Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-5 (2021) 283–288

Bounds on set exit times of affine systems, using Linear Matrix Inequalities ⋆

Guillaume O. Berger * **Maben Rabi** **

* *Université catholique de Louvain (UCLouvain), B-1348 Louvain-la-Neuve, Belgium (e-mail: guillaume.berger@uclouvain.be)*

** *Østfold University College, NO-1757 Halden, Norway (e-mail: maben.rabi@hiof.no)*

Abstract: Efficient computation of trajectories of switched affine systems becomes possible, if for any such hybrid system, we can manage to efficiently compute the sequence of switching times. Once the switching times have been computed, we can easily compute the trajectories between two successive switches as the solution of an affine ODE. Each switching time can be seen as a positive real root of an analytic function, thereby allowing for efficient computation by using root finding algorithms. These algorithms require a finite interval, within which to search for the switching time. In this paper, we study the problem of computing upper bounds on such switching times, and we restrict our attention to stable time-invariant affine systems. We provide semi-definite programming models to compute upper bounds on the time taken by the trajectories of an affine ODE to exit a set described as the intersection of a few generalized ellipsoids. Through numerical experiments, we show that the resulting bounds are tighter than bounds reported before, while requiring only a modest increase in computation time.

Keywords: Hybrid and switched systems modelling, reachability analysis, verification and abstraction of hybrid systems

1. INTRODUCTION

Simulation of hybrid systems has been quite an important topic of research in recent years; see, e.g., Carloni et al. (2006); Goebel et al. (2009); Schweiger et al. (2019) for surveys. Hybrid systems are generally characterized by the presence of piecewise continuous dynamics. A major challenge for the simulation of such systems is thus to determine the switching times of trajectories, that is, when they hit a *switching surface* (boundary between two continuous components of the vector flow) (Cremona et al., 2019). Most hybrid system simulation tools rely on step refinement algorithms for reliably detecting the event of the trajectory crossing a boundary (Esposito et al., 2001; Wang et al., 2015; Copp and Sanfelice, 2016; Farkas et al., 2019). These methods are variations of classical ODE solvers, which also use step refinement techniques to compute accurate solutions of continuous ODE flows. But these methods handle both linear flows and nonlinear flows in the same way; nor do they discriminate between linear, ellipsoidal, or general nonlinear and non-convex switching surfaces.

On the other hand, some classes of systems allow for a *piecewise analytic* expression of their solutions, making the use of numerical-integration-based ODE solvers superfluous. This is the case for instance for autonomous *switched affine systems,* for which the trajectories of each individual mode can be expressed using elementary transcendental functions of the type: e^{at}, $\sin(\omega t)$, $\cos(\omega t)$ and t^k. These systems constitute a paradigmatic class of hybrid and cyber-physical systems, and appear naturally in many engineering applications or as abstractions of more complicated systems; see, e.g., Heemels et al. (2001); Bell et al. (2010); Legat et al. (2020). The recent work of Rabi (2020) proposes to use the piecewise analytic expression of the trajectories and to combine it with root-finding algorithms for the detection of boundary crossing. This allows to benefit from the long-standing maturity of root-finding solvers for transcendental equations (Boyd, 2014), in order to compute fast and accurate solutions for these systems.

The main challenge of the above approach is to produce reliable time intervals for the crossing of a switching surface by the trajectory, as root-finding algorithms generally require a bounded interval in which to search for potential roots of the analytic function. In Rabi (2020), upper bounds on the crossing time are computed using Lyapunov functions. The idea is that if the current mode of the system is stable and its equilibrium lies away from the switching surface, then after some time the trajectory will enter an invariant region not intersecting the switching surface. Quadratic Lyapunov functions are used to compute such invariant regions and upper bounds on the time for reaching them. The approach in Rabi (2020) relies on arbitrarily restricting the Lyapunov equation to have a simple right-hand side, as in $A^\top P + PA = -I$. A geometric interpretation of the resulting Lyapunov function leads to upper bounds on the switching time that grow quadratically with the distance of the initial point to the boundary.

⋆ The work of the first author is supported by a FRIA (F.R.S.–FNRS) fellowship.

10.1016/j.ifacol.2021.08.512

In this work, we push further the above approach by combining it with convex optimization techniques to reduce its conservativeness. The idea is to use semi-definite programs (SDPs) to compute Lyapunov-like functions, and invariant regions that are not necessarily centered at the equilibrium. The SDPs shall be formulated to provide tight upper bounds on the crossing times. The upper bounds obtained in this way are optimal within the considered framework of Lyapunov-like functions, and grow either quadratically or logarithmically with the distance of the initial point to the boundary, depending on the SDP formulation. When our method is used to compute switched system trajectories, each continuous mode is associated with a corresponding SDP, whose solution can be computed off-line.

We study two such optimization models for the computation of upper bounds on the switching time of switched affine systems, and compare them in terms of computation time and tightness of the bounds. We also show improvements over the bounds of Rabi (2020). Our comparisons are empirical and are based on a large set of random matrices. We observe that more complex models indeed lead to substantially tighter bounds while not requiring a significant increase of the computation time (which can be performed off-line if needed). We focus on the comparison of the different models for computing upper bounds on the switching times, and we leave for further work the implementation and tests of these methods in a standalone software for the simulation of switched affine systems.

