On the Identification of Autoregressive Time Series with Measurement Error by Repetitive Autocorrelation

Jan Kybic, (xkybic@sun.felk.cvut.cz)

Abstract — The problem of identification of a stochastic linear dynamic system is an important one in the fields of engineering, statistics etc. Numerous methods exists for estimating autoregressive model parameters. However, if a measurement noise is not negligible, the simple and fast methods often yield very inefficient estimates while better methods, such as those based on maximum likelihood estimate are often too slow to be used on-line. One of the classical methods uses the autocorrelation function. In this paper we demonstrate that by using this method repetitively better estimates can be obtained with relatively small computational complexity permitting on-line applications. This new approach is compared with the ML-estimate.

1. Statement of the problem

A discrete stochastic linear dynamic system with no inputs can be represented as

$$x(t) = \sum_{i=1}^{n} a_i x(t-i) + u(t) \tag{1}$$

$$y(t) = x(t) + v(t) \tag{2}$$

where u and v are uncorrelated Gaussian white noises with zero means and variances q and r respectively. Given the order of the system n and finite length sample $\{y(t)\}_{t=1}^{N} = \{y(1), y(2), \ldots, y(N)\}$, it is required to determine the coefficients $\boldsymbol{a} = [a_1, \ldots, a_n]$ assuming that the output data $\{y(t)\}$ come from a system described by equations (1) and (2) after it has reached a steady state, i.e. $y(t)$ is stationary.

2. Negligible measurement noise case

Assuming negligible measurement noise case, i.e. exact knowledge of $\{x(t)\}_{t=1}^{N}$ we can use the following traditional method to estimate $\boldsymbol{a}$. Taking the equation (1) for $t = n+1, n+2, \ldots N$ we get a set of linear equations that can be symbolically written as follows:

$$\boldsymbol{x} = \boldsymbol{Z}\boldsymbol{a} + \boldsymbol{u} \tag{3}$$

where $\boldsymbol{x} = [x(n+1), \ldots, x(N)]$, $[\boldsymbol{Z}]_{ij} = x(n-i-j+2)$ and $\boldsymbol{u} = [u(n+1), \ldots, u(N)]$.

According to [1], reasonable estimate of $\boldsymbol{a}$ is

$$\hat{\boldsymbol{a}} = \mathcal{E}\left\{\boldsymbol{a}|x(1),\ldots,x(N)\right\} = (\boldsymbol{Z}^T\boldsymbol{Z})^{-1}\boldsymbol{Z}^T\boldsymbol{x} \tag{4}$$

You may note that the estimate (4) is both the MS (mean square) and ML (maximum likelihood) estimate.

3. Measurement noise effect

Let us now consider the effects of non-negligible measurement noise v on the results obtained from (4). The known output y can be used as an estimate of x which changes to $y(t) = x(t) + \mathrm{d}x(t)$. The equations (3) thus become:

$$\boldsymbol{x} + \mathrm{d}\boldsymbol{x} = (\boldsymbol{Z} + \mathrm{d}\boldsymbol{Z})\boldsymbol{a} + \boldsymbol{u} \tag{5}$$

Solving (5) using (4) we get

$$\hat{\boldsymbol{a}} = \left((\boldsymbol{Z} + \mathrm{d}\boldsymbol{Z})^T(\boldsymbol{Z} + \mathrm{d}\boldsymbol{Z})\right)^{-1}(\boldsymbol{Z} + \mathrm{d}\boldsymbol{Z})^T(\boldsymbol{x} + \mathrm{d}\boldsymbol{x}) \tag{6}$$

Taking expectations and using the fact that u and v are not correlated

$$\bar{\boldsymbol{a}} = \mathcal{E}\left\{\hat{\boldsymbol{a}}\right\} = \left(\boldsymbol{Z}^T\boldsymbol{Z} + \mathrm{d}\boldsymbol{Z}^T\mathrm{d}\boldsymbol{Z}\right)^{-1}\boldsymbol{Z}^T\boldsymbol{x}$$

Furthermore, using the matrix inversion lemma and assuming that $\mathcal{E}\left\{\mathrm{d}\boldsymbol{Z}^T\mathrm{d}\boldsymbol{Z}\right\} = r\boldsymbol{I}$ is small in comparison to $\mathcal{E}\left\{\boldsymbol{Z}^T\boldsymbol{Z}\right\}$ we arrive to the following approximate formula:

$$\bar{\boldsymbol{a}} \approx (\boldsymbol{Z}^T\boldsymbol{Z})^{-1}\left(\boldsymbol{I} - r(\boldsymbol{Z}^T\boldsymbol{Z})^{-1}\right)\boldsymbol{Z}^T\boldsymbol{x} \tag{7}$$

Realizing that $\mathcal{E}\left\{[\boldsymbol{Z}^T\boldsymbol{Z}]_{ij}\right\} = C_x(i-j)$, where

$$C_x(k) = \mathcal{E}\left\{x(t)x(t+k)\right\} \tag{8}$$

is the autocorrelation of x and $C_x(k) = C_x(-k) \leq C_x(0)$ (see [2]) we may conclude that the relative error of $\bar{\boldsymbol{a}}$ is approximately proportional to the ratio of the measurement noise variance r and the original signal variance $C_x(0)$. It is stated in [3] that this conclusion is true even when $r\boldsymbol{I}$ is not small in comparison to $\boldsymbol{Z}^T\boldsymbol{Z}$.

