2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

State Space Subspace Noise Modeling with Guaranteed Stability

Xinhui Rong and Victor Solo

***Abstract*— A fundamental problem for state space system identification is guaranteeing stability of the fitted model. Here we consider state space subspace methods for noise models. The few existing algorithms that guarantee stability have various limitations. Most of these algorithms do not scale well to large state space dimension; have statistical biases and some have arbitrary tuning parameters that can introduce additional bias in the estimates. Here we present a new simple, computationally cheap method that guarantees stability and needs no tuning parameters. We illustrate its strong performance in comparative simulations.**

I. Introduction

Over several decades, state space identification problems have been widely explored by researchers in control [1], signal processing [2] and other areas. The class of state space subspace (S^4) methods are among the most important solution techniques thanks to special advantages, such as no need for explicit system parameterization, computational simplicity, and a range of 'tailored' algorithms [3].

Subspace methods involve projections on the row spaces of some data matrices extracted from the input-output data (or output only data for noise modeling) which are used to estimate the unobserved states. The associated computations only involve little more than Cholesky decomposition and singular value decomposition (SVD). Then further simple computations such as least squares generate estimates of the system matrices [3].

However, in a wide range of applications, it is crucial that the estimated system be stable; but S^4 methods do not guarantee this. In an early attempt to rectify this, authors of [4] proposed an algorithm which alters the unstable poles by iteratively appending the state-related matrix. Authors of [5] enforced stability by using regularizations and showed that the methods in [4] were special cases of their method. However, they had a matrix tuning parameter and the regularization might be too restrictive. In [2], an optimization problem was formulated to minimize the weighted 2-norm between the original unstable state transition matrix $\hat{A}_0$ and the stable one on the noise model, subject to a stability constraint captured in a Lyapunov equation. It was solved by semidefinite-programming (SP). In [6], SP in [2] was extended to the input-output model. Linear-matrix inequality (LMI) constraints were used in [7] for the more general constraints on poles, besides stability. Two methods were developed in [8] : Firstly, they iteratively kept restricting a singular value constraint on $\hat{A}_0$ and refitting until stability is attained. Secondly, they proposed a gradient sampling and searching algorithm, which exhibits better computational efficiency.

However, all the methods above have problems. Firstly, they need a preliminary estimate of the state transition matrix. Secondly, they use iterative methods to find a stable surrogate from this initial estimate. The method of [9] is not iterative, but is known to have large bias. In this paper, we deal with the purely stochastic state space model (aka, noise model), i.e. where there are no input signals. We propose a new method, which does not need a preliminary estimate, is not iterative, has low bias and yet guarantees stability. The new method builds on a vector auto-regression algorithm of Nuttall-Strand (N-S) [10][11]. It is, to the best of the authors' knowledge, the first time that the N-S algorithm has been applied for S^4 identification.

The rest of the paper is organized as follows. Section II reviews state space identification and the state space subspace approach. Section III summarizes the existing S^4 algorithms that guarantee stability. In section IV, we propose the new closed-form stable subspace algorithm, which we call S^5. In section V, we compare the algorithms in simulations and show the new algorithm has several advantages. Section VI contains conclusions and likely future work.

The following notations will be used throughout the paper. $(\cdot)'$ is transpose. $(\cdot)^*$ is conjugate transpose. $\bar{\cdot}$ takes the conjugate. $(\cdot)^\dagger$ is Moore-Penrose pseudo-inverse. $\cdot \geq 0 (\leq 0)$ means positive semidefinite/psd (negative semidefinite/nsd). $\cdot > 0 (< 0)$ means positive definite/pd (negative definite/nd). $\|\cdot\|_2$ denotes the matrix 2-norm, and $\|\cdot\|_F$ denotes the Frobenius norm.

II. **Preliminaries: State Space Subspace Methods**

We first specify the state space identification problem and review a standard state space subspace (S^4) method.

A. Stochastic State Space Model

We focus on S^4 identification of the stochastic state space model.

$$\begin{aligned} x_{t+1} &= Ax_t + w_t \\ y_t &= Cx_t + \nu_t, \quad t = 1, \cdots, \bar{T}, \end{aligned}$$

where x_t is the n-dimensional vector state, y_t is the d-dimensional vector output signal, w_t, ν_t are zero-mean white noises with covariance

$$\mathrm{E}\left[\left(\begin{smallmatrix} w_t \\ \nu_t \end{smallmatrix}\right)\left(\begin{smallmatrix} w_s' & \nu_s' \end{smallmatrix}\right)\right] = \left(\begin{smallmatrix} Q_{n\times n} & \Psi_{n\times d} \\ \Psi' & R_{d\times d} \end{smallmatrix}\right)\delta_{t,s},$$

X. Rong and V. Solo are with School of Electrical Engineering & Telecommunications, UNSW, Sydney, Australia. V. Solo is the corresponding author v.solo@unsw.edu.au.

