Synthesizing Decentralized Local State Feedback Controller for Distributed Systems Using Nuclear Norm Approximation

Yoke Peng Leong, Yuh-Shyang Wang

June 6, 2013
1 Introduction

Synthesizing a decentralized controller for a distributed system has been a long studied subject in the area of control theory. Numerous researchers have found analytical and numerical solutions to such problems [1]. However, they still have not considered the idea of a decentralized local state feedback controller for a distributed system with communication delays – that is a controller at a node of a network depends only on a small amount of neighboring nodes and it receives information from the neighboring nodes with some delays. In this project, we are interested in synthesizing a decentralized local state feedback controller for a distributed system given the dynamics of the plant and the delay pattern of its communication network.

This new idea of synthesizing a localized state feedback controller for a distributed system is based on a simple intuition - a controller at a node (let us call it node 1) should not need to know the states of nodes far away if neighboring controllers at neighboring nodes can ensure that any disturbance far enough will never reach node 1. To illustrate this idea, consider a smart grid network across the United States. Ideally, a blackout at New York should not affect any power users or generators at Los Angeles because generators/controllers at cities near New York should have cooperatively restricted the effect of the blackout within some radius away from New York. This anecdote highlights two features that one desires in controller synthesis: (a) a localized controller that only depends on local states, and (b) a collection of local controllers that restricts the effect of a disturbance to a small neighborhood of the source. Such localized method is preferable because the computational cost of a controller at each node of a large distributed network is greatly reduced. In addition, the risk of losing information is decreased because each control node is required to communicate only a small amount of information for short distances.

In general, solving for a localized controller is not necessarily a convex nor tractable problem. Hence, several relaxations and transformations are performed on the original problem so that the controller can be computed numerically in a tractable way. In Section 2, the original problem is transformed into a rank minimization problem with affine constraints. Generally, rank minimization is a NP-hard problem. Nonetheless, other researchers [2, 3] have shown that a rank minimization problem can be relaxed into a nuclear norm minimization problem. This relaxed formulation results in a convex optimization problem with affine constraints. With this relaxation, our simulations show that it is possible to recover a reasonable decentralized local state feedback controller. However, further research is required to determine the technical conditions.

The rest of the report is organized as follows: Section 2 formally describes the problem statement and formulates the optimization problem. Section 3 illustrates the controller synthesis method using a specific example. Numerical results are discussed in Section 4. Lastly, Section 5 summarizes this project and presents some possible future directions based on this work.
2 Problem Formulation

In this section, a formal description of the problem we are interested in and its connection to nuclear norm minimization is presented.

2.1 Problem Statement

This project aims to derive a method to synthesize a localized state feedback controller to achieve a finite closed loop impulse response. Formally, consider a discrete linear time invariant distributed plant of the following form:

\[
\begin{align*}
    x[k+1] &= Ax[k] + w[k] + Bu[k] \\
    y[k] &= x[k] \\
    z[k] &= x[k] 
\end{align*}
\]

(1)

where \( A \in \mathbb{R}^{p \times p} \), \( B \in \mathbb{R}^{p \times n_u} \), \( x[k] \in \mathbb{R}^p \), and \( u[k] \in \mathbb{R}^{n_u} \). The states at time \( k \) are denoted as \( x[k] \) and the total number of states is \( p \). The control inputs at time \( k \) are denoted as \( u[k] \) and the total number of controllers is \( u_n \). The impulse disturbance at time \( k \) is denoted as \( w[k] \) where \( w[k] = \delta[k]e_j \) for \( j = 1, \ldots, p \). Full state measurements are assumed by identifying \( y \) to \( x \) and \( z[k] \) is the controlled output at time \( k \).

The goal is to synthesize a linear state feedback controller, \( u[z] = K[z] x[z] \), that results in finite impulse responses for both the closed loop system and the controller respectively. Note that \( z \) is the variable of Z-transform of the system in frequency domain. Controller at node \( i \), \( K_i[z] \), will only depend on the measurement from some local states, and \( K[z] \) conforms to a structure that is derived from the delay pattern of the communication network. Define a set \( \Psi \) as \( \Psi = \{ K | K \text{ conforms to the communication delay pattern} \} \). In other words, \( \Psi \) is a set of all possible controllers, \( K \).