The paper is organized as follows. In Section 2, we introduce the problem of interest. In Section 3, we present the different models for the computation of upper bounds on the switching time. In Section 4, we present the numerical experiments for the comparisons of the different models.

Notation. For $N \in \mathbb{N}_{>0}$, we let $[N] = \{1, \dots, N\}$. For $A \in \mathbb{R}^{n\times n}$, $\sigma(A)$ denotes the *stability margin* of A, that is, $\sigma(A) = \min\{-\mathrm{Re}(\lambda) : \lambda \text{ is an eigenvalue of } A\}$.

2. PROBLEM FORMULATION

Consider an affine system $\dot{x}(t) = Ax(t)+b$, with $A \in \mathbb{R}^{n\times n}$ strictly stable and $b \in \mathbb{R}^n$. Let $\mathcal{R} \subseteq \mathbb{R}^n$ be a closed convex region with nonempty interior. For $x_0 \in \mathcal{R}$, we define the *escaping time from* $\mathcal{R}$ of the trajectory starting at x_0, by

$$t_*(x_0, \mathcal{R}; A, b) = \min\{t \geq 0 : \xi(t, x_0) \in \mathrm{bd}\,\mathcal{R}\},$$

where $\xi(t, x_0)$ is the trajectory of the affine system given by A and b, starting from x_0. If $\xi(t, x_0) \in \mathrm{int}\,\mathcal{R}$ for all $t \geq 0$, we let $t_*(x_0, \mathcal{R}; A, b) = 0$.

If the system is a switched affine system, $\dot{x} = A_i x + b_i$ when $x \in \mathcal{R}_i$, then the escaping times of $\dot{x} = A_i x + b_i$ from the regions $\mathcal{R}_i$ gives the switching times of the trajectory. The convention that $t_*(x_0, \mathcal{R}_i; A_i, b_i) = 0$ if the trajectory remains in $\mathcal{R}_i$ is motivated by the problem of computing the switching time using root-finding algorithms, for which we seek upper bounds as small as possible on the interval in which the first crossing, *if it exists*, occurs.

To simplify the notation, we will assume in the rest of the paper that each mode is *linear*, i.e., has the form $\dot{x} = A_i x$. Thus we will focus on the problem of finding upper bounds on the escaping time of the LTI system $\dot{x} = Ax$, where A is strictly stable. This conversion of stable affine systems into stable linear systems can be done, *without loss of generality*, by translating the state x and the region $\mathcal{R}$ around the equilibrium point $\bar{x} = -A^{-1}b$.

We also make the following assumptions on the region $\mathcal{R}$:

- $\mathcal{R}$ is described as the intersection of a finite set of ellipsoids [1]:
$$\mathcal{R} = \textstyle\bigcap_{i=1}^{N} \mathcal{E}_i,$$
where each $\mathcal{E}_i$ is an ellipsoid. Note that this includes the case of a polyhedral region $\mathcal{R}$.
- $\mathcal{R}$ is included in a set described as the convex hull of a finite set of ellipsoids or singletons:
$$\mathcal{R} \subseteq \mathrm{conv}\,\{\mathcal{F}_1, \dots, \mathcal{F}_M\},$$
where each $\mathcal{F}_i$ is an ellipsoid or a singleton.
- The origin does not belong to the boundary of $\mathcal{R}$.

Remark 1. In our analysis, we assume that the origin is not on the boundary of $\mathcal{R}$ and that A is strictly stable. This implies that for every $x_0 \in \mathcal{R}$ there is a finite time $T_* \geq 0$ such that for all times $t \geq T_*$, the state $\xi(t, x_0)$ is bounded away from the boundary of $\mathcal{R}$. On the other hand, if the origin was on the boundary of $\mathcal{R}$, or if A was unstable or marginally stable, then there could be situations in which the trajectories of the system tend to the boundary of $\mathcal{R}$ but never reach it, or tend to the boundary of $\mathcal{R}$ and cross it an infinite number of times (e.g., convergence with rotation); so that this more general class of matrices would be very difficult to handle for the problem of escaping time estimation, at least with the techniques described in this work. Some techniques, based on the eigenvalue decomposition, were proposed in Rocca et al. (2015) to exploit the rotation of the trajectories to detect reachability of LTI systems. However, since even with these techniques it is not guaranteed to obtain a finite upper bound on the escaping time, we preferred to focus here on stable affine systems.

Remark 2. We assume that the convex set $\mathcal{R}$ can be enclosed in the convex hull of a set of ellipsoids or singletons. This includes the two extreme cases: when $\mathcal{R}$ is enclosed in a single ellipsoid and when it is enclosed in the convex hull of a finite set of points. When the number of points increases, the second case can provide arbitrarily accurate enclosures of $\mathcal{R}$. However, for some regions (e.g., the hypercube), the number of points needed to describe the convex hull grows exponentially with the dimension. In these cases, it may be beneficial to consider enclosures described by a few ellipsoids (instead of many points), even if it provides less accurate enclosures of $\mathcal{R}$.