979-8-3503-0123-6/23/$31.00 ©2023 IEEE

where $\delta_{t,s}= \begin{cases} 1, & \text{if } t=s \\ 0, & \text{if } t \neq s \end{cases}$ is the Kronecker delta function.

We assume that the underlying system is stable, i.e. eigenvalues of A are inside the unit circle, so that there is a steady state for which the state variance matrix obeys a Lyapunov equation

$$\mathrm{E}[x_t x_t'] = \Pi = A\Pi A' + Q.$$

The task is to identify the system order n, the system matrices $A \in \mathbb{R}^{n\times n}, C \in \mathbb{R}^{d\times n}$ (within a similarity transformation) and the noise covariances Q, Ψ, R, given only the output data y_t. For completeness, we give a brief review of S^4 methods and the unconstrained estimates of system matrices. The noise covariances can be obtained from the residual mean squares.

B. Canonical Correlations Analysis Subspace Algorithm

We review the canonical correlations analysis (CCA) subspace algorithm originally due to Akaike (see [3]). Some other S^4 algorithms are spelled out in [3].

First, choose a lag $m > n$. Let $N = md, T = \bar{T} - 2m + 1$, and define the past and future output matrices

$$Y_p = \begin{bmatrix} Y_p^{(1)} \\ Y_p^{(2)} \\ \vdots \\ Y_p^{(m)} \end{bmatrix} \quad Y_f = \begin{bmatrix} Y_f^{(1)} \\ Y_f^{(2)} \\ \vdots \\ Y_f^{(m)} \end{bmatrix},$$

where $Y_p^{(r)} = \begin{bmatrix} y_{m-r+1} & y_{m-r+2} & \cdots & y_{m-r+T} \end{bmatrix}$ and $Y_f^{(r)} = \begin{bmatrix} y_{m+r} & y_{m+r+1} & \cdots & y_{m+r+T-1} \end{bmatrix}$.

Define the near block Hankel matrix $H = \frac{1}{T} Y_f Y_p'$, and the near block Toeplitz matrices $\Sigma_p = \frac{1}{T} Y_p Y_p'$ and $\Sigma_f = \frac{1}{T} Y_f Y_f'$. Then carry out the singular value decomposition (SVD) $\Sigma_f^{-\frac{1}{2}} H \Sigma_p^{-\frac{1}{2}} = U \Lambda V'$, where Λ has decaying singular values down the diagonal.

We choose $n = \arg\min_r \ AIC(r)$ by minimizing the AIC [12] as a function of r

$$AIC(r) = (T-1) \textstyle\sum_{j=r+1}^{N} \ln(1-\lambda_j^2) - 2(N-r)^2, \tag{2.1}$$

where λ_r is the r-th largest singular value in Λ.

Then, let U_1 and V_1 be the first n columns of U and V, respectively, and Λ_1 be a diagonal matrix with first n diagonal entries of Λ. The state estimates are given by

$$\hat{X} = \begin{bmatrix} \hat{x}_{m+1}, \cdots, \hat{x}_{m+T} \end{bmatrix} = \Lambda_1^{\frac{1}{2}} V_1' \Sigma_p^{-\frac{1}{2}} Y_p.$$

For subsequent computation, we need the following state covariances. Define the *shifted state matrices*

$$\hat{X}_0 = \begin{bmatrix} \hat{x}_{m+1} & \cdots & \hat{x}_{m+T-1} \end{bmatrix}$$
$$\hat{X}_1 = \begin{bmatrix} \hat{x}_{m+2} & \cdots & \hat{x}_{m+T} \end{bmatrix}$$

and set

$$S_{ij} = \frac{1}{T-1} \hat{X}_i \hat{X}_j', \quad i, j \in \{0, 1\}.$$

C. Preliminary Estimates of A and C

Some of the methods (but not ours) require preliminary estimates of the system matrices. The simplest are given as follows. Introduce

$$P = \frac{1}{T} \hat{X} \hat{X}' = \Lambda_1, \quad \text{the estimate of } \Pi = \mathrm{E}[x_t x_t'],$$

$$S_{yy} = \frac{1}{T} Y_f^{(1)} Y_f^{(1)'} \quad \text{and} \quad S_{yx} = \frac{1}{T} Y_f^{(1)} \hat{X}'.$$

The least squares problems $\min_A \|\hat{X}_1 - A\hat{X}_0\|_F^2$ and $\min_C \|Y_f^{(1)} - C\hat{X}\|_F^2$, deliver estimates

$$\hat{A}_0 = \hat{X}_1 \hat{X}_0^{\dagger} = S_{10} S_{00}^{-1} \tag{2.2}$$

$$\hat{C}_0 = Y_f^{(1)} \hat{X}^{\dagger} = S_{yx} P^{-1}. \tag{2.3}$$

Of course, $\hat{A}_0$ is not guaranteed to be stable.

III. Existing Methods Guaranteeing S^4 Stability

We now summarize three existing stability enforced S^4 algorithms. Algorithm 1 and 2 are iterative, whereas Algorithm 3 has explicit formulae.

