5-27-2009

Approximations of Stochastic Hybrid Systems

A Agung Julius
Rensselaer Polytechnic Institute

George J. Pappas
University of Pennsylvania, pappasg@seas.upenn.edu

Approximations of Stochastic Hybrid Systems

Abstract

This paper develops a notion of approximation for a class of stochastic hybrid systems that includes, as special cases, both jump linear stochastic systems and linear stochastic hybrid automata. Our approximation framework is based on the recently developed notion of the so-called stochastic simulation functions. These Lyapunov-like functions can be used to rigorously quantify the distance or error between a system and its approximate abstraction. For the class of jump linear stochastic systems and linear stochastic hybrid automata, we show that the computation of stochastic simulation functions can be cast as a tractable linear matrix inequality problem. This enables us to compute the modeling error incurred by abstracting some of the continuous dynamics, or by neglecting the influence of stochastic noise, or even the influence of stochastic discrete jumps.

Keywords

Approximation, bisimulation, stochastic hybrid systems, verification

Reprinted from:
Julius, A. A.; Pappas, G. J., "Approximations of Stochastic Hybrid Systems," Automatic Control, IEEE Transactions on , vol.54, no.6, pp.1193-1203, June 2009

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4982634&isnumber=5071740

Approximations of Stochastic Hybrid Systems

A. Agung Julius, *Member, IEEE*, and George J. Pappas, *Fellow, IEEE*

***Abstract*—This paper develops a notion of approximation for a class of stochastic hybrid systems that includes, as special cases, both jump linear stochastic systems and linear stochastic hybrid automata. Our approximation framework is based on the recently developed notion of the so-called stochastic simulation functions. These Lyapunov-like functions can be used to rigorously quantify the distance or error between a system and its approximate abstraction. For the class of jump linear stochastic systems and linear stochastic hybrid automata, we show that the computation of stochastic simulation functions can be cast as a tractable linear matrix inequality problem. This enables us to compute the modeling error incurred by abstracting some of the continuous dynamics, or by neglecting the influence of stochastic noise, or even the influence of stochastic discrete jumps.**

***Index Terms*—Approximation, bisimulation, stochastic hybrid systems, verification.**

I. Introduction

STOCHASTIC hybrid systems are hybrid systems where both the discrete and the continuous dynamics may contain stochastic behavior. Their tremendous modeling expressivity has enabled various researchers to use stochastic hybrid systems as models in various application domains such as air traffic management system [1]–[4], systems biology [5]–[8], biochemistry [9], and communication networks [10], [11].

There are several modeling formalisms for stochastic hybrid systems. In [12], a general type of stochastic hybrid systems, whose continuous dynamics is described by diffusion stochastic differential equation [13], is presented. Mode switching occurs when some invariant condition in the corresponding mode is violated. Earlier work on stochastic hybrid systems can be found in [14], which features multi-modal diffusion equation called the switched diffusion processes. A later work by Ghosh and Bagchi [15] enriched the previous framework with reset. Another framework that is also popular is the piecewise deterministic Markov processes [10], [16]. This framework does not feature stochastic differential equations, but uses deterministic continuous dynamics described by ordinary differential equations. In this framework, the discrete switching is modeled as a Poisson process. For a more thorough survey on the various modeling formalisms for stochastic hybrid systems, the interested reader is referred to [2], [17].

A basic analysis problem for stochastic hybrid systems is the so-called safety verification problem which tries to compute the probability that the system will enter a particular region of the state space. Since the complexity of the safety verification problem depends on the size of the state space, a natural approach is to abstract the original systems by a simpler abstracted model. If the modeling error between the two systems can be quantified, then analysis can be performed on the abstracted, simpler system and the results can then be carried over into the original system.

Our approximation framework for stochastic hybrid systems is inspired by the recent notion of approximate bisimulation [18], [19], developed for non-stochastic discrete or continuous, systems. Approximate bisimulation naturally generalize exact notions of system refinement and equivalence [20], [21]. Notions of exact bisimulation have been recently developed for some classes of stochastic hybrid systems in [22], [23]. In particular, in [22], a notion of exact bisimulation for general stochastic hybrid systems is developed using notions from category theory, whereas in [23] the issue of exact bisimulation is treated for the so called communicating piecewise deterministic hybrid systems.

In the context of stochastic hybrid systems, requiring that the abstraction of a system is *exactly* equivalent to the original system can be too restrictive. If we allow for some error in the notion of equivalence, we can obtain an abstraction that is only approximately equivalent to the original system, but has lesser complexity than any of the systems that are exactly equivalent. Notions of approximate equivalence of systems have been developed, for example in [18], [19], [24]. Critical to these approaches is the use of a metric with which we measure the distance between systems and hence the quality of the approximation.

In this paper, we consider approximate abstraction of stochastic hybrid systems, using the idea of *approximate bisimulation*, which is developed in [18], [19] for non-stochastic systems. There have been other works in the same direction. In [25], [26], the authors develop some metrics for labeled Markov processes and probabilistic transition systems, inspired by the Hutchinson metric, which gives the distance between two distributions of the transition probability. The approach that we take in this paper differs from that, in two aspects. First, we use a different kind of metric. The metric that we use is based on the L_∞ distance between the output trajectories of the systems. Second, the modeling formalism that we use is also different. Rather than embedding the stochastic hybrid systems as stochastic transition systems, we adopt the idea of *bisimulation function* from [18], [19], and develop a stochastic version of it.

In this paper, we develop a theory of approximate bisimulation for a class of stochastic hybrid automata, in which the

Manuscript received June 08, 2006; revised June 10, 2007. First published May 27, 2009; current version published June 10, 2009. This work was supported in part by the National Science Foundation Presidential Early CAREER (PECASE) Grant 0132716. Recommended by Associate Editor J. P. Hespanha.

A. A. Julius is with the Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: agung@ecse.rpi.edu).

G. J. Pappas is with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia PA 19104 USA (e-mail: pappasg@seas.upenn.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2009.2019791

continuous dynamics is modeled by stochastic differential equations and the switches are modeled as Poisson processes. The model that we consider in this paper is thus close to the stochastic hybrid systems in [15], with the exception that we always associate reset with a switch in the discrete state. Establishing approximate bisimulation between two stochastic systems amounts to constructing a Lyapunov-like stochastic bisimulation function. We first develop a general framework for systems with potentially nonlinear dynamics and resets. In order to make our framework computational, we then focus on a special class of stochastic hybrid systems with linear continuous dynamics and linear reset maps. This class of systems is called the Jump Linear Stochastic Systems (JLSS) [27]. We show that for the class of jump linear stochastic systems, the problem of constructing a stochastic bisimulation function in the class of quadratic functions can be posed as a linear matrix inequality problem, which can be efficiently solved using available computational tools.

In modeling deterministic hybrid systems, an automata-based framework called hybrid automata is very popular [28]. In this paper, we also introduce an automata-based class of hybrid systems with linear stochastic dynamics and random switches called Linear Stochastic Hybrid Automata (LSHA). This class is introduced because it is generally more intuitive for the users to formulate hybrid model in an automata-like structure, where different continuous dynamics can be explicitly specified for different discrete states. We then show that a class of stochastic hybrid automata with linear dynamics can be 'flattened' and reformulated as jump linear stochastic systems.

The rest of this paper is organized as follows. In Section II we present the modeling framework that we consider and the concept of stochastic simulation function. In Section III, we present a subclass of the systems presented in the preceding section, namely Jump Linear Stochastic Systems (JLSS). We also develop the method for the computation of stochastic simulation functions for JLSS. In Section IV, we introduce the class of Linear Stochastic Hybrid Automata (LSHA), and show how an LSHA can be reformulated to an equivalent JLSS. In Section V, we present some numerical examples where stochastic simulation functions are calculated based on the procedures described in the preceding sections. We conclude the paper with some discussion on potential further extensions of the research.

II. Stochastic Hybrid Models

A. Modeling Formalism

In this paper, we discuss stochastic systems of the following form:

$$dx_t = f(x_t, u_t)dt + g(x_t)dw_t + r(x_t)dp_t \tag{1}$$

$$x_t \in \mathcal{X}, u_t \in \mathcal{U},\ \forall t \in \mathbb{R}_+ \tag{2}$$

$$y_t = h(x_t) \tag{3}$$

where $f(\cdot,\cdot)$ is continuously differentiable and $g(\cdot)$, $h(\cdot)$ and $r(\cdot)$ are locally Lipschitz continuous functions.