2.2 Closed loop impulse response

The plant in (1) can be written as

\[
P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} = \begin{bmatrix} A & I & B \\ I & 0 & 0 \\ I & 0 & 0 \end{bmatrix}
\]

(2)

where \( P_{11} \) is the transfer function from \( w \) to \( z \), \( P_{12} \) is the transfer function from \( u \) to \( z \), \( P_{21} \) is the transfer function from \( w \) to \( y \), and \( P_{22} \) is the transfer function from \( u \) to \( y \). The closed loop system has dynamics given by:

\[
\begin{align*}
    z &= (P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}) w \\
    y &= (I - P_{22}K)^{-1}P_{21}w
\end{align*}
\]

(3)
Hence, the transfer function from $w$ to $z$ in the closed loop system is

$$P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}$$

(4)

where $P_{11} = P_{21} = (zI - A)^{-1}$ and $P_{12} = P_{22} = (zI - A)^{-1}B$.

First, rewrite (4) as

$$P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} = (I - (zI - A)^{-1}BK)(I - (zI - A)^{-1}BK)^{-1}(zI - A)^{-1}.$$

(5)

Then, substitute $I = (I - (zI - A)^{-1}BK)(I - (zI - A)^{-1}BK)^{-1}$ into (5) and simplify:

$$P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21} = (I - (zI - A)^{-1}BK)^{-1}(zI - A)^{-1}.$$ 

(6)

Thus, assuming a linear state feedback controller, the closed loop impulse response is $(zI - A - BK)^{-1}$.

### 2.3 Controller Design

Suppose the controller is strictly proper and has a finite impulse response. Then, $BK$ in (6) can be rewritten as

$$BK = \sum_{i=1}^{N_c} \frac{1}{z^i}C_i$$

(7)

where $C_i \in \mathbb{R}^{p \times p}$ and $N_c$ is the length of the finite impulse response of controller $K$. Substitute (7) into (6) and factor out a $\frac{1}{z}$ to get

$$(zI - A - BK)^{-1} = \frac{1}{z} \left( I - \frac{1}{z}A - \frac{1}{z} \sum_{i=1}^{N_c} \frac{1}{z^i}C_i \right)^{-1}.$$ 

(8)

Because $\{C_i\}$ is derived from a controller $K$, $\{C_i\}$ has to satisfy constraints that is derived from the fact that $K \in \Psi$. Denote these constraints as controller sparsity constraints and define $\Omega$ as the set of all $\{C_i\}$ that satisfy controller sparsity constraints.

Suppose the desired closed loop response has a finite impulse response. Then, it has the following form:

$$(zI - A - BK)^{-1} = \frac{1}{z} \left( I + \sum_{j=1}^{N_d} \frac{1}{z^j}D_j \right)^{-1}.$$ 

(9)
where $D_i \in \mathbb{R}^{p \times p}$ and $N_d$ is the desired length of the finite impulse response of the closed loop system. In order to ensure locality of the effect of an impulse disturbance, $\{D_j\}$ has to satisfy some locality structure. Denote these constraints as closed loop sparsity constraints and define the set of $\{D_j\}$ that satisfy the closed loop sparsity constraints as $\Phi$.

Some examples of a controller sparsity constraint or a closed loop sparsity constraint would be

$$C_i = \begin{bmatrix} * & 0 & 0 \\ 0 & * & 0 \\ 0 & 0 & * \end{bmatrix} \quad D_j = \begin{bmatrix} * & * & 0 \\ * & * & * \\ 0 & * & * \end{bmatrix}$$

where * is any scalar constant. Note that these constraints are affine (specifically, linear) constraints that require certain elements in a matrix to be zero.

Identify (8) and (9), to obtain

$$\frac{1}{z} \left( I - \frac{1}{z} A - \frac{1}{z} \sum_{i=1}^{N_c} \frac{1}{z^i} C_i \right)^{-1} = \frac{1}{z} \left( I + \sum_{j=1}^{N_d} \frac{1}{z^j} D_j \right)$$

Equation (10) implies that

$$\left( I - \frac{1}{z} A - \frac{1}{z} \sum_{i=1}^{N_c} \frac{1}{z^i} C_i \right) \left( I + \sum_{j=1}^{N_d} \frac{1}{z^j} D_j \right) = I.$$  \hspace{1cm} (11)

To design a controller that has a finite impulse response and results in a closed loop system that also has a finite impulse response, one searches for $\{C_i\} \in \Omega$ and $\{D_j\} \in \Phi$ such that the coefficient of $\frac{1}{z^k}$ in the left hand side of (11) is a zero matrix for all $k = 1, \ldots, N_c + N_d + 1$.