A. Iterative Augmentation (IA)

Iterative Augmentation (IA) [1] [4] starts with an unstable initial estimate $\hat{A}_0$ and iteratively appends the shifted state matrices $\hat{X}_0, \hat{X}_1$, in a way such that at convergence, the resulting least squares estimate $\hat{A} = \hat{X}_1 \hat{X}_0^{\dagger}$ is guaranteed stable and all unstable initial eigenvalues are moved to a magnitude of the tuning parameter $0 < \delta < 1$. We state below their key stability lemma without proof.

Lemma I. [4] Let $Z_0 = \begin{bmatrix} z_0 & z_1 & \cdots & z_{N-1} \end{bmatrix}, Z_1 = \begin{bmatrix} z_1 & z_2 & \cdots & z_N \end{bmatrix}, Z_0, Z_1 \in \mathbb{R}^{n\times Nm}$ and $\hat{A}_{n\times n} = Z_0 Z_1^{\dagger}$. If the last block $z_N = z_0 \Gamma \in \mathbb{R}^{n\times m}$ is a linear transformation of the first block z_0, where the largest singular value of $\Gamma_{m\times m}$ is less than or equal to 1, then the spectral radius of $\hat{A}$ is less than or equal to 1.

We have applied IA to noise modeling as follows.

Algorithm 1. Iterative augmentation (IA).

1) Choose a large $0 < \delta < 1$.
2) Find an unstable $\hat{A} = \hat{A}_0$ by (2.2).
3) Choose an unstable eigenvalue $|\lambda| > 1$ of $\hat{A}$ and the corresponding right eigenvector $v \in \mathbb{R}^n$.
4) If λ is real, set $c = \sqrt{\frac{|\lambda|/\delta - 1}{v'(\hat{X}_0 \hat{X}_0^*)\bar{v}}}$ and append a column cv to $\hat{X}_0$ and 0 to $\hat{X}_1$. Skip step 5) and step 6).
5) If λ is complex, compute $\left[\begin{smallmatrix} v^* \\ \bar{v}^* \end{smallmatrix}\right] (\hat{X}_0 \hat{X}_0^*)^{-1} \left[\begin{smallmatrix} v & \bar{v} \end{smallmatrix}\right] = \left[\begin{smallmatrix} w & x \\ y & z \end{smallmatrix}\right]$. Then solve $0 = (wz - xy)c^4 + (w+z)c^2 + 1 - \frac{|\lambda|^2}{\delta^2}$ for $c > 0$. Append $\left[\begin{smallmatrix} cv & c\bar{v} \end{smallmatrix}\right]$ to $\hat{X}_0$ and $\left[\begin{smallmatrix} 0 & 0 \end{smallmatrix}\right]$ to $\hat{X}_1$.
6) If the resulting solutions s_1, s_2 to the equation $s^2 - \frac{(1+c^2 z)\lambda + (1+c^2 w)\bar{\lambda}}{\Phi} s + \frac{|\lambda^2|}{\Phi}$, where $\Phi = (1 + c^2 w)(1 + c^2 z) - c^4 xy$, are real but either $|s_1|$ or $|s_2| > \delta$, then replace $c = \sqrt{\frac{-q + (q^2 - 4pr)^{\frac{1}{2}}}{2p}}$, where $p = z^2 r + xy|\lambda|^2$, $q = 2zr$, $r = Re\{\lambda\} - |\lambda|^2$.
7) Get $\hat{A} = \hat{X}_1 \hat{X}_0^{\dagger}$ with the augmented $\hat{X}_0, \hat{X}_1$. Go to step 3) until $\hat{A}$ has all eigenvalues inside the unit circle.

[1] We have modified the name used in [4] to indicate the iterative nature.

B. Semidefinite Programming (SP)

The SP method [2] starts with an unstable $\hat{A}_0$. It then modifies $\hat{A}_0$ by solving a SP problem that constrains $\hat{A}$ to be stable by a penalty based on a Lyapunov inequality as follows

$$\begin{aligned} \min_{\hat{A},P_0} \quad & \|(\hat{A}_0 - \hat{A})P_0\|_2^2 \\ \text{s.t.} \quad & P_0 > \epsilon I \\ & P_0 - \hat{A}P_0\hat{A}' > \epsilon I, \end{aligned} \tag{3.1}$$

The tuning parameter $\epsilon > 0$ must be chosen to ensure P_0 is not too close to 0. ϵ will also cause problems in high state dimension cases. The problem can be converted to a linear programming problem with linear matrix inequality (LMI) constrains as follows.

Algorithm 2. Semidefinite Programming (SP).