3. UPPER BOUNDS ON SWITCHING TIMES

In this section, we present two frameworks, inspired by Lyapunov theory, to compute upper bounds on the escaping time of stable LTI systems. By computing Lyapunov-like functions that have guaranteed rates of decrease inside $\mathcal{R}$, we can find invariant regions not intersecting the boundary of $\mathcal{R}$ and obtain upper bounds on the time for

[1] In this paper, the term *ellipsoid* is understood in the broad sense, meaning that an ellipsoid can be degenerate (or unbounded in some directions). However, an ellipsoid is always assumed to be closed and to have nonempty interior.

the trajectories to enter the invariant region. To compute these functions, we follow the approach of quadratic Lyapunov theory, and formulate their existence as the feasibility of a convex optimization program in which the constraints on the functions are encoded as *Linear Matrix Inequalities* (LMIs).

3.1 Quadratic functions as decision variables

A *quadratic function* (on $\mathbb{R}^n$) is a function of the form $V(x) = x^\top Q x + 2b^\top x + c$ with $Q = Q^\top \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$. A quadratic function $V(x) = x^\top Q x + 2b^\top x + c$ is *convex* if $Q \succeq 0$. Given a quadratic function V and a scalar $s \in \mathbb{R}$, we let $\mathrm{Sub}_s(V)$ be the *sublevel set* $\{x \in \mathbb{R}^n : V(x) \leq s\}$. Note that any ellipsoid (possibly degenerate) can be described as $\mathrm{Sub}_0(V)$ for some convex quadratic function V. A quadratic function V is *nonnegative* if $V(x) \geq 0$ for all $x \in \mathbb{R}^n$. This will be denoted by $V \geqq 0$. We also use the notation $V_1 \geqq V_2$ for $V_1 - V_2 \geqq 0$, where V_1, V_2 are two quadratic functions.

In our framework, we will use quadratic functions as *decision variables*. More precisely, we will define optimization problems where the objective is to find unknown quadratic functions that have to satisfy several constraints. Because quadratic functions are parameterized by Q, b and c, finding them will in fact amount to compute the associated Q, b and c. Moreover, the constraints on the quadratic functions will always have the form $V_1 \geqq V_2$. The proposition below states that this type of constraints can be encoded as LMIs, so that the unknown quadratic functions can be computed using Semi-Definite Programming.

Proposition 3. The quadratic function $V(x) = x^\top Q x + 2b^\top x + c$ is nonnegative ($V \geqq 0$) if and only if $\begin{bmatrix} Q & b \\ b^\top & c \end{bmatrix} \succeq 0$.

Proof. This follows from $V(x) = [x\,;1]^\top \begin{bmatrix} Q & b \\ b^\top & c \end{bmatrix} [x\,;1]$. See for instance Boyd and Vandenberghe (2004). □

Given a matrix A and a quadratic function V, the *Lie derivative* of V along the vector field $x \mapsto Ax$ is defined as $\mathcal{L}_A V(x) = V'(x)Ax$. Observe that if $V(x) = x^\top Q x + 2b^\top x + c$, then $\mathcal{L}_A V(x) = x^\top (A^\top Q + QA)x + 2b^\top Ax$; in particular, $\mathcal{L}_A V$ depends linearly on V.

In the next subsections, we introduce optimization models to compute upper bounds on the set escaping times of stable linear systems. For the sake of readability, these models are presented using quadratic functions as variables. From the above discussion, these models can be reformulated as Semi-Definite Programs in the associated variables Q, b and c, thereby allowing for efficient computation of the solution, using for instance interior-point algorithms (Ben-Tal and Nemirovski, 2001; Boyd and Vandenberghe, 2004).

In the following, we let $F_1, \dots, F_M$ and $E_1, \dots, E_N$ be convex quadratic functions whose 0-sublevel sets are equal to the ellipsoids describing $\mathcal{R}$ (see Section 2); i.e., $\mathrm{Sub}_0(F_i) = \mathcal{F}_i$ for all $i \in [M]$, $\mathrm{Sub}_0(E_i) = \mathcal{E}_i$ for all $i \in [N]$.

3.2 Upper bound on escaping time. Case I: $0 \in \operatorname{int}\mathcal{R}$

This is the case where the origin is in the interior of the region $\mathcal{R}$. Consider the following optimization program.

