Constructing Sparse Realizations of Finite-Precision Digital Controllers Based on a Closed-Loop Stability Related Measure

Jun Wu†, Sheng Chen‡, Gang Li§ and Jian Chu†

† National Laboratory of Industrial Control Technology
Institute of Industrial Process Control
Zhejiang University, Hangzhou, 310027, P. R. China

‡ Department of Electronics and Computer Science
University of Southampton, Highfield
Southampton SO17 1BJ, U.K.

§ School of Electrical and Electronic Engineering
Nanyang Technological University, Singapore

Abstract

We present a study of the finite word length (FWL) implementation for digital controller structures with sparseness consideration. A new closed-loop stability related measure is derived, taking into account the number of trivial elements in a controller realization. A practical design procedure is presented, which first obtains a controller realization that maximizes a lower bound of the proposed measure, and then uses a stepwise algorithm to make the realization sparse. Simulation results show that the proposed design procedure yields computationally efficient controller realizations with enhanced FWL closed-loop stability performance.

Index Terms — digital controller, finite word length, closed-loop stability, sparse realization, optimization, stepwise algorithm, real-time computation.

1 Introduction

It is well-known that a designed stable control system may achieve a lower than predicted performance or even become unstable when the control law is implemented with a finite-precision device due to FWL effects. In real-time applications where computational efficiency is critical, a digital controller implemented in fixed-point arithmetic has certain advantages. With a fixed-point processor, the detrimental FWL effects are markedly increased due to a reduced precision. As the FWL effects on the closed-loop stability depend on the controller realization structure,

[Contact author. Tel./Fax: +44 (0)23 8059 6660/4508; Email: sqc@ecs.soton.ac.uk]
many studies have addressed the problem of finding “optimal” realizations of finite-precision controller structures based on various FWL stability measures [1]-[7]. Except [5], these design methods usually yield fully parameterized controller structures, that is, they generally do not produce sparse controller realizations.

It is highly desirable that a controller realization has a sparse structure, containing many trivial elements of 0, 1 or -1. This is particularly important for real-time applications with high-order controllers, as it will achieve better computational efficiency. It is known that canonical controller realizations have sparse structures but may not have the required FWL stability robustness. This poses a complex problem of finding sparse controller realizations with good FWL closed-loop stability characteristics. In [8], sparseness consideration is imposed as constraints in optimizing a FWL stability measure using an adaptive simulated annealing (ASA) algorithm. This approach is difficult to extend to high-order controllers due to high computational requirements. In our previous works [9],[10], a design procedure has been given to obtain sparse controller realizations based on a FWL pole-sensitivity stability related measure.

In this study we derive a new improved FWL closed-loop stability related measure, which takes into account the number of trivial elements in a controller realization. The true optimal realization that maximizes this measure will possess an optimal trade-off between robustness to FWL errors and sparse structure. However, it is not known how to obtain such an optimal realization. We extend an iterative algorithm [2],[11] to search for a suboptimal solution. Specifically, we first obtain the realization that maximizes a lower bound of the proposed stability measure. This can easily be done [5],[7] but the resulting realization is not sparse. A stepwise algorithm is then applied to make the realization sparse without sacrificing FWL stability robustness too much. The proposed method has some advantages over the existing methods [5],[9],[10]: it is less conservative in estimating the robustness of the FWL closed-loop stability and the computational complexity is considerably reduced. Numerical examples are used to test this design procedure and to compare its performance with the previous method [9],[10].

2 A stability related measure with sparseness considerations

Consider the discrete-time closed-loop control system, consisting of a linear time-invariant plant \( P(z) \) and a digital controller \( C(z) \). The plant model \( P(z) \) is assumed to be strictly proper with a state-space description \((A_P, B_P, C_P)\), where \( A_P \in \mathbb{R}^{m \times m}\), \( B_P \in \mathbb{R}^{m \times l}\) and \( C_P \in \mathbb{R}^{q \times m}\).
Let \((A_C, B_C, C_C, D_C)\) be a state-space description of the controller \(C(z)\), with \(A_C \in \mathbb{R}^{n \times n}\), \(B_C \in \mathbb{R}^{n \times q}\), \(C_C \in \mathbb{R}^{l \times n}\) and \(D_C \in \mathbb{R}^{l \times q}\). A linear system with a given transfer function matrix has an infinite number of state-space descriptions. In fact, if \((A_C^0, B_C^0, C_C^0, D_C^0)\) is a state-space description of \(C(z)\), all the state-space descriptions of \(C(z)\) form a realization set

\[
S_C = \left\{ (A_C, B_C, C_C, D_C) \middle| A_C = T^{-1} A_C^0 T, B_C = T^{-1} B_C^0, C_C = C_C^0 T, D_C = D_C^0 \right\}
\]

where \(T \in \mathbb{R}^{n \times n}\) is any non-singular matrix. Denote \(N = (l + n)(q + n)\) and

\[
X = \begin{bmatrix} D_C & C_C \\ B_C & A_C \end{bmatrix} = \begin{bmatrix} x_1 & x_{l+n+1} & \cdots & x_{N-l-n+1} \\ x_2 & x_{l+n+2} & \cdots & x_{N-l-n+2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{l+n} & x_{2l+2n} & \cdots & x_N \end{bmatrix}
\]

The stability of the closed-loop control system depends on the eigenvalues of the closed-loop system matrix

\[
\overline{A}(X) = \begin{bmatrix} A_P + B_P D_C C_P & B_P C_C \\ B_C C_P & A_C \end{bmatrix} = \begin{bmatrix} A_P & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} B_P & 0 \\ 0 & I_n \end{bmatrix} X \begin{bmatrix} C_P & 0 \\ 0 & I_n \end{bmatrix} \overset{\triangle}{= M_0 + M_1 XM_2}
\]

where \(0\) denotes the zero matrix of appropriate dimension and \(I_n\) the \(n \times n\) identity matrix.

All the different realizations \(X\) in \(S_C\) have exactly the same set of closed-loop poles if they are implemented with infinite precision. Since the closed-loop system has been designed to be stable, all the eigenvalues \(\lambda_i(\overline{A}(X))\), \(1 \leq i \leq m + n\), are within the unit disk.