### 2.4 Existence of Solutions

Given $A$, $N_c$, and $N_d$, an exact solution for (11) is not easily found in general. Furthermore, the solution might not be unique nor exist. In other words, there might be multiple $\{C_i\} \in \Omega$ and $\{D_j\} \in \Phi$ that satisfy (11) or no $\{C_i\} \in \Omega$ and $\{D_j\} \in \Phi$ that satisfies (11). Several known conditions for existence of solutions to (11) with specific $A$, $N_c$ and $N_d$ are listed:

1. If $A$ is a scalar, no solution exists.
2. Solutions exist for $N_c = 0 \iff A$ is nilpotent.
3. Solutions exist for $N_c = 1$ and $N_d = 2$ even when $A$ is not nilpotent.

Example:

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad C_1 = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \quad D_1 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \quad D_2 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

Because an exact solution is not easily found, a numerical method is proposed to approximate the solution to (11) for any given $A$. 

4
2.5 Rank Minimization Formulation

To solve for a set of \( \{C_i\} \) and \( \{D_j\} \) that satisfies (11), the original problem is reformulated as a rank minimization problem.

Define a matrix \( M \) to be

\[
M = \begin{bmatrix}
I & -A & -C_1 & \cdots & -C_{N_c} \\
-I & D_1 & \cdots & D_{N_d} \\
-AD_1 & -AD_{N_d} & \cdots & -C_1D_{N_d} \\
\vdots & \vdots & \ddots & \vdots \\
-C_{N_c} & -C_{N_c}D_1 & \cdots & -C_{N_c}D_{N_d}
\end{bmatrix}
\]

where \( \{C_i\} \in \Omega \) and \( \{D_j\} \in \Phi \). This matrix \( M \) has a rank of \( p \), the total number of states of the system in (1).

The constraints defined by (11) (i.e., coefficient of \( \frac{1}{z^k} \) is a zero matrix for all \( k = 1, \ldots, N_c + N_d + 1 \)) are equivalent to setting the sum of each anti-diagonal matrix blocks of \( M \) to zero. In other words, \(-A + D_1 = 0, -C_1 - AD_1 + D_2 = 0, \) and so on until \(-C_{N_c}D_{N_d} = 0\). Denote this set of constraints as the anti-diagonal constraint. Then, define \( \Theta \) as the set of all matrices that satisfy the anti-diagonal constraint, and each matrix has matrix blocks that satisfy the appropriate controller sparsity constraints and closed loop sparsity constraints. Note that the matrix block on the top left corner is fixed as an identity matrix and \( A \) is given by the plant dynamics in (1). Formally,

\[
\Theta = \left\{ M \in \mathbb{R}^{(N_c+2)p \times (N_d+1)p} \begin{array}{c}
M \text{ satisfies anti-diagonal constraint} \\
M_{1,1} = I \\
M_{2,1} = -A \\
M_{i,j} \in \Omega \ \forall \ i = 3, \ldots, N_c + 2 \text{ and } j = 1 \\
M_{i,j} \in \Phi \ \forall i = 1 \text{ and } j = 2, \ldots, N_d + 1 \\
\text{rank}(M) = p
\end{array} \right\}
\]

where each \( M_{i,j} \) is a \( p \times p \) matrix. Any \( \{C_i\} \) and \( \{D_j\} \) that satisfies (11) form a matrix \( M \in \Theta \) and vice versa. However, this set of matrices is not a convex set because of the rank constraint. So, the set is relaxed by defining a new set

\[
\Upsilon = \left\{ M \in \mathbb{R}^{(N_c+2)p \times (N_d+1)p} \begin{array}{c}
M \text{ satisfies anti-diagonal constraint} \\
M_{1,1} = I \\
M_{2,1} = -A \\
M_{i,j} \in \Omega \ \forall \ i = 3, \ldots, N_c + 2 \text{ and } j = 1 \\
M_{i,j} \in \Phi \ \forall i = 1 \text{ and } j = 2, \ldots, N_d + 1
\end{array} \right\}
\]
where the rank constraint is removed. Thus, $\Theta \subset \Upsilon$. All constraints of $\Upsilon$ are affine constraints, so $\Upsilon$ is a convex set. Note that all matrices in this set has a rank of at least $p$ because of the second constraint (i.e., $M_{1,1} = I$).