Decision variables: convex quadratic functions V and W, and scalars $\lambda_1, \dots, \lambda_M \geq 0$, $\mu_1, \dots, \mu_N \geq 0$, $\nu_1, \dots, \nu_N \geq 0$ and $r \in \mathbb{R}$; *Objective and constraints:*

$$\begin{aligned}
\max \quad & r && \text{(1a)}\\
\text{s.t.} \quad & V \leqq 1 + \lambda_i F_i, \quad \forall i \in [M], && \text{(1b)}\\
& V \geqq r + \mu_i E_i, \quad \forall i \in [N], && \text{(1c)}\\
& \mathcal{L}_A W \leqq 0, && \text{(1d)}\\
& W \geqq \nu_i E_i, \quad \forall i \in [N], && \text{(1e)}\\
& \mathcal{L}_A V \leqq G(V) - W, && \text{(1f)}
\end{aligned}$$

where $G(V)$ is equal to either -1 (leading to quadratically increasing bounds), or to $-2\gamma V$ for some fixed parameter $\gamma > 0$ (leading to logarithmically increasing bounds).

Explanations. The goal of these constraints is to find an ellipsoidal invariant region, defined by $\mathrm{Sub}_0(W)$ where W is a convex quadratic function, that is included in $\mathcal{R}$. This is enforced by the constraints (1d) (implying that the Lie derivative of W is nonpositive everywhere, so that any sublevel set of W is invariant) and (1e) (implying that, for all $i \in [N]$, $E_i(x) < 0$ whenever $W(x) < 0$ since $\nu_i \geq 0$, so that $\mathrm{Sub}_s(W) \subseteq \operatorname{int}\mathcal{R}$ for all $s < 0$). Using this invariant region, we seek a quadratic function V whose Lie derivative is smaller than $G(V)$ for all points outside the invariant region $\mathrm{Sub}_0(W)$. This is enforced by (1f) (implying that $\mathcal{L}_A V(x) \leq G(V(x))$ whenever $W(x) \geq 0$). We also require that $\mathcal{R} \subseteq \mathrm{Sub}_1(V)$. This is enforced by (1b) (implying that, for all $i \in [M]$, $V(x) \leq 1$ whenever $F_i(x) \leq 0$ since $\lambda_i \geq 0$). Finally, we try to find the largest r such that $\mathrm{Sub}_r(V) \subseteq \mathcal{R}$. This is enforced by (1c) (implying that $\mathrm{Sub}_s(V) \subseteq \operatorname{int}\mathcal{R}$ for any $s < r$; similarly to the case of (1e)). See also Figure 1 and Example 5 for an illustration.

The significance of the above optimization program (1) is that any feasible solution provides an upper bound on the escaping time from $\mathcal{R}$:

Theorem 4. Assume $0 \in \operatorname{int}\mathcal{R}$. Let $(r, V, \dots)$ be a feasible solution [2] of (1) with $G(V) = -1$. Then, for any $x_0 \in \mathcal{R}$,

$$t_*(x_0, \mathcal{R}; A) \leq \max(V(x_0) - r, 0) \leq \max(1 - r, 0).$$

Similarly, let $(r, V, \dots)$ be a feasible solution of (1) with $G(V) = -\gamma V$, such that $r > 0$. Then, for any $x_0 \in \mathcal{R}$,

$$t_*(x_0, \mathcal{R}; A) \leq \frac{\log^+\left(\frac{V(x_0)}{r}\right)}{2\gamma} \leq \frac{\log^+\left(\frac{1}{r}\right)}{2\gamma},$$

where $\log^+(\alpha) \triangleq \log(\max(\alpha, 1))$.

Proof. Let $x_0 \in \mathcal{R}$ and denote $\tau_* = t_*(x_0, \mathcal{R}; A)$. Assume $\tau_* > 0$, as otherwise the inequalities are trivially satisfied. Let $x(\cdot)$ be the trajectory of the system $\dot{x} = Ax$, starting from x_0. Then, for all $t \in [0, \tau_*]$, it holds that $W(x(t)) \geq 0$; indeed, otherwise $x(t)$ would be in an invariant set inside $\operatorname{int}\mathcal{R}$ (see "Explanations"). First, consider the model with $G(V) = -1$. For all $t \in [0, \tau_*]$, it holds that $V(x(t)) \geq r$ and $\dot{V}(x(t)) \leq -1$, by (1b), (1f) and the fact that $W(x(t)) \geq 0$. This implies that

$$r \leq V(x(\tau_*)) = V(x_0) + \int_0^{\tau_*} \dot{V}(x(t))\,\mathrm{d}t \leq V(x_0) - \tau_*.$$

[2] Note that r can be negative.

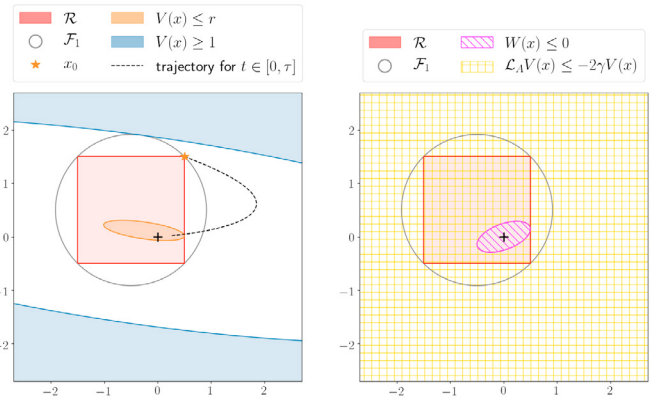

Fig. 1. Illustration of Example 5. We verify that $\mathcal{R} \subseteq \mathrm{Sub}_1(V)$, $\mathrm{Sub}_r(V) \subseteq \mathcal{R}$, and $\mathcal{L}_A V(x) \le G(V(x))$ whenever $W(x) \ge 0$.