The process y_t is the observation of the process x_t, the signal u_t is a smooth input taking value in a compact set $\mathcal{U}$, the process w_t is a standard Brownian motion, while p_t is a Poisson process with a measurable state dependent rate $\lambda(x_t)$. We assume that the Poisson processes and the Brownian motion are independent of each other. The input u can be thought of as an internal disturbance that generates nondeterminism in the systems, rather than external control input.

Notation: We denote the class of smooth function taking value in the compact set $\mathcal{U}$ as $\mathfrak{U}$.

A Poisson process with a state dependent rate λ is a piecewise constant, monotonously nondecreasing process

$$p_t = \begin{cases} 0, & 0 \le t < t_1 \\ n, & t_n \le t < t_{n+1}, n \in \mathbb{Z}_+ \end{cases} \tag{4}$$

where $t_1, t_2, \cdots, t_n, \cdots$ are random time instants t_n are called *event times*. Poisson processes are commonly used in modeling stochastic arrival processes (see [16], [29]).

The system described in (1) then can be interpreted as follows. In between the event times generated by the Poisson process p_t, the process is described by the stochastic differential equation

$$dx_t = f(x_t, u_t)dt + g(x_t)dw_t \tag{5}$$

$$y_t = h(x_t). \tag{6}$$

At the event time t_n, the process undergoes a jump

$$x_{t_n} = \lim_{t \uparrow t_n} x_t + r\left(\lim_{t \uparrow t_n} x_t\right). \tag{7}$$

We use a Poisson process to model the occurrences of an event. The effect of an occurrence of the event is expressed as a reset (7). However, it is possible that we need to include more than just one kind of events in the model. Thus, generally the model (1) can be slightly extended to be

$$dx_t = f(x_t, u_t)dt + g(x_t)dw_t + \sum_{i=1}^{N} r_i(x_t)dp_t^i. \tag{8}$$

That is, we model N kinds of events whose occurrences are independent one from the others. The function $r_i, i = 1, \ldots, N$, parametrizes the jump associated with event i. We also assume that the Poisson process p_t^i has the rate of $\lambda_i(x_t)$.

Remark 1: Note that the modeling framework (1) that we introduce here is a special class of that introduced in [15], [17]. The existence, uniqueness and strong Markov property of the solution of (1) has been shown, and the infinitesimal generator has also been given.

B. Stochastic (Bi)Simulation Function

In this subsection we present the main machinery that is used in approximate abstraction of stochastic hybrid systems in this paper. That is, the stochastic (bi)simulation function.

Given two systems in the form given in (8). For $i = 1, 2$, the system S_i is given by

$$dx_{i,t} = f_i(x_{i,t}, u_{i,t})dt + g_i(x_{i,t})dw_t + \sum_{k=1}^{N} r_{i,k}(x_{i,t})dp_t^k \tag{9a}$$

$$x_{i,t} \in \mathcal{X}_i, u_{i,t} \in \mathcal{U}_i, \forall t \in \mathbb{R}_+ \tag{9b}$$

$$y_{i,t} = h_i(x_{i,t}). \tag{9c}$$

We assume that $y_{1,t}$ and $y_{2,t}$ are of equal dimension.

By defining the following processes:

$$x_t := \begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix},\ u_t := \begin{bmatrix} u_{1,t} \\ u_{2,t} \end{bmatrix},\ y_t := y_{1,t} - y_{2,t} \tag{10}$$

we can define the following system.

$$S : \begin{cases} dx_t = f(x_t, u_t)dt + g(x_t)dw_t + \sum_{k=1}^{N} r_k(x_t)dp_t^k \\ y_t = h(x_t) \end{cases} \tag{11}$$

where

$$f(x_t, u_t) = \begin{bmatrix} f_1(x_{1,t}, u_{1,t}) \\ f_2(x_{2,t}, u_{2,t}) \end{bmatrix},\ g(x_t) = \begin{bmatrix} g_1(x_{1,t}) \\ g_2(x_{2,t}) \end{bmatrix}$$
$$r_k(x_t) = \begin{bmatrix} r_{1,k}(x_{1,t}) \\ r_{2,k}(x_{2,t}) \end{bmatrix},\ h(x_t) = h_1(x_{1,t}) - h_2(x_{2,t}).$$

We also define

$$\mathcal{X} := \mathcal{X}_1 \times \mathcal{X}_2,\ \mathfrak{U} := \mathfrak{U}_1 \times \mathfrak{U}_2. \tag{12}$$

Observe that if u_t and the distribution of the initial state x_0 are known, then x_t is a stochastic process.

Definition 2: A continuous function $\phi : \mathcal{X} \to \mathbb{R}_+ \cup \{+\infty\}$ is a **stochastic simulation function** of S_1 by S_2 if

i) For any $x \in \mathcal{X}_1 \times \mathcal{X}_2$, $\phi(x) \geq \|h(x)\|^2$;

ii) For any $u_1 \in \mathfrak{U}_1$, there exists a $u_2 \in \mathfrak{U}_2$ such that the stochastic process $\phi(x_t)$ is a supermartingale[1] for any initial state x_0.

The significance of a stochastic simulation function can be intuitively stated as follows. Condition (i) states the ϕ is an upper bound for the distance between the observations of the two systems S_1 and S_2. Condition (ii) states that for any decision u_1 that S_1 makes, S_2 can make another decision u_2 such that the expectation of ϕ is nonincreasing, which hints at the ability of S_2 to track S_1.

A precise statement about the significance of the stochastic simulation function is given as follows.

Theorem 3: Given a system described by (11), and $\phi(\cdot)$ a stochastic simulation function. For any $u_1 \in \mathfrak{U}_1$ there exists a $u_2 \in \mathfrak{U}_2$ such that the following relation holds:

$$P\left\{\sup_{0\leq t<\infty} \|y_t\|^2 > \delta | x_0\right\} \leq \frac{\phi(x_0)}{\delta}. \tag{13}$$

Proof: Following Definition 2, for any $u_1 \in \mathfrak{U}_1$ there exists a $u_2 \in \mathfrak{U}_2$ such that $\phi(x_t)$ is a supermartingale. Since $\phi(x_t)$ is a nonnegative supermartingale, we have the following result [30], [31]:

$$P\left\{\sup_{0\leq t<\infty} \phi(x_t) > \delta | x_0\right\} \leq \frac{\phi(x_0)}{\delta}. \tag{14}$$

Moreover, since $\phi(x) \geq \|h(x)\|^2$ by construction, we also have that

$$P\left\{\sup_{0\leq t<\infty} \|y_t\|^2 > \delta | x_0\right\} \leq P\left\{\sup_{0\leq t<\infty} \phi(x_t) > \delta | x_0\right\}. \tag{15}$$

■

[1]i.e. its expectation is monotonously nonincreasing.

Remark 4: Simulation for nonstochastic systems is typically seen as a two-player tracking game [18], [32], [33]. For stochastic systems, one can think of the stochasticity as a third player in the game. In this point of view, there are multiple interpretations about the order, with which the game is played. That is, when the third player (stochasticity) makes its decision. There are three possibilities, namely before the other two players, in between, or after them. The definition of simulation that we adopt in this paper is based on the interpretation that the third player makes its decision after the other two players, that is, the inputs u_1 and u_2. Our choice is mainly based on computation consideration, although we can see later that this choice also leads to a sensible relationship between bisimulation and safety verification. That being said, we acknowledge that exploring the relations between all three interpretations of simulation is a separate interesting research direction.

A symmetric version of a stochastic simulation function is called a stochastic bisimulation function.

Definition 5: A function $\phi : \mathcal{X} \to \mathbb{R}_+ \cup \{+\infty\}$ is a **stochastic bisimulation function** between S_1 and S_2 if it is both a stochastic simulation function of S_1 by S_2 and of S_2 by S_1.

It is straightforward to infer that a stochastic bisimulation function satisfies the symmetric version of Theorem 3.

Supposed that we are given a complex stochastic system S_1 and its abstraction S_2 in the form of (9). Theorem 3 tells us that a stochastic simulation function can be used to quantify the distance between the two systems S_1 and S_2. For a better illustration, consider the following corollary.

Corollary 6: Given two systems S_1 and S_2 as in ((9)–(11)), and suppose that $\phi(\cdot)$ is a stochastic simulation function of S_1 by S_2. We have the following relations. For any $u_1 \in \mathfrak{U}_1$ there exists a $u_2 \in \mathfrak{U}_2$ such that

$$P\left\{\sup_{0\leq t<\infty} \|y_{1,t} - y_{2,t}\|^2 > 10 \cdot \phi(x_0) | x_0\right\} \leq 0.1 \tag{16a}$$
$$P\left\{\sup_{0\leq t<\infty} \|y_{1,t} - y_{2,t}\|^2 > 20 \cdot \phi(x_0) | x_0\right\} \leq 0.05. \tag{16b}$$

Thus, we can say S_2 can simulate S_1 (by selecting appropriate input u_2), such that the supremum of the difference in the observations will not exceed $\sqrt{10 \cdot \phi(x_0)}$ and $\sqrt{20 \cdot \phi(x_0)}$ with 90% and 95% confidence respectively.