With an initial unstable $\hat{A}_0$, choose a small $\epsilon > 0$ and solve for G, $P_0 = P_0'$ and scalar γ

$$\begin{aligned} \min_{\gamma,P_0,G} \quad & \gamma \\ \text{s.t.} \quad & \begin{bmatrix} \gamma I & P_0\hat{A}_0' - G' \\ \hat{A}_0P_0 - G & I \end{bmatrix} \geq 0 \\ & \begin{bmatrix} P_0 - \epsilon I & G \\ G' & P_0 \end{bmatrix} \geq 0. \end{aligned}$$

Then the estimate $\hat{A} = GP_0^{-1}$ is guaranteed to be stable. SP can be solved iteratively by e.g. [13], [14], [15] and MATLAB LMI solvers which use state-of-the-art tools.

C. Data Attenuation (DA)

The following algorithm is a simpler version of IA which avoids iterations.

Algorithm 3. Data Attenuation (DA).

Append the estimated state matrix as $\hat{X} = \begin{bmatrix} \hat{x}_{m+1} & \cdots & \hat{x}_{m+T} & \hat{x}_{m+1} \end{bmatrix}$, and then $\hat{A} = S_{10}S_{00}^{-1}$ is stable.

DA is a special case of IA where we append the state matrix only once and let the transformation matrix $\Gamma = I$. Note that now $S_{00} = S_{11}$. Then, the residual mean square $S_w = S_{11} - S_{10}\hat{A} - \hat{A}'S_{01} + \hat{A}S_{00}\hat{A}' = S_{00} - \hat{A}S_{00}\hat{A}' > 0$, forms a discrete-time Lyapunov inequality to establish stability.

D. Other methods

Space limits preclude comparisons to other (iterative) methods. But we note that [7], [6] are essentially identical to [2] under noise modeling.

IV. New Method: S^5

We now propose a simple S^4 algorithm that guarantees system stability with closed-form formulae and does not depend on an initial estimate of A as others do. The idea is to apply a version of Nuttall-Strand (N-S) algorithm [10][11] to estimate A. We call it S^5 (S^4 with guaranteed stability).

N-S algorithm is a Burg-type algorithm developed in 1970's for vector autoregressive (VAR) model fitting. It involves minimizing a sum of forwards and backwards prediction error mean squares. The resulting estimates of the VAR filters are then guaranteed to be stable.

Although Strand [10] has a proof, it is for the more general VAR(p). It is also very hard to follow because of a proliferation of terminology and notation together with some ambiguities and absences in the defined notation. For these reasons we provide a simple, direct and accessible proof here. It only covers the VAR(1) case, but that is all we need, thanks to state space modeling.

First, we derive N-S method in the state space context. Note that we assume the state sequence is stationary and follows a stable forwards Markovian model in Section II.A, therefore, it is associated with a backwards model [16]

$$x_{t-1} = A_b x_t + w_{b,t}, \quad t = m+1, \cdots, m+T,$$

where $A_b = PA'P^{-1}$ and P is any consistent estimator of $\Pi = \mathrm{E}[x_t x_t']$. In practice, if we choose $P = \frac{1}{T}\hat{X}\hat{X}'$, then the best estimate $\hat{Q}_b = P - A_b P A_b'$ of backwards prediction error covariance $\mathrm{E}[w_{b,t}w_{b,t}']$ is minimized.

Given the state estimate $\hat{X}$, we estimate A by minimizing the sample mean squares of the weighted forwards and backwards errors as follows

$$\hat{A} = \arg\min_A J, \quad J = \mathrm{tr}\{P^{-1}(S_{w,f} + S_{w,b})\},$$

where

$$\begin{aligned} S_{w,f} &= \frac{1}{T-1}\sum\nolimits_{t=m+1}^{m+T-1}(\hat{x}_{t+1} - A\hat{x}_t)(\hat{x}_{t+1} - A\hat{x}_t)' \\ &= S_{11} - AS_{01} - S_{10}A' + AS_{00}A' \\ S_{w,b} &= \frac{1}{T-1}\sum\nolimits_{t=m+1}^{m+T-1}(\hat{x}_t - A_b\hat{x}_{t+1})(\hat{x}_t - A_b\hat{x}_{t+1})' \\ &= S_{00} - S_{01}A_b' - A_bS_{10} + A_bS_{11}A_b'. \end{aligned}$$

This leads to the following result.

Theorem I.

(a) The solution to $\min_A J$ obeys the following Sylvester equation

$$AS_{00}P^{-1} + S_{11}P^{-1}A = 2S_{10}P^{-1}. \tag{4.1}$$

and is the solution to the explicit equation

$$(I \otimes S_{11}P^{-1} + P^{-1}S_{00} \otimes I)vec(\hat{A}) = vec(2S_{10}P^{-1}), \tag{4.2}$$

where $vec(\hat{A})$ vectorizes matrix $\hat{A}$ by stacking the columns.

(b) $\hat{A}$ is stable.

