2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Linear Computational Complexity Design of Constrained Optimal ILC

Aleksandar Haber, Rufus Fraanje and Michel Verhaegen

Abstract— **In this paper we present a linear computational complexity framework for design and implementation of (constrained) lifted Iterative Learning Control (ILC) systems with quadratic cost. The problem of designing constrained lifted ILC with quadratic cost is formulated as a convex optimization problem. We solve this problem using the primal-dual interior point method. High computational complexity of the primal-dual method, which render this method computationally infeasible for high dimensional lifted ILC systems, is significantly decreased by exploiting the sequentially semi-separable (SSS) structure of lifted system matrices. More precisely, $\mathcal{O}(N^3)$ computational cost of one iteration of the primal-dual method is reduced to $\mathcal{O}(N)$, where N characterizes the size of the lifted system matrices. Furthermore, by exploiting the SSS structure the large lifted system matrices can be efficiently stored in computer memory. We also show that SSS structure can be exploited to efficiently implement analytical solution of the unconstrained lifted ILC problem with quadratic cost and for calculation of the norm and stability radius of ILC system.**

Index Terms— **Learning control; Efficient algorithms; Constrained optimization; Primal-Dual methods.**

I. INTRODUCTION

Iterative Learning Control (ILC) has proven to be an effective method for achieving a high tracking performance of systems that have to execute the same task a number of times. Each execution of the task is referred to as a trial, and ILC strategy derives a control action for the next trial by updating the control action from the previous trial(s) with the signals that are derived on the basis of measured data from the previous trial(s). Detailed and recent overviews of the theoretical foundations and fields of application of ILC, with an extensive bibliography are available in [1], [5]. The system controlled by ILC evolves during the trial length, a domain that we refer to as the local time domain, and from one trial to another, a domain that we refer to as the trial domain. In lifted system representation [16] or Super-Vector ILC (SVILC) representation [14], this 2D system is reformulated as 1D system that evolves only in the trial domain.

In [12], [9], [4], [5] lifted ILC law has been obtained as a solution of the convex unconstrained quadratic optimization problem. This type of ILC law is referred to as the ILC with quadratic cost (Q-ILC) or as the norm-optimal ILC. In [13] actuator's saturation constrains have been incorporated in the design of Q-ILC.

This research is supported by the Dutch Ministry of Economic Affairs and the Provinces of Noord-Brabant and Limburg in the frame of the "Pieken in de Delta" program.

A. Haber and M. Verhaegen are with Delft Center for Systems and Control, Delft University of Technology, Delft, 2628 CD, The Netherlands, (e-mail: a.haber@tudelft.nl; p.r.fraanje@tudelft.nl m.verhaegen@tudelft.nl).

In the lifted ILC framework, dimensions of the system matrices are proportional to the number of the sampling steps in one trial (trial length). This implies that for large trial lengths or for a relatively high sampling frequencies with respect to the trial length, the lifted system matrices (system matrices in the trial domain) have extremely large dimensions. This is the case in robotic applications where the lifted system matrices can have several million elements [10]. Consequently, a straightforward application of lifted ILC design methods proposed in [12][9][4][5][13] may turn out to be computationally infeasible.

Analysis of the lifted ILC laws consists of stability check and convergence rate evaluation. The stability and the convergence rate are determined by the spectral properties (the spectral radius and the spectral norm) of the lifted system matrices of the ILC system [1], [5]. In the case of large trial lengths, the high computational complexity of algorithms used for determining the spectral properties of the lifted matrices [6] will hinder the analysis of the ILC system.

Beside the computational complexity, another difficulty in the design of lifted ILC for large trial lengths, is an increased demand for memory locations [2]. The necessary memory, that is needed to store lifted system matrices, is a quadratic function of N.

The problem of high computational complexity of the lifted ILC and Repetitive Control has been recognized by several researchers. The Sequentially-Semi Separable (SSS) structure of the lifted system matrices has been exploited in [17] to derive LQG (H_2) repetitive controller with $\mathcal{O}(N)$ complexity. In [11], a computationally efficient method for implementing unconstrained Q-ILC law is presented. This method is restricted to block-diagonal selection of weighting matrices of the Q-ILC cost function. In [2], a computationally efficient method for determining monotonicity and convergence rate of ILC systems with large number of samples in a trial has been presented. In order to apply the methodology presented in [2], the lifted learning matrices must be precomputed in advance ($\mathcal{O}(N^3)$ complexity) or they must be described as filters in the local frequency domain associated with the local-time domain. Due to this, the method presented in [2] is restricted.

In this paper we present a linear computational complexity framework for design and implementation of (constrained) lifted ILC systems with quadratic cost. The problem of designing constrained lifted ILC with quadratic cost is formulated as a convex optimization problem. We solve this problem using the primal-dual interior point method. High computational complexity of the primal-dual method, that render this method computationally infeasible for high

dimensional lifted ILC systems, is significantly decreased by exploiting the sequentially semi-separable (SSS) structure of lifted system matrices. More precisely, $\mathcal{O}(N^3)$ computational cost of one iteration of the primal-dual method is reduced to $\mathcal{O}(N)$. Furthermore, by exploiting the SSS structure the large lifted system matrices can be efficiently stored in computer memory. We also show that SSS structure can be exploited to efficiently implement analytical solution of the unconstrained lifted ILC problem with quadratic cost and for calculation of the norm and stability radius of ILC system.