As $\hat{\boldsymbol{a}} - \boldsymbol{a} \sim \mathrm{d}\boldsymbol{x}$ (see [3]) and therefore

$$\mathrm{cov}(\boldsymbol{a}) \sim r\boldsymbol{I} \tag{9}$$

Combining (9) with the previous conclusion we finally arrive to the result that the mean square error of the estimate is still approximately proportional to $(r/C_x(0))^2$.

$$\frac{\mathcal{E}\left\{(\hat{\boldsymbol{a}} - \boldsymbol{a})^2\right\}_i}{a_i^2} \sim (r/C_x(0))^2 \tag{10}$$

4. Using autocorrelation

Let us take the autocorrelation C_y of the measured signal y.

$$C_y(k) = \mathcal{E}\left\{y(t)y(t+k)\right\} = \mathcal{E}\left\{(x(t)+v(t))\,(x(t+k)+v(t+k))\right\} = C_x(k) + C_v(k) \tag{11}$$

where we have used the fact that x and v are uncorrelated. As v is a white noise, $C_v(0) = r$ and $C_v(k) = 0$ for $k \neq 0$ it holds

$$C_y(k) = C_x(k) \quad \text{for} \quad k \neq 0 \tag{12}$$

An expression for $C_x(k)$ can be obtained from (8) and (1)

$$C_x(k) = \mathcal{E}\left\{x(t)\left(\sum_{i=1}^{n} a_i x(t+k-i) + u(t+k)\right)\right\} = \sum_{i=1}^{n} a_i C_x(k-i) \quad \text{for} \quad k > n \tag{13}$$

using the fact that $u(t)$ is not correlated with $x(t_1)$, $t_1 < t$. The same holds for C_y, see (12). For generalized version of (13) (called Yule-Walker equation) see [3].

The crucial observation is that the equation (13) is almost identical to (1), i.e. the sequence $\{C_x(k)\}$ (and $\{C_x(k)\}$) can be regarded as a product of an auto-regressive process with no input. Consequently we can use the same procedure as described in (3) and (4) to get the true values of $\boldsymbol{a}$ from C_x or C_y.

5. Autocorrelation estimation

In the preceding section we have shown how to obtain $\boldsymbol{a}$ from C_y. Unfortunately, only one sample of y is given and C_y can therefore only be estimated using the ergodic property of y. According to [3], reasonable estimate is

$$\hat{C}_y(k) = \frac{1}{N}\sum_{t=|k|+1}^{N} y(t)y(t-|k|) \tag{14}$$

The equation (14) can be also regarded as describing an operator, converting the sequence $\{y(t)\}_{k=1}^{N}$ to the sequence $\{\hat{C}_y(k)\}_{k=0}^{N-1}$. We shall skip the $\hat{C}_y(0)$ term, for it is of little use for us for (see (12)) and we shall denote the resulting operator Ψ, i.e.

$$\{\hat{C}_y(k)\}_{k=1}^{N-1} = \Psi\left(\{y(t)\}_{k=1}^{N}\right) \tag{15}$$

The estimate (14) is biased for finite sample sizes,

$$\mathcal{E}\left\{\hat{C}_y(k)\right\} = (1-k/N)C_y(k) \tag{16}$$

however, it has been shown in [4] that the estimate (14) is preferable since it gives less mean-square error than the corresponding unbiased estimate. Heuristically speaking, the values of $\hat{C}_y(k)$ are less accurate for bigger k and it is therefore beneficial that they are smaller for they have less influence on the solution of (3). We shall see later how to compensate the systematic error introduced by the bias.

It can be shown [5] that (14) is an asymptotically unbiased, normal, and consistent estimate of C_y. An expression for the asymptotic covariance of $\hat{C}_y(k)$ has been given by Bartlett [6] (simplified):

$$\mathrm{cov}(\hat{C}_y(k), \hat{C}_y(l)) \approx \begin{cases} 0, & k \neq l \\ (1/N) C_y(0)^2, & k = l \neq 0 \\ C_y(0)^2, & k = l = 0 \end{cases} \tag{17}$$

6. Bias compensation

As we already mentioned, the estimate (14) is biased and therefore taking the set of equations (13) and substituting $\hat{C}_y$ for C_y, i.e.

$$\hat{C}_x(k) = \sum_{i=1}^{n} a_i \hat{C}_x(k-i) \quad \text{for} \quad k = n+1, n+2, \ldots, N-1 \tag{18}$$

and solving (18) for $\boldsymbol{a}$ yields a biased result. We know that $\mathcal{E}\left\{\hat{C}_y(k)\right\} = w(k) C_y(k)$ where the weights $w(k) = (1 - k/N)$. Note that

$$w(k) = \Psi([\underbrace{1, 1, \ldots, 1}_{N}]) \tag{19}$$

In order to get an unbiased $\boldsymbol{a}$, we change (18) to

$$\hat{C}_x(k) = \sum_{i=1}^{n} a_i \hat{C}_x(k-i) \frac{w(k)}{w(k-i)} \quad \text{for} \quad k = n+1, n+2, \ldots, N-1 \tag{20}$$

In (20) we have removed the bias from all equations while retaining the value of the left side thus preserving the beneficial property of (13) where all the left sides have the same variance (see (17)) that makes the least-squares solution estimate (4) applied to (20) viable.