When a \(X\) is implemented with a fixed-point processor, it is perturbed to \(X + \Delta X\) due to the FWL effect. Each element of \(\Delta X\) is bounded by \(\pm \varepsilon/2\), that is,

\[
\mu(\Delta X) \overset{\triangle}{=} \max_{j \in \{1, \ldots, N\}} |\Delta x_j| \leq \varepsilon/2
\]

For a fixed-point processor of \(B_i\) bits, let \(B_i = B_i + B_f\), where \(2^{B_i}\) is a “normalization” factor to make the absolute value of each element of \(2^{-B_i} X\) no larger than 1. Thus, \(B_i\) are bits required for the integer part of a number and \(B_f\) are bits used to implement the fractional part of a number. It can easily be seen that \(\varepsilon = 2^{-B_f}\). With the perturbation \(\Delta X\), \(\lambda_i(\overline{A}(X))\) is moved to \(\lambda_i(\overline{A}(X + \Delta X))\). If an eigenvalue of \(\overline{A}(X + \Delta X)\) is outside the open unit disk, the closed-loop system, designed to be stable, becomes unstable with \(B_i\)-bit implemented \(X\). It is therefore critical to choose a realization \(X\) that has a good closed-loop stability robustness to the FWL error. Another important consideration is the sparseness of \(X\). Those elements of \(X\), which have values 0, 1 and -1, are called trivial parameters. A trivial parameter requires no operations
in the fixed-point implementation and does not cause any computational error at all. Thus 
\( \Delta x_j = 0 \) when \( x_j = 0, 1 \) or \(-1\). In order to take into account this property of trivial controller 
parameters, we define an indicator function as 
\[
\delta(x) = \begin{cases} 
0, & \text{if } x = 0, 1 \text{ or } -1 \\
1, & \text{otherwise} 
\end{cases}
\] 

(5)

We are now ready to propose a new FWL closed-loop stability related measure which takes 
into account the sparseness of a controller realization. When the FWL error \( \Delta \mathbf{X} \) is small,
\[
\Delta |\lambda_i| \equiv |\lambda_i(\mathbf{A}(\mathbf{X} + \Delta \mathbf{X}))| - |\lambda_i(\mathbf{A}(\mathbf{X}))| \approx \sum_{j=1}^{N} \frac{\partial |\lambda_i|}{\partial x_j} \Delta x_j \delta(x_j), \quad \forall i \in \{1, \ldots, m + n\}
\] 

(6)

where \( \frac{\partial |\lambda_i|}{\partial x_j} \) is evaluated at \( \mathbf{X} \). It follows from the Cauchy inequality that
\[
|\Delta |\lambda_i|| \leq \sqrt{N_i \sum_{j=1}^{N} \left| \frac{\partial |\lambda_i|}{\partial x_j} \right|^2 \delta(x_j)} \leq \mu(\Delta \mathbf{X}) \sqrt{N_i \sum_{j=1}^{N} \left| \frac{\partial |\lambda_i|}{\partial x_j} \right|^2 \delta(x_j)}, \quad \forall i
\] 

(7)

where \( N_i \) is the number of the nontrivial elements in \( \mathbf{X} \). This leads to the following FWL 
closed-loop stability related measure
\[
\mu_1(\mathbf{X}) = \min_{i \in \{1, \ldots, m+n\}} \frac{1 - |\lambda_i(\mathbf{A}(\mathbf{X}))|}{\sqrt{N_i \sum_{j=1}^{N} \delta(x_j) \left| \frac{\partial |\lambda_i|}{\partial x_j} \right|^2}}
\] 

(8)

The rationale of this measure is obvious. If the norm of the FWL error \( \Delta \mathbf{X} \) is smaller than 
\( \mu_1(\mathbf{X}) \), i.e. \( \mu(\Delta \mathbf{X}) < \mu_1(\mathbf{X}) \), it follows from (7) and (8) that 
\( |\Delta |\lambda_i|| < 1 - |\lambda_i(\mathbf{A}(\mathbf{X}))| \). Therefore
\[
|\lambda_i(\mathbf{A}(\mathbf{X} + \Delta \mathbf{X}))| \leq |\Delta |\lambda_i|| + |\lambda_i(\mathbf{A}(\mathbf{X}))| < 1
\] 

(9)

which means that the closed-loop system remains stable under the FWL error \( \Delta \mathbf{X} \). In other 
words, for a given controller realization \( \mathbf{X} \), the closed-loop system can tolerate those FWL 
perturbations \( \Delta \mathbf{X} \) whose norms, as defined in (4), are less than \( \mu_1(\mathbf{X}) \). The larger \( \mu_1(\mathbf{X}) \) is, 
the larger FWL errors the closed-loop system can tolerate. Hence, \( \mu_1(\mathbf{X}) \) is a stability related 
measure describing the FWL closed-loop stability performance of a controller realization \( \mathbf{X} \). 
This measure clearly considers the number of trivial parameters in a controller realization. We 
can now discuss how to compute \( \mu_1(\mathbf{X}) \). First we have the following lemma from [5],[7].

**Lemma 1** Let \( \mathbf{A}(\mathbf{X}) = \mathbf{M}_0 + \mathbf{M}_1 \mathbf{X} \mathbf{M}_2 \) given in (3) be diagonalisable, and have eigenvalues 
\( \{\lambda_i\} = \{\lambda_i(\mathbf{A}(\mathbf{X}))\} \). Denote \( \mathbf{p}_i \) a right eigenvector of \( \mathbf{A}(\mathbf{X}) \) corresponding to the eigenvalue
\( \lambda_i \). Define \( M_p \triangleq [p_1 \ p_2 \ \cdots \ p_{m+n}] \) and \( M_y \triangleq [y_1 \ y_2 \ \cdots \ y_{m+n}] = M_p^{-H} \), where \( H \) is the transpose and conjugate operator and \( y_i \) the reciprocal left eigenvector related to \( \lambda_i \). Then

\[
\frac{\partial \lambda_i}{\partial \mathbf{X}} = \begin{bmatrix}
\frac{\partial \lambda_i}{\partial x_1} & \cdots & \frac{\partial \lambda_i}{\partial x_{N-L+n+1}} \\
\vdots & \ddots & \vdots \\
\frac{\partial \lambda_i}{\partial x_{i+n}} & \cdots & \frac{\partial \lambda_i}{\partial x_N}
\end{bmatrix} = M_1^T y_i^* p_i^T M_2^T
\]

(10)

where the superscript * denotes the conjugate operation and \( T \) the transpose operator.