This work is organized as follows: In Section 2 we present problem formulation; in Section 3 we revise the main properties of the SSS structure that are relevant for this work; in Section 4 we present a linear computational complexity framework for lifted ILC analysis and design. Finally, simulation experiments that confirm the linear computational complexity are presented in Section 5 and conclusions are presented in Section 6.

II. Problem statement

The ILC problem formulation presented in this paper is kept as simple as possible due to simplicity reasons. The results presented in this paper can be generalized for much more richer and complex ILC problem formulations (for example ILC problem formulation that includes a local-time domain feedback). We consider the local-time domain Linear Time Varying (LTV) system:

$$\mathbf{x}_k(t+1) = A(t)\mathbf{x}_k(t) + B(t)\mathbf{u}_k(t) + F(t)\mathbf{d}_k(t) \quad (1)$$

$$\mathbf{y}_k(t) = C(t)\mathbf{x}_k(t) + D(t)\mathbf{u}_k(t) \quad (2)$$

where t is the local-time, k is the trial index, $\mathbf{x}_k(t) \in \mathbb{R}^{n_x}$ is the state, $\mathbf{y}_k(t) \in \mathbb{R}^{n_y}$ is the measured output, $\mathbf{u}_k(t) \in \mathbb{R}^{n_u}$ is the control input and $\mathbf{d}_k(t) \in \mathbb{R}^{n_d}$ is the process disturbance. All the matrix dimensions are in accordance with the corresponding vector dimensions. For the sake of simplicity we assume $n_y = n_u = 1$. The state transition matrix of the system (1)-(2) is defined as:

$$\Phi_{(N,N_0)} = \begin{cases} A(N-1)A(N-2)\dots A(N_0) & \text{if } N > N_0 \\ I & \text{if } N = N_0 \end{cases}$$

For the trial length of $N+1$ we have

$$\underline{\mathbf{y}}_k = \underline{C}\mathbf{x}_k(0) + \underline{D}\underline{\mathbf{u}}_k + \underline{H}\underline{\mathbf{d}}_k \quad (3)$$

where

$$\underline{D} = \begin{bmatrix} D(0) & 0 & \dots & 0 \\ C(1)\Phi_{(1,1)}B(0) & D(1) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C(N)\Phi_{(N,1)}B(0) & C(N)\Phi_{(N,2)}B(1) & \dots & D(N) \end{bmatrix}$$

$$\underline{C} = \begin{bmatrix} C(0)\Phi_{(0,0)} \\ C(1)\Phi_{(1,0)} \\ \vdots \\ C(N)\Phi_{(N,0)} \end{bmatrix} \quad \underline{\mathbf{y}}_k = \begin{bmatrix} \mathbf{y}_k(0) \\ \mathbf{y}_k(1) \\ \vdots \\ \mathbf{y}_k(N) \end{bmatrix} \quad \underline{\mathbf{u}}_k = \begin{bmatrix} \mathbf{u}_k(0) \\ \mathbf{u}_k(1) \\ \vdots \\ \mathbf{u}_k(N) \end{bmatrix} \quad \underline{\mathbf{d}}_k = \begin{bmatrix} \mathbf{d}_k(0) \\ \mathbf{d}_k(1) \\ \vdots \\ \mathbf{d}_k(N) \end{bmatrix} \quad (4)$$

and $\underline{H}$ is similarly defined. In (4) underline notation denotes a lifted system matrix or a lifted vector. The matrix $\underline{D}$ is referred to as the lifted impulse response matrix. If $D(t) = 0$, $\forall t \in \{0, 1, \dots, N\}$, we define $\underline{\mathbf{y}}_k = [\mathbf{y}_k(1)^T \dots \mathbf{y}_k(N)^T]^T$, $\underline{\mathbf{u}}_k = [\mathbf{u}_k(0)^T \dots \mathbf{u}_k(N-1)^T]^T$ and

$$\underline{D} = \begin{bmatrix} C(1)B(0) & 0 & \dots & 0 \\ C(2)\Phi_{(2,1)}B(0) & C(2)B(1) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C(N)\Phi_{(N,1)}B(0) & C(N)\Phi_{(N,2)}B(1) & \dots & C(N)B(N-1) \end{bmatrix} \quad (5)$$

We assume that $\underline{D}$ has a full column rank. For simplicity, we also assume that each trial starts from the zero initial state, that is $\mathbf{x}_k(0) = 0$, $\forall k \in \mathbb{N}_0$, and that the process disturbances are trial invariant, that is, $\underline{\mathbf{d}}_k = \underline{\mathbf{d}}$, $\forall k \in \mathbb{N}_0$. We define the difference operator $\Delta\underline{\mathbf{z}}_k = \underline{\mathbf{z}}_{k+1} - \underline{\mathbf{z}}_k$, where $\underline{\mathbf{z}}_k$ is an arbitrary lifted vector. The lifted tracking error is defined as:

$$\underline{\mathbf{e}}_k = \underline{\mathbf{y}}_{ref} - \underline{\mathbf{y}}_k \quad (6)$$

where $\underline{\mathbf{y}}_{ref}$ is the lifted reference trajectory vector. Taking into account (3), we represent the lifted tracking error as:

$$\underline{\mathbf{e}}_k = \underline{\mathbf{y}}_{ref} - \underline{\mathbf{b}} - \underline{D}\underline{\mathbf{u}}_k \quad (7)$$

where $\underline{\mathbf{y}}_{ref}$ and $\underline{\mathbf{b}} = \underline{C}\mathbf{x}_k(0) + \underline{H}\underline{\mathbf{d}}$ are trial invariant parts of $\underline{\mathbf{e}}_k$. Taking into account all previous assumptions, we have:

$$\underline{\mathbf{e}}_{k+1} = \underline{\mathbf{e}}_k - \underline{D}\Delta\underline{\mathbf{u}}_k \quad (8)$$