Now, the original problem of searching for $\{C_i\}$ and $\{D_j\}$ that satisfy (11) is transformed into a new problem of searching for a matrix $M \in \Upsilon$ that has a rank of $p$. To solve this new problem, a rank minimization problem is formulated whereby the optimal solution is a $p$-rank matrix that satisfies all appropriate constraints. Formally, the rank minimization problem is

$$\begin{align*}
\text{minimize} & \quad \text{rank}(G) \\
\text{subject to} & \quad G \in \Upsilon.
\end{align*}$$

(13)

If $M$, as defined in (12), exists (i.e., there exist $\{C_i\} \in \Omega$ and $\{D_j\} \in \Phi$ that satisfy (11)), the optimal solution is $M$ because $M$ is a matrix with the smallest rank in $\Upsilon$.

### 2.6 Nuclear Norm Formulation

A rank minimization problem is in general not a tractable optimization problem. Thus, to solve for the optimal matrix, $G^*$, of (13), we further relax the optimization problem using nuclear norm approximation. Mathematically, we relax (13) into

$$\begin{align*}
\text{minimize} & \quad ||G||_* \\
\text{subject to} & \quad G \in \Upsilon.
\end{align*}$$

(14)

This is a convex optimization problem where the objective function is a convex function and the constraints are affine constraints. Although (14) is not the same as (13) or (11), it is a tractable approximation technique that might return $\{C_i\} \in \Omega$ and $\{D_j\} \in \Phi$ that satisfy (11).

In the next sections, numerical simulations are performed to determine the performance of this formulation.

### 3 Numerical Simulations

This section illustrates the localized state feedback controller synthesis method described in Section 2 using specific examples. The numerical simulation is ran by the CVX package in MATLAB.

#### 3.1 Plant Description

Figure 1 shows the distributed plant model that is used in the simulations. The distributed plant is a chain of nine connected nodes (local plants). Every node has a sensor that directly measures its own state, but only five out of the nine nodes have a controller and an actuator each that can directly modifies its respective state. The sensing delay and actuating delay are set to be one
time step. In addition, the disturbance propagates from one node to its two neighboring nodes at each time step. The communication network is twice as fast as the physical plant network, so the measurement information propagates for two nodes at each time step. The communication network defines the structure of the controller $K$. The plant dynamics $(A, B)$ used in the simulations is given by

$$A = \begin{bmatrix} 0.50 & 0.45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.45 & 0.50 & 0.45 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.45 & 0.50 & 0.45 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.45 & 0.50 & 0.45 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.45 & 0.50 & 0.45 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.45 & 0.50 & 0.45 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.45 & 0.50 & 0.45 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.45 & 0.50 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.45 \\ \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$  

This plant is chosen to be marginally stable, which means that the maximum absolute eigenvalue of $A$ is closed to 1. In addition, the plant $(A, B)$ is controllable.

### 3.2 Simulation Parameters

Three parameters are adjusted in the simulations. The first parameter, $C_n$, defines the scope of the control region of each controller node. The closed loop response of the disturbance at each node is confined within the control region. Likewise, the controller at each node only use the measurement
data from its own localized region. The second parameter, \( m \), defines the length of the closed loop impulse response (i.e., \( N_d = m \)). The third parameter, \( N_c \), defines the length of the controller impulse response at each node.

In the simulations, the disturbance is applied at the first node, \( P1 \). The communication network catches up with the plant propagation at the fifth node. Thus, the minimum possible value of \( C_n \) is five.

### 3.3 Performance Metric

To measure the performance of the algorithm, the 2-norm of the differences between the ideal closed loop impulse response transfer function and the computed closed loop impulse response transfer function is considered.

After solving the nuclear norm minimization problem, the singular value decomposition is used to decompose the optimal solution \( M \), such that \( M = USV^T \).\(^1\) Keeping the largest \( p \) singular values of \( S \) and replacing all the other singular values by 0, a new matrix \( S' \) can be constructed. Let \( M' = US'V^T \). The matrix \( M' \) has rank exactly \( p \), but the anti-diagonal constraint might not be satisfied anymore. However, if the removed singular values are small, the sum of the anti-diagonal of \( M' \) are still small.

