International Journal of Computer Science & Emerging Technologies (E-ISSN: 2044-6004)
Volume 1, Issue 4, December 2010

State-Space Modelling of Dynamic Systems Using Hankel Matrix Representation

H. Olkkonen[1], S. Ahtiainen[1], J.T. Olkkonen[2] and P. Pesola[1]

[1] Department of Physics and Mathematics, University of Eastern Finland, 70211 Kuopio, Finland
[2] VTT Technical Research Centre of Finland, B.O. Box 1000, 02044 VTT, Finland
(email: hannu.olkkonen@uef.fi)

***Abstract*:** In this work a dynamic state-space model was constructed using a Hankel matrix formulation. A novel update algorithm for computation of the state transition matrix and its eigenvalues was developed. The method suits for analysis and synthesis of the rapidly changing dynamic systems and signals corrupted with additive random noise. The knowledge of the time varying state transition matrix and its eigenvalues enables accurate and precise numerical operators such as differentiation and integration in the presence of noise.

***Keywords*:** State-space modelling, dynamic systems analysis

1. Introduction

Estimation of the state of the dynamic systems and signals has been an object of vital research for many years impacted by the discovery of the Kalman filter (KF), extended Kalman filter (EKF), neural network algorithms and the LMS and RLS algorithms [1-7]. The adaptive algorithms have found to be suitable methods for many kind of linear and nonlinear system modelling. The update of the model parameters is based on the use of the forgetting functions. State-space models, which are based on matrix formulations have gained acceptance in various control system analysis and synthesis. A state-space approach differs significantly from the adaptive methods such as KF, EKF and RLS algorithms in that the system matrices are solved directly form the measured data matrices using least squares (LS) or total least squares (TLS) methods. The noise inherent in data matrices is usually cancelled by the singular value decomposition (SVD) based subspace methods [8-9]. A disadvantage in the SVD based solutions is the treatment of the data matrix blocks, which give the system matrices in a defined time interval. The matrices are then supposed to be time-invariant within the time interval. However, in rapidly changing dynamic systems the matrices may change abruptly and the SVD based methods give only a time averaged estimates.

In this work we present a dynamic state-space model, where the state transition matrix is updated at every time increment. The dynamic system modelling is based on the Hankel data matrix representation. We present a novel update algorithm for computation of the state-transition matrix and its eigenvalues. The method can be adapted for system state-space modelling and filtering the measurement signals in the presence of noise.

2. Theoretical considerations

2.1 The dynamic state-space model

The dynamic state-space model under consideration is defined as

$$\begin{aligned} X_{n+1} &= F_n X_n \\ y_n &= C X_n + w_n \end{aligned} \tag{1}$$

where the state vector $X_n \in R^{Nx1}$, the state transition matrix $F_n \in R^{NxN}$ and the vector $C = [1\ 0 \cdots 0] \in R^{1xN}$. The scalar $w_n \in R^{1x1}$ is a random zero mean observation noise. The signal $y_n \in R^{1x1}$ consists of measurements at time intervals $t = nT\ (n = 0,1,2,...)$, where T is the sampling period. Let us define the data vector $Y_n = [y_n\ y_{n-1}\ y_{n-2} \cdots y_{n-N-1}]^T$ where T denotes matrix transpose. The Hankel structured data matrix $H_n \in R^{NxM}$ is defined as

$$\begin{aligned} H_n &= \begin{bmatrix} y_n & y_{n-1} & \cdots & y_{n-M-1} \\ y_{n-1} & y_{n-2} & \cdots & y_{n-M-2} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n-N-1} & y_{n-N-2} & \cdots & y_{n-N-M-2} \end{bmatrix} \\ &= \begin{bmatrix} Y_n & Y_{n-1} & \cdots & Y_{n-M-1} \end{bmatrix} \end{aligned} \tag{2}$$

where the subscript n in H_n refers to the most recent data point y_n. The antidiagonal elements of the H_n data matrix are equal. The state-space model (1) can be represented in the following form