The main goal of the ILC strategy is to reach a small value of the tracking error in a relatively small number of trials. The widely used ILC law [1], [5] can be expressed in a generic form:

$$\underline{\mathbf{u}}_{k+1} = Q\left(\underline{\mathbf{u}}_k + L\underline{\mathbf{e}}_k\right) \quad (9)$$

where Q and L are matrices of appropriate dimensions, and are referred to as the Q-filter and the learning matrix respectively. The system (8) controlled by (9) is referred to as the ILC system.

A. Design of optimal ILC laws

In the ILC with quadratic criterion (Q-ILC or norm-optimal ILC) [12], [9], [4], [5], the ILC law has been derived as the solution of the unconstrained quadratic optimization problem:

Unconstrained ILC with quadratic cost design problem.

$$\min_{\underline{\mathbf{u}}_{k+1}} g(\underline{\mathbf{u}}_{k+1}) = \min_{\underline{\mathbf{u}}_{k+1}} \; (\underline{\mathbf{e}}_{k+1})^T W_e \underline{\mathbf{e}}_{k+1} + (\Delta\underline{\mathbf{u}}_k)^T W_{\Delta u} \Delta\underline{\mathbf{u}}_k + (\underline{\mathbf{u}}_{k+1})^T W_u \underline{\mathbf{u}}_{k+1} \quad (10)$$

---

where W_e is positive definite, $W_{\Delta u}$ and W_u are positive semi-definite weighting matrices. The solution of (10) can be expressed analytically, with the learning matrices given by:

$$Q = (\underline{D}^T W_e \underline{D} + W_{\Delta u} + W_u)^{-1}(\underline{D}^T W_e \underline{D} + W_{\Delta u})$$
$$L = (\underline{D}^T W_e \underline{D} + W_{\Delta u})^{-1} \underline{D}^T W_e \quad (11)$$

In practice, actuators of ILC system are often subjected to different types of physical constraints. We name just a few: the constraint on the rate of the input change between two trials (input rate constraint), the constraint on the input amplitude (input saturation), the constraint on the total input energy in one trial (input energy constraint) and etc. For simplicity reasons we consider the input rate constraint and input energy constraint. The input rate constraint is expressed analytically:

$$-a_2 \leq \Delta \mathbf{u}_k(t) \leq a_1 \tag{12}$$

where $a_1, a_2 > 0$ represent the constraints. The constraint (12) can be represented in the trial domain:

$$W_c \Delta \underline{\mathbf{u}}_k \preceq \underline{\mathbf{c}} \tag{13}$$

where $W_c = \begin{bmatrix} I & -I \end{bmatrix}^T$ and $\underline{\mathbf{c}} = \begin{bmatrix} a_1 & \dots & a_1 & a_2 & \dots & a_2 \end{bmatrix}^T$ have dimensions corresponding to $\Delta \underline{\mathbf{u}}_k$, and the symbol $\preceq$ denotes the element-wise less than or equal relation. The input energy constraint is expressed analytically:

$$\frac{1}{2} \Delta \underline{\mathbf{u}}_k^T W_E \Delta \underline{\mathbf{u}}_k \leq b \tag{14}$$

where W_E is a positive-definite weighting matrix and $b > 0$ is a constraint. Taking these constraints into account, we formulate the optimization problem:

Constrained ILC with quadratic cost design problem

$$\begin{aligned} &\min_{\Delta \underline{\mathbf{u}}_k} f_0(\Delta \underline{\mathbf{u}}_k) = \min_{\Delta \underline{\mathbf{u}}_k} \ \frac{1}{2} (\underline{\mathbf{e}}_{k+1})^T W_e \underline{\mathbf{e}}_{k+1} \\ &\text{subject to } \ f(\Delta \underline{\mathbf{u}}_k) \preceq 0 \end{aligned} \tag{15}$$

---

where W_e is positive-definite, $\underline{\mathbf{e}}_{k+1}$ is given by (8) and

$$f(\Delta \underline{\mathbf{u}}_k) = \begin{bmatrix} W_c \Delta \underline{\mathbf{u}}_k - \underline{\mathbf{c}} \\ \frac{1}{2} \Delta \underline{\mathbf{u}}_k^T W_E \Delta \underline{\mathbf{u}}_k - b \end{bmatrix} \tag{16}$$

Due to simplicity we have only included the penalization of $\underline{\mathbf{e}}_{k+1}$ in (15). The optimization problem (15) is convex.