Hence, we find that $\tau_* \le V(x_0) - r$. Finally, the inequality $\tau_* \le 1 - r$ comes from the fact that $V(x_0) \le 1$ for every $x_0 \in \mathcal{R}$ (see "Explanations").

The proof for the model with $G(V) = -\gamma V$ is along the same lines: we get that, for all $t \in [0, \tau^*]$,

$$V(x(t)) \le V(x_0) + \int_0^t -2\gamma V(x(s))\,\mathrm{d}s.$$

The classical argument (like Grönwall's inequality) then implies that $r \le V(x(\tau_*)) \le V(x_0)e^{-2\gamma\tau_*}$, concluding the proof. □

It is not difficult to see that (1) with $G(V) = -1$ always admits a feasible solution. Likewise, for any $\gamma < \sigma(A)$, (1) with $G(V) = -2\gamma V$ admits a feasible solution satisfying $r > 0$. In the numerical experiments (see Section 4), we have used the value $\gamma = \sigma(A)/2$. This choice comes from a trade-off between (i) flexibility in the shaping of the invariant regions (the constraint (1f) is softer when γ is smaller) and (ii) rate of decrease of V along the trajectories of the system inside $\mathcal{R}$ (equal to $e^{-2\gamma t}$). From Theorem 4, it follows that by choosing $\gamma = \sigma(A)/2$, we have an upper bound that is in the worst case within a factor 2 of the upper bound that we would get if using γ close to $\sigma(A)$, while allowing much more flexibility in the shaping of the invariant regions, so that in practice the upper bound is better that the ones with $\gamma \approx \sigma(A)$.

Finally, the objective (1a) of (1) guarantees that the worst-case upper bound on $t_*(x_0, \mathcal{R}; A)$, when x_0 varies in $\mathcal{R}$, obtained from the model is the smallest possible within the considered framework (see Theorem 4).

Example 5. Consider the matrix $A = \left[\begin{smallmatrix} -1 & 3 \\ 0 & -1 \end{smallmatrix}\right]$, and the region $\mathcal{R}$ depicted in Figure 1. The application of (1) with $G(V) = -2\gamma V$ for this A and this $\mathcal{R}$ is illustrated in Figure 1. We have used $\gamma = \sigma(A)/2 = 0.5$. We have also represented a sample trajectory of the system, for $t \in [0, \tau]$ where $\tau = \log(V(x_0)/r)/(2\gamma)$ is obtained from the optimal solution of (1); see Theorem 4. We verify that the trajectory does not exit the region $\mathcal{R}$ after time τ.

3.3 Upper bound on escaping time. Case II: $0 \notin \mathcal{R}$

Next, we consider the case where the origin is outside the region $\mathcal{R}$. Consider the following optimization program.

Decision variables: a convex quadratic function V, and scalars $\lambda_1, \dots, \lambda_M \ge 0$, $\mu_1, \dots, \mu_N \ge 0$, $\nu_1, \dots, \nu_N \ge 0$ and $r \in \mathbb{R}$; *Objective and constraints:*

$$\begin{aligned} \max \quad & r && \text{(2a)} \\ \text{s.t.} \quad & V \leqq 1 + \lambda_i F_i, \quad \forall i \in [M], && \text{(2b)} \\ & V \geqq r - \textstyle\sum_{i=1}^N \mu_i E_i, && \text{(2c)} \\ & \mathcal{L}_A V \leqq G(V) + \textstyle\sum_{i=1}^N \nu_i E_i, && \text{(2d)} \end{aligned}$$

where $G(V)$ is equal to either -1 (leading to quadratically increasing bounds), or to $-2\gamma V$ for some fixed parameter $\gamma > 0$ (leading to logarithmically increasing bounds).

Explanations. The goal is to find a quadratic function V whose Lie derivative is smaller than $G(V)$ inside $\mathcal{R}$. This is enforced by (2d) (implying that $\mathcal{L}_A V(x) \le G(V)$ whenever $E_i(x) \le 0$ for all $i \in [N]$ since $\nu_i \ge 0$). We also require that $\mathcal{R} \subseteq \mathrm{Sub}_1(V)$. This is enforced by (2b) (similar to Subsection 3.2). Finally, we try to find the largest r such that $\mathrm{Sub}_r(V)$ is outside $\mathcal{R}$. This is enforced by (2c) (implying that $V(x) \ge r$ whenever $E_i(x) \le 0$ for all $i \in [N]$ since $\mu_i \ge 0$, so that $\mathrm{Sub}_s(V) \cap \mathcal{R} = \emptyset$ for all $s < r$). See also Figure 2 and Example 7 for an illustration.

