State-Space Model Identification via Exploitation of Displacement Structure: Batch and Recursive Processing

Y. M. Cho, T. Kailath
ISL, Dept. of Electrical Eng.
Stanford University
Stanford, CA 94305-4055

G. Xu
Dept. of Electrical & Computer Eng.
The University of Texas at Austin
Austin, TX 78712-1084

Abstract

The computational burden of state space model identification has limited its real-time application though it offers some important advantages over other methods based on input/output transfer functions. A recently proposed state space identification method [1] uses ideas from the theory of displacement structure and significantly reduces the computational burden. For an M x N data matrix (N > M), this batch method requires only O(MN) flops while the existing method takes O(MN^2) flops. However, for slowly changing systems, it is more desirable to update the model as time goes on with the minimal computational burden and we extend our results of the batch processing algorithm to allow updating of the identified state space model with O(M^2) flops. Some computer simulation results are also presented.

1 Introduction

The direct identification of state space models has recently attracted much attention [2, 3, 4, 5, 6, 7]. This recent surge of interest stems from some unique advantages of state space identification over the input-output (I/O) based identification: better numerical conditioning, estimation of the number of states (or the model order), extension to multi-input, multi-output (MIMO) systems.

Direct identification of state space models certainly has some drawbacks, e.g. it is suboptimal. The major drawbacks of direct state space identification techniques are its high computation and storage costs. We recently proposed [1] a faster batch identification method for state space models that exploits the displacement structure of Toeplitz-like matrices arising in the identification procedure and requires only O(MN) flops when the size of the data matrix is M x N. In some cases, the computational burden of our algorithm is acceptable even for on-line identification. However, since the data window N is usually much larger than the sliding window size M, the resulting heavy computational cost may preclude the possibility of on-line identification. Here, we shall extend our fast algorithm for batch processing and present a new recursive method to update the identification results provided a few new data samples. Given d new data samples, by exploiting the Hankel (or Toeplitz) structure of the data matrix, the new method reduces computational complexity to O(M^2 + dM) flops, d << M.

As in the batch algorithm, this recursive method also update the order of the system based on the new data samples.

In Section 2, we introduce the existing state space identification algorithms of De Moor et al. [3]. Section 3 covers our batch identification method. In Section 4, we adapt the identification technique for batch processing to allow recursive processing. In Section 5, we apply the newly developed algorithm to identify a simulated system.

2 A Conventional State Space Identification Method

A state space model for discrete-time lumped systems is considered,

\[ x_{k+1} = Ax_k + Bu_k, \quad y_k = Cx_k + Du_k, \]  

(1)

where \( x_k (\in \mathbb{R}^n) \) is the state of the linear system at time \( k \), \( u_k (\in \mathbb{R}^m) \) is the input to the linear system, and \( y_k (\in \mathbb{R}^p) \) is the observed output. The objective is to estimate \( A, B, C, \) and \( D \) from \( \{u_k\} \) and \( \{y_k\} \).
Following the notation of De Moor et al. [3], we start with the easily obtained equation

\[ Y = \Gamma_M X + HU, \]  

where

\[ Y = \begin{bmatrix} Y_k & Y_{k+1} & \cdots & Y_{k+N-1} \\ Y_{k+1} & Y_{k+2} & \cdots & Y_{k+N} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{k+N-1} & Y_{k+N} & \cdots & Y_{k+N+M-2} \end{bmatrix} \]

\[ \Gamma_M = \begin{bmatrix} C^T & A^T C^T & \cdots & (A^{M-1})^T C^T \end{bmatrix}^T \]

\[ X = [x_k \ x_{k+1} \ \cdots \ x_{k+N-1}] \]

\[ H = \begin{bmatrix} D & 0 & \cdots & 0 \\ CB & D & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CAM-2B & CAM-3B & \cdots & D \end{bmatrix} \]

where \( Y \in \mathbb{R}^{M \times N} \), \( \Gamma_M \in \mathbb{R}^{M \times M} \), \( X \in \mathbb{R}^{n \times N} \), \( H \in \mathbb{R}^{M \times M} \), \( U \in \mathbb{R}^{M \times N} \) (constructed in the same fashion as \( Y \)), and \( M_o = n_o M \), \( M_i = n_i M \). Next, we compute the complete SVD of \( U \),

\[ U = \Phi \begin{bmatrix} S & 0 \end{bmatrix} \begin{bmatrix} \Psi_1 \\ \Psi_2 \end{bmatrix}, \quad U = \Phi S \Psi_1. \]