From \( M' \), \( \{C^*_i\} \) and \( \{D^*_i\} \) can be derived based on (12). Define the ideal closed loop impulse response transfer function as

\[
CL_i = \left( I + \sum_{j=1}^{N_d} \frac{1}{z^j} D^*_j \right)
\]  

and the computed closed loop impulse response transfer function as

\[
CL_c = \left( I - \frac{1}{z} A - \frac{1}{z} \sum_{i=1}^{N_c} \frac{1}{z^i} C^*_i \right)^{-1}.
\]  

The performance measure is defined as \( ||CL_i - CL_c||_2 \).

Stability of the plant \( A \) affects the performance of the algorithm (refer to Section 4.2 for more details). When \( A \) is substituted by \( \alpha A \) for some \( \alpha \) between 0 and 1, the performance improves. Thus, one can possibly obtain a better performance by solving (14), using \( \alpha A \) instead of \( A \) and rescale the solutions accordingly to obtain the solutions for the original \( A \). In general, there is no simple rule to determine the value of scaling factor \( \alpha \) that generates the best performance. Thus, grid search is applied for some range of \( \alpha \) to find the approximated optimal \( \alpha \).

\(^1\)Columns of \( U \) are the left singular vectors of \( M \). \( S \) is a diagonal matrix consisting of singular values of \( M \). Columns of \( V \) are the right singular vectors of \( M \).
4 Simulation Results and Discussion

4.1 Changing Parameters

For fixed localized region, $C_n$, and closed loop impulse response time, $m$, the $\alpha$-curves for different values of controller length, $N_c$, are simulated.

![Graphs showing the $\alpha$-curves for different values of controller length, $N_c$.](image)

(a) $C_n = 5, m = 5$

(b) $C_n = 5, m = 6$

(c) $C_n = 5, m = 7$

(d) $C_n = 5, m = 8$

Figure 2: The $\alpha$-curves for different values of controller length, $N_c$, when localized region, $C_n = 5$. Error norm is $\|CL_t - CL_c\|_2$. 

9
Figure 3: The $\alpha$-curves for different values of controller length, $N_c$, when localized region, $C_n = 6$. Error norm is $\|CL_i - CL_c\|_2$. 

(a) $C_n = 6, m = 6$

(b) $C_n = 6, m = 7$

(c) $C_n = 6, m = 8$

(d) $C_n = 6, m = 9$
Based on Figure 2, 3, and 4, increasing the length of the controller, $N_c$, would improve the error performance (i.e., smaller value of $||CL_i - CL_c||_2$) in general. However, some results violate this rule because nuclear norm minimization (14) is an approximation to the rank minimization problem (13), and the rank minimization problem does not necessarily generate the exact solution to the original problem (11).
Figure 5: The error norm, $||CL_i - CL_c||_2$, for different localized regions, $C_n$. Closed loop length is equivalent to $N_d = m$. Controller length is equivalent to $N_c$.

Figure 5 shows plots of error norm, $||CL_i - CL_c||_2$, when $C_n$ and $m$ are varied after choosing the best $\alpha$ for each case. In general, increasing $N_c$ or $N_d$ results in smaller error norms, $||CL_i - CL_c||_2$. However, this observation is not always true for the same reason described earlier for Figure 2, 3, and 4.

4.2 Changing Stability

Controlling a plant that is not just marginally stable is easier than controlling a marginally stable plant. Therefore, the optimizations generate better results. To demonstrate this fact, the simulation results of using plant $\beta A$ are compared with the simulation results of $A$ for $\beta = 0.5, 0.75$. 
Figure 6: The error norm, $||CL_i - CL_c||_2$, for $A$’s with different eigenvalues when $C_n = 5$, $m = 6$, and $n = 9$.

Figure 6 shows that the error norm of 0.75$A$ is only $\frac{1}{9}$ of the error norm of $A$. Moreover, the error norm of 0.5$A$ is $\frac{1}{15}$ of the error norm of 0.75$A$. This results support the usage of scaling factor $\alpha$ to find a better solution to a given plant.

4.3 Fixed Stability Margin and Random A

Plant dynamic, $A$, satisfying a specified structure is generated randomly to determine the general trend of error norm, $||CL_i - CL_c||_2$, using the nuclear norm formulation. The maximum absolute value of the eigenvalue of $A$ is scaled to be 0.99 for all $A$. Twenty random plants (i.e., $A$) are generated, and ten $\alpha$’s are randomly chosen from 0.5 to 0.7 for each $A$. The parameters of the simulation are chosen to be $C_n = 6$, $m = 7$, and $N_c = 6$. 
Figure 7: The error norm, $||CL_i - CL_c||_2$, for randomly generated $A$’s using different values of $\alpha$.

