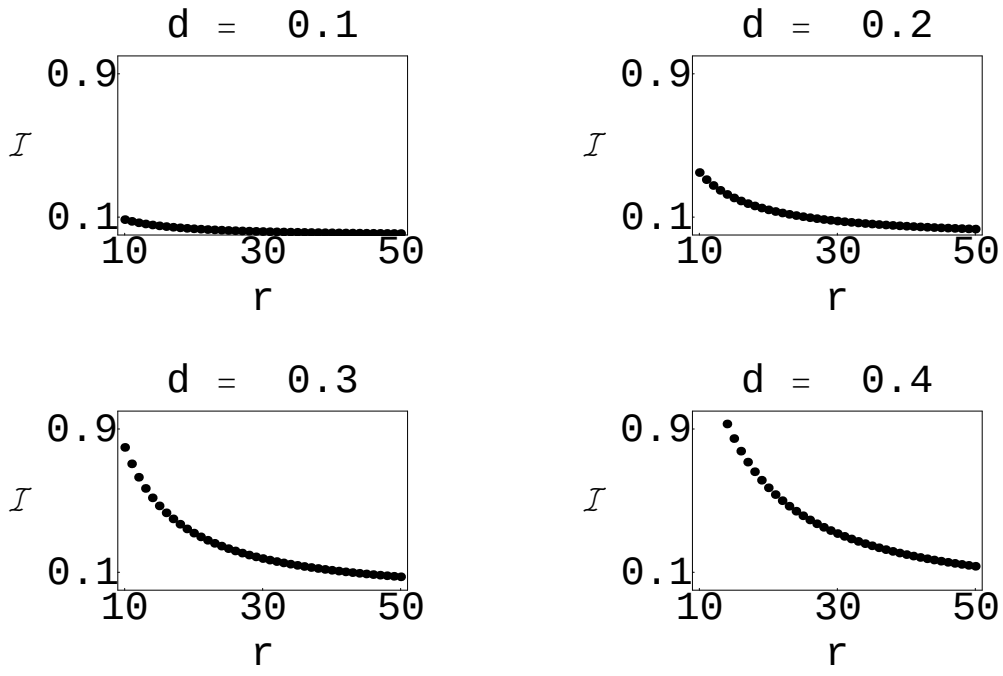

Figure 3: Kullback-Leibler discrepancy for AR(r) approximation to fractionally differenced white noise, $\nabla^d z_t = a_t$ for $d = 0.1, 0.2, 0.3, 0.4$ and $n = 200$.