B. Stability and convergence analysis

Substituting the ILC law (9) into (8) and using the relation $\underline{\mathbf{e}}^k = \underline{\mathbf{y}}^{ref} - \underline{\mathbf{y}}^k$ we arrive at:

$$\underline{\mathbf{u}}_{k+1} = Q(I - L\underline{D})\underline{\mathbf{u}}_k + QL(\underline{\mathbf{y}}_{ref} - \underline{\mathbf{b}}) \tag{17}$$

The stability and monotonic convergence properties of the ILC system (17) have been extensively studied in the literature [1], [5] and are mainly determined by the spectral properties of $Q(I - LD)$. The ILC system is stable if and only if:

$$\rho(Q(I - L\underline{D})) < 1 \tag{18}$$

where $\rho(.)$ denotes the matrix spectral radius. The convergence rate γ of ILC system is given by:

$$\gamma = \|Q(I - L\underline{D})\|_2 \tag{19}$$

The lifted impulse response matrix (4) (or (5)) satisfies the Sequentially Semi Separable (SSS) matrix structure [7]. In this paper we exploit the SSS structure to derive a linear computational complexity ($\mathcal{O}(N)$) framework for ILC design and analysis. In that sense we define:

Problem 1. Assuming that all weighting matrices in optimization problems (10) and (15) are chosen to satisfy the Sequentially-Semi Separable (SSS) structure,

- Implement analyitical solution of (10) given by (11) with $\mathcal{O}(N)$ complexity.
- Using the primal-dual method (PDM) [3], solve (15) and implement one iteration of the PDM method with $\mathcal{O}(N)$ complexity.
- Assuming that the Q-filter and learning matrix L of the ILC law (9) satisfy (11), compute the convergence rate (19) of ILC system with $\mathcal{O}(N)$ complexity.

In the next section we revise some basic properties of SSS structure, whereas in Section 4 we exploit these properties to solve Problem 1.

III. Sequentially Semi Separable Structure

The informal definition of the SSS structure can be given as follows [7].

Let S be an $M \times M$ (possibly complex) matrix satisfying the matrix structure. Then there exist N positive integers $m_1, \dots, m_N$ with $M = m_1 + \dots + m_N$ to block partition S as $S = (S_{i,j})$, where $S_{i,j} \in \mathbb{C}^{m_i \times m_j}$ satisfies

$$S_{i,j} = \begin{cases} D_i, & \text{if } i = j \\ U_i W_{i+1} \dots W_{j-1} V_j^H, & \text{if } j > i \\ P_i R_{i-1} \dots R_{j+1} Q_j^H, & \text{if } j < i \end{cases} \tag{20}$$

where the superscript H denotes the Hermitian transpose. For example, if $N = 4$ the structure of S can be visualized as follows:

$$S = \begin{bmatrix} D_1 & U_1 V_2^H & U_1 W_2 V_3^H & U_1 W_2 W_3 V_4^H \\ P_2 Q_1^H & D_2 & U_2 V_3^H & U_2 W_3 V_4^H \\ P_3 R_2 Q_1^H & P_3 Q_2^H & D_3 & U_3 V_4^H \\ P_4 R_3 R_2 Q_1^H & P_4 R_3 Q_2^H & P_4 Q_3^H & D_4 \end{bmatrix} \tag{21}$$

Since the SSS matrix structure is uniquely determined with the sequences of matrices $P_i, R_i, Q_i, U_i, W_i, V_i$ and D_i they are called generator matrices of S. We denote the SSS matrices in a compact manner [17]:

$$S = \mathcal{SSS}\left(P_i, R_i, Q_i, U_i, W_i, V_i, D_i\right) \tag{22}$$

In the spirit of (22) and with slight abuse of the notation, the matrix $\underline{D}$ defined by (4) can be written in terms of its generators:

$$\underline{D} = \mathcal{SSS}\left(C(i), A(i), B(i), 0, 0, 0, D(i)\right) \tag{23}$$

Similarly, matrix $\underline{D}$ defined in (5) can also be written in terms of its generators.

The class of SSS matrices is attractive from computational point of view due to following facts [8], [7], [18]:

- The computational cost of the basic matrix operations $(+-,\times,^{-1})$ and matrix-vector multiplications is linear in N. Furthermore, the SSS matrices can be compactly stored in the computer memory.
- The algebra of SSS matrices is closed under the basic matrix operations.

The generators of a matrix obtained as a result of basic matrix operations on SSS matrices, are directly computed by the efficient SSS algorithms summarized in [7], [18].

IV. Linear computational complexity framework for ILC analysis and design

In Problem 1 (see Section 2), it is assumed that all weighting matrices of optimization problems (10) and (15) are choosen to satisfy the SSS structure. This assumption is not restrictive due to following. The most simple selection of weighting matrices, that satisfies the SSS structure, is a diagonal selection. Next, in [15] (see Chapter 14) a non-diagonal selection of weighting matrices has been proposed. The authors select $W_u = (HH^T)^{-1}$, where H is the Toeplitz matrix with the first column being the first $N+1$ Markov parameters of a low pass filter (e.g. Butterworth filter), whereas W_e and $W_{\Delta u}$ are selected as diagonal. The matrix H with the Toeplitz structure satisfies the SSS structure and consequently W_u satisfies the SSS structure. All vector-matrix and matrix operations, that are used in the algorithms presented in this section, are performed using the efficient SSS algorithms presented in [7], [18].

A. $\mathcal{O}(N)$ implementation of unconstrained ILC with quadratic cost

The solution of the unconstrained optimization problem (10) has analytic form (11) that is computed by performing vector-matrix multiplications and the basic matrix operations $(+-,\times,^{-1})$ on the lifted impulse response matrix $\underline{D}$ and weighting matrices W_e, W_u and $W_{\Delta u}$. Since these matrices satisfy the SSS structure, the solution can be computed with $\mathcal{O}(N)$ computational complexity.

