A general framework for verification and control of dynamical models via certificate synthesis

Edwards, Alec; Peruffo, Andrea; Abate, Alessandro

DOI
10.1016/j.arcontrol.2025.101028

Publication date
2025

Document Version
Final published version

Published in
Annual Reviews in Control

Annual Reviews in Control 60 (2025) 101028

Contents lists available at ScienceDirect

Annual Reviews in Control

journal homepage: www.elsevier.com/locate/arcontrol

Review Article

A general framework for verification and control of dynamical models via certificate synthesis

Alec Edwards [a],*, Andrea Peruffo [b], Alessandro Abate [a]

[a] Department Of Computer Science, Oxford, UK
[b] Center for Systems and Control, Delft, The Netherlands

ARTICLE INFO

Keywords:
Continuous-time control systems
Formal verification
Machine learning
Stability
Safety
Reachability

ABSTRACT

An emerging branch of control theory specialises in *certificate learning*, concerning the specification of a desired (possibly complex) system behaviour for an autonomous or control model, which is then analytically verified by means of a function-based proof. However, the synthesis of controllers abiding by these complex requirements is in general a non-trivial task and may elude the most expert control engineers. This results in a need for automatic techniques that are able to design controllers and to analyse a wide range of elaborate specifications. In this paper, we provide a general framework to encode system specifications and define corresponding certificates, and we present an automated approach to formally synthesise controllers and certificates. Our approach contributes to the broad field of safe learning for control, exploiting the flexibility of neural networks to provide candidate control and certificate functions, whilst using SAT-modulo-theory (SMT)-solvers to offer a formal guarantee of correctness. We test our framework by developing a prototype software tool, and assess its efficacy at verification via control and certificate synthesis over a large and varied suite of benchmarks.

Contents

1. Introduction 2
2. Preliminaries 3
   2.1. Dynamical models 3
   2.2. Systems and properties 3
   2.3. Neural networks 3
   2.4. SAT-Modulo Theory (SMT) 4
3. Properties and certificates 4
   3.1. Stability 4
   3.2. Region of attraction 4
   3.3. Safety 5
   3.4. Stable while avoid 5
   3.5. Reach while avoid 5
   3.6. Reach-and-stay while avoid 6
   3.7. Reach, avoid and remain 6
   3.8. Summary and classification of properties 6
   3.9. Certificates for control models 7
4. Synthesis of certificates and controllers 7
   4.1. Learner 7
      4.1.1. Certificate loss 8
      4.1.2. Controller loss 8
   4.2. Verifier 8
   4.3. Enhanced communication amongst components 8
   4.4. Comments on specific certificates 9
      4.4.1. Lyapunov functions 9

* Corresponding author.
E-mail addresses: research@aleccedwards.slmail.me (A. Edwards), alessandro.abate@cs.ox.ac.uk (A. Abate).

https://doi.org/10.1016/j.arcontrol.2025.101028
Received 28 June 2024; Received in revised form 24 September 2025; Accepted 25 September 2025
Available online 4 October 2025

4.4.2. ROA ... 9
4.4.3. RSWA ... 9
5. Computational experiments and benchmarks ... 9
5.1. Main results ... 9
5.2. Control loss evaluation ... 10
5.3. Comparison to Fossil 1.0 baseline ... 10
6. Discussions on generality ... 11
6.1. Asymptotic reachability ... 11
6.2. Sufficiency of the certificates and completeness of their synthesis ... 12
6.3. Modularity of the synthesis and nested properties ... 12
6.4. Issues of scale ... 12
6.5. Broader connections and taxonomy of properties ... 12
7. Concluding remarks ... 12
Declaration of competing interest ... 13
Acknowledgements ... 13
Appendix A. Proof of theorems ... 13
Appendix B. Broader connections and taxonomy of properties ... 13
Data availability ... 14
References ... 14

1. Introduction

The analysis of the behaviour of continuous-time dynamical systems focuses on a wide range of properties, which are themselves suitable for an even wider range of applications. Properties of interest include arguably the most common property, (asymptotic) stability, namely the convergence of trajectories to an equilibrium (Sastry, 1999); to safety, namely the avoidance of an unsafe region of the state space at all time (Blanchini & Miani, 2008); to its dual, reachability, that is the hitting of a target region of the state space in finite time (Henzinger, 1996). As we shall see, with the combination and the slight modification of these basic properties (we shall alternatively denote them as specifications or requirements), an engineer may design a broad range of desired dynamical behaviours for any model at hand.

Given a model of a dynamical system, such spectrum of properties can be investigated from different perspectives and with diverse approaches: either analytical (e.g., via local linearisation and eigenvalues computation) (Sastry, 1999), or computational ones (e.g., via dynamic flow propagation or via reach-set computation). In general, non-linearity in dynamical models is difficult to deal with: on account of this, approaches that are *indirect* or *sufficient* can be successful: a proof that the system actually fulfils a given requirement can be offered in the form of a *certificate*: the onus is to find, or to synthesise, a real-valued function defined over the state space with proper characteristics. A celebrated instance of indirect methods is the synthesis of Lyapunov functions (Lyapunov, 1992), whereby one ought to hand-craft a bespoke energy function, oftentimes based on intuition and on physical properties of the underlying dynamical model.

In recent years numerical optimisation methods have automated the synthesis of certificates, employing *templates*, i.e. candidate functions where only the coefficients (parameters) ought to be determined. Commonly, the choice falls onto polynomial templates framed as sum-of-squares convex problems (Goubault, Jourdan, Putot, & Sankaranarayanan, 2014; Papachristodoulou et al., 2013; Papachristodoulou & Prajna, 2002; Prajna, 2006), which admit globally optimal solutions. However, these techniques operate solely on models with polynomial dynamics and various convexity assumptions. Alternative formulations include linear programs (Ben Sassi, Sankaranarayanan, Chen, & Ábrahám, 2016; Ratschan & She, 2010; Sankaranarayanan, Chen, & Ábrahám, 2013) and semi-algebraic systems (She, Li, Xue, Zheng, & Xia, 2013; She, Xia, Xiao, & Zheng, 2009), all of which raise structural requirements on the dynamical models at hand.

Despite the usefulness of the mentioned synthesis approaches, they are numerically sensitive and generally unsound, and thus undesirable for robust solutions and safety-critical applications (Abate, 2017; Bohrer, Tan, Mitsch, Myreen, & Platzer, 2018; Knight, 2002). Consequently, in recent years interest has grown in approaches for synthesis that can yield *provably-correct* certificates, much in the same line of research as *correct-by-design* control synthesis (Belta, Yordanov, & Gol, 2017; Tabuada, 2009). A powerful technique to reason formally about correctness involves SMT-solving (Barrett, Stump, Tinelli, et al., 2010). SAT-modulo-theory (SMT) extends satisfiability (SAT) solving to richer theories, enabling, for example, finding feasible assignments of real numbered variables over nonlinear formulae.

SMT can be in particular leveraged for synthesis tasks. *Inductive* approaches (Solar-Lezama, Tancau, Bodik, Seshia, & Saraswat, 2006), leveraging SMT, have been used to synthesise certificates (Ravanbakhsh & Sankaranarayanan, 2015b, 2019), controllers (Abate et al., 2020; Huang, Wang, Mitra, Dullerud, & Chaudhuri, 2015) and abstractions (Abate, Edwards, & Giacobbe, 2022) for dynamical models. Such techniques have been used first for stability certification of dynamical models using polynomial Lyapunov functions and later extended to more general reach-avoid requirements (Ahmed, Peruffo, & Abate, 2020; Kapinski, Deshmukh, Sankaranarayanan, & Arechiga, 2014; Ravanbakhsh & Sankaranarayanan, 2015a, 2015b). Related to SMT-based solutions, approaches that formulate synthesis problems as a mixed-integer linear programs have also been used to synthesise provably-correct Lyapunov functions (Dai, Landry, Pavone, & Tedrake, 2020; Dai, Landry, Yang, Pavone, & Tedrake, 2021) for stability analysis, encompassing linear matrix inequalities for uncertain systems (Masti, Fabiani, Gnecco, & Bemporad, 2023), and barrier certificates for safety (Chen, Fazlyab, Morari, Pappas, & Preciado, 2020, 2021; Zhao, Zeng, Chen, Liu & Woodcock, 2021). Notably, mixed-integer problems also encompass candidates in the form of neural networks with ReLU activation functions, and may employ an optimisation engine like Gurobi (Gurobi Optimization, LLC, 2021) to certify the soundness of the proposed functions (Zhao et al., 2021).

Related work The flexibility of neural networks has permeated the field of certificate synthesis, including their use for synthesis of Lyapunov-like functions. For instance, Jin, Wang, Yang, and Mou (2020), Noroozi, Karimaghaee, Safaei, and Javadi (2008) and Richards, Berkenkamp, and Krause (2018) describe generally unsound procedures for gradient descent-based training of a Lyapunov neural network. Sound, counter-example based techniques are proposed in e.g. Abate, Ahmed, Giacobbe and Peruffo (2020), Ahmed et al. (2020), Chang, Roohi, and Gao (2019), Grande, Anderlini, Peruffo and Salavasidis (2023), Grande et al. (2023) and Samanipour and Poonawala (2023), specifically for Lyapunov functions, and solely for barrier certificates in e.g. Peruffo, Ahmed, and Abate (2021), Ratschan (2017) and Zhao, Zeng, Chen, and Liu (2020). The choice of SMT solver depends on the models under consideration and desired certificate template: Z3 (de

Moura & Bjørner, 2008) handles polynomial functions, dReal (Gao, Kong, & Clarke, 2013), iSat3 (isat3, 0000) and CVC5 (Barbosa et al., 2022) enable analysis of non-polynomial functions as well as polynomials. The synthesis of certificates includes more complex properties, as proposed in Verdier (2020) and Verdier and Mazo (2020) where bespoke genetic algorithms are leveraged to generate *reach-while-stay* and *reach-and-stay-while-stay* functions, alongside controllers, for hybrid systems. Certificates for reach-avoid properties have been previously synthesised using counterexample-based approaches (Ravanbakhsh & Sankaranarayanan, 2015b). Meanwhile reach-avoid-stay properties and corresponding certificates have been well studied from a theoretical perspective (Meng, Li, Fitzsimmons & Liu, 2021; Meng, Li & Liu, 2021). In this work, we collate certificates for these more complex properties, and categorise them in order to unify them within simpler certificates for stability, reachability and safety. We also generalise the concept of reach-avoid-stay property by separating allowing the *stay* set to be different from the *reach* set. The interested reader may find a survey on neural certificates with application in control synthesis and robotics in Dawson, Gao, and Fan (2022).

Contributions We summarise our contributions as follows:

- We collate and add to existing certificates across literature for nonlinear continuous-time dynamical models.
- We categorise the properties these certify into a simplified and general framework, which we newly describe through notions *arrive, avoid* and *remain*.
- We describe a unified algorithm to concurrently synthesise *both* controllers and certificates in parallel for dynamical models, to prove they satisfy these properties.
- We implement our framework[1] on top of the computational library Fossil (Abate, Ahmed, Edwards, Giacobbe, & Peruffo, 2021; Edwards, Peruffo, & Abate, 2024), offering a new prototype software tool that can verify general *arrive, avoid* and *remain* properties for nonlinear control models using controllers and certificates based on neural networks.

Organisation This manuscript is organised as follows. Section 2 provides relevant background information and notation used across this work. In Section 3, we describe a range of properties for dynamical models and certificates which prove that they hold. Next, we describe a unified and computationally correct algorithm for synthesising these certificates in Section 4. We present experimental results for a prototype tool to synthesise these certificates in Section 5, before discussing the limitations of our framework in Section 6 and providing concluding remarks in Section 7. Finally, we outline the proofs of all theorems in Appendix A, and we reserve a discussion on the broader taxonomic connections of our work to Appendix B.