Notice that given two systems, the stochastic simulation between them is not unique. For example, if $\phi(\cdot)$ is a stochastic simulation function, then $\lambda\phi(\cdot)$, where λ is a real number larger than 1, is also a stochastic simulation function. As mentioned in Theorem 3 and Corollary 6, the stochastic simulation function is used to construct an *upper bound* on how much the observations can differ. In general, we would like to have a tight upper bound, which corresponds to small stochastic simulation function.

This idea can be used in conjunction with stochastic safety/reachability analysis of hybrid systems, which is one of the main challenges in the field [28], [34]. Suppose that $\phi(\cdot)$ is a stochastic simulation function of S_1 by S_2, and that the initial condition of the composite system is $x_0 = (x_{10}, x_{20})$. Given the open unsafe set for the original system S_1, $\texttt{unsafe}_1$, we can

construct another set $\texttt{unsafe}_2$, which is the δ neighborhood of $\texttt{unsafe}_1$ for some $\delta > 0$. That is,

$$\texttt{unsafe}_2 = \{y | \exists y' \in \texttt{unsafe}_1, \|y - y'\| \leq \delta\}. \tag{17}$$

If we define the events $\mathbf{unsafe}_1(v)$ and $\mathbf{unsafe}_2(v)$ as functions of the external input signal, for $i = 1,2$

$$\mathbf{unsafe}_i(v) := \{\exists t \geq 0 \text{ s.t. } y_{i,t} \in \texttt{unsafe}_i | u_i = v \in \mathfrak{U}_i\} \tag{18}$$

then we have the following theorem holds.

Theorem 7:

$$\sup_{u_1 \in \mathfrak{U}_1} P\{\mathbf{unsafe}_1(u_1)\} \leq \sup_{u_2 \in \mathfrak{U}_2} P\{\mathbf{unsafe}_2(u_2)\} + \frac{\phi(x_0)}{\delta^2}. \tag{19}$$

Proof: We start with the following relation. Take any $u_1 \in \mathfrak{U}_1$, let $\tilde{u}_2 \in \mathfrak{U}_2$ be such that $\phi(x_t)$ is a supermartingale (see Definition 2). We then have the following relation:

$$\begin{aligned} P\{\mathbf{unsafe}_1(u_1)\} &= P\{\mathbf{unsafe}_1(u_1) \cap \mathbf{unsafe}_2(\tilde{u}_2)\} \\ &\quad + P\left\{\mathbf{unsafe}_1(u_1) \cap \mathbf{unsafe}_2^C(\tilde{u}_2)\right\} \end{aligned} \tag{20}$$

where $\mathbf{unsafe}_2^C(\tilde{u}_2)$ denotes the complement of the event $\mathbf{unsafe}_2(\tilde{u}_2)$. Now, notice that

$$\begin{aligned} &P\{\mathbf{unsafe}_1(u_1) \cap \mathbf{unsafe}_2(\tilde{u}_2)\} \\ &\quad \leq P\{\mathbf{unsafe}_2(\tilde{u}_2)\} \\ &\quad \leq \sup_{u_2 \in \mathfrak{U}_2} P\{\mathbf{unsafe}_2(u_2)\}. \end{aligned} \tag{21}$$

and because of Theorem 3

$$P\left\{\mathbf{unsafe}_1(u_1) \cap \mathbf{unsafe}_2^C(\tilde{u}_2)\right\} \leq \frac{\phi(x_0)}{\delta^2}. \tag{22}$$

Thus we have that for any $u_1 \in \mathfrak{U}_1$

$$P\{\mathbf{unsafe}_1(u_1)\} \leq \sup_{u_2 \in \mathfrak{U}_2} P\{\mathbf{unsafe}_2(u_2)\} + \frac{\phi(x_0)}{\delta^2}.$$

■

The term $\sup_{u_1 \in \mathfrak{U}_1} P\{\mathbf{unsafe}_1(u_1)\}$ gives us the risk of unsafety of S_1 in the worst scenario. That is, we choose the input so as to maximize the risk. Similarly, the term $\sup_{u_2 \in \mathfrak{U}_2} P\{\mathbf{unsafe}_2(u_2)\}$ gives us the risk of unsafety of S_2 in the worst scenario. Theorem 7 tells us that we can get an upper bound of the risk of the complex system by performing the risk calculation on the simple abstraction and adding a factor that depends on the stochastic simulation function.

The infinitesimal generator of the process x_t is given in [15], [29] as

$$\begin{aligned} \mathcal{L}\phi = \frac{\partial \phi}{\partial x} f(x, u_t) + \frac{1}{2} \text{trace}\left(g^T(x) \frac{\partial^2 \phi}{\partial x^2} g(x)\right) \\ + \sum_{j=1}^{N} \lambda_j(x) \left(\phi\left(x + r_j(x)\right) - \phi(x)\right) \end{aligned} \tag{23}$$

assuming that $\phi(\cdot)$ is in the domain of the generator.

The following theorem gives a sufficient condition for the construction of a stochastic simulation function of S_1 by S_2.

Theorem 8: Suppose that $\phi(\cdot)$ satisfies

$$\begin{aligned} \phi(x) &\geq \|h(x)\|^2 \\ \max_{u_1 \in \mathfrak{U}_1} \min_{u_2 \in \mathfrak{U}_2} \mathcal{L}\phi(x,t) &\leq 0, \ \forall x \in \mathcal{X}, \forall t \in \mathbb{R}_+ \end{aligned}$$

then $\phi(\cdot)$ is a stochastic simulation function of S_1 by S_2.

Proof: Condition (i) in Definition 2 is trivially satisfied. To show that ϕ_t is a supermartingale, we use Dynkin's formula [30]

$$\begin{aligned} E[\phi_t | \phi_s] &= \phi_s + E\left[\int_s^t \mathcal{L}\phi_\tau d\tau\right], \ s < t \\ &\leq \phi_s. \end{aligned} \tag{24}$$

■

In the rest of the paper, we are going to impose certain assumptions on the structure of the dynamics of the system, such that this condition can be put into a computationally tractable framework.

III. Jump Linear Stochastic Systems

A. Definition

In this section, we shall focus on a specific class of the systems discussed in the previous section. We shall assume that the underlying dynamics and the reset map of the systems in this class are linear.

A *jump linear stochastic system* (JLSS) can be modeled as a stochastic system that satisfies the following stochastic differential equation:

$$dx_t = Ax_t dt + Bu_t dt + Fx_t dw_t + Rx_t dp_t \tag{25a}$$

$$u(t) \in \mathcal{U}, \ \forall t \in \mathbb{R}_+ \tag{25b}$$

$$y_t = Cx_t. \tag{25c}$$

Here, y_t is the observation of the process x_t, the signal u_t is an input taking value in a compact set $\mathcal{U}$, the process w_t is a standard Brownian motion, while p_t is a Poisson process with a constant rate λ.

The JLSS described in (25) can be interpreted as follows. In between the event times generated by the Poisson process p_t, the process behaves like a linear stochastic system

$$dx_t = Ax_t dt + Bu(t)dt + Fx_t dw_t \tag{26}$$

$$y_t = Cx_t. \tag{27}$$

At the event time t_n, the process undergoes a jump

$$x_{t_n} = (I + R) \lim_{t \uparrow t_n} x_t. \tag{28}$$

Notice that with R we can parametrize any linear jump. Hence the name, jump linear stochastic system.

As is the case in the previous section, the JLSS model (25) can be slightly extended to include multiple events

$$dx_t = Ax_t dt + Bu_t dt + Fx_t dw_t + \sum_{i=1}^{N} R_i x_t dp_t^i. \tag{29}$$

That is, we model N kinds of event whose occurrences are independent one from the others. The matrices $R_i, i = 1, \ldots, N$, parametrize the jump associated with event i. We also assume that the Poisson process p_t^i has the rate of λ_i.