B. Computationally efficient implementation of the primal-dual interior point method

We briefly summarize the basic primal-dual method (PDM) presented in [3] and focus on the computational aspects of its implementation. The modified Karush-Kuhn-Tucker (KKT) equations associated with the optimization problem (15) are:

$$\underline{\mathbf{r}} = \begin{bmatrix} \underline{\mathbf{r}}_1 \\ \underline{\mathbf{r}}_2 \end{bmatrix} = \begin{bmatrix} \nabla f_0(\Delta\underline{\mathbf{u}}_k) + Df(\Delta\underline{\mathbf{u}}_k)^T\underline{\boldsymbol{\lambda}} \\ -diag(\underline{\boldsymbol{\lambda}})f(\Delta\underline{\mathbf{u}}_k) - (1/l)\underline{\mathbf{1}} \end{bmatrix} = \mathbf{0} \qquad (24)$$

where $Df(\Delta\underline{\mathbf{u}}_k)^T = \left[\nabla f_1(\Delta\underline{\mathbf{u}}_k) \dots \nabla f_{2(N+1)+1}(\Delta\underline{\mathbf{u}}_k)\right]$ and f_i denotes the ith element of (16), $\underline{\boldsymbol{\lambda}} \in \mathbb{R}^{2(N+1)+1}$ is a vector of Lagrange multipliers, $\underline{\mathbf{1}} \in \mathbb{R}^{2(N+1)+1}$ is a vector of ones and $l > 0$. The Newton step $\underline{\mathbf{n}} = \begin{bmatrix} \underline{\mathbf{n}}_1^T & \underline{\mathbf{n}}_2^T \end{bmatrix}^T$, for solving nonlinear system of equations (24), is computed as a solution of:

$$J\underline{\mathbf{n}} = -\underline{\mathbf{r}} \qquad (25)$$

where the Jacobian matrix J is:

$$J = \begin{bmatrix} \nabla^2 f_0(\Delta\underline{\mathbf{u}}_k) + \sum_{i=1}^{2(N+1)+1} \lambda_i \nabla^2 f_i(\Delta\underline{\mathbf{u}}_k) & Df(\Delta\underline{\mathbf{u}}_k)^T \\ -diag(\underline{\boldsymbol{\lambda}})Df(\Delta\underline{\mathbf{u}}_k) & -diag(f(\Delta\underline{\mathbf{u}}_k)) \end{bmatrix} \qquad (26)$$

Next we have:

$$\nabla f_0(\Delta\underline{\mathbf{u}}_k) = \underline{D}^T W_e \underline{D}\Delta\underline{\mathbf{u}}_k - \underline{D}^T W_e \underline{\mathbf{e}}_k$$

$$\nabla^2 f_0(\Delta\underline{\mathbf{u}}_k) = \underline{D}^T W_e \underline{D} \ , Df(\Delta\underline{\mathbf{u}}_k) = \begin{bmatrix} I \\ -I \\ \Delta\underline{\mathbf{u}}_k^T W_E \end{bmatrix}$$

$$\sum_{i=1}^{2(N+1)+1} \lambda_i \nabla^2 f_i(\Delta\underline{\mathbf{u}}_k) = \lambda_{2(N+1)+1} W_E \qquad (27)$$

We introduce following partitions:

$$diag(\underline{\boldsymbol{\lambda}}) = \begin{bmatrix} \Lambda_1 & & \\ & \Lambda_2 & \\ & & \lambda_{2(N+1)+1} \end{bmatrix}$$

$$diag(f(\Delta\underline{\mathbf{u}}_k)) = \begin{bmatrix} F_1 & & \\ & F_2 & \\ & & f_{2(N+1)+1} \end{bmatrix} \qquad (28)$$

where $\Lambda_1 = diag(\underline{\boldsymbol{\lambda}}(1:N+1))$, $\Lambda_2 = diag(\underline{\boldsymbol{\lambda}}(N+2:2(N+1)))$, $F_1 = diag(f(1:N+1))$, $F_2 = diag(f(N+2:2(N+1)))$, $\lambda_{2(N+1)+1} = \boldsymbol{\lambda}(2(N+1)+1)$, $f_{2(N+1)+1} = f(2(N+1)+1)$ and $(1:N+1)$ denotes the standard MATLAB® notation for choosing first $N+1$ elements of a vector. Substituting the last expression in (26) we obtain:

$$J = \left[\begin{array}{c|ccc} \underline{D}^T W_e \underline{D} + \lambda_{2(N+1)+1} W_E & I & -I & W_E\Delta\underline{\mathbf{u}}_k \\ \hline -\Lambda_1 & -F_1 & & \\ \Lambda_2 & & -F_2 & \\ -\lambda_{2(N+1)+1}\Delta\underline{\mathbf{u}}_k^T W_E & & & -f_{2(N+1)+1} \end{array}\right] \qquad (29)$$

The PDM for solving optimization problem (15) can be summarized as follows:

Algorithm 1: Primal-dual interior point method (PDM) [3]. Given initial point $\Delta\underline{\mathbf{u}}_k$ that satisfies $f(\Delta\underline{\mathbf{u}}_k) \prec 0$,$\underline{\boldsymbol{\lambda}} \succ 0$, $\mu > 1$, $\epsilon > 0$ and $\epsilon_f > 0$,

repeat

1) Compute $\hat{\eta} = -f(\Delta\underline{\mathbf{u}}_k)\underline{\boldsymbol{\lambda}}$ and $l = \mu(2(N+1)+1)/\hat{\eta}$.
2) Compute the primal-dual search direction $\underline{\mathbf{n}}$ by solving system (25).
3) Determine the step length $s > 0$ and update $\Delta\underline{\mathbf{u}}_k := \Delta\underline{\mathbf{u}}_k + s\underline{\mathbf{n}}_1$ and $\underline{\boldsymbol{\lambda}} := \underline{\boldsymbol{\lambda}} + s\underline{\mathbf{n}}_2$

until $\|\mathbf{r}_1\|_2 \leq \epsilon_f$, $\|\mathbf{r}_2\|_2 \leq \epsilon_f$ and $\hat{\eta} \leq \epsilon$.

