International Journal of Control
Vol. 86, No. 2, February 2013, 309–321

Including Errata:
Equations (6.10), (6.11), (6.14), (6.22).
Mapping current paper onto De Koning (1992):
$X_1 = S,\ X_2 = \hat{S},\ X_3 = P,\ X_4 = \hat{P}.$

Taylor & Francis
Taylor & Francis Group

Minimal representation of matrix valued white stochastic processes and U–D factorisation of algorithms for optimal control

L. Gerard Van Willigenburg[a]* and Willem L. De Koning[b]

[a]*Systems and Control Group of Wageningen University, PO Box 17, 6700 AA Wageningen, The Netherlands;*
[b]*Department of Mathematics of Delft University of Technology, Kroeskarper 6, Leiden, The Netherlands*

(*Received 1 February 2012; final version received 28 August 2012*)

Two different descriptions are used in the literature to formulate the optimal dynamic output feedback control problem for linear dynamical systems with white stochastic parameters and quadratic criteria, called the optimal compensation problem. One describes the matrix valued white stochastic processes involved, using a sum of deterministic matrices each one multiplied by a scalar stochastic process that is independent of the others. Another, that is more general and concise, uses Kronecker products instead. This article relates the statistics of both descriptions and shows their advantages and disadvantages. As to the first description, an important result that comes out is the minimum number of matrices multiplied by scalar, independent, stochastic processes needed to represent a certain matrix valued white stochastic process, together with an associated minimal representation. As to the second description, an important result concerns the generation of all Kronecker products that represent relevant statistics. Both results facilitate the specification of statistics of systems with white stochastic parameters. The second part of this article further exploits these results to perform an U–D factorisation of an algorithm to compute optimal dynamic output feedback controllers (optimal compensators) for linear discrete-time systems with white stochastic parameters and quadratic sum criteria. U–D factorisation of this type of algorithm is new. By solving several numerical examples, the U–D factored algorithm is compared with a conventional algorithm.

Keywords: minimal representation; matrix valued white stochastic processes; multiplicative white noise; stochastic parameters; optimal compensation; compensatability; UDU factorisation

1. Introduction

White stochastic processes are often used in engineering to describe uncertainty, such as uncertainty in dynamical system descriptions. White stochastic processes are among the simplest stochastic processes because they have no memory, in the sense that past values do not provide additional information on future values. Using them is a way to communicate that one is highly ignorant about the uncertainty and that any form of estimating the future uncertainty, using past values, is useless. The latter simplifies mathematical solutions. Also, after thorough investigation, this is the type of uncertainty one is generally left with.

Vector-valued stochastic processes are the ones generally used in dynamical system descriptions to represent uncertainty, because the state of a dynamical system is generally represented by a vector. Then additive uncertainty is represented by a vector stochastic process. Since it often concerns uncertainty one is highly ignorant about, and to simplify the mathematics, white vector stochastic processes are generally used. For the same reasons, one usually considers the first and second moments of the vector stochastic process only. They provide information about the mean value and average error, which is the main engineering purpose (Athans 1971).

It is well known (Arnold 1971) that the mean and average squared errors of n-dimensional continuous-time white vector stochastic processes are represented by the mean vector of dimension n and an $n \times n$ intensity matrix that is non-negative symmetric. For an n-dimensional discrete-time white vector stochastic process, the $n \times n$ intensity matrix becomes a $n \times n$ covariance matrix that is also non-negative symmetric. The symmetry and non-negativeness characterise all possible intensity and covariance matrices whereas the components of the mean vector can take on any value.

White additive uncertainty cannot destabilise a linear system, indicating that it is a rather weak type of uncertainty (Athans 1971). White multiplicative uncertainty can, and therefore may be considered a stronger type of uncertainty (De Koning 1982). White multiplicative uncertainty is obtained by turning deterministic system parameters into white stochastic processes. One important application area is the control of

*Corresponding author. Email: gerard.vanwilligenburg@wur.nl

ISSN 0020–7179 print/ISSN 1366–5820 online

http://dx.doi.org/10.1080/00207179.2012.725866
http://www.tandfonline.com

computer networks where stochastic parameters are often used to describe uncertainty caused by stochastic sampling (De Koning and Van Willigenburg, 2001; Immer, Yükselb, and Basar 2006; Antunes, Hespanha, and Silvestre 2009; Kögel, Blind, Allgöwer, and Findeisen 2011; Li, Zhoua, and Wub 2011). Other practical and more theoretically oriented applications may be found in Fujimoto, Ota and Nakayama (2011), Hounkpevi and Yaz (2008), Meenakshi and Bhat (2006) and Boje (2005). When parameters are taken to be stochastic variables, the system matrices of linear dynamical systems turn into matrix valued stochastic processes. The first moment of those is a deterministic matrix containing the mean of each entry. The second moment is described using Kronecker products of matrices (De Koning 1982, 1992). As opposed to vector valued white stochastic processes, where the second moment is characterised by intensity and covariance matrices being non-negative symmetric, the second moment of white matrix valued stochastic processes is not as easily characterised. Probably due to this, two different representations of white matrix valued stochastic processes are used in the literature, when describing linear dynamical systems with multiplicative white noise. One uses Kronecker products to represent the second moment (De Koning 1982, 1992). The other uses a sum of deterministic matrices each multiplied with a scalar white stochastic process that is independent of the others (Bernstein and Haddad 1987). This representation raises the issue of minimal representation: 'What is the minimum number of deterministic matrices and scalar white stochastic processes that represents a certain second moment, and how should they be selected?'

This article solves the minimal representation problem stated above, using vector valued white stochastic processes as an intermediate, to link the two descriptions of matrix valued white stochastic processes. This also provides a method to easily generate all Kronecker products that represent second moments of matrix valued white stochastic processes. The key computation in finding the minimal representation is a generalised Cholesky decomposition of a certain matrix, that is non-negative symmetric.

In the second part of this article, the minimal representation turns out to be the key in performing U–D factorisation of an algorithm that computes the unique optimal full-order dynamic output feedback controller (optimal compensator) for discrete-time linear time-invariant systems, with additive and multiplicative white noise, given an infinite horizon quadratic sum criterion. By solving several numerical examples the U–D factored algorithm, which relies on the representation that uses independent scalar stochastic processes, is compared with a conventional algorithm, that relies on the Kronecker product representation.

2. Linear systems with white stochastic parameters

Since our main research interest concerns digital optimal control system design, in this article we will consider discrete-time linear systems with white stochastic parameters (multiplicative white noise). In addition to the multiplicative white noise, the discrete-time linear systems are also corrupted by additive white system and measurement noise. These discrete-time systems are described by,

$$x_{i+1} = \Phi_i x_i + \Gamma_i u_i + v_i, \quad i = 0, 1, \ldots, \tag{2.1}$$

$$y_i = C_i x_i + w_i, \quad i = 0, 1, \ldots. \tag{2.2}$$

In Equations (2.1) and (2.2), $x_i \in R^n$ represents the system state, $u_i \in R^m$ the control inputs and $y_i \in R^l$ the observations at time $i = 0, 1, \ldots$. Furthermore, v_i is the discrete-time zero-mean additive white system noise, with covariance $V_i \in R^{n\times n}$, and w_i the discrete-time zero-mean additive white measurement noise with covariance $W_i, i = 0, 1, \ldots$. Because the discrete-time system has white stochastic parameters, at each discrete-time instant $i = 0, 1, \ldots$, the system matrices Φ_i, Γ_i, C_i have entries that instead of deterministic, are white stochastic variables. Accordingly, the processes $\{\Phi_i, i = 0, 1, \ldots\}$, $\{\Gamma_i, i = 0, 1, \ldots\}$, $\{C_i, i = 0, 1, \ldots\}$ become matrix valued white stochastic processes. Again their first and second moments will be the only ones considered for control system design. Representing these first and second moments will be the topic of the next two sections. Although in this article, we restrict ourselves to discrete-time linear systems with white stochastic parameters, all results concerning stochastic matrices carry over to system matrices of continuous-time linear systems with white stochastic parameters. In that case covariance matrices must be identified with intensity matrices.