Any feasible solution of the above optimization program provides an upper bound on the escaping time from $\mathcal{R}$, in the exact same way as in Theorem 4:

Theorem 6. Assume $0 \notin \mathcal{R}$. Let $(r, V, \dots)$ be a feasible solution [3] of (2) with $G(V) = -1$. Then, for any $x_0 \in \mathcal{R}$,

$$t_*(x_0, \mathcal{R}; A) \le \max(V(x_0) - r, 0) \le \max(1 - r, 0).$$

Similarly, let $(r, V, \dots)$ be a feasible solution of (1) with $G(V) = -\gamma V$, such that $r > 0$. Then, for any $x_0 \in \mathcal{R}$,

$$t_*(x_0, \mathcal{R}; A) \le \frac{\log^+\left(\frac{V(x_0)}{r}\right)}{2\gamma} \le \frac{\log^+\left(\frac{1}{r}\right)}{2\gamma},$$

where $\log^+(\alpha) \triangleq \log(\max(\alpha, 1))$.

Proof. Similar to that for Theorem 4; omitted. □

As for the model (1) in the previous subsection, it is not difficult to see that (2) with $G(V) = -1$ is always feasible. The same holds with $G(V) = -2\gamma V$ and $r > 0$, whenever $\gamma < \sigma(A)$. In the numerical experiments (see Section 4), we have used the value $\gamma = \sigma(A)/2$, which is a trade-off between (i) flexibility in the shaping of the invariant regions and (ii) rate of decrease of V along trajectories of the system inside $\mathcal{R}$ (see also Subsection 3.2 for details).

In the same way, the objective (2a) of (2) guarantees that the worst-case upper bound on $t_*(x_0, \mathcal{R}; A)$, when x_0 varies in $\mathcal{R}$, obtained from the model is the smallest possible within the considered framework (see Theorem 6).

Example 7. Consider the matrix $A = \left[\begin{smallmatrix} -0.1 & 1 \\ -1 & -0.1 \end{smallmatrix}\right]$, and the region $\mathcal{R}$ depicted in Figure 2. The application of (2), with $G(V) = -2\gamma V$ for this A and this $\mathcal{R}$ is illustrated in Figure 2. We have used $\gamma = \sigma(A)/2 = 0.05$. We have also represented a sample trajectory of the system, for $t \in [0, \tau]$ where $\tau = \log(V(x_0)/r)/(2\gamma)$ is obtained from the optimal solution of (2); see Theorem 6. We verify that the trajectory exits the region $\mathcal{R}$ before time τ.

[3] Note that r can be negative.

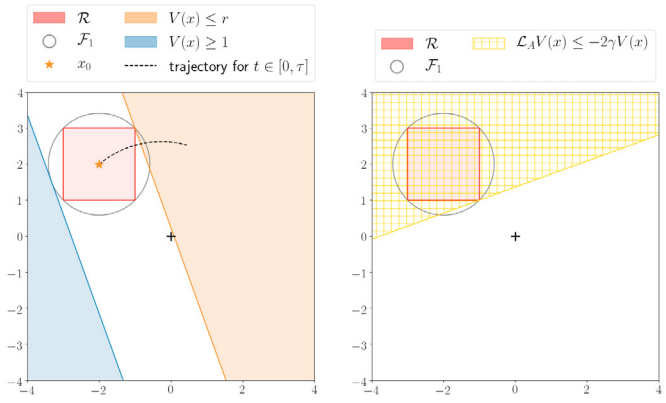

Fig. 2. Illustration of Example 7. We verify that $\mathcal{R} \subseteq \mathrm{Sub}_1(V)$, $\mathrm{Sub}_r(V) \cap \operatorname{int} \mathcal{R} = \emptyset$, and $\mathcal{L}_A V(x) \le G(V(x))$ for all $x \in \mathcal{R}$.

4. NUMERICAL EXPERIMENTS AND COMPARISONS

In this section, we would like to compare the performances of the two models of Section 3 (with $G(V) = -1$ and with $G(V) = -2\gamma V$). We also compare them with the approach of Rabi (2020), which relies on quadratic Lyapunov functions obtained by arbitrarily restricting the right-hand side of the Lyapunov equation, as in $A^\top P + PA = -I$. To make the comparison, we have applied the different approaches on randomly generated matrices of dimension $n = 10$, as explained below.

Random matrix generation. To generate random *stable* matrices, we proceed as follows: (1) Start from a random block-diagonal matrix D with 2-dimensional blocks on the diagonal. Each block is obtained from a pair of real or complex conjugated eigenvalues λ_1, λ_2 defined as the roots of $\lambda^2 + \sqrt{\alpha}\lambda + \beta/4$ where α and β are chosen uniformly at random over the interval $[0, 1]$.[4] (2) From D, build the random matrix A by applying the similarity transformation $A = UDU^{-1}$, where the elements of U are randomly generated from the standard normal distribution (which implies that U is invertible with probability 1).