In the third step of Algorithm 1, it is required to compute the step length s. This step can be performed using the modified version of backtracking line search that ensures that the next

iterates of $\boldsymbol{\lambda}$ and $\Delta\underline{\mathbf{u}}_k$ are feasible. For more details about the backtracking line search and how to chose the constants μ,ϵ and ϵ_f see Section 11.7.3 of [3] and references therein.

In each iteration of the Algorithm 1, the stopping criterion needs to be checked. That is, the values of $\|\underline{\mathbf{r}}_1\|_2$ and $\|\underline{\mathbf{r}}_2\|_2$ need to be computed. Due to the fact that all weighting matrices, together with lifted impulse matrix $\underline{D}$ and diagonal matrices in expressions (24) and (27) satisfy the SSS structure, we conclude that $\|\underline{\mathbf{r}}_1\|_2$ and $\|\underline{\mathbf{r}}_2\|_2$ can be computed with $\mathcal{O}(N)$ complexity.

In the first step of Algorithm 1, the value of $\hat{\eta}$ needs to be computed. This can be done with $\mathcal{O}(N)$ complexity, since matrices in (16) satisfy the SSS structure.

Similarly we can conclude that the step 3 of Algorithm 1 can be performed with $\mathcal{O}(N)$.

In the second step of Algorithm 1, we need to determine primal-dual search direction $\underline{\mathbf{n}}$. Its value can be computed using the Schur complement:

$$\underline{\mathbf{n}}_1 = (J_{11} - J_{12}J_{22}^{-1}J_{21})^{-1}(J_{12}J_{22}^{-1}\underline{\mathbf{r}}_2 - \underline{\mathbf{r}}_1) \tag{30}$$

$$\underline{\mathbf{n}}_2 = J_{22}^{-1}(-\underline{\mathbf{r}}_2 - J_{21}\underline{\mathbf{n}}_1) \tag{31}$$

where J_{ij} coresponds to the (i,j) block of (29). Next we have:

$$J_{11} - J_{12}J_{22}^{-1}J_{21} = \underline{D}^T W_e \underline{D} + \lambda_{2(N+1)+1} W_E - F_1^{-1}\Lambda_1 - F_2^{-1}\Lambda_2 - \frac{\lambda_{2(N+1)+1}}{f_{2(N+1)+1}} W_E \Delta\underline{\mathbf{u}}_k \Delta\underline{\mathbf{u}}_k^T W_E \tag{32}$$

and where:

$$\Delta\underline{\mathbf{u}}_k\Delta\underline{\mathbf{u}}_k^T = \begin{bmatrix} \Delta\mathbf{u}_k(1)^2 & \Delta\mathbf{u}_k(1)\Delta\mathbf{u}_k(2) & \dots & \Delta\mathbf{u}_k(1)\Delta\mathbf{u}_k(N+1) \\ \Delta\mathbf{u}_k(2)\Delta\mathbf{u}_k(1) & \Delta\mathbf{u}_k(2)^2 & \dots & \Delta\mathbf{u}_k(2)\Delta\mathbf{u}_k(N+1) \\ \vdots & \vdots & \vdots & \vdots \\ \Delta\mathbf{u}_k(N+1)\Delta\mathbf{u}_k(1) & \dots & \dots & \Delta\mathbf{u}_k(N+1)^2 \end{bmatrix}$$

By comparing $\Delta\underline{\mathbf{u}}_k\Delta\underline{\mathbf{u}}_k^T$ and the general form of the SSS structure (21) we conclude that $\Delta\underline{\mathbf{u}}_k\Delta\underline{\mathbf{u}}_k^T$ satisfies the SSS structure, with the generators given by:

$$D_i = \mathbf{u}_k(i)^2 \ , \ P_i = \mathbf{u}_k(i) \ , \ R_i = 1 \ , \ Q_i = \mathbf{u}_k(i) \ \dots \tag{33}$$

Since F_i and Λ_i, $i = 1, 2$ are diagonal matrices they satisfy the SSS structure. Due to the fact that W_E, $\underline{D}$, I, and $\Delta\underline{\mathbf{u}}_k\Delta\underline{\mathbf{u}}_k^T$ satisfy the SSS structure, we conclude that the Schur complement (32) (together with its inverse) satisfies the SSS structure. We can transform the second term of product in (30) as follows:

$$J_{12}J_{22}^{-1}\underline{\mathbf{r}}_2 - \underline{\mathbf{r}}_1 = F_2^{-1}\underline{\mathbf{r}}_2^{(2)} - F_1^{-1}\underline{\mathbf{r}}_2^{(1)} - \Delta\underline{\mathbf{u}}_k W_E \frac{r_2^{(3)}}{f_{2(N+1)+1}} - \underline{\mathbf{r}}_1 \tag{34}$$

where $\underline{\mathbf{r}}_2 = \begin{bmatrix} (\underline{\mathbf{r}}_2^{(1)})^T & (\underline{\mathbf{r}}_2^{(2)})^T & r_2^{(3)} \end{bmatrix}^T$ and the dimensions of $\underline{\mathbf{r}}_2^{(i)}$, $i = 1, 2, 3$ correspond to the partition of $diag(f)$ (28). Since all matrices in (34) have diagonal structure, the computational effort of determining (34) is $\mathcal{O}(N)$. Due to this and due to the fact that the inverse of the Schur complement satisfies the SSS structure, the computational cost of evaluating (30) is $\mathcal{O}(N)$. Similarly it can be shown that computational cost of (31) is $\mathcal{O}(N)$ This implies that the second step of Algorithm 1 can be performed with $\mathcal{O}(N)$.