Proof: (a) Introduce the auxiliary matrix $G = AP = PA_b'$. Then elementary algebra gives

$$\begin{aligned} S_{w,f} &= S_{11} - GP^{-1}S_{01} - S_{10}P^{-1}G' + GP^{-1}S_{00}P^{-1}G' \\ S_{w,b} &= S_{00} - S_{01}P^{-1}G - G'P^{-1}S_{01} + G'P^{-1}S_{11}P^{-1}G. \end{aligned}$$

We use the perturbation method to minimize J. Perturbing G by δG results in a first-order perturbation in J of

$$\delta J = 2\,\mathrm{tr}\{P^{-1}[(GP^{-1}S_{00} + S_{11}P^{-1}G - 2S_{10})]P^{-1}\delta G'\}.$$

For this to vanish for arbitrary $\delta G'$, we must have

$$\begin{aligned} &GP^{-1}S_{00} + S_{11}P^{-1}G = 2S_{10} \\ \Rightarrow &AS_{00} + S_{11}P^{-1}AP = 2S_{10} \\ \Rightarrow &AS_{00}P^{-1} + S_{11}P^{-1}A = 2S_{10}P^{-1}, \end{aligned}$$

as required.

Then, take vec operation on both sides and use $vec(ABC) = (C' \otimes A)vec(B)$ to get (4.1).

(b) We now prove the stability of $\hat{A}$. It suffices to show that $U_1 = P - APA'$ is pd. Then the discrete-time Lyapunov theorem delivers stability.

Rearrange the Sylvester equation (4.1) to get

$$AP = PS_{11}^{-1}(2S_{10} - AS_{00}).$$

Now put this into the U_1 equation to get

$$\begin{aligned} U_1 &= P - PS_{11}^{-1}(2S_{10} - AS_{00})A' \\ \Rightarrow S_{11}P^{-1}U_1 &= S_{11} - 2S_{10}A' + AS_{00}A' \qquad (4.3) \\ \Rightarrow U_1P^{-1}S_{11} &= S_{11} - 2AS_{01} + AS_{00}A'. \qquad (4.4) \end{aligned}$$

Setting $\Phi_f = -P^{-1}S_{11}$ and adding (4.3) and (4.4) gives

$$\Phi_f'U_1 + U_1\Phi_f = -2S_{w,f}.$$

Note that Φ_f has the same eigenvalues as $-P^{-\frac{1}{2}}S_{11}P^{-\frac{1}{2}}$, which are real and negative, and also that $-2S_{w,f} < 0$. Applying the Lyapunov theorem shows U_1 is pd and the result follows. ■

We conclude S^5 in the following algorithm.

Algorithm 4. S^5

Given the state estimates $\hat{X}$, let $P = \frac{1}{T}\hat{X}\hat{X}', S_{ij} = \frac{1}{T-1}\hat{X}_i\hat{X}_j', \ i,j \in \{0,1\}$. Estimate $\hat{A}$ as (4.2) and $\hat{A}$ is guaranteed stable.

We make the following remarks. Firstly, to effectively solve (4.2) in the higher dimensional case, one would like to avoid directly inverting the Kronecker sum $I \otimes S_{11}P^{-1} + P^{-1}S_{00} \otimes I$. Instead, one should use the Schur decomposition on $S_{11}P^{-1}$ and $P^{-1}S_{00}$ and solve an equivalent Sylvester equation [17]. The MATLAB command 'sylvester' does the above automatically.

Secondly, N-S algorithm should not be confused with the algorithm of Morf et al [18] which reparameterizes in terms of partial correlation matrices and thus generates a positive definite spectrum, but cannot be used to generate a stable state space model. This basically is because of the difference between singular value decomposition (Morf) and eigenvalue decomposition (N-S).

Lastly, S^5 offers a brand new mechanism for the stability enforced subspace algorithms, because almost all existing ones seek to minimize the distance of the stable estimate and the preliminary unstable one, under stability constrains, while in S^5, minimizing the forwards-backwards errors automatically produces stable estimates. Also, S^5 does not rely on a preliminary unstable estimate as IA and SP do, has explicit results and is simple.

V. Simulations

We now compare our new method with the existing methods in some simulations. We compare:

1) their accuracy in a low order case.
2) higher order cases ($n \geq 100$) - rarely done in the S^4 literature.
3) computation times - also rarely done in S^4 literature.

A. *Simulations with model order $n = 5$*

We simulate $n = 5$-th order single-output noise model as follows:

We design the poles of the state space model to be $\substack{0.7 \pm 0.707i \\ = 0.995e^{\pm 0.791i}}$, $\substack{-0.2 \pm 0.775i \\ =0.8e^{\pm 1.82i}}$, -0.95. The poles are chosen close to the unit circle such that the standard least squares is more prone to give unstable estimates.

