Controller Coefficient Truncation Using Lyapunov Performance Certificate

Joëlle Skaf and Stephen P. Boyd

***Abstract*— We describe a method for truncating the coefficients of a linear controller while guaranteeing that a given set of relaxed performance constraints is met. Our method sequentially and greedily truncates individual coefficients, using a Lyapunov certificate, typically in linear matrix inequality (LMI) form, to guarantee performance. Numerical examples show that the method is surprisingly effective at finding controllers with aggressively truncated coefficients, that meet typical performance constraints.**

***Index Terms*— Coefficient truncation, Lyapunov function, linear matrix inequality, state-feedback controller, decay rate.**

I. Introduction

A. The controller coefficient truncation problem

We consider a discrete-time linear time-invariant control system, with plant

$$\begin{aligned} x_p(t+1) &= A_p x_p(t) + B_1 w(t) + B_2 u(t), \\ z(t) &= C_1 x_p(t) + D_{11} w(t) + D_{12} u(t), \\ y(t) &= C_2 x_p(t) + D_{21} w(t), \end{aligned}$$

and controller

$$\begin{aligned} x_c(t+1) &= A_c x_c(t) + B_c y(t), \\ u(t) &= C_c x_c(t) + D_c y(t). \end{aligned}$$

Here $t = 0, 1, 2, \ldots$ is the discrete time index, $x_p(t)$ is the plant state, $u(t)$ is the control input, $y(t)$ is the sensor output, $w(t)$ and $z(t)$ are the exogenous input and output, respectively, and $x_c(t)$ is the controller state.

The vector $\theta \in \mathbf{R}^N$ will represent the design parameters or coefficients in the controller. Typically these are (some of) the entries in the matrices A_c, B_c, C_c and D_c. We are given a *nominal controller design*, described by the coefficient vector θ^{nom}, and a set of acceptable controller designs $\mathcal{C} \subseteq \mathbf{R}^N$. The set $\mathcal{C}$ gives the (coefficients of the) controllers that achieve acceptable closed-loop performance. We assume that $\theta^{\mathrm{nom}} \in \mathcal{C}$, *i.e.*, the nominal controller meets the performance specifications. For example, we can give $\mathcal{C}$ in terms of a single scalar performance measure $J : \mathbf{R}^N \to \mathbf{R}$, as

$$\mathcal{C} = \{\theta \mid J(\theta) \leq (1+\epsilon) J(\theta^{\mathrm{nom}})\},$$

which are the designs that are no more than ϵ worse than the nominal design. If the nominal design is the controller that minimizes J, then $\mathcal{C}$ is the set of ϵ-suboptimal designs. Our goal is to find $\theta \in \mathcal{C}$ that achieves closed-loop performance close to nominal, and at the same time has low complexity.

The *complexity* of a vector of controller coefficients θ is measured by the function $\Phi : \mathbf{R}^N \to \mathbf{R}$,

$$\Phi(\theta) = \sum_{i=1}^{N} \phi_i(\theta_i),$$

where $\phi_i(\theta_i)$ gives the complexity of the ith coefficient of θ. We can take, for example, $\phi_i(a)$ to be the number of bits needed to express a, or the total number of 1s in the binary expansion of a, in which case $\Phi(\theta)$ gives the total number of bits (or 1s) in the controller coefficients, which is closely related to the Kolmogorov-Chaitin complexity of the controller [1]. Of course the functions ϕ_i, and therefore also Φ, can be discontinuous.

Our goal is to find the lowest complexity controller among the acceptable designs. We can express this as the optimization problem

$$\begin{array}{ll} \text{minimize} & \Phi(\theta) \\ \text{subject to} & \theta \in \mathcal{C}, \end{array} \tag{1}$$

with variable $\theta \in \mathbf{R}^N$. We call this the *controller coefficient truncation problem* (CCTP), since we can think of the controller coefficient θ_i as a truncated version of the nominal controller coefficient θ_i^{nom}.

The CCTP (1) is in general very difficult to solve. For example, when Φ measures bit complexity, the CCTP can be cast as a combinatorial optimization problem, with the binary expansions of the coefficients as Boolean (*i.e.*, $\{0,1\}$) variables. Branch-and-bound, or other global optimization techniques, could be used to solve small CCTPs, with perhaps 10 coefficients. But we are interested in methods that can handle much larger problems, with perhaps hundreds (or more) of controller coefficients. In addition, it is not crucial to find the global solution of the CCTP (1); it is enough to find a controller with low (if not lowest) complexity.

In this paper we describe a heuristic algorithm for the CCTP (1), that runs quickly and scales to large problems. While the designs produced are very likely not globally

This work was funded in part by Focus Center Research Program Center for Circuit and System Solutions (www.c2s2.org), under contract 2003-CT-888, by AFOSR grant AF F49620-01-1-0365, by NSF grant ECS-0423905, by NSF grant 0529426, by DARPA/MIT grant 5710001848, by AFOSR grant FA9550-06-1-0514, DARPA/Lockheed contract N66001-06-C-2021, and by AFOSR/Vanderbilt grant FA9550-06-1-0312.

J. Skaf is with the Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford CA 94305 jskaf@stanford.edu

S. Boyd is with the Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford CA 94305 boyd@stanford.edu

optimal, they appear to be quite good. The method typically produces aggressively truncated controller designs, even when the allowed performance degradation over the nominal design is just a few percent.

In our method, we greedily truncate individual coefficients sequentially, in random order, using a Lyapunov certificate (which is updated at each step) to guarantee performance, *i.e.*, $\theta \in \mathcal{C}$. When the algorithm is run multiple times, the randomness in the truncation order produces designs that are different, but have very similar total complexity. Running the algorithm a few times, and taking the best controller found, can give a modest improvement over running it just once.

Before proceeding we mention a related issue that we do *not* consider: the effects of truncation or saturation of the control signals $u(t)$, $y(t)$, and $x_c(t)$. This makes the entire control system nonlinear, and can lead to instability, large and small limit cycles, and other behavior. However, the Lyapunov-based methods described in this paper can be extended to handle nonlinearities.

B. Previous and related work

The subject of coefficient truncation is relatively old. It was initially discussed in the context of filter design: there was an understandable interest in designing finite wordlength filters that would be easily implemented in hardware with a small degradation in performance (see [2], [3]). The idea of coefficient truncation subsequently appeared in other fields like speech processing [4] and control [5].

