ABSTRACT

We present an analytic solution to the past input reconstruction problem, which consists in describing all past input sequences which would give rise to a given set of variables in fast least-squares algorithms, whenever the variables in question are reachable.

1 INTRODUCTION

Let \( u_i \) be a sequence of row vectors, each with \( M+1 \) elements, and stack these one atop another to build a data matrix \( U(n) \), with \( u_n \) as the top row. With \( \Lambda(n) = \text{diag}[1, \lambda, \ldots, \lambda^n] \), recursive least-squares filtering algorithms often invoke the time-propagation of the covariance matrix \( P(n) = U(n) \Lambda(n) U(n) \), or its inverse, or its Cholesky factor, etc. Fast least-squares algorithms may be developed when the vectors \( u_n \) derive from a delay line, and the resulting algorithms feature order \( M \) complexity in both storage and computation. The matrix recursions involving \( P(n) \) are replaced by a prediction section, which takes the form of a time-recursive computation

\[
\xi(n) = T[\xi(n-1), u_n]
\]

in which \( u_n \) is a scalar input sample, \( \xi(\cdot) \) is the state vector which collects all variables that need be written for storage, and \( T[\cdot, \cdot] \) is a nonlinear map which implements the fast least-squares prediction subroutine at each time instant.

Suppose the past input \( u_0, u_{-1}, u_{-2}, \ldots \), is allowed to vary arbitrarily, and let \( \mathcal{S}_i \) be the set of state variables \( \xi(n) \) that are reachable in exact arithmetic. It is known \cite{1}–\cite{3} that unstable error propagation is possible only if the finite precision version of \( \xi(\cdot) \) exits \( \mathcal{S}_i \), so that \( \mathcal{S}_i \) furnishes a stability domain. Deducing necessary conditions for a candidate state \( \xi(\cdot) \) to belong to \( \mathcal{S}_i \) involves exploiting known least-squares consistency conditions \cite{2}, \cite{3}; showing these conditions to be sufficient involved further labor \cite{4}. But by definition of \( \mathcal{S}_i \), if a given state \( \xi(\cdot) \) is indeed reachable, then it must be possible to place in evidence some past input sequence \( u_n, u_{n-1}, u_{n-2}, \ldots \) which gives rise to this state. We solve here the past input reconstruction problem, which consists in describing all valid past inputs for a given state, whenever the state is reachable. This complements the stability domain concept initiated in \cite{1}.

2 PROBLEM STRUCTURE

In fast least-squares algorithms the input vector derives from a scalar sequence passed through a delay line:

\[
u_n = [u_n \ u_{n-1} \ \cdots \ u_{n-M}].
\]

If \( u_n = 0 \) for \( n < 0 \), the matrix \( U(n) \) then assumes a “prewinded” Hankel structure:

\[
U(n) = \begin{bmatrix} u_n & u_{n-1} & \cdots & u_{n-M} \\ u_{n-1} & u_{n-2} & \cdots & u_{n-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ u_1 & u_2 & \cdots & u_M \\ u_0 & 0 & \cdots & 0 \end{bmatrix}
\]

Let us introduce the correlation lags

\[
r_k = \sum_{i=0}^{n} u_{n-i} u_{n-i-k}, \quad k = 0, 1, \ldots, M,
\]

and likewise rename the most recent input samples as

\[
x_1 = u_n, \quad x_2 = u_{n-1}, \quad \ldots \quad x_M = u_{n-M+1}.
\]

Then for any \( n \), one may check that the gramian of \( U(n) \), using \( \lambda = 1 \), takes the form

\[
P(n) = U(n) U(n) = \begin{bmatrix} r_0 & r_1 & \cdots & r_M \\ r_1 & r_0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & r_1 \\ r_M & \cdots & r_1 & r_0 \end{bmatrix} \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & x_M \\ \cdots & \cdots & \ddots & \vdots \\ 0 & x_1 & \cdots & 0 \end{bmatrix}^2
\]

The matrix \( P(n) \) is completely specified by \( 2M+1 \) values, namely \( r_0, \ldots, r_M \) and \( x_1, \ldots, x_M \).