3. Second moments represented by Kronecker products

Reconsider the discrete-time matrix valued white stochastic process $\{\Phi_i, i = 0, 1, \ldots\}$ introduced in the previous section. Assume it has time-varying first and second moments. The first moment of the process is defined by $\bar{\Phi}_i$. Let,

$$\tilde{\Phi}_i = \Phi_i - \bar{\Phi}_i, \tag{3.1}$$

and consider the decomposition,

$$\Phi_i = \bar{\Phi}_i + \tilde{\Phi}_i. \tag{3.2}$$

Then the second moment of the process is defined by,

$$\overline{\Phi_i \otimes \Phi_i} = \bar{\Phi}_i \otimes \bar{\Phi}_i + \overline{\tilde{\Phi}_i \otimes \tilde{\Phi}_i}. \tag{3.3}$$

with,

$$\overline{\tilde{\Phi}_i \otimes \tilde{\Phi}_i} = V_i^{\Phi\Phi} \in R^{n^2 \times n^2} \tag{3.4}$$

representing covariances. Similarly, the first moment of $\{\Gamma_i, i = 0, 1, \ldots\}$ and $\{C_i, i = 0, 1, \ldots\}$ is defined by $\bar{\Gamma}_i(t)$, $\bar{C}_i(t)$ and the second moment by,

$$\overline{\Gamma_i \otimes \Gamma_i} = \bar{\Gamma}_i \otimes \bar{\Gamma}_i + \overline{\tilde{\Gamma}_i \otimes \tilde{\Gamma}_i} \tag{3.5}$$

$$\overline{C_i \otimes C_i} = \bar{C}_i \otimes \bar{C} + \overline{\tilde{C}_i \otimes \tilde{C}_i} \tag{3.6}$$

with

$$\overline{\tilde{\Gamma}_i \otimes \tilde{\Gamma}_i} = V_i^{\Gamma\Gamma} \in R^{n^2 \times m^2} \tag{3.7}$$

$$\overline{\tilde{C}_i \otimes \tilde{C}_i} = V_i^{CC} \in R^{l^2 \times n^2} \tag{3.8}$$

representing covariances. If the processes $\{\Phi_i, i = 0, 1, \ldots\}$, $\{\Gamma_i, i = 0, 1, \ldots\}$ are correlated at time i, this is described by,

$$\overline{\tilde{\Phi}_i \otimes \tilde{\Gamma}_i} = V_i^{\Phi\Gamma} \in R^{n^2 \times nm} \tag{3.9}$$

being non-zero. Similarly if the processes $\{\Phi_i, i = 0, 1, \ldots\}$, $\{C_i, i = 0, 1, \ldots\}$ are correlated at time i, this is described by,

$$\overline{\tilde{\Phi}_i \otimes \tilde{C}_i} = V_i^{\Phi C} \in R^{n^2 \times nl} \tag{3.10}$$

being non-zero. The specification of the first moments $\bar{\Phi}_i$ is straightforward because they are matrices, the entries of which are mean values of the associated stochastic variables. The specification of the second moment (3.3) is less straightforward. This is mainly due to the Kronecker product $V_i^{\Phi\Phi} = \overline{\tilde{\Phi}_i \otimes \tilde{\Phi}_i}$. The properties $V_i^{\Phi\Phi}$ must have to represent covariances of Φ_i are not easily recognised.

4. Second moments represented by sums of matrices multiplied by scalar stochastic processes

Consider the decomposition (3.2) of the discrete-time matrix valued white stochastic process $\{\Phi_i, i = 0, 1, \ldots\}$ and similarly for $\{\Gamma_i, i = 0, 1, \ldots\}$, $\{C_i, i = 0, 1, \ldots\}$. Let,

$$\begin{aligned} \tilde{\Phi}_i &= \sum_{j=1}^{r} \Phi_{i,j}\eta_{i,j}, \quad \tilde{\Gamma}_i = \sum_{j=1}^{r} \Gamma_{i,j}\eta_{i,j}, \\ \tilde{C}_i &= \sum_{j=1}^{r} C_{i,j}\eta_{i,j}, \quad i = 0, 1, \ldots, \end{aligned} \tag{4.1}$$

where $\Phi_{i,j}$, $\Gamma_{i,j}$, $C_{i,j}$, are the known deterministic matrix functions and $\eta_{i,j}$ the independent, scalar, zero-mean, unit variance, white stochastic processes,

$$\overline{\eta_{i,j}} = 0, \quad \overline{\eta_{i,j}\eta_{i,j}} = 1. \tag{4.2}$$

Equation (4.1) is a description that uses a sum of matrices multiplied by scalar stochastic processes. From Equations (4.1) and (4.2), the second moment (3.3) of $\{\Phi_i, i = 0, 1, \ldots\}$ is fully specified by the first moment $\bar{\Phi}_i$ and the covariance matrix,

$$\overline{\tilde{\Phi}_i \otimes \tilde{\Phi}_i} = V_i^{\Phi\Phi} = \sum_{j=1}^{r} \Phi_{i,j} \otimes \Phi_{i,j}. \tag{4.3}$$

Similarly,

$$\overline{\tilde{\Gamma}_i \otimes \tilde{\Gamma}_i} = V_i^{\Gamma\Gamma} = \sum_{j=1}^{r} \Gamma_{i,j} \otimes \Gamma_{i,j}, \tag{4.4}$$

$$\overline{\tilde{C}_i \otimes \tilde{C}_i} = V_i^{CC} = \sum_{j=1}^{r} C_{i,j} \otimes C_{i,j}, \tag{4.5}$$

while cross correlations are described by,

$$\overline{\tilde{\Phi}_i \otimes \tilde{\Gamma}_i} = V_i^{\Phi\Gamma} = \sum_{j=1}^{r} \Phi_{i,j} \otimes \Gamma_{i,j}, \tag{4.6}$$

$$\overline{\tilde{\Phi}_i \otimes \tilde{C}_i} = V_i^{\Phi C} = \sum_{j=1}^{r} \Phi_{i,j} \otimes C_{i,j}, \tag{4.7}$$

$$\overline{\tilde{\Gamma}_i \otimes \tilde{C}_i} = V_i^{\Gamma C} = \sum_{j=1}^{r} \Gamma_{i,j} \otimes C_{i,j}. \tag{4.8}$$

In the equations stated so far, r is the *total* number of independent scalar stochastic processes needed to describe covariances of Φ_i, Γ_i and C_i. To simplify the presentation, and because it enables U–D factorisation, from now on all matrix valued stochastic processes will be considered independent. As a result, each matrix valued stochastic process is described by its own number of scalar stochastic processes, that are independent of the others. Then r should be interpreted as a number associated with the single matrix valued stochastic process under consideration. If Γ is such a matrix valued stochastic process, from now on, we will denote the associated r by r_Γ.

5. Kronecker products and minimal representations of second moments

To find Kronecker products and minimal representations representing covariances of matrix valued white

stochastic processes, it is convenient to convert white matrix valued stochastic processes to white vector valued stochastic processes. This is because for white vector valued processes symmetry and non-negativeness are two 'easy' properties fully characterising covariance matrices, as stated by Lemma 1 in this section. The conversion is performed by *stacking* the columns of the white matrix valued stochastic process,

$$\phi_i^s = st(\Phi_i) \in R^{n^2\times 1}, \quad i = 0, 1, \ldots, \tag{5.1}$$

where *st* denotes the stack operator (which is also known as the *vec* operator) and $\{\phi_i^s, i = 0, 1, \ldots\}$ a white vector stochastic process of dimension $n^2 \times 1$. We also need to consider the inverse operation of (5.1),

$$\Phi_i = st^{-1}\left(\phi_i^s\right), \quad i = 0, 1, \ldots, \tag{5.2}$$

where st^{-1} denotes the unstack operator. After the stacking (5.1), we can specify covariance matrices $V_i^{\phi^s\phi^s}$ for the vector stochastic process $\{\phi_i^s, i = 0, 1, \ldots\}$,

$$\overline{\tilde{\phi}_i^s\left(\tilde{\phi}_i^s\right)^T} = V_i^{\phi^s\phi^s} \in R^{n^2\times n^2} \tag{5.3}$$

Similarly for $\gamma_i^s = st(\Gamma_i), c_i^s = st(C_i)$,

$$\overline{\tilde{\gamma}_i^s\left(\tilde{\gamma}_i^s\right)^T} = V_i^{\gamma^s\gamma^s} \in R^{nm\times nm}, \quad i = 0, 1, \ldots, \tag{5.4}$$

$$\overline{\tilde{c}_i^s\left(\tilde{c}_i^s\right)^T} = V_i^{c^sc^s} \in R^{nl\times nl}, \quad i = 0, 1, \ldots, \tag{5.5}$$

Lemma 1: *The covariance of a discrete-time white vector stochastic process $h_i \in R^n$, $i = 0, 1, \ldots$ at time i is represented by a non-negative symmetric matrix $H_i \in R^{n\times n}$. Conversely, any non-negative symmetric matrix $H_i \in R^{n\times n}$ may be interpreted as the covariance of a certain discrete-time white vector stochastic process $h_i \in R^n$, $i = 0, 1, \ldots$ at time i.*

Proof: Let $\tilde{h}_i = h_i - \bar{h}_i$. Then the first part follows directly from $H_i = \overline{\tilde{h}_i\tilde{h}_i^T}$. Consider any $n \times n$ symmetric matrix $H_i \geq 0$. Then the symmetric square root $H_i^{1/2}$ exists. Let h_i' be a stochastic vector of dimension n having covariance $\overline{\tilde{h}_i'(\tilde{h}_i')^T} = I$ where I denotes the identity matrix. Then the stochastic vector $h_i = H_i^{1/2}h_i'$ has covariance $\overline{\tilde{h}_i(\tilde{h}_i)^T} = \overline{H_i^{1/2}\tilde{h}_i'(H_i^{1/2}\tilde{h}_i')^T} = \overline{H_i^{1/2}\tilde{h}_i'(\tilde{h}_i')^TH_i^{1/2}} = H_i^{1/2}IH_i^{1/2} = H_i$.

According to Lemma 1, all matrices representing covariances of vector stochastic processes are precisely those that are nonnegative symmetric.