We can define \( U^\perp = \Psi_2^T \) because of the fact that \( UU^\perp = 0 \). Applying \( U^\perp \) to (2) yields

\[ YU^\perp = \Gamma_M XU^\perp. \]  

This relation shows that the column span of \( \Gamma_M \) can be determined from \( YU^\perp \). Exploiting the special structure of \( \Gamma_M \), we can estimate \( A \) and \( C \) by using the ESPRIT algorithm [1, 9, 10] originated from sensor array processing.

Knowing \( \Gamma_M \), we can form an \( (M_o - n) \times M_o \) matrix \( Q \) such that \( Q \Gamma_M = 0 \). Then premultiplying \( Q \) and postmultiplying \( U^\perp \), we obtain

\[ QYU^\perp = QH \hat{=} K. \]

Since \( H \) is a linear function of \( B \) and \( D \) given \( A \) and \( C \), \( B \) and \( D \) can be obtained by solving a set of linear equations.

**Additional treatment of noise**

De Moor et al. [3] did not specifically discuss how to handle the additive noise. Here we shall show how to do this so that the obtained estimates are asymptotically unbiased. With the additive noise, (2) becomes

\[ Y = \Gamma_M X + HU + \nu, \]  

where \( \nu \) denotes the noise Hankel matrix constructed in a manner similar to \( Y \). Postmultiplying \( U^\perp \) yields

\[ YU^\perp = \Gamma_M XU^\perp + \nu U^\perp. \]  

The noise \( \nu \) is assumed to be a zero-mean Gaussian random vectors with variance \( \sigma^2 I \). Therefore, the covariance matrix of \( YU^\perp \) is

\[ C_y = \mathbb{E}(YU^\perp(YU^\perp)^T) \]

\[ = \Gamma_M XX^T \Gamma_M^T + \sigma^2 \Sigma_n \]

where the new noise covariance matrix \( \Sigma_n \) can be explicitly evaluated given \( U^\perp \) (see [4] for the details). With \( \Sigma_n \), we can pre- and post-multiply both sides of (5) by \( \Sigma_n^{-1/2} \) and \( \Sigma_n^{-1/2} \), respectively, to prewhiten the noise. Thus,

\[ \bar{C}_y = \Sigma_n^{-1/2} C_y \Sigma_n^{-1/2} \]

\[ = \Sigma_n^{-1/2} \Gamma_M XX^T \Gamma_M^T \Sigma_n^{-1/2} + \sigma^2 I. \]

Since \( \Gamma_M \) is a rank \( n(\ll M_o) \) matrix, \( \bar{C}_y \) can be expressed as a low-rank matrix plus a shift \( (\sigma^2 I) \). Such matrix structure falls into a standard subspace fitting framework and various subspace decomposition methods (e.g., eigendecomposition) can be applied to estimate the range space of the low-rank matrix, i.e., \( \text{span} \{\Sigma_n^{-1/2} \Gamma_M \} \).

Let \( (\lambda_i, e_i)_{i=1}^{M_o} \) denote all the eigenpairs of \( \bar{C}_y \). It is not difficult to see that the eigenvalues have the following distribution: \( \lambda_1 \geq \cdots \geq \lambda_n > \lambda_{n+1} = \cdots = \lambda_{M_o} \) and the desired subspace \( \text{span} \{\Sigma_n^{-1/2} \Gamma_M \} \) equals \( \text{span} \{e_1, \ldots, e_n\} \). By checking the number of smallest repeated eigenvalues, we can determine \( n \) and then the desired subspace \( \text{span} \{\Gamma_M \} \) is just \( \text{span} \{\Sigma_n^{-1/2} \{e_1, \ldots, e_n\}\} \). The above shows a way of exactly determining \( \text{span} \{\Gamma_M \} \) in the presence of noise.