When using a forgetting factor \( \lambda \), with \( \lambda < 1 \), the matrix \( P(n) \) becomes

\[
P(n) = U(n) \Lambda(n) U(n)
\]

Set \( L = \text{diag}[1, \lambda^{1/2}, \ldots, \lambda^{M/2}] \); since \( U(n) \) is a Hankel matrix, \( \Lambda^{1/2}(n) U(n) = U(n) L^{-1} \).
in which \( \mathbf{U}(n) \) is a Hankel matrix akin to (1), but built from the sequence
\[
\hat{u}_{n-k} = \lambda^{k/2} u_{n-k}.
\]
As such, the matrix \( \mathbf{P}(n) \) from (5), once multiplied from the left and right by the matrix \( \mathbf{L} \), will assume the same structure as if \( \lambda = 1 \) had been used [cf. (4)], and the past input had been exponentially weighted, as in (6). As this removes the influence of \( \lambda \), we may set \( \lambda = 1 \) with no loss of generality.

We now review more common parametrizations of \( \mathbf{P}(n) \).

### 2.1 Fast Transversal Filters

The fast transversal equations (with their many variants) are well defined only when \( \mathbf{P}(n) \) is invertible. The inverse \( \mathbf{P}^{-1} \) (time index \( n \) suppressed) has low displacement rank according to [2]
\[
\begin{bmatrix}
\mathbf{P}^{-1} & 0 \\ 0 & \mathbf{P}^{-1}
\end{bmatrix} = 
\begin{bmatrix}
\mathbf{A}_M & 0 \\ 0 & \mathbf{B}_M
\end{bmatrix} + 
\begin{bmatrix}
\mathbf{C}_M & 0 \\ 0 & \mathbf{B}_M
\end{bmatrix}
\]
in which the vectors \( \mathbf{A}_M, \mathbf{B}_M, \) and \( \mathbf{C}_M \) contain, respectively, normalized versions of the forward prediction error filter, the backward prediction error filter, and the Kalman gain vector. We refer to Slock [2] for more detail. These algorithms perform time updates not on the matrix \( \mathbf{P}^{-1}(n) \), but on the corresponding generator vectors \( \mathbf{A}_M(n), \mathbf{B}_M(n), \) and \( \mathbf{C}_M(n) \); these variables in turn yield the state vector \( \mathbf{x}(n) \).

### 2.2 Order Recursive Algorithms

Suppose \( \mathbf{P} \) is truncated to its \((k+1)\times(k+1)\) principal submatrix; the resulting matrix, once inverted and displaced akin to (7), yields generator vectors \( \mathbf{A}_k, \mathbf{B}_k, \) and \( \mathbf{C}_k \), each of \( k+1 \) elements. For any order \( k \), set
\[
\begin{align*}
\mathbf{A}_k(z) &= [1 \ z \ \cdots \ z^k] \mathbf{A}_k \\
\mathbf{B}_k(z) &= [1 \ z \ \cdots \ z^k] \mathbf{B}_k \\
\mathbf{C}_k(z) &= [1 \ z \ \cdots \ z^k] \mathbf{C}_k
\end{align*}
\]
These polynomials (evaluated at a common time index \( n \)) are known to be related by the order recursion [5]
\[
\begin{bmatrix}
\mathbf{A}_{k+1}(z) \\
\mathbf{C}_{k+1}(z) \\
\mathbf{B}_{k+1}(z)
\end{bmatrix} = 
\begin{bmatrix}
1 & \frac{1}{\cos \theta_k} & \frac{\sin \theta_k}{\cos \theta_k} \\
\sin \theta_k & 1 & \frac{1}{\cos \theta_k} \\
\cos \theta_k & \frac{1}{\cos \theta_k} & 1
\end{bmatrix}
\begin{bmatrix}
\mathbf{A}_k(z) \\
\mathbf{C}_k(z) \\
\mathbf{B}_k(z)
\end{bmatrix}
\]
in which \( \sin \theta_k \) is the correlation coefficient between the normalized forward and backward prediction errors of degree \( k \); and \( \sin \theta_k \) is the angle normalized backward prediction error of degree \( k \), divided by the square-root of the corresponding backward prediction error energy. These rotation angles appear in fast QR algorithm studied in [3], yielding the state vector \( \mathbf{x}(n) \), and many other variants may be found in fast QR/lattice algorithms [5–7].