Theorem 1:

(1) *There exists the following one to one mapping of elements of $V_i^{\gamma^s\gamma^s} \in R^{nm\times nm}$, specified by* (5.4), *to those of $V_i^{\Gamma\Gamma} \in R^{n^2\times m^2}$, specified by* (3.7). *For $j_1 = 1, 2, \ldots, n$, $j_2 = 1, 2, \ldots, m$, $j_3 = 1, 2, \ldots, n$, $j_4 = 1, 2, \ldots, m$ define,*

$$i_1 = (j_2 - 1)n + j_1, \quad i_2 = (j_4 - 1)n + j_3 \tag{5.6}$$

$$i_3 = (j_1 - 1)n + j_3, \quad i_4 = (j_2 - 1)n + j_4 \tag{5.7}$$

Then for each combination j_1, j_2, j_3, j_4 the corresponding element i_1, i_2 of $V_i^{\gamma^s\gamma^s}$ equals element i_3, i_4 of $V_i^{\Gamma\Gamma} \in R^{n^2\times m^2}$.

(2) *All matrices $V_i^{\Gamma\Gamma} \in R^{n^2\times m^2}$ representing covariances in* (3.7) *map one to one on all non-negative symmetric matrices $V_i^{\gamma^s\gamma^s} \in R^{nm\times nm}$.*

Proof:

(1) Let γ_{i,i_1} indicate component i_1 of the column vector γ_i and let Γ_{i,j_1,j_2} denote element j_1, j_2 of matrix Γ_i. Consider the covariance matrix $V_i^{\gamma^s\gamma^s}$ in (5.4). Element i_1, i_2 of this matrix represents the cross covariance of γ_{i,i_1}^s and γ_{i,i_2}^s. Because $\gamma_i^s = st(\Gamma_i)$, this is the cross covariance of Γ_{i,j_1,j_2} and Γ_{i,j_3,j_4} when Equation (5.6) applies. Applying the definition by means of Kronecker products (3.7), the cross covariance of Γ_{i,j_1,j_2} and Γ_{i,j_3,j_4} is represented by element i_3, i_4 of $V_i^{\Gamma\Gamma}$, if Equation (5.7) applies. (2) Follows from (1) in Theorem 1 and Lemma 1.

Observe that Theorem 1 applies also to the pairs $V_i^{\phi^s\phi^s}$, $V_i^{\Phi\Phi}$ and $V_i^{c^sc^s}$, V_i^{CC}.

Consider the representation (4.4) of $V_i^{\Gamma\Gamma}$ by $\Gamma_{i,j}$, $j = 1, 2, \ldots, r_\Gamma$. Let,

$$\gamma_{i,j}^s = st\left(\Gamma_{i,j}\right), \quad j = 1, 2, \ldots, r_\Gamma, \quad i = 0, 1, \ldots \tag{5.8}$$

Then $V_i^{\gamma^s\gamma^s}$ that corresponds with $V_i^{\Gamma\Gamma}$ is given by,

$$V_i^{\gamma^s\gamma^s} = \sum_{j=1}^{r_\Gamma} \gamma_{i,j}^s\left(\gamma_{i,j}^s\right)^T. \tag{5.9}$$

From Equation (5.9), observe that the $nm \times nm$ matrix $V_i^{\gamma^s\gamma^s}$ is obtained as a sum of column vectors each one multiplied by its transpose. The number of column vectors equals r_Γ. Noting that each $nm \times nm$ matrix $\gamma_{i,j}^s(\gamma_{i,j}^s)^T$ in (5.9) has rank 1 and is non-negative symmetric, it is clear that (5.9) produces non-negative symmetric matrices $V_i^{\gamma^s\gamma^s}$. Moreover, to represent a certain non-negative symmetric $V_i^{\gamma^s\gamma^s}$, r_Γ needs to be at least $rank(V_i^{\gamma^s\gamma^s})$. This raises the question of minimal representation: whether we can always find $\gamma_{i,j}^s$, $j = 1, 2, \ldots, r_\Gamma$ in (5.9) realising any non-negative symmetric $V_i^{\gamma^s\gamma^s}$ having rank r_Γ.

Theorem 2: *Any non-negative symmetric matrix $V_i^{\gamma^s\gamma^s} \in R^{nm\times nm}$, having rank r_Γ, can be represented by*

Equation (5.9), *i.e.* $V_i^{\gamma^s\gamma^s} = \sum_{j=1}^{r_\Gamma} \gamma_{i,j}^s (\gamma_{i,j}^s)^T$, *taking* $\gamma_{i,j}^s \in R^{nm\times 1}$, $j = 1, 2, \ldots, r_\Gamma$ *to be the* r_Γ *non-zero columns of an upper triangular matrix* $U \in R^{nm\times nm}$ *obtained from the following generalised Cholesky decomposition*,

$$V_i^{\gamma^s\gamma^s} = U_i U_i^T, \quad V_i^{\gamma^s\gamma^s}, U_i \in R^{nm\times nm}. \tag{5.10}$$

This representation is a minimal representation. An associated minimal representation (4.4), *i.e.* $V_i^{\Gamma\Gamma} = \sum_{j=1}^{r_\Gamma} \Gamma_{i,j} \otimes \Gamma_{i,j}$, *is obtained with* $\Gamma_{i,j} = st^{-1}(\gamma_{i,j}^s), j = 1, 2, \ldots, r_\Gamma$.

Proof: U_i in Equation (5.10) is an upper triangular matrix square root having $r_\Gamma = rank(V_i^{\gamma^s\gamma^s})$ non-zero columns denoted by $U'_{i,j}, j = 1, 2, \ldots, r_\Gamma$. An algorithm for computing the matrix square root is presented in Bierman (1977, p. 53). This algorithm is designed for matrices that are positive. The generalisation of this algorithm to non-negative matrices is easily obtained. The algorithm contains a parameter α indicating rank deficiency when it grows very large. In that case, all elements multiplied by α in the algorithm should be set to zero. From Equation (5.10), it follows that Equation (5.9) is satisfied because,

$$U_i U_i^T = \sum_{j=1}^{nm} U_{i,j} U_{i,j}^T = \sum_{j=1}^{r_\Gamma} U'_{i,j} U'^T_{i,j} = \sum_{j=1}^{r_\Gamma} \gamma_{i,j}^s \left(\gamma_{i,j}^s\right)^T. \tag{5.11}$$

The representation $\sum_{j=1}^{r_\Gamma} \gamma_{i,j}^s (\gamma_{i,j}^s)^T$ in (5.11) is minimal because $r_\Gamma = rank(V_i^{\gamma^s\gamma^s})$. Due to (5.8) i.e. $\gamma_{i,j}^s = st(\Gamma_{i,j})$ taking,

$$\Gamma_{i,j} = st^{-1}\left(\gamma_{i,j}^s\right), \quad j = 1, 2, \ldots, r_\Gamma, \tag{5.12}$$

provides an associated minimal representation (4.4).

In summary, starting from $V_i^{\gamma^s\gamma^s}$, by computing the generalised Cholesky factorisation (5.10) and taking the $r_\Gamma = rank(V_i^{\gamma^s\gamma^s})$ non-zero columns of U_i in (5.11), a minimal representation (5.9) is obtained. From this minimal representation application of (5.12) provides an associated minimal representation (4.4) of $V_i^{\Gamma\Gamma}$. Starting from $V_i^{\gamma^s\gamma^s}$ implies that we have to consider $\gamma_i^s = st(\Gamma_i)$ when specifying covariances associated with Γ_i. This greatly simplifies specifying meaningful covariances because, according to Lemma 1 and Theorem 1, these correspond one to one with non-negative symmetric matrices. According to Theorem 2 associated minimal representations (4.4) of $V_i^{\Gamma\Gamma}$ are easily computed by the one to one mapping described in Theorem 1. In a similar manner, starting from $V_i^{\phi^s\phi^s}$, $V_i^{c^sc^s}$ minimal representations of $V_i^{\Phi\Phi}$ and V_i^{CC} are easily computed.

6. U–D factored discrete-time optimal compensation algorithms

6.1 *Compensatability and the optimal compensation problem*

For motivation and details of the optimal compensation problem considered in this section refer to De Koning (1992). In this section, we only present a summary involving all problem data needed to solve the problem. Consider the discrete-time linear system (2.1) and (2.2). Besides having white stochastic parameters, this system is also corrupted by additive discrete-time white system and measurement noise $\{v_i, i = 0, 1, \ldots\}$, $\{w_i, i = 0, 1, \ldots\}$. The mutually independent processes $\{\Phi_i, i = 0, 1, \ldots\}$, $\{\Gamma_i, i = 0, 1, \ldots\}$, $\{C_i, i = 0, 1, \ldots\}$ are sequences of independent random matrices with *constant* statistics and $\{v_i, i = 0, 1, \ldots\}$, $\{w_i, i = 0, 1, \ldots\}$ are sequences of independent stochastic vectors with *constant* statistics. Φ_i, Γ_i and C_i, are independent of v_j and w_j, $i \neq j$ and uncorrelated with v_i, w_i. Because the statistics are constant, the time index i is dropped from their specification. The processes $\{v_i, i = 0, 1, \ldots\}$, $\{w_i, i = 0, 1, \ldots\}$ are zero-mean with covariance matrices $V \geq 0$, $W \geq 0$ and cross covariance matrix Y, $\left[\begin{smallmatrix} V & Y \\ Y^T & W \end{smallmatrix}\right] \geq 0$. Consider the dynamic output feedback compensator,

$$\hat{x}_{i+1} = F\hat{x}_i + Ky_i, \tag{6.1}$$

$$u_i = -L\hat{x}_i, \quad i = 0, 1, \ldots, \tag{6.2}$$

and the associated closed loop system,

$$\begin{bmatrix} x_{i+1} \\ \hat{x}_{i+1} \end{bmatrix} = \begin{bmatrix} \Phi_i & -\Gamma_i L \\ KC_i & F \end{bmatrix} \begin{bmatrix} x_i \\ \hat{x}_i \end{bmatrix}, \quad i = 0, 1, \ldots. \tag{6.3}$$

