Two updating schemes of iterative learning control for networked control systems with random data dropouts

Dong Shen*, Chao Zhang, Yun Xu

College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, PR China

A R T I C L E   I N F O
Article history:
Received 9 April 2016
Revised 19 October 2016
Accepted 26 November 2016
Available online 2 December 2016

Keywords:
Iterative learning control
Data dropout
Intermittent updating scheme
Successive updating scheme
Networked control system

A B S T R A C T
The iterative learning control (ILC) problem is addressed in this paper for stochastic linear systems with random data dropout modeled by a Bernoulli random variable. Both intermittent updating scheme and successive updating scheme are provided on the basis of the available tracking information only and shown to be convergent to the desired input almost certainly. In the intermittent updating scheme, the algorithm only updates its control signal when data is successfully transmitted. In the successive updating scheme, the algorithm continuously updates its control signal with the latest available data in each iteration whether the output information of the last iteration is successfully transmitted or lost. Illustrative simulations verify the convergence and effectiveness of the proposed algorithms.

© 2016 Elsevier Inc. All rights reserved.

1. Introduction

Iterative learning control (ILC) is an important branch of intelligent control, especially for repetitive systems. Since first proposed by Arimoto in 1984 [5], ILC has been developed for three decades and many excellent achievements have been reported [2,6,31]. The inherent idea of ILC is to generate the input signal for the current iteration by using the input and output information from previous iterations as well as the desired trajectory. Thus, tracking performance is successively improved along the iteration axis, unlike in the traditional control strategies that improve control performance along the time axis [6]. The system has to complete the given tracking task in a finite time interval and then repeat it again and again because of this operation mechanism. Examples of these systems are chemical processes, robotics, and hard disk drives, to name a few.

Many studies have been conducted on different ILC topics, such as update law design [19], robustness [17,20,37], frequency analysis [43], and application research [7,22,41]. Moreover, the exploration of ILC has been extended to new problems, such as multi-phase processes [38], varying tasks [44], iteration varying lengths [23,35,36], event-triggered control [40], control with quantization information [11,34], collaborative tracking [16], and initial state vibration [39]. However, most of these studies mainly concern ILC in terms of centralized control systems, the controller and plant of which are placed together, so that each piece of information can be well received and processed. Recently, networked control systems (NCSs) have been widely used because of their facility, flexibility, and robustness with the help of the fast developments of

* This work is supported by National Natural Science Foundation of China (61673045, 61304085) and Beijing Natural Science Foundation (4152040).
* Corresponding author.
E-mail address: shendong@mail.buct.edu.cn (D. Shen).

http://dx.doi.org/10.1016/j.ins.2016.11.022
0020-0255/© 2016 Elsevier Inc. All rights reserved.
communication and network techniques. In this kind of system implementation, data transmission is a critical topic as data dropouts damage the tracking performance. This situation motivates the research on ILC for NCSs.

Other papers have also focused on ILC for a class of networked systems called multi-agent systems [21,26–28]. However, the major difference between these studies and ours is the description of a networked system. In [21,26–28], the multi-agent is a networked complex system combined by multiple subsystems, in which a network denotes the topological relationship among subsystems. The main problem is how to achieve synchronization and/or consensus. Conversely, in the field of NCS, a network means the transmission channel between the plant and the controller, and the main problem is how to guarantee good performance under severe transmission conditions, such as data dropouts.

When considering the ILC for NCSs, three major aspects should be taken into account, namely, data dropout condition, compensation mechanism design, and convergence analysis. In existing papers, some progresses were obtained according to one or more of these aspects. However, this state is still far from the topic being complete. In what follows, we present in brief the existing results on the data dropout condition and demonstrate our motivations and contributions.

Early attempts were first conducted by Ahn et al. [13,4] and succeeded by Bu and his co-workers [9,10,12]. In these studies, data dropout is modeled by a Bernoulli random variable, which has a value of 1 when data are successfully transmitted and 0 otherwise. As a result of such a condition, data dropouts can occur randomly and successively. Thus, the Bernoulli model has been widely used in many papers to describe random data dropouts. On the basis of the statistics of random data dropout, Ahn et al. showed mean square stability following the Kalman filtering-based design and analysis techniques that were first proposed by Saab in [30]. The major differences among these papers were the locations of the data dropouts. Specifically, only measurement loss was discussed in [13], and the case in which data dropouts occurred in the control and in the output was addressed in [4]. Bu et al. addressed the data dropout problem from the mathematical expectation perspective in [9,10,12]. Generally, mathematical expectations are taken to both sides of the iteration recursive equation of tracking errors directly in [9] for the linear system, and then the stability condition is given to derive the convergence of the expectation of tracking error. The corresponding nonlinear case is expressed in [12]. This technique is also used in [25], in which the stochastic equations were first converted into deterministic ones by taking mathematical expectation and the following analysis was conducted by the conventional contraction mapping method. Moreover, [10] provided an H-infinity ILC analysis for a discrete time system with random data dropouts, where the H-infinity performance is defined on the basis of the mathematical expectations of the random variables. In addition, Bu and his co-workers provided a 2D analysis approach for ILC under data dropouts, and the learning gain was generated using the linear matrix inequality (LMI) technique [8]. The mean square asymptotic stability was established.

Several observations of these results are listed as follows. First, conducting a rigorous analysis directly according to the generated sequence itself is difficult because of the inherent randomness. Therefore, the above works employed mathematical expectation, such as [9,10,12,25], or covariance, such as [13,4,8], to remove the random data dropout effect. Moreover, the probability of a data dropout should be known prior to designing the learning gain matrix because of the introduction of expectation and/or covariance. This condition limits the application range of the proposed algorithms as the statistics of data dropout is commonly unknown prior. Furthermore, the algorithms given in [1,3,4,8–10,25] all adopt the intermittent strategy to deal with the random data dropouts, i.e., the algorithms simply stop updating whenever data are lost during transmission. If the tracking information is available, the algorithms will use this information for updating; if the tracking information is lost, the algorithms will not update until the arrival of the new data. In short, the general Bernoulli data dropout condition still faces many open problems from the perspective of compensation mechanism design and convergence analysis.

To further extend the view in the compensation mechanism design, some limitations are imposed on the data dropout condition as in [12,18,24,29]. In practice, if the corresponding information is lost, considering whether or not one can compensate for this information will be of interest. These studies identify two types of compensation mechanisms. The first type is the time axis-based compensation method used in [12,29]. If the data at time t are lost, then the data at time t − 1 in the same iteration will be used to compensate for the lost data. Then, this compensation prevents the data at adjacent time instances from dropping. In [12], the expectation of the output converging to the desired reference under a condition depending on the probability of successful transmission was proved. The paper [29] mainly discussed the effect in the case in which a packet at a specified sampling time was lost. As the authors indicated, producing mathematical proof was difficult for the general multiple packet loss case. Thus, they had to assume that the rate of data dropout was far less than 100%. This conclusion demonstrated another limitation of data dropout. The second type is the iteration axis-based compensation mechanism [18,24]. If the data at kth iteration are lost, then the data at the same time instant but at the k − 1th iteration will be used to compensate for the lost data. At the same time, the data from adjacent iterations will not be dropped, a situation indicating the limitation of the data dropout condition. This inherent mechanism guarantees the convergence property of the proposed algorithms in [18,24].