$$H_{n+1} = F_n H_n + W_{n+1} \tag{3}$$

where W_{n+1} is the Hankel structured noise matrix. The least squares estimate of the state transition matrix F_n comes from

$$F_n = H_{n+1} H_n^T (H_n H_n^T)^{-1} = H_{n+1} H_n^{\#} = R_n C_n^{-1} \tag{4}$$

where the pseudoinverse matrix $H_n^{\#} = H_n^T (H_n H_n^T)^{-1} \in R^{MxN}$. The matrices $R_n = H_{n+1} H_n^T \in R^{NxN}$ and $C_n = H_n H_n^T \in R^{NxN}$. The C_n matrix is symmetrical, i.e. $C_n^T = C_n$. It has a stable inverse since it is positive definite and all eigenvalues are nonnegative. The rank of the state transition matrix F_n defines the system order. In many applications the state transition matrix should be evaluated at T intervals. In

International Journal of Computer Science & Emerging Technologies (E-ISSN: 2044-6004)
Volume 1, Issue 4, December 2010

complex dynamic systems the dimension of the state transition matrix is high and the computation of the pseudoinverse matrix $H_n^{\#}$ would be an overwhelming task. In this work we show that with a special partitioning the R_n and C_n matrices into submatrices the computational load is drastically diminished.

2.2 Computation of the state transition matrix F_n

The data matrix H_n is partitioned as

$$H_n = \begin{bmatrix} D_n \\ d_n \end{bmatrix} \tag{5}$$

where the matrix $D_n \in R^{(N-1)xM}$ and the vector $d_n \in R^{1xM}$. The data matrix H_{n+1} is partitioned as

$$H_{n+1} = \begin{bmatrix} d_{n+1} \\ D_n \end{bmatrix} \tag{6}$$

where the vector $d_{n+1} \in R^{1xM}$ and the matrix D_n is identical to that in (5). Now we have

$$\begin{aligned} C_n &= \begin{bmatrix} D_n \\ d_n \end{bmatrix} \begin{bmatrix} D_n^T & d_n^T \end{bmatrix} = \begin{bmatrix} D_n D_n^T & D_n d_n^T \\ d_n D_n^T & d_n d_n^T \end{bmatrix} \\ &= \begin{bmatrix} A_n & b_n \\ b_n^T & c_n \end{bmatrix} \end{aligned} \tag{7}$$

where the matrix $A_n \in R^{(N-1)x(N-1)}$, the vector $b_n \in R^{(N-1)x1}$ and the scalar $c_n \in R^{1x1}$. The analytic solution of the inverse matrix is

$$C_n^{-1} = \begin{bmatrix} A_n & b_n \\ b_n^T & c_n \end{bmatrix}^{-1} = \begin{bmatrix} A_n^{-1} + s_n m_n m_n^T & -s_n m_n \\ -s_n m_n^T & s_n \end{bmatrix} \tag{8}$$

where the vector $m_n = A_n^{-1} b_n \in R^{(N-1)x1}$ and the scalar $s_n = (c_n - b_n^T m_n)^{-1} \in R^{1x1}$. The inverse matrix has the same block dimensions as the C_n matrix. Correspondingly, we have

$$\begin{aligned} R_n = H_{n+1} H_n^T &= \begin{bmatrix} d_{n+1} \\ D_n \end{bmatrix} \begin{bmatrix} D_n^T & d_n^T \end{bmatrix} \\ &= \begin{bmatrix} d_{n+1} D_n^T & d_{n+1} d_n^T \\ D_n D_n^T & D_n d_n^T \end{bmatrix} = \begin{bmatrix} e_n & g_n \\ A_n & b_n \end{bmatrix} \end{aligned} \tag{9}$$

where the vector $e_n \in R^{1x(N-1)}$ and the scalar $g_n \in R^{1x1}$. The matrix A_n and the vector b_n are identical to in (7). Now we obtain the state transition matrix as