Introduce,

$$x'_i = \begin{bmatrix} x_{i+1} \\ \hat{x}_{i+1} \end{bmatrix}, \quad \Phi'_i = \begin{bmatrix} \Phi_i & -\Gamma_i L \\ KC_i & F \end{bmatrix}. \tag{6.4}$$

Then the closed loop system is also represented by,

$$x'_{i+1} = \Phi'_i x'_i, \quad i = 0, 1, \ldots \tag{6.5}$$

Let ρ denote spectral radius. From De Koning (1982, 1992), the closed loop system (6.5) is *mean-square stable (ms-stable)* if,

$$\rho\left(\overline{\Phi' \otimes \Phi'}\right) < 1. \tag{6.6}$$

If for the system (2.1) and (2.2), there exist a compensator (6.1) and (6.2) such that the closed loop system (6.5) is ms-stable the system (2.1) and (2.2) is called *mean-square compensatable* (*ms-compensatable*). Such a compensator is called *mean-square stabilising (ms-stabilising)*.

Let E denote expectation. The optimal compensation problem is to find the compensator matrices F^*, K^*, L^* of an ms-stabilising compensator that minimises the infinite horizon quadratic sum criterion,

$$\sigma_\infty(F,K,L)=\lim_{N\to\infty}\frac{1}{N}E\left\{\sum_{i=1}^{N}\begin{bmatrix}x_i & u_i\end{bmatrix}\begin{bmatrix}Q & M\\ M^T & R\end{bmatrix}\begin{bmatrix}x_i\\ u_i\end{bmatrix}\right\}, \tag{6.7}$$

and to find the associated minimum costs $\sigma^*_\infty(F^*,K^*,L^*)$. The following problem data entirely determine the solution of the optimal compensation problem (De Koning 1992),

$$\begin{aligned}&\bar{\Phi},\overline{\tilde{\Phi}\otimes\tilde{\Phi}}=V^{\Phi\Phi},\bar{\Gamma},\overline{\tilde{\Gamma}\otimes\tilde{\Gamma}}=V^{\Gamma\Gamma},\\ &\bar{C},\overline{\tilde{C}\otimes\tilde{C}}=V^{CC},Q,R,M,V,W,Y.\end{aligned} \tag{6.8}$$

In Equation (6.8), $\bar{\Phi},\bar{\Gamma},\bar{C}$ are the mean values of Φ_i, Γ_i, C_i while $\overline{\tilde{\Phi}\otimes\tilde{\Phi}}$ represents covariances of Φ_i, $\overline{\tilde{\Gamma}\otimes\tilde{\Gamma}}$ represents covariances of Γ_i and $\overline{\tilde{C}\otimes\tilde{C}}$ represents covariances of C_i. The second moment $\overline{\Phi\otimes\Phi}$ of Φ_i satisfies (De Koning 1992),

$$\overline{\Phi\otimes\Phi}=\bar{\Phi}\otimes\bar{\Phi}+\overline{\tilde{\Phi}\otimes\tilde{\Phi}}. \tag{6.9}$$

Similar relations hold for the second moments of Γ_i and C_i. Therefore (6.8), fully determines the first and second moments of Φ_i, Γ_i and C_i.

6.2 *Algorithms for compensatability and optimal compensation*

The U-D factored algorithms presented in this section are based on the ones presented in De Koning (1992). We adopt, as much as possible, the same notation. The algorithms hold for the case $M=\theta$, $Y=\theta$ where θ denotes a zero matrix. The algorithms solve four coupled matrix equations, two of them being generalized Riccati equations and two of them being generalized Lyapunov equations. For convenience these four coupled matrix equations are captured in a single transformation. Let S^n denote the set of nonnegative symmetric real $n\times n$ matrices. Let $X=\{X_1,X_2,X_3,X_4\}$, $X_1,X_2,X_3,X_4\in S^n$ and define,

$$K_X=\bar{\Phi}X_3\bar{C}^T\left(\overline{CX_3C^T}+\overline{\tilde{C}X_4\tilde{C}^T}+W\right)^{\dagger}, \tag{6.10}$$

$$L_X=\left(\overline{\Gamma^TX_1\Gamma}+\overline{\tilde{\Gamma}^TX_2\tilde{\Gamma}}+R\right)^{\dagger}\bar{\Gamma}^TX_1\bar{\Phi}, \tag{6.11}$$

$$F_X=\bar{\Phi}-\bar{\Gamma}L_X-K_X\bar{C}, \tag{6.12}$$

where $\dagger$ denotes a generalised inverse of a square matrix having the properties,

$$AA^{\dagger}A=A,\quad A^{\dagger}AA^{\dagger}=A^{\dagger} \tag{6.13}$$

Next define the nonlinear transformation C: $S^n\times S^n\times S^n\times S^n\to S^n\times S^n\times S^n\times S^n$ by,

$$CX=\begin{pmatrix}\overline{(\Phi-\Gamma L_X)^TX_1(\Phi-\Gamma L_X)}+\overline{\left(\tilde{\Phi}-K_X\tilde{C}\right)^TX_2\left(\tilde{\Phi}-K_X\tilde{C}\right)}\\ +L_X^T\left(\overline{\tilde{\Gamma}^TX_2\tilde{\Gamma}}+R\right)L_i+Q,\\ \left(\bar{\Phi}-K_X\bar{C}\right)^TX_2\left(\bar{\Phi}-K_X\bar{C}\right)+L_X^T\left(\overline{\Gamma^TX_1\Gamma}+\overline{\tilde{\Gamma}^TX_2\tilde{\Gamma}}+R\right)L_X,\\ \overline{(\Phi-K_XC)X_3(\Phi-K_XC)^T}+\overline{\left(\tilde{\Phi}-\tilde{\Gamma}L_X\right)X_4\left(\tilde{\Phi}-\tilde{\Gamma}L_X\right)^T}\\ +K_X\left(\overline{\tilde{C}X_4\tilde{C}^T}+W\right)K_X^T+V,\\ \left(\bar{\Phi}-\bar{\Gamma}L_X\right)X_4\left(\bar{\Phi}-\bar{\Gamma}L_X\right)^T+K_X\left(\overline{CX_3C^T}+\overline{\tilde{C}X_4\tilde{C}^T}+W\right)K_X^T\end{pmatrix} \tag{6.14}$$

The algorithms consist of repeated execution of the transformation C in (6.14). The same transformation C is specified in De Koning (1992) using a different, more convenient, representation. However, that representation is not suitable for U–D factorisation. The equivalent representation (6.14) is as demonstrated in the next section. To compute the transformation (6.14), we have to apply repeatedly three computational rules. An example of the first rule is given by Equation (6.9). Examples of the second and third rule are stated below,

$$\begin{aligned}\overline{\Phi^TX_1\Phi}&=st^{-1}\left(\overline{\Phi\otimes\Phi}^T\,st(X_1)\right),\\ \overline{\Phi X_3\Phi^T}&=st^{-1}\left(\overline{\Phi\otimes\Phi}\,st(X_3)\right),\end{aligned} \tag{6.15}$$

$$\begin{aligned}\overline{\tilde{\Gamma}L_X\otimes\tilde{\Gamma}L_X}&=\overline{\tilde{\Gamma}\otimes\tilde{\Gamma}}(L_X\otimes L_X),\\ \overline{K_X\tilde{C}\otimes K_X\tilde{C}}&=(K_X\otimes K_X)\overline{\tilde{C}\otimes\tilde{C}}.\end{aligned} \tag{6.16}$$

As an example, we show how to compute the first term in Equation (6.14),

$$\overline{(\Phi-\Gamma L_X)^TX_1(\Phi-\Gamma L_X)}. \tag{6.17}$$

Using the three computational rules, (6.15), (6.9), (6.16), respectively, we find,

$$\begin{aligned}&\overline{(\Phi-\Gamma L_X)^TX_1(\Phi-\Gamma L_X)}\\ &\quad=\left(\bar{\Phi}-\bar{\Gamma}L_X\right)^TX_1\left(\bar{\Phi}-\bar{\Gamma}L_X\right)\\ &\qquad+\overline{(\Phi-\widetilde{\Gamma L_X})^TX_1(\Phi-\widetilde{\Gamma L_X})}\end{aligned} \tag{6.18}$$

$$\begin{aligned}&\overline{(\Phi-\widetilde{\Gamma L_X})^TX_1(\Phi-\widetilde{\Gamma L_X})}\\ &\quad=st^{-1}\left(\overline{(\Phi-\widetilde{\Gamma L_X})\otimes(\Phi-\widetilde{\Gamma L_X})}^T\,st(X_1)\right)\end{aligned} \tag{6.19}$$

$$\begin{aligned}&\overline{(\Phi-\widetilde{\Gamma L_X})\otimes(\Phi-\widetilde{\Gamma L_X})}\\ &\quad=\overline{\tilde{\Phi}\otimes\tilde{\Phi}}+\overline{\tilde{\Gamma}L_X\otimes\tilde{\Gamma}L_X}\\ &\quad=\overline{\tilde{\Phi}\otimes\tilde{\Phi}}+\overline{\tilde{\Gamma}\otimes\tilde{\Gamma}}(L_X\otimes L_X).\end{aligned} \tag{6.20}$$

First, the right hand side of Equation (6.20) is computed using the problem data (6.8) and L_X. Then, using (6.19), the right hand side of (6.18) is computed. In a similar manner, starting from the problem data (6.8), all other terms of the transformation (6.14) can be computed. The computation of L_X, specified by (6.11), and K_X, specified by (6.10), is considered in the next section.