In sum, the papers [12,18,24,29] provided primary compensation mechanisms for dropped data instead of setting them to zero. These mechanisms show the possibility of compensating for lost data, but these studies failed to address the random successive data dropout case along the time axis and/or iteration axis. Generally, these additional limitations imply that the data dropout is not completely stochastic and remains blank for the general random multiple data dropout case. Under a general data dropout environment, considering a generic algorithm that updates itself with the latest available data unknown prior to the learning process is of great interest.

In addition, a new model of random data dropouts was given in [32,33], in which a stochastic sequence was introduced to model the data dropouts with bounded length requirements. That is, for any given time instant, an unknown maximum
of successive iteration number exists that data could be dropped consecutively. As a result, this bounded length requirement of successive dropouts is somewhat tight as it is not completely stochastic.

Based on the above literature review, the motivation of this paper is that a Bernoulli random variable is the most common model for random data dropout. For the generic Bernoulli model, existing works focus on the intermittent strategy and the proposed algorithms converge in the mathematical expectation sense or mean square sense. A few papers have considered the compensation of lost data or direct convergence analysis to the generated sequence, but restrictive conditions are imposed on data dropout environments. Thus, in this paper, we are motivated to study the compensation scheme and convergence analysis in an almost sure sense under the generic Bernoulli model.

Specifically, we consider the stochastic linear system with both system and measurement noises under random data dropouts. The random data dropout is modeled by a Bernoulli random variable and no further condition is imposed. The main contributions of this paper are twofold. The first one is that two updating schemes, namely, intermittent updating scheme (IUS) and successive updating scheme (SUS), are proposed, analyzed, and compared. Moreover, the specific probability is not required beforehand in these schemes. In the IUS, the algorithm updates its control signal only when the corresponding packet is successfully transmitted. In the SUS, the algorithm keeps updating with the latest available data whether or not data dropouts occur. The other contribution is the almost sure convergence of the proposed algorithms under the Bernoulli model of data dropouts. Specifically, the input sequences generated by both algorithms are proved to converge to the desired input even if stochastic noises are involved. Note that the SUS and its associated convergence analysis is novel, and they have not been addressed in existing papers because of the newly involved randomness of successive dropouts.

The rest of the paper is organized as follows. The problem formulation, including system setup, control objective, and preliminary lemma, is presented in Section 2. The intermittent updating scheme and successive updating scheme with their convergence analysis are provided in Sections 3 and 4, respectively. Illustrative simulations are given in Section 5 to verify the theoretical results and Section 6 concludes the paper.

Notations: \( \mathbb{R} \) is the real number field, and \( \mathbb{R}^n \) is the \( n \)-dimensional real space. \( \mathbb{N} \) is the set of all positive integers. \( I_{n \times n} \) is the \( n \)-dimensional identity matrix. \( P \) is the probability of an event and \( \mathbb{E} \) is the mathematical expectation. \( \| \cdot \|_2 \) is the Euclidean norm or 2-norm of a vector or a matrix. For a concise expression, the subscript \( 2 \) is omitted and the norm notation is abbreviated as \( \| \cdot \| \) in the rest of the paper. The superscript \( T \) is the transpose of a matrix or vector. For two sequences \( \{ a_n \} \) and \( \{ b_n \} \), we call \( a_n = O(b_n) \) if \( b_n \to 0 \) and there exists \( L > 0 \) such that \( |a_n| \leq Lb_n \) for all \( n \). \( a_n = o(b_n) \) if \( b_n \to 0 \) and \( a_n/b_n \to 0 \) as \( n \to \infty \).

The abbreviations "i.o." and "a.s." denote "infinitely often" and "almost surely", respectively.

2. Problem formulation

Consider the following discrete stochastic system:

\[
\begin{align*}
x_k(t+1) &= A(t)x_k(t) + B(t)u_k(t) + w_k(t+1) \\
y_k(t) &= C(t)x_k(t) + v_k(t)
\end{align*}
\]

where \( k = 1, 2, \ldots, \) is the different iteration numbers, \( t = 0, 1, \ldots, N \) is the different time instances in an iteration, and \( N \) is the length of each iteration. \( x_k(t) \in \mathbb{R}^n, u_k(t) \in \mathbb{R}^p, \) and \( y_k(t) \in \mathbb{R}^q \) are the state, input, and output of the system, respectively. \( A(t), B(t), \) and \( C(t) \) are system matrices with appropriate dimensions. The random variables \( w_k(t) \) and \( v_k(t) \) are system noises and measurement noises, respectively.

Let \( y_d(t), t = 0, 1, \ldots, N \) be the tracking reference. The following mild assumptions are given for the system (1).

A1. The input-output coupling matrix \( C(t+1)B(t) \) is assumed to have a full-column rank for all \( t \).

A2. For each \( t \), the independent and identically distributed (i.i.d.) sequence \( \{ w_k(t), k = 0, 1, \ldots \} \) is independent of the i.i.d. sequence \( \{ v_k(t), k = 0, 1, \ldots \} \) with \( \mathbb{E}w_k(t) = 0, \mathbb{E}v_k(t) = 0, \sup_{k \geq 1} \mathbb{E} \| w_k(t) \| ^2 < \infty, \sup_{k \geq 1} \mathbb{E} \| v_k(t) \| ^2 < \infty, \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} w_k(t) w_k^T(t) = R_w^c, \) and \( \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} v_k(t) v_k^T(t) = R_v^c \) a.s., where \( R_w^c \) and \( R_v^c \) are unknown matrices.

A3. The initial state sequence \( \{ x_k(0) \} \) is i.i.d. with \( \mathbb{E}x_k(0) = x_d(0), \sup_{k \geq 1} \mathbb{E} \| x_k(0) \| ^2 < \infty, \) and \( \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} x_k(0)x_k^T(0) = R_0. \) Further, the sequences \( \{ x_k(0), k = 0, 1, \ldots \}, \{ w_k(t), k = 0, 1, \ldots \}, \) and \( \{ v_k(t), k = 0, 1, \ldots \} \) are mutually independent.

Remark 1. For any given initial value \( x_d(0) \), the following expression of \( u_d(t) \) can be computed recursively from the nominal model,

\[
u_d(t) = [(C^+B(t))(C^+B(t))]^{-1}(C^+B(t))^T(y_d(t+1) - C(t + 1)A(t)x_d(t))
\]

where \( A1 \) is used and \( C^+B(t) \) \( \equiv C(t+1)B(t) \in \mathbb{R}^{q \times p} \). Evidently, the following equations are fulfilled:

\[
\begin{align*}
x_d(t+1) &= A(t)x_d(t) + B(t)u_d(t) \\
y_d(t) &= C(t)x_d(t)
\end{align*}
\]

Moreover, \( A1 \) further implies that the relative degree is one and the dimension of input is not larger than the dimension of output, i.e., \( p \leq q \). \( A1 \) is used to guarantee the existence of the desired control that generates the desired tracking reference.
from the nominal model. \( A2 \) is a common condition in unknown random noises in stochastic control. The independence condition in \( A2 \) is required along the iteration axis, and thus it is rational for practical applications because the process is repeatable. The initial resetting condition \( A3 \) enables a random initial shift around the desired initial state. The classical precise resetting condition of the initial state can be regarded as a special case of \( A3 \). To facilitate the expression, denote \( w_k(0) = x_k(0) - x_d(0) \). Then, defining \( \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} w_k(0)w_k^T(0) = R_0 \) is easy to satisfy the formulation of \( A2 \). In other words, \( A3 \) can be compressed into \( A2 \). Thus, all the assumptions are mild.