The true state transition matrix A is taken as the transpose of the observable canonical form and

$$C = \begin{bmatrix} -0.97 & 0 & -0.98 & 1 & 0.75 \end{bmatrix},$$

and the zero-mean noises have covariances

$$Q = \begin{bmatrix} 2.14 & \cdot & \cdot & \cdot & \cdot \\ -0.51 & 2.43 & \cdot & \cdot & \cdot \\ 0 & -0.26 & 2.24 & \cdot & \cdot \\ -0.27 & -0.31 & -0.71 & 0.5 & \cdot \\ 2.55 & -2.69 & -1.18 & 0 & 0.45 \end{bmatrix}, \quad R = 5.46,$$
$$\Psi' = \begin{bmatrix} 0 & -0.38 & -0.74 & 1.07 & 0.77 \end{bmatrix},$$

where the dots in Q can be completed by symmetry and Q has eigenvalues $0.0027, 0.13, 1.89, 2.65, 8.48$.

We consider three record lengths: $\bar{T} = 120, 240, 480$ and set the lag $m = 7$. Since IA and SP need to alter an unstable initial estimate, we simulate a large number of repeats, sufficient to generate $1,000$ 'unstable' realizations for each $\bar{T}$, and apply each algorithm to each 'unstable' realization. $18.1\%, 7\%$ and 0.68% initial least squares estimates $\hat{A}_0$'s are unstable for $\bar{T} = 120, 240, 480$ respectively.

For IA we set $\delta = 0.99$. For SP we set $\epsilon = 10^{-12}$ and we use the MATLAB LMI solver to solve the SP.

We compare the algorithms by comparing: 1) the system pole locations, 2) the relative prediction errors, and 3) computational times.

1) Pole Locations: We first display the results by plotting the locations of the system poles (i.e. eigenvalues of $\hat{A}$) on the complex plane in Fig. 1, Colors represent different algorithms and the black '$\times$' marks are the true poles. It is sufficient to show only the poles with zero or positive imaginary part.

Fig. 1 gives a rough impression of estimated pole locations. All algorithms give stable solutions and give better estimates as $\bar{T}$ increases and the poles are more concentrated around the true ones, including the non-dominating poles.

However, to better compare the algorithms, in Fig. 2, we plot the histograms of the magnitudes of the estimated

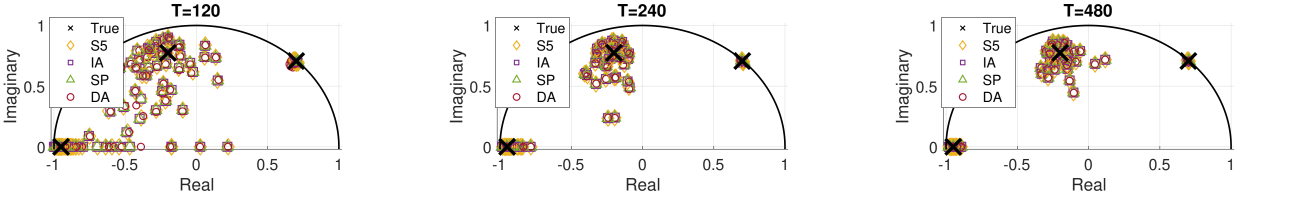

Fig. 1. Pole locations for $\bar{T} = 120, 240, 480$: The black cross marks are the true poles and the colored shapes are poles of estimated systems, with each color representing a method. Only the first 50 of the total 1000 repeats are shown for clearer presentation.

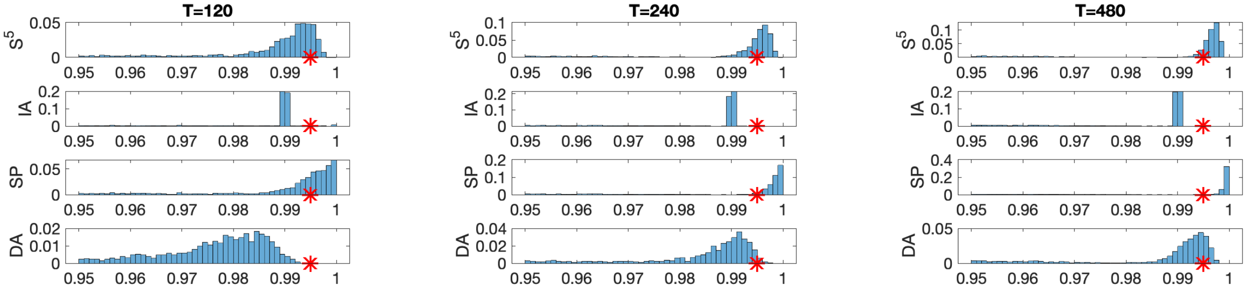

Fig. 2. Histograms of the magnitude of the dominating poles for the 1000 estimated systems for $\bar{T} = 120, 240, 480$: The red mark means the true dominating magnitude.

dominating poles based on the $1,000$ 'unstable' repeats. The true magnitude 0.995 is marked '$*$' in red.

Some interesting features are apparent from Fig. 2:

(a) The estimated poles from DA are furthest from the unit circle and have the greatest mean square errors.

(b) SP has the poles closest to the unit circle.

(c) The histogram of S^5 is centered at the true value at $\bar{T} = 120$, but it moves closer to the unit circle for larger $\bar{T}$.