7. Accuracy improvement

Let us compare the accuracy of the direct estimate of $\boldsymbol{a}$ through (6) and the autocorrelation based estimate using (14) and (20).

Using (10) we have concluded that the relative error of the direct estimate of $\boldsymbol{a}$ using (6) is

$$e_0 \approx \frac{r}{C_x(0)} \tag{21}$$

.

Using (17) and (11) we see that

$$\mathcal{E}\left\{C_y(k)^2\right\} = C_x(k)^2 \tag{22}$$

and

$$\operatorname{cov}(\hat{C}_y) = \frac{(C_x(0)+r)^2}{N} \tag{23}$$

This gives us (exploiting the similarity with the direct estimate case) the relative error

$$e_1 \approx \frac{1}{N}\frac{(C_x(0)+r)^2}{C_x(0)^2} \tag{24}$$

We see that the autocorrelation (indirect) estimate is better than the direct estimate if $r < N C_x(0)$ (approximately). Comparing (21) and (24) and approximating for small r we can expect

$$\frac{e_0}{e_1} = \frac{r}{C_x(0)}\frac{N C_x(0)^2}{(C_x(0)+r)^2} \tag{25}$$

$$\frac{e_0}{e_1} \approx \frac{rN}{C_x(0)} \tag{26}$$

i.e. the relative error to decrease by a factor of $C_x(0)/(rN)$

8. Repetitive autocorrelation

Let us reformulate and summarize preceding sections. We have shown that to estimate the $\boldsymbol{a}_0$ parameters of an unknown system A_0 described by (1,2) from the measurement $\{y_0(t)\}$ is is beneficial to apply onto $\{y_0(t)\}$ the autocorrelation operator Ψ, defined by (15), because the resulting sequence $\{y_1(t)\} = \{\hat{C}_y(k)\} = \Psi(\{y_0(t)\})$ can (neglecting the bias (16) for a while for the sake of explanation) be regarded as an output from some system A_1 described also by (1,2) with the same parameters $\boldsymbol{a}_1 = \boldsymbol{a}_0$ as A_0 but with smaller measurement noise. Consequently the estimate $\hat{\boldsymbol{a}}_1$ from $\{y_1(t)\}$ is more accurate than the estimate $\hat{\boldsymbol{a}}_0$ from $\{y_0(t)\}$.

Straightforward extension of the procedure described above is to apply the operator Ψ repetitively

$$\{y_1(t)\} = \Psi(\{y_0(t)\}), \quad \{y_2(t)\} = \Psi(\{y_1(t)\}), \quad \ldots \tag{27}$$

We expect the corresponding estimates

$$\hat{\boldsymbol{a}}_0, \hat{\boldsymbol{a}}_1, \hat{\boldsymbol{a}}_2, \ldots \tag{28}$$

to have decreasing covariances (increasing accuracy).

To get unbiased $\hat{\boldsymbol{a}}_i$ we can use (20) reformulated as

$$y_i(k) = \sum_{l=1}^{n} a_i y_i(k-l)\frac{w_i(k)}{w_i(k-l)} \quad \text{for} \quad k = n+1, n+2, \ldots, N-i \tag{29}$$

It remains to calculate the weights $w_i(k)$ defined by

$$\mathcal{E}\left\{y_i(k)\right\} = w_i(k) y_i^*(k) \tag{30}$$

where $y_i^*(k)$ are the true unbiased values described by

$$y_0^*(k) = y_0(k) \quad \text{and} \quad y_{i+1}^*(k) = \mathcal{E}\left\{y_i^*(t) y_i^*(t+k)\right\} \tag{31}$$

Substituting into (14) we get

$$\begin{aligned}
\mathcal{E}\left\{y_{i+1}(k)\right\} &= \mathcal{E}\left\{\frac{1}{N-i}\sum_{t=|k|+1}^{N-i} y_i(t) y_i(t-|k|)\right\} = \\
&= \mathcal{E}\left\{\frac{1}{N-i}\sum_{t=|k|+1}^{N-i} w_i(t) y_i^*(t) w_i(t-|k|) y_i^*(t-|k|)\right\} = \\
&= \mathcal{E}\left\{y_i^*(t) y_i^*(t-|k|)\right\} \frac{1}{N-i}\sum_{t=|k|+1}^{N-i} w_i(t) w_i(t-|k|) = \\
&= y_{i+1}^*(k) \frac{1}{N-i}\sum_{t=|k|+1}^{N-i} w_i(t) w_i(t-|k|)
\end{aligned} \tag{32}$$

Comparing (32) and (30) we conclude that

$$w_{i+1}(k) = \frac{1}{N-i}\sum_{t=|k|+1}^{N-i} w_i(t) w_i(t-|k|) \tag{33}$$

which can be viewed as an extension of (19), i.e.

$$w_1(k) = [\underbrace{1, 1, \ldots, 1}_{N}], \quad w_2(k) = \Psi\left(w_1(k)\right), \quad w_3(k) = \Psi\left(w_2(k)\right), \quad \ldots \tag{34}$$

9. Practical considerations

As the value of the autocorrelation coefficients gets squared in each iteration, it is advisable to normalize them to avoid numerical overflow or underflow.