The setup of the control system is illustrated in Fig. 1, where the plant and learning controller are located separately and communicate via networks. The data may be dropped out through the networks because of network congestion, link-age interruption, and transmission error. However, to make the expression concise and without loss of any generality, data dropout is only considered for the output side. That is, random data dropouts only occur in the network from the measurement output to the buffer, and the network from the learning controller to the control plant is assumed to work well.

Similar to [1,3,4,9,12], we adopt a Bernoulli random variable to model the random data dropouts. Specifically, a random variable \( y_k(t) \) is introduced to indicate whether or not the measurement packet \( y_k(t) \) is successfully transmitted,

\[
    y_k(t) = \begin{cases} 
        1, & \text{if } y_k(t) \text{ is successfully transmitted} \\
        0, & \text{otherwise}
    \end{cases}
\]

and without loss of any generality,

\[
    P(y_k(t) = 1) = \rho, \quad P(y_k(t) = 0) = 1 - \rho
\]

where \( 0 < \rho < 1 \). That is, the probability that the measurement \( y_k(t) \) is successfully transmitted is \( \rho \) for all \( t \) and \( k \).

**Remark 2.** The Bernoulli random variable has been used in many publications to describe random data dropouts. In the field of ILC, papers [1,3,4,9,12] are typical illustrations. However, in most cases, only convergence in the mathematical expectation sense and/or mean square sense is obtained. No result on convergence with probability 1 (w.p.1) has been reported until now. In our early papers [32,33], the convergence w.p.1 of ILC input sequence is proved strictly for stochastic systems. However, data dropout in previous studies is modeled by a stochastic sequence with a finite length requirement. A finite length requirement means that data dropout is not completely stochastic. Therefore, the Bernoulli random variable model is revisited in this paper and the convergence w.p.1 is expected to be obtained.

If the measurement packet is successfully transmitted, then one can compare it with the desired reference value and compute the tracking error \( e_k(t) = y_d(t) - y_k(t) \) for the updating. Otherwise, no output information is received, and no tracking error can be obtained for further updating.

The conventional control objective for a deterministic system is to build an ILC algorithm that generates the input sequence, so that the actual output of the system \( y_k(t) \) can track some given trajectory \( y_d(t) \) asymptotically. However, when considering stochastic systems, system noises and measurement noises, which cannot be predicted and eliminated by any algorithm, are observed. Thus, we cannot expect that \( y_k(t) \to y_d(t) \), \( \forall t \), for stochastic systems, as the iteration number \( k \) goes to infinity. Therefore, for stochastic systems, the best achievable tracking performance is that the tracking error only consists of the noise terms. To this end, the control objective of this paper is to design an ILC algorithm such that \( u_k(t) \to u_d(t) \), \( \forall t \) as \( k \to \infty \).

**Remark 3.** As stochastic noises cannot be predicted and eliminated, an intuitive idea is to minimize the following averaged tracking error index:

\[
    V_t = \limsup_n \frac{1}{n} \sum_{k=1}^{n} \|y_d(t) - y_k(t)\|^2
\]

Through simple calculations, the above index is found to be minimized if \( u_k(t) \to u_d(t) \), \( \forall t \) as \( k \to \infty \). Thus, in what follows, we show the direct convergence of the input sequence to the desired input.
To achieve this control objective under stochastic noises, two updating schemes are proposed in this paper. The convergence analysis of these schemes is presented in the following two sections.

1. **IUS**
   
   \[ u_{k+1}(t) = u_k(t) + a_k \gamma_k(t + 1)L_te_k(t + 1) \]  
   
   where \( a_k \) is the learning step-size and \( L_t \) is the learning gain matrix.

2. **SUS**
   
   \[ u_{k+1}(t) = u_k(t) + a_kL_te^*_k(t + 1) \]  
   
   where \( a_k \) and \( L_t \) mean the same as those in the IUS case, and \( e^*_k(t) \) is the latest available tracking error defined as
   
   \[ e^*_k(t) = \begin{cases} e_k(t), & \text{if } \gamma_k(t) = 1 \\ e^*_{k-1}(t), & \text{if } \gamma_k(t) = 0 \end{cases} \]  

   The learning step-size \( \{a_k\} \) is a decreasing sequence, and it should satisfy the following:
   
   \[ a_k > 0, a_k \to 0, \sum_{k=1}^{\infty} a_k = \infty, \sum_{k=1}^{\infty} a_k^2 < \infty \]  

**Remark 4.** The decreasing sequence is added to the algorithms as the stochastic system is considered in the paper. Clearly, \( a_k = \alpha/k \) meets all the requirements in (9), where the constant \( \alpha > 0 \) can be regarded as a tuning parameter. This learning step-size \( a_k \) is introduced to suppress the effect of stochastic noises as the iteration number goes to infinity and to guarantee a zero-error convergence of the input sequence. Specifically, the tracking error contains stochastic noises as its part. After enough learning iteration, the stochastic noise dominates the tracking error. If no mechanism is available to suppress the effect of stochastic noises, the algorithm will fail to converge to a stable limitation. Therefore, we add the decreasing sequence to the conventional P-type learning algorithm.

**Remark 5.** Note that the proposed algorithms (6) and (7) are slight modified versions of the conventional P-type algorithm. In other words, this study attempts to provide the first rigorous convergence analysis for ILC under random data dropout that is modeled by a Bernoulli random variable. Thus, designing new ILC algorithms is not our principle objective. We believe that the conventional P-type can provide robustness against severe learning conditions. Accordingly, we adopt the common ILC algorithms. The introduction of the decreasing sequence \( a_k \) makes algorithms (6) and (7) classic stochastic approximation algorithms. Thus, our next analysis is based on stochastic approximation techniques.

**Remark 6.** The inherent mechanism of IUS is that the algorithm updates the input when the output is successfully received and stops updating otherwise. That is, the input can keep the latest one if no new output is received; this is why it is called IUS. In addition, the updating frequency is equal to the successful transmission rate. Thus, the learning step-size \( a_k \) goes to 0 fast when the data dropout rate is large and a slow learning speed is obtained. An alternative to (6) is given as follows to improve the performance:

\[ u_{k+1}(t) = u_k(t) + a_k\gamma_k(t + 1)L_te_k(t + 1) \]

\[ k_k(t) = \sum_{i=1}^{k} \gamma_i(t + 1) \]

**Remark 7.** The inherent mechanism of SUS is that the algorithm keeps updating by using the latest available packet. In other words, if the output of the last iteration is received, then the algorithm will update its input using this information. If the output of the last iteration is lost, then the algorithm will update its input using the latest available output packet received previously. That is, algorithm (7) can be rewritten as

\[ u_{k+1}(t) = u_k(t) + a_k\gamma_k(t + 1)L_te_k(t + 1) + a_k(1 - \gamma_k(t + 1))L_te^*_{k-1}(t + 1) \]

Note that no finite length condition in the successive data dropouts is required. In fact, the number of successive data dropout iteration can be arbitrarily large.

**Remark 8.** The differences between IUS and SUS are listed as follows. First, IUS is an event-triggered updating, whereas SUS is an iteration-triggered updating. That is, IUS only updates its signal when the measurement output of the last iteration is received, whereas SUS updates its signal in every iteration. Moreover, the updating frequency of IUS depends on the rate of successful transmission and is usually low. Generally, the larger the data dropout rate is, the lower the updating frequency is. Conversely, SUS keeps updating all the time. In sum, the respective convergence speeds of IUS and SUS are markedly different.
The following technical lemma is used for the subsequent analysis. Its proof is found in [13].