B. Construction of the Stochastic Simulation Functions

Given two JLSS, for $i = 1,2$

$$S_i : \begin{cases} dx_{i,t} = A_i x_{i,t} dt + B_i u_{i,t} dt + F_i x_{i,t} dw_t \\ \quad + \sum_{j=1}^{N} R_{ij} x_{i,t} dp_t^j \\ u_{i,t} \in \mathfrak{U}_i \\ y_{i,t} = C_i x_{i,t}. \end{cases} \tag{30}$$

We define the following composite process:

$$x_t := \begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix}, y_t := y_{1,t} - y_{2,t}, u_t := \begin{bmatrix} u_{1,t} \\ u_{2,t} \end{bmatrix}$$
$$A := \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix}, B := \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix}, F := \begin{bmatrix} F_1 & 0 \\ 0 & F_2 \end{bmatrix}$$
$$R_j := \begin{bmatrix} R_{1j} & 0 \\ 0 & R_{2j} \end{bmatrix}, C := [C_1 \quad -C_2].$$

Hence we have the following process:

$$S : \begin{cases} dx_t = Ax_t dt + Bu_t dt + Fx_t dw_t + \sum_{j=1}^{N} R_j x_t dp_t^j \\ u \in \mathfrak{U}_1 \times \mathfrak{U}_2 \\ y_t = Cx_t. \end{cases} \tag{31}$$

A stochastic simulation function of S_1 by S_2 can be constructed following Definition 2. We assume that a stochastic simulation function can be constructed as a quadratic function of the (composite) state, that is, a function of the following form:

$$\phi(x) = x^T M x \tag{32}$$

where M is symmetric nonnegative definite.

Using the infinitesimal generator given in (23), we obtain

$$\begin{aligned} \mathcal{L}\phi = & \frac{\partial \phi}{\partial x} f(x, u_t) + \frac{1}{2} \text{trace}\left(g^T(x) \frac{\partial^2 \phi}{\partial x^2} g(x) \right) \\ & + \sum_{j=1}^{N} \lambda_j(x) \left(\phi\left(x + r_j(x)\right) - \phi(x) \right) \qquad (33) \\ = & 2x^T M(Ax + Bu_t) + x^T F^T M F x \\ & + \sum_{j=1}^{N} \lambda_j x^T R_j^T M R_j x + \sum_{j=1}^{N} \lambda_j x^T R_j^T M x \\ & + \sum_{j=1}^{N} \lambda_j x^T M R_j x. \qquad (34) \end{aligned}$$

Denote

$$Q := M\left(A + \sum_{j=1}^{N} \lambda_j R_j \right) + \left(A + \sum_{j=1}^{N} \lambda_j R_j \right)^T M + F^T M F + \sum_{j=1}^{N} \lambda_j R_j^T M R_j$$

then we have that

$$\mathcal{L}\phi = x^T Q x + 2x^T M B u_t. \tag{35}$$

Lemma 9: The function ϕ is a stochastic simulation function of S_1 by S_2 if and only if the following relations are satisfied:

$$M - C^T C \geq 0 \tag{36}$$

and for almost all $t \geq 0$

$$\max_{u_1 \in \mathfrak{U}_1} \min_{u_2 \in \mathfrak{U}_2} x^T Q x + 2x^T M B u_t \leq 0 \tag{37}$$

for any distribution of the initial state x_0.

Proof: (if) Suppose that (36) is satisfied, then we know that condition (i) of Definition 2 is satisfied by ϕ. Moreover, if (37) is satisfied, then from (35) and the Dynkin's formula we know that ϕ_t is a supermartingale as required by conditions (ii) of Definition 2.

(only if) We can see that (36) is clearly a necessary condition, otherwise condition (i) of Definition 2 will not be satisfied. The condition (37) is also necessary, because otherwise we can select some input $u_1 \in \mathfrak{U}_1$ such that there is no $u_2 \in \mathfrak{U}_2$ that will make ϕ_t a supermartingale, as required by condition (ii) in Definition 2. ■

The reason why we focus on systems with linear dynamics, is because we want to actually compute the stochastic simulation function. In (37) we can see that in order to compute a stochastic simulation function, we need to solve a game, which is computationally difficult. In order to make the problem computationally more tractable, in the remaining of the paper, we shall impose the following assumption.

Assumption: Hereafter, we assume that the disturbances are absent. That is

$$B = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} = 0. \tag{38}$$

This assumption leads to the absence of the nondeterminism. As a consequence, the composite system is essentially a stochastic process and (37) is reduced to an inequality (instead of a game). A function ϕ is a stochastic simulation function of S_1 by S_2 if and only if

$$M - C^T C \geq 0 \tag{39}$$

and for all x

$$x^T Q x \leq 0. \tag{40}$$

Notice that, by symmetry, such a function is also a stochastic bisimulation function between S_1 and S_2.

The problem of constructing a matrix M that satisfies (39), (40) is a linear matrix inequality problem (LMI) that can be solved using some available tools, such as YALMIP [35] and `cvx` [36].

Remark 10: If we think of the stochastic bisimulation function (32) as a stochastic Lyapunov function, then the strict inequality version of (40) together with (39) guarantee that y_t converges to 0 in probability [37].

IV. Linear Stochastic Hybrid Automata

A. Definition

In this section, we present a class of stochastic hybrid systems, called the *linear stochastic hybrid automata* (LSHA).

An LSHA has an automata-like structure similar to that of the widely known hybrid automata [28], [34]. This class is introduced because it is generally more intuitive for the users to formulate hybrid model in an automata-like structure. We formally define a linear stochastic hybrid automaton (LSHA) as a 5-tuple $\mathcal{A} = (L, n, m, T, \mathrm{Dyn})$, where

- L is a finite set, which is the set of locations or discrete states. The number of locations is denoted by $|L|$;
- $n : L \to \mathbb{N}$, where for every $l \in L$, $n(l)$ is the dimension of the continuous state space in location l;
- $m \in \mathbb{N}$, is the dimension of the output of the automaton $\mathcal{A}$;
- T is the set of random transitions. A transition $\tau \in T$ can be written as a 4-tuple $(l, \lambda_\tau, l', R_\tau)$. This is a transition from location $l \in L$ to $l' \in L$ that is triggered by a Poisson process with intensity $\lambda_\tau \in \mathbb{R}_+$. The matrix $R_\tau \in \mathbb{R}^{n(l') \times n(l)}$ is the linear reset map associated with the transition τ. The number of transitions is denoted by $|T|$;
- Dyn defines the continuous dynamics in each location. For every $l \in L$, $\mathrm{Dyn}(l)$ is a triple (A_l, F_l, C_l), where $A_l \in \mathbb{R}^{n(l) \times n(l)}$, $G_l \in \mathbb{R}^{n(l) \times n(l)}$ and $C_l \in \mathbb{R}^{m \times n(l)}$.

The state space of the automaton can be written as

$$\mathcal{X} = \bigcup_{i=1}^{|L|} \left(\{l_i\} \times \mathbb{R}^{n(l_i)} \right). \tag{41}$$

We also define the functions $\mathtt{source} : T \to L$ and $\mathtt{dest} : T \to L$, such that if $\tau \in T$ is $(l, \lambda_\tau, l', R_\tau)$ then

$$\mathtt{source}(\tau) = l, \ \mathtt{dest}(\tau) = l'. \tag{42}$$

The semantics of the linear stochastic hybrid automaton $\mathcal{A}$ can be explained as follows. The state trajectory $\xi_t = (l_t, x_t)$ of the LSHA $\mathcal{A}$ is inherently a stochastic process. Every state trajectory that the automaton executes is a realization of the process. In each location $l \in L$, the continuous state of the system satisfies the following stochastic differential equation (SDE):

$$dx_{l,t} = A_l x_{l,t} dt + F_l x_{l,t} dw_t \tag{43a}$$

$$y_t = C_l x_{l,t} \tag{43b}$$

$$x_{l,t} \in \mathbb{R}^{n(l)}, \ y_t \in \mathbb{R}^m. \tag{43c}$$

The process w_t is a standard Brownian motion. The $\mathbb{R}^m$ valued stochastic process y_t is the output/observation of automaton A.

Remark 11: In general, it is possible to incorporate multi dimensional Brownian motions in the framework. In this case, the term $F_l x_{l,t} dw_t$ in (43a) would be replaced by $\sum_{i=1}^{N} F_{l,i} x_{l,t} dw_{i,t}$ to incorporate an N-dimensional Brownian motion. Hereafter, we use one dimensional Brownian motion for simplicity.