Using the approximate formula (26) we see that the accuracy of estimates $\hat{\boldsymbol{a}}_i$ increases steadily with i. Analyzing (25) we find that the relative error is convergent to $1/N$. Experiments show that the estimates reach this optimum within 5 to 10 iterations and then slowly deteriorate, probably due to numerical errors.

To speed up the calculation for larger N at the expense of the accuracy, we can calculate just first $M < N$ members of $\{y_1(t)\}_{t=1}^{M}$. M must be at least $2n$ and $M \approx 100n$ is reasonable.

Although the algorithm is described in its off-line form, recursive version suitable for on-line processing is readily obtainable. Mehra ([3]) shows how to get the recursive formulas.

10. Implementation

In this section we shall present the MATLAB source of the above mentioned algorithm. First an auxiliary function to calculate the autocorrelation coefficients:

```
function c=corrn(y,n)
% Computes the first n coefficients of the autocorrelation
% function of the signal y
% c(1) is the correllation at lag 0, c(2) at lag 1 etc.

N=length(y) ;
c=zeros(n,1) ;

for i=1:n,
  c(i)=sum(y(1:N-i+1).*y(i:N))/N ;
end ;
```

Now the main function to calculate the coefficients.

```
function a=clhamre(x,n)
% Given a signal x, this function estimates the
% a parameters of an ARX model of the order n
% x must be at least 2n elements long.
% This routine uses more than 2n autocorr. coefs and does repetitive
% autocorrelation to improve efficiency.
% Usage: a=clhamre(x,n)

% The length of the input vector
T=length(x) ;
m=100*n ;  % We shall use this many coefficients
m=min(m,T) ;

if (2*n>T), error('The signal is too short.') ; end ;

w=ones(m,1) ; % initial weights
xc=x ;

iter=7 ; % Number of iterations
for j=1:iter,
 xc=corrn(xc,m+1-j) ; % xc(1)
 xc=xc(2:m+1-j) ; xc=xc/max(abs(xc)) ; % scaling seems necessary
 w=corrn(w,m+1-j) ;
 w=w(2:m+1-j) ; w=w/max(w) ;
 end ; % of for j

% Arrange the coefficients into a Toeplitz matrix
mh=length(xc)-n ; % the number of equations
C=zeros(mh,n) ;
```

```
for i=1:mh,
     C(i,:)=( xc(i:i+n-1) ./ (w(i:i+n-1) / w(i+n)) )' ;
end ;

% Compute the ARX model parameters
a=(C\xc(n+1:mh+n))' ;

% Revert the a-coefficients into the usual order
a=a(n:-1:1) ;
```

11. Maximum likelihood estimate

To evaluate the performance of the repetitive autocorrelation method described above we shall compare it to the maximum likelihood (ML) estimate. The ML estimate is defined as follows:

$$\hat{\boldsymbol{a}}_{\mathrm{ML}} = \arg\max_{\boldsymbol{a}} p\left(a \mid \{y(t)\}_{t=1}^{N}\right) \tag{35}$$

Remark: We originally wanted to use the *pem* function from MATLAB. However it proved to be unusable for our purposes (system with no input of relatively high order, high noise conditions). Depending on the settings it either did not attain the convergence or stopped iterating even though the objective function clearly could be improved. We therefore had to implement the ML estimator ourselves.

It is known ([7]) that for the white noise case (35) can be alternatively stated as a PEM (prediction error method) estimate

$$\hat{\boldsymbol{a}}_{\mathrm{ML}} = \arg\min_{\boldsymbol{a}} J \quad J = \sum_{t=1}^{N} \left(y(t) - \hat{y}(t)\right)^2 \tag{36}$$

where $\hat{y}(t)$ is the optimal prediction.

As $y(t)$ is stationary and (1,(2)) is linear the optimal predictor is a steady-state Kalman filter ([1]). To apply it, we use the following canonical representation of the system (1,(2)):

$$z(t) = [x(t-n+1), x(t-n+2), \ldots, x(t-1), x(t)]^T \tag{37}$$

$$
\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ a_n & a_{n-1} & a_{n-2} & \dots & a_1 \end{bmatrix} \tag{38}
$$

$$
\boldsymbol{z}(t) = \boldsymbol{A}\boldsymbol{z}(t-1) + \boldsymbol{h}u(t), \quad \boldsymbol{h} = [\underbrace{0, 0, \dots, 0, 1}_{n}]^T \tag{39}
$$

$$
y(t) = \boldsymbol{h}^T \boldsymbol{z}(t) + v(t) \tag{40}
$$

The filtering equations are:

$$
\hat{\boldsymbol{z}}(t+1 \mid t) = \boldsymbol{A} \left(\hat{\boldsymbol{z}}(t \mid t-1) + \boldsymbol{k} \left(y(t) - \boldsymbol{h}^T \hat{\boldsymbol{z}}(t \mid t-1) \right) \right) \tag{41}
$$