**Lemma 1.** Let $H$ be a stable matrix with dimension of $l \times l$. Let $a_k$ satisfy the conditions in (9) and let $l$-dimensional vectors $(\mu_k), (v_k)$ satisfy the following conditions:

$$
\sum_{k=1}^{\infty} a_k \mu_k < \infty, \quad v_k \underset{k \to \infty}{\longrightarrow} 0
$$

(11)

Then, $(\alpha_k)$ generated by the following recursion with an arbitrary initial value $\alpha_0$ converges to zero w.p.1

$$
\alpha_{k+1} = \alpha_k + a_k H \alpha_k + a_k (\mu_k + v_k)
$$

(12)

Here by stability of a matrix we mean that all its eigenvalues are with negative real parts.

### 3. Convergence analysis for the intermittent updating scheme

In this section, we present the technical convergence analysis of the IUS case. Compared with that of the SUS case, the proof of the IUS case is more intuitive as IUS will keep the input signal invariant if the corresponding output of the last iteration is lost. Thus, one only needs to focus on the updating iterations.

In observing the probability of (4), clearly, $E \gamma_k(t) = \rho$ and $E \gamma_k^2(t) = \rho$. Denote $\delta x_k(t) = x_d(t) - x_k(t)$ and $\delta u_k(t) = u_d(t) - u_k(t)$. Then, the update law (6) is rewritten as

$$
u_{k+1}(t) = u_k(t) + a_k \gamma_k(t + 1) L_v e_k(t + 1)
$$

$$= u_k(t) + a_k \gamma_k(t + 1) L_v (y_d(t + 1) - y_k(t + 1))
$$

$$= u_k(t) + a_k \gamma_k(t + 1) L_v C(t + 1) \delta x_k(t + 1)
$$

$$+ a_k \gamma_k(t + 1) L_v v_k(t + 1)
$$

$$= u_k(t) + a_k \gamma_k(t + 1) L_v C(t + 1) A(t) \delta x_k(t)
$$

$$+ a_k \gamma_k(t + 1) L_v v_k(t + 1)
$$

where $C^A(t) \triangleq C(t + 1) A(t)$.

Subtracting both sides of the last equation from $u_d(t)$ leads to

$$
\delta u_{k+1}(t) = \delta u_k(t) - a_k \rho L_v C^+ B(t) \delta u_k(t)
$$

$$- a_k \gamma_k(t + 1) - \rho) L_v C^+ B(t) \delta u_k(t)
$$

$$- a_k \gamma_k(t + 1) L_v C^+ A(t) \delta x_k(t)
$$

$$+ a_k \gamma_k(t + 1) L_v C(t + 1) w_k(t + 1)
$$

$$+ a_k \gamma_k(t + 1) L_v v_k(t + 1)
$$

(13)

In the following, the argument $t$ or its specific value may be omitted if no confusion exists to make the expressions concise.

**Theorem 1.** Consider the stochastic system (1) and update law (6). Design $L_v \in \mathbb{R}^{p \times q}$ such that all eigenvalues of $L_v C^+ B(t)$ are with positive real parts. Then, the input $u_k(t)$ generated by (6) converges to $u_d(t)$ w.p.1 as $k \to \infty$ for $\forall t$.

**Proof.** The proof is performed by mathematical induction along the time axis $t$. The steps for $t = 1, 2, \ldots, N - 1$ are identical to those in the case of $t = 0$, which will be expressed in detail in the following.

**Initial Step.** Consider the case $t = 0$.

For $t = 0$, (13) is

$$
\delta u_{k+1}(0) = \delta u_k(0) - a_k \rho L_v C^+ B(0) \delta u_k(0)
$$

$$- a_k \gamma_k(1) - \rho) L_v C^+ B(0) \delta u_k(0)
$$

$$- a_k \gamma_k(1) L_v C^+ A(0) \delta x_k(0)
$$

$$+ a_k \gamma_k(1) L_v C(1) w_k(1) + a_k \gamma_k(1) L_v v_k(1)
$$

(14)
Considering A3 and Remark 1, we find that $\delta x_i(0) = w_i(0)$. On the other side, $\gamma_k(1)$ is independent of $\delta x_i(0)$. Both $\{\gamma_k(1)\}$ and $\{\delta x_i(0)\}$ are i.i.d. sequences along the iteration axis with a finite second moment. In addition, $\mathbb{E}\delta x_i(0) = \mathbb{E}x_i(0) - x_i(0) = 0$. Therefore, if we denote $\epsilon_k \triangleq \gamma_k(1)L_0C^+A(0)\delta x_i(0)$, then $\{\epsilon_k\}$ is an i.i.d. sequence with zero mean and a finite second moment.

Through direct calculations, we have

$$
\sum_{k=1}^{\infty} \mathbb{E}(\epsilon_k^2) = \sum_{k=1}^{\infty} \mathbb{E}(a_k\gamma_k(1)L_0C^+A(0)\delta x_i(0))^2 \\
= \sum_{k=1}^{\infty} a_k^2 \cdot \mathbb{E}\gamma_k^2(1) \cdot L_0C^+A(0) \cdot \mathbb{E}(\delta x_i(0))^2 \\
\leq c_0 \sum_{k=1}^{\infty} a_k^2 < \infty
$$

where $c_0 > 0$ is a suitable constant. Then, we have $\sum_{k=1}^{\infty} a_k\gamma_k(1)L_0C^+A(0)\delta x_i(0) < \infty$ w.p.1 by the Khintchine–Kolmogorov convergence theorem [15].

Similarly, both $\{w_i(1)\}$ and $\{v_i(1)\}$ are i.i.d. sequences with zero means and finite second moments. In addition, they are independent of $\gamma_k(1)$. Clearly,

$$
\sum_{k=1}^{\infty} \mathbb{E}(a_k\gamma_k(1)L_0C(1)w_k(1))^2 \\
\leq \|L_0C(1)\|^2 \sum_{k=1}^{\infty} a_k^2 \cdot \mathbb{E}\gamma_k^2(1) \cdot \mathbb{E}w_k^2(1) < \infty
$$

which further leads to $\sum_{k=1}^{\infty} a_k\gamma_k(1)L_0C(1)w_k(1) < \infty$. $\sum_{k=1}^{\infty} \mathbb{E}a_k\gamma_k(1)L_0v_k(1) < \infty$ w.p.1.