2. Preliminaries

2.1. Dynamical models

We denote the set of positive real numbers and its extended version as $\mathbb{R}_+$ and $\overline{\mathbb{R}} = \mathbb{R}_+ \cup \{+\infty\}$, respectively. A function is said to be of class C^1 if its first derivative exists and is continuous. Let us consider models described by

$$\dot{\xi}(t) = f_u(\xi(t), u(t)), \quad x(t_0) = x_0 \in \mathcal{X}_I \subseteq \mathcal{X} \tag{1}$$

where $x = \xi(t) \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state of the system, $f_u : \mathcal{X} \times \mathcal{U} \to \mathbb{R}^n$ is a Lipschitz-continuous vector field describing the model dynamics. We refer to these models as control models. We denote a trajectory over a time horizon $T \in \overline{\mathbb{R}}$ as $\xi(t) : [t_0, T] \to \mathbb{R}^n$, where $\xi(t)$ admits a time derivative everywhere, and such that $\dot{\xi}(t) = f_u(\xi(t), u(t))$ and $\xi(t_0) \in \mathcal{X}$, namely the trajectory is a solution of the model in (1). Finally, $u(t) \in \mathcal{U} \subseteq \mathbb{R}^m$ is the input and $\mathcal{X}_I$ denotes the set of initial conditions. Once a state-feedback controller $u(t) = k(\xi(t))$ has been specified, we may interpret the dynamics described by (1) as those of a closed-loop model, as follows

$$\dot{\xi}(t) = f(\xi(t)), \quad \xi(t_0) = \xi_0 \in \mathcal{X}_I \subseteq \mathcal{X}. \tag{2}$$

We refer to models of this kind simply as autonomous dynamical models. We denote with x^* an equilibrium point of (2), namely where $f(x^*) = 0$. We note that we can assume without loss of generality that this equilibrium point is the origin, since we are always able to translate the dynamics to ensure this Khalil (2002).

2.2. Systems and properties

The goal of this work is to find a feedback controller, namely a signal $u(t)$ in time, such that the dynamics above satisfy some desired temporal requirements (e.g., safety), or to show that given closed-loop (autonomous) dynamics are endowed with some given property (e.g., asymptotic stability).

We interpret properties of dynamical models in (2) in terms of their trajectories, and of binary relations that these trajectories have with given sets within the state space $\mathcal{X}$. To this end, we now introduce the notations and the semantics of the sets characterising properties of dynamical models. We denote as $\mathcal{X}_U$ an unsafe set, indicating a region of the state space where the system's trajectories should avoid; $\mathcal{X}_G$ represents a goal set, indicating the region that the system's trajectories should enter; and $\mathcal{X}_F$ represents a final set, indicating a set where the system's trajectories should remain for all times after arriving at the goal set. These sets will be employed in the next section for the definition of properties, as depicted in Fig. 1. Implicitly, we consider that the unsafe and final sets are disjoint, i.e. $\mathcal{X}_U \cap \mathcal{X}_F = \emptyset$ and that the goal set is contained within the final set, i.e. $\mathcal{X}_G \subset \mathcal{X}_F$. Often we will assume these sets to be compact; the motivation for assuming sets to be compact (rather than just closed) is mainly to ease automated synthesis and verification. Additional topological properties of sets will be clarified later, within formal statements. Given a set S in a domain $\mathcal{X}$, we denote by $S^\complement$ its complement, i.e. $\mathcal{X} \setminus S$, and by $\mathrm{int}(S)$ its interior, namely the set without its border, i.e. $\mathrm{int}(S) = S \setminus \partial S$.

We consider a set S to be (forward) *invariant* if at some initial time t_0, $\xi(t_0) \in S$ implies that for all $t > t_0$, $\xi(t) \in S$ (Blanchini & Miani, 2008). We consider a set S_A to be attracting (Blanchini & Miani, 2008) with *region of attraction* S_B if for some initial time t_0 and any initial state $\xi(t_0) \in S_B$, the trajectory $\xi(t)$ converges to S_A as $t \to \infty$, i.e. if $\lim_{t\to\infty} dist(\xi(t), S_A) = 0$. Finally, given a function $C : \mathbb{R}^n \to \mathbb{R}$, we denote the Lie derivative of this function with respect to the vector field f simply as $\dot{C} = \langle \nabla C, f(x) \rangle$.

2.3. Neural networks

Denote a neural network $\mathcal{N}$ with input layer $z_0 \in \mathbb{R}^n$, corresponding to the dimension of the dynamical model in (2). This is followed by k hidden layers $z_1, \ldots, z_k$ with dimensions $h_1, \ldots, h_k$ respectively, and finally followed by an output layer $z_{k+1} \in \mathbb{R}^d$. In this work, $d \in \{1, m\}$, where $d = 1$ corresponds to a scalar-valued certificate, whereas $d = m$ is used for a state-feedback controller with m control variables.

We denote the hidden layers and output layer as h_i with index $i = 0, \ldots, k+1$, where $k+1$ denotes the output layer; to these layers are associated matrices of weights $W_i \in \mathbb{R}^{h_i \times h_{i-1}}$ and a vector of biases $b_i \in \mathbb{R}^{h_i}$ (MacKay, 2003). Every ith hidden layer is associated with an activation function $\sigma_i : \mathbb{R} \to \mathbb{R}$. The valuation of output and hidden layers is given by

$$z_i = \sigma_i(W_i \cdot z_{i-1} + b_i), \; i = 1, \ldots, k, \tag{3}$$

$$z_{k+1} = W_{k+1} \cdot z_k + b_{k+1}, \tag{4}$$

where each σ_i is applied element-wise to its h_i-dimensional argument.

[1] Avaliable at https://github.com/oxford-oxcav/fossil.

2.4. SAT-Modulo Theory (SMT)

Here, we offer a brief introduction to SMT solving to aid understanding of how they are used in this work; for a more comprehensive introduction the reader is directed to Barrett and Tinelli (2018).

Let $\phi(x)$ be a formula in first-order logic, expressed in terms of variables x in a given domain. SMT is the problem of determining if there exists an assignment of the variables x such that $\phi(x)$ is satisfied (i.e., if it evaluates to true). SMT extends the Boolean satisfiability (SAT) problem: SAT solvers use modern forms of the DPLL algorithm (Davis, Logemann, & Loveland, 1962; Davis & Putnam, 1960) to search for satisfying assignments for boolean formulae. As an example, consider the following boolean formula, $\phi_1(x) = (x_0 \vee x_1) \wedge (\neg x_0 \vee \neg x_2), x = (x_0, x_1, x_2), x_i \in \{T, F\}$. When provided with this formula, a SAT solver would return an assignment such as $x_0 = \mathrm{T}, x_1 = \mathrm{T}, x_2 = \mathrm{F}$, which satisfies $\phi_1(x)$.

Now let us consider an example that is more relevant to this work. Let σ_t be the hyperbolic tangent function, and consider the following function $g : \mathbb{R} \to \mathbb{R}$

$$g(x) = \sigma_t(2 - 2x) - 3\sigma_t(1 - x) + \frac{3}{2}.$$

Suppose we wish to find if g is negative at some point where x is non-negative. Formally, this is

$$\exists x \in \mathbb{R}, x \geq 0 : g(x) < 0.$$

An SMT solver over the theory of real algebra, which can in particular handle transcendental functions, can handle this problem. This solver will return *unsat* to this problem, as no satisfying assignment exists — it is unsatisfiable. Crucially, SMT-solvers are *sound*: if a satisfying assignment exists then it cannot return that the problem is unsatisfiable.

We note that $g(x)$ is in fact a trivial feed-forward neural network and we have proven that it is positive in a region of the state-space. Later, we will use the same ideas to prove similar properties for non-trivial networks.

3. Properties and certificates

We present a number of properties (or requirements) for dynamical models defined over their trajectories, alongside definitions of corresponding certificates, whose existence serve as *sufficient conditions* for the satisfaction of the desired properties. We focus on continuous-time models; while the presented properties may be seamlessly applied to discrete time models, the corresponding certificates would not necessarily align with the presentation of the work, hence we omit their discussion as outside the scope of this work. The proofs of the theorems relating certificates to dynamical properties are outlined in Appendix A. Several properties (and corresponding certificates) are ubiquitous across the control theory literature, whilst others have been more recently introduced or inherited from analogues in formal verification. Further, we include a practical variant on Lyapunov functions to allow for constrained local stability and an original certificate called Reach-Avoid-Remain, and suggest that more can be obtained in a modular, composable fashion. We emphasise that while many of these properties are similar to each other, they are subtly, and importantly, different and distinguished. Later in the section we provide a summary and clarification on their distinguishing features, which may also be seen in Fig. 1. Abbreviations for these properties (and other acronyms used in this work) may be found in Table 5 at the end of the work.

3.1. Stability

Stability is the most-studied property of dynamical models, and many definitions of this property exist. Stability is most commonly characterised in a Lyapunov (asymptotic) sense (Sastry, 1999), namely in terms of the distance of a trajectory from the equilibrium point. However, we choose to characterise stability in terms of set containment in order to achieve a consistent characterisation with subsequent properties and certificates. The corresponding certificate defined in the sequel may also be used to prove solely the more common asymptotic stability, which we leverage in the experimental results.

$$\exists \mathcal{X}_I : \forall \xi(t_0) \in \mathcal{X}_I, \exists T \in \overline{\mathbb{R}}, \forall \tau \geq T,\ \xi(\tau) \in \{x^*\}, \tag{5}$$

where $\mathcal{X}_I$ has non empty interior. In words, there exists some initial set $\mathcal{X}_I$ such that for all trajectories initialised in $\mathcal{X}_I$, there exists a time instant T (possibly at infinity) when the trajectory reaches the equilibrium state x^* and remains there for all times after T. A model can be proven to satisfy this property using a Lyapunov function, which we introduce next.

Certificate 1 (*Lyapunov Function*). *Given a model f with unique equilibrium point $x^* \in \mathrm{int}(\mathcal{X})$, consider a function $V : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}, V \in C^1$. V is a Lyapunov function if:*

$$V(x^*) = 0, \tag{6a}$$

$$V(x) > 0 \quad \forall x \in \mathcal{X} \setminus \{x^*\}, \tag{6b}$$

$$\dot{V}(x) = \langle \nabla V(x), f(x) \rangle < 0 \quad \forall x \in \mathcal{X} \setminus \{x^*\}. \tag{6c}$$

Theorem 1 (*Stability*). *Given a model (2), if a Lyapunov function exists, then (5) holds for some set of initial conditions $\mathcal{X}_I$.* ■

Let us clarify the issue of finding $\mathcal{X}_I$ in the next section.

3.2. Region of attraction

Theorem 1 proves the existence of some region of the state space in which initialised trajectories will converge asymptotically towards the origin — this region is known as a *region of attraction* (ROA). In general, Lyapunov functions merely prove the existence of a region of attraction within the state space, without additional information about its size or shape. Lyapunov functions may prove global asymptotic stability under additional conditions, but these can be difficult to synthesise and verify automatically for some models. Here, we offer a certificate the acts as a middle ground between local and global asymptotic stability.

Proving that all trajectories initialised within a given $\mathcal{X}_I$ are stable amounts to proving that $\mathcal{X}_I$ is contained wholly within a sub-level set of a Lyapunov function (which must also lie in $\mathcal{X}$), and that the Lyapunov conditions in (6) hold over this entire sub-level set. This is treated in the following equation and corollary.

First, we modify (5) to now require a specified set of initial states $\mathcal{X}_I$, which should be a region of attraction for an equilibrium point, as follows:

$$\forall \xi(t_0) \in \mathcal{X}_I, \exists T \in \overline{\mathbb{R}}, \forall \tau \geq T,\ \xi(\tau) \in \{x^*\}. \tag{7}$$

We can certify that an autonomous model satisfies this property using the following certificate.

Certificate 2 (*ROA Certificate*). *Let a dynamical model f be given with unique equilibrium point $x^* \in \mathcal{X}$. A Lyapunov function V is an ROA certificate if there exists a β such that $\mathcal{X}_I \subset \{x \in \mathcal{X} : V(x) \leq \beta\}$ and that the conditions in (6) hold over the set $\{x \in \mathcal{X} : V(x) \leq \beta\}$.* ■

Alternatively, a Lyapunov function can be interpreted as a proof that a region of attraction (here $\mathcal{X}_I$) exists within some larger set (here $\mathcal{X}$), whereas a ROA certificate proves the converse: that a given initial set $\mathcal{X}_I$ lies within a larger region of attraction, which is defined by a sublevel set of V. This certificate offers a more practical guarantee over classical Lyapunov functions: all trajectories initialised within the pre-defined set $\mathcal{X}_I$ indeed converge to x^*.