$$
\hat{y}(t) = \boldsymbol{h}^T \hat{\boldsymbol{z}}(t \mid t-1) \tag{42}
$$

The optimal steady-state Kalman gain is given by

$$
\boldsymbol{k} = \boldsymbol{A}\boldsymbol{h}^T \left(\boldsymbol{h}^T \boldsymbol{P} \boldsymbol{h} + r \right)^{-1} \tag{43}
$$

where $\boldsymbol{P} = \mathrm{cov}(\boldsymbol{z})$ is a solution of the following equation

$$
\boldsymbol{P} = \boldsymbol{A} \left(\boldsymbol{P} - \boldsymbol{P}\boldsymbol{h} \left(\boldsymbol{h}^T \boldsymbol{P} \boldsymbol{h} + r \right)^{-1} \boldsymbol{h}^T \boldsymbol{P} \right) \boldsymbol{A}^T + \boldsymbol{h}^T q \boldsymbol{h} \tag{44}
$$

Analyzing the preceding equations we find that for given $\{y(t)\}_{t=1}^{N}$ the loss function J depends only on $\boldsymbol{a}$, q/r and $\boldsymbol{z}(1 \mid 0)$. The algorithm to find $\hat{\boldsymbol{a}}_{\mathrm{ML}}$ can thus be described as

$$
\hat{\boldsymbol{a}}_{\mathrm{ML}} = \arg \min_{\boldsymbol{a}} \min_{q/r} \min_{\hat{\boldsymbol{z}}(1|0)} J \tag{45}
$$

From (41,42) we observe that $\hat{\boldsymbol{z}}(t \mid t-1)$ and $\hat{\boldsymbol{y}}(t)$ depends linearly on $\boldsymbol{z}(1 \mid 0)$ and thus the innermost minimization in (45) is essentially a least squares solution of a set of linear equations, we used SVD to solve it. For minimization with respect to q/r Newton method was used. For the outermost minimization with respect to $\boldsymbol{a}$ we finally used adaptive simplex search as the higher order methods did prove to be robust enough.

Generally, the above described method of ML estimate is inherently very slow and is definitely not suitable for on-line usage.

MATLAB implementation follows:

```
function A=ar2tm(a) ;
% ar2tm takes the a coefficients of an AR model and creates
% the appropriate state transfer matrix A
% Usage: A=ar2tm(a) ;

n=length(a) ;
```

```
A=zeros(n) ;
A(n,:)=a(n:-1:1) ;

for i=1:n-1, A(i,i+1)=1 ; end ;
```

```
function a=mlaroe(y,n)
% Given a signal y, this function computes ML estimate of the parameters
% Usage: a=mlaroe(y,n)

a=clhamre(y,n) ; % Starting estimate
a=fmins('mlaroefn2',a,[1],[],y) ;
```