Consider the third term on the right-hand side of (14), $a_k(\gamma_k(1) - \rho)L_0C^+B(0)\delta u_k(0)$. The sequence of this term is no longer mutually independent, unlike the last three terms of (14). To deal with this term, let $\mathcal{F}_k$ be the increasing $\sigma$-algebra generated by $y_j(t), x_j(t), w_j(t), v_j(t), \, 0 \leq t \leq N, \, 0 \leq j \leq k$, i.e., $\mathcal{F}_k \triangleq \sigma\{y_j(t), x_j(t), w_j(t), v_j(t), \, 0 \leq j \leq k, \, t \in [0, \ldots, N]\}$. According to the update law (6), $u_k(t) \in \mathcal{F}_{k-1}$. Note that $\gamma_k(1)$ is independent of $\mathcal{F}_{k-1}$ and is thus independent of $\delta u_k(0)$. Therefore,

$$
\mathbb{E}\{(\gamma_k(1) - \rho)L_0C^+B(0)\delta u_k(0)|\mathcal{F}_{k-1}\} = L_0C^+B(0)\delta u_k(0)\mathbb{E}\{\gamma_k(1) - \rho|\mathcal{F}_{k-1}\} = 0
$$

That is, $(a_k(\gamma_k(1) - \rho)L_0C^+B(0)\delta u_k(0), \mathcal{F}_k, k \geq 1)$ is a martingale difference sequence. In addition,

$$
\sum_{k=1}^{\infty} \mathbb{E}[\|a_k(\gamma_k(1) - \rho)L_0C^+B(0)\delta u_k(0)\|^2|\mathcal{F}_{k-1}] \\
\leq \sum_{k=1}^{\infty} a_k^2 \mathbb{E}[(\gamma_k(1) - \rho)^2]|\mathcal{F}_{k-1}] \\
\leq \sup_k \|L_0C^+B(0)\delta u_k(0)\|^2 \sum_{k=1}^{\infty} a_k^2 \mathbb{E}(\gamma_k(1) - \rho)^2 \\
\leq c_1 \sum_{k=1}^{\infty} a_k^2 < \infty
$$

where $c_1 > 0$ is a suitable constant. Then, by the Chow convergence theorem of martingale [15], we have $\sum_{k=1}^{\infty} a_k(\gamma_k(1) - \rho)L_0C^+B(0)\delta u_k(0) < \infty$ w.p.1.

If the learning gain matrix is designed such that all eigenvalues of $L_0C^+B(0)$ are with positive real parts, then $-\rho L_0C^+B(0)$ is clearly stable. Thus, applying Lemma 1 to the recursion (14), we have $\delta u_k(0) \to 0$ as $k \to \infty$ w.p.1.

**Inductive Step.** Assume that the convergence of $u_k(t)$ has been proved for $t = 0, 1, \ldots, s-1$ and that the aim is to show the convergence for $t = s$. 


From (1) and (2), we have
\[
\delta x_k(s) = A(s-1)\delta x_k(s-1) + B(s-1)\delta u_k(s-1) - w_k(s) \\
= A(s-1)A(s-2)\delta x_k(s-2) + A(s-1)B(s-2)\delta u_k(s-2) \\
+ B(s-1)\delta u_k(s-1) - A(s-1)w_k(s-1) - w_k(s) \\
= \sum_{i=0}^{s-1} \left( \prod_{j=i+1}^{s-1} A(j) \right) B(i)\delta u_k(i) - \sum_{i=0}^{s-1} \left( \prod_{j=i}^{s-1} A(j) \right) w_k(i)
\]
where \( \prod_{j=k}^{s-1} A(k) = A(j)A(j-1) \ldots A(i). \) \( \forall j \geq i \) and \( \prod_{j=0}^{s-1} A(k) = I. \) Replacing all \( t \) in (14) with \( s \) and substituting the above equation, we have
\[
\delta u_{k+1}(s) = \delta u_k(s) - a_k \rho L_t C^+ B(s)\delta u_k(s) \\
- a_k \gamma_k(s+1) - \rho) L_t C^+ B(s)\delta u_k(s) \\
- a_k \gamma_k(s+1)L_t C^+ A(s) \sum_{i=0}^{s-1} \left( \prod_{j=i+1}^{s-1} A(j) \right) B(i)\delta u_k(i) \\
- a_k \gamma_k(s+1)L_t C^+ A(s) \sum_{i=0}^{s-1} \left( \prod_{j=i}^{s-1} A(j) \right) w_k(i) \\
+ a_k \gamma_k(s+1)L_t C^+(s+1)w_k(s+1) + v_k(s+1)
\]
(15)

Through the induction assumption, we have that \( \delta u_k(t) \rightarrow 0 \) as \( k \rightarrow \infty \) w.p.1 for \( t = 0, 1, \ldots, s - 1. \) Thus, \( \gamma_k(s+1)L_t C^+ A(s) \sum_{i=0}^{s-1} \left( \prod_{j=i}^{s-1} A(j) \right) w_k(i) \rightarrow 0 \) w.p.1.

Following the same step as the initial step, we can obtain the following results:
\[
\sum_{k=1}^{\infty} a_k \gamma_k(s+1) - \rho) L_t C^+ B(s)\delta u_k(s) < \infty,
\]
\[
\sum_{k=1}^{\infty} a_k \gamma_k(s+1)L_t C^+ A(s) \sum_{i=0}^{s-1} \left( \prod_{j=i}^{s-1} A(j) \right) w_k(i) < \infty,
\]
\[
\sum_{k=1}^{\infty} a_k \gamma_k(s+1)L_t C(s+1)w_k(s+1) < \infty,
\]
\[
\sum_{k=1}^{\infty} a_k \gamma_k(s+1)L_t v_k(s+1) < \infty \text{ w.p.1.}
\]

Again, using Lemma 1, we can easily conclude that \( \delta u_k(s) \rightarrow 0 \) as \( k \rightarrow \infty. \) The proof is completed using the mathematical induction method. \( \square \)

**Remark 9.** As shown in Theorem 1, the condition on the design of the learning matrix \( L_t \) is relaxed, as it can be solved by finding a feasible solution for an LMI \( L_t C^+ B(t) > 0. \) If the system matrices \( C(t) \) and \( B(t) \) are known, then an intuitive selection of \( L_t \) is \( L_t = (C^+ B(t))^T, \) which leads \( L_t C^+ B(t) \) to become a positive definite matrix. In addition, this condition can be assured under some uncertainties of the system model.

**Remark 10.** The proof of Theorem 1 indicates that the condition in the initial state \( A3 \) can be replaced by the following: the initial state \( x_k(0) \) is independent of stochastic noises \( w_k(t) \) and \( v_k(t). \) Moreover, the deviation between \( x_k(0) \) and \( x_{d(0)} \) approaches zero, i.e., \( \delta x_k(0) \rightarrow 0 \) as \( k \rightarrow \infty. \) By incorporating some learning strategy of initial state such as [14], the applications can be further enlarged.

**Remark 11.** With slight modifications of the above proof, the convergence w.p.1 of the alternative algorithm proposed in Remark 6 is easily shown. If the learning step size \( a_k \) is eliminated from (6), i.e.,
\[
u_{k+1}(t) = u_k(t) + \gamma_k(t+1)L_t e_k(t+1)
\]
(16)
then the convergence w.p.1 can also be obtained as long as \( L_t \) satisfies that the spectral norm of \( I - \rho L_t C^+ B(t) \) is less than 1. The selection of \( L_t \) is more restrictive than the one given in Theorem 1. The tracking performance comparisons of the proposed algorithm (6) and the conventional P-type update law (16) are simulated in Section 5.
4. Convergence analysis for the successive updating scheme

This section presents the convergence analysis of the SUS case. The proof of this case is more technically complex than that of the IUS case, as the update information in (7) or (10) is no longer relatively definitive. In other words, if the measurement output of the last iteration is lost during transmission, then the one used in (7) will be unknown because of the possibility of successive data dropouts. Thus, update information can come from any previous iteration.

To form this situation, stochastic stopping times \( \{ \tau_k^f \}, k = 1, 2, \ldots, 0 \leq t \leq N \) are introduced to denote the random iteration-delays of the update caused by random data dropouts. Thus, the updating scheme (7) is rewritten as

\[
u_{k+1}(t) = u_k(t) + a_k L_f e_{k-t_{k}^f}(t + 1)
\]

(17)

where the stopping time \( \tau_k^f = k \). In other words, for the updating of input at time \( t \) of \( (k+1) \)th iteration, no information of \( e_m(t + 1) \) with \( m > k - \tau_k^f \) is received and only \( e_{k-t_{k}^f}(t + 1) \) is available. According to the SUS settings, for the iterations \( k - \tau_k^f < m \leq k \), the input \( u_m(t) \) is successively updated with the same error \( e_{k-t_{k}^f}(t + 1) \).