Next, we have the following result

**Lemma 2** For \( \mathbf{X} \), \( \mathbf{A}(\mathbf{X}) \) and \( \{ \lambda_i \} \) as defined in lemma 1,

\[
\frac{\partial |\lambda_i|}{\partial \mathbf{X}} = \frac{1}{|\lambda_i|} \text{Re} \left[ \lambda_i^* \frac{\partial \lambda_i}{\partial \mathbf{X}} \right]
\]

(11)

where \( \text{Re}[\cdot] \) denotes the real part.

**Proof:** Noting \( |\lambda_i| = \sqrt{\lambda_i^* \lambda_i} \) leads to

\[
\frac{\partial |\lambda_i|}{\partial \mathbf{X}} = \frac{1}{2\sqrt{\lambda_i^* \lambda_i}} \left( \frac{\partial \lambda_i^*}{\partial \mathbf{X}} \lambda_i + \lambda_i^* \frac{\partial \lambda_i}{\partial \mathbf{X}} \right) = \frac{1}{2|\lambda_i|} \left( (\frac{\partial \lambda_i^*}{\partial \mathbf{X}})^* \lambda_i + \lambda_i^* \frac{\partial \lambda_i}{\partial \mathbf{X}} \right) = \frac{1}{|\lambda_i|} \text{Re} \left[ \lambda_i^* \frac{\partial \lambda_i}{\partial \mathbf{X}} \right]
\]

(12)

Combining lemma 1 with lemma 2 results in the following proposition, which shows that, given \( \mathbf{X} \), the value of \( \mu_1(\mathbf{X}) \) can easily be calculated.

**Proposition 1** For \( \mathbf{X}, \ M_1, \ M_2, \ \mathbf{A}(\mathbf{X}), \ \{ \lambda_i \}, \ p_i \) and \( y_i \) as defined in lemma 1,

\[
\frac{\partial |\lambda_i|}{\partial \mathbf{X}} = \begin{bmatrix}
\frac{\partial |\lambda_i|}{\partial x_1} & \cdots & \frac{\partial |\lambda_i|}{\partial x_{N-L+n+1}} \\
\vdots & \ddots & \vdots \\
\frac{\partial |\lambda_i|}{\partial x_{i+n}} & \cdots & \frac{\partial |\lambda_i|}{\partial x_N}
\end{bmatrix} = \frac{1}{|\lambda_i|} M_1^T \text{Re} \left[ \lambda_i^* y_i^* p_i^T \right] M_2^T
\]

(13)

It should be emphasized that the FWL stability related measure (8) is different with the one used in [5],[9],[10], which is given by

\[
\mu_2(\mathbf{X}) = \min_{i \in \{1, \cdots, m+n\}} \frac{1 - |\lambda_i(\mathbf{A}(\mathbf{X}))|}{\sqrt{N_s \sum_{j=1}^N \delta(x_j) \left| \frac{\partial \lambda_i}{\partial x_j} \right|^2}}
\]

(14)

The key difference between \( \mu_1(\mathbf{X}) \) and \( \mu_2(\mathbf{X}) \) is that the former considers the sensitivity of \( |\lambda_i(\mathbf{A}(\mathbf{X}))| \) while the latter considers the sensitivity of \( \lambda_i(\mathbf{A}(\mathbf{X})) \). It is well-known that the stability of a linear discrete-time system depends only on the moduli of its eigenvalues. As \( \mu_2(\mathbf{X}) \)
includes the unnecessary eigenvalue arguments in consideration, it is generally conservative in comparison with $\mu_1(\mathbf{X})$. This can be verified strictly. From lemma 2,

$$\left| \frac{\partial \lambda_i(\mathbf{X})}{\partial x_j} \right| \leq \left| \frac{\lambda_i(\mathbf{X}) \frac{\partial \lambda_i(\mathbf{X})}{\partial x_j}}{\lambda_i(\mathbf{X})} \right| = \left| \frac{\partial \lambda_i(\mathbf{X})}{\partial x_j} \right|$$

(15)

which means that $\mu_2(\mathbf{X}) \leq \mu_1(\mathbf{X})$. The result given in [7] has confirmed that by considering the sensitivity of eigenvalue moduli rather than the sensitivity of eigenvalues, a better FWL closed-loop stability related measure can be obtained. It is worth pointing out that the proposed measure $\mu_1(\mathbf{X})$ also has considerable computational advantages over the existing $\mu_2(\mathbf{X})$. This is because $\frac{\partial \lambda_i}{\partial \mathbf{X}}$ is real-valued while $\frac{\partial \lambda_i}{\partial x_j}$ is complex-valued. Thus the optimisation process and sparse transformation procedure, discussed in the next section, require much less computation than the previous approach [5],[9],[10], unless all the system eigenvalues are real-valued in which case $\mu_1(\mathbf{X})$ and $\mu_2(\mathbf{X})$ become identical.

3 Suboptimal controller realizations with sparse structures

The optimal sparse controller realization with a maximum tolerance to FWL perturbation in principle is the solution of the following optimization problem:

$$\omega \triangleq \max_{\mathbf{X} \in \mathcal{S}_c} \mu_1(\mathbf{X})$$

(16)

However, it is difficult to solve for the above optimization problem because $\mu_1(\mathbf{X})$ includes $\delta(x_j)$ and is not a continuous function with respect to controller parameters $x_j$. To get around this difficulty, we consider a lower bound of $\mu_1(\mathbf{X})$ defined by

$$\mu_1(\mathbf{X}) = \min_{i \in \{1, \cdots, m+n\}} \frac{1 - |\lambda_i(\mathbf{X})|}{\sqrt{N} \sum_{j=1}^{N} \left| \frac{\partial \lambda_i}{\partial x_j} \right|^2}$$

(17)

Obviously, $\mu_1(\mathbf{X}) \leq \mu_1(\mathbf{X})$ and $\mu_1(\mathbf{X})$ is a continuous function of controller parameters. It is relatively easy to optimize $\mu_1(\mathbf{X})$ (e.g. [7]). Let the “optimal” controller realization $\mathbf{X}_{\text{opt}}$ be the solution of the optimization problem

$$\omega \triangleq \max_{\mathbf{X} \in \mathcal{S}_c} \mu_1(\mathbf{X})$$

(18)