Several methods have been proposed for coefficient truncation: exhaustive search over possible truncated coefficients [2], successive truncation of coefficients and reoptimization over remaining ones [3], [6], local bivariate search around the scaled and truncated coefficients [7], tree-traversal techniques for truncated coefficients organized in a tree according to their complexity [8], [9], coefficient quantization using information-theoretic bounds [10], weighted least-squares [11], simulated annealing [12], [13], genetic algorithms [14], [15], Tabu search [16], design of optimal filter realizations that minimize coefficient complexity [17], [12]. Other approaches have formulated the problem as a nonlinear discrete optimization problem [18], or have used integer programming techniques over the space of powers-of-two coefficients [19], [20]. The reference [21] surveys different methods for quantizing lifting coefficients for wavelet filters: mostly uniform bit allocation, exhaustively searched allocation, simulated annealing with lumped scaling and/or gain compensation. In [22], the authors show how to choose the optimal realization for an LQG controller to be robust to finite wordlength effects. The effects of quantization and finite wordlength on robust stability of digital controllers and performance bounds derived using Lyapunov theory are presented in [23].

C. Outline

In §II we describe the general algorithm. In the next three sections we present examples, in each case working out the details for the general case, and illustrating the algorithm with a numerical instance of the problem. In §III the controller has constant state-feedback form, the nominal controller is linear quadratic regular (LQR) optimal, and the set of acceptable controllers is determined by the LQR cost. In §IV the controller is dynamic, and the objective is the decay rate of the closed-loop system.

II. The algorithm

Our algorithm uses two subroutines or methods: $\mathbf{interv}$, which finds an interval of acceptable values of a coefficient, and $\mathbf{trunc}$, which truncates a coefficient, given an interval of acceptable choices. We first describe these methods more precisely, but still abstractly; more concrete descriptions will be given later in §II-A and §II-B.

The method $\mathbf{interv}(\theta, i)$ takes as input the coefficient vector $\theta \in \mathcal{C}$, and a coefficient index i. It returns an interval $[l, u]$ of allowed values for θ_i, with the other parameters held fixed, *i.e.*, numbers l and u, with $\theta_i \in [l, u]$, with

$$(\theta_1, \ldots, \theta_{i-1}, z, \theta_{i+1}, \ldots, \theta_N) \in \mathcal{C} \text{ for } z \in [l, u].$$

Of course the simple choice $l = u = \theta_i$ is always valid. At the other extreme, the largest valid interval that can be returned by $\mathbf{interv}$ is given by

$$l^\star = \inf\{l \quad | \quad (\theta_1, \ldots, \theta_{i-1}, z, \theta_{i+1}, \ldots, \theta_N) \in \mathcal{C} \text{ for } z \in [l, \theta_i]\},$$

$$u^\star = \sup\{u \quad | \quad (\theta_1, \ldots, \theta_{i-1}, z, \theta_{i+1}, \ldots, \theta_N) \in \mathcal{C} \text{ for } z \in [\theta_i, u]\}.$$

A typical implementation of $\mathbf{interv}$ falls between these two extremes, returning a reasonably large interval guaranteed to lie in $\mathcal{C}$, with reasonable computational effort. In the examples we will consider, this can be done using linear matrix inequalities (LMIs).

The method $\mathbf{trunc}_i(x, l, u)$ is a truncation method which, given a number x to be truncated, and an interval $[l, u]$ of acceptable choices (containing x), returns a number z in the interval $[l, u]$, with $\Phi_i(z) \leq \Phi_i(x)$. One valid choice is $z = x$; at the other extreme, the algorithm can return the point with smallest complexity in the interval, *i.e.*, the minimizer of $\Phi_i(z)$ over $[l, u]$. For the complexity measures we use in the examples shown later, we can easily compute the latter.

The algorithm is initialized with the nominal design, which we assume has finite complexity. At each step an index i is chosen, and all parameters except θ_i are fixed. We use $\mathbf{interv}$ to find an interval of acceptable values for θ_i, and then $\mathbf{trunc}$ to find a value of θ_i with (possibly) lower complexity. We have experimented with various methods for choosing the index i in each step, and found the best results by organizing the algorithm into passes, each of which involves updating each parameter once; in each pass, the ordering of the indices is random. The algorithm stops when the parameter does not change over one pass. A high-level decription of the algorithm is as follows.

$\theta := \theta^{\mathrm{nom}}$
repeat
  $\theta^{\mathrm{prev}} := \theta$

choose a permutation π of $(1, \ldots, N)$
for $i = 1$ **to** N
$j := \pi(i)$
$[l, u] := \mathbf{interv}(\theta, j)$
$\theta_j := \mathbf{trunc}_i(\theta_j, l, u)$
until $\theta = \theta^{\rm prev}$

Since the algorithm is random, it can and does converge to different points in different runs. It can be run several times, with the best controller coefficient vector found taken as our final choice.

A. Complexity measures and truncation methods

In this section we describe various possible complexity measures, and the associated truncation methods. Any $z \in \mathbf{R}$ can be written as

$$z = s \sum_{i=-\infty}^{\infty} b_i 2^{-i},$$

where $s \in \{-1, 1\}$ is the sign, and $b_i \in \{0, 1\}$ are the bits of z in a binary expansion. (This representation can be made unique by ruling out any sequence that ends with all ones, *i.e.*, $b_i = 1$ for $i \geq k$, for some k.)

One possible complexity measure is the number of ones in the binary expansion of z,

$$\phi_{\rm ones}(z) = \sum_{i=-\infty}^{\infty} b_i,$$

which gives the number of adders needed to implement multiplication by z using a shift and sum method.

Another complexity measure is the width of the range of the nonzero bits, more commonly referred to as the number of bits in the expansion of z,

$$\phi_{\rm bits}(z) = \max\{i \mid b_i \neq 0\} - \min\{i \mid b_i \neq 0\} + 1.$$

This measure is useful if multiplication by z will be carried out in fixed-point arithmetic.

Yet another complexity measure is the number of bits needed in the fractional part of the binary expansion of z,

$$\phi_{\rm frac-bits}(z) = \max\{0, \max_i\{i \mid b_i \neq 0\}\}.$$

For these complexity measures, it is straightforward to find the number z that minimizes the measure in a given interval, *i.e.*, to implement (the most powerful) $\mathbf{trunc}$ method. We assume that the binary expansions of l and u are finite (though possibly long),

$$s_l l_{-L} \ldots l_0 . l_1 \ldots l_R, \qquad s_u u_{-L} \ldots u_0 . u_1 \ldots u_R,$$

respectively. The number z will have at most L bits in its integer part and R bits in its fractional part, and we denote its bits as

$$s_z z_{-L} \ldots z_0 . z_1 \ldots z_R.$$

With complexity measure $\phi_{\rm ones}$ or $\phi_{\rm bits}$, z can be found as follows.