$$\begin{aligned} F_n &= R_n C_n^{-1} \\ &= \begin{bmatrix} e_n & g_n \\ A_n & b_n \end{bmatrix} \begin{bmatrix} A_n^{-1} + s_n m_n m_n^T & -s_n m_n \\ -s_n m_n^T & s_n \end{bmatrix} \end{aligned}$$

which finally yields

$$F_n = \begin{bmatrix} h_n & k_n \\ I & \varnothing \end{bmatrix} \tag{10}$$

where the scalar $k_n = g_n s_n - s_n e_n m_n$ and the vector $h_n = e_n A_n^{-1} - k_n m_n^T$. The identity matrix $I \in R^{(N-1)x(N-1)}$ and the zero vector $\varnothing \in R^{(N-1)x1}$.

2.3 Updating the state transition matrix F_n

The updated matrix C_{n+1} is partitioned into the four sub-blocks

$$\begin{aligned} C_{n+1} &= \begin{bmatrix} d_{n+1} \\ D_n \end{bmatrix} \begin{bmatrix} d_{n+1}^T & D_n^T \end{bmatrix} = \begin{bmatrix} d_{n+1} d_{n+1}^T & d_{n+1} D_n^T \\ D_n d_{n+1}^T & D_n D_n^T \end{bmatrix} \\ &= \begin{bmatrix} p_{n+1} & e_n \\ e_n^T & A_n \end{bmatrix} \end{aligned} \tag{11}$$

where p_{n+1} is a scalar. The essential observation is that C_{n+1} contains a submatrix A_n, which is the same as in partitioning the C_n matrix (7). The vector e_n equals to that in (9). The analytic matrix inversion gives

$$\begin{aligned} C_{n+1}^{-1} &= \begin{bmatrix} p_{n+1} & e_n \\ e_n^T & A_n \end{bmatrix}^{-1} \\ &= \begin{bmatrix} r_{n+1} & -r_{n+1} q_n \\ -r_{n+1} q_n^T & A_n^{-1} + r_{n+1} q_n^T q_n \end{bmatrix} \end{aligned} \tag{12}$$

where the vector $q_n = e_n A_n^{-1}$ and the scalar $r_{n+1} = (p_{n+1} - q_n e_n^T)^{-1}$. We may note that the vector q_n is previously computed as a first term in vector h_n in (10). Finally, the computed C_{n+1}^{-1} matrix is represented in the repartitioned form (7)

$$\begin{aligned} C_{n+1}^{-1} &= \begin{bmatrix} A_{n+1} & b_{n+1} \\ b_{n+1}^T & c_{n+1} \end{bmatrix}^{-1} \\ &= \begin{bmatrix} A_{n+1}^{-1} + s_{n+1} m_{n+1} m_{n+1}^T & -s_{n+1} m_{n+1} \\ -s_{n+1} m_{n+1}^T & s_{n+1} \end{bmatrix} \\ &= \begin{bmatrix} T_{n+1} & z_{n+1} \\ z_{n+1}^T & w_{n+1} \end{bmatrix} \end{aligned} \tag{13}$$

where the vector $m_{n+1} = A_{n+1}^{-1} b_{n+1}$ and the scalar $s_{n+1} = (c_{n+1} - b_{n+1}^T m_n)^{-1}$. The matrix $T_{n+1} \in R^{(N-1)x(N-1)}$, the vector $z_{n+1} \in R^{(N-1)x1}$ and the scalar $w_{n+1} \in R^{1x1}$ are picked up from the computed inverse matrix C_{n+1}^{-1} (12). By comparing the block matrices we obtain the solution for the vector m_{n+1} and the inverse block matrix A_{n+1}^{-1} as

$$\begin{aligned} m_{n+1} &= -w_{n+1}^{-1} z_{n+1} \\ A_{n+1}^{-1} &= T_{n+1} + m_{n+1} z_{n+1}^T \end{aligned} \tag{14}$$