Denote the set of outgoing transitions of a location as

$$\mathtt{out} : L \to 2^T, \ \mathtt{out}(l) := \{\tau \in T | \mathtt{source}(\tau) = l\} \tag{44}$$

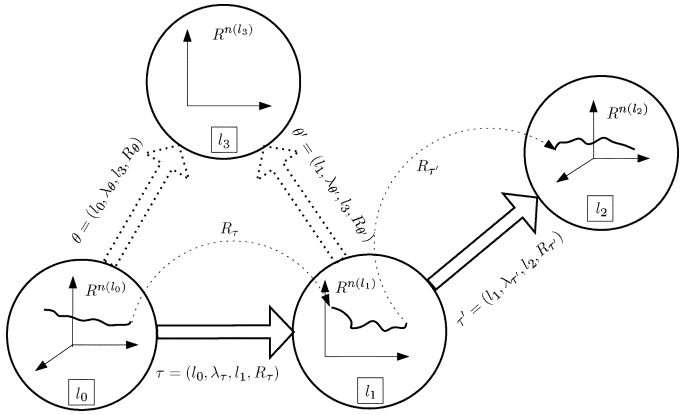


Fig. 1. Illustration of the execution of an LSHA. The solid bold arrows represent transitions between locations that occur. The dotted bold arrows indicate transitions that do not occur, since the associated Poisson process do not generate a point fast enough. The dotted arrows denote the linear reset maps associated with the transitions that occur.

and $|\mathtt{out}(l)|$ as the number of outgoing transitions from location l. While the system is evolving in a location $l \in L$, each transition in $\mathtt{out}(l)$ is represented by an active Poisson process. Each of these Poisson processes has a constant rate indicated by the transition. The first Poisson process to generate a point triggers a transition. Suppose that $\tau = (l, \lambda_\tau, l', R_\tau)$ is the transition that corresponds to the first process that generates a point (at time t), then the evolution of the system will switch to location l'. The matrix R_τ defines a linear reset map

$$x_t = R_\tau x_{t-} \tag{45}$$

where $x_{t-} := \lim_{s \uparrow t} x_s$.

Fig. 1 illustrates a realization of the execution of an LSHA. In Fig. 1, the execution starts in location l_0 by following the SDE that defines the dynamics in the location. The set of outgoing transitions from l_0, $\mathtt{out}(l_0) = \{\tau, \theta\}$. In this particular realization, the Poisson process associated with τ generates a point before that of θ. Hence, a transition occurs that brings the trajectory to location $\mathtt{dest}(\tau) = l_1$. The continuous state of the trajectory is reset by the linear map R_τ. In the new location, the continuous dynamics proceeds with the SDE that defines the dynamics in location l_1. The set $\mathtt{out}(l_1) = \{\tau', \theta'\}$. In this particular realization, the Poisson process associated with τ' generates a point before that of θ'. Hence, a transition occurs that brings the trajectory to location l_2. The continuous state of the system is then subsequently reset by the linear map $R_{\tau'}$.

B. Casting LSHA as JLSS

In this subsection, we demonstrate how an LSHA can be cast as a jump linear stochastic system (JLSS), defined in the previous section. We shall then use the tools that have been developed for JLSS to construct stochastic bisimulation functions for LSHA.

Given an LSHA $\mathcal{A} = (L, n, m, T, F)$ as defined in the previous subsection, the following is an algorithm to define a JLSS, structured as in (25), that represents $\mathcal{A}$.

- The state space of the JLSS has the dimension of $\sum_{i=1}^{|L|} n(l_i)$, $l_i \in L$.

- The A and F matrices of the JLSS has a block diagonal structure, with $|L|$ blocks. That is

$$A := \begin{bmatrix} A_1 & 0 & \cdots & 0 \\ 0 & A_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{|L|} \end{bmatrix}$$

$$G := \begin{bmatrix} F_1 & 0 & \cdots & 0 \\ 0 & F_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & F_{|L|} \end{bmatrix}$$

  where $A_i := A_{l_i}$ and $F_i := F_{l_i}$ are the A and F matrices of the LSHA in location l_i.
- The C matrix of the JLSS is structured as $C := [C_1\ C_2\ \cdots\ C_{|L|}]$, where $C_i := C_{l_i}$ is the C matrix of the LSHA in location l_i.
- There are $|T|$ independent Poisson processes. Thus, $N = |T|$. Each Poisson process represents a transition in T. Denote the transitions as $T = \{\tau_i\}_{1 \le i \le |T|}$ and $\tau_i := (loc_i, \lambda_i, loc_i', R_i)$. Then the Poisson process p_t^i has the rate of λ_i, and the matrix Q_i has a block diagonal structure as A and F, where

$$Q_i := \begin{bmatrix} 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & & \vdots \\ 0 & & -I & 0 & & 0 \\ 0 & & R_i & 0 & & 0 \\ \vdots & & & & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{matrix} \\ \\ \longleftarrow loc_i \\ \longleftarrow loc_i' \\ \\ \\ \end{matrix} \tag{46}$$

  that is, almost all the blocks are zero, except for two blocks:
  i) the diagonal block associated with loc_i, which is $-I$, and
  ii) the block whose row is associated with loc_i' and its column with loc_i, which is R_i.

The idea behind this procedure is as follows. We formulate a JLSS with $|L|$ invariant dynamics. That is, the state space can be written as the direct sum of $|L|$ subspaces, each of which is invariant with respect to the following dynamics:

$$dx_t = Ax_t dt + Fx_t dw_t. \tag{47}$$

Each invariant subspace represents a location in the LSHA. Further, we can observe that the origin is also invariant with respect to (47). As the result, if we start the evolution of the system in one of the invariant subspaces (hence, in one of the locations of the LSHA), the trajectory will remain in the subspace. Let us call the location l. When a Poisson process generates a point, if the process does not correspond to a transition whose source location is l, then the reset map does not change the continuous state of the system. This is due to the construction of (46). If the source location is l and the target is, say, l', then the continuous state is reset to another invariant space that corresponds to the location l'.

One apparent difference between the JLSS realization of the system and the original LSHA is that in the LSHA, only the Poisson processes in the active location are active. However, this difference does not affect the probabilistic properties of the trajectories, since Poisson processes are memoryless [16]. When we enter a location, it does not matter if we assume that the Poisson processes in the location are just started or that they have been running before.

V. Numerical Examples

In this section, we present some numerical examples, where the computation of the stochastic bisimulation functions is performed. We present two examples. The first example is about approximate abstraction of a JLSS. The second example is about approximate abstraction of an LSHA.

A. Approximate Abstraction of JLSS

The original system is a JLSS with sixth order linear dynamics. The system S is given as

$$S : \begin{cases} dx_t = Ax_t dt + Fx_t dw_t + Rx_t dp_t \\ y_t = Cx_t \end{cases} \tag{48}$$

where

$$A = \operatorname{diag}\left(\begin{bmatrix} -1 & -10 \\ 10 & -1 \end{bmatrix}, \begin{bmatrix} -2 & -20 \\ 20 & -1 \end{bmatrix}, \begin{bmatrix} -2 & 0 \\ 0 & -2.5 \end{bmatrix} \right)$$

$$F = 0.5 \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix},\ R = 0.7 \cdot I$$

$$C = \begin{bmatrix} 0.84 & -1.03 & 1.07 & -0.88 & 0.5 & 0 \\ -0.60 & -1.35 & -0.26 & -0.27 & 0 & -0.5 \end{bmatrix}.$$

The rate of the Poisson process p_t is 0.5.

We construct three kinds of abstraction, namely:
1) abstracting the continuous dynamics (4th order instead of 6th);
2) neglecting the influence of the Brownian motion;
3) neglecting the influence of the jump events (Poisson process).

For each of the above mentioned cases, we compute a stochastic bisimulation function between the original system and its abstraction. We then simulate several realizations of the composite system for the first 500 seconds of the evolution and plot the realizations of the error (between the observations). We then use Corollary 6 to establish a 90% confidence interval for the error.