The algorithms comprise two constructive ms-compensatability tests, one of them producing a *measure* of compensatability, and an algorithm to compute the optimal compensator. Let θ denote a square zero matrix and I the identity matrix, both having compatible dimensions.

Compensatability test 1 (De Koning 1992)
Choose $Q = V = I$, $R = \theta$, $W = \theta$. Then the system (2.1) and (2.2), with the properties described at the start of this section, is ms-compensatable $\Leftrightarrow C^i(\theta, I, \theta, I)$ converges as $i \to \infty$.

Compensatability test 2 (De Koning 1992)
Choose $Q = V = \theta$, $R = \theta$, $W = \theta$. Let $(X_{1,i}, X_{2,i}, X_{3,i}, X_{4,i}) = C^i(\theta, I, \theta, I)$. Then the system (2.1) and (2.2), with the properties described at the start of this section, is ms-compensatable $\Leftrightarrow \lim_{i\to\infty} \left[\frac{\mathrm{tr}(X_{1,i+1}+X_{3,i+1})}{\mathrm{tr}(X_{1,i}+X_{3,i})}\right] < 1$.

As explained in De Koning (1992) $\lim_{i\to\infty}\left[\frac{\mathrm{tr}(X_{1,i+1}+X_{3,i+1})}{\mathrm{tr}(X_{1,i}+X_{3,i})}\right]$ usually converges faster than $\lim_{i\to\infty}[\mathrm{tr}(X_{1,i}+X_{3,i})]^{1/i}$, which is the same. Moreover,

$$\tilde{\rho}\left(\overline{\Phi' \otimes \Phi'}\right) = \lim_{i\to\infty}\left[\frac{\mathrm{tr}\left(X_{1,i+1}+X_{3,i+1}\right)}{\mathrm{tr}\left(X_{1,i}+X_{3,i}\right)}\right] \tag{6.21}$$

where $\tilde{\rho}(\overline{\Phi' \otimes \Phi'})$ denotes the *minimal* spectral radius of the closed loop system (6.5), achievable with a compensator. Therefore (6.21), computed by Compensatability test 2, is actually a *measure* of compensatability. The main theorem below states a constructive solution of the optimal compensation problem. Call $X = \{X_1, X_2, X_3, X_4\}$ non-negative if $X_1, X_2, X_3, X_4 \geq 0$.

Theorem 3 (De Koning 1992)**:** *Assume the system* (2.1) *and* (2.2), *with the properties described at the start of this section, is ms-compensatable and* $Q > 0$, $V > 0$. *Then* $X^* = \lim_{i\to\infty} C^i(\theta, I, \theta, I)$ *exists*, X^* *is the unique non-negative solution of the equation* $X = CX$ *and* $F^* = F_{X^*}$, $K^* = K_{X^*}$, $L^* = L_{X^*}$. *Furthermore*,

$$\begin{aligned}\sigma_\infty^* &= \mathrm{tr}\left[QX_3^* + \left(Q + L_{X^*}^T R L_{X^*}\right) X_4^*\right] \\ &= \mathrm{tr}\left[VX_1^* + \left(V + K_{X^*} V K_{X^*}^T\right) X_2^*\right].\end{aligned} \tag{6.22}$$

Observe from De Koning (1992), *Theorem* 3, *that the condition* $Q > 0$, $V > 0$ *may be replaced with* $R > 0$, $W > 0$, $(\Phi_i, V^{1/2}, Q^{1/2})$ *ms-detectable*.

6.3 *U–D factorisation of the algorithms*

U–D factorisation of algorithms generally improves both their numerical accuracy and efficiency. It has mainly been employed in the past to improve algorithms for Kalman filtering (Bierman 1977; Thornton and Bierman 1980). A notable exception is the application to reduced-order compensator design for systems with deterministic parameters (Van Willigenburg and De Koning 2004).

Having stated the transformation C, that is fundamental to the algorithms, in the appropriate format (6.14), U–D factorisation may be employed. Let P represent non-negative symmetric matrices, U unit upper triangular matrices and D diagonal matrices. Then the following two equations represent the basic U–D factored computations needed to factorise the algorithm,

$$P_1 = U_{P_1} D_{P_1} U_{P_1}^T, \quad P_2 = U_{P_2} D_{P_2} U_{P_2}^T, \tag{6.23}$$

$$P_3 = F P_1 F^T + P_2 = U_{P_3} D_{P_3} U_{P_3}^T. \tag{6.24}$$

In Equations (6.23) and (6.24), all matrices are square and have the same dimension. Equation (6.23) represents U–D factorisations of P_1, P_2 whereas Equation (6.24) specifies P_3 as well as its associated U–D factorisation. Equation (6.24) is a discrete Lyapunov equation that describes a time update of a Kalman filter (Bierman 1977). Starting from P_1, P_2 one algorithm (A1) computes U_{P_1}, D_{P_1} and U_{P_2}, D_{P_2},

$$\mathrm{A1}\colon P \to U_P, D_P. \tag{6.25}$$

Another algorithm (A2) computes U_{P_3}, D_{P_3} in (6.24) from U_{P_1}, D_{P_1}, U_{P_2}, D_{P_2} and F,

$$\mathrm{A2}\colon U_{P_1}, D_{P_1}, U_{P_2}, D_{P_2}, F \to U_{P_3}, D_{P_3}. \tag{6.26}$$

Algorithms A1 and A2 are described in Bierman (1977, pp. 100–101, 131–133). The modification needed to apply them to non-negative instead of positive matrices is the same as the one described in Section 5.

To illustrate how Equations (6.23) and (6.24) and the associated algorithms A1 and A2 are employed consider,

$$\begin{aligned}\overline{\Phi X_3 \Phi^T} &= \overline{\Phi} X_3 \overline{\Phi}^T + \overline{\tilde{\Phi} X_3 \tilde{\Phi}^T} \\ &= \overline{\Phi} X_3 \overline{\Phi}^T + st^{-1}\left(\overline{\tilde{\Phi} \otimes \tilde{\Phi}}\; st(X_3)\right) \\ &= \overline{\Phi} X_3 \overline{\Phi}^T + \sum_{j=1}^{r_\Phi} \Phi_j X_3 \Phi_j^T, \quad X_3, \bar{\Phi}, \Phi_j \in R^{n\times n}.\end{aligned} \tag{6.27}$$

Observe that an U–D factorisation of the last two terms $\overline{\Phi} X_3 \overline{\Phi}^T + \sum_{j=1}^{r_\Phi} \Phi_j X_3 \Phi_j^T$ in (6.27) can be computed by recursively applying A2, specified

by (6.24) and (6.26), after and U–D factorisation of X_3 has been obtained from A1, specified by (6.23) and (6.25). Observe that $\overline{\Phi}X_3\overline{\Phi}^T + \sum_{j=1}^{r_\Phi}\Phi_j X_3 \Phi_j^T$ in (6.27) is obtained from the covariance representation $V^{\Phi\Phi} = \sum_{j=1}^{r_\Phi}\Phi_{i,j}\otimes\Phi_{i,j}$ in (4.3), once dropping the time index i. Recall that the representation (4.3) uses sums of matrices multiplied by scalar stochastic processes as specified by Equations (4.1) and (4.2). So it is this representation that allows for the U–D factorisation.