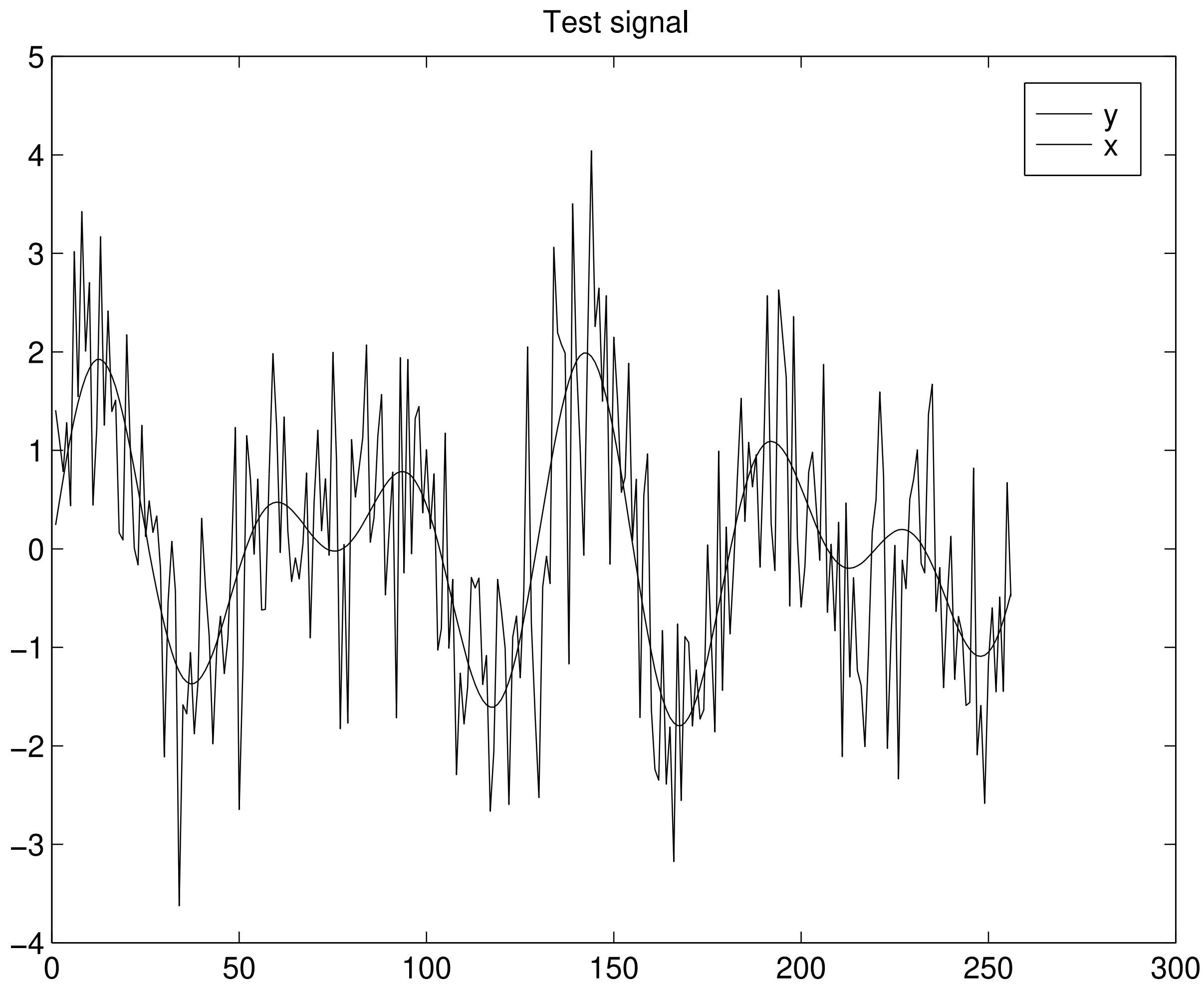
```
function e=mlaroefn2(a,z)

% Minimize with respect to r, q is fixed to 1
r=fmin('mlaroefn1',-10,10,foptions,a,z) ;
e=mlaroefn1(r,a,z) ;


function e=mlaroefn1(r,a,z)
% Compute the prediction error function
% r is log10 of the relative output measurement error dispersion
% Usage e=mlaroefn1(r,a,z)

n=length(a) ;
T=length(z) ;
%q=a(n+1) ;
a=a(1:n) ;
A=ar2tm(a) ;
q=1 ; r=10^r ;
h=[ zeros(1,n-1) 1] ; % output vector h
   P=eye(n) ;
   k=P(:,n)/(P(n,n)+r) ;
   kold=k+ones(n,1) ;
   iter=0 ; maxiter=2000 ;
   while max(abs(k-kold))>1e-6 & iter<maxiter,
      Pold=P ; iter=iter+1 ;
%     P=A*(P-P*h'*inv(h*P*h'+r)*h*P)*A'+h*q*h' ;
      P=A*(P-P(:,n)/(P(n,n)+r)*P(n,:))*A' ;
      P(n,n)=P(n,n)+q ;
      kold=k ; k=P(:,n)/(P(n,n)+r) ;
      end ;
if iter>=maxiter,
        disp('mlaroefn1: Maximum number of iterations reached.') ;
```

```
end ;

% k now contains the Kalman gain
% find the optimal starting estimate x0
% so as to minimize the total errors
M=zeros(T,n) ; % M*x0-p = err
p=zeros(T,1) ;   % the right side of the equation
yhat=zeros(T) ;
w=eye(n) ;              % the actual estimate of z is x0*w
wa=zeros(n,1) ;         % the absolute factor
IK=zeros(n) ; IK(:,n)=k ; IK=A*(eye(n)-IK) ;
for i=1:T,
  M(i,:)=-w(n,:) ; p(i)=wa(n) ;
  % Compute y(k|k-1) -> y(k+1|k)
  wa=A*(wa+k*(z(i)-wa(n))) ;
  w=IK*w ;
  end ;
p=z'-p ;

% the following is a more robust implementation of x0=M \ p ;

[U,S,V]=svd(M,0) ;

ms=max(diag(S))*1e-12 ;
for i=1:n,
  if S(i,i)>ms, S(i,i)=1/S(i,i) ;
           else S(i,i)=0 ;
  end ;
end ;

x0=V*S*U'*p ;
d=p-M*x0 ;
e=sum(d.*d) ;
end ;
```

12. Experimental results

Due to limited space and also due to the big time complexity of the ML estimator, we show only one example here.

We chose $x(t) = \sin(t/7) + \sin(t/10)$, negligible q and $r = 1$. We created 256 samples long test signal y that is shown on the picture.



The true value of $\boldsymbol{a} \doteq [3.9696, -5.9395, 3.9696, -1.0000]$ with corresponding loss function $J \doteq 254.31$.