### 2.3 Shift Invariance

Suppose the parameter values \( \{r_k\}_{k=0}^M \) and \( \{x_k\}_{k=1}^M \) are reachable at time \( n \), i.e., there exists some input sequence \( \{u_i\}_{i=0}^n \) fulfilling (2) and (3). Then these same values are reachable at time \( n+1 \), by applying a causal shift to the input sequence. Conversely, any parameter set \( \{r_k\}_{k=0}^M \) and \( \{x_k\}_{k=1}^M \) reachable at time \( n+1 \) is also reachable at time \( n \), provided the starting time is pushed back to \( i=-1 \). The set of asymptotically reachable parameters may be understood as those reachable by fixing the starting time at \( i=0 \) and letting the final time extend to \( n=+\infty \), or equivalently, by fixing the final time to \( n=-1 \) and letting the starting time extend back to \( i=-\infty \).

Upon adopting the latter convention, the \( z \)-transform of any valid past input sequence takes the form
\[
U(z) = \sum_{i=1}^{\infty} u_{-i} z^i, \quad |z| < 1, \quad (9)
\]
which yields a function analytic in \( |z| < 1 \). Moreover, the set of parameters \( \{r_k\} \) and \( \{x_k\} \) reachable at time \( n=-1 \) corresponds precisely to the set of valid initial conditions for the fast least-squares algorithm to proceed correctly from time \( n=0 \) onward. The past input reconstruction problem is then:

**Problem 1** Given the structured matrix
\[
\mathbf{P}(-1) = 
\begin{bmatrix}
r_0 & r_1 & \cdots & r_M \\
r_1 & r_0 & \cdots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
r_M & \cdots & r_1 & 0
\end{bmatrix}
\]
find all anti-causal functions as in (9) which satisfy the interpolation conditions
\[
\sum_{i=1}^{\infty} u_{-i} = x_k, \quad k=1,2,\ldots,M; \quad (10)
\]

This problem first arose in model reduction in Mullis and Roberts [8]; see also [9] and [10]. These works claim that a solution exists if and only if \( \mathbf{P}(-1) \) is nonnegative definite. Connections to classical interpolation theory surfaced in [11] and [12], from which one may show that a solution need not exist when \( \mathbf{P}(-1) \) is positive semi-definite.

### 3 A RELATED INTERPOLATION PROBLEM

Let \( \mathcal{Z} \) be the shift matrix with ones on the subdiagonal and zeros elsewhere. The matrix \( \mathbf{P}(-1) \) has low displacement rank, and its displacement residue \( \mathbf{P}(-1) - \mathcal{Z} \mathbf{P}(-1) \mathcal{Z}' \) becomes
\[
\mathbf{P}(-1) - \mathcal{Z} \mathbf{P}(-1) \mathcal{Z}' = 
\begin{bmatrix}
r_0 & r_1 & \cdots & r_M \\
r_1 & r_0 & \cdots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
r_M & \cdots & r_1 & 0
\end{bmatrix} - 
\begin{bmatrix}
x_1 \\
x_1 \\
\vdots \\
x_M
\end{bmatrix} \mathcal{Z}'^{\prime}
\]

\[
= 
\begin{bmatrix}
\sqrt{r_0} & 0 & \cdots & 0 \\
\sqrt{r_1} \mathcal{Z}'^{\prime} & r_0 & \cdots & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
\sqrt{r_M} \mathcal{Z}'^{\prime} & \cdots & \sqrt{r_1} \mathcal{Z}'^{\prime} & 0
\end{bmatrix}
\]
where \[ |\cdot| \] means “repeat the previous vector”. Now, behind most any displacement structure lurks an interpolation problem [13], [14]; that corresponding to (12) may be introduced as follows.