Now we get the updated R_{n+1} matrix

$$R_{n+1} = \begin{bmatrix} e_{n+1} & g_{n+1} \\ A_{n+1} & b_{n+1} \end{bmatrix} \tag{15}$$

International Journal of Computer Science & Emerging Technologies (E-ISSN: 2044-6004)
Volume 1, Issue 4, December 2010

where the vector $e_{n+1} \in R^{1x(N-1)}$ and the scalar $g_{n+1} \in R^{1x1}$. Finally, the uptake of the state transition matrix F_{n+1} comes from

$$F_{n+1} = R_{n+1} C_{n+1}^{-1} = \begin{bmatrix} e_{n+1} & g_{n+1} \\ A_{n+1} & b_{n+1} \end{bmatrix} \begin{bmatrix} A_{n+1}^{-1} + s_{n+1} m_{n+1} m_{n+1}^T & -s_{n+1} m_{n+1} \\ -s_{n+1} m_{n+1}^T & s_{n+1} \end{bmatrix} = \begin{bmatrix} h_{n+1} & k_{n+1} \\ I & \varnothing \end{bmatrix} \quad (16)$$

where the scalar $k_{n+1} = g_{n+1} s_{n+1} - s_{n+1} e_{n+1} m_{n+1}$ and the vector $h_{n+1} = e_{n+1} A_{n+1}^{-1} - k_{n+1} m_{n+1}^T$. An interesting feature is that the vector b_{n+1} needs not to be known in the uptake process (16).

2.4 Fast uptake of the R_n matrix

The uptake process (16) needs the computation of the vector $e_{n+1} = d_{n+2} D_{n+1}^T$ and the scalar $g_{n+1} = d_{n+2} d_{n+1}^T$, if partitioning (5) is used. This would require one vector-matrix and one vector-vector multiplication. Fast uptake of the R_n matrix is obtained if we adapt the partitioning (2) for the data matrix

$$H_n = \begin{bmatrix} Y_n & Y_{n-1} & \cdots & Y_{n-M-1} \end{bmatrix} \quad (17)$$

Now we have

$$R_n = [Y_{n+1} Y_n^T \quad Y_n Y_{n-1}^T \quad \cdots \quad Y_{n-M} Y_{n-M-1}^T] \quad (18)$$

and the uptake of the R_n matrix is yielded as

$$R_{n+1} = R_n + Y_{n+2} Y_{n+1}^T - Y_{n-M} Y_{n-M-1}^T \quad (19)$$

The uptake of the R_n matrix needs only two vector-vector multiplications and the vector e_{n+1} and the scalar g_{n+1} are simply picked up from the R_{n+1} matrix (19).

2.5 Computational complexity

The uptake of the state transition matrix requires the computation of the matrix inverse C_{n+1}^{-1} (12), which needs one vector-vector multiplication. The uptake of the vector m_{n+1} and the inverse block matrix A_{n+1}^{-1} (14) needs one vector-vector multiplication. Finally, the uptake of the scalar k_{n+1} and the vector h_{n+1} in (16) requires one vector-matrix and three vector-vector multiplications. Thus the computational complexity of the algorithm is $O(n^2) + 5O(n)$.

3. Applications of the method

3.1 Computation of the eigenvalues of the state transition matrix

The Hankel data matrix representation (3,5) of the dynamic state-space model leads to a companion matrix structure of the state transition matrix F_n (10), which involves only the vector h_n and the scalar k_n. An important advantage of the companion matrix structure is that the eigenvalues $\lambda_1, \lambda_2, ..., \lambda_N$ of the state transition matrix can be directly computed as the roots of the polynomial having coefficients $[1 \quad -h_n \quad -k_n]$. The eigenvalues of the state transition matrix give important knowledge of the order and the stability of the system and its dynamic behaviour. The eigenvalues also aid in selection of the model order. The occurrence of very small eigenvalues indicates that the system order is smaller that the model order, which leads to overmodelling. When the model order equals the system order, the scalar coefficient k_n attains a value $k_n = -1$.