1) Abstraction of the Continuous Dynamics: We construct a JLSS with simpler linear dynamics. Namely, we remove the last two modes of the original linear dynamics and hence create a fourth order linear system. Thus, we compute the stochastic bisimulation function between S and S' where

$$S' : \begin{cases} dx_t' = A'x_t' dt + F' dw_t + R'x_t' dp_t \\ y_t' = C'x_t', \end{cases}$$

$$A' = \operatorname{diag}\left(\begin{bmatrix} -1 & -10 \\ 10 & -1 \end{bmatrix}, \begin{bmatrix} -2 & -20 \\ 20 & -1 \end{bmatrix} \right), F' = 0.5 \cdot I$$

$$C' = \begin{bmatrix} 0.84 & -1.03 & 1.07 & -0.88 \\ -0.60 & -1.35 & -0.26 & -0.27 \end{bmatrix}, R' = 0.7 \cdot I. \tag{49}$$

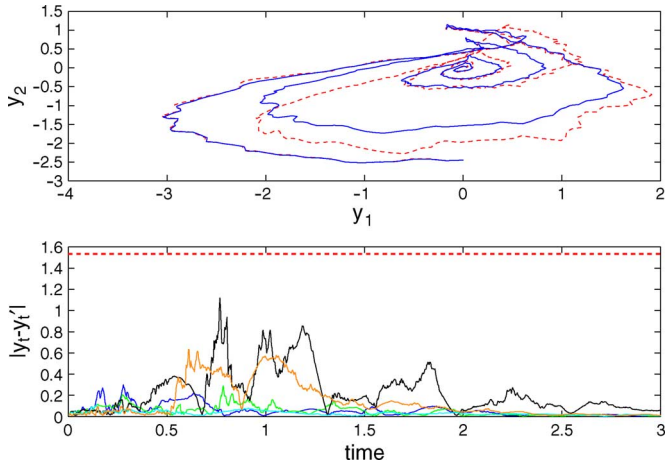


Fig. 2. Top: One realization of y_t (dashed) and y'_t (solid). Bottom: Five realizations of $\|y_t - y'_t\|$. The dashed line indicates the 75% confidence bound.

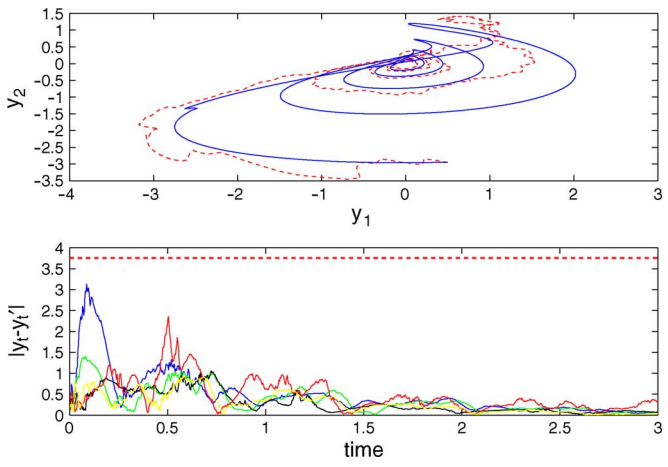


Fig. 3. Top: One realization of y_t (dashed) and y'_t (solid). Bottom: Ten realizations of $\|y_t - y'_t\|$. The dashed line indicates the 75% confidence bound.

In the simulation, the initial state of the original system is chosen as $[1,1,1,1,1,1]^T$ and the initial state of the approximation is the same as that of the original system without the last two components.

The computed bisimulation function is a quadratic function $x^T M x$, where

$$M = \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{bmatrix}$$

is a symmetric matrix with

$$M_{11} = \begin{bmatrix} 5.1441 & -0.1392 & 0.3387 & -0.5431 & -0.0103 \\ * & 5.4557 & 0.2 & 0.4068 & -0.3304 \\ * & * & 4.0435 & -0.0553 & 0.0441 \\ * & * & * & 3.8735 & -0.1853 \\ * & * & * & * & 1.0846 \end{bmatrix}$$

$$M_{12} = \begin{bmatrix} 0.0996 & -4.9 & 0.1364 & -0.3405 & 0.5387 \\ 0.1765 & 0.1336 & -5.2081 & -0.1979 & -0.4135 \\ -0.0840 & -0.3420 & -0.2003 & -3.9511 & 0.0589 \\ 0.1476 & 0.5424 & -0.4074 & -.0533 & -3.7816 \\ 0.4943 & 0.0119 & 0.3136 & -0.0427 & 0.1718 \end{bmatrix}$$

$$M_{22} = \begin{bmatrix} 1.0789 & -0.0894 & -0.2122 & 0.0847 & -0.1570 \\ * & 5.0742 & -0.1377 & 0.3445 & -0.5464 \\ * & * & 5.3785 & 0.2024 & 0.4135 \\ * & * & * & 4.0445 & -0.0556 \\ * & * & * & * & 3.8758 \end{bmatrix}.$$

Fig. 2 shows the simulation results. On the top part of the figure we see a realization of the observed process, y_t and y'_t. On the bottom part, we see five realizations of $\|y_t - y'_t\|$. The dashed line denotes the 75% confidence bound given by the computed stochastic bisimulation function (see Corollary 6).

2) Abstraction of the Brownian Motion: We construct an abstraction of S by neglecting the Brownian motion. We therefore create another system S', where

$$S' : \begin{cases} dx'_t = Ax'_t dt + Rx'_t dp_t \\ y'_t = Cx'_t. \end{cases}$$

In the simulation, the initial state of the original system is chosen as $[1,1,1,1,1,1]^T$ and the initial state of the approximation is the same as that of the original system.

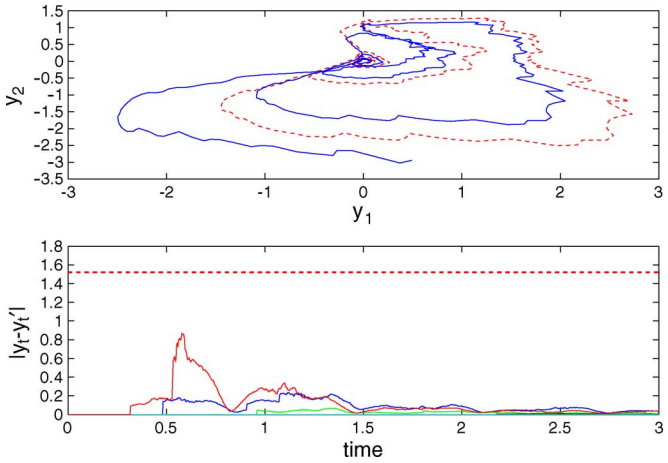


Fig. 4. Top: One realization of y_t (dashed) and y'_t (solid). Bottom: Ten realizations of $\|y_t - y'_t\|$. The dashed line indicates the 75% confidence bound.

On the top part of Fig. 3, we see a realization of y_t and y'_t. On the bottom part, five realizations of $\|y_t - y'_t\|$ are shown with the 75% confidence bound (see Corollary 6).

3) Abstraction of the Jump Events (Poisson Process): We construct an abstraction of S with zero R. That is, in the abstraction, we neglect the effect of the Poisson process. We therefore create another system $\tilde{S}$, where

$$\tilde{S} : \begin{cases} dx'_t = Ax'_t dt + Fx'_t dw_t \\ x'_t = Cx'_t. \end{cases}$$

In the simulation, the initial state of the original system is chosen randomly and the initial state of the approximation is the same as that of the original system.

Fig. 4 shows the simulation results. On the top part of the figure, we see a realization of y_t and y'_t. On the bottom part, five realizations of $\|y_t - y'_t\|$ are plotted with the 75% confidence bound (see Corollary 6). In this plot we can see clearly that y_t and y'_t coincide until the Poisson process generates a jump, which is accomodated in the original system, but not in the abstraction.

4) Discussion: As mentioned earlier, the stochastic bisimulation function can be used to provide an upper bound on how

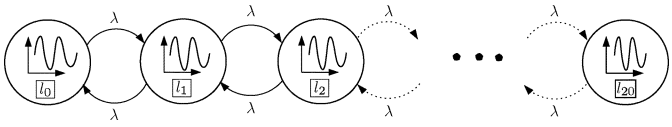


Fig. 5. Chain-like automaton $\mathcal{A}$ with 21 locations.

much the observations can differ. The 75% confidence interval that we constructed can be used to indicate how tight the upper bound is. We can observe that while the bound is not very tight, it is still in the right order of magnitude. Computation of tighter valid bounds can probably be obtained by considering more general construction of the stochastic bisimulation functions, i.e. not necessary quadratic functions, which might involve more complicated computation. Candidates for such a bisimulation function are matrix polynomials with degree higher than two.

B. Approximate Abstraction of LSHA

Here we present an example, where we apply the framework of approximate abstraction of linear stochastic hybrid automata. The original automaton $\mathcal{A}$ has a chain like structure, with 21 locations. See Fig. 5.

Chain-like automata is a structure that can be found, for example in modeling of systems that involve birth and death process. That is, each location represents the number of a certain object in the system, for example, persons in a queue or molecules in a chemical reaction. The underlying dynamics of the system is influenced by the number of such objects. However, it is usually reasonable to assume that the continuous dynamics does not abruptly change with the change in the number of objects. Researchers have been working towards approximating such systems in a way that allows for both fast and accurate simulations [9], as well as faster computation [38].