The coupling of stochastic stopping times and the successive updating mechanism make the convergence analysis much complex. Thus, the analysis will be proved by two steps: we first show the convergence of (17) with \( \tau_k^f = 0, \forall k, t \), and then we consider the effect of stopping times \( \tau_k^f \).

We first show the convergence of the following updating scheme

\[
u_{k+1}(t) = u_k(t) + a_k L_f e_{k-t_{k}^f}(t + 1)
\]

(18)

This case actually is the conventional ILC for systems without random data dropouts. We have the following theorem.

**Theorem 2.** Consider the stochastic system (1) and update law (18). Design \( L_f \in \mathbb{R}^{p \times q} \) such that all eigenvalues of \( L_f \) are with positive real parts. Then, the input \( u_k(t) \) generated by (18) converges to \( u_d(t) \) with \( p \) w.p.1 as \( k \) goes to infinity, \( \forall t \).

The proof is put in the appendix. Now it comes to the general case (17).

**Theorem 3.** Consider the stochastic system (1) and update law (17). Design \( L_f \in \mathbb{R}^{p \times q} \) such that all eigenvalues of \( L_f \) are with positive real parts. Then, the input \( u_k(t) \) generated by (18) converges to \( u_d(t) \) with \( p \) w.p.1 as \( k \) goes to infinity, \( \forall t \).

**Proof.** Comparing (17) and (18), the effect of the random data dropout is an additional error:

\[
a_k L_f(e_k(t) + e_{k-t_{k}^f}(t + 1))
\]

Thus the aim of the proof is to show that the above term satisfies the condition (11). Specifically, we have

\[

a_k L_f(e_k(t) + e_{k-t_{k}^f}(t + 1))
\]

\[

= a_k L_f C^a B(t) [\delta u_k(t) - \delta u_{k-t_{k}^f}(t) + \delta x_k(t) - \delta x_{k-t_{k}^f}(t)]
\]

\[

+ a_k L_f C^a(t) [\delta x_k(t) - \delta x_{k-t_{k}^f}(t)]
\]

\[

- a_k L_f C(t + 1) [w_k(t) + w_{k-t_{k}^f}(t)]
\]

\[

- a_k L_f [u_k(t) - u_{k-t_{k}^f}(t)]
\]

Undoubtedly, the last two terms satisfy condition (11). Similar to the proofs in Theorems 1 and 2, they can be proved by mathematical induction that the second term on the right-hand side of the last equation satisfies condition (11). Thus, only the first term, i.e., \( a_k L_f C^a B(t) [\delta u_k(t) - \delta u_{k-t_{k}^f}(t)] \), is left for further analysis.

Recalling update (17), the difference is expanded to

\[

\delta u_k(t) - \delta u_{k-t_{k}^f}(t) = \sum_{m=k-t_{k}^f}^{k-1} a_m L_f C^a B(t) [\delta u_m - \delta u_{m-t_{m}^f}(t)]
\]

(20)

\[

\sum_{m=k-t_{k}^f}^{k-1} a_m L_f C^a B(t) [\delta x_m - \delta x_{m-t_{m}^f}(t)]
\]

\[

\sum_{m=k-t_{k}^f}^{k-1} a_m L_f C(t + 1) [w_m - w_{m-t_{m}^f}(t + 1)]
\]

\[

\sum_{m=k-t_{k}^f}^{k-1} a_m L_f [v_m - v_{m-t_{m}^f}(t + 1)]
\]


To analyze the effect of (20), we need to estimate the number of successive data dropout iterations, i.e., $\tau_k^t$. As data dropouts are modeled by a Bernoulli random variable, $\tau_k^t$ obeys the geometric distribution. To make the notations concise, we let $\tau$ denote a random variable satisfying the same geometric distribution, i.e., $\tau \sim G(\rho)$. Clearly, $E\tau = \frac{1}{\rho}$ and $\text{Var}(\tau) = \frac{1-\rho}{\rho^2}$. As $\text{Var}(\tau) = E(\tau - E\tau)^2$, then $E\tau^2 = \frac{2-\rho}{\rho^2}$. Using direct calculations, we have
\[
\sum_{n=1}^{\infty} P\{\tau \geq n^2\} = \sum_{n=1}^{\infty} P\{\tau^2 \geq n\} \\
= \sum_{n=1}^{\infty} \sum_{j=n}^{\infty} P\{j \leq \tau^2 < j + 1\} \\
= \sum_{j=1}^{\infty} jP\{j \leq \tau^2 < j + 1\} \\
\leq E\tau^2 < \infty
\]
Using the Borel–Cantelli lemma, we have
\[P\{\tau \geq n^2 \text{ i.o.}\} = 0\]
Consequently, we have
\[\frac{\tau_k^t}{n} \to 0 \text{ a.s. } \forall t \tag{21}\]
Therefore,
\[\frac{n - \tau_k^t}{n} \to 1 \text{ a.s. } \forall t \tag{22}\]
and
\[n - \tau_n^t \to \infty \text{ a.s.} \tag{23}\]
Now, let us prove that the three terms on the right-hand side of (20) satisfy the condition (11) of Lemma 1.
Through the same steps of the proof of Theorem 1, $\sum_{m=0}^{k-1} a_m L^C(t+1) w_{m-t_m^{t+1}}(t+1) - v_{m-t_m^{t+1}}(t+1)$ converges to an unknown constant a.s., $\forall t$. Therefore, in view of (23), we have
\[
\left\| \sum_{m=k-t_k^{t+1}}^{k-1} a_m L^C(t+1) w_{m-t_m^{t+1}}(t+1) - v_{m-t_m^{t+1}}(t+1) \right\| = o(1)
\]
Therefore, the last term of (20) satisfies condition (11) of Lemma 1.
Again, the same as the steps of the proof in Theorem 1, the second term on the right-hand side of (20) can be split into two parts: a finite summation of input error of past time instances and a finite summation of stochastic noises.
\[
\sum_{m=k-t_k^{t+1}}^{k-1} a_m L^C(t) \delta m-t_m^{t+1}(t) \\
= \sum_{m=k-t_k^{t+1}}^{k-1} a_m L^C(t) \sum_{i=0}^{t-1} \left( \prod_{j=i+1}^{t-1} A(j) \right) B(i) \delta m-t_m^{t+1}(i) \\
- \sum_{m=k-t_k^{t+1}}^{k-1} a_m L^C(t) \sum_{i=0}^{t} \left( \prod_{j=i}^{t} A(j) \right) w_{m-t_m^{t+1}}(i)
\]
In consideration of (23), the former part can be proven convergent to zero following the typical mathematical induction steps, and the latter part can be shown to satisfy condition (11) similar to the last term of (20).