3.2 Signal prediction and state-space filtering

The knowledge of the state transition matrix F_n enables the prediction of the measurement signal y_n as

$$H_{n+1} = F_n H_n \Rightarrow \hat{y}_{n+1} = C F_n H_n \quad (20)$$

where $C = [1 \; 0 \cdots 0] \in R^{1xN}$. Using the Hankel data matrix representation we may define the prediction data matrix as

$$\hat{H}_n = \begin{bmatrix} \hat{y}_n & \hat{y}_{n-1} & \cdots & \hat{y}_{n-M-1} \\ \hat{y}_{n-1} & \hat{y}_{n-2} & \cdots & \hat{y}_{n-M-2} \\ \vdots & \vdots & \vdots & \vdots \\ \hat{y}_{n-N-1} & \hat{y}_{n-N-2} & \cdots & \hat{y}_{n-N-M-2} \end{bmatrix} \Rightarrow \hat{H}_{n+1} = F_n \hat{H}_n \quad (21)$$

The state-space filtered signal $\hat{y}_n$ can be obtained as a mean of the antidiagonal elements. In the following we describe several matrix operators based on the state transition matrix. In all computations the filtered data matrix (21) is applied.

3.3 Numerical signal processing

The knowledge of the state transition matrix F_n enables the numerical signal processing of the state-space filtered signal. In the following we develop matrix operators based on the state transition matrix for numerical interpolation, differentiation and integration of the measurement signal.

The state transition matrix can be presented in the eigenvalue decomposited form $F_n = U_n D_n U_n^{-1}$, where $D_n \in R^{NxN} = diag(\lambda_1 \; \lambda_2 \ldots \lambda_N)$ and $U_n \in R^{NxN}$. Based on (21) we have a result

$$\hat{H}_{n+1} = F_n \hat{H}_n = U_n D_n U_n^{-1} \hat{H}_n \Rightarrow \hat{H}_{n+\Delta} = U_n D_n^{\Delta} U_n^{-1} \hat{H}_{n+\Delta} = F_n^{\Delta} \hat{H}_n \quad (22)$$

where the time-shift $\Delta \in [0, T]$. Now we may define the interpolating time-shift operator $S_{n,\Delta} \in R^{NxN}$ as

$$\hat{H}_{n+\Delta} = S_{n,\Delta} \hat{H}_n \Rightarrow S_{n,\Delta} = F_n^{\Delta} \quad (23)$$

Next, we may define the differentiation operator $D_n \in R^{NxN}$ as

$$\frac{d}{dt} \hat{H}_n = D_n \hat{H}_n \Rightarrow \hat{H}_{n+1} = e^{D_n} \hat{H}_n \quad (24)$$

Due to (20) we have

$$F_n = e^{D_n} \Rightarrow D_n = \log m(F_n) \quad (25)$$

where $\log m(\cdot)$ denotes matrix logarithm.

Further, by defining the integral operator $I_n \in R^{NxN}$ as

$$\int \hat{H}_n dt = I_n \hat{H}_n \quad (26)$$

International Journal of Computer Science & Emerging Technologies (E-ISSN: 2044-6004)
Volume 1, Issue 4, December 2010

Since the differentiation and integral operator are inverse operators

$$I_n \hat{H}_n = D_n^{-1} \hat{H}_n = \left[\log m(F_n)\right]^{-1} \hat{H}_n \Rightarrow$$
$$I_n = \left[\log m(F_n)\right]^{-1} \quad (27)$$

and the definite integral is yielded as

$$\int_{(n-d)T}^{nT} \hat{H}_n dt$$
$$= \left(\left[\log m(F_n)\right]^{-1} - \left[\log m(F_{n-d})\right]^{-1}\right) \hat{H}_n \quad (28)$$

The interpolating, differentiation and integral operators are commutative, i.e. $S_n D_n = D_n S_n$ and $S_n I_n = I_n S_n$. The computation of the second, third etc. derivatives and integrals of the signals are also possible using the matrix operators, e.g. the second derivative operator is obtained as $D_n^2 = \left[\log m(F_n)\right]^2$. It should be pointed out that applied to the state-space filtered signals the numerical operators are analytic, i.e. they produce results with machine precision.