Corollary 2 (*Region of Attraction*). *Given a model (2), a bounded set of initial conditions $\mathcal{X}_I$, and a ROA certificate, then (7) holds.* ■

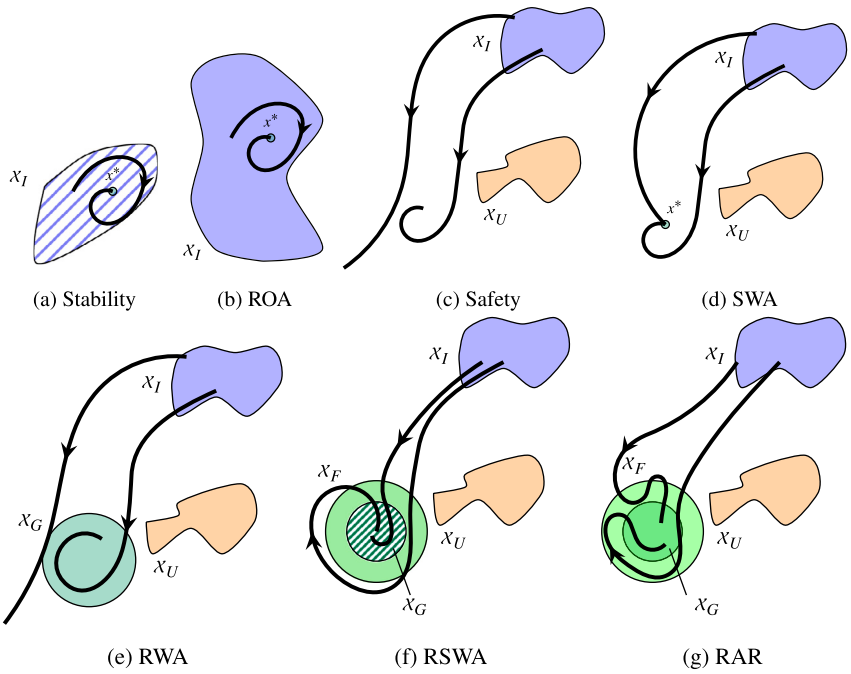


Fig. 1. Pictorial depiction of relevant properties in this work. Here, $\mathcal{X}_I$ is the initial set, $\mathcal{X}_U$ the unsafe set ($\mathcal{X}_S$ is its safe complement), $\mathcal{X}_G$ the goal/target set, $\mathcal{X}_F$ the final set. (The entire state space is $\mathcal{X}$.) Here, a dashed background denotes that the corresponding set's existence is implied by the corresponding certificate, but that it is not explicitly defined in the property. Notably, this means that the set cannot be *specified* a-priori when defining the property: e.g., the invariant set $\mathcal{X}_G$ may be any size contained within $\mathcal{X}_F$. This motivates the construction of the additional certificates (and corresponding properties) ROA and RAR, which instead allow for a-priori set specifications.

3.3. Safety

Safety is another fundamental property we can require from dynamical models. It involves the *avoidance* of some unsafe region: namely, that no trajectory starting from $\mathcal{X}_I$ may enter the unsafe set $\mathcal{X}_U$[2]; formally

$$\forall \xi(t_0) \in \mathcal{X}_I, \forall t \in \overline{\mathbb{R}}, t \geq t_0, \xi(t) \in {\mathcal{X}_U}^{\complement}. \tag{8}$$

For continuous-time models, safety over an unbounded time horizon can be proved via barrier certificates (Prajna, 2006; Prajna, Jadbabaie, & Pappas, 2004).

Certificate 3 (*Barrier Certificate*). *Consider a dynamical model f, a compact unsafe set $\mathcal{X}_U$ and compact initial set $\mathcal{X}_I$. A function $B : \mathcal{X} \subset \mathbb{R}^n \to \mathbb{R}, B \in C^1$, is a Barrier certificate if the following holds:*

$$B(x) \leq 0 \ \forall x \in \mathcal{X}_I, \tag{9a}$$

$$B(x) > 0 \ \forall x \in \mathcal{X}_U, \tag{9b}$$

$$\dot{B}(x) = \langle \nabla B, f(x) \rangle < 0 \ \forall x \in \{x : B(x) = 0\}. \tag{9c}$$

Many characterisations of barrier certificates exist, to cover different applications or to abide by additional constraints. These include reciprocal (Ames, Xu, Grizzle, & Tabuada, 2017), high-order (zeroing) (Tan, Cortez, & Dimarogonas, 2022), or barrier conditions with modifications on the Lie derivative (9c) (Prajna et al., 2004). These certificates are alike in that they certify safety properties. However, the formulation used in this work are known to exist for any system which is safe (Ratschan, 2018), and hence we consider this sufficient for this work.

[2] We shall later draw connections between the concept of safety and the dual notion of (unconstrained) *reachability*. Please refer to the discussions in Section 3.8 and in Appendix B.

Theorem 3 (*Safety*). *Given a model (2), a compact domain $\mathcal{X}$, a compact initial set $\mathcal{X}_I \subset \mathcal{X}$ and an unsafe set $\mathcal{X}_U$, alongside a barrier certificate, then (8) holds.* ■

3.4. Stable while avoid

It is natural to extend the aforementioned notions of stability and safety towards a combination of both, whereby all relevant trajectories converge towards an equilibrium point, while also avoiding a given unsafe set. Such a property is formally described as follows:

$$\forall \xi(t_0) \in \mathcal{X}_I, \exists T \in \overline{\mathbb{R}}, \forall t \in [t_0, T), \xi(t) \in {\mathcal{X}_U}^{\complement} \\ \wedge \forall \tau \geq T, \xi(\tau) \in \{x^*\}. \tag{10}$$

Since (10) is simply the conjunction of a stability property and a safety property, certifying this is equivalent to concurrently certifying both stability and safety hold: this task can be thus formally tackled by the following corollary.

Corollary 4 (*Stable While Avoid (SWA)*). *Given a model (2) with unique equilibrium point $x^* \in \mathcal{X}$, a compact domain $\mathcal{X}$, a compact initial set $\mathcal{X}_I \subset \mathcal{X}$ and an unsafe set $\mathcal{X}_U$ alongside a ROA certificate V and barrier certificate B, then (10) holds.* ■

Notice that it is alternatively possible to combine the conditions of (6) and (9) into that of a *single* certificate for stability and safety. Such a function is sometimes referred to as a Lyapunov-Barrier certificate (Romdlony & Jayawardhana, 2016; Wu et al., 2019). However, in this work we choose to use two separate functions, as this makes synthesis easier and more modular.

3.5. Reach while avoid

Let us now set asymptotic stability aside, and instead study properties over a finite time horizon: namely, we require that trajectories enter a non-singleton set in finite time. These are reachability-like

properties, which require trajectories to reach a region known as a goal set. A reach-while-avoid (RWA) property requires a non-singleton goal set to be reached within a finite time horizon T, while avoiding an unsafe region; in formal terms,

$$\forall \xi(t_0) \in \mathcal{X}_I, \exists T \in \mathbb{R}, \forall t \in [t_0, T] : \\ \xi(t) \in {\mathcal{X}_U}^{\complement} \wedge \xi(T) \in \mathcal{X}_G. \tag{11}$$

Next, we introduce an RWA certificate to guarantee that this property holds for a model under consideration.

Certificate 4 (*RWA*). *Define an unsafe set $\mathcal{X}_U = \mathcal{X} \setminus \mathcal{X}_S$, where $\mathcal{X}_S$ is a compact safe set, a compact initial set $\mathcal{X}_I \subset \text{int}(\mathcal{X}_S)$, and a compact goal set $\mathcal{X}_G \subset \text{int}(\mathcal{X}_S)$ with non-empty interior. A reach-while-avoid (RWA) certificate (Verdier, 2020) is a function $V : \mathbb{R}^n \to \mathbb{R}, V \in C^1$ if there exists $\gamma \in \mathbb{R}_+$, such that*

$$V(x) \leq 0 \quad \forall x \in \mathcal{X}_I, \tag{12a}$$

$$V(x) > 0 \quad \forall x \in \partial\mathcal{X}_S, \tag{12b}$$

$$\dot{V}(x) \leq -\gamma \quad \forall x \in \{x \in \mathcal{X}_S : V(x) \leq 0\} \setminus \mathcal{X}_G. \tag{12c}$$

Theorem 5 (*Reach-While-Avoid*). *Given a model (2) and a RWA Certificate corresponding to the given sets of interest, then (11) holds.* ■

Remark 6 (*Unconstrained Reachability*). Unconstrained reachability can be defined as a special case of RWA, where we set $\mathcal{X}_U = \emptyset$ (i.e. $\mathcal{X}_S = \mathcal{X}$). Hence, a certificate can be provided accordingly, as special instance of Certificate Certificate 4. ■

We emphasise that a RWA certificate does not prove that trajectories will *remain* within the goal set, or that trajectories shall avoid the unsafe set for all (unbounded) time, nor does the specification in (11) indeed encode these requirements. In fact, since the Lie derivative condition (12c) does not hold across the goal set $\mathcal{X}_G$, it is possible for trajectories to leave the goal set after entering it, and thereafter possibly enter the unsafe set $\mathcal{X}_U$. This alternative, more restrictive scenario is addressed by the next certificate.

3.6. Reach-and-stay while avoid

A reach-and-stay while avoid (RSWA) property is similar to the RWA property, consisting of RWA with an additional requirement that trajectories will eventually remain within some final set indefinitely. Formally, trajectories should satisfy the property

$$\exists \mathcal{X}_G \subset \text{int}(\mathcal{X}_F) : \forall \xi(t_0) \in \mathcal{X}_I, \exists T \in \mathbb{R}, \forall t \in [t_0, T], \\ \xi(t) \in {\mathcal{X}_U}^{\complement} \wedge \xi(T) \in \mathcal{X}_G \\ \wedge \forall \tau \geq T : \xi(\tau) \in \mathcal{X}_F. \tag{13}$$

Notably, this property does not require that trajectories reach the final set and remain within it, and may enter and leave the final set as long as they eventually remain within the final set. However, in finite time trajectories must reach some subset of the final set a goal set, after which point they must remain within the final set for all time.

Certificate 5 (*RSWA*). *Define an unsafe set $\mathcal{X}_U = \mathcal{X} \setminus \mathcal{X}_S$, where $\mathcal{X}_S$ is a compact safe set, then define a compact initial set $\mathcal{X}_I \subset \text{int}(\mathcal{X}_S)$, and a compact final set $\mathcal{X}_F \subset \text{int}(\mathcal{X}_S)$. A RSWA (Verdier, 2020) certificate is a function $V : \mathbb{R}^n \to \mathbb{R}$, $V \in C^1$ if there exists $\gamma \in \mathbb{R}_+$ such that the following is satisfied:*

$$V(x) \leq 0 \quad \forall x \in \mathcal{X}_I, \tag{14a}$$

$$V(x) > 0 \quad \forall x \in \partial\mathcal{X}_S, \tag{14b}$$

$$\dot{V}(x) \leq -\gamma \quad \forall x \in \{x \in \mathcal{X}_S : V(x) \leq 0\} \setminus \mathcal{X}_F, \tag{14c}$$

$$V(x) > \beta \quad \forall x \in \partial\mathcal{X}_F, \tag{14d}$$

$$\dot{V}(x) \leq -\gamma \quad \forall x \in \mathcal{X}_F \setminus \text{int}(\{x \in \mathcal{X}_S : V(x) \leq \beta\}), \tag{14e}$$

for some constant $\beta \in \mathbb{R}^-$.

Theorem 7 (*Reach-And-Stay While Avoid*). *Given a model (2) and a certificate corresponding to the given sets of interest, then (13) holds.* ■

The sub-level set of V given by β defines an invariant set contained with the final set, and ensures that trajectories reach this set in finite time without entering an unsafe region. Note that the specification described in (13) – and the corresponding certificate – permit trajectories to enter and leave the final set, on the condition that eventually (and within finite time), they enter goal set and do not leave the final set again. Here, the goal set is implicitly defined by the intersection of $\mathcal{X}_F$ and the β level set of the certificate, and is an invariant set contained fully within $\mathcal{X}_F$. However, the shape and size of this $\mathcal{X}_G$ is not specified a-priori, and can only be obtained after synthesis. The next certificate is motivated by a desire to allow $\mathcal{X}_G$ and $\mathcal{X}_F$ to be specified fully within the property.