Notice that $\mathbf{X}_{\text{opt}}$ is generally not the optimal solution of (16) and does not have a sparse structure. However, it can readily be attempted by the following optimization procedure.
3.1 Optimization of the lower-bound measure

Assume that an initial controller realization has been obtained by some design procedure and is denoted as \( \mathbf{X}_0 \). According to (1)–(3), a similarity transformation of \( \mathbf{X}_0 \) by \( \mathbf{T} \) is

\[
\mathbf{X} = \mathbf{X}(\mathbf{T}) = \begin{bmatrix} \mathbf{I}_t & 0 \\ 0 & \mathbf{T}^{-1} \end{bmatrix} \mathbf{X}_0 \begin{bmatrix} \mathbf{I}_q & 0 \\ 0 & \mathbf{T} \end{bmatrix}
\]

where \( \det(\mathbf{T}) \neq 0 \). The closed-loop system matrix for the realization \( \mathbf{X} \) is

\[
\overline{\mathbf{A}}(\mathbf{X}) = \begin{bmatrix} \mathbf{I}_m & 0 \\ 0 & \mathbf{T}^{-1} \end{bmatrix} \overline{\mathbf{A}}(\mathbf{X}_0) \begin{bmatrix} \mathbf{I}_m & 0 \\ 0 & \mathbf{T} \end{bmatrix}
\]

(20)

Obviously, \( \overline{\mathbf{A}}(\mathbf{X}) \) has the same set of eigenvalues as \( \overline{\mathbf{A}}(\mathbf{X}_0) \), denoted as \( \{ \lambda_i^0 \} \). From (20), applying proposition 1 results in

\[
\frac{\partial |\lambda_i|}{\partial \mathbf{X}} \bigg|_{\mathbf{X}(\mathbf{T})} = \begin{bmatrix} \mathbf{I}_t & 0 \\ 0 & \mathbf{T}^T \end{bmatrix} \frac{\partial |\lambda_i|}{\partial \mathbf{X}} \bigg|_{\mathbf{X}_0} \begin{bmatrix} \mathbf{I}_q & 0 \\ 0 & \mathbf{T}^T \end{bmatrix}
\]

(21)

For a complex-valued matrix \( \mathbf{M} \in \mathbb{C}^{(i+n) \times (q+n)} \) with elements \( m_{s,k} \), denote the Frobenius norm

\[
\| \mathbf{M} \|_F = \sqrt{\sum_{s=1}^{i+n} \sum_{k=1}^{q+n} m_{s,k}^2}
\]

(22)

Then the lower-bound measure (17) can be rewritten as

\[
\mu_1(\mathbf{X}) = \min_{i \in \{1, \ldots, m+n\}} \frac{1 - |\lambda_i^0|}{\sqrt{N} \| \begin{bmatrix} \mathbf{I}_t & 0 \\ 0 & \mathbf{T}^T \end{bmatrix} \frac{\partial |\lambda_i|}{\partial \mathbf{X}} \bigg|_{\mathbf{X}_0} \begin{bmatrix} \mathbf{I}_q & 0 \\ 0 & \mathbf{T}^T \end{bmatrix} \|_F}
\]

(23)

where

\[
\Phi_i = \frac{\partial |\lambda_i|}{\partial \mathbf{X}} \bigg|_{\mathbf{X}_0} \begin{bmatrix} \mathbf{I}_q & 0 \\ 0 & \mathbf{T}^T \end{bmatrix}
\]

(24)

are fixed matrices that are independent of \( \mathbf{T} \). Thus, if we introduce the cost function

\[
f(\mathbf{T}) = \min_{i \in \{1, \ldots, m+n\}} \frac{1}{\sqrt{N} \| \begin{bmatrix} \mathbf{I}_t & 0 \\ 0 & \mathbf{T}^T \end{bmatrix} \Phi_i \begin{bmatrix} \mathbf{I}_q & 0 \\ 0 & \mathbf{T}^T \end{bmatrix} \|_F} = \mu_1(\mathbf{X})
\]

(25)

the optimal similarity transformation \( \mathbf{T}_{opt} \) can be obtained by solving for the following unconstrained optimization problem

\[
\omega = \max_{\mathbf{T} \in \mathbb{R}^{n \times n}} f(\mathbf{T})
\]

(26)

with a measure of monitoring the singular values of \( \mathbf{T} \) to make sure that \( \det(\mathbf{T}) \neq 0 \) [12]. The unconstrained optimization problem (26) can be solved, for example, using the simplex search
algorithm [13], the simulated annealing algorithm [14], the ASA algorithm [15] or the genetic algorithm [16]. In our previous study, we have found that the ASA is very efficient in solving for this kind of optimization problems [7]. With \( T_{\text{opt}} \), the corresponding optimal realization \( \mathbf{X}_{\text{opt}} \) that is the solution of (18) can readily be computed.

3.2 Stepwise transformation algorithm for sparse realizations

As the optimal sparse realization that maximizes \( \mu_1 \) is difficult if not impossible to obtain, we will search for a suboptimal solution of (16). More precisely, we will search for a realization that is sparse with a large enough value of \( \mu_1 \). Since \( \mathbf{X}_{\text{opt}} \) maximizes \( \underline{\mu_1} \) and \( \underline{\mu_1} \) is a lower-bound of \( \mu_1 \), \( \mathbf{X}_{\text{opt}} \) will produce a satisfactory large value of \( \mu_1 \), although it usually contains no trivial elements. We can make \( \mathbf{X}_{\text{opt}} \) sparse by changing one nontrivial element of \( \mathbf{X}_{\text{opt}} \) into a trivial one at a step, under the constraint that the value of \( \underline{\mu_1} \) does not reduce too much. This process will produce a sparse realization \( \mathbf{X}_{\text{spa}} \) with a satisfactory value of \( \underline{\mu_1} \). Clearly such a \( \mathbf{X}_{\text{spa}} \) is not a true optimal solution of (16). Notice that, even though \( \underline{\mu_1}(\mathbf{X}_{\text{spa}}) \leq \underline{\mu_1}(\mathbf{X}_{\text{opt}}) \), it is possible that \( \mu_1(\mathbf{X}_{\text{spa}}) \geq \mu_1(\mathbf{X}_{\text{opt}}) \). In other words, \( \mathbf{X}_{\text{spa}} \) may actually achieve better FWL stability performance than \( \mathbf{X}_{\text{opt}} \). The design procedure is very similar to the one used in [9], [10]. We now describe the detailed stepwise procedure for obtaining \( \mathbf{X}_{\text{spa}} \).