4. Discussion

The distinct difference between the present algorithm and the SVD based methods is that the present algorithm updates the state transition matrix F_n at every time interval, while the SVD based algorithms [8-9] compute the state transition matrix in data blocks. Our algorithm is more feasible in the analysis of the fastly changing dynamic systems and especially for real-time applications, where the eigenvalues of the state transition matrix give actual information on the system functioning.

A key idea in this work is the repartitioning scheme (13), which yields the uptake of the vector m_{n+1} and the inverse block matrix A_{n+1}^{-1} (14) and then the uptake of the state transition matrix F_{n+1} (16). The companion matrix structure of the matrix F_n enables the computation of the eigenvalues of the state transition matrix via the roots of the polynomial $[1 \;\; -h_n \;\; -k_n]$. This procedure is much faster than the direct eigenvalue decomposition of the F_n matrix. The knowledge of the eigenvalues yields a plenty of numerical signal processing tools, such as interpolation, differentiation and integration operators (21,22,26), which compete with the conventional B-spline signal processing algorithms [10-12].

References

[1] F. Daum, Nonlinear filters: Beyond the Kalman Filter, IEEE A&E Systems Magazine, vol. 20, pp. 57-69, Aug. 2005.

[2] A. Moghaddamjoo and R. Lynn Kirlin, "Robust adaptive Kalman filtering with unknown inputs," IEEE Trans. Acoustics, Speech and Signal Process. vol. 37, No. 8, pp. 1166-1175, Aug. 1989.

[3] J. L. Maryak, J.C. Spall and B.D. Heydon, "Use of the Kalman filter for interference in state-space models with unknown noise distributions," IEEE Trans. Autom. Control, vol, 49, No. 1, pp. 87-90, Sep. 2005.

[4] R. Diversi, R. Guidorzi and U. Soverini, "Kalman filtering in extended noise environments," IEEE Trans. Autom. Control, vol. 50, No. 9, pp. 1396-1402, Sep. 2005

[5] D.-J. Jwo and S.-H. Wang, Adaptive fuzzy strong tracking extended Kalman Filtering for GPS navigation, IEEE Sensors Journal, vol. 7, no. 5, pp. 778-789, May 2007.

[6] S. Attallah, The wavelet transform-domain LMS adaptive filter with partial subband-coefficient updating, IEEE Trans. Circuits and Systems II, vol. 53, no. 1, pp. 8-12, Jan. 2006

[7] H. Olkkonen, P. Pesola, A. Valjakka and L. Tuomisto, "Gain optimized cosine transform domain LMS algorithm for adaptive filtering of EEG," *Comput. Biol. Med*, vol, 29, pp. 129-136, 1999.

[8] S. Park, T.K. Sarkar and Y. Hua, A singular value decomposition-based method for solving a deterministic adaptive problem, Digital Signal processing 9, 57-63, 1999.

[9] T.J. Willink, "Efficient adaptive SVD algorithm for MIMO applications," IEEE Trans. Signal Process., vol. 56, no. 2, pp.615-622, Feb. 2008.

[10] M. Unser, A. Aldroubi and M. Eden, "B-spline signal processing. I. Theory," *IEEE Trans. Signal Process.*, vol. 41, no. 2, pp. 821-833, Feb. 1993.

[11] M. Unser, A. Aldroubi and M. Eden, "B-spline signal processing. II. Efficiency design and applications," *IEEE Trans. Signal Process.*, vol. 41, No. 2, pp. 834-848, Feb. 1993.

[12] J.T. Olkkonen and H. Olkkonen, "Fractional time-shift B-spline filter," IEEE Signal Process. Letters**,** vol. 14, No. 10, pp. 688-691, Oct. 2007.