3.7. Reach, avoid and remain

The final property we consider is again similar to the previous *Reach and Stay While Avoid* property, but, as with the ROA certificate, we seek to remove the existential quantifier over the goal set from (13). This means that the Reach Avoid Remain (RAR) property requires that trajectories remain within a final set after reaching a goal set, but for two given goal and final sets. We express this formally, as follows:

$$\forall \xi(t_0) \in \mathcal{X}_I, \exists T \in \mathbb{R}, \forall t \in [t_0, T] : \\ \xi(t) \in {\mathcal{X}_U}^{\complement} \wedge \xi(T) \in \mathcal{X}_G \wedge \forall \tau \geq T : \xi(\tau) \in \mathcal{X}_F. \tag{15}$$

Certificate 6 (*RAR*). *Define an unsafe set $\mathcal{X}_U = \mathcal{X} \setminus \mathcal{X}_S$, where $\mathcal{X}_S$ is a compact safe set, a compact initial set $\mathcal{X}_I \subset \text{int}(\mathcal{X}_S)$, a compact final $\mathcal{X}_F \subset \text{int}(\mathcal{X}_S)$, and a compact goal set $\mathcal{X}_G \subset \text{int}(\mathcal{X}_F)$ with non-empty interior. Let $V : \mathbb{R}^n \to \mathbb{R}$ be a RWA certificate, and a function $B : \mathbb{R}^n \to \mathbb{R}$, $B \in C^1$, such that:*

$$B(x) \leq 0 \ \forall x \in \mathcal{X}_G, \tag{16a}$$

$$B(x) > 0 \ \forall x \in \partial\mathcal{X}_F, \tag{16b}$$

$$\dot{B}(x) < 0 \ \forall x \in \{x : B(x) = 0\}. \tag{16c}$$

The pair (V, B) define a Reach-Avoid-Remain certificate. ■

As with the Stable-While-Avoid certificate, we choose to formulate this certificate as a pair of separate functions, rather than collapsing the conditions to a single function. This choice, which of course does not affect the soundness of the approach, renders synthesis practically easier and more modular.

Theorem 8 (*Reach-Avoid-Remain*). *Given a model (2) and a certificate pair V, B satisfying the conditions in Certificate 6, then (15) holds.* ■

We note that here, the certificate B is similar to a Barrier certificate as defined in (9), though with $\mathcal{X}_G$ as the initial set and $\partial\mathcal{X}_F$ as the unsafe set. We have restated the function in this context for clarity.

3.8. Summary and classification of properties

So far, we have presented a number of different properties that a dynamical model may conform to. These properties, and the certificates that sufficiently prove them to hold, can be complex and subtly different. However, we observe the following similarities between them:

- All certificates rely on a set on initial conditions $\mathcal{X}_I$. In the case of a Lyapunov function, this set is implicitly defined a-posteriori to the synthesis of the certificate.

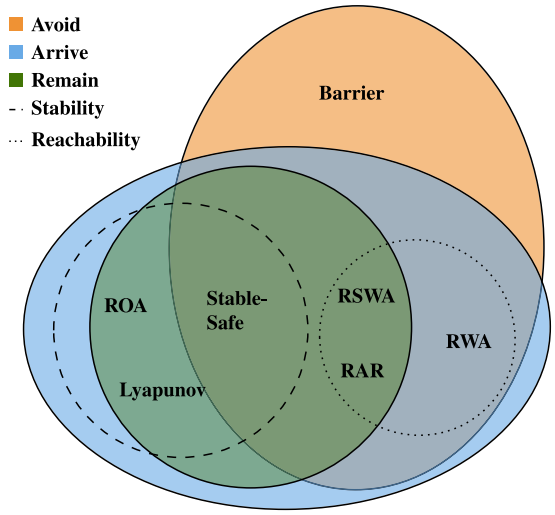


Fig. 2. Euler Diagram depicting the semantic labels, *arrive, avoid*, and *remain*, associated with each certificate in this work. By the dashed line, we group properties that exhibit asymptotic stability. By the dotted line, we denote properties that exhibit (finite-time) reachability.

- $\mathcal{X}_U$ denotes a region trajectories should *avoid* (and thus relate to a safety requirement).
- Either $\mathcal{X}_G$ or $\{x^*\}$ denote a region which trajectories should enter or *arrive* at. We leverage this notion to encompass both finite-time reachability and asymptotic stability, and note that it can be thought as the dual of the *avoid* category.
- Either $\mathcal{X}_F$ or $\{x^*\}$ denote a region which trajectories should eventually *remain* in, for all time, as soon as they have *arrived* in a goal set. This notion is thus related to both (forward) set invariance and asymptotically stable equilibria.

Based on these analogies, we introduce three labels *avoid*, *arrive* and *remain*, and assign them to each certificate, in order to clarify their purpose and differences. We portray this relationships in Fig. 2, where the label *arrive* encompasses both stability and finite-time reachability.

We note that this classification does not distinguish between some certificates, which we clarify here. Firstly, as previously mentioned, a Lyapunov function and a ROA certificate differ in that for the latter an initial set is explicitly specified a priori to synthesis. Meanwhile, the SWA, RSWA and RAR certificates satisfy all three labels. Stable while Avoid is easy to distinguish, as it handles asymptotic stabilty, whereas the others treat finite time reachability. Finally, we differentiate the RSWA and RAR certificates as follows. A RSWA certificate proves that there exists a goal set, contained within a given final set, that trajectories will reach and afterwards never leave the final set. Meanwhile a RAR certificate explicitly defines both the goal set and final set associated with this, allowing for a more elaborate specification.

3.9. Certificates for control models

Much work in the literature of certificate synthesis refers to, e.g., *control* Lyapunov functions and *control* Barrier certificates. In this work, we consider such terms to concern certificates that refer to a model expressed in terms of both state x and control input u, as in (1). Existing approaches often specify a control set, over which the specifications quantify existentially. In other words, they seek certificates for which for all states, there always exists a valid control action that allows for the property to hold. A control Lyapunov function therefore proves that there always exists a suitable control input such that the Lyapunov conditions hold, and hence the system is (asymptotically) stabilisable. This approach entails that, after a valid control-certificate is synthesised, control actions can be determined, e.g. by solving an optimisation program over the input space and the found certificate.

We emphasise that this is *not* the approach taken in this work. Instead, we synthesise a feedback control law for the model described by (1), and "apply" this state feedback to obtain a closed-loop model of the form of (2), for which we synthesise a certificate. Whilst we synthesise the control law *concurrently* with the certificate, we do not refer to these as "control certificates". Our approach of assuming some parametric form of a feedback controller and calculating a separate certificate is also common across literature, such as in SOSTOOLS (Papachristodoulou et al., 2013). We note that this notation is broadly, but not entirely, in line with other literature, and add this discussion as clarification on the terminology used in this work. Hence, we verify properties for control models using both a controller and a certificate, and in particular do not delegate the controller synthesis a-posteriori.

We note that in general neither approach has a clear benefit over the other, and more variants exist in some literature. However, verifying a control-Lyapunov certificate based on an exists/for-all query would, in general, scale poorly with SMT solvers. As such, we prefer to break the problem into two tasks of *guessing* a candidate controller and *checking* it with a certificate only dependent on the closed-loop dynamics.

4. Synthesis of certificates and controllers

Following the introduction of specifications that are salient for verification purposes, and certificates whose existence prove that a given model satisfies these conditions, we describe next a unified, efficient, and sound algorithm for the synthesis of these certificates, for both dynamical and controlled models. The objective of the synthesis task is indeed twofold: we seek a feedback controller $k(x)$ that ensures a control model satisfies a desired specification, and concurrently we synthesise a certificate $C(x)$ that serves as a proof that the specification holds. As such, allow us to use $C(x)$ to refer to each function in a possible pair separately. Call with θ_u the parameters of the state feedback law and θ_c the parameters of the certificate function. The synthesis task therefore amounts to finding values for the parameters θ_u and θ_c such that the certificate conditions hold, i.e.,

$$\exists\ \theta_u, \theta_c\ \forall x \in \mathcal{X}\ :\ \ \phi(C(x)), \tag{17}$$

where $\phi(C(x))$ denotes the conditions, related to a given specification (as formalised in the previous section), that the certificate ought to satisfy. Our procedure is based on counter-example guided inductive synthesis (CEGIS) (Solar-Lezama et al., 2006), an established approach to formally solving such exists-forall queries. CEGIS consists of two opposing components: a *leaner* and a *verifier* (cf. Fig. 4), which we detail in turn.

4.1. Learner

The learner fulfils the twofold task of designing both a stabilising control law (when required) and the desired certificate, as exemplified in Fig. 3. In this work, we seek *feedback* control laws and employ neural networks as a general template for these functions: neural architectures allow for a wide variety of control laws, as determined by the choice of their activation functions and of the number of neurons.

Our candidate controller is thus simply the output of a neural network, contributing to the closed-loop dynamics, and which is then employed within the certificate synthesis procedure.

For the certificate synthesis, we also employ a neural network as a template for $C(x)$: this allows not only for polynomial templates akin to classical (e.g., SOS-based) certificates, but also for highly non-linear functions, leveraging the expressive power of neural networks. One of the crucial components of a successful training within the learner is the definition of a tailored loss function.

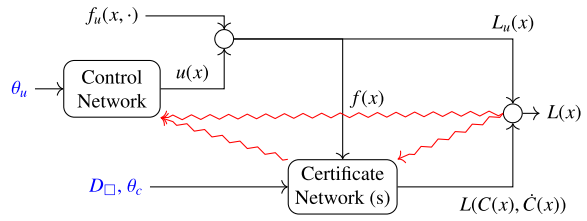


Fig. 3. Loss calculation block diagram. Blue indicates the inputs required for the loss calculation: namely the sampled data sets $D_\square$, the control and certificate network parameters θ_u and θ_c, respectively. Red arrows represent the back-propagation steps, updating the networks' parameters during training. $f_u(x,\cdot)$ corresponds to the model in (1), while $f(x)$ corresponds to the model in (2). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.1.1. Certificate loss

We note that all of the certificate conditions described in the previous section can be expressed in terms of inequalities, either as

$$C(x) \bowtie c \ \forall x \in \mathcal{X}_\square, \text{ or as } \dot{C}(x) \bowtie c \ \forall x \in \mathcal{X}_\square, \tag{18}$$

where $\bowtie := \{<, >, \geq, \leq\}$, $C(x)$ represents the certificate, $\mathcal{X}_\square$ represents the relevant set (e.g., $\mathcal{X}_I$ or $\mathcal{X}$) and $c \in \mathbb{R}$. We can then construct a loss function based on this observation, as follows. Consider a monotonically increasing function $m(\cdot)$ and suppose we have a finite set of sampled data points over the set $\mathcal{X}_\square$, which we denote as $D_\square$. A general loss function for any of the discussed certificate conditions is thus

$$\sum_{d \in D_\square} m(p \cdot C(d)), \tag{19}$$

where $p = 1$ if $\bowtie = \{\leq, <\}$ and $p = -1$ if $\bowtie = \{\geq, >\}$. Note that this function penalises points in $D_\square$ where the required condition is not satisfied. Suitable choices for function m are leaky-ReLU, which is piecewise linear, and softplus, which is smooth. Since several of the sets $\mathcal{X}_\square$ are boundaries of sets, or represent level sets, in practice we consider a small band around them, in order to encompass a sufficient number of data points in $D_\square$.

Example 1 (*Example*)**.** Let us consider a barrier certificate $B(x)$ for safety verification (see (9)). We create separate sets of finite samples for each set defined by the property (initial, unsafe, state-space), and the resulting loss function is given by

$$L = \frac{1}{N_I} \sum_{d \in D_I} m(B(d)) + \frac{1}{N_U} \sum_{d \in D_U} m(-B(d)) + \frac{1}{N} \sum_{d \in Z_B} m(\dot{B}(d)). \tag{20}$$

Here $Z_B(d) = \{d \in D : |B(d)| \leq \epsilon\}$, where D is the set of N samples over the whole state space, D_I is the set of N_I samples over the initial set and D_U is the set N_U of samples of the unsafe set. ■

4.1.2. Controller loss

Let us now discuss controlled models. Note that, as described in Fig. 3, the parameters of the neural network controller appear in the loss function for the certificate, as described previously. Specifically, they manifest themselves in the terms corresponding to conditions on the Lie derivative of the certificate, since these in turn depend on the control-dependent dynamics f. This should encourage the learner to seek a controller that enables a valid certificate. However, in practice we find that this is insufficient for robust synthesis in the case of *arrive* requirements. While our certificates and properties make no assumption on the location of the goal set or of the equilibrium, oftentimes an equilibrium point (without loss of generality, the origin) lies within the goal set. If this is the case, trajectories converging to the origin exhibiting a negative Lie derivative are in turn attracted to the goal set. Since the Lie derivative consists of two components, $f(x)$ and $\nabla C(x)$, it can be helpful to separate these elements within the loss function to encourage better learning. We thus propose the additional term for the loss function:

$$L_u = \frac{1}{N} \sum_{d \in D} \frac{\langle d, f(d) \rangle}{\|d\| \cdot \|f(d)\|}, \tag{21}$$

where $\langle \cdot, \cdot \rangle$ is the inner product of its inputs and $\| \cdot \|$ is the 2-norm of its input, and D is the set of N samples over the state space. This loss is known as the *cosine similarity* of the two vectors d and $f(d)$, and rewards vectors for pointing in opposite directions. We interpret this loss function as separating $f(x)$ from $\nabla C(x)$ when calculating the Lie derivative, and instead replacing the corresponding certificate with a default positive definite function (that of Euclidean distance). This encourages the loss function to specifically focus on the parameters in the feedback law to learn a desirable $f(x)$. As an alternative interpretation, since any vector d is fixed and points away from the origin, this encourages a controller which guides the dynamics to point towards to origin, and hence eventually converge towards it. We demonstrate the efficacy of this loss component in Section 5.2.