Regions definition. We also have to define the region $\mathcal{R}$. We made the choice to consider a fixed region because possible transformations are included in the random similarity transformations UDU^{-1}. For the case I (origin inside), we let $\mathcal{R}$ be the set of points x such that $-2.5 \le x[1] \le 1.5$ and $-2 \le x[i] \le 2$ for $i = 2, \dots, 10$ (where $x[i]$ is the ith component of x). We also let $x_0 = [-1.5, -1, \dots, -1]^\top$. For the case II (origin outside), we let $\mathcal{R}$ be the set of points x such that $-6 \le x[i] \le -2$ for $i = 1, 2$, and $-2 \le x[i] \le 2$ for $i = 3, \dots, 10$. We let $x_0 = [-5, -5, -1, \dots, -1]^\top$.

We have used the SDP solver Mosek (https://www.mosek.com) with the package JuMP (Dunning et al., 2017) in Julia, to compute the solutions of the different models. All computations were done on a laptop with processor Intel Core i7-7600u and 16 GB RAM running Windows.

[4] This implies that λ_1, λ_2 have negative real parts with probability 1. Moreover, they are real with probability $\frac{1}{2}$ (useful to investigate the effect of both real and complex eigenvalues on the performance of the techniques).

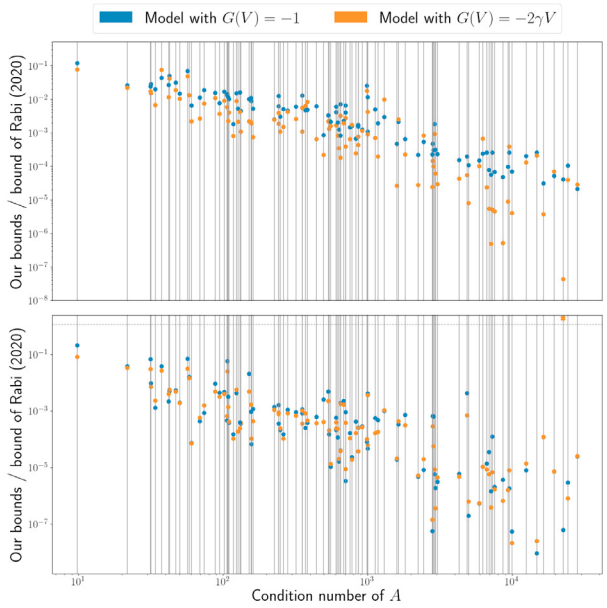

Fig. 3. *Top:* Case I (origin inside). *Bottom:* Case II (origin outside). The same matrices were used for both cases.

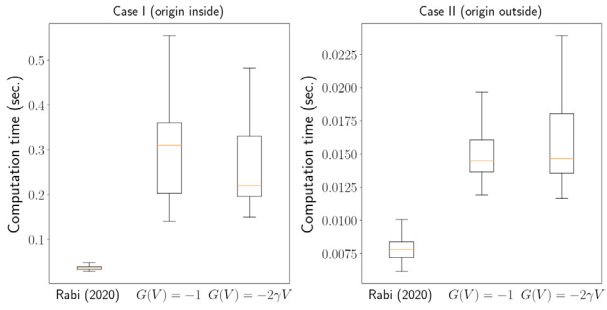

Fig. 4. Box plots of computation times of the different approaches.

4.1 Experimental results

For 100 randomly generated matrices A_i, $i = 1, \dots, 100$, we have used the approaches described in Subsections 3.2 and 3.3 and in Rabi (2020) to compute upper bounds on the escaping time $t_*(x_0, \mathcal{R}; A_i)$. The plots in Figure 3 show the pairs (κ_i, ρ_i), where κ_i is the condition number of A_i, and ρ_i is the ratio between the upper bound obtained from the models presented in this paper (with different colors for the different models) divided by the upper bound obtained using the approach of Rabi (2020).[5,6] The goal is to evaluate the improvement provided by the approaches in this paper, and how such improvements depend on the condition number of A. Finally, the computation times of the different approaches are given in Figure 4.

[5] Comparisons between the obtained upper bound and the actual escaping time $t_*(x_0, \mathcal{R}; A_i)$ for the different models presented in this paper are discussed in the extended version of this work (see Berger and Rabi, 2021).

[6] In one case, the solver failed to compute a feasible solution within the allotted time (using the solver's default options). This case is shown in the second chart of Figure 3; it is indicated by the cross near the chart's top-right corner.

4.2 Conclusions from the numerical experiments

In Figure 3, we observe that the models presented in this paper provide upper bounds on the escaping time that are between 10^1 and 10^8 times better than the upper bounds obtained with the approach described in Rabi (2020). We also notice that the higher the condition number of A, the better the improvement tends to be. Finally, Figure 3 also shows that the models with $G(V) = -2\gamma V$ provide slightly better upper bounds than the models with $G(V) = -1$ for the case I, and comparable upper bounds for the case II.

Regarding computation time, the approach of Rabi (2020) is in average up to 10 times faster than the methods in this paper. This is not really surprising since our models involves a number of variables growing as $n^2 + N + M$, while the approach of Rabi (2020) involves a number of variables proportional to N (number of ellipsoids $\mathcal{E}_i$).