The first term on the right-hand side of (20) is the only one that remains. This term can be almost surely bounded from above by a sample path dependent on constant times $\sum_{m=k-t_k^{t+1}}^{k-1} a_m$. Noting the selection of $a_k$, this term is then bounded by $c_0 a_{m-k-t_k^{t+1}}^{t+1}$, where $c_0$ is an suitable constant. Thus, this quantity is $o(1)$. For simplicity, we directly select the standard step-size of $a_k = \frac{1}{k}$. The general case is similar but with a complicated explanation. In this case,
\[
a_{k-t_k^{t+1}}^{t+1} \tau_k = \frac{1}{k} \tau_k = \frac{k}{k-t_k^{t+1}} \tau_k = O\left(\frac{1}{k}\right) \tau_k = O\left(\frac{1}{k^2}\right) = o(1)
\]
In sum, we show that the effect of random data dropouts, i.e., (20), satisfies condition (11). The convergence proof of this theorem is also achieved using the same steps of Theorem 1 and using Lemma 1. □
Remark 12. Unlike in the IUS case, the update is conducted in each iteration using the latest available information in the SUS case. As shown in (17), two random factors are involved: the random data dropout and the uncertain length of successive iterations that data dropouts occur. As data dropout is described by a Bernoulli random variable and is independent in different iterations, the length of successive dropout iterations is not a bounded variable. Therefore, we have to make an intensive estimation of the effect of this factor. This part is the kernel step of our proof, and it was not dealt with in previous ILC results.

5. Illustrative simulations

5.1. System descriptions

Let us consider a direct current motor control problem for velocity tracking. The dynamics of a permanent magnet linear motor (PMLM) is described as follows [42]:

\[
\begin{align*}
\dot{x}(t) &= v(t) \\
u(t) &= k_1 \psi_f \dot{x}(t) + R i(t) + L \dot{i}(t) \\
f_1(t) &= m \dot{v}(t) + f_{\text{fr}}(t) + f_{\text{ip}}(t) + f_{\text{load}}(t) + f_{\text{fr}}(t)
\end{align*}
\]

where \( k_1 = \frac{\pi}{12} \). The definitions of the notations are listed in Table 1.

Following the simplification procedures in [42], the PMLM model can be transformed into the following case:

\[
\begin{align*}
\dot{x}(t) &= v(t) \\
\dot{v}(t) &= -\frac{k_1 k_2 \psi_f^2}{R m} v(t) + \frac{k_2 \psi_f}{R m} u(t) \\
y(t) &= v(t)
\end{align*}
\]

where \( k_2 = 1.5 \frac{\pi}{12} \). According to our formulations, the discrete time interval is set to \( \Delta = 10 \) ms, and the whole iteration length is 1 s. That is, \( N = 100 \). The system is discretized using the Euler method. Taking the disturbances and noises into consideration, the following stochastic system is obtained:

\[
\begin{align*}
x(t + 1) &= x(t) + v(t) \Delta + \epsilon_1(t + 1) \\
v(t + 1) &= v(t) - \frac{k_1 k_2 \psi_f^2}{R m} v(t) + \Delta \frac{k_2 \psi_f}{R m} u(t) + \epsilon_2(t + 1) \\
y(t) &= v(t) + \epsilon(t)
\end{align*}
\]

where the parameters are given as follows: \( \psi_f = 0.031 \) m, \( R = 8.6 \Omega, \ m = 1.635 \) kg, and \( \psi_f = 0.35 \) Wb. The product of the input/output coupling value is 0.0378.

The reference trajectory is \( y_d(t) = 1/3(\sin(t/20) + 1 - \cos(3t/20)) \). \( 0 \leq t \leq 100 \). The initial control action is simply given as \( u_0(t) = 0 \), \( \forall t \).

The stochastic noises \( \epsilon_1(t), \epsilon_2(t), \) and \( \epsilon(t) \) obey normal distribution \( N(0, 0.02^2) \). Note that the upper bound of the tracking reference is small, and thus large noises dominate the output. To make the convergence clear, the stochastic noises here are set with a small derivation.

In the following, we present in detail the performance of the proposed algorithms and associated comparisons. Specifically, the tracking performances of IUS and SUS are illustrated in Section 5.2, which shows the advantage of SUS compared with IUS. The comparisons under different data dropout rates (DDRs) and different learning gains for both IUS and SUS are detailed in Sections 5.3 and 5.4, respectively. The comparison between the proposed algorithms and the conventional P-type algorithm is discussed in Section 5.5, which reveals the effect of decreasing sequence \( \{a_k\} \). These comparisons illustrate the application information of the proposed algorithms.

5.2. Tracking performances of IUS and SUS

In this subsection, we first verify the convergence properties of the proposed updating schemes and then compare IUS and SUS. Any positive number \( L_c \) can satisfy the conditions in the convergence theorems.

<table>
<thead>
<tr>
<th>notation</th>
<th>meaning</th>
<th>notation</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x(t) )</td>
<td>motor position</td>
<td>( f_1(t) )</td>
<td>developed force</td>
</tr>
<tr>
<td>( v(t) )</td>
<td>rotor velocity</td>
<td>( u(t) )</td>
<td>stator voltage</td>
</tr>
<tr>
<td>( l(t) )</td>
<td>current of stator</td>
<td>( R )</td>
<td>resistance of stator</td>
</tr>
<tr>
<td>( L )</td>
<td>inductance of stator</td>
<td>( \zeta )</td>
<td>pole pitch</td>
</tr>
<tr>
<td>( \psi_f )</td>
<td>flux linkage</td>
<td>( m )</td>
<td>rotor mass</td>
</tr>
<tr>
<td>( f_{\text{fr}}(t) )</td>
<td>frictional force</td>
<td>( f_{\text{ip}}(t) )</td>
<td>ripple force</td>
</tr>
<tr>
<td>( f_{\text{load}}(t) )</td>
<td>applied load force</td>
<td>( f_{u}(t) )</td>
<td>uncertainties/disturbances</td>
</tr>
</tbody>
</table>
We first set the probability as $\rho = 0.9$. That is, for any data packet, the probability of a successful transmission is 90%. To make expressions clear, let $\gamma = 1 - \rho$ denote the DDR, which is the average ratio of lost data to the whole data. In this case, the DDR is 10%, which is low. The decreasing sequence is set to $a_k = 1/(k + 10)$, the learning gain is $L_t = 55$, and algorithms (6) and (7) are run for 300 iterations.

The final outputs of both algorithms are shown in Fig. 2, where the solid, dash, and dotted lines denote the tracking reference, the final output of IUS, and the final output of SUS, respectively. As shown in the figure, three lines are almost coincident. Therefore, both schemes can converge to the desired target quickly under low DDR and stochastic noises. Moreover, the performances of both schemes are close to each other.

To understand the performances, we further plot the averaged absolute tracking error along the iteration axis. The average absolute tracking error (AATE) is defined as $\|e_k\| = (\sum_{t=1}^{N} \|e_k(t)\|)/N$ for the $k$th iteration. As shown in Fig. 3, little difference is observed between both schemes, thus confirming the above findings. Generally, the lower the DDR is, the closer the performances of IUS and SUS are. If no data dropout exists, then IUS and SUS will be the same.

Subsequently, we set $\rho = 0.3$, or equivalently $\gamma = 70\%$, to further compare the performance of both schemes. This percentage implies that the transmission channel is poor, as about 70% of the data will be dropped. The parameters of both algorithms are the same as those in the low DDR case above. The final outputs of both algorithms are displayed in Fig. 4.
Different from the low DDR case, SUS is more advantageous than IUS with respect to tracking accuracy for the same learning iterations under the high DDR condition.

As presented in Fig. 5, where the AATE along the iteration axis is shown for both schemes, the tracking errors of SUS are much smaller than those of IUS. The inherent reason for this condition is that the IUS scheme will stop updating if the corresponding packet is lost, whereas the SUS scheme keeps updating whether or not the corresponding packet is lost. In other words, SUS updates more times than IUS.