C. Computationally efficient stability check and convergence rate computation

In order to compute the spectral radius of the ILC system, we need to compute the maximal eigenvalue of $Q(I - L\underline{D})$. Similarly, to determine the convergence rate of the ILC system we need to determine the square root of the maximal eigenvalue of $(I - L\underline{D})^T Q^T Q (I - L\underline{D})$.

One of the most simplest algorithms to determine the maximal eigenvalue of a matrix is the Power Method (PM) [6]. The PM for computing the maximal eigenvalue of $(I - L\underline{D})^T Q^T Q (I - L\underline{D})$ consits of the following iteration:

$$\underline{\mathbf{h}}^{j+1} = \frac{(I - L\underline{D})^T Q^T Q (I - L\underline{D})\underline{\mathbf{h}}^j}{\left\|(I - L\underline{D})^T Q^T Q (I - L\underline{D})\underline{\mathbf{h}}^j\right\|_2} \tag{35}$$

where j is an iteration index. The iteration (35) starts with a vector $\underline{\mathbf{h}}^0$, which can be chosen as a random vector or as an approximation of the dominant eigenvector of $(I - L\underline{D})^T Q^T Q (I - L\underline{D})$. If the sequence generated by (35) converges, the sequence:

$$\lambda_j = \frac{(\underline{\mathbf{h}}^j)^T (I - L\underline{D})^T Q^T Q (I - L\underline{D})\underline{\mathbf{h}}^j}{(\underline{\mathbf{h}}^j)^T \underline{\mathbf{h}}^j} \tag{36}$$

converges to the dominant eigenvalue of $(I - L\underline{D})^T Q^T Q (I - L\underline{D})$. Convergence properties of PM, together with other methods for computing the maximal eigenvalue of a matrix (that are based on iteration (35)), are summarized in [6]. Since the weighting matrices W_e, W_u and $W_{\Delta u}$ satisfy the SSS structure, Q and L (defined in (11)) satisfy the SSS structure. This is due to the fact that the SSS algebra is closed under the basic matrix operations. Similarly, the matrices $Q(I - L\underline{D})$ and $(I - L\underline{D})^T Q^T Q (I - L\underline{D})$ satisfy the SSS structure. Due to this, the computational cost of one iteration of PM (35) (as well as (36)) is $\mathcal{O}(N)$.

V. Simulations

All simulations are performed on the standard desktop personal computer. The SSS algorithms summarized in [7], [18] are implemented in MATLAB®. We consider the system (1)-(2) defined by:

$$A = \begin{bmatrix} -0.7 & -0.5 \\ 1 & 0.2 \end{bmatrix} \ , \ B = \begin{bmatrix} 2 \\ 0.5 \end{bmatrix} \ , C = \begin{bmatrix} 1 & 0 \end{bmatrix} \ , \ D = 0 \tag{37}$$

We assume zero initial state. The graph of reference trajectory is left out due to space constraint. For the simplicity reasons weighting matrices W_e and W_E are chosen as diagonal. The same computational efficiency can be also observed for the fully populated and SSS structured weighting matrices. For the trial length of $N = 1000$, we have solved the optimization problem (15) using Algorithm 1. The parameters are $\mu = 25$, $\epsilon_f = \epsilon = 0.005$. In order to illustrate the effect of the constraints on the learning transients, we have computed the solution for the two set of the constraints

$K_1 = (a_1, a_2, b) = (2, 2, 15)$ and $K_2 = (a_1, a_2, b) = (3, 3, 100)$. The tracking performance in the trial domain is presented in Figure 1 (left). The input rate values between trial 1 and 2, for the set of weights K_1 and K_2, are presented in Figure (1) (right).

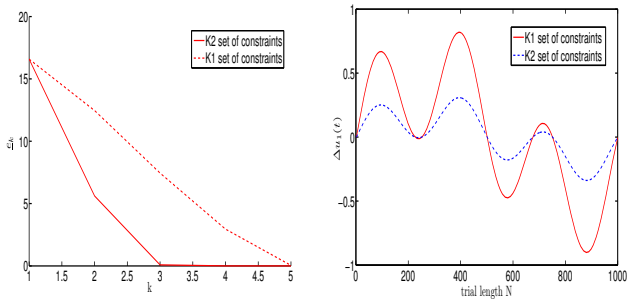

Fig. 1. Left: tracking performance in the trial domain for the set of constraints $K_1 = (a_1, a_2, b) = (2, 2, 15)$ and $K_2 = (a_1, a_2, b) = (3, 3, 100)$. Right: input rate values between trial 1 and 2.

In Figure 1 the effect of the input energy constraint on the learning transient can be observed. By decreasing the input energy constraint (the value of b), the convergence rate of the ILC system is decreased and vice-versa. It can be also shown that by decreasing the input rate constraint the convergence rate of the ILC system will be decreased and vice-versa. From Figure 1 (right) we can see that the input rates and input energy (by integrating the amplitude) are in the set defined by the constraints.