### 3 A Fast State Space Model Identification: Batch Processing

In the existing methods, the SVD (singular value decomposition) of \( U \) requires \( O(MN^2) \) flops, while the ED (eigendecomposition) of \( YU^\perp U^\perp Y^T \) costs \( O(M^2) \) flops. As \( M \) and \( N \) become large, these computational costs are very burdensome. In addition,
We use the Fast Subspace Decomposition (FSD) procedure. For details, refer to [1].

To whiten the colored noise, we compute the noise covariance matrix. The product, inverse and pseudo-inverse of Hankel (or Toeplitz) matrices can also be computed through generator matrices, which results in a significant computational reduction [11, 12].

We present a summary of the new identification method to exploit the low-rank plus shift structure whenever U has such structure. Multiplying through in (4) by \( P_u^\perp \) yields

\[
YP_u^\perp = \Gamma_M XP_u^\perp + \nu P_u^\perp
\]

To whiten the colored noise, we compute the noise covariance matrix,

\[
\Sigma_n \overset{\text{def}}{=} E\{\nu P_u^\perp \nu^T\} = E\{\nu P_u \nu^T\}
\]

\( \Sigma_n \) is a Toeplitz matrix and the Toeplitz structure can be exploited to compute \( \Sigma_n^{1/2} \) and \( \Sigma_n^{-1/2} \) simultaneously with \( O(M^2) \) flops [1, 11].

We use the Fast Subspace Decomposition (FSD) method to exploit the low-rank plus shift structure of the covariance matrix \( \Sigma_n^{-1/2} Y P_u^\perp Y^T \Sigma_n^{-1/2} \). We estimate the signal subspace \( E_s \), including its dimension (i.e. the order of the system), with only \( O(M^2 + MN) \) flops.

Then A and C are estimated through the TLS ESPRIT algorithm mentioned in Section 2, which requires \( O(MN^2 + n^2) \) flops. Since \( n \ll M \) and \( N \), this particular computational cost is marginal.

We compute \( K = QYU^\perp \) to estimate B and D while exploiting the displacement structure of U to find \( U^\perp \) with only \( O(MN) \) flops. \( Q \) is replaced by \( E_s = I - E_s (E_s^TE_s)^{-1}E_s^T \) since the computation of \( \Gamma_M \) requires more flops.

By selecting proper rows of \( K, B \) and \( D \) can be found with a marginal computational cost.

4 A Fast State Space Model Identification: Recursive Processing

In the following, we focus our discussion on on-line recursive identification techniques to cope with time-varying scenarios. In recursive identification techniques, update/downdate and exponential weighting are commonly used. We adopt the former in our new recursive method for the identification of state space models.

Since the fast batch technique for state space identification reduces the computational load to \( O(M^2 + MN) \) flops, we may be able to use it in several on-line applications. However, the computational load \( O(MN) \) may not be acceptable for certain on-line applications when the number of data samples, \( N \), is quite large. Since only one measurement comes into and only one goes out of the data window \( N \), most of the computations involved are redundant. Judicious use of such redundancy can reduce the computational load further. It is reasonable to conjecture that the computational cost is only a function of \( M \) but not \( N \).

The inherent difficulty in making the fast batch algorithm recursive stems from the nonrecursive nature of the Schur algorithm and FSD. These two problems seem to be invincible and are not dealt with in this paper. As a result, the recursive algorithm requires at least \( O(M^2) \) flops which is the computational load of the Schur algorithm and FSD. Another major difficulty in the recursive identification technique arises from the noise covariance matrix update. Since the noise covariance matrix update in the fast identification method for batch processing involves the computation of a projection matrix of size \( N \times N \) and the summation of \( O(NM) \) elements, this step should be modified significantly. Except for updating of noise
covariance matrix, the basic structure of the batch algorithm is maintained. In fact, the recursive identification technique is an extension of the fast batch technique. Updating of the so-called generator matrices with sizes much smaller than the original matrices, constitutes a basic idea in the recursive algorithm. Of course, the displacement structure of the data matrices $U, Y$ and the low rank + shift structure of the covariance matrix play crucial roles. For detailed description of the algorithm, refer to [15].

Finally, we compare the computational load of our batch and recursive algorithms in Table 1.