4.2. Verifier

We now discuss the dual component of the CEGIS architecture, as portrayed is Fig. 4. The verifier's role is assumed by an SMT solver and its operation is described as follows: let ϕ denote the conditions required for a given certificate. We seek a point $x \in \mathcal{X}$ that violates any of the constraints ϕ associated to the certificate. To this end, we express the *negation* of such requirements ϕ, and formulate a nonlinear constrained problem over real numbers. Formally, we ask an SMT solver to find a witness to

$$\exists x \in \mathcal{X} : \neg\phi(C(x)), \tag{22}$$

where any such witness x would be considered a counterexample to the validity of the certificate ϕ.

Example (*Example Cont'd*)**.** Consider the negation of barrier certificate conditions $\phi(C(x))$ as in (9), namely

$$(x \in \mathcal{X}_I \wedge B(x) > 0) \vee (x \in \mathcal{X}_U \wedge B(x) \leq 0) \vee (B(x) = 0 \wedge \dot{B}(x) \geq 0). \tag{23}$$

The verifier searches for solutions x of the constraints in (23). This in general requires manipulating non-convex functions and is therefore handled by an SMT solver. Whenever such x is found, it represents a witness that the candidate $B(x)$ is *not* a valid certificate function. ■

The correctness of our algorithm hinges upon the soundness of the verification engine: we use two solvers, Z3 (de Moura & Bjørner, 2008) and dReal (Gao et al., 2013), both of which are sound over nonlinear real arithmetic. Z3 is restricted to polynomial reasoning, whereas dReal can handle non-polynomial expressions, for instance containing trigonometric or exponential terms, thus allowing for more complex models and certificates (via their activation functions). We note that dReal is a δ-complete SMT-solver: while this guarantees dReal will always find a counterexample if one exists, it may return also spurious counterexamples within a δ-perturbation of the original formula (Gao et al., 2013). This implies that our procedure may not terminate, even when the candidate is valid. However quite importantly it does not compromise the correctness of certificates we verify successfully.

4.3. Enhanced communication amongst components

Our CEGIS approach builds on that of the software tool Fossil (Abate et al., 2021). As part of its CEGIS loop, Fossil adds two elements to enhance the communication between components.

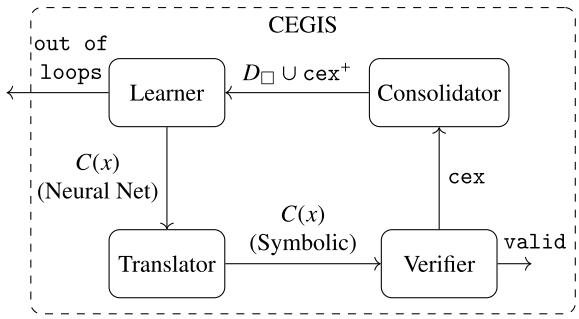


Fig. 4. Enhanced CEGIS architecture within Fossil.

Translator. The translator is tasked with the conversion of the neural networks into a symbolic candidate $C(x)$ and the corresponding $\dot{C}(x)$, ready to be processed by the verifier (see Fig. 4). The efficiency of SMT solvers depends also upon the numerical expressions of the formulae to be verified. Oftentimes the training returns numerically ill-conditioned expressions, e.g. unreasonably small coefficients (e.g., in the order of 10^{-8}); these candidates might slow the verification step, and thus the whole procedure. The translator thus rounds the coefficients of the candidate function to a specified precision, in order to help human interpretation and the verification process. Notably, this rounding step is performed *before* the SMT-based verification step (cf. Fig. 4, meaning the correctness of the synthesised certificates and controllers is not compromised.

Consolidator. Generating counterexamples is, in general, an expensive procedure, and the verification engine returns a single counterexample (`cex` in Fig. 4); e.g. an instance satisfying (23). Naturally, an isolated sample does not provide enough information for the learner to improve the candidate certificate. To overcome this issue, we randomly generate a *cloud* of points around the `cex` point, since samples around a counterexample are also likely to invalidate the certificate conditions. Secondly, starting from `cex`, we compute the gradient of C (or of $\dot{C}$) thanks to an automatic differentiation feature, and follow the direction that maximises the violation of the certificate constraints. The consolidator then aggregates the original counterexample and the newly generated points (denoted $\texttt{cex}^+$ in Fig. 4) with the sample set S for a new synthesis round by the learner.

4.4. Comments on specific certificates

We add a few remarks next, elaborating on practical synthesis details that are unique to specific certificates.

4.4.1. Lyapunov functions

We assume without loss of generality that the given model has an equilibrium at the origin, and that $k(0) = 0$ for the control law. We aid synthesis by enforcing the positivity condition by construction: in the case of positive-definite polynomial functions this is done by fixing the output layer's weights W_{k+1} to be all positive.

4.4.2. ROA

For verification, we estimate the smallest level set that contains $\mathcal{X}_I$ using sample points. Then, we verify that $\mathcal{X}_I$ is contained wholly within this level set and that the Lyapunov conditions hold over the entire level set. Due to dReal's inner workings, when verifying either Lyapunov or ROA certificates, we remove a very small region around the origin from the verification domain. This is common practice (Chang et al., 2019) across similar works, and does not affect sub-level sets outside this region. In fact, this notion is studied as ϵ-stability (Gao et al., 2019). This caveat does not apply when using Z3.

4.4.3. RSWA

The choice of γ for the RSWA and RWA certificates is arbitrary and may be fixed prior to synthesis. Meanwhile, the conditions described in (14d), (14e) depend on the existence of the parameter β. We must prove a value for this parameter exists such that the relevant conditions hold in order to prove the certificate is valid. After successfully verifying the conditions which do not depend on β, we perform a line search for β to find a suitable value (Verdier, 2020). If we do not find a valid β, we return to synthesis and keep training.

5. Computational experiments and benchmarks

We have implemented the proposed framework based on the computational library of Fossil. We have tested our approach across a large number of case studies, and benchmarked it against the first release of Fossil, showing improved results.

5.1. Main results

Our new prototype tool, which we refer to as Fossil 2.0,[3] is able to verify all properties described in Section 3 for continuous-time models, both autonomous and controlled. We showcase the efficacy of our framework and corresponding tool across 26 benchmarks, borrowed from existing literature on certificate synthesis (Abate et al., 2021; Sankaranarayanan et al., 2013; Vannelli & Vidyasagar, 1985; Verdier & Mazo, 2020). Note that, in some cases, we have modified these benchmarks to further challenge our approach, for instance by using disjoint, non-convex sets in the specifications. We consider a key strength of our approach to be its flexibility — we are able to perform well on straightforward and challenging benchmarks using certificates that represent both polynomials and more complex non-polynomial functions (as determined by the activation function of the neural network). We reflect this in our selection of benchmarks, including dynamics that are relatively simple and dynamics that involve transcendental and trigonometric functions. Due to the large number of benchmarks, details on the dynamics and sets can be found in an extended version of this paper (Edwards, Peruffo, & Abate, 2023), and in the corresponding code-base, https://github.com/oxford-oxcav/fossil, where additional benchmarks can be also found.

The results are reported in Table 1, where for each benchmark we outline the number of variables N_s and of control input N_u, the property to be verified (cf. acronyms introduced earlier), the number of neurons in each hidden layer and the corresponding activation functions for these layers. The number of neurons is denoted as a list, e.g. $[n_1, n_2]$ indicates that the first and second hidden layers are composed of n_1 and n_2 neurons, respectively. The activation functions for these hidden layers are denoted similarly. As mentioned, a strength of our methodology is its flexibility in terms of the form that certificates may take: we are able to synthesise polynomial certificates as well as non-polynomial certificates that represent more "neural-typical" functions —- this is illustrated in the "Activations" column of Table 1. By φ_j we denote that the layer represents a polynomial function of order j; σ_{sig} represents the sigmoid function, σ_{t} represents the hyperbolic tangent function, σ_{t^2} is the square of σ_{t} and σ_{soft} is the softplus function.

For almost all benchmarks, we use a linear control function. Our approach can handle more general nonlinear templates, but we emphasise that, as we solve a verification problem, rather than a control problem, we only seek a feedback law such that the property is satisfied by the closed-loop dynamics, and thus offer no guarantee on the optimality of this controller. Still, we use a nonlinear controller employing σ_{t} functions for the benchmark number 10 of Table 1.

[3] The features and interface of Fossil 2.0 are detailed in Edwards et al. (2024). We note for clarity that Fossil 2.0 is able to prove some properties for discrete-time models, but that this is not relevant to the topic of study in this work: continuous time dynamical models.

Table 1
Results of synthesising certificates for all properties presented in this work. The first column indexes the benchmarks. N_s: Number of states, N_u: number of control inputs. We show the *Property* being verified and network structure (*Neurons* and *Activations*). For certificates of two functions, comma-separated lists shows the different structures. Finally, we report success rate (S) and the minimum, mean (μ) and maximum computation time T over successful runs, in seconds. In brackets we show the time spent during the *learning* phase.

	N_s	N_u	Property	Neurons	Activations	T (s)			S (%)
						Min	μ	Max	
1	2	0	Stability	[6]	$[\varphi_2]$	0.01 (≈0.00)	0.16 (0.15)	1.50 (1.48)	100
2	3	0	Stability	[8]	$[\varphi_2]$	0.28 (≈0.00)	2.22 (0.45)	12.57 (3.31)	100
3	2	2	Stability	[4]	$[\varphi_2]$	0.07 (0.01)	0.19 (0.02)	0.47 (0.04)	100
4	2	2	Stability	[5]	$[\varphi_2]$	0.09 (0.01)	0.26 (0.02)	0.54 (0.03)	100
5	2	0	ROA	[5]	$[\sigma_{\mathrm{t2}}]$	0.71 (0.02)	1.17 (0.02)	2.59 (0.03)	50
6	3	3	ROA	[8]	$[\varphi_2]$	1.24 (0.02)	39.08 (0.03)	287.89 (0.04)	100
7	2	0	Safety	[15]	$[\sigma_{\mathrm{t}}]$	0.44 (0.35)	3.36 (2.90)	7.61 (7.11)	100
9	8	0	Safety	[10]	$[\varphi_1]$	12.63 (7.71)	51.97 (32.75)	70.59 (44.66)	70
10	3	1	Safety	[15]	$[\sigma_{\mathrm{t}}]$	1.57 (0.19)	11.87 (2.50)	51.08 (7.52)	90
11	3	0	SWA	[6], [5]	$[\varphi_2]$, $[\sigma_{\mathrm{t}}]$	0.19 (0.05)	2.46 (0.100)	12.10 (0.20)	90
12	2	0	SWA	[5], [5, 5]	$[\varphi_2]$, $[\sigma_{\mathrm{sig}}, \varphi_2]$	0.13 (0.06)	0.27 (0.14)	0.39 (0.20)	100
13	2	1	SWA	[8], [5]	$[\varphi_2]$, $[\varphi_2]$	0.06 (0.03)	0.20 (0.10)	0.58 (0.24)	90
14	3	1	SWA	[10], [8]	$[\varphi_2]$, $[\sigma_{\mathrm{t}}]$	4.06 (0.87)	19.81 (2.73)	103.49 (7.23)	90
15	2	0	RWA	[4]	$[\varphi_2]$	0.14 (0.09)	1.81 (1.75)	4.70 (4.63)	100
16	3	0	RWA	[16]	$[\varphi_2]$	1.36 (0.09)	14.10 (0.14)	72.97 (0.20)	90
17	2	1	RWA	[4, 4]	$[\sigma_{\mathrm{sig}}, \varphi_2]$	0.59 (0.27)	6.82 (3.32)	20.07 (11.46)	100
18	3	1	RWA	[5]	$[\varphi_2]$	0.46 (0.11)	16.06 (5.81)	72.47 (44.64)	80
19	2	2	RWA	[5]	$[\sigma_{\mathrm{sig}}]$	0.69 (0.40)	1.38 (0.94)	2.14 (1.90)	100
20	2	0	RSWA	[4]	$[\varphi_2]$	0.19 (0.03)	1.29 (1.04)	3.79 (3.37)	100
21	3	0	RSWA	[16]	$[\varphi_2]$	4.81 (0.13)	27.14 (0.19)	80.95 (0.25)	100
22	2	0	RSWA	[5, 5]	$[\sigma_{\mathrm{sig}}, \varphi_2]$	1.52 (0.06)	4.45 (0.19)	10.97 (0.35)	100
23	2	1	RSWA	[8]	$[\varphi_2]$	0.21 (0.05)	0.67 (0.25)	1.19 (0.91)	100
24	2	2	RSWA	[5, 5]	$[\sigma_{\mathrm{sig}}, \varphi_2]$	0.98 (0.16)	1.23 (0.28)	1.61 (0.46)	100
25	2	0	RAR	[6], [6]	$[\sigma_{\mathrm{soft}}]$, $[\varphi_2]$	6.65 (1.08)	24.74 (6.46)	77.80 (15.06)	100
26	2	2	RAR	[6, 6], [6, 6]	$[\sigma_{\mathrm{sig}}, \varphi_2]$, $[\sigma_{\mathrm{sig}}, \varphi_2]$	5.13 (1.34)	26.99 (9.90)	101.23 (60.14)	100

We measure the robustness performance by running each experiment 10 times, where we initialise the network with different weights and a new dataset across separate random seeds. Our procedure is not guaranteed to terminate, so after a maximum number of CEGIS loops we stop it and consider the overall run a failure. In general, we allow 25 CEGIS loops; for the SWA and RAR certificates we allow 100 CEGIS loops as these certificates are composed of two functions.