**Step 1:** Set \( \tau \) to a very small positive real number (e.g., \( 10^{-5} \)). The transformation matrix \( \mathbf{T} \in \mathbb{R}^{n \times n} \) is initially set to \( T_{\text{opt}} \) so that \( \mathbf{X}(\mathbf{T}) = \mathbf{X}_{\text{opt}} \).

**Step 2:** Find out all the trivial elements \( \{\eta_1, \ldots, \eta_r\} \) in \( \mathbf{X}(\mathbf{T}) \) (a parameter is considered to be trivial if its distance to 0, 1 or -1 is less than a tolerance value, say \( 10^{-8} \)). Denote \( \xi \) the nontrivial element in \( \mathbf{X}(\mathbf{T}) \) that is the nearest to 0, 1 or -1.

**Step 3:** Choose \( \mathbf{S} \in \mathbb{R}^{n \times n} \) such that

i) \( \underline{\mu_1}(\mathbf{X}(\mathbf{T} + \tau \mathbf{S})) \) is close to \( \underline{\mu_1}(\mathbf{X}(\mathbf{T})) \).

ii) \( \{\eta_1, \ldots, \eta_r\} \) in \( \mathbf{X}(\mathbf{T}) \) remain unchanged in \( \mathbf{X}(\mathbf{T} + \tau \mathbf{S}) \).

iii) \( \xi \) in \( \mathbf{X}(\mathbf{T}) \) is changed as nearer as possible to 0, 1 or -1 in \( \mathbf{X}(\mathbf{T} + \tau \mathbf{S}) \).

iv) \( \|\mathbf{S}\|_F = 1. \)

If \( \mathbf{S} \) does not exist, \( \mathbf{T}_{\text{spa}} = \mathbf{T} \) and terminate the algorithm.

**Step 4:** \( \mathbf{T} = \mathbf{T} + \tau \mathbf{S} \). If \( \xi \) in \( \mathbf{X}(\mathbf{T}) \) is nontrivial, go to step 3. If \( \xi \) becomes trivial, go to step 2.
The key of the above algorithm is Step 3 which guarantees that $X(T_{\text{ppa}})$ has good performance as measured by $\mu_1$ and contains many trivial parameters. We now discuss how to obtain $S$. Denote $\text{Vec}(-)$ the column stacking operator. With a very small $\tau$, condition i) means that

$$\left( \text{Vec} \left( \frac{d\mu_1}{dT} \right) \right)^T \text{Vec} (S) = 0$$

(27)

and condition ii) means that

$$\begin{cases}
\left( \text{Vec} \left( \frac{dn_t}{dT} \right) \right)^T \text{Vec} (S) = 0 \\
\vdots \\
\left( \text{Vec} \left( \frac{dn_r}{dT} \right) \right)^T \text{Vec} (S) = 0
\end{cases}$$

(28)

Denote the matrix

$$E \triangleq \begin{bmatrix}
\left( \text{Vec} \left( \frac{dn_1}{dT} \right) \right)^T \\
\left( \text{Vec} \left( \frac{dn_2}{dT} \right) \right)^T \\
\vdots \\
\left( \text{Vec} \left( \frac{dn_r}{dT} \right) \right)^T
\end{bmatrix} \in \mathcal{R}^{(r+1) \times n^2}$$

(29)

$\text{Vec}(S)$ must belong to the null space $\mathcal{N}(E)$ of $E$. If $\mathcal{N}(E)$ is empty, $\text{Vec}(S)$ does not exist and the algorithm is terminated. If $\mathcal{N}(E)$ is not empty, it must have basis $\{b_1, \cdots, b_t\}$, assuming that the dimension of $\mathcal{N}(E)$ is $t$. Condition iii) requires moving $\xi$ to its desired value (0, 1 or -1) as fast as possible, and we should choose $\text{Vec}(S)$ as the orthogonal projection of $\text{Vec} \left( \frac{d\xi}{dT} \right)$ onto $\mathcal{N}(E)$. Noting condition iv), we can compute $\text{Vec}(S)$ as follows:

$$a_i = b_i^T \text{Vec} \left( \frac{d\xi}{dT} \right) \in \mathcal{R}, \quad \forall i \in \{1, \cdots, t\}$$

(30)

$$v = \sum_{i=1}^{t} a_i b_i \in \mathcal{R}^{n^2}$$

(31)

$$\text{Vec}(S) = \pm \frac{v}{\sqrt{v^T v}} \in \mathcal{R}^{n^2}$$

(32)

The sign in (32) is chosen in the following way. If $\xi$ is larger than its nearest desired value, the minus sign is taken; otherwise, the plus sign is used.

In the above algorithm, the derivatives $\frac{dn_1}{dT}, \frac{dn_2}{dT}, \cdots, \frac{dn_r}{dT}$ are needed. For calculating these required derivatives, the following well-known fact is useful. Given any element $y_{ij}$ in a nonsingular $Y \in \mathcal{R}^{n \times n}$ with $i \in \{1, \cdots, n\}$ and $j \in \{1, \cdots, n\}$,

$$\frac{\partial Y}{\partial y_{ij}} = e_i e_j^T \quad \text{and} \quad \frac{\partial Y^{-1}}{\partial y_{ij}} = -Y^{-1} e_i e_j^T Y^{-1}$$

(33)

where $e_i$ denotes the $i$th coordinate vector. In (19), define

$$U_1 = \begin{bmatrix} I_r & 0 \\ 0 & T \end{bmatrix} \quad \text{and} \quad U_2 = \begin{bmatrix} I_{q} & 0 \\ 0 & T \end{bmatrix}$$