<table>
<thead>
<tr>
<th>Steps</th>
<th>Find</th>
<th>Batch</th>
<th>Recursive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$U_k^+$</td>
<td>$O(MN)$</td>
<td>$O(M^2 + dM)$</td>
</tr>
<tr>
<td>2</td>
<td>$\Sigma_n^{1/2}, \Sigma_n^{-1/2}$</td>
<td>$O(M^3)$</td>
<td>$O(M^2 + dM)$</td>
</tr>
<tr>
<td>3</td>
<td>$E_z, n$</td>
<td>$O(MN)$</td>
<td>$O(M^2 + dM)$</td>
</tr>
<tr>
<td>5</td>
<td>$QYU^+$</td>
<td>$O(MN)$</td>
<td>$O(M^2 + dM)$</td>
</tr>
</tbody>
</table>

Table 1: Comparative flop counts of the batch and recursive methods. In the recursive processing, the flop counts are tabulated for the block update of block size $d$.

5 Simulations

![Figure 1: A snapshot of input/output data used in the simulation.](image)

Figure 1: A snapshot of input/output data used in the simulation.

We applied the new identification technique to the following single input, single output and two state-simulated system,

$$A_k = \begin{bmatrix}
0.7 + 0.2 \exp\left(\frac{t}{100}\right) & 0 \\
0 & 0.2 - 0.1 \exp\left(\frac{t}{100}\right)
\end{bmatrix}$$

$$B_k = \begin{bmatrix} 2 & 1 \end{bmatrix}^T, \quad C_k = \begin{bmatrix} 1 & 2 \end{bmatrix}, \quad D_k = 0.05$$

where $k$ is a time index starting from 0. The eigenvalues of the system matrix $A_k$ (the system "poles") drift from $(0.7, 0.2)$ to $(0.9, 0.1)$ during the simulation as shown in Fig. 2. The exponentially changing eigenvalues can arise, for example in a spring-mass system where the mass is exponentially decreasing due to leakage, say. A PRBS (pseudo random binary sequence) input (see Fig. 1) was designed and applied to the system to obtain a system model. No input process noise was added. The total time span of the simulation is 2100 samples. White Gaussian noise of $\sigma = 0.5$ was added to the output. Fig. 1 shows the resulting output. The signal to noise ratio for each state output was computed under the assumption that noise was added to each state output and turns out to be 28 dB for state 1 and 13 dB for state 2.

![Figure 2: Trajectory of the actual and estimated eigenvalues of the simulated system.](image)

The new identification technique was applied to the input/output data in Fig. 1 with data window size $N = 100$ and sliding window size $M = 20$. The identification technique could successfully detect the number of states, 2, throughout the simulation (this is mainly due to the high signal to noise ratio). Fig. 2 shows the identified eigenvalues together with the actual eigenvalues. The algorithm clearly tracks the actual eigenvalues. There are two interesting things to
note in Fig. 2. First, we can observe that the estimates lag behind the true values. This is due to the relatively big data window size. Since we assume that the system is time-invariant within the data window, the estimated parameters should be such that they best explain the data within the data window. Then, roughly speaking, the average eigenvalues within the data window would be proper estimates and the estimates would lag behind the true values in our example where the eigenvalues are exponentially decaying or monotonically changing. When the simulation was performed with a smaller data window size, the extent of lag became smaller. Second, the eigenvalue estimate corresponding to the first state is better than the eigenvalue estimate of the second state. This is mainly due to the higher signal to noise ratio of the first state output than the second. Overall, we can conclude the proposed identification scheme performs well for the given example.

6 Conclusions

New batch and recursive algorithms for identification of state space models have been presented. They exploit the displacement and low-rank shift structures of the matrices that arise in the identification procedure [1, 15]. The key idea in the new batch (or recursive) algorithms is to manipulate (or update) the so-called generator vectors instead of matrices, which reduces the computational cost down to $O(M^2 + dM)$ when we update the system parameters at every $d$ samples. Such reductions of computational burden may allow on-line identification of state space models in many potential applications. The results of computer simulation have given some evidence of the viability of the fast identification technique.

Acknowledgement

The authors wish to thank G. Franklin, A. Paulraj and R. Roy for useful discussions about this paper.

References