To further illustrate the U–D factorisation, consider again (6.17), which is among the most complicated terms of the transformation (6.14). Using Equation (6.18), after working out its final term, we obtain,

$$\begin{aligned}
&\overline{(\Phi-\Gamma L_X)^T X_1(\Phi-\Gamma L_X)}\\
&\quad=\big(\bar{\Phi}-\bar{\Gamma}L_X\big)^T X_1\big(\bar{\Phi}-\bar{\Gamma}L_X\big)\\
&\qquad+\overline{\tilde{\Phi}^T X_1\tilde{\Phi}}+L_X^T\overline{\tilde{\Gamma}^T X_1\tilde{\Gamma}}L_X\\
&\quad=\big(\bar{\Phi}-\bar{\Gamma}L_X\big)^T X_1\big(\bar{\Phi}-\bar{\Gamma}L_X\big)\\
&\qquad+st^{-1}\left(\overline{\tilde{\Phi}\otimes\tilde{\Phi}}^T st(X_1)\right)\\
&\qquad+L_X^T st^{-1}\left(\overline{\tilde{\Gamma}\otimes\tilde{\Gamma}}^T st(X_1)\right)L_X
\end{aligned}\tag{6.28}$$

Applying the alternative representations (4.3) and (4.4), i.e. $V^{\Phi\Phi}=\sum_{j=1}^{r_\Phi}\Phi_j\otimes\Phi_j$ and $V^{\Gamma\Gamma}=\sum_{j=1}^{r_\Gamma}\Gamma_j\otimes\Gamma_j$, instead of $V^{\Phi\Phi}=\overline{\tilde{\Phi}\otimes\tilde{\Phi}}$ and $V^{\Gamma\Gamma}=\overline{\tilde{\Gamma}\otimes\tilde{\Gamma}}$ we obtain,

$$\begin{aligned}
&\overline{(\Phi-\Gamma L_X)^T X_1(\Phi-\Gamma L_X)}\\
&\quad=\big(\bar{\Phi}-\bar{\Gamma}L_X\big)^T X_1\big(\bar{\Phi}-\bar{\Gamma}L_X\big)\\
&\qquad+\sum_{j=1}^{N}\Phi_j^T X_1\Phi_j+L_X^T\left(\sum_{j=1}^{N}\Gamma_j^T X_1\Gamma_j\right)L_X.
\end{aligned}\tag{6.29}$$

Like Equation (6.27), observe that an U–D factorisation of the right hand side of Equation (6.29) can be computed by recursively applying algorithm A2, once and U–D factorisation of X_1 is obtained from algorithm A1. The outer multiplication by L_X^T and L_X of the last term in (6.29) is realised by algorithm A2, setting $P_2=\theta$ in (6.24). In a similar manner, all other terms of the transformation (6.14) can be computed. Adding these terms can also be performed by algorithm A2, taking Φ in (6.24) to be the identity matrix. Preferably, when Φ is the identity matrix, a simplified version of algorithm A2 can be employed. Finally, K_X, L_X specified by (6.10) and (6.11) are computed as follows. The part which must be inverted is computed similar to (6.27). The result is therefore U–D factored. The standard inverse of the U–D factored matrix P in (6.25) satisfies,

$$P^{-1}=U_P^{-T}D_P^{-1}U_P^{-1}\tag{6.30}$$

U_P^{-1} is computed using the efficient algorithm Bierman (1977, p. 65) that exploits the triangular nature of U_P. We further simplified this algorithm exploiting the fact that U_P is *unit* upper triangular. If P is singular, several diagonal elements of D_P are zero. The associated columns of U_P are irrelevant and set to zero. Then the algorithm computes a generalised inverse with the properties (6.13) by setting to zero the same diagonal elements of D_p^{-1} and the associated columns of U_p^{-1} in (6.30). Although this generalised inverse is not necessarily equal to the Moore–Penrose pseudo-inverse used by De Koning (1982, 1992), via completion of the square one may establish that satisfying (6.13) is sufficient to find a minimum, as required by the two compensatability tests. This fact is demonstrated by Example 1 in the next section. As to Theorem 3, note that $R>0$ and $W>0$ imply that the generalised inverse in Equations (6.10) and (6.11) turns into the standard inverse. Having computed U–D factorisations of the inverted part of K_X, L_X in this manner, the matrices K_X, L_X themselves are obtained from ordinary matrix computations.

Two more computational issues require attention. Recall from Section 5, Equations (5.11) and (5.12), that $\Phi_j\in R^{n\times n}$, $j=1,2,\ldots,r_\Phi$ in Equation (4.3) are obtained by unstacking the r_Φ non-zero columns of an upper triangular matrix $U\in R^{n^2\times n^2}$. Knowing this matrix is upper triangular, we establish the number of terminal elements of each column being zero. After unstacking, these zero terminal elements make up a known number of zero terminal columns of Φ_j. In the algorithms, these zero terminal columns may be exploited by properly ignoring them, saving computational effort. Our implementation of the algorithms does so. To illustrate how, consider the term,

$$\sum_{j=1}^{r_\Phi}\Phi_j^T X_1\Phi_j,\tag{6.31}$$

in Equation (6.29). Suppose only the first $k_j<n$ columns of Φ_j are non-zero. Then,

$$\Phi_j^T X_1\Phi_j=\begin{bmatrix}\Phi_j^T(:,1{:}k_j)X_1\Phi_j(:,1{:}k_j) & 0\\ 0 & 0\end{bmatrix},\tag{6.32}$$

where $\Phi_j(:,1{:}k_j)$ denotes all rows and the first k_j columns of Φ_j. From (6.32) observe that we only have to compute the non-zero upper left block. An U–D factorisation of this upper left block can also be obtained from algorithm A2, if instead of $\Phi=\Phi_j$ we select $\Phi=\Phi_j(:,1:k_n)$ as an input in (6.24).

As mentioned in Van Willigenburg and De Koning (2004), algorithm A2 is easily adapted to enable this type of non-square input. Dual to terms like (6.31), terms like the following, taken from (6.27), occur,

$$\sum_{j=1}^{r_\Phi} \Phi_j X_3 \Phi_j^T \tag{6.33}$$

Using only the non-zero columns of Φ_j in (6.33) results in the following computations,

$$\Phi_j X_3 \Phi_j^T = \Phi_j(:, 1:k_j) X_3(1:k_j, 1:k_j) \Phi_j^T(:, 1:k_j), \tag{6.34}$$

where $X_3(1:k_j, 1:k_j)$ denotes the first k_j rows and columns of X_3. Equation (6.34) is not directly suitable for U–D factorisation because a row and column reduced version of X_3 appears in it. One would therefore prefer a computation like (6.32). This computation is obtained from a factorisation that is dual to (4.3),

$$V^{\Phi\Phi} = \sum_{j=1}^{r_\Phi} \Phi_j' T \otimes \Phi_j'^T. \tag{6.35}$$

The dual factorisation is easily obtained from the original as can be seen from Equation (6.15). Instead of the minimal representation (4.3), Φ_j' in (6.35) are obtained from a minimal representation of $V'^{\Phi\Phi} = V^{\Phi\Phi^T} = \overline{\tilde{\Phi} \otimes \tilde{\Phi}}^T$. To compute this minimal representation along the lines of Theorem 2, we have to convert $V'^{\Phi\Phi}$ to $V'^{\phi\phi}$ using the one to one mapping described in Theorem 1. This conversion is just a reallocation of matrix elements. Our implementation of the algorithms uses the dual factorisation (6.35). This dual factorisation has to be computed only ones, at the start of each algorithm. It enables the computation (6.32) that ignores zero columns to improve the computational efficiency. Similar arguments apply to the factorisations (4.4) and (4.5).

7. Numerical considerations and examples

To check the algebraic equivalence of the U–D factored algorithms with the conventional ones based on Kronecker products, both are used to solve examples that appear in De Koning (1992, 1993). These examples illustrate some fundamental properties related to white parameter uncertainty and compensatability as well as optimal compensation. The examples are also suitable for presentation, because the system has small dimensions. To investigate the performance of the U–D factored algorithms, computation times obtained for examples with system dimensions up to 70 are presented, and compared to those obtained with the conventional algorithms based on Kronecker products. Observe that a system dimension of $n = 70$ results in Kronecker products associated with the system matrix Φ of dimension $n^2 = 4900$. Finally in this section, we will comment on the possible improvement of accuracy of the U–D factored algorithm over the conventional ones. Our algorithms are programmed and executed in MATLAB®. In an attempt to achieve comparable performance with standard operations, the U–D factorisations were implemented using MEX files the source of which was written in C language.

Example 1: (De Koning 1992, 1993)

$$\bar{\Phi} = \begin{bmatrix} 0.7092 & 0.3017 \\ 0.1814 & 0.9525 \end{bmatrix}, \quad \bar{\Gamma} = \begin{bmatrix} 0.7001 \\ 0.1593 \end{bmatrix}, \quad \bar{C} = \begin{bmatrix} 0.3088 & 0.5735 \end{bmatrix} \tag{7.1}$$

$$\begin{aligned} \overline{\tilde{\Phi} \otimes \tilde{\Phi}} &= V^{\Phi\Phi} = \beta_1 \bar{\Phi} \otimes \bar{\Phi}, \\ \overline{\tilde{\Gamma} \otimes \tilde{\Gamma}} &= V^{\Gamma\Gamma} = \beta_2 \bar{\Gamma} \otimes \bar{\Gamma}, \\ \overline{\tilde{C} \otimes \tilde{C}} &= V^{CC} = \beta_3 \bar{C} \otimes \bar{C}, \quad \beta_1, \beta_2, \beta_3 \geq 0 \end{aligned} \tag{7.2}$$

$$\begin{aligned} V &= diag(0.5627 \quad 0.7357), \quad W = 0.2588, \\ Q &= diag(0.7350 \quad 0.9820), \quad R = 0.6644 \end{aligned} \tag{7.3}$$

Observe that Equation (7.2), representing covariances of the system matrices, is of the form (4.3)–(4.5) without the time-index i and with $r_\Phi = 1$, $\Phi_1 = \sqrt{\beta_1}\bar{\Phi}$, $r_\Gamma = 1$, $\Gamma_1 = \sqrt{\beta_2}\bar{\Gamma}$, $r_C = 1$, $C_1 = \sqrt{\beta_3}\bar{C}$. Because $r_\Phi = 1$, the elements of Φ_i are fully correlated. The same applies to Γ_i and C_i. Also, according to Theorem 2, (7.2) is a minimal representation. From De Koning (1993) observe that β_1, β_2, $\beta_3 \geq 0$ in (7.2) are measures of uncertainty of the system matrices Φ_i, Γ_i, C_i, respectively. As these measures increase the minimal spectral radius $\tilde{\rho}(\overline{\Phi' \otimes \Phi'})$ of the closed loop system, achievable with a compensator, increases (De Koning 1993). This is confirmed by Table 1. The same applies to the minimum value σ_∞^* of the cost function (6.7). If $\tilde{\rho}(\overline{\Phi' \otimes \Phi'}) \geq 1$, the system is not ms-compensatable and $\sigma_\infty^* = \infty$. The values of $\tilde{\rho}(\overline{\Phi' \otimes \Phi'})$ and σ_∞^* were computed with both the conventional and U–D factored algorithm. Both gave the same results within the specified convergence tolerance of 10^{-6}, indicating algebraic equivalence. The required number of algorithm iterations to obtain convergence was almost identical as well.