(34)
For any element $x_{k,s}$ in $X = U_1^{-1}X_0U_2$, where $k \in \{1, \ldots, l+n\}$ and $s \in \{1, \ldots, q+n\}$, and any $t_{ij}$ in $T$, where $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, n\}$,

$$
\frac{\partial x_{k,s}}{\partial t_{ij}} = e_k^T U_1^{-1} \frac{\partial U_1^{-1}}{\partial t_{ij}} X_0 U_2 e_s + e_k^T U_1^{-1} \frac{\partial U_2}{\partial t_{ij}} e_s
$$

$$
= -e_k^T U_1^{-1} e_{t+i}^T e_{t+j}^T U_1^{-1} X_0 U_2 e_s + e_k^T U_1^{-1} X_0 e_{q+i}^T e_{q+j}^T e_s
$$

$$
= -e_k^T U_1^{-1} e_{t+i}^T e_{t+j}^T X e_s + e_k^T U_1^{-1} X_0 e_{q+i}^T e_{q+j}^T e_s
$$

That is,

$$
\frac{dx_{k,s}}{dT} = \begin{bmatrix}
    e_k^T U_1^{-1} \\
    \vdots \\
    e_k^T U_1^{-1}
\end{bmatrix}
\begin{bmatrix}
    X_0 e_{q+1}^T e_{q+1} \cdots X_0 e_{q+n}^T e_{q+n} \\
    \vdots \\
    \vdots
\end{bmatrix}
\begin{bmatrix}
    e_s \\
    \vdots \\
    e_s
\end{bmatrix}
\begin{bmatrix}
    X_0 e_{q+1}^T e_{q+1} \cdots X_0 e_{q+n}^T e_{q+n} \\
    \vdots \\
    \vdots
\end{bmatrix}
\begin{bmatrix}
    e_s \\
    \vdots \\
    e_s
\end{bmatrix}
\begin{bmatrix}
    e_{t+i}^T e_{t+j}^T X \\
    \vdots \\
    e_{t+i}^T e_{t+j}^T X
\end{bmatrix}
\begin{bmatrix}
    e_s \\
    \vdots \\
    e_s
\end{bmatrix}
\begin{bmatrix}
    e_{t+i}^T e_{t+j}^T X \\
    \vdots \\
    e_{t+i}^T e_{t+j}^T X
\end{bmatrix}
\begin{bmatrix}
    e_s \\
    \vdots \\
    e_s
\end{bmatrix}
$$

Thus, we can readily calculate $\frac{d\mu_1}{dT}$, $\frac{d\mu_2}{dT}$, $\ldots$, $\frac{d\mu_n}{dT}$. Next, define

$$
i_0 = \arg \min_{i \in \{1, \ldots, m+n\}} \frac{1}{\sqrt{N} \left\| \Phi_i \right\|_F} \left\| \frac{\sum_{k=1}^{l+n} \sum_{s=1}^{q+n} w_{k,s} w_{k,s}}{\sum_{k=1}^{l+n} \sum_{s=1}^{q+n} w_{k,s}} \right\|_F
$$

Similar to the derivation of $\frac{d\mu_1}{dT}$, for any element $w_{k,s}$ in $W = U_1^T \Phi i_0 U_2^{-T}$, where $k \in \{1, \ldots, l+n\}$ and $s \in \{1, \ldots, q+n\}$, we have

$$
\frac{dw_{k,s}}{dT} = \begin{bmatrix}
    e_k^T \\
    \vdots \\
    e_k^T
\end{bmatrix}
\begin{bmatrix}
    \begin{bmatrix}
    X_0 e_{q+1}^T e_{q+1} \cdots X_0 e_{q+n}^T e_{q+n} \\
    e_{t+i}^T e_{t+j}^T X \\
    \vdots \\
    e_{t+i}^T e_{t+j}^T X
\end{bmatrix}
\end{bmatrix}
\begin{bmatrix}
    e_s \\
    \vdots \\
    e_s
\end{bmatrix}
\begin{bmatrix}
    \begin{bmatrix}
    X_0 e_{q+1}^T e_{q+1} \cdots X_0 e_{q+n}^T e_{q+n} \\
    e_{t+i}^T e_{t+j}^T X \\
    \vdots \\
    e_{t+i}^T e_{t+j}^T X
\end{bmatrix}
\end{bmatrix}
\begin{bmatrix}
    e_s \\
    \vdots \\
    e_s
\end{bmatrix}
$$

Since

$$
\mu_1 = \frac{1}{\sqrt{N} \sqrt{\sum_{k=1}^{l+n} \sum_{s=1}^{q+n} w_{k,s} w_{k,s}}}
$$

We can calculate

$$
\frac{d\mu_1}{dT} = -\frac{1}{\sqrt{N \left\| W \right\|_F^2}} \text{Re} \left[ \sum_{k=1}^{l+n} \sum_{s=1}^{q+n} w_{k,s} \frac{dw_{k,s}}{dT} \right]
$$

Before presenting some simulation results, we point out that given a FWL pole-sensitivity measure, such as $\mu_1(X)$, an estimated minimum bit length for guaranteeing closed-loop stability can be estimated using [6],[7]

$$
\hat{B}_{s,\min} = B_s + \text{Int} \left[ -\log_2 (\mu_1(X)) \right] - 1
$$

where the integer $\text{Int}[x] \geq x$. 

10
4 Numerical examples

We present two design examples to show how our approach can be used efficiently to search for sparse controller realizations with satisfactory FWL closed-loop stability performance.

Example 1. This was a single-input single-output fluid power speed control system studied in [17],[18]. The plant model was in the continuous-time form and a continuous-time $H_\infty$ optimal controller was designed in [17]. In this study, we obtained a discrete-time plant $P(z)$ and a discrete-time controller $C(z)$ by sampling the continuous-time plant and $H_\infty$ controller using a sampling rate of 2 kHz. The discrete-time plant $P(z)$ was given by

$$A_P = \begin{bmatrix}
9.9988e-01 & 1.9432e-05 & 5.9320e-05 & -6.2286e-05 \\
-4.9631e-07 & 2.3577e-02 & 2.3709e-05 & 2.3672e-05 \\
-1.5151e-03 & 2.3709e-02 & 2.3751e-05 & 2.3898e-05 \\
1.5908e-03 & 2.3672e-02 & 2.3898e-05 & 2.3667e-05
\end{bmatrix}, \quad B_P = \begin{bmatrix}
3.0504e-03 \\
-1.2373e-02 \\
-1.2375e-02 \\
-8.8703e-02
\end{bmatrix}, \quad C_P = \begin{bmatrix}
1 & 0 & 0 & 0
\end{bmatrix}$$