(d) Estimated poles being closer to the unit circle than the true value is natural because the realizations are such that the least squares gives unstable results. Therefore, the estimated poles are expected to be closer to the unit circle, minimizing the prediction errors.

(e) For IA, the dominating poles are fixed at the magnitude 0.99 equal to the value of the user defined parameter δ. It maps the initial unstable poles only onto a small subset of the open unit disc, which results in a distortion/bias.

2) Relative Prediction Errors: We now compare the relative prediction errors of the algorithms defined as

$$e = \frac{\|\hat{X}_1 - \hat{A}\hat{X}_0\|_F - \|\hat{X}_1 - \hat{A}_0\hat{X}_0\|_F}{\|\hat{X}_1 - \hat{A}_0\hat{X}_0\|_F} \times 100\%.$$

Note that $e > 0$ because $\hat{A}_0 = S_{10}S_{00}^{-1}$ minimizes the Frobenius prediction error.

The histograms are shown in Fig. 3. Red '$*$' marks indicate the median and blue '$|$' marks indicate the upper and lower quantiles. Some observations can be drawn:

(a) SP has the smallest errors for all $\bar{T}$, because it is designed to minimize such error.

(b) DA has the largest errors.

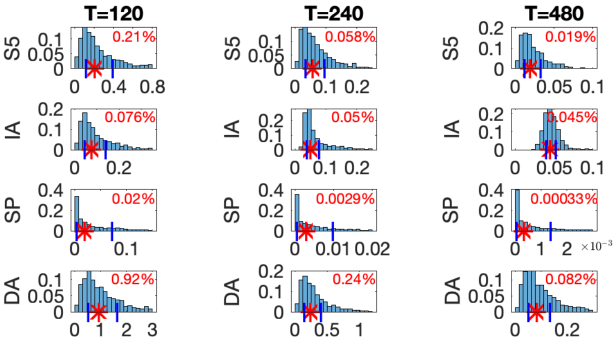

Fig. 3. Histograms of the relative errors (%) of $1,000$ repeats for $\bar{T} = 120, 240, 480$: The values at the upper right corners is the median values. The medians are also marked $*$ in the histogram.

(c) S^5 has reducing errors as $\bar{T}$ grows. Its error equals IA's at $\bar{T} = 240$ and outperforms IA at $\bar{T} = 480$.

(c) The errors of IA stop reducing at $\bar{T} = 240$, because the dominating poles are fixed at δ as discussed above. IA only has small error when the user defined parameter δ happens to be set close to the true magnitude of the dominating pole.

Despite SP showing the smallest relative errors, the relative errors of S^5 are quite acceptable.

Now we turn to computation times.

3) Computation Times: The total computation time for completing the $1,000$ repeats, is summarized in the Table I for each algorithm. We conclude that,

(a) DA is fastest but has the worst performance.

(b) SP performs best, but takes $\approx \times 700$ more time than S^5.

(c) IA takes $\approx$ $\times 2$ time as long as S^5 does at $\bar{T} =$

Method	$\bar{T} = 120$	$\bar{T} = 240$	$\bar{T} = 480$
S^5	0.15s	0.215s	0.300s
IA	14.4s	0.315s	0.551s
SP	18.5s	18.3s	18.4s
DA	0.094s	0.115s	0.226s

TABLE I

TOTAL COMPUTATIONAL TIMES (S) FOR $1,000$ REPEATS OF $n = 5$-ORDER MODEL .

240, 480. But at small $\bar{T} = 120$, it takes $\approx \times 100$ time as long as S^5. This is probably because, if the data size is small and a real pole has similar magnitude to the dominating complex pole, then slow convergence of IA will occur. This also happens when IA is unable to continue with large $\delta = 0.99$ and needs to be restarted with a smaller δ.

(d) S^5 is the most computationally efficient regarding also the performance. Also note that S^5 does not need the initial computation of $\hat{A}_0$ while IA and SP require it.

B. Higher Order Cases

We randomly generate 100 models each, of order $n = 100, 150, 250, 500, 1000$ with $\bar{T} = 5n$, and simulate a single realization to compare the computational times. We assume that the model order n is known so that each computation will be conducted on the system of the correct order.

Here we need to use an external MATLAB package, SDPT-3, from [19]. By comparison, LMI is more efficient in lower order case; it takes $1/10$ time compared to SDPT-3 with order $n = 5$. However, with order $n = 100$, LMI solver could not even finish one iteration in 12 hours.

The average computation times are summarized in Table II. We completed all model computations for S^5, IA and DA. But for SP we could only take the average on up to 10 models and up to order $n = 150$, because computation became prohibitive. The main problem was storage. Even state space models with sparse SP constraint matrices did not make much difference. Of course a bigger storage system or the use of clusters could overcome this. But it would not change the growth of storage required by SP as n rises.