Based on Figure 7, the error norms differ a lot for each case. One of the trials generates an unstable computed closed loop impulse response, and thus the error norm becomes infinity. The $A$ matrix of this trial is shown below:

$$A = \begin{bmatrix}
0.2911 & -0.3307 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1.4721 & 1.6060 & 0.7285 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0.2007 & -1.1969 & 1.7368 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -0.2483 & 0.1422 & 1.4618 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -0.2725 & -0.0684 & -0.2048 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -0.4291 & -0.3672 & 0.4734 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0.4787 & 0.3601 & -0.2213 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -0.4378 & -0.1988 & 0.0305 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.7978 & 0.4156 & 0
\end{bmatrix}$$

Although this $A$ is marginally stable, the positive feedbacks between node 1 - node 2, and node 3 - node 4 make it hard to control the plant. One way to generate a stable closed loop response is by searching for larger values of $C_n$, $m$, and $N_c$. For example, $C_n = 7$, $m = 13$, $n = 11$, and $\alpha = 0.43$ has an error norm, $||CL_i - CL_c||_2 = 3.94$. 

14
4.4 Comparison with $H_2$ controller

The performance of the synthesized localized finite impulse response (FIR) controller is compared with the $H_2$ optimal infinite impulse response (IIR) controller [1]. The parameter for the localized FIR controller is chosen to be $C_n = 7$, $m = 8$, $N_c = 9$, and $\alpha = 0.775$.

During the $H_2$ controller synthesis, only state deviation is penalized in the objective function. Therefore, the synthesized IIR controller results in the best stabilizing state trajectories. Figure 8 shows that the trajectories of the states using our localized FIR controller is closed to that of using the $H_2$ optimal IIR controller. One difference is that the FIR controller takes one more step to eliminate the effects of an impulse disturbance. This result is reasonable because our FIR controller only use measurement data from a localized region. Figure 8 also suggests that a smaller
control effort is required to achieve the similar state trajectories to state trajectories generated by $H_2$ optimal controller.

5 Conclusion

In this project, a nuclear norm minimization is formulated to synthesize a decentralized local state feedback controller for distributed systems. This method shows some promising results when it is tested in simulations. However, further research is required to determine the technical conditions on the plant dynamics and sparsity constraints to guarantee an exact or “close” solution (i.e., a controller) when this method is implemented. In addition, other approach such as alternating projection method can be considered to solve for a constrained fixed rank matrix apart from nuclear norm approximation.

References


clear all

Plant parameters

globalVar();
p = 9; % Number of states of the given plant model
un = 5; % number of controller in the distributed system

%%Plant and sparsity pattern
SparArr = eye(p);
SparArr(1:end-1,2:end)=SparArr(1:end-1,2:end)+eye(p-1);
SparArr(2:end,1:end-1)=SparArr(2:end,1:end-1)+eye(p-1);

% Fixed A matrix
A=0.5*eye(p);
A(1:end-1,2:end)=A(1:end-1,2:end)+0.45*eye(p-1);
A(2:end,1:end-1)=A(2:end,1:end-1)-0.45*eye(p-1);

B3{1} = eye(p);
B3{1} = B3{1}(:,1:2:end);
B = B3{1};
%rank([B A*B])=p

Simulation

Sensing+Actuation delay = 2 Plant propagation speed = 1, Communication speed = 2

alpha_all = 0.45:0.025:0.80;
Err = zeros(3,4,4,length(alpha_all));

for w=1:1:3
    for x=1:1:4
        for y=1:1:4
            cn = 5+w-1; % Control region, the disturbance is confined in cn nodes
            m = cn+x-1; % number of B(i) matrices (CL time FIR)
            n = m+y-1; % number of A(i) matrices (controller time FIR + 2)

            % Sparsity for controller
            SparA = cell(ceil((cn+1)/2),1);
            SparA{1} = eye(p);
            SparA{1} = SparA{1}(1:2:end,:);
            for ii = 2:ceil((cn+1)/2)
                SparA{ii} = SparArr^(2*(ii-1));
                SparA{ii} = SparA{ii}(1:2:end,:);
            end