The initial realization of the controller $C(z)$ given in a controllable canonical form was

$$X_0 = \begin{bmatrix}
-8.0843e-04 & -1.6112e-03 & -1.5998e-03 & -1.5885e-03 & -1.5773e-03 \\
1 & 0 & 0 & 0 & -3.3071e-01 \\
0 & 1 & 0 & 0 & 1.9869e+00 \\
0 & 0 & 1 & 0 & -3.9816e+00 \\
0 & 0 & 0 & 1 & 3.3255e+00
\end{bmatrix}$$

Notice that the controllable canonical form was very sparse, containing only 9 non-trivial elements. The closed-loop transition matrix $\bar{A}(X_0)$ was then formed using (3), from which the eigenvalues and the corresponding eigenvectors of the ideal (infinite-precision) closed-loop system were computed. The closed-loop eigenvalues were:

$$\lambda_1 = 9.956e-01 + j 2.5674e-04$$
$$\lambda_2 = 9.956e-01 - j 2.5674e-04$$
$$\lambda_3 = 9.9955e-01$$
$$\lambda_4 = 9.9333e-01$$
$$\lambda_5 = 3.3333e-01$$
$$\lambda_6 = 2.3625e-02$$
$$\lambda_7 = 2.7819e-19$$
$$\lambda_8 = -3.8735e-09$$

The optimisation problem (26) was constructed, and the ASA algorithm [15] obtained the following solution

$$T_{\text{opt}} = \begin{bmatrix}
2.3644e+07 & 2.0268e+06 & 1.0498e+08 & 4.7194e+06 \\
-1.1839e+08 & -9.9623e+06 & -5.2570e+08 & 2.3636e+07 \\
1.6622e+08 & 1.3872e+07 & 7.3801e+08 & -3.191e+07 \\
-7.1475e+07 & -5.9364e+06 & -3.1729e+08 & 1.4274e+07
\end{bmatrix}.$$
The corresponding controller realization, which maximises the lower-bound measure $\mu_1$, was

$$X_{\text{opt}} = \begin{bmatrix}
2.7588e-03 & 1.0010e+00 & -1.4054e-02 & 1.0924e-03 & -8.9552e-03 \\
-2.2776e-04 & -5.8175e-02 & 3.3649e-01 & 7.5457e-02 & 1.3962e-03 \\
-2.5200e-04 & 1.0668e-03 & 1.6778e-02 & 9.7966e-01 & 1.5423e-03 \\
8.1179e-03 & 5.1520e-03 & 3.1311e-02 & -3.8681e-03 & 9.9031e-01
\end{bmatrix}$$

The stepwise transformation algorithm was then applied to make $X_{\text{opt}}$ sparse, which yielded the following similarity transformation matrix and corresponding controller realization

$$T_{\text{spa}} = \begin{bmatrix}
-1.7499e+05 & -4.5848e+05 & 2.1159e+08 & 3.0140e+02 \\
8.1616e+05 & 1.8611e+06 & -1.0592e+09 & -1.2931e+03 \\
-1.0789e+06 & -2.3503e+06 & 1.4869e+09 & 1.8162e+03 \\
4.3753e+05 & 9.4770e+05 & -6.3921e+08 & -7.8105e+02
\end{bmatrix}$$

$$X_{\text{spa}} = \begin{bmatrix}
-8.0843e-04 & 1.6372e-02 & -5.4228e-04 & -1.8348e-03 & -6.9866e-02 \\
0 & 1 & 0 & 0 & -1.4073e-03 \\
0 & -6.8678e-02 & 3.3285e-01 & 4.2230e-01 & 5.8895e-04 \\
0 & -5.6623e-06 & -7.6002e-04 & 1 & 0 \\
\end{bmatrix}$$

Table 1 compares the FWL closed-loop stability performance and the number of non-trivial elements for the three controller realizations $X_0$, $X_{\text{opt}}$ and $X_{\text{spa}}$, respectively. For a comparison purpose, the values of the previous stability related measure $\mu_2$ and its lower-bound $\mu_2^*$ together with their corresponding estimated minimum bit lengths [9],[10] are also given in Table 1 for the three realizations. We also exploited the true minimum bit length that guaranteed closed-loop stability for a controller realization $X$ using the following computer simulation. Starting with a large enough bit length, e.g. $B_s = 1000$, we rounded the controller $X$ to $B_s$ bits and checked the stability of the closed-loop system, i.e. observing whether the closed-loop poles were within the open unit disk. Reduced $B_s$ by 1 and repeated the process until there appeared to be closed-loop instability at $B_u$ bits. Then $B_{s,\text{min}} = B_u + 1$. The values of $B_{s,\text{min}}$ for the three realizations are given in Table 1. Notice that for $B_s \geq B_{s,\text{min}}$, the $B_s$-bit implemented controller will always guarantee closed-loop stability. However, there may exist some $B_s < B_u$, which regains closed-loop stability. For example, for the initial realization $X_0$, $B_u = 32$, i.e. when the bit length is smaller than 33, the closed-loop becomes unstable. At $B_s = 16$ or 15, the closed-loop becomes stable again. With $B_s < 15$ instability is observed again.

For this example, the canonical realization $X_0$ is the most sparse with only 9 non-trivial parameters, but its FWL closed-loop stability related measure $\mu_1(X_0)$ is very poor. The realization $X_{\text{opt}}$ has a much better FWL stability robustness as indicated by $\mu_1(X_{\text{opt}})$, but its all 25 elements are non-trivial. The realization $X_{\text{spa}}$ has the largest $\mu_1(X_{\text{spa}})$ and, moreover, it is sparse.
with only 16 non-trivial parameters. This example only has a pair of complex eigenvalues. Even so, the results shown in Table 1 indicate that the proposed $\mu_1$ ($\mu_1$ respectively) is less conservative in estimating the robustness of FWL closed-loop stability than the previous measure $\mu_2$ ($\mu_2$ respectively)\footnote{If $\arg \mu_1 = \arg \mu_2 = \imath_0$ ($\arg \mu_1 = \arg \mu_2$ respectively) and $\lambda_{\imath_0}$ is real valued, then obviously $\mu_1 = \mu_2$ ($\mu_1 = \mu_2$ respectively).}. We also computed the unit impulse response of the closed-loop control system when the controllers were the infinite-precision implemented $X_0$ and 16-bit implemented three different controller realizations. Notice that any realization $X \in \mathcal{X}_C$ implemented in infinite precision will achieve the exact performance of the infinite-precision implemented $X_0$, which is the designed controller performance. For this reason, the the infinite-precision implemented $X_0$ is referred to as the ideal controller realization $X_{\text{ideal}}$. Fig. 1 compares the unit impulse response of the plant output $y(k)$ for the ideal controller $X_{\text{ideal}}$ with those of the 16-bit implemented $X_0$, $X_{\text{opt}}$ and $X_{\text{spx}}$. It can be seen that the performance of the 16-bit implemented $X_{\text{spx}}$ is almost identical to that of the 16-bit implemented $X_{\text{opt}}$, which is very close to the ideal performance.