Next we compare the efficiency of the U–D factored algorithm with the conventional one. Because the algorithms perform mainly second moment computations, these are compared first in Table 2.

Table 1. Minimal spectral radius and minimal costs against parameter uncertainty for Example 1.

β_1	β_2	β_3	$\tilde{\rho}(\overline{\Phi' \otimes \Phi'})$	σ^*_∞
0.1	0.1	0.1	6.6542E − 01	9.7487E + 00
0.2	0.2	0.2	9.4573E − 01	5.3826E + 01
0.3	0.3	0.3	1.1658E + 00	∞
0	0.1	0.1	4.5278E − 01	5.7552E + 00
0.2	0.1	0.1	8.4555E − 01	2.3197E + 01
0.4	0.1	0.1	1.1685E + 00	∞
0.6	0.1	0.1	1.4662E + 00	∞
0.8	0.1	0.1	1.7499E + 00	∞
0.1	0	0.1	5.7289E − 01	8.9161E + 00
0.1	0.2	0.1	7.2631E − 01	1.0714E + 01
0.1	0.4	0.1	8.0961E − 01	1.3206E + 01
0.1	0.6	0.1	8.6730E − 01	1.6907E + 01
0.1	0.8	0.1	9.1098E − 01	2.3004E + 01
0.1	0.1	0	5.7289E − 01	8.7427E + 00
0.1	0.1	0.2	7.2631E − 01	1.0890E + 01
0.1	0.1	0.4	8.0961E − 01	1.3781E + 01
0.1	0.1	0.6	8.6730E − 01	1.8028E + 01
0.1	0.1	0.8	9.1098E − 01	2.4998E + 01

Execution times are recorded and for some cases also floating-point operation counts (flops). The conventional algorithm performs second moment computations using Kronecker products as represented by Equation (6.15). Each second moment computation (6.15) requires n^4 scalar multiply accumulate operations each one counted as a single flop. Instead, the U–D factored algorithm computes second moments as represented by the terms after the last equality in Equation (6.27). Straightforward computation of (6.27), indicated by MAC (multiply accumulate) in Table 2, requires $2(r_\Phi + 1)n^3$ scalar multiply accumulate operations (flops). Although U–D factorisation exploits the symmetry in Equation (6.27), not exploited by MAC, algorithm A2 described by (6.24) and (6.26) still requires in between $2.5(r_\Phi + 1)n^3$ (for $n \to \infty$), and $4(r_\Phi + 1)n^3$ (for $n = 1$) flops. So U–D factorisation of (6.27) slightly decreases computational efficiency. Both MAC and U–D outperform the conventional algorithm denoted by Kronecker product for small r_Φ ($r_\Phi = 1, 5$) and high n ($n = 50, 70$). According to Theorem 2, $r_\Phi \leq n^2$. When n is sufficiently large, in specifying common stochastic parameter statistics, one generally has $r_\Phi \ll n$. For instance $r_\Phi = 1$ when using the specification of stochastic parameters statistics (7.2) in Example 1, which applies to systems of arbitrary dimension n.

Table 2 also specifies the computation times obtained by properly ignoring *a priori* known zeros, as described by and before Equations (6.31) and (6.32). The higher r_Φ the larger the number of *a priori* known zeros as can be seen from Equation (5.11). But Equation (6.32) requires indexing parts of a matrix which appears to be rather inefficient as compared to performing computations. Only when $n = 70$, $r_\Phi = n^2$, the MAC execution time is reduced. For U–D execution times, which are larger, this happens more often. Finally observe that, as expected, the Kronecker product execution time is the only one almost independent of the value r_Φ.

Finally, Table 3 records execution times of the main parts of the conventional and U–D factored algorithm. As expected from Table 2 and the associated discussion, only if r_Φ is significantly smaller than n, the U–D algorithm may outperform the conventional one. The outperformance is hampered by the need for Cholesky decompositions (5.11) to determine Φ_j, Γ_j and C_j from the variances $V^{\Phi\Phi}$, $V^{\Gamma\Gamma}$ and V^{CC} such that $V^{\Phi\Phi} = \sum_{j=1}^{r_\Phi} \Phi_j \otimes \Phi_j$, $V^{\Gamma\Gamma} = \sum_{j=1}^{r_\Gamma} \Gamma_j \otimes \Gamma_j$ and $V^{CC} = \sum_{j=1}^{r_\Gamma} C_j \otimes C_j$ are minimal representations, according to Theorem 2. Also the dual factorisations (6.35) may be required. The execution time of these Cholesky factorisations, including the dual ones, is also mentioned in Table 3. Although they have to be executed only once, their execution time is significant and increases significantly with n. These Cholesky decompositions are avoided if the covariances of the system matrices are formulated directly in terms of Φ_j, Γ_j and C_j, like in Example 1. Note that r_Φ, r_Γ, r_C need not necessarily be minimal. The examples in Table 3 concern randomly generated problems with $m = 3$, $l = 4$, $r_\Gamma = \min(nm, r_\Phi)$, $r_C = \min(nl, r_\Phi)$ that are all ms-compensatable. Generation of proper random covariances of the system matrices is greatly facilitated by Theorems 1 and 2. The algorithms in each case were verified to produce the same solution in approximately the same number of iterations. The total execution time of the conventional algorithm is approximately equal to the total iteration time. The same applies to the U–D factored algorithm if Cholesky decompositions are not required. If required their execution time must be added. Adding these in Table 3, only for the case $n = 70$, $r_\Phi = 1$ the U–D algorithm outperforms the conventional one.

In terms of computational efficiency in MATLAB®, the U–D algorithm has advantages in cases of large n and small r_Φ, r_Γ and r_C only. U–D factorisation however has other advantages such as guaranteeing non-negativeness of the matrices during iterations of the algorithm that enhances numerical stability. Another potential advantage is doubling of precision (Bierman 1977). Doubling of precision is achieved only if recovery of the U–D factored matrices into ordinary representation is avoided during the iterations. This type of recovery occurs in Equations (6.10) and (6.11) in the part that is not inverted. Further investigation into U–D factorisation is needed to see if this recovery can be avoided. As to the

Table 2. Execution time (s) and flops of different second moment computations.

n	r_Φ	Kronecker product (6.15)(s)	flops	MAC (6.27) (s)	MAC exploiting zeros (s)	U–D (6.24) and (6.26)(s)	flops	U–D exploiting zeros (s)
20	1	2.45E−05	16E4	1.99E−05	2.26E−05	4.81E−05	4.158E4	3.97E−05
50	1	3.01E−03	625E4	9.26E−05	9.01E−05	3.18E−04	63.495E4	3.44E−04
70	1	1.31E−02	2401E4	9.60E−05	1.26E−04	1.02E−03	173.453E4	8.05E−04
20	5	2.38E−05	16E4	6.27E−05	8.50E−05	1.64E−04	12.474E4	1.55E−04
50	5	4.70E−03	625E4	3.09E−04	4.34E−04	1.47E−03	190.485E4	1.46E−03
70	5	1.34E−02	2401E4	5.58E−04	7.10E−04	4.02E−03	520.359E4	4.19E−03
20	20	3.28E−05	16E4	2.32E−04	3.27E−04	5.83E−04	43.659E4	5.54E−04
50	50	2.86E−03	625E4	1.94E−03	6.28E−03	1.43E−02	1619.1225E4	2.09E−02
70	70	1.12E−02	2401E4	5.61E−03	9.55E−03	5.56E−02	6157.5815E4	5.36E−02
20	400	2.31E−05	16E4	5.68E−03	7.68E−03	1.24E−02	833.679E4	7.40E−03
50	2500	2.85E−03	625E4	1.34E−01	1.45E−01	6.98E−01	7.94E8	2.82E−01
70	4900	1.10E−02	2401E4	5.23E−01	4.15E−01	3.69E+00	4.25E9	1.39E+00

Table 3. Execution time (s) of the main parts of the conventional and U–D factored algorithm.

n	r_Φ	Single iteration time	Total iteration time	U–D single iteration time	U–D total iteration time	U–D time of Cholesky decompositions
20	1	1.05E−03	4.41E−02	7.21E−04	3.03E−02	9.65E−03
50	1	8.02E−03	5.05E−01	5.55E−03	3.39E−01	3.42E−01
70	1	2.76E−02	4.97E+00	1.47E−02	2.61E+00	1.22E+00
20	5	4.01E−04	1.73E−02	1.32E−03	5.66E−02	1.46E−02
50	5	7.77E−03	4.74E−01	9.89E−03	6.03E−01	3.75E−01
70	5	2.71E−02	7.95E+00	2.52E−02	7.37E+00	1.37E+00
20	20	3.84E−04	1.65E−02	3.24E−03	1.39E−01	1.09E−02
50	50	8.06E−03	4.92E−01	5.43E−02	3.31E+00	7.34E−01
70	70	2.71E−02	9.80E+00	1.69E−01	6.11E+01	3.27E+00

conventional algorithm programmed in MATLAB®, we conclude that it appears to be very efficient. Moreover, this efficiency is almost independent of r_Φ, r_Γ and r_C.