$z_i := 0$ for all i
for $i = -L$ **to** R
if $l_i = u_i$, $z_i := l_i$
else
if all bits after l_i **are** 0**, break**
else $z_i := 1$

When the complexity measure is $\phi_{\rm frac-bits}$, the same algorithm can be used, with z_i initially set to zero for $i > 0$ and the for loop index modified to run from 1 to R, instead of from $-L$ to R.

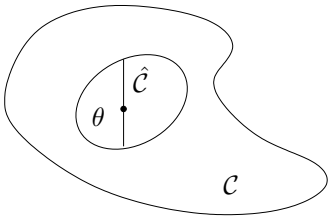

Fig. 1. The set $\mathcal{C}$ of acceptable design parameters need not be convex, as shown in this example. The set $\hat{\mathcal{C}}$ is a convex subset of $\mathcal{C}$, that contains θ. The interval of values of θ, shown as the vertical line segment, gives an interval of values in $\mathcal{C}$.

B. Interval computation via Lyapunov performance certificate

Our approach to determining an interval $[l, u]$ for which

$$(\theta_1, \ldots, \theta_{i-1}, z, \theta_{i+1}, \ldots, \theta_N) \in \mathcal{C} \text{ for } z \in [l, u]$$

will be based on a conservative approximation of $\mathcal{C}$. Given $\theta \in \mathcal{C}$ we first find a convex set $\hat{\mathcal{C}}$ that satisfies $\theta \in \hat{\mathcal{C}}$ and $\hat{\mathcal{C}} \subseteq \mathcal{C}$. We then take

$$\begin{aligned} l &= \inf\{z \mid (\theta_1, \ldots, \theta_{i+1}, z, \theta_{i+1}, \ldots, \theta_N) \in \hat{\mathcal{C}}\}, \\ u &= \sup\{z \mid (\theta_1, \ldots, \theta_{i+1}, z, \theta_{i+1}, \ldots, \theta_N) \in \hat{\mathcal{C}}\}. \end{aligned} \tag{2}$$

Since $\hat{\mathcal{C}}$ is convex, it follows that

$$(\theta_1, \ldots, \theta_{i+1}, z, \theta_{i+1}, \ldots, \theta_N) \in \hat{\mathcal{C}} \subseteq \mathcal{C} \text{ for } z \in [l, u].$$

This is illustrated in figure 1. For more on convex sets, see [24].

To find the set $\hat{\mathcal{C}}$, we use a *Lyapunov performance certificate*. The details depend on the particular performance measure or measures, but the common form is as follows. We express the set of acceptable controllers using linear matrix inequalities:

$$\theta \in \hat{\mathcal{C}} \iff \exists \nu \; L(\theta, \nu) \succeq 0,$$

where L is a function that is bi-affine, *i.e.*, affine in θ for fixed ν, and affine in ν for fixed θ. The symbol $\succeq$ refers to matrix inequality, between symmetric matrices, so the condition above is that $L(\theta, \nu)$ is positive semidefinite. The variable ν represents the coefficients in the Lyapunov function used to certify performance. For more on representing control system specifications via LMIs, see, *e.g.*, [25], [26], [27].

For a given $\theta \in \mathcal{C}$, we compute a value of ν such that $L(\theta, \nu) \succeq 0$. We then fix ν, and take

$$\hat{\mathcal{C}} = \{\theta \mid L(\theta, \nu) \succeq 0\}. \tag{3}$$

This set depends on the particular choice of ν; but in all cases, it is convex, indeed, it is described by an LMI in θ. For a given $\theta \in \mathcal{C}$, ν can be typically chosen to maximize the minimum eigenvalue of $L(\theta,\nu)$ or to maximize the determinant of $L(\theta,\nu)$. Both of these problems are convex: maximizing the minimum eigenvalue can be reduced to solving a semidefinite program (SDP) and maximizing the determinant can be reduced to solving a MAXDET problem.

To find l or u in (2), we need to minimize or maximize a scalar variable over an LMI. This can be reduced to an eigenvalue computation [24, Exer. 4.38], and can be carried out efficiently. Since $L(\theta,\nu)$ is bi-affine in θ and ν, it can be expressed as

$$L(\theta,\nu) = L_0 + \sum_{i=1}^{N} \theta_i L_i,$$

where we have obscured the fact that the matrices L_0 and L_i depend on ν. When $\tilde{\theta} = (\theta_1,\ldots,\theta_{i-1},z,\theta_{i+1},\ldots,\theta_N)$, we have

$$L(\tilde{\theta},\nu) = L(\theta,\nu) + (z-\theta_i)L_i.$$

Assuming that $L(\theta,\nu) \succ 0$, the range $[l,u]$ of θ_i consists of the values of z for which $L(\tilde{\theta},\nu) \succeq 0$. It can be shown that

$$l = \theta_i - \min\{1/\lambda_i \mid \lambda_i > 0\}, \quad (4)$$

$$u = \theta_i - \max\{1/\lambda_i \mid \lambda_i < 0\}, \quad (5)$$

where λ_i are the eigenvalues of $L(\theta,\nu)^{-1/2} L_i L(\theta,\nu)^{-1/2}$.

In the examples we will consider, the LMIs that arise have an even more specific form,

$$L(\theta,\nu) = \begin{bmatrix} I & Z^T \\ Z & I \end{bmatrix} \succeq 0,$$

where

$$Z = Z_0 + \sum_{i=1}^{N} \theta_i v_i w_i^T.$$

Here Z_0 is a matrix, and v_i and w_i are vectors, with dimensions and data that depend on the particular problem. In the general notation used above, this corresponds to

$$L_0 = \begin{bmatrix} I & Z_0^T \\ Z_0 & I \end{bmatrix}, \qquad L_i = \begin{bmatrix} 0 & w_i v_i^T \\ v_i w_i^T & 0 \end{bmatrix},$$

for $i = 1,\ldots,N$. We can then express $\hat{\mathcal{C}}$ as

$$\hat{\mathcal{C}} = \{\theta \mid \|Z\| \leq 1\},$$

where $\|\cdot\|$ denotes the spectral norm (maximum singular value).