Let \( \text{S}(z) \) be a \( 2 \times 1 \) vector-valued Schur function, meaning that \( \text{S}(z) \) is analytic in \( |z| < 1 \) and contractive, i.e., \( \|\text{S}(z)\| < 1 \) in \( |z| < 1 \), where \( \|\cdot\| \) denotes the Euclidean norm. Let us set

\[
a(z) \equiv \sqrt{r_0} + \frac{r_1}{\sqrt{r_0}} z + \cdots + \frac{r_M}{\sqrt{r_0}} z^M,
\]

as well as \( [\frac{c(z)}{b(z)}] = \text{S}(z)a(z) \). We then have:

**Problem 2** Given the parameters \( \{r_k\}_{k=0}^M \) and \( \{x_k\}_{k=1}^M \), find a Schur function \( \text{S}(z) \) such that the resulting \( b(z) \) and \( c(z) \) assume the forms

\[
c(z) = 0 + x_1 z + x_2 z^2 + \cdots + x_M z^M + O_1(z^{M+1})
\]

\[
b(z) = 0 + \frac{r_1}{\sqrt{r_0}} z + \cdots + \frac{r_M}{\sqrt{r_0}} z^M + O_2(z^{M+1})
\]

where \( O(z^{M+1}) \) denotes a function analytic in \( |z| < 1 \) which vanishes \( M+1 \) times at \( z = 0 \).

This problem admits a solution \( \text{S}(z) \) if and only if a certain Pick matrix is nonnegative definite [15]; that corresponding to the present problem is simply \( \mathbf{P}(-1) \) from (12).

Since \( b(z) \) and \( c(z) \) both vanish at \( z = 0 \), while \( a(z) \) does not, we see that any solution \( \text{S}(z) \) to Problem 2 must vanish at \( z = 0 \). This allows us to write \( \text{S}(z) = \frac{[c(z)]}{[b(z)]} \). It is known [15] that whenever solutions exist, then lossless solutions exist, where lossless refers to a Schur function which has unit norm along the unit circle \( z = e^{j\alpha} \):

\[
|S_1(e^{j\alpha})|^2 + |S_2(e^{j\alpha})|^2 = 1, \quad \text{for all } \alpha.
\]

**Proposition 3** Let \( \text{S}(z) \) be a lossless solution to Problem 2. If the resulting \( z\text{S}(z) \) obeys the constraint

\[
1 - z\text{S}(z) \neq 0, \quad \text{for all } |z| = 1,
\]

then the function

\[
U(z) = \sqrt{r_0} \frac{z\text{S}(z)}{1 - z\text{S}(z)}
\]

is a solution to Problem 1. Moreover, all solutions to Problem 1 may be generated in this way.

For a proof, see [11]. In case (17) is violated, i.e., \( e^{j\alpha_0} \text{S}_2(e^{j\alpha_0}) = 0 \) for some value \( \alpha_0 \), then (16) gives \( \text{S}_1(e^{j\alpha_0}) = 0 \), producing a pole-zero cancellation on the unit circle in \( U(z) \). This possibility did not appear in [8]–[10], which explains the shortcoming of their claimed sufficient conditions.

4 CONSTRUCTING \( \text{S}(z) \)

Solutions to Problem 2 may be constructed by using a Schur algorithm; that to follow is adapted from [16].

We begin with the data array

\[
\mathbf{G} = \begin{bmatrix}
\sqrt{r_0} & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_M/\sqrt{r_0} \\
0 & x_1 & x_2 & \cdots & x_M \\
0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_M/\sqrt{r_0}
\end{bmatrix}, \quad (19)
\]

which contains the leading terms of the functions \( a(z) \), \( c(z) \), and \( b(z) \) from (13), (15), and (14).

1. Shift the first row of the array (19) one position to the right:

\[
(19) \Rightarrow \begin{bmatrix}
0 & \sqrt{r_0} & r_1/\sqrt{r_0} & \cdots & r_M-1/\sqrt{r_0} \\
0 & x_1 & x_2 & \cdots & x_M \\
0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_M/\sqrt{r_0}
\end{bmatrix}.
\]

2. Choose a hyperbolic rotation to knock off the second element of the first nonzero column. In the first pass, this appears as

\[
\begin{bmatrix}
1/\cos \theta_0 & \sin \theta_0/\cos \theta_0 \\
\sin \theta_0/\cos \theta_0 & 1/\cos \theta_0 \\
0 & \times \\
0 & \times \\
0 & \times
\end{bmatrix}
\times
\begin{bmatrix}
0 & \sqrt{r_0} & r_1/\sqrt{r_0} & \cdots & r_M-1/\sqrt{r_0} \\
0 & x_1 & x_2 & \cdots & x_M \\
0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_M/\sqrt{r_0}
\end{bmatrix}.
\]

in which \( y_1 = \sqrt{r_0-x_1} \) and \( \sin \theta_0 = -x_1/\sqrt{r_0} \).