The ML estimator yields (after about 1 hour of computation)

$$\hat{\boldsymbol{a}}_{\mathrm{ML}} \doteq [3.9714, -5.9441, 3.9735, -1.0011]$$

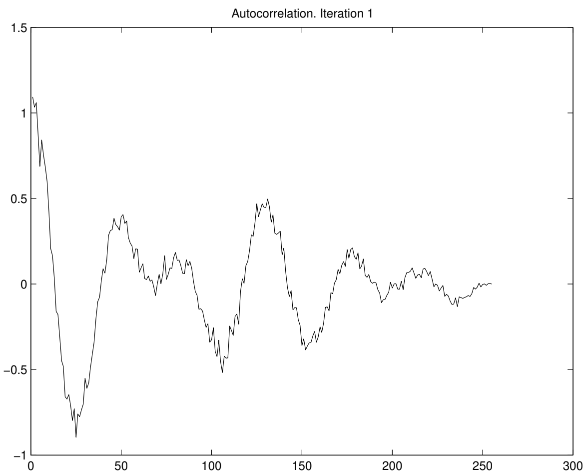
with corresponding loss function $J \doteq 250.64$

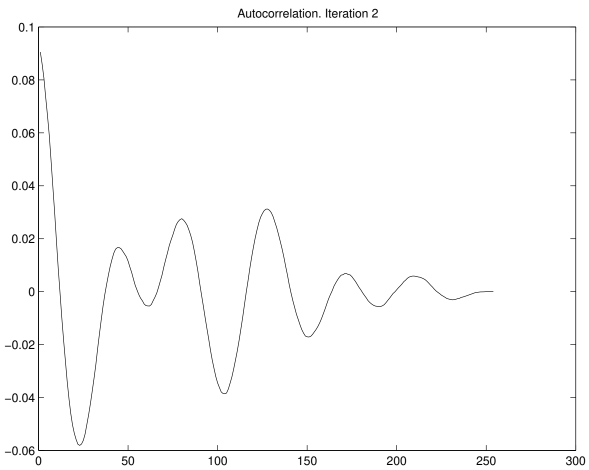
The progress of the repetitive autocorrelation estimator is described in the following table:

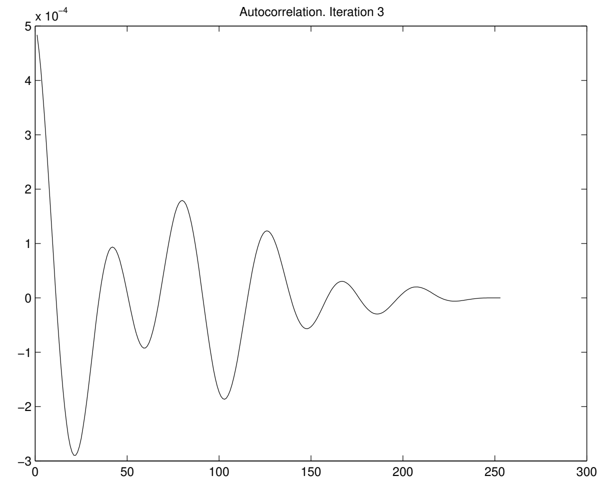
i	a_1	a_2	a_3	a_4	J
0	0.2327	0.2004	0.0973	0.2212	325.27
1	0.0887	−0.4125	1.1678	−0.0733	572.74
2	1.6674	−1.3587	1.5521	−0.8986	296.61
3	1.6173	−1.1601	1.3425	−0.8344	296.43
4	2.8182	−4.6705	4.7800	−1.9630	1200.08
5	1.9974	−0.1912	−1.6538	0.8454	300.24
6	1.1317	2.6231	−4.7005	1.9476	1174.63
7	3.9975	−6.0249	4.0573	−1.0301	296.38
8	3.9996	−6.0375	4.0763	−1.0387	318.18
9	3.9717	−5.9589	4.0022	−1.0154	321.94
10	3.9857	−6.0046	4.0518	−1.0334	335.39
11	4.0043	−6.0615	4.1101	−1.0535	345.59
12	4.0096	−6.0773	4.1261	−1.0590	348.43
13	4.0093	−6.0769	4.1262	−1.0591	349.24
14	4.0081	−6.0741	4.1240	−1.0587	350.20
15	4.0074	−6.0730	4.1238	−1.0589	351.73
16	4.0160	−6.1002	4.1528	−1.0692	359.20

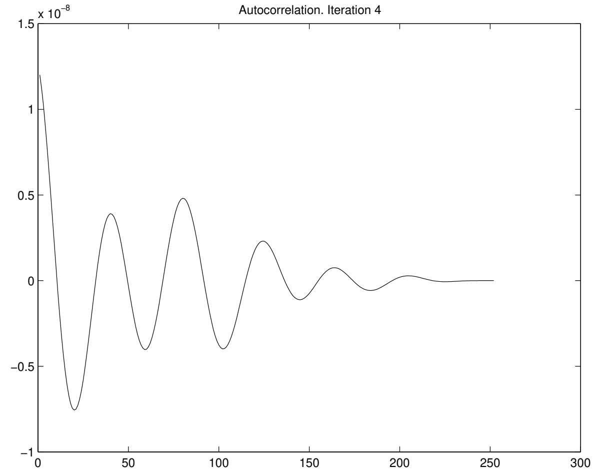
We can see that the process reaches optimum in the 7^{th} iteration. Total running time was about 7 s.