We now give the details of how to find the range of the coefficient θ_i in the convex set $\hat{\mathcal{C}}$, *i.e.*, how to compute l and u in (2).

Note that the rank of L_i is exactly 2. Assuming that $L(\theta,\nu) \succ 0$ and v_i and w_i are both nonzero, the matrix $L(\theta,\nu)^{-1/2} L_i L(\theta,\nu)^{-1/2}$ has one positive eigenvalue $\lambda_{\max}$, $(2n+m-2)$ zero eigenvalues and one negative eigenvalue $\lambda_{\min}$. We now proceed to find more explicit expressions for $\lambda_{\min}$ and $\lambda_{\max}$. Let $Z = U\Sigma V^T$ be the full singular value decomposition of Z where U and V are orthogonal matrices and Σ has the same dimensions as Z. If $m = n$, Σ is diagonal. If $m \geq n$, we have

$$\Sigma = \begin{bmatrix} \mathbf{diag}(\sigma_1,\ldots,\sigma_n) \\ 0 \end{bmatrix}.$$

Otherwise, we have

$$\Sigma = \begin{bmatrix} \mathbf{diag}(\sigma_1,\ldots,\sigma_m) & 0 \end{bmatrix}.$$

Let

$$x = U^T v_i, \qquad y = V^T w_i,$$

$$E = \begin{bmatrix} I & \Sigma^T \\ \Sigma & I \end{bmatrix}, \qquad F = \begin{bmatrix} 0 & yx^T \\ xy^T & 0 \end{bmatrix}.$$

Using a block Cholesky factorization, we can write $E = CC^T$, where

$$C = \begin{bmatrix} I & 0 \\ \Sigma & (I - \Sigma\Sigma^T)^{1/2} \end{bmatrix}.$$

Note that

$$C^{-1} = \begin{bmatrix} I & 0 \\ -A\Sigma & A \end{bmatrix},$$

where $A = (I - \Sigma\Sigma^T)^{-1/2}$.

It is easy to show that $\lambda_{\min}$ and $\lambda_{\max}$ are, respectively, the minimum and maximum eigenvalues of $C^{-1}FC^{-T}$. Since

$$F = \begin{bmatrix} 0 & y \\ x & 0 \end{bmatrix} \begin{bmatrix} y^T & 0 \\ 0 & x^T \end{bmatrix},$$

and since nonzero eigenvalues of MN and NM are identical for any two matrices $M \in \mathbf{R}^{n\times m}$ and $N \in \mathbf{R}^{m\times n}$, $\lambda_{\min}$ and $\lambda_{\max}$ are the eigenvalues of

$$\begin{bmatrix} y^T & 0 \\ 0 & x^T \end{bmatrix} C^{-T} C^{-1} \begin{bmatrix} 0 & y \\ x & 0 \end{bmatrix}.$$

These can be found analytically as

$$\lambda_{\min} = -\alpha - \sqrt{3\alpha^2 + \beta y^T y + \beta\gamma}, \quad (6)$$

$$\lambda_{\max} = -\alpha + \sqrt{3\alpha^2 + \beta y^T y + \beta\gamma}. \quad (7)$$

The terms α, β and γ can be computed more easily as

$$\alpha = x^T A^2 \Sigma y = \sum_{i=1}^{\min\{m,n\}} \frac{x_i y_i \sigma_i}{1-\sigma_i^2}, \quad (8)$$

$$\beta = x^T A^2 x = \sum_{i=1}^{m} \frac{x_i^2}{1-\sigma_i^2}, \quad (9)$$

$$\gamma = y^T \Sigma^T A^2 \Sigma y = \sum_{j=1}^{n} \frac{y_j^2 \sigma_j^2}{1-\sigma_j^2}. \quad (10)$$

In summary, to find l and u, we start by computing the SVD of the Z and setting $x = U^T v_i$, $y = V^T w_i$. We proceed then to compute the 3 terms in (8), (9), (10) and compute $\lambda_{\min}$ and $\lambda_{\max}$ from (6) and (7). Finally, l and u are found from (4) and (5):

$$l = \theta_i - 1/\lambda_{\max}, \quad u = \theta_i - 1/\lambda_{\min}. \quad (11)$$

III. State feedback controller with LQR cost

We will demonstrate how to apply the algorithm to a specific problem class where the plant is given by

$$x(t+1) = Ax(t) + Bu(t), \quad x(0) = x_0, \tag{12}$$

and is controlled by a state feedback gain controller given by

$$u(t) = Kx(t), \tag{13}$$

where $A \in \mathbf{R}^{n\times n}$, $B \in \mathbf{R}^{n\times m}$, $C \in \mathbf{R}^{k\times n}$, $K \in \mathbf{R}^{m\times n}$ is the feedback gain matrix, $x(t) \in \mathbf{R}^n$ is the state of the system and $u(t) \in \mathbf{R}^m$ is the input to the system. The design variables are the entries of the matrix K.

A. Admissible controllers

Given $Q \in \mathbf{R}^{n\times n}$ positive semidefinite and $R \in \mathbf{R}^{m\times m}$ positive definite, the performance measure is given by the LQR cost

$$\begin{aligned} J(K) &= \mathbf{E}\left[\sum_{t=0}^{\infty} x(t)^T Q x(t) + u(t)^T R u(t)\right] \\ &= \mathbf{E}\left[\sum_{t=0}^{\infty} x(t)^T (Q + K^T R K) x(t)\right], \end{aligned} \tag{14}$$

where the expectation is taken over $x_0 \sim \mathcal{N}(0, \Sigma)$. If $A + BK$ is unstable $J(K)$ is infinite. Otherwise, let P be the (unique) solution to the Lyapunov equation

$$(A+BK)^T P (A+BK) - P + Q + K^T R K = 0. \tag{15}$$

The cost in (14) can be expressed as $J(K) = \mathbf{Tr}(\Sigma P)$. This holds because

$$\begin{aligned} J(K) &= \mathbf{E}\left[\sum_{t=0}^{\infty} x(t)^T P x(t) - x(t+1)^T P x(t+1)\right] \\ &= \mathbf{E}\left[x_0^T P x_0\right] \\ &= \mathbf{Tr}(\mathbf{E}\left[x_0 x_0^T\right] P) \\ &= \mathbf{Tr}(\Sigma P). \end{aligned}$$

The nominal design K^{nom} is chosen to be the optimal state feedback controller, *i.e.*, the one that minimizes the LQR cost J. It can be found as follows,

$$K^{\mathrm{nom}} = -(R + B^T P^{\mathrm{nom}} B)^{-1} B^T P^{\mathrm{nom}} A,$$

where P^{nom} is the solution of the discrete-time algebraic Riccati equation

$$\begin{aligned} P^{\mathrm{nom}} = Q &+ A^T P^{\mathrm{nom}} A \\ &- A^T P^{\mathrm{nom}} A (R + B^T P^{\mathrm{nom}} B)^{-1} B^T P^{\mathrm{nom}} A. \end{aligned}$$

When $K = K^{\mathrm{nom}}$, P^{nom} is also the solution of the Lyapunov equation (15). The LQR cost associated with the optimal controller is $J^{\mathrm{nom}} = \mathbf{Tr}(\Sigma P^{\mathrm{nom}})$.