3. Choose a hyperbolic rotation to knock off the third element of the first nonzero column. In the first pass, this appears as

\[
\begin{bmatrix}
1/\cos \phi_0 & \sin \phi_0/\cos \phi_0 \\
\sin \phi_0/\cos \phi_0 & 1/\cos \phi_0 \\
0 & \times \\
0 & \times \\
0 & \times
\end{bmatrix}
\times
\begin{bmatrix}
0 & y_1 & \times & \cdots & \times \\
0 & 0 & \times & \cdots & \times \\
0 & y_2 & \times & \cdots & \times \\
0 & 0 & \times & \cdots & \times
\end{bmatrix},
\]

in which \( y_2 = \sqrt{y_1^2-(r_1/\sqrt{r_0})} \) and \( \sin \phi_0 = -(r_1/\sqrt{r_0})/y_1 \).

4. Replace the array (19) with (20) and reiterate the above \( M-1 \) times, to eliminate all the elements of the second and third rows.

This procedure continues \( M \) full iterations yielding \( |\sin \theta_k| < 1 \) and \( |\sin \phi_k| < 1 \) if and only if the matrix \( \mathbf{P}(-1) \) is positive definite [16]. If \( \mathbf{P}(-1) \) is positive semi-definite, of rank \( k < M+1 \), the procedure terminates after \( k \) stages, yielding \( |\sin \theta_{k-1}| = 1 \) or \( |\sin \phi_{k-1}| = 1 \) [16].

A flowgraph of this operation for the positive definite case, applied to the functions \( a(z) \), \( c(z) \), and \( b(z) \), appears in Figure 1, for the case \( M = 3 \). Each successive stage introduces another leading zero into the three functions. The flowgraph of
Any such value of \( \phi \) may lose observability or controllability, and thus a continual lossless load, according to 

\[
\begin{bmatrix}
    c_M(z) \\
    b_M(z)
\end{bmatrix} = zS_L(z)a_M(z),
\]

the resulting function mapping \( a(z) \) to \( \begin{bmatrix} c(z) \\ b(z) \end{bmatrix} \) is lossless, and all lossless solutions to Problem 2 are exhausted by varying \( S_L(z) \) over all lossless possibilities (e.g., \([15]\)). A realization of \( U(z) \) as per (18), finally, is obtained by closing the input port and scaling the remaining output, as in Figure 3.

We now relate the rotation angles in Figure 2 to the state variables of fast least-squares algorithms. The following identity may be attributed to Lev-Ari et al. \([5]\):

**Identity 4** The rotation angles \( \{ \theta_k \} \) and \( \{ \phi_k \} \) of the order recursion (8) are precisely the angles determined from the above Schur algorithm.

These angles make an explicit appearance in, e.g., the fast QR algorithm studied in \([3]^{,4}\), and can be inferred from other minimal lattice and QR algorithms (e.g., \([5, 6, 7]\)).

This then specifies the fixed rotation angles building \( \Sigma(z) \) in Figure 3. For the lossless load \( S_L(z) \), the simplest choice is a constant: \( S_L = \begin{bmatrix} \sin \alpha \\ \cos \alpha \end{bmatrix} \), where \( \alpha \) may be varied freely. If \( P(-1) \) is positive definite, then one may show that finitely many values of \( \alpha \) exist for which the realization of Figure 3 may lose observability or controllability, and thus a continuum of values exists for which the realization is minimal (no pole-zero cancellation). Any such value of \( \alpha \) must give a \( zS_L(z) \) for which (17) is satisfied.

\(^4\)The angles \( \theta_k \) in Figure 2 are precisely those of \([3]\), but the angles \( \phi_k \) are denoted by \( \phi_{k+1} \) in \([3]\). The index on \( \phi \) is decremented in this paper so that rotations within a common section of Figure 2 take the same index.

References