5.3. Comparison with different data dropout rates

To determine the influence of different DDRs, a comparison is made with respect to DDR between IUS and SUS. Here four DDRs are considered, namely, $\gamma = 10\%, 30\%, 50\%$, and $70\%$, respectively. Algorithms (6) and (7) are run for 300 iterations. The parameters are set to $a_k = 1/(k + 10)$ and $L_t = 55$. 

---

**Fig. 4.** Tracking performances of IUS and SUS at the final iteration: DDR = 70%.

**Fig. 5.** AATE of IUS and SUS along the iteration axis: DDR = 70%.
Fig. 6. AATE for IUS with respect to different DDGs: DDR = 10%, 30%, 50%, and 70%.

Fig. 7. AATE for SUS with respect to different DDGs: DDR = 10%, 30%, 50%, and 70%.

The AATE of the IUS case with respect to different DDGs is shown in Fig. 6, where the dash, dotted, solid, and dash-dotted lines denote the AATE from low DDR to high DDR, respectively. We find that tracking accuracy and convergence speed worsen distinctly as the DDR increases or the transmission condition worsens.

The SUS case is illustrated in Fig. 7, where the lines have the same meaning as those in the IUS case. The tracking accuracy of SUS has no visible changes for different DDGs. However, most data fail to be transmitted back when DDR increases. Thus, if the learning gain is quite large, there may be a large increase before the algorithm converges because of excessive updating of some available data.

5.4. Comparison with different learning gains

To determine how the learning gain $L_t$ is affected in practical applications, we compare the proposed algorithms with different learning gains. The parameters of the algorithms are given as $a_k = 1/(k + 10)$ and $\gamma = 30\%$.

Three different learning gains are simulated: $L = 55, 110, 165$. The results for IUS and SUS are shown in Figs. 8 and 9, respectively, where the profiles of AATE are plotted. A large learning gain may lead to a fast convergence speed for both IUS and SUS cases. However, the final tracking accuracy is not significantly improved by increasing the learning gains.
5.5. Comparison with conventional P-type algorithm

This subsection analyzes the effect of decreasing the sequence $a_k$ and the comparison between the proposed algorithms and the conventional P-type algorithms. As has been analyzed in previous sections, $a_k$ relaxes the selection of learning gain and guarantees a zero-convergence under stochastic noises. When the decreasing sequence $a_k$ in (6) and (7) is taken out, algorithms (6) and (7) become conventional P-type algorithms.

We first verify the advantage of learning gain selection with the introduction of $a_k$. To do so, we select $L_t = 40$. As shown in Fig. 10, the conventional P-type algorithms fail to converge whether they are intermittent updating or successive updating. In this case, the DDR is set to a low value, $\gamma = 10\%$. The lines denote the profiles of maximal errors, defined as $\max_t |e_k(t)|$, along the iteration axis.

However, the proposed algorithms still guarantee convergence because of the introduction of $a_k$. In this example, we let the decreasing sequence be $a_k = 1/(k + 10)$. To make a fair comparison, the learning gain is set to $L_t = 440$, so that for the first iteration, the first coupling gain $a_1 L_t = 40$. The convergence is shown in Fig. 11.

Next, we compare the learning gain with the convergence of the proposed algorithms and the conventional P-type algorithms. For the conventional P-type algorithms, the learning gain is set to $L_t = 5$; for the proposed algorithms, the learning
Fig. 10. Maximal error profiles for both IUS and SUS without $a_k$ along the iteration axis: DDR = 10%.

Fig. 11. Maximal error profiles for both IUS and SUS with $a_k$ along the iteration axis: DDR = 10%.

Figures 10 and 11 show the maximal error profiles for both IUS and SUS without and with $a_k$ along the iteration axis, respectively. The DDR is 10% in both cases.

6. Conclusions

In this paper, the ILC is considered for stochastic linear systems with random data dropout. Data dropout is modeled by a Bernoulli random variable, which is valued 1 or 0, which denotes that the data are successfully transmitted or not, respectively. Two schemes are proposed to deal with the random data dropouts: the intermittent updating scheme, which updates its control signal if data are transmitted and does nothing otherwise, and the successive updating scheme, which always updates its control signal whether the data is transmitted or not. That is, if the data are successfully transmitted, then the algorithm will use these data; if the data are lost, then the algorithm will use the latest available data that have been stored. SUS has an advantage over IUS in tracking accuracy for the same learning iteration number under a high data dropout rate, as it can successively improve its performance. However, the difference is rather subtle. In both schemes, the gain is set to $L_t = 55$ and $a_k = 1/(k + 10)$. Therefore, the initial coupling learning gain is the same as that in the conventional P-type learning algorithms. The comparisons are illustrated in Figs. 12 and 13, in which the DDR is 50%.

In Fig. 12, the proposed IUS algorithm converges slower than the conventional P-type algorithm, as the decreasing sequence $a_k$ weakens the effect of the learning process. However, the difference in the SUS case is not obvious in Fig. 13.
input sequences are proved to converge to the desired input in an almost sure sense under stochastic noises and random data dropout. Illustrative simulations verify the theoretical analysis. For further research, more details on the relationship between the data dropout rate and the tracking performance are of great interest.

Appendix

Proof of Theorem 2. The proof is carried out by mathematical induction similar to the proof of Theorem 1.

Initial Step. Consider the case \( t = 0 \).

For \( t = 0 \), subtracting both sides of (18) from \( u_d(0) \) leads to

\[
\delta u_{k+1}(0) = \delta u_k(0) - a_k L_0 c_k(1)
\]

\[= \delta u_k(0) - a_k L_0 C^* B(0) \delta u_k(0)
\]

\[+ a_k L_0 C^* A(0) \delta x_k(0)
\]

\[+ a_k L_0 C(1) w_k(1) + a_k L_0 v_k(1) \]

(27)
Similar to the proof of Theorem 1, \( \sum_{k=0}^{\infty} a_k Q_k C(0) \delta x_k(0) < \infty \), \( \sum_{k=0}^{\infty} a_k Q_k C(1) w_k(1) < \infty \), and \( \sum_{k=1}^{\infty} a_k Q_k v_k(1) < \infty \). w.p.1. Note that \(-L_0 C(0)B(0)\) is stable through the suitable selection of \(L_0\). Thus, applying Lemma 1 to the recursion (27), we have that \( \delta u_k(0) \to 0 \) as \( k \to \infty \) w.p.1.

**Inductive Step.** Assume that the convergence of \( u_k(t) \) has been proved for \( t = 0, 1, \ldots, s-1 \) and that the aim is to show the convergence for \( t = s \).

The following recursion is easy to establish:

\[
\delta u_{k+1}(s) = \delta u_s(0) - a_k L_0 C^2 B(s) \delta u_k(s) + a_k L_0 C^2 A(s) \sum_{j=0}^{s-1} \left( \prod_{j=1}^{s-1} A(j) \right) B(i) \delta u_k(i) + a_k L_0 C(s + 1) w_k(s + 1) + a_k L_0 v_k(s + 1)
\]

(28)

Through the induction assumption, we have \( L_0 C^2 A(s) \sum_{j=0}^{s-1} \left( \prod_{j=1}^{s-1} A(j) \right) B(i) \delta u_k(i) \to 0 \) w.p.1, and the infinite summation of the last three terms on the right-hand side of (28) is finite. Thus, using Lemma 1 again leads to the conclusion. The proof is completed. □

**References**