The results of repeated autocorrelation y_i (not normalized) are shown on the pictures:

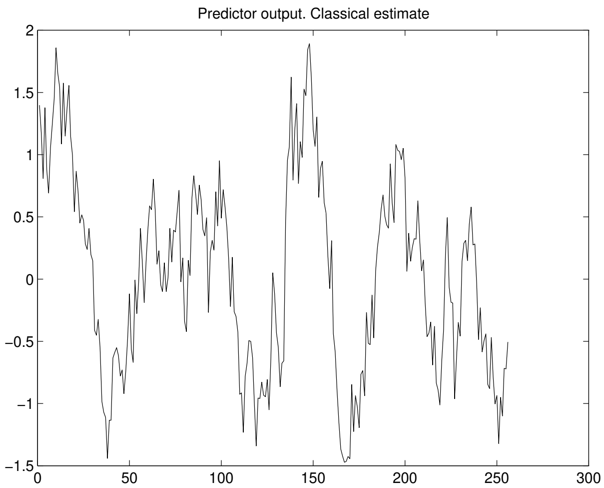


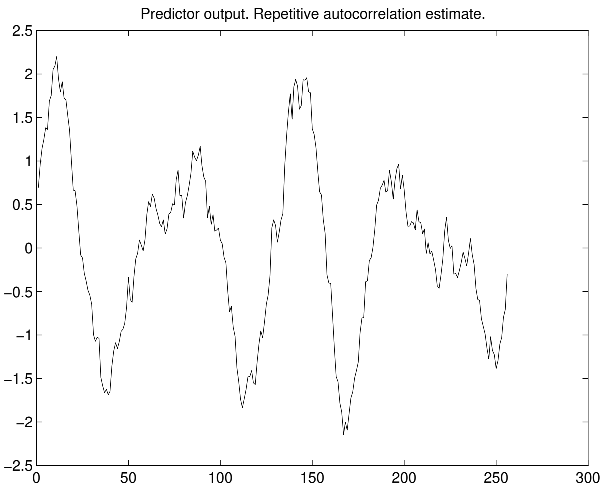


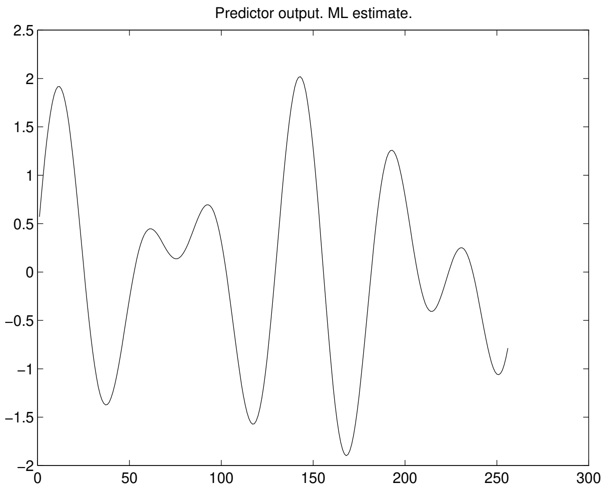


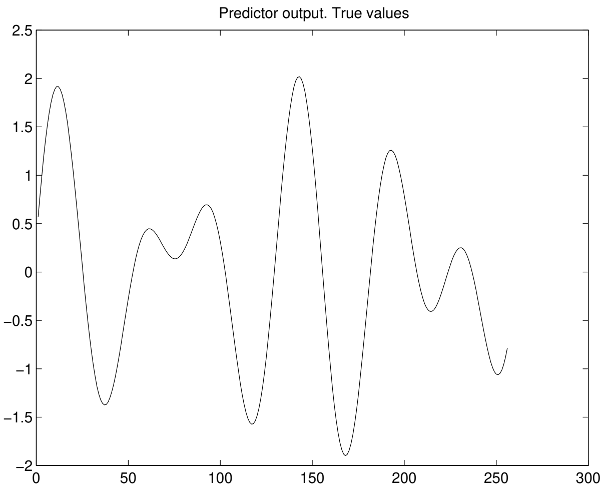


To compare the estimates, predictor outputs are shown on the pictures for various $\boldsymbol{a}$:









13. Conclusion

We have described a new innovative method of estimating parameters of autoregressive time-series with output error (measurement noise). Even though the method's results are not as good as those of the ML estimate they are better than those of the classical method and good enough for many practical purposes. Furthermore the method is relatively fast and is therefore suitable for on-line applications.

14. References

[1] J. Štecha a V. Havlena, *Moderní teorie řízení,* skriptum ČVUT, Praha, 1996.

[2] Z. Hrdina, *Statistická radiotechnika,*skriptum ČVUT, Praha, 1996.

[3] R. Mehra, *On-Line Identification of Linear Dynamic Systems with Applications to Kalman Filtering,*IEEE Trans. Automat. Contr., February 1971.

[4] G. Jenkins and D. Watts, *Spectral Analysis and its Applications,*Holden Day Publ. San Francisco, 1968.

[5] E. Parzen, *An Approach to Time-Series Analysis,*Ann. Math. Stat., vol. 32, December 1961

[6] M. Bartlett, *An Introduction to Stochastic Processes,*Cambridge Univ. Press, London, 1962

[7] R. Kashyap, *Maximum Likelihood Identification of Linear Dynamic Systems,*IEEE Trans. Automat. Contr., February 1970.

[8] R. Mehra, *On the Identification of Variances and Adaptive Kalman Filtering,*IEEE Trans. Automat. Contr., April 1970.