We define the set of admissible controller design as

$$\mathcal{C} = \{K \mid J(K) \le (1+\epsilon) J^{\mathrm{nom}}\},$$

where ϵ is a given positive number. This means that a controller design is admissible if and only if it is ϵ-suboptimal.

We choose the Lyapunov performance certificate L to be the block diagonal matrix with the following blocks on the diagonal $P - (A+BK)^T P (A+BK) - Q - K^T R K$, $(1+\epsilon) J^{\mathrm{nom}} - \mathbf{Tr}(\Sigma P)$ and P. Here K and P correspond, respectively, to θ and ν introduced in §II-B. The condition that $L(K,P) \succeq 0$ is equivalent to

$$\begin{aligned} P - (A+BK)^T P (A+BK) &\succeq Q + K^T R K, \\ \mathbf{Tr}(\Sigma P) &\le (1+\epsilon) J^{\mathrm{nom}}, \quad (16) \\ P &\succeq 0. \quad (17) \end{aligned}$$

Since (16) and (17) do not depend on K, and for a particular choice P, (3) becomes

$$\hat{\mathcal{C}} = \{K \mid (A+BK)^T P (A+BK) - P + Q + K^T R K \preceq 0\}. \tag{18}$$

Given $K \in \mathcal{C}$, any matrix P that satisfies $L(K,P) \succeq 0$ is a valid choice. We take P to be the solution of the following optimization problem

$$\begin{array}{ll} \text{maximize} & \lambda_{\min}(L(K,P)) \\ \text{subject to} & L(K,P) \succeq 0. \end{array}$$

Here $\lambda_{\min}(L(K,P))$ is the minimum eigenvalue of $L(K,P)$ and P is the variable we are optimizing over. Recall that K is fixed.

We will now show that $\hat{\mathcal{C}} \subseteq \mathcal{C}$. Let $K \in \hat{\mathcal{C}}$. Consider the Lyapunov function $V : \mathbf{R}^n \to \mathbf{R}$ defined as $V(z) = z^T P z$. For any $T > 0$,

$$\begin{aligned} \Delta V(T) &= V(x(T)) - V(x(0)) \\ &= \sum_{t=0}^{T} V(x(t+1)) - V(x(t)) \\ &= \sum_{t=0}^{T} x(t+1)^T P x(t+1) - x(t)^T P x(t) \\ &\le -\sum_{t=0}^{T} x(t) \left(Q + K^T R K\right) x(t). \end{aligned}$$

Therefore,

$$\begin{aligned} \sum_{t=0}^{T} x(t) \left(Q + K^T R K\right) x(t) &\le V(x(0)) - V(x(T)) \\ &\le V(x(0)), \end{aligned}$$

where the last inequality follows because $V(x(T)) \ge 0$ from (17). Letting T tend to infinity and taking expectation over x_0, we obtain $J(K) \le \mathbf{Tr}(\Sigma P)$. It follows from (16) that

$$J(K) \le (1+\epsilon) J^{\mathrm{nom}}.$$

B. Coefficient range calculation

Let (l, u) be the range of coefficient K_{ij}. Given (18), problem (2) becomes

$$\begin{aligned} l = \min\{K_{ij} \mid (A+BK)^T P (A+BK) - P \\ + Q + K^T R K \preceq 0\}, \\ u = \max\{K_{ij} \mid (A+BK)^T P (A+BK) - P \\ + Q + K^T R K \preceq 0\}. \end{aligned}$$

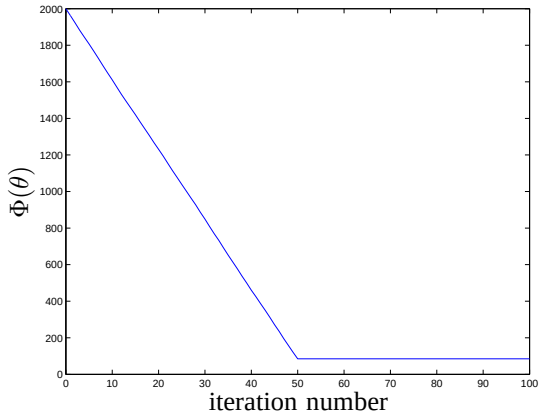

Fig. 2. Total number of bits required to express θ versus iteration number.

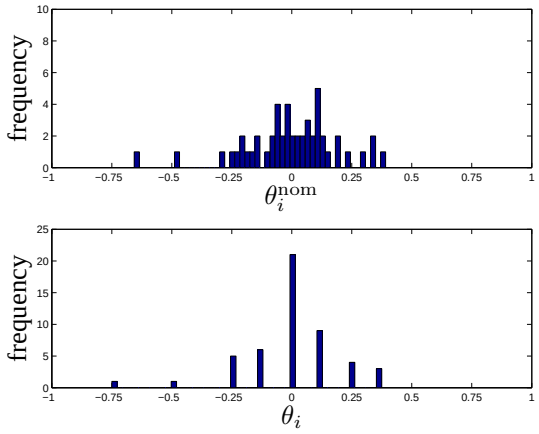

Fig. 3. Top: histogram showing the distribution of the coefficients in the nominal design. Bottom: histogram showing the distribution of the coefficients in the best design in 100 random runs.