5. CONCLUSIONS

In this paper, we improved an existing way of using quadratic Lyapunov theory to compute upper bounds on the set escaping time of stable linear systems. Our bounds are based on convex programming techniques. In particular, our approach uses Lyapunov-like functions that are not necessarily centered at the origin, and whose rates of decrease are guaranteed only in a specific subset of the state space. Numerical experiments demonstrate that in many cases our approach is highly favorable in terms of the accuracy of the upper bound (up to 10^8 times better), while requiring only a modest increase in computation time (up to 10 times slower).

For further work, we plan to investigate the usefulness of both approaches in practical applications, for instance as part of a stand-alone software for the simulation of switched linear systems (see the introduction). Finally, we also plan to investigate ways to speed up the computation of the Lyapunov-like functions in our models; for instance, by leveraging the fact that in general we do not need an accurate optimal solution of the optimization models (a near-optimal feasible solution is often sufficient), or that a feasible initial solution can always be obtained from the Lyapunov equation.

REFERENCES

Bell, P.C., Delvenne, J.C., Jungers, R.M., and Blondel, V.D. (2010). The continuous Skolem-Pisot problem. *Theoretical Computer Science*, 411(40–42), 3625–3634. doi:10.1016/j.tcs.2010.06.005.

Ben-Tal, A. and Nemirovski, A. (2001). *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, volume 2 of *MPS-SIAM Series on Optimization*. SIAM. doi:10.1137/1.9780898718829.

Berger, G.O. and Rabi, M. (2021). Bounds on set exit times of affine systems, using Linear Matrix Inequalities. arXiv preprint: 2104.12682.

Boyd, J.P. (2014). *Solving transcendental equations: the Chebyshev polynomial proxy and other numerical rootfinders, perturbation series, and oracles*. SIAM. doi:10.1137/1.9781611973525.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Carloni, L.P., Passerone, R., and Pinto, A. (2006). Languages and tools for hybrid systems design. *Foundations and Trends in Electronic Design Automation*, 1(1–2), 1–193. doi:10.1561/1000000001.

Copp, D.A. and Sanfelice, R.G. (2016). A zero-crossing detection algorithm for robust simulation of hybrid systems jumping on surfaces. *Simulation Modelling Practice and Theory*, 68, 1–17. doi:10.1016/j.simpat.2016.07.005.

Cremona, F., Lohstroh, M., Broman, D., Lee, E.A., Masin, M., and Tripakis, S. (2019). Hybrid co-simulation: it's about time. *Software & Systems Modeling*, 18, 1655–1679. doi:10.1007/s10270-017-0633-6.

Dunning, I., Huchette, J., and Lubin, M. (2017). JuMP: a modeling language for mathematical optimization. *SIAM Review*, 59(2), 295–320. doi:10.1137/15M1020575.

Esposito, J.M., Kumar, V., and Pappas, G.J. (2001). Accurate event detection for simulating hybrid systems. In *Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control*, 204–217. Springer-Verlag. doi:10.5555/646881.710634.

Farkas, R., Bergmann, G., and Horváth, Á. (2019). Adaptive step size control for hybrid CT simulation without rollback. In *Proceedings of the 13th International Modelica Conference, Regensburg, Germany, March 4–6, 2019*, 157. Linköping University Electronic Press. doi:10.3384/ecp19157503.

Goebel, R., Sanfelice, R.G., and Teel, A.R. (2009). Hybrid dynamical systems. *IEEE Control Systems Magazine*, 29(2), 28–93. doi:10.1109/MCS.2008.931718.

Heemels, W.M.H., De Schutter, B., and Bemporad, A. (2001). Equivalence of hybrid dynamical models. *Automatica*, 37(7), 1085–1091. doi:10.1016/S0005-1098(01)00059-0.

Legat, B., Gomes, C., Karalis, P., Jungers, R.M., Navarro-López, E.M., and Vangheluwe, H. (2020). Stability of planar switched systems under delayed event detection. *arXiv:2009.04505*.

Rabi, M. (2020). Piece-wise analytic trajectory computation for polytopic switching between stable affine systems. In *Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control*, 1–11. ACM. doi:10.1145/3365365.3382204.

Rocca, A., Dang, T., and Fanchon, E. (2015). Exploiting the eigenstructure of linear systems to speed up reachability computations. In O. Maler, Á. Halász, T. Dang, and C. Piazza (eds.), *International Workshop on Hybrid Systems Biology*, volume 7699 of *Lecture Notes in Computer Science*, 111–127. Springer International Publishing. doi:10.1007/978-3-319-27656-4_7.

Schweiger, G., Gomes, C., Engel, G., Hafner, I., Schoeggl, J., Posch, A., and Nouidui, T. (2019). An empirical survey on co-simulation: promising standards, challenges and research needs. *Simulation modelling practice and theory*, 95, 148–163. doi:10.1016/j.simpat.2019.05.001.

Wang, H., Chen, L., and Hu, Y. (2015). A state event detecting algorithm for hybrid dynamic systems. *Simulation*, 91(11), 959–969. doi:10.1177/0037549715606968.