Adjacent locations in the automaton $\mathcal{A}$ are connected by a pair of transitions with constant rate $\lambda = 0.02$. The continuous dynamics of $\mathcal{A}$ is such that the dynamics changes gradually from location l_0 to location l_{20}. The stochastic differential equation that describes the dynamics in location l_i, $0 \leq i \leq 20$, is as follows:

$$\begin{aligned} dx_{i,t} &= A_i x_{i,t} dt + F_i x_t dw_t \\ y_t &= C_i x_{i,t}, \text{ where} \\ A_i &= \begin{bmatrix} -0.01 & -0.1(1+\alpha\cdot i) \\ 0.1(1+\alpha\cdot i) & -0.01 \end{bmatrix} \\ F_i &= \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \; C_i = [0 \quad 1], \; i = 0 \ldots 20. \end{aligned}$$

We are going to apply the procedure for several values of α.

We can easily observe that the continuous dynamics in each location is a damped 2-dimensional oscillator driven by Brownian motion. A realization of the output of $\mathcal{A}$ is plotted in Fig. 6. As we go from location l_0 to l_{20}, the frequency of the oscillation increases. We want to see if we can approximate $\mathcal{A}$ with another automaton $\mathcal{A}'$ that has only one location by abstracting the influence of the Poisson process. The continuous dynamics of $\mathcal{A}'$ is the same as that in location l_{10} of $\mathcal{A}$. Hence we compute a stochastic bisimulation function between

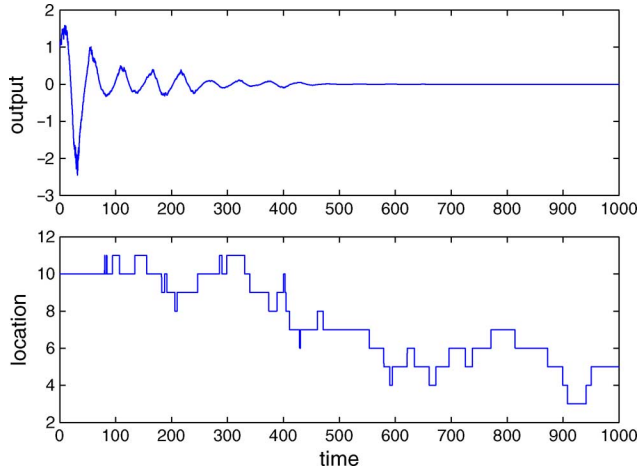


Fig. 6. Realization of the output trajectory (top) and the location (bottom) of the linear stochastic hybrid automaton $\mathcal{A}$.

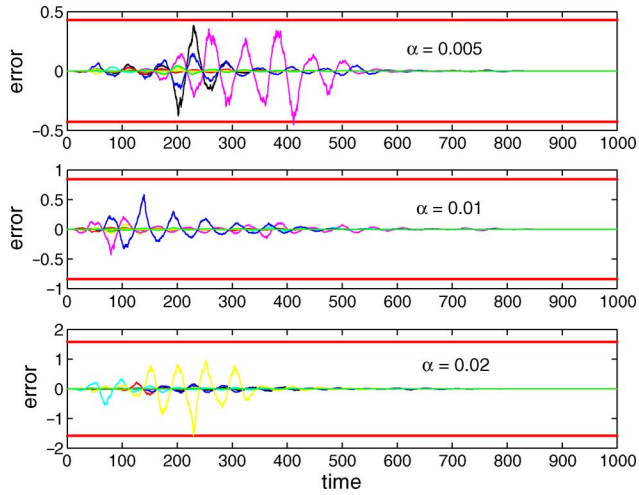


Fig. 7. Ten realizations of the error trajectory for each of the α value. The parallel lines indicate the 90% confidence interval stipulated by the stochastic bisimulation functions.

$\mathcal{A}$ and $\mathcal{A}'$. The computation is done by solving the linear matrix inequality problem explained in the previous section.

Three different values for α are used, namely 5×10^{-3}, 10^{-2}, and 2×10^{-2}. For these values of α, the ratio between the oscillation frequency in location l_{20} and l_0 are 1.1, 1.2, and 1.4 respectively. We simulate the execution of the original automaton $\mathcal{A}$ and its abstraction $\mathcal{A}'$. In the simulation we use $[1\ 1]^T$ as the initial condition for the continuous dynamics, and assume that automaton $\mathcal{A}$ starts in location l_{10}. With the computed stochastic bisimulation function, we can also compute the 90% confidence interval for the error between the outputs of $\mathcal{A}$ and $\mathcal{A}'$ (see Corollary 6).

In Fig. 7 we can see ten realizations of the error trajectory for each of the value of α. The 90% confidence intervals are also shown. We can observe that the quadratic stochastic bisimulation function seems to give a good estimate for the error, as the confidence intervals seem quite tight. We can also observe that as the dynamics in the locations vary more, the error in the approximation becomes larger.

VI. Conclusion

In this paper we present a framework for approximate abstraction of a class of stochastic hybrid systems. The idea is based on the construction of the so called stochastic (bi)simulation functions that can be used for establishing an upper bound on how much the observations of a system and its abstraction can differ.

For the two classes of systems presented in this paper, the jump linear stochastic systems (JLSS) and the linear stochastic hybrid automata (LSHA), we can assume that a stochastic simulation function assumes the form of a quadratic function. Based on this assumption, the computation of a stochastic simulation function can be cast as a linear matrix inequality (LMI) problem, that can be solved using available tools. We demonstrate the use of the methodology presented in this paper by presenting several numerical examples in the previous section.

The assumption that we can find a stochastic simulation function in the form of a quadratic function is a consequence of the linear dynamics and linear reset map of the systems. If we want to relax this assumption, then we need to resort to a more general computational framework, most likely at a cost of higher computational complexity. For example, if we assume that all the functions involved are polynomials, then a stochastic simulation function can be searched in the space of polynomial functions, using tools such as SOSTOOLS [39]. This idea stems from the fact that for polynomial functions, the inequalities in Theorem 8 can be written as sum-of-squares program.

In the computational results that we reported in this paper, we assume that nondeterminism is not present. This assumption is taken in order to simplify the computation. However, the general result presented in this paper is not restricted to that particular case. We identify the issue of finding a suitable computational implementation of the theory, where nondeterminism is taken into account as a challenging research direction. We also reiterate the statement in Remark 4 that in the presence of nondeterminism, there can be more than one interpretation of (bi)simulation. The relation between these interpretations is also an interesting research topic, as it can lead to multiple interpretation of approximate abstraction of stochastic systems[40], [41].

Acknowledgment

The authors would like to thank A. Girard for the valuable discussions and input during the research reported in this paper, B. Krogh for a suggestion on relating JLSS and LSHA, the associate editor J. Hespanha for his inputs on a number of technical issues in this paper, and the anonymous reviewers for their constructive and helpful remarks.

References

[1] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *IEEE Trans. Intell. Transportation Syst.*, vol. 1, no. 4, pp. 199–220, 2000.

[2] G. Pola, M. Bujorianu, J. Lygeros, and M. D. Benedetto, "Stochastic hybrid models: An overview," in *Proc. IFAC Conf. Anal. Design Hybrid Syst.*, St. Malo, France, 2003, pp. 45–50.

[3] W. Glover and J. Lygeros, "A stochastic hybrid model for air traffic control simulation," in *HSCC*, ser. Lecture Notes in Computer Science, R. Alur and G. J. Pappas, Eds. New York: Springer Verlag, 2004, vol. 2993, pp. 372–386.

[4] S. Strubbe, A. J. van der Schaft, and A. A. Julius, "Value passing for communicating piecewise deterministic Markov processes," in *Proc. Amer. Control Conf.*, Minneapolis, MN, 2006.

[5] J. Hu, W. C. Wu, and S. Sastry, "Modeling subtilin production in bacillus subtilis using stochastic hybrid systems," in *HSCC*, ser. Lecture Notes in Computer Science, R. Alur and G. J. Pappas, Eds. New York: Springer Verlag, 2004, vol. 2993, pp. 417–431.

[6] J. Lygeros, X. Mao, and C. Yuan, "Stochastic hybrid delay population dynamics," in *HSCC*. New York: Springer Verlag, 2006, pp. 436–450.

[7] A. Singh and J. P. Hespanha, "Moment closure techniques for stochastic models in population biology," in *Proc. Amer. Control Conf.*, Minneapolis, MN, 2006, pp. 4730–4735.

[8] H. El-Samad, M. Fazel, X. Liu, A. Papachristodoulou, and S. Prajna, "Stochastic reachability analysis in complex biological networks," in *Proc. Amer. Control Conf.*, Minneapolis, MN, 2006, pp. 4748–4753.