The inequality in (18) is equivalent to

$$\left\| \begin{bmatrix} P^{1/2}(A+BK) \\ R^{1/2}K \end{bmatrix} (P-Q)^{-1/2} \right\| \leq 1. \qquad (19)$$

The method outlined in §II-B can be used to compute l and u by taking

$$Z = \begin{bmatrix} P^{1/2}(A+BK) \\ R^{1/2}K \end{bmatrix} (P-Q)^{-1/2},$$
$$v = \begin{bmatrix} P^{1/2}B \\ R^{1/2} \end{bmatrix} e_i, \quad w = (P-Q)^{-1/2} e_j,$$

where e_i and e_j are, respectively, the ith unit vector in $\mathbf{R}^m$ and jth unit vector in $\mathbf{R}^n$ and $K \in \hat{\mathcal{C}}$ is the current admissible controller design.

C. Numerical instance

Our example has dimensions $n = 10$ and $m = 5$. We generated the plant randomly, as $A = I + 0.1X/\sqrt{n}$, where X_{ij} are independent identically distributed (IID) $\mathcal{N}(0,1)$. We generated the matrix $B \in \mathbf{R}^{10\times 5}$, with B_{ij} IID $\mathcal{N}(0,1)$. We take $\Sigma = I$, $Q = I$ and $R = I$.

The complexity measures $\phi_i(z)$ are chosen to be $\phi_{\rm frac-bits}$. The fractional part of each entry of $K^{\rm nom}$ is expressed with 40 bits, requiring a total of 2000 bits to express $K^{\rm nom}$, *i.e.*, $\Phi(\theta^{\rm nom}) = 2000$ bits. We take $\epsilon = 15\%$, *i.e.*, admissible feedback controllers are those that are up to 15%-suboptimal.

The progress of the complexity $\Phi(\theta)$ during a sample run of the algorithm is shown in figure 2. In this sample run the algorithm converges to a complexity of 85 bits in one pass over the variables. During the run of the algorithm the cost J is approximately constant and equal to its maximum allowed value $1.15J^{\rm nom}$.

The best design after 10 random runs of the algorithm achieves a complexity of $\Phi(\theta) = 81$ bits, with a cost of $J(\theta) = 1.1494J(\theta^{\rm nom})$. The best design found after 100 random runs of the algorithm achieves a complexity of $\Phi(\theta) = 75$ bits and $J(\theta) = 1.1495J(\theta^{\rm nom})$.

This best design gives very aggressive coefficient truncation, with only 1.5 bits per coefficient. This is illustrated in figure 3, which shows the distribution of the (50) coefficients of the nominal design and the coefficients of the best design.

IV. Dynamic controller with decay rate specification

We demonstrate how to apply the algorithm to the problem class where the plant is given by

$$x_p(t+1) = A_p x_p(t) + B_p u(t), \quad y(t) = C_p x_p(t), \quad (20)$$

and is controlled by a dynamic controller given by

$$x_c(t+1) = A_c x_c(t) + B_c y(t), \quad u(t) = C_c x_c(t). \quad (21)$$

where $x_p(t) \in \mathbf{R}^{n_p}$, $u(t) \in \mathbf{R}^{m_c}$, $y(t) \in \mathbf{R}^{m_p}$, $A_p \in \mathbf{R}^{n_p\times n_p}$, $B_p \in \mathbf{R}^{n_p\times m_c}$, $C_p \in \mathbf{R}^{m_p\times n_p}$, $x_c(t) \in \mathbf{R}^{n_c}$, $A_c \in \mathbf{R}^{n_c\times n_c}$, $B_c \in \mathbf{R}^{n_c\times m_p}$ and $C_c \in \mathbf{R}^{m_c\times n_c}$.

The closed-loop system is given by $x(t+1) = Ax(t)$ where

$$x(t) = \begin{bmatrix} x_p(t) \\ x_c(t) \end{bmatrix}, \quad A = \begin{bmatrix} A_p & B_pC_c \\ B_cC_p & A_c \end{bmatrix}. \qquad (22)$$

The design variables are the entries of the controller matrices A_c, B_c and C_c.

A. Admissible controllers

A controller (A_c, B_c, C_c) is admissible if the decay rate of the closed-loop system is less than a given rate α, where $0 \leq \alpha \leq 1$. The decay rate is given by $\rho(A)$, where A is the matrix specified in (22).

The performance measure is chosen to be the decay rate of the closed-loop system, *i.e.*, $J(A_c, B_c, C_c) = \rho(A)$.

We are given a nominal controller design $(A_c^{\rm nom}, B_c^{\rm nom}, C_c^{\rm nom})$ such that

$$J(A_c^{\rm nom}, B_c^{\rm nom}, C_c^{\rm nom}) = \rho.$$

We define the set of admissible controller designs as

$$\mathcal{C} = \{(A_c, B_c, C_c) \mid J(A_c, B_c, C_c) \leq \alpha\},$$

where $\alpha = (1+\epsilon)\rho$ and ϵ is a given positive number.

We choose the Lyapunov performance certificate L to be

$$L(A_c, B_c, C_c, P) = \left[\begin{array}{cc} \alpha^2 P - A^T P A & 0 \\ 0 & P \end{array} \right],$$

where A is the matrix defined in (22). Here (A_c, B_c, C_c) and P correspond, respectively, to θ and ν introduced in §II-B. The condition that $L(A_c, B_c, C_c, P) \succeq 0$ is equivalent to

$$\begin{array}{rcl} A^T P A & \preceq & \alpha^2 P \\ P & \succeq & 0. \end{array} \tag{23}$$

Since (23) doesn't depend on (A_c, B_c, C_c), for a fixed choice of P, (3) becomes

$$\hat{\mathcal{C}} = \{(A_c, B_c, C_c) \mid A^T P A \leq \alpha^2 P\}. \tag{24}$$

Any matrix P that satisfies $L(A_c, B_c, C_c, P) \succeq 0$ for $(A_c, B_c, C_c) \in \mathcal{C}$ is a valid choice. We take P to be the solution of the following optimization problem

$$\begin{array}{ll} \text{maximize} & \lambda_{\min}(L(A_c, B_c, C_c, P)) \\ \text{subject to} & L(A_c, B_c, C_c, P) \succeq 0 \\ & \mathbf{Tr}(P) = 1. \end{array}$$

Here $\lambda_{\min}(L(A_c, B_c, C_c, P))$ is the minimum eigenvalue of $L(A_c, B_c, C_c, P)$, and P is the variable we are maximizing over. Recall that A_c, B_c and C_c are fixed. The constraint $\mathbf{Tr}(P) = 1$ is added because $L(A_c, B_c, C_c, P)$ is homogeneous in P.