The primal-dual search direction (25) is computed using efficient SSS algorithms for different trial lengths. For comparation purpose, the primal-dual search direction is computed using standard MATLAB® matrix operations. The computational times are expressed in seconds and presented in Figure 2 (left). The linear computational complexity of computing the primal-dual search direction using SSS algorithms is evident.

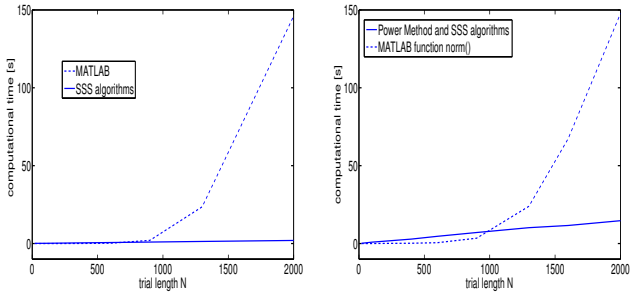

Fig. 2. Left: necessary time to compute the primal-dual search direction (25) using the SSS algorithms and the standard MATLAB matrix operations. Right: necessary time to compute $\|Q(I - \underline{D}L)\|_2$, using the MATLAB function norm() and by using the PM in a combination with SSS algorithms.

We have computed the convergence rate (19) for different trial lengths N. The learning matrices are defined in (11). The weighting matrices of (11) are: $W_e = I$, $W_{\Delta u} = I$ and $W_u = I$ (also any other selection that satisfies the SSS structure will give similar results). Following the methodology presented in Section 4 we have computed the convergence rate using the Power Method (PM) and by exploiting the SSS structure. For comparation, we have used the MATLAB® function $norm(.)$. The computation times are presented in Figure 2 (right). The linear computational complexity of the PM and SSS algorithms is evident. This is in contrast to the computational complexity of MATLAB function $norm(.)$

VI. CONCLUSION

In this paper we have exploited the sequentially-semi separable structure of the lifted system matrices to reduce the computational complexity of the constrained ILC design methods. We have also proposed a computationally efficient method for stability check and computation of convergence rate of ILC system.

REFERENCES

[1] H.S. Ahn, Y.Q. Chen, and K.L. Moore. Iterative learning control: Brief survey and categorization. *IEEE Transactions On Systems, Man, And Cybernetics, Part C: Applications and Reviews*, 37(6):1099, 2007.

[2] K.L. Barton, D.A. Bristow, and A. G. Alleyne. A Numerical Method for Determining Monotonicity and Convergence Rate In Iterativel Learning Control. *International Journal of Control*, 2010.

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

[4] D. A. Bristow. Weighting matrix design for robust monotonic convergence in norm optimal iterative learning control. In *Proceedings of the American Control Conference*, pages 4554–4560, 2008.

[5] D.A. Bristow, M. Tharayil, and A.G. Alleyne. A survey of iterative learning control. *IEEE control systems magazine*, 26(3):96–114, 2006.

[6] R.L. Burden and J.D. Faires. Numerical Analysis. *Brooks/Cole*, 2001.

[7] S. Chandrasekaran, P. Dewilde, M. Gu, T. Pals, A.J. van der Veen, and D. White. Fast Stable Solvers for Sequentially Semi-Separable Linear Systems of Equations. *Report, Lawrence Livermore National Library*, 2003.

[8] P. Dewilde and A.J. Van der Veen. *Time-varying systems and computations*. Kluwer Academic Pub, 1998.

[9] S. Gunnarsson and M. Norrlöf. On the design of ILC algorithms using optimization. *Automatica*, 37:2011–2016, 2001.

[10] W.B.J. Hakvoort, R.G.K.M. Aarts, J. van Dijk, and J.B. Jonker. Lifted system iterative learning control applied to an industrial robot. *Control Engineering Practice*, 2008.

[11] W.B.J. Hakvoort, R.G.K.M. Aarts, J. van Dijk, and J.B. Jonker. A computationally efficient algorithm of iterative learning control for discrete-time linear time-varying systems. *Automatica*, 45:2925–2929, 2009.

[12] J.H. Lee, K.S. Lee, and W.C. Kim. Model-based iterative learning control with a quadratic criterion for time-varying linear systems. *Automatica*, 36:641–657, 2000.

[13] S. Mishra, U. Topcu, and M. Tomizuka. Optimization-based Constrained Iterative Learning Control. *to appear in the IEEE Transactions on Control Systems Technology*, 2010.

[14] K.L. Moore, M. Johnson, and M.J. Grimble. *Iterative Learning Control for Deterministic Systems*. Springer-Verlag New York, Inc. Secaucus, NJ, USA, 1993.

[15] M. Norrlöf. *Iterative Learning Control: Analysis, Design and Experiments*. Thesis no. 653, Linköpings universitet, 2000.

[16] M.Q. Phan, R.W. Longman, and K.L. Moore. Unified formulation of linear iterative learning control. *Spaceflight mechanics 2000*, pages 93–111, 2000.

[17] J.K. Rice and M. Verhaegen. A Structured Matrix Approach to Efficient Calculation of LQG Repetitive Learning Controllers in the Lifted Setting. *Accepted for publication in International Journal of Control*, 2010.

[18] Justin K. Rice. *Efficient Algorithms for Distributed Control: A Structured Matrix Approach*. PhD thesis, Delft University of Technology, 2010.