            % Sparsity for close loop response
            SparCL = cell(cn,1);
            SparCL{1} = eye(p);
            for ii = 2:cn

        end
    end
end

SparCL{ii} = SparArr^{ii-1};
end

SpArow = zeros(p,m*p);
SpAcol = zeros(n*p,p);
SpAcol(1:p,:) = eye(p);
SpArow(:,1:p) = eye(p);
SpAcol(p+1:2*p,:) = (A~=0);
SpB = (B~=0);
for ii = 1:n-2
    if ii < ceil((cn+1)/2)
        SpAcol(p*(ii+1)+1:(ii+2)*p,:) = SpB*SparA{ii};
    else
        SpAcol(p*(ii+1)+1:(ii+2)*p,:) = SpB*SparA{end};
    end
end
for ii = 2:m
    if ii < cn
        SpArow(:,p*(ii-1)+1:ii*p) = SparCL{ii};
    else
        SpArow(:,p*(ii-1)+1:ii*p) = SparCL{end};
    end
end
SpA = SpAcol*SpArow;

% Compute matrices with smallest nuclear norm
for z=1:1:length(alpha_all)
    alpha = alpha_all(z);
    [MM,K,ncNorm] = nuclear_norm(A,B,SparA,SpA,alpha);

    [U,S,V]=svd(MM);
    Mod_M = U(:,1:p)*S(1:p,1:p)*V(:,1:p)';
    Mod_M(:,1:p)=Mod_M(:,1:p).*(SpAcol==0);
    Mod_M(1:p,:)=Mod_M(1:p,:).*(SpArow==0);

del = ss(0,1,1,0,-1);
CL = eye(p);
Control = eye(p) - del*A;
for i=3:1:n
    Control = Control + del^{i-1}*Mod_M((i-1)*p+1:i*p,1:p)/alpha^{(i-1)};
end
for j=2:1:m
    CL = CL + del^{j-1}*Mod_M(1:p,(j-1)*p+1:j*p)/alpha^{(j-1)};
end
Iv = inv(Control);
Err(w,x,y,z) = norm(Iv-CL);
save Data.mat
savefile = 'Perf.mat';
save(savefile, 'Err')
end
end
end

Published with MATLAB® R2013a
% globalVar.m
% List of global variables

global p  % number of states
global n  % number of A(i) matrices (controller time FIR)
global m  % number of B(i) matrices (CL time FIR)
global cn  % controller state FIR
global un  % number of controller

Published with MATLAB® R2013a
% NUCLEAR_NORM finds a matrix with the smallest nuclear norm given some
% structural constraints.

% Inputs:
% A, B - plant
% SparA - sparsity structure for controller K
% SpA - sparsity structure for augmented matrix
% aplha - scaling factor

% Outputs:
% MM - a matrix with the smallest nuclear norm
% K - controller
% ncNorm - nuclear norm of MM

function [MM,K,ncNorm] = nuclear_norm(A,B,SparA,SpA,alpha)

globalVar();

cvx_begin
quiet
variables MM(n*p,m*p) K((n-2)*un,p)
expressions MAug(n*p,(n+m-1)*p)
for ii = 1:n
    MAug(p*(ii-1)+1:ii*p,p*(ii-1)+1:p*(ii+m-1)) = MM(p*(ii-1)+1:ii*p,:);
end

minimize norm_nuc(MM)
subject to

    MM(1:p,1:p) == eye(p);
    MM(p+1:2*p,1:p) == -alpha*A;
    MM(p+1:2*p,p+1:2*p) == -alpha^2*A^2;

    for ii = 1:n-2
        if ii < ceil((cn+1)/2)
            K(un*(ii-1)+1:ii*un,:).*(SparA{ii}==0)==0;
        else
            K(un*(ii-1)+1:ii*un,:).*(SparA{end}==0)==0;
        end
        MM(p*(ii+1)+1:(ii+2)*p,1:p)==B*K(un*(ii-1)+1:ii*un,1:p);
    end

    % constraints for sum of diagonal = 0
    for jj = 1:p
        sum(MAug(jj:p:end,p+1:end)) == zeros(1,(n+m-2)*p);
    end

    % constraints for matrix structure
    MM.*(SpA==0)==0;

    cvx_end
K = -K;
ncNorm = sum(svd(MM));