We will now show that $\hat{\mathcal{C}} \subseteq \mathcal{C}$. Let $(A_c, B_c, C_c) \in \hat{\mathcal{C}}$. Consider the Lyapunov function $V : \mathbf{R}^{n_p+n_c} \to \mathbf{R}$ defined as $V(z) = z^T P z$. Since $A^T P A \leq \alpha^2 P$ then for all $t \geq 0$

$$\begin{array}{rcl} x(t)^T A^T P A x(t) & \leq & \alpha^2 x(t)^T P x(t) \\ x(t+1)^T P x(t+1) & \leq & \alpha^2 x(t)^T P x(t) \\ V(x(t+1)) & \leq & \alpha^2 V(x(t)). \end{array}$$

This means that for all $t \geq 0$, $V(x(t)) \leq \alpha^{2t} V(x(0))$ and

$$\begin{array}{rcl} \lambda_{\min}(P)\|x(t)\|^2 & \leq & x(t)^T P x(t) \\ & \leq & \alpha^{2t} x(0)^T P x(0) \\ & \leq & \alpha^{2t} \lambda_{\max}(P)\|x(0)\|^2, \end{array}$$

then $\|x(t)\| \leq \sqrt{\kappa(P)}\alpha^t \|x(0)\|$, where $\kappa(P)$ is the condition number of P. The decay rate of the system is then less than α, as required.

B. Coefficient range calculation

Let (l, u) be the range of coefficient $(A_c)_{ij}$. Given (24), problem (2) becomes

$$\begin{array}{rcl} l & = & \min\{(A_c)_{ij} \mid A^T P A \leq \alpha^2 P\}, \\ u & = & \max\{(A_c)_{ij} \mid A^T P A \leq \alpha^2 P\}. \end{array}$$

The inequality in (24) is equivalent to

$$\left\| P^{1/2} A P^{-1/2} \right\| \leq \alpha. \tag{25}$$

The method outlined in §II-B can be used to compute l and u by taking

$$Z = (1/\alpha) P^{1/2} A P^{-1/2},$$

and

$$v = (1/\alpha) P^{1/2} \left[\begin{array}{c} 0 \\ e_i \end{array} \right], \quad w = P^{-1/2} \left[\begin{array}{c} 0 \\ e_j \end{array} \right],$$

where e_i and e_j are, respectively, the ith and jth unit vectors in $\mathbf{R}^{n_c}$ and A is the closed-loop matrix associated with $(A_c, B_c, C_c) \in \hat{\mathcal{C}}$.

The same method can be used to find the ranges of coefficients in B_c and C_c. The same formulas can be used but with sightly modified definitions for v and w.

To find the range of coefficient $(B_c)_{ij}$, use the same definitions for Z and v but let

$$w = P^{-1/2} \left[\begin{array}{c} C_p^T e^j \\ 0 \end{array} \right],$$

where e_j is the jth unit vector in $\mathbf{R}^{m_c}$.

To find the range of coefficient $(C_c)_{ij}$, use the same definitions for Z and w but let

$$v = (1/\alpha) P^{1/2} \left[\begin{array}{c} B_p e^i \\ 0 \end{array} \right],$$

where e_i is the ith unit vector in $\mathbf{R}^{m_c}$.

C. Numerical instance

We test the proposed method in the case where the plant is given by

$$x_p(t+1) = A_p x_p(t) + B_p u(t) + w(t), \quad y(t) = C_p x_p(t) + v(t),$$

where $w(t) \sim \mathcal{N}(0, I)$ is the input noise and $v(t) \sim \mathcal{N}(0, I)$ is the measurement noise. The plant is controlled by an LQG controller with $Q = I$, $R = I$. The matrices describing the controller are

$$A_c = A_p + B_p K - L C_p, \quad B_c = L, \quad C_c = K,$$

where

$$\begin{array}{rcl} K & = & -(B_p^T P_1 B_p + R)^{-1} B_p^T P_1 A_p, \\ L & = & A_p P_2 C_p^T (C_p P_2 C_p^T + V)^{-1}. \end{array}$$

P_1 and P_2 are the unique positive semidefinite solutions to the discrete-time algebraic Riccati equations

$$\begin{array}{rcl} P_1 & = & A_p^T P_1 A_p + Q \\ & & -A_p^T P_1 B_p (R + B_p^T P_1 B_p)^{-1} B_p^T P_1 A_p, \\ P_2 & = & A_p P_2 A_p^T + W \\ & & -A_p P_2 C_p^T (C_p P_2 C_p^T + V)^{-1} C_p P_2 A_p^T. \end{array}$$

Our example has dimensions $n_p = 5$, $m_c = 2$ and $m_p = 2$. The plant matrix A_p is randomly generated using the same method used to generate A in §3.3. The entries of B_p and C_p are IID $\mathcal{N}(0, 1)$. The matrices A_c^{nom}, B_c^{nom} and C_c^{nom} are then computed using the formulas presented above.

The complexity measures $\phi_i(z)$ are chosen to be $\phi_{\text{frac-bits}}$. The fractional part of each entry of A_c^{nom}, B_c^{nom} and C_c^{nom} is expressed with 40 bits, requiring a total of 1800 bits, *i.e.*, $\Phi(\theta^{\text{nom}}) = 1800$ bits. We run the algorithm with $\epsilon = 5\%$.

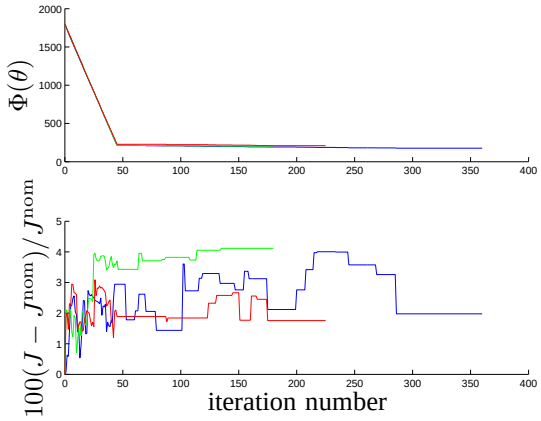

Fig. 4. Top: total number of bits required to express θ versus iteration number. Bottom: percentage deterioration in performance versus iteration number.