[9] N. A. Neogi, "Dynamic partitioning of large discrete event biological systems for hybrid simulation and analysis," in *HSCC*, ser. Lecture Notes in Computer Science, R. Alur and G. J. Pappas, Eds. New York: Springer Verlag, 2004, vol. 2993, pp. 463–476.

[10] J. P. Hespanha, "Stochastic hybrid systems: applications to communication networks," in *HSCC*, ser. Lecture Notes in Computer Science, R. Alur and G. J. Pappas, Eds. New York: Springer Verlag, 2004, vol. 2993, pp. 387–401.

[11] J. Hu, "Application of stochastic hybrid systems in power management of streaming data," in *Proc. Amer. Control Conf.*, Minneapolis, MN, 2006, pp. 5754–5759.

[12] J. Hu, J. Lygeros, and S. Sastry, "Towards a theory of stochastic hybrid systems," in *Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, N. Lynch and B. H. Krogh, Eds. New York: Springer Verlag, 2000, vol. 1790, pp. 160–173.

[13] B. Oksendal, *Stochastic Differential Equations: An Introduction With Applications*. Berlin, Germany: Springer-Verlag, 2000.

[14] M. K. Ghosh, A. Arapostathis, and S. Marcus, "Ergodic control of switching diffusions," *SIAM J. Control Optim.*, vol. 35, no. 6, pp. 1952–1988, 1997.

[15] M. K. Ghosh and A. Bagchi, "Modeling stochastic hybrid systems," in *System Modeling and Optimization*, J. Cagnol and J. P. Zolésio, Eds. Norwell, MA: Kluwer Academic Publisher, 2003, pp. 269–280, IFIP.

[16] M. H. A. Davis, *Markov Models and Optimization*. London, U.K.: Chapman and Hall, 1993.

[17] J. Krystul, H. A. P. Blom, and A. Bagchi, "Stochastic differential equations on hybrid state spaces," in *Stochastic Hybrid Systems*, ser. Automation and Control Engineering. Orlando, FL: Taylor & Francis CRC Press, 2006, pp. 15–46.

[18] A. Girard and G. J. Pappas, "Approximate bisimulation for constrained linear systems," in *Proc. IEEE Conf. Decision Control*, Seville, Spain, 2005, pp. 4700–4705.

[19] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Trans. Automat. Control*, vol. 52, no. 5, pp. 782–798, May 2007.

[20] G. J. Pappas, "Bisimilar linear systems," *Automatica*, vol. 39, pp. 2035–2047, Dec. 2003.

[21] A. J. van der Schaft, "Equivalence of dynamical systems by bisimulation," *IEEE Trans. Automat. Control*, vol. 49, no. 12, pp. 2160–2172, Dec. 2004.

[22] M. L. Bujorianu, J. Lygeros, and M. C. Bujorianu, "Bisimulation for general stochastic hybrid systems," in *HSCC*, ser. Lecture Notes in Computer Science, M. Morari and L. Thiele, Eds. New York: Springer Verlag, 2005, vol. 3414, pp. 198–214.

[23] S. Strubbe and A. J. van der Schaft, "Bisimulation for communicating piecewise deterministic Markov processes," in *HSCC*, ser. Lecture Notes in Computer Science, M. Morari and L. Thiele, Eds. New York: Springer Verlag, 2005, vol. 3414, pp. 623–639.

[24] M. Ying and M. Wirsing, "Approximate bisimilarity," in *AMAST 2000*, ser. Lecture Notes in Computer Science, T. Rus, Ed. New York: Springer Verlag, 2000, vol. 1816, pp. 309–322.

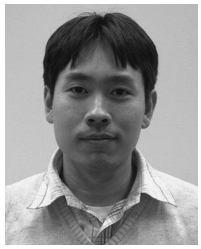
[25] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden, "Metrics for labelled Markov processes," *Theoret. Comput. Sci.*, vol. 318, no. 3, pp. 323–354, 2004.

[26] F. van Breugel and J. Worrell, "An algorithm for quantitative verification of probabilistic transition systems," in *Proc. CONCUR*, 2001, pp. 336–350.

[27] A. A. Julius, A. Girard, and G. J. Pappas, "Approximate bisimulation for a class of stochastic hybrid systems," in *Proc. Amer. Control Conf.*, Minneapolis, MN, 2006, pp. 4724–4729.

[28] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho, *Hybrid Automata: An Algorithmic Approach to the Specification and Verification of Hybrid Systems*, ser. Lecture Notes in Computer Science. New York: Springer, 1993, vol. 736, pp. 209–229.

[29] J. P. Hespanha and A. Teel, "Stochastic impulsive systems driven by renewal processes," in *Proc. 17th Int. Symp. Math. Theory Networks Syst.*, Kyoto, Japan, 2006 [Online]. Available: http://www.ece.ucsb.edu/~hespanha/techrep.html
[30] H. J. Kushner, *Stochastic Stability and Control*, ser. Mathematics in Science and Engineering. New York: Academic Press, 1967.
[31] S. Prajna, A. Jadbabaie, and G. J. Pappas, "Stochastic safety verification using barrier certificates," in *Proc. 43rd IEEE Conf. Decision Control*, 2004, pp. 929–934.
[32] R. Milner, *Communication and Concurrency*. Englewood Cliffs, NJ: Prentice Hall International, 1989.
[33] A. Girard and G. J. Pappas, Approximate (Bi)-Simulations for Linear Systems With Constrained Inputs 2005.
[34] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P. H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *Theoret. Comput. Sci.*, vol. 138, pp. 3–34, 1995.
[35] J. Löfberg, "YALMIP: A toolbox for modeling and optimization in MATLAB," in *Proc. CACSD Conf.*, 2004 [Online]. Available: http://control.ee.ethz.ch/~joloef/yalmip.php
[36] S. Boyd and M. C. Grant, `cvx`—MATLAB Software for Disciplined Convex Programming 2005 [Online]. Available: http://www.stanford.edu/~boyd/cvx/
[37] P. Florchinger, "Lyapunov-like techniques for stochastic stability," *SIAM J. Control Optim.*, vol. 33, pp. 1151–1169, 1995.
[38] J. P. Hespanha, "Polynomial stochastic hybrid systems," in *HSCC*, ser. Lecture Notes in Computer Science, M. Morari and L. Thiele, Eds. New York: Springer Verlag, 2005, vol. 3414, pp. 322–338.
[39] S. Prajna and A. Papachristodoulou, "Introducing SOSTOOLS: A general purpose sum of squares programming solver," in *Proc. IEEE Conf. Decision Control*, Las Vegas, NV, 2002, pp. 741–746.
[40] *Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, M. Morari and L. Thiele, Eds. New York: Springer Verlag, 2005, vol. 3414.
[41] *Hybrid Systems: Computation and Control*, ser. Lecture Notes in Computer Science, R. Alur and G. J. Pappas, Eds. New York: Springer Verlag, 2004, vol. 2993.



A. Agung Julius (S'97–M'06) received the B.Eng. degree from the Institut Teknologi Bandung, Indonesia, in 1998 and the M.Sc. and Ph.D. degrees in applied mathematics from Universiteit Twente, The Netherlands, in 2001 and 2005, respectively.

He is an Assistant Professor with the Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY, since December 2008. He was previously a Postdoctoral Researcher at the GRASP Laboratory, School of Engineering and Applied Sciences, University of Pennsylvania, Philadelphia. His research interests include hybrid systems, systems biology, and systems and control theory.

George J. Pappas (S'91–M'98–SM'04–F'08) received the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, in 1998.

He is currently a Professor with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, where he also holds secondary appointments with the Departments of Computer and Information Sciences and Mechanical Engineering and Applied Mechanics. He is a member and a former Director of the General Robotics and Active Sensory Perception (GRASP) Laboratory and is currently the Deputy Dean in the School of Engineering and Applied Science. He is the Co-Editor of *Hybrid Systems: Computation and Control* (New York: Springer-Verlag, 2004). His current research interests include the areas of hybrid and embedded systems, hierarchical control systems, distributed control systems, nonlinear control systems, and geometric control theory, with applications to robotics, unmanned aerial vehicles, and biomolecular networks.

Dr. Pappas received numerous awards including the National Science Foundation (NSF) CAREER Award in 2002, the NSF Presidential Early Career Award for Scientists and Engineers (PECASE) in 2002, and the Eliahu Jury Award for Excellence in Systems Research from the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in 1999. He was among the finalists for the Best Student Paper Award at the IEEE Conference on Decision and Control in 1998, 2001, 2004, and 2006, the American Control Conference in 2001 and 2004, and the IEEE Conference on Robotics and Automation in 2007.