S^5 is shown to be computationally efficient for all orders. It completes in almost no time for all cases. But SP becomes computationally prohibitive as order increases.

C. Summary

IA: places dominating poles at a user defined magnitude, which distorts pole locations.

SP: has the smallest prediction error but is computationally intensive in time and storage, becoming prohibitive in high order cases.

DA: introduces largest bias.

S^5: is computationally cheap even in higher order cases and has competitive precision.

VI. Conclusions and Future Work

In this paper, we proposed a new, closed-form state space subspace algorithm, S^5, to identify stable noise models from time series data. Compared to existing methods, S^5 has several advantages as shown in the simulation study: low bias; high precision; extremely cheap computation which allow scaling up to huge state dimensions. Such big dimension state space models are increasingly arising in numerous applications.

Method	$n = 100$	$n = 150$	$n = 250$	$n = 500$	$n = 1000$
S^5	0.0085s	0.022s	0.053s	0.28s	1s
IA	22.2s	49.9s	114s	155s	1861s =31mins
SP	1947s =32mins	14026s =4hrs	-	-	-
DA	0.0042s	0.014s	0.035s	0.2s	0.75s

TABLE II

AVERAGE COMPUTATIONAL TIME FOR HIGHER ORDER MODELS.

In the future, we will extend S^5 to input-output state space models and also seek to develop maximum-likelihood methods.

REFERENCES

[1] T. McKelvey, H. Akcay, and L. Ljung, “Subspace-based multivariable system identification from frequency response data,” *IEEE Trans. Autom. Contr.*, vol. 41, no. 7, pp. 960–979, 1996.

[2] J. Mari, P. Stoica, and T. McKelvey, “Vector ARMA estimation: a reliable subspace approach,” *IEEE Trans. Signal Process.*, vol. 48, no. 7, pp. 2092–2104, 2000.

[3] P. Van Overschee and B. DE Moor, *Subspace identification for linear systems : theory, implementation, applications*. Boston: Kluwer, 1996.

[4] N. L. C. Chui and J. M. Maciejowski, “Realization of stable models with subspace methods,” *Automatica*, vol. 32, no. 11, pp. 1587–1595, 1996.

[5] T. Van Gestel, J. Suykens, P. Van Dooren, and B. De Moor, “Identification of stable models in subspace identification by using regularization,” *IEEE Trans. Autom. Contr.*, vol. 46, no. 9, pp. 1416–1420, 2001.

[6] S. L. Lacy and D. S. Bernstein, “Subspace identification with guaranteed stability using constrained optimization,” *IEEE Trans. Autom. Contr.*, vol. 48, no. 7, pp. 1259–1263, 2003.

[7] D. Miller and R. de Callafon, “Subspace identification with eigenvalue constraints,” *Automatica*, vol. 49, no. 8, pp. 2468–2473, 2013.

[8] G. Mallet and S. Gasso, Gand Canu, “New methods for the identification of a stable subspace model for dynamical systems,” in *IEEE Workshop on Machine Learning for Signal Process.* IEEE, 2008, Conference Proceedings, pp. 432–437.

[9] J. M. Maciejowski, “Guaranteed stability with subspace methods,” *Systems & Control Letters*, vol. 26, no. 2, pp. 153–156, 1995.

[10] O. Strand, “Multichannel complex maximum entropy (autoregressive) spectral analysis,” *IEEE Trans. Autom. Contr.*, vol. 22, no. 4, pp. 634–640, 1977.

[11] A. H. Nuttall, “Multivariate linear predictive spectral analysis employing weighted forward and backward averaging: a generalization of Burg’s algorithm,” Naval Underwater Systems Center, London, Report, 1977.

[12] Y. Fujikoshi and L. G. Veitch, “Estimation of dimensionality in canonical correlation analysis,” *Biometrika*, vol. 66, no. 2, pp. 345–351, 1979.

[13] P. Gahinet and A. Nemirovski, “The projective method for solving linear matrix inequalities,” *Math. Programming*, vol. 77, no. 1, pp. 163–190, 1997.

[14] I. E. Nesterov and A. S. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. Philadelphia: SIAM, 1994.

[15] L. Vandenberghe and S. Boyd, “Semidefinite programming,” *SIAM review*, vol. 38, no. 1, pp. 49–95, 1996.

[16] T. Kailath, A. Sayed, and B. Hassibi, *Linear estimation*. New Jersey: Prentice Hall, 2000.

[17] L. A., *Matrix analysis for scientists and engineers*. Philadelphia, PA: SIAM, 2005.

[18] M. Morf, A. Vieira, D. L. Lee, and T. Kailath, “Recursive multichannel maximum entropy spectral estimation,” *IEEE Trans. Geosci. Electron.*, vol. 16, no. 2, pp. 85–94, 1978.

[19] T. M. J. Tutuncu R. H., Toh K. C., “SDPT3 — a MATLAB software package for semidefinite-quadratic-linear programming,” Dept. Math. Sciences, Carnegie Mellon, Report, 2001.