We consider two metrics to assess the quality of our framework when attempting to synthesise a certificate and controller: how often it returns a successful result, the success rate S, and how long the algorithm takes for these successful runs, the time T. Table 1 thus reports S, along with the average, minimum and maximum time for the procedure to terminate, under the T column, denoted μ, min and max, respectively. In brackets, we also denote the amount of time spent during the learning phase of our procedure, with approximately all remaining time spent during the verification phase.

The success rate is consistently close to 100%, with the minimum standing at 50% for a single benchmark, highlighting the robustness of our approach over the presented broad range of complex properties. The benchmark with a success rate of 50% is *Benchmark 5*, and is shown in Fig. 5(b). The difficulty of this benchmark is due to the nature of the dynamics, which admit only a non-polynomial global Lyapunov function (studied in Vannelli and Vidyasagar (1985)). Further, we seek to test our approach by seeking a region of attraction for this model with a *disjoint*, and thus non-convex, initial set. It is clear from the figure that the region of attraction cannot become too wide, or else it might fail over points of instability. We include a simple Lyapunov function for the model to demonstrate the benefit of using a ROA certificate. We depict a selection of other certificates in the rest of Fig. 5 for the interested reader, either through the phase portraits or through surface plots.

5.2. Control loss evaluation

We have presented a novel loss function with the purpose of encouraging trajectories to converge to the origin, which is often desirable in the case of *arrive* conditions, as discussed in Section 4.1.2 We evaluate the inclusion of this loss function in two ways: first, we consider the effect on success rate and computation time relative to not having the term across all control benchmarks for RWA, RSWA and RAR properties, of which there are 6. These results are presented in Table 2, where as before we present the success rate (this time over all 60 runs), and the average computation time for successful runs. It is clear that the inclusion of this loss term improves both the robustness and efficiency of the automated synthesis.

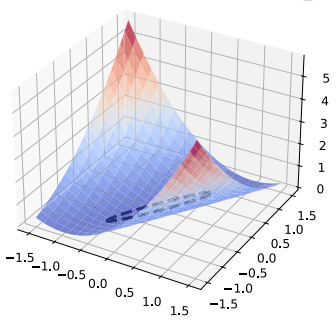
Table 2
Comparison of success rate and computation time for the combined RWA, RSWA and RAR benchmarks with control, with and without the loss term described in (21).

	Success (%)	Time (s)
With L_u	**96.67**	**7.14**
Without L_u	80.00	10.64

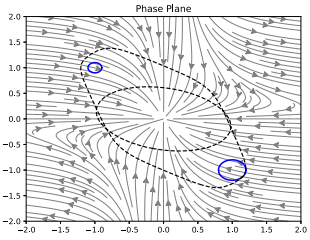
Secondly, we compare again these two metrics when instead using an LQR feedback controller. This is the purpose of benchmarks 22 and 24, which are identical except for one key difference: benchmark 22 is equipped with a pre-computed LQR controller, *de facto* representing an autonomous model, whilst we shall compute a control law in benchmark 24. This allows us to compare our framework's ability to learn feedback laws which satisfy properties relative to a common baseline controller, the known LQR. The results for these two benchmarks are very similar, with our control approach performing slightly more efficiently. Note, we do not claim to outperform LQR controllers, as other cost matrices may perform better or worse; further, we do not provide the cost minimisation guarantees — we simply use these benchmarks as a baseline comparison.

5.3. Comparison to Fossil 1.0 baseline

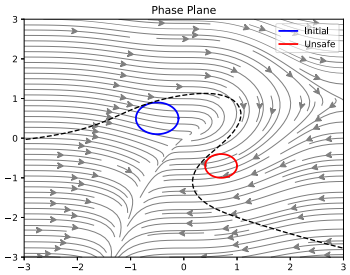
We have built a prototype tool based on our framework, improving



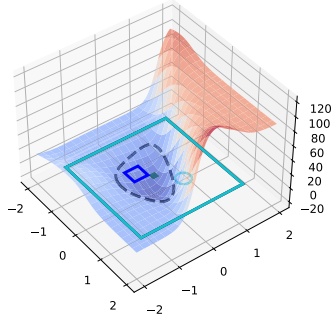
(a) Surface of a Lyapunov function (Benchmark 4) with the proven region of attraction shown in dashed line. This ellipsoidal shape is a typical example which motivates the use of the ROA certificate to enforce a specific region as stable.



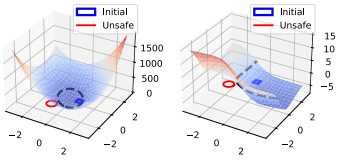
(b) ROA certificate (Benchmark 5) showing the largest verified sub-level set and corresponding region of attraction, with the level set (smaller dashed line) of a synthesised Lyapunov function shown for comparison.



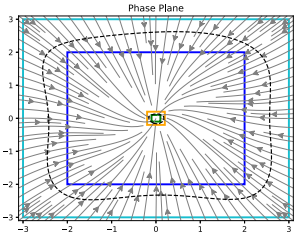
(c) Barrier certificate (Benchmark 7) showing the initial (blue) and unsafe (red) overlaid on a phase portrait, and the zero contour of the barrier certificate (dashed black)



(d) Surface plot of a RWA certificate (Benchmark 17) with zero level set as dashed line. The light blue set depicts a nonconvex safe set (note the circular region which is unsafe, inside the safe rectangular set). The safe set is the complement of the unsafe set; we choose to show the safe set here as it is more clear in the context.



(e) Surface plot of a SWA certificate (Benchmark 12) showing the constituent ROA (left) and Barrier (right) components of the certificate. The region of attraction and zero contour are shown of the ROA and Barrier components respectively in dashed black line.



(f) RAR certificate (Benchmark 26) with zero level sets for both the RWA function (outer dashed line) and function B (inner dashed line), see Certificate 6. See the main caption figure for a description of the coloured sets.

Fig. 5. Visualisations of a selection of certificates as either phase portraits (depicting the dynamics with relevant level sets of the certificate overlaid), or surface plots of the certificates (with relevant sets and level sets shown) from experiments in Table 1. We show salient level sets as dashed lines, and denote others sets as follows. Dark blue: $\mathcal{X}_I$; Red: $\mathcal{X}_U$; light blue: $\mathcal{X}_S$; green: $\mathcal{X}_G$; orange: $\mathcal{X}_F$. The grey arrows show the underlying vector field. The 3d surfaces of the certificates are also coloured to show the relative magnitude of the function (from blue for low relative values, to red for large ones). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

on the work initially presented in Abate et al. (2021). This tool compares against competitive state-of-the-art techniques such as SOS-tools, proving to deliver a faster synthesis for stability and safety properties for autonomous models. Table 3 collects works providing an automated and sound synthesis of relevant certificates. In general these related works either lack maintained or accessible supporting software tools with which experiments can be repeated and modified. This makes a direct comparison with these approaches infeasible beyond what has already been presented in Abate et al. (2021). Therefore, we focus our benchmarking against Fossil 1.0 as the state-of-the-art for certificate synthesis, as an accredited software tool for certificate synthesis that has been previously benchmarked against other approaches, such as SOS tools. These results are presented in Table 4.

We employ the benchmarks originally outlined in Abate et al. (2021), and for a fair comparison we use the same network structure (width and activations) for both tools. It is clear that the approaches are very similar when synthesising the more straightforward Lyapunov functions. However, we achieve significant improvements in terms of both success rate and synthesis time relative to the baseline, on account of a fine-tuned loss function, an enhanced communication between the CEGIS components, and an overall improved software implementation.

6. Discussions on generality

6.1. Asymptotic reachability

Oscillations are important and common phenomena occurring in dynamical systems, typically linked to (stable) limit cycles. Certificates for stability analysis in the presence of limit cycles, or equivalently for the asymptotic convergence to such set, have been so far notably absent from the synthesis framework in this manuscript. We discuss them next, recalling Barbashin-Krasowskii-Lasalle's Principle (Khalil, 2002).

Theorem 9 (*Invariance Principle Khalil, 2002*)**.** *Let $\Omega \subset \mathcal{X}$ be a compact set that is positively invariant with respect to (2). Let $V : \mathcal{X} \to \mathbb{R}$ be a continuously differentiable function such that $\dot{V}(x) \leq 0$. Let E be the set of all points in Ω where $V(x) = 0$. Let M be the largest invariant set in E. Then every solution starting in Ω approaches M as $t \to \infty$.* ■

Crucially, the principle states that we can prove asymptotic convergence towards a set thanks to a Lyapunov-like function which is equal to zero *exactly* at the limit cycle. Such certificates have recently been studied from the perspective of disturbed models (Meng, Li, Fitzsimmons et al., 2021; Meng, Li & Liu, 2021). Nonetheless, without prior knowledge of the existence and location of a limit cycle it is, in our experience, impractical to automatically synthesise such certificates, as they would need to be strongly templated based on the limit cycle (namely, precisely tailored to that set). Whilst in principle our approach could offer certificates for such properties, in this work we omit certificates requiring such an extensive analysis of the model dynamics.

Table 3
Comparison of works for automated (and sound) synthesis of certificates for continuous-time dynamical models. We show the properties verified in the respective works, and whether they are also able to verify these properties for control models (either using control certificates or controller and certificate). Publications have been grouped together if they represent the same line of work by the same set of authors or research groups and they study the same kind of property.

	Stability	ROA	Safety	SWA	RWA	RSWA	RAR	Control
Fossil 2.0	✓	✓	✓	✓	✓	✓	✓	✓
Fossil 1.0 (Abate et al., 2021)	✓	✗	✓	✗	✗	✗	✗	✗
F4CS (Verdier, 2020; Verdier & Mazo, 2020)	✓	✗	✓	✗	✓	✓	✗	✓
NLC (Chang et al., 2019)	✓	✗	✗	✗	✗	✗	✗	✓
Ravanbakhsh and Sankaranarayanan (2015a, 2015b, 2019)	✗	✗	✗	✗	✓	✓	✗	✓
Zhao, Chen et al. (2021)	✗	✗	✓	✗	✗	✗	✗	✗
Zhao et al. (2020) and Zhao, Zeng et al. (2021)	✗	✗	✓	✗	✗	✗	✗	✓
Ratschan (2017)	✗	✗	✓	✗	✗	✗	✗	✗
Kapinski et al. (2014)	✓	✗	✓	✗	✗	✗	✗	✗
Dai et al. (2020)	✓	✗	✗	✗	✗	✗	✗	✗
Grande, Anderlini et al. (2023) and Grande, Fenucci et al. (2023)	✓	✗	✗	✗	✗	✗	✗	✓

Table 4
Comparison of Fossil 1.0 vs. Fossil 2.0 (the present work). Here, we use the same naming scheme for benchmarks as used in Fossil 1.0, rather than indexing by number. See Table 1 for details on the columns.