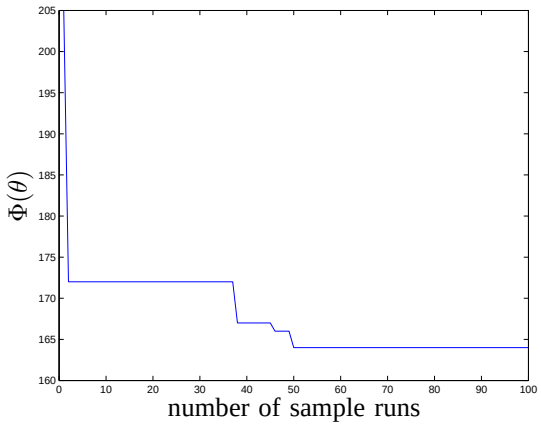

Fig. 5. Best design complexity versus number of sample runs of the algorithm

The progress of the complexity $\Phi(\theta)$ and percentage deterioration in performance $100(J - J^{\mathrm{nom}})/J^{\mathrm{nom}}$ during 3 sample runs of the algorithm are shown in figure 4.

The best design after 10 random runs of the algorithm achieves a complexity of $\Phi(\theta) = 171$ bits with a cost of $J(\theta) = 1.0246J(\theta^{\mathrm{nom}})$. The best design after 100 random runs of the algorithm achieves a complexity of $\Phi(\theta) = 164$ bits and $J(\theta) = 1.0362J(\theta^{\mathrm{nom}})$. Figure 5 shows the best available design complexity versus the number of sample runs of the algorithm.

REFERENCES

[1] J. Whidborne, J. Mckernan, and D. Gu, "Kolmogorov-Chaitin complexity of linear digital controllers implemented using fixed-point arithmetic," in *Proceeding of the 8th Control, Automation, Robotics and Vision conference*, vol. 3, 2004, pp. 1597–1592.

[2] E. Avenhaus, "On the design of digital filters with coefficients of limited word length," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 3, pp. 206–212, 1972.

[3] F. Brglez, "Digital filter design with short word-length coefficients," *IEEE Transactions on Circuits and Systems*, vol. 25, no. 12, pp. 1044–1050, 1978.

[4] A. Gray and J. Markel, "Quantization and bit allocation in speech processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 6, pp. 459–473, 1976.

[5] R. Rink and H. Chong, "Performance of state regulator systems with floating point computation," *IEEE Transactions on Automatic Control*, vol. 14, pp. 411–412, 1979.

[6] Y. Lim and Y. Yu, "A successive reoptimization approach for the design of discrete coefficient perfect reconstruction lattice filter bank," in *IEEE International Symposium on Cicuits and Systems*, vol. 2, 2000, pp. 69–72.

[7] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 7, pp. 1044–1047, 1989.

[8] J. Lee, C. Chen, and Y. Lim, "Design of discrete coefficient FIR digital filters with arbitrary amplitude and phase response," *IEEE Transactions on Circuits and Systems - II: Analog and Digital Signal Processing*, vol. 40, no. 7, pp. 444–448, 1993.

[9] Y. Lim and Y. Yu, "A width-recursive depth-first tree search approach for the design of discrete coefficient perfect reconstruction lattice filter bank," *IEEE Transactions on Circuits and Systems II*, vol. 50, pp. 257–266, 2003.

[10] P. Frossard, P. Vandergheynst, R. Figueras, and M. Kunt, "A posteriori quantization of progressive matching pursuit streams," *IEEE Transactions on Signal Processing*, vol. 52, no. 2, pp. 525–535, 2004.

[11] Y. Lim and S. Parker, "Discrete coefficient FIR digital filter design based upon an LMS criteria," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 30, no. 10, pp. 723–739, 1983.

[12] S. Chen and J. Wu, "The determination of optimal finite precision controller realizations using a global optimization strategy: a pole-sensitivity approach," in *Digital controller Implementation and Fragility: A Modern Perpective*, R. Istepanian and J. Whidborne, Eds. New York: Springer-Verlag, 2001, ch. 6, pp. 87–104.

[13] N. Benvenuto, M. Marchesi, G. Orlandi, F. Piazza, and A. Uncini, "Finite wordlength digital filter design using an annealing algorithm," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1989, pp. 861–864.

[14] R. Istepanian and J. Whidborne, "Multi-objective design of finite word-length controller structures," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, 1999, pp. 61–68.

[15] P. Gentili, F. Piazza, and A. Uncini, "Efficient genetic algorithm design for power-of-two FIR filters," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1995, pp. 1268–1271.

[16] S. Traferro, F. Capparelli, F. Piazza, and A. Uncini, "Efficient allocation of power-of-two terms in FIR digital filter design using tabu search," in *International Symposium on Circuits and Systems (ISCAS)*, 1999, pp. III– 411–414.

[17] G. Li and M. Gevers, "Optimal finite precision implementation of a state-estimate feedback controller," *IEEE Transactions on Circuits and Systems*, vol. 37, no. 12, pp. 1487–1498, 1990.

[18] B. Wah, Y. Shang, and Z. Wu, "Discrete Lagrangian methods for optimizing the design of multiplierless QMF banks," *IEEE Transactions on Circuits and Systems II*, vol. 46, no. 9, pp. 1179–1191, 1999.

[19] D. Kodek, "Design of optimal finite wordlength FIR digital filters using integer programming techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 3, pp. 304–308, 1980.

[20] Y. Lim and S. Parker, "FIR filter design over a discrete powers-of-two coefficient space," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 31, no. 3, pp. 583–591, 1983.

[21] S. Barua, K. Kotteri, A. Bell, and J. Carletta, "Optimal quantized lifting coefficients for the 9/7 wavelet," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, 2004, pp. V– 193–196.

[22] K. Liu, R. Skelton, and K. Grigoriadis, "Optimal controllers for finite wordlength implementation," *IEEE Transactions on Automatic Control*, vol. 37, no. 9, pp. 1294–1304, 1992.

[23] A. Molchanov and P. Bauer, "Robust stability of digital feedback control systems with floating point arithmetic," in *Proceedings of the 34th Conference on Decision and Control*, 1995, pp. 4251–4258.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[25] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in Systems and Control Theory*. Philadelphia: SIAM books, 1994.

[26] S. Boyd and C. Barratt, *Linear Controller Design: Limits of Performance*. Prentice-Hall, Inc., 1991.

[27] G. Dullerud and F. Paganini, *A Course in Robust Control Theory: a Convex Approach*. New York: Springer-Verlag, 2000.