8. Conclusions

The minimal representation of matrix valued white stochastic processes, by means of sums of matrices multiplied by independent scalar stochastic processes, was developed. An algorithm to compute minimal representations was described. Central to this algorithm is a generalised Cholesky decomposition producing a matrix square-root of a non-negative square matrix. This non-negative square matrix is the intensity or covariance matrix of the vector valued white stochastic process obtained by stacking the columns of the matrix valued white stochastic process. Because a matrix square-root is not unique, neither is the associated minimal representation.

Another representation of matrix valued white stochastic processes uses Kronecker products and is more concise. This article confirmed the latter by revealing that the Kronecker product representations do not require minimal representation since, according to Theorem 1, they map one to one to the intensity or covariance matrix of the vector valued white stochastic process obtained after stacking. The one to one mapping also determines precisely which Kronecker products are valid representations. Kronecker product representations are used by algorithms to compute optimal controllers for linear dynamical systems having white stochastic parameters. As demonstrated in this article, U–D factorisation of these algorithms, on the other hand, rely on the representation by sums of matrices multiplied by independent scalar stochastic processes, and their minimal representation described by Theorem 2. We described how these enable U–D factorisation of the algorithm to compute optimal dynamic output feedback controllers (compensators). These compensators exist only if the system is mean-square compensatable (ms-compensatable). Two related algorithms that determine ms-compensatability of a system were also U–D factored. Finally, the performance of the U–D factored and conventional

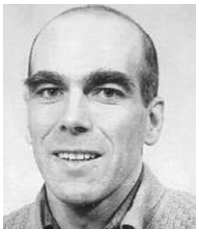
algorithms was compared by means of solving several numerical examples using MATLAB®. The generation of these examples is greatly facilitated by the results of this article.

The algorithm comparison reveals that U–D factorisation becomes more efficient as the system dimension n increases and as the ranks r_Φ, r_Γ, r_C of matrices describing system parameter uncertainty become less. Only for high ratios n/r_Φ, n/r_Γ, n/r_C, the U–D factored algorithm is more efficient. On the other hand, U–D factorisation ensures non-negativeness of matrices during iteration of the algorithm, which enhances numerical stability (Bierman 1977).

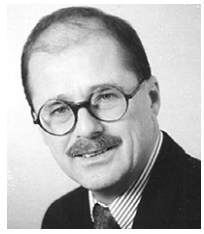
The conventional algorithms using Kronecker product representations turn out very efficient, when executed in MATLAB®. This is partly due to the high efficiency of large matrix vector multiplications as compared to other programming operations. Moreover, the conventional algorithm efficiency is not affected by the rank of the aforementioned matrices. The efficiency of the U–D factored algorithm decreases significantly when this rank increases.

To benefit from the possible doubling of precision, obtained by U–D factorisation, the expressions of the compensator gains need further investigation. Presently, their computation is not fully U–D factored. Another topic for future research concerns the U–D factorisation of optimal *reduced-order* compensator algorithms for system with white stochastic parameters. Two types of algorithms, one based on the strengthened optimal projection equations (De Koning and Van Willigenburg 1998) and another based on Lyapunov equations (Van Willigenburg and De Koning 2004) are candidates.

Notes on contributors



L. Gerard Van Willigenburg was born in Leiden, The Netherlands, in 1958. He received his MSc in electrical engineering and his PhD degree in control engineering at Delft University of Technology, The Netherlands, in 1983 and 1991, respectively. From 1986 to 1991, he was a Research Engineer at the process dynamics and control group in the Department of Applied Physics. Since 1991, he is an Assistant Professor of the systems and control group at Wageningen University. His professional research interests include digital optimal control, reduced-order control, model predictive control and adaptive dual control. The application areas range from indoor climate control (storage rooms, greenhouses, stables and buildings), robot control, automatic guidance, to process control (fermentation) and the control of economic and biological systems (plants). The modelling of fundamental physics (thermodynamics, quantum mechanics) has become a private research interest.



Willem L. De Koning was born in Leiden, The Netherlands, in 1944. He received his MSc in electrical engineering and his PhD degree in control engineering at Delft University of Technology, The Netherlands, in 1975 and 1984 respectively. From 1969 to 1975, he was a Research Engineer of power electronics and control at the Department of Electrical Engineering. From 1987 to 2006, he was an Associate Professor of mathematical system theory at the Department of Mathematical System Theory in the Department of Technical Mathematics and informatics. He has also held a visiting position at the Florida Institute of Technology, Melbourne. His research interests include control of distributed parameter systems, robust control, adaptive control, reduced-order control and digital control. The main application areas are the process industry and mechatronics. Although he retired in 2006, he continues some of his research on a personal basis.

References

Antunes, D., Hespanha, J.P., and Silvestre, C. (2009), 'Control of Impulsive Renewal Systems: Application to Direct Design in Networked Control', in *Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, Shanghai, P.R. China, December 16–18, pp. 6882–6887.

Arnold, L. (1974), *Stochastic Differential Equations*, New York, NY: Wiley.

Athans, M. (1971), 'The Role and use of the Stochastic Linear-Quadratic-Gaussian Problem in Control System Design', *IEEE Transactions on Automatic Control*, AC-16, 529–552.

Bernstein, D.S., and Haddad, W.M. (1987), 'Optimal Projection Equations for Discrete-time Fixed-order Dynamic Compensation of Linear Systems with Multiplicative White Noise', *International Journal of Control*, 46, 65–73.

Bierman, G.J. (1977), *Factorization Methods for Discrete Sequential Estimation*, New York, NY: Academic Press.

Boje, E. (2005), 'Approximate Models for Continuous-time Linear Systems with Sampling Jitter', *Automatica*, 41, 2091–2098.

De Koning, W.L. (1982), 'Infinite Horizon Optimal Control of Linear Discrete-time Systems with Stochastic Parameters', *Automatica*, 18, 443–453.

De Koning, W.L. (1992), 'Compensatability and Optimal Compensation of Systems with White Parameters', *IEEE Transactions on Automatic Control*, 37, 579–588.

De Koning, W.L. (1993), 'The Influence of Uncertainty on Stability and Compensatability', *International Journal of Control*, 58, 697–705.

De Koning, W.L., and Van Willigenburg, L.G. (1998), 'Numerical Algorithms and Issues Concerning the Discrete-time Optimal Projection Equations for Systems with White Parameters', in *Proceedings UKACC*

International Conference on Control '98, 1–4 September, University of Swansea, UK, Vol. 2, pp. 1605–1610.

De Koning, W.L., and Van Willigenburg, L.G. (2001), *Randomized Digital Optimal Control, Chapter 12 in Non Uniform Sampling Theory and Practice*, New York, NY: Kluwer Acadamic/Plenum Publishers.

Fujimoto, K., Ota, Y., and Nakayama, M. (2011), 'Optimal Control of Linear Systems with Stochastic Parameters for Variance Suppression', in *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, December 12–15, Orlando, FL, USA.

Hounkpevi, F.O., and Yaz, E.E. (2008), 'Current Output Observer for Stochastic-parameter Models and Application to Sensor Failure', in *Proceedings of the 17th World Congress, The International Federation of Automatic Control*, July 6–11, Seoul, Korea.

Immer, O.C., Yükselb, S., and Basar, T. (2006), 'Optimal Control of LTI Systems over Unreliable Communication Links', *Automatica*, 24, 1429–1439.

Kögel, M., Blind, R., Allgöwer, F., Findeisen R. (2011), 'Optimal and Optimal-Linear Control over Lossy, Distributed Networks', in *Preprints of the 18th IFAC World Congress*, August 28–September 2, Milano, Italy.

Li, F., Zhoua, J., and Wub, D. (2011), 'Optimal Filtering for Systems with Finite-step Autocorrelated Noises and Multiple Packet Dropouts', *Aerospace Science and Technology*, doi10.1016/j.ast.2011.11.013.

Meenakshi, M., and Bhat, M.S. (2006), 'Robust Fixed-order H2 Controller for Micro Air Vehicle Design and Validation', *Optimal Control Applications and Methods*, 27, 183–210.

Thornton, C.L., and Bierman, G.J. (1980), *U-D Covariance Factorization for Kalman Filtering, Control and dynamic systems (A81-15351 04-63)*, New York, NY: Academic Press Inc., pp. 177–248.

Van Willigenburg, L.G., and De Koning, W.L. (2004), 'UDU Factored Discrete-time Lyapunov Recursions Solve Optimal Reduced-order LQG Problems', *European Journal of Control*, 10, 588–601.