**Example 2.** This was a dual wrist assembly which was a prototype telerobotic system used in micro-surgery experiments [19]. This dual wrist assembly is a two-input ($l = 2$) two-output ($q = 2$) system with a plant order $m = 4$, and the digital controller designed using $\mathcal{H}_\infty$ method had an order of $n = 10$ [19]. The total number of controller parameters was $N = 144$. The $\mathcal{H}_\infty$ controller designed in [19], which was fully parameterised with $N_s = N$, was used as the initial controller realization $X_0$, and the realization $X_{\text{opt}}$ that maximized the lower-bound measure $\mu_1$ was obtained using the ASA algorithm. This realization was then made sparse using the algorithm given in subsection 3.2 to yield $X_{\text{spx}}$. Table 2 summarizes the performance of these three different controller realizations. It can be seen that the proposed measure $\mu_1$ ($\mu_1$ respectively) yielded less conservative results in estimating the robustness of FWL closed-loop stability than the previous measure $\mu_2$ ($\mu_2$ respectively).

Fig. 2 compares the first-input to first-output unit impulse response of the closed-loop system obtained using the ideal controller $X_{\text{ideal}}$ with those obtained using the 20-bit implemented controller realizations $X_{\text{opt}}$ and $X_{\text{spx}}$. The 20-bit implemented $X_0$ is unstable and therefore is not shown. It can be seen that the performance of the 20-bit implemented $X_{\text{opt}}$ is close to the ideal performance, and the 20-bit implemented $X_{\text{spx}}$, although deviating from the ideal one, achieves a stable closed-loop performance. Fig. 3 compares the second-input to second-output ideal unit impulse response of the closed-loop system with those of the 24-bit implemented
\(X_0\), \(X_{opt}\) and \(X_{spa}\). It can be seen that the performance of the 24-bit implemented \(X_{spa}\) closely matches that of the 24-bit implemented \(X_{opt}\), which itself is almost identical to the ideal performance. Deviation from the ideal performance by the 24-bit implemented \(X_0\) can clearly be seen from Fig. 3. This example clearly demonstrates the effectiveness of the proposed design procedure. The sparse controller realization \(X_{spa}\) obtained has almost half of its parameters being trivial, and it has a much improved FWL closed-loop stability robustness over the initial controller realization \(X_0\).

5 Conclusions

We have studied FWL implementation of digital controller structures with sparseness consideration. A new FWL closed-loop stability related measure has been derived, which takes into account the number of trivial parameters in a controller realization. It has been shown that this new measure yields a more accurate estimate for the robustness of FWL closed-loop stability. A practical procedure has been presented to obtain sparse controller realizations with satisfactory FWL closed-loop stability characteristics. Two examples demonstrate that the proposed design procedure yields computationally efficient controller structures suitable for FWL implementation in real-time applications.
Acknowledgements

J. Wu and S. Chen wish to thank the support of the UK Royal Society under a KC Wong fellowship (RL/ART/CN/XFI/KCW/11949).

References


<table>
<thead>
<tr>
<th>realization</th>
<th>( X_0 )</th>
<th>( X_{opt} )</th>
<th>( X_{spa} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N_s )</td>
<td>9</td>
<td>25</td>
<td>16</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_1 )</td>
<td>2.604531e-12</td>
<td>6.862889e-05</td>
<td>6.108122e-05</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_1 )</td>
<td>4.417941e-12</td>
<td>6.862889e-05</td>
<td>1.348887e-04</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_2 )</td>
<td>2.604531e-12</td>
<td>5.500982e-05</td>
<td>6.108052e-05</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_2 )</td>
<td>4.417941e-12</td>
<td>5.500982e-05</td>
<td>1.348839e-04</td>
</tr>
<tr>
<td>( B_{s,\text{min}} )</td>
<td>33</td>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 1: Performance comparison of the three different controller realizations for Example 1.

<table>
<thead>
<tr>
<th>realization</th>
<th>( X_0 )</th>
<th>( X_{opt} )</th>
<th>( X_{spa} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N_s )</td>
<td>144</td>
<td>144</td>
<td>75</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_1 )</td>
<td>4.306085e-04</td>
<td>3.224443e-03</td>
<td>1.279414e-03</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_1 )</td>
<td>4.306085e-04</td>
<td>3.224443e-03</td>
<td>2.331625e-03</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_2 )</td>
<td>1.173382e-04</td>
<td>1.057405e-03</td>
<td>4.393420e-04</td>
</tr>
<tr>
<td>( \hat{B}_{s,\text{min}} ) based on ( \mu_2 )</td>
<td>1.173382e-04</td>
<td>1.057405e-03</td>
<td>9.249032e-04</td>
</tr>
<tr>
<td>( B_{s,\text{min}} )</td>
<td>22</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 2: Performance comparison of the three different controller realizations for Example 2.
Figure 1: Comparison of unit impulse response of the infinite-precision controller implementation $X_{ideal}$ with those of the three 16-bit implemented controller realizations $X_0$, $X_{opt}$ and $X_{spa}$ for Example 1.
Figure 2: Comparison of first-input first-output unit impulse response of the infinite-precision controller implementation $X_{\text{ideal}}$ with those of the 20-bit implemented controller realizations $X_{\text{opt}}$ and $X_{\text{spa}}$ for Example 2. The 20-bit implemented $X_0$ is unstable.
Figure 3: Comparison of second-input second-output unit impulse response of the infinite-precision controller implementation $X_{\text{ideal}}$ with those of the 24-bit implemented controller realizations $X_0$, $X_{\text{opt}}$ and $X_{\text{spa}}$ for Example 2.