Benchmark	N_s	Property	Neurons	Activations	Fosill 1.0				Fossil 2.0			
					Min	μ	Max	S	Min	μ	Max	S
NonPoly0	2	Stability	[5]	$[\varphi_2]$	0.04	0.21	1.58	100	0.01	0.16	1.54	100
Poly2	2	Stability	[5]	$[\varphi_2]$	0.35	11.71	70.39	90	0.08	4.77	6.50	100
Barr1	2	Safety	[10]	$[\varphi_1]$	0.34	1.00	2.72	40	0.02	0.27	0.63	100
Barr3	2	Safety	[10, 10]	$[\sigma_{sig}, \sigma_{sig}]$	16.80	101.72	334.79	50	3.81	14.14	30.63	100

6.2. Sufficiency of the certificates and completeness of their synthesis

The certificates provided in this work are *sufficient* proofs for the corresponding properties. We do not in general provide guarantees that a certificate exists if the property is satisfied, i.e. *necessary* proofs. Such converse results exists for Lyapunov and barrier functions. In particular, a construction method of Lyapunov functions is known for globally exponentially stable models (Khalil, 2002), whilst the necessity of barrier certificates is studied in, e.g., Prajna and Rantzer (2005), Ratschan (2018) and Wisniewski and Sloth (2015).

Contextually, whilst *sound*, our counterexample-based inductive synthesis method is not *complete*: whenever it finds a certificate, the desired property formally holds for the model under consideration. On the other hand, if our algorithm fails to find a certificate we cannot draw any conclusion about the validity of the property for the given model. We show that our procedure consistently terminates with a successful outcome.

6.3. Modularity of the synthesis and nested properties

We have not considered properties describing that of sequential reachability — namely trajectories arriving at a series of target sets before the goal set. We have considered properties obtained by conjunction of requirements, and we shall comment in Appendix B instances obtained via disjunction. We could similarly obtain certificates by manipulating via propositional logic, (e.g., conjunction and disjunction, rather than temporally, as suggested previously) requirements and corresponding certificates. Such properties tie into our approach of using certificates in a modular fashion, which is a relatively unexplored concept and represents an area of future work.

6.4. Issues of scale

Our CEGIS-based approach consists of a gradient descent based learning phase followed by an SMT dependent verification phase. While gradient descent is known to scale well to higher dimensions, nonlinear real arithmetic SMT does not in general scale well to higher dimensions. This problem is shared to all methodologies which rely on SMT. Possible mitigations, beyond an improvement of SMT performance, include the use of an alternative verification method, e.g., software as Marabou (Katz et al., 2019) for ReLU networks, or interval bound propagation based techniques.

6.5. Broader connections and taxonomy of properties

In Section 3 we have presented a diverse set of certifiable properties in terms of the behaviour of trajectories (that is, of solutions) of a given dynamical model. Furthermore, in Section 3.8, we have laid connections across such properties through the notions of *avoid, arrive*, and *remain*. In Appendix B, we further frame such requirements in broader contexts, providing a categorisation of these properties within known classes of specifications. To this end, we draw connections with formal languages (in particular, regular expressions) and with automata theory (specifically, deterministic finite automata) (Hopcroft & Ullman, 1979), and in passing we also informally relate to temporal logic for specifications of reactive models (Baier & Katoen, 2008; Clarke, Grumberg, Kroening, Peled, & Veith, 2018; Pnueli, 1977).

Appendix B is written for the benefit of readers with a background in the mentioned areas, or with an interest in a perspective on dynamical models grounded upon formal methods (Belta et al., 2017; Tabuada, 2009) — this part may be otherwise dispensed with, at no loss of understanding of the overall material.

7. Concluding remarks

We have presented a general framework to formally verify dynamical and control models via certificate synthesis, and introduced Fossil 2.0, a prototype software tool for the automated formal synthesis of a broad range of certificates. Certificate synthesis is based on a CEGIS loop, exploiting neural networks to provide candidate functions, which are then formally verified with the help of SMT solvers. Our approach is able to efficiently synthesise certificates for a wide range of properties

Table 5
List of Abbreviations defined in this work.

Abbreviation	Definition
ROA	Region of Attraction
RAR	Reach-Avoid-Remain
RSWA	Reach-and-Stay While Avoid
RWA	Reach-While-Avoid
SWA	Stable While Avoid
CEGIS	Counterexample-Guided Inductive Synthesis
SAT	Satisfiability
DPLL	Davis–Putnam–Logemann–Loveland
SMT	Satisfiability Modulo Theories

for both autonomous and control models, with varying complexity of dynamics and sets structure. We test our framework and corresponding tool on a number of benchmarks, outperforming a state-of-the-art tool for certificate synthesis and showing robustness to initialisation.

In future work, we hope to explore further the modular synthesis of certificates, as well as to extend our framework to stochastic models.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Alec was supported by the EPSRC Centre for Doctoral Training in Autonomous Intelligent Machines and Systems (EP/S024050/1)

Appendix A. Proof of theorems

Proof of Theorem 1. We state without proof from Lyapunov theory (Sastry, 1999) that the conditions in (6) imply that $f(x)$ is asymptotically stable, and that all sub-level sets of $V(x)$ fully contained within $\mathcal{X}$ are forward invariant. Since V is continuous and $\mathcal{X}$ is non-empty, then there exists some β such that the sub-level set $\Omega_\beta = \{x \in \mathcal{X} : V(x) \leq \beta\}$ is fully contained within $\mathcal{X}$ and $x^* \in \Omega_\beta$. Since x^* is asymptotically stable, all trajectories in Ω_β converge towards it, and (5) holds for the initial set Ω_β. □

Proof of Corollary 2. Define the β sub-level of the ROA certificate V as $\Omega_\beta := \{x \in \mathcal{X} : V(x) \leq \beta\}$. By construction, $\mathcal{X}_I \subset \Omega_\beta$, and the condition of (6) hold. The proof then follows directly from Theorem 1. □

Theorem 10 (*Nagumo's Theorem*)**.** *Let us state without proof Nagumo's Theorem. For a proof, see Blanchini and Miani (2008). Consider the system $\dot{\xi}(t) = 3f(\xi(t))$, where f is Lipschitz continuous such that for each initial condition $\xi(t_0) \in \mathcal{X}$ it admits a unique solution. Let $\Omega \in \mathcal{X}$ be a closed set. Ω is positively invariant if and only if for every exterior normal vector v at point x on the border of $\partial\Omega$, the inner product satisfies $\langle f(x), v\rangle \leq 0$.*

Proof of Theorem 3. By definition we have that $\mathcal{X}_I \cap \mathcal{X}_U = \emptyset$. The set $\Omega_0 = \{x \in \mathcal{X} : \dot{B}(x) \leq 0\}$ defines a closed set for which (9c) ensures that $f(x)$ points inwards along its border. It follows from 10 that Ω_0 is an forward invariant set which contains $\mathcal{X}_I$, and that at all trajectories initialised in $\mathcal{X}_I$ remain within Ω_0 for all time $t > t_0$. (9b) ensures that $\Omega_0 \cap \mathcal{X}_U = \emptyset$, and hence (8) holds. □

Proof of Corollary 4. We note that the specification described by (10) is the conjunction of those in (5) and (8). Therefore, the proof follows directly from Corollary 2 and Theorem 3. □

Proof of Theorem 5 (from Verdier & Mazo (2020)). Let $A = \{x \in \mathcal{X}_S : V(x) \leq 0\}$. For $\xi(t_0) \in \mathcal{X}_I$, it follows from (12a) and the definition of A that $\xi(t_0) \in A$. From (12c), for all $\xi(t_k) \in A \setminus \mathcal{X}_G$, $\dot{V}(\xi(t_k)) < -\gamma$. Using $\forall x \in A, V(x) \leq 0$ and the comparison principle (Khalil, 2002), it follows that $\forall k \in \mathbb{Z}_{\geq 0}, \forall t \in [t_k, t_k + h], \forall \xi(t_k) \in A \setminus \mathcal{X}_G : V(\xi(t)) \leq V(\xi(t_k)) - \gamma h \leq -\gamma h$. Therefore, $\xi(t_k) \in A \setminus \mathcal{X}_G$ implies $\forall t \in [t_k, t_k + h], V(\xi(t))$ will decrease and thus cannot reach $\partial\mathcal{X}_S$, since by (12b) $\forall x \in \partial\mathcal{X}_S : V(x) > 0$. Since $\mathcal{X}_S$ is compact and $V(x)$ is continuous, $\exists e \in \mathbb{R}$ s.t. $e = \inf_{x \in \mathcal{X}_S \setminus \mathcal{X}_G} V(x)$ and the sublevel sets of $V(x)$ is compact. It follows that $V(x)$ is lower bounded on $A \setminus \mathcal{X}_G \subset \mathcal{X}_S \setminus \mathcal{X}_G$, so $V(\xi(t))$ will decrease until in finite time $\xi(t)$ leaves $A \setminus \mathcal{X}_G$ and may only enter $\mathcal{X}_G$. □

Proof of Theorem 7 (from Verdier & Mazo (2020)). Let $B = \{x \in \mathcal{X}_S : V(x) \leq \beta\}$. From Theorem 5, there exists a time $T \geq t_0$ such that $\xi(T) \in \mathcal{X}_F$. Using a similar argument as before, we conclude that $\forall \xi(T) \in \mathcal{X}_F, \xi(t)$ with $t \geq T$ enters in finite time $\mathcal{X}_F \cap B$. Since B is a sub-level set of the compact set S and continuous V, then B is also compact and so is $\mathcal{X}_F \cap B$. From (14e) we have $\forall x \in \partial(\mathcal{X}_F \cap B) : \dot{V}(x) \leq -\gamma$. Combining with (14d), we have that all states $\xi(t) \in \partial(\mathcal{X}_F \cap B)$ cannot reach $\partial\mathcal{X}_F$ and $V(\xi(t))$ decreases, meaning these trajectories remain in $\mathcal{X}_F \cap B$. Therefore, $\mathcal{X}_F \cap B$ is forward invariant. Since $\mathcal{X}_F \subset \text{int}(\mathcal{X}_S)$, we have that (13) holds. □

Proof of 8. This proof follows directly from Theorems 3 and 5. □

Appendix B. Broader connections and taxonomy of properties

In this part, we further frame the presented requirements in broader contexts, thus providing a categorisation of the properties under study within specific classes of *formal specifications* (Baier & Katoen, 2008; Clarke et al., 2018). In order to do so, we draw connections with formal languages (in particular, regular expressions) and with automata theory (specifically, deterministic finite automata) (Hopcroft & Ullman, 1979), and in passing we also informally relate to the use of temporal logic for specifications of reactive models (Baier & Katoen, 2008; Clarke et al., 2018; Pnueli, 1977). However, these connections can be drawn under an important proviso: that is, in formal methods, properties by and large are expressed over finite alphabets, namely over a finite set of labels (which can be related to our sets – or complement thereof – of interest), but most importantly concern traces in discrete time. Similarly, formal languages and automata deal with finite or countably-infinite strings of (finite) characters.

Safety and reachability - language and semantic duality. We remark that all the presented properties consist of combinations of two fundamental specifications in formal verification (Kupferman & Vardi, 1999; Manna & Pnueli, 1990). The first is *safety*, which qualitatively concerns trajectories always staying clear from a nominative region in the state space that is deemed to be unsafe. Thus, safety specifications are infinite-horizon requirements, which however can be equivalently defined as requirements admitting finite-horizon counter-examples: namely, they are specifications that are necessarily invalidated by trajectories that enter the given unsafe in finite time.

The second specification we refer to as *reachability*, and comprise trajectories reaching a given desirable goal set (which is also known as *reach* or *target* set). In the area of formal languages, reachability is known to be a *co-safety* property, namely a dual to a safety requirement and, as such, it is a finite-horizon property. Indeed, in formal verification co-safe specifications admit finite-horizon witnesses, that is satisfying trajectories that enter the goal set in finite time. In Section 3 we have defined this as unconstrained reachability, and as a special instance of the reach-while avoid specification. We could have alternatively introduced it as the logical dual of safety, except that the certificate synthesis would not have followed as seamlessly. We should mention that, more generally, co-safety properties (and, in

particular, reachability) are special instances of another broad class of specifications, known as *liveness* properties (Baier & Katoen, 2008; Clarke et al., 2018): in this work we do not deal with general liveness properties, and in particular the synthesis of sufficient certificates for this class is left as future work.

Finite- and infinite-horizon requirements. We have discussed that safety and reachability properties are dual in the sense that the earlier raises a requirement over infinite-horizon trajectories, asking that *nothing bad ever happens*, whereas the latter requires reasoning about finite-horizon solutions, asking that over a finite time horizon *something good eventually happens* (Baier & Katoen, 2008). However note that, whenever safety requirements are added to finite-time reachability properties, as in the case of the *avoid* constraint in RWA, RSWA, and RAR, these safety requirements ought to hold only over the finite time spans inherited from the reachability requirements.

Dually, as much as reachability requirements natively encompass co-safety properties, in this work we have discussed extension of such reachability requirements over *infinite-time* horizons: this is quite natural in control theory, namely in the context of asymptotic stability. Similarly, we have not only raised requirements that trajectories reach a goal set within a finite time horizon, which we have simply denoted as *reachability*, but additionally discussed (cf. 6.1) that they may approach the set (e.g., a set of equilibria, or a limit cycle) asymptotically, which we have denoted as *asymptotic reachability*.

Automata theory. Next, we qualitatively relate these classes of properties to automata theory. Through their duality, both safety and co-safety properties can be expressed by the same class of finite-state models, namely deterministic finite automata (DFA) (Hopcroft & Ullman, 1979). In other words, traces within these two classes of specifications can be compiled by a DFA model, which reads them when they are started in one of its initial states, and when they terminate upon hitting one its accepting states. (Conversely, liveness properties, which we saw generalise co-safety specifications, require automata of different nature, such as Büchi or Rabin automata, which specifically accept infinite traces.) Summarising this discussion, we can conclude that we can provide sufficient certificates for properties that can be expressed as DFAs: an enticing extension of our work is looking at richer specifications obtained by modularly composing such finite-state models.

Linear-time properties and temporal logic. Finally, let us draw connections to temporal logic, a class of modal logic that was natively developed to specify requirements of (models of) reactive programs (Pnueli, 1977). Whilst originally employed for finite-state programs evolving over discrete-time steps, temporal logics, such as LTL (linear TL), have been also widely employed in the context of dynamical and control models (Belta et al., 2017; Tabuada, 2009). Bearing in mind that the above caveat on discrete-time semantics holds also in this context (which for instance renders its *next* operator useless for our purposes), we can express safety requirements over safe set S as *always* S - in LTL syntax this is expressed as the formula $\square S$ (or $\mathsf{G}S$) - and, dually, reachability requirements over target set T as *eventually* T - in LTL this is $\diamond T$ (or $\mathsf{F}T$); incidentally, the discussed duality between these requirements is crisply expressed logically as: $\square S \equiv \neg \diamond \neg S \equiv \neg \diamond S^{\complement}$. Similar discussions follow through for finite-time properties, which in LTL require selecting a (finite) horizon $\tau \in \mathbb{R}$, yielding for example $\square^{\leq \tau} S$. Reach-avoid can be expressed via the *until* operator U in LTL, say $S \,\mathsf{U}\, T$, which may be tailored to unconstrained reachability as $\diamond T \equiv \texttt{true}\,\mathsf{U}\,T$.

Dovetailing to the discussion above, we can thus flexibly and modularly synthesise certificates also for the known *weak until* W, and even for the *release* R specifications in LTL. Indeed, recall that Baier and Katoen (2008)

$$S \,\mathsf{W}\, T = (S \,\mathsf{U}\, T) \vee \square S,$$

We can thus obtain a certificate for weak until from either a certificate for reach-avoid or one for safety, respectively. Additionally notice in particular, that $\square S \equiv S \,\mathsf{W}\, \texttt{false}$.

Finally, the useful *release* operator in LTL expresses a safety requirement (set S) that is released upon reaching a target set (T), as follows:

$$T \,\mathsf{R}\, S = \neg(\neg T \,\mathsf{U}\, \neg S),$$

and in fact (the next equivalent expression is easier for finding certificates)

$$T \,\mathsf{R}\, S = (\neg T \wedge S) \mathsf{W} (T \wedge S),$$

and so, in particular, $\square S \equiv \texttt{false}\,\mathsf{R}\,S$.

Data availability

The code is publicly available and linked in the paper.

References

Abate, A. (2017). Formal verification of complex systems: Model-based and data-driven methods. In *MEMOCODE 2017 - 15th ACM-IEEE international conference on formal methods and models for system design* (pp. 91–93).

Abate, A., Ahmed, D., Edwards, A., Giacobbe, M., & Peruffo, A. (2021). FOSSIL: A software tool for the formal synthesis of Lyapunov functions and barrier certificates using neural networks. In *HSCC '21, Proceedings of the 24th international conference on hybrid systems: computation and control* (pp. 1–11). New York, NY, USA: Association for Computing Machinery.

Abate, A., Ahmed, D., Giacobbe, M., & Peruffo, A. (2020). Automated formal synthesis of Lyapunov neural networks. *IEEE Control Systems Letters*.

Abate, A., Bessa, I., Cattaruzza, D., Cordeiro, L., David, C., Kesseli, P., et al. (2020). Automated formal synthesis of provably safe digital controllers for continuous plants. *Acta Informatica, 57*(3), 223–244.

Abate, A., Edwards, A., & Giacobbe, M. (2022). Neural abstractions. In *Thirty-sixth conference on neural information processing systems*.

Ahmed, D., Peruffo, A., & Abate, A. (2020). Automated and sound synthesis of Lyapunov functions with SMT solvers. In *International conference on tools and algorithms for the construction and analysis of systems*.

Ames, A. D., Xu, X., Grizzle, J. W., & Tabuada, P. (2017). Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control, 62*(8), 3861–3876.

Baier, C., & Katoen, J.-P. (2008). *Principles of model checking*. MIT Press.

Barbosa, H., Barrett, C. W., Brain, M., Kremer, G., Lachnitt, H., Mann, M., et al. (2022). cvc5: A versatile and industrial-strength SMT solver. In D. Fisman, G. Rosu (Eds.), *Lecture notes in computer science: vol. 13243, Tools and algorithms for the construction and analysis of systems - 28th international conference, TACAS 2022, held as part of the European joint conferences on theory and practice of software, ETAPS 2022, Munich, Germany, April 2-7, 2022, proceedings, part I* (pp. 415–442). Springer.

Barrett, C., Stump, A., Tinelli, C., et al. (2010). The smt-lib standard: Version 2.0. *Vol. 13*, In *Proceedings of the 8th international workshop on satisfiability modulo theories (Edinburgh, UK)* (p. 14).

Barrett, C., & Tinelli, C. (2018). Satisfiability modulo theories. In E. M. Clarke, T. A. Henzinger, H. Veith, & R. Bloem (Eds.), *Handbook of model checking* (pp. 305–343). Springer International Publishing.

Belta, C., Yordanov, B., & Gol, E. A. (2017). *Formal methods for discrete-time dynamical systems: vol. 15*, Springer.

Ben Sassi, M. A., Sankaranarayanan, S., Chen, X., & Ábrahám, E. (2016). Linear relaxations of polynomial positivity for polynomial Lyapunov function synthesis. *IMA Journal of Mathematical Control and Information, 33*(3), 723–756.

Blanchini, F., & Miani, S. (2008). *Set-theoretic methods in control*. Boston, MA: Birkhäuser Boston.

Bohrer, R., Tan, Y. K., Mitsch, S., Myreen, M. O., & Platzer, A. (2018). VeriPhy: Verified controller executables from verified cyber-physical system models. In *ACM SIGPLAN conference on programming language design and implementation* (pp. 617–630).

Chang, Y.-C., Roohi, N., & Gao, S. (2019). Neural Lyapunov control. *Advances in Neural Information Processing Systems, 32*.

Chen, S., Fazlyab, M., Morari, M., Pappas, G. J., & Preciado, V. M. (2020). Learning Lyapunov functions for piecewise affine systems with neural network controllers. arXiv:2008.06546 [math].

Chen, S., Fazlyab, M., Morari, M., Pappas, G. J., & Preciado, V. M. (2021). Learning Lyapunov functions for hybrid systems. In *Proceedings of the 24th international conference on hybrid systems: computation and control* (pp. 1–11). Nashville Tennessee: ACM.

Clarke, E., Grumberg, O., Kroening, D., Peled, D., & Veith, H. (2018). *Principles of model checking* (2nd ed.). MIT Press.

Dai, H., Landry, B., Pavone, M., & Tedrake, R. (2020). Counter-example guided synthesis of neural network Lyapunov functions for piecewise linear systems. In *2020 59th IEEE conference on decision and control* (pp. 1274–1281).

Dai, H., Landry, B., Yang, L., Pavone, M., & Tedrake, R. (2021). Lyapunov-stable neural-network control. In *Robotics: science and systems XVII*. Robotics: Science and Systems Foundation.

Davis, M., Logemann, G., & Loveland, D. W. (1962). A machine program for theorem-proving. *Communications of the ACM*, *5*(7), 394–397.

Davis, M., & Putnam, H. (1960). A computing procedure for quantification theory. *Journal of the ACM*, *7*(3), 201–215.

Dawson, C., Gao, S., & Fan, C. (2022). Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods. *IEEE Transactions on Robotics*, *39*, 1749–1767.

de Moura, L., & Bjørner, N. (2008). Z3: An efficient SMT solver. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. R. Ramakrishnan, & J. Rehof (Eds.), *Tools and algorithms for the construction and analysis of systems: vol. 4963*, (pp. 337–340). Berlin, Heidelberg: Springer Berlin Heidelberg.

Edwards, A., Peruffo, A., & Abate, A. (2023). A general framework for verification and control of dynamical models via certificate synthesis. arXiv:2309.06090.

Edwards, A., Peruffo, A., & Abate, A. (2024). Fossil 2.0: Formal certificate synthesis for the verification and control of dynamical models. In *Proceedings of the 27th ACM international conference on hybrid systems: computation and control*. Association for Computing Machinery.

Gao, S., Kapinski, J., Deshmukh, J., Roohi, N., Solar-Lezama, A., Arechiga, N., et al. (2019). Numerically-robust inductive proof rules for continuous dynamical systems. In I. Dillig, S. Tasiran (Eds.), *Computer aided verification* (pp. 137–154). Cham: Springer International Publishing.

Gao, S., Kong, S., & Clarke, E. M. (2013). dReal: An SMT solver for nonlinear theories over the reals. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, & M. P. Bonacina (Eds.), *Automated deduction – CADE-24: vol. 7898*, (pp. 208–214). Berlin, Heidelberg: Springer Berlin Heidelberg.

Goubault, E., Jourdan, J.-H., Putot, S., & Sankaranarayanan, S. (2014). Finding non-polynomial positive invariants and Lyapunov functions for polynomial systems through Darboux polynomials. In *American control conference*. Portland, United States.

Grande, D., Anderlini, E., Peruffo, A., & Salavasidis, G. (2023). Augmented neural Lyapunov control. *IEEE Access*.

Grande, D., Fenucci, D., Peruffo, A., Anderlini, E., Phillips, A. B., Giles, T., et al. (2023). Systematic synthesis of passive fault-tolerant augmented neural Lyapunov control laws for nonlinear systems. In *2023 62nd IEEE conference on decision and control*.

Gurobi Optimization, LLC (2021). Gurobi optimizer reference manual.

Henzinger, T. (1996). The theory of hybrid automata. In *proceedings 11th annual IEEE symposium on logic in computer science* (pp. 278–292).

Hopcroft, J. E., & Ullman, J. D. (1979). *Introduction to automata theory, languages, and computation*. Addison-Wesley Publishing.

Huang, Z., Wang, Y., Mitra, S., Dullerud, G. E., & Chaudhuri, S. (2015). Controller synthesis with inductive proofs for piecewise linear systems: An SMT-based algorithm. In *2015 54th IEEE conference on decision and control* (pp. 7434–7439). IEEE.

isat3. https://projects.informatik.uni-freiburg.de/projects/isat3/.

Jin, W., Wang, Z., Yang, Z., & Mou, S. (2020). Neural certificates for safe control policies. arXiv:2006.08465 [cs, eess].

Kapinski, J., Deshmukh, J. V., Sankaranarayanan, S., & Arechiga, N. (2014). Simulation-guided Lyapunov analysis for hybrid dynamical systems. In *HSCC '14*, *Proceedings of the 17th international conference on hybrid systems: computation and control* (pp. 133–142). New York, NY, USA: Association for Computing Machinery.

Katz, G., Huang, D. A., Ibeling, D., Julian, K., Lazarus, C., Lim, R., et al. (2019). The marabou framework for verification and analysis of deep neural networks. In I. Dillig, S. Tasiran (Eds.), *Lecture notes in computer science*, *Computer aided verification* (pp. 443–452). Cham: Springer International Publishing.

Khalil, H. K. (2002). *Nonlinear systems* (3rd ed.). Upper Saddle River, N.J: Prentice Hall.

Knight, J. C. (2002). Safety critical systems: challenges and directions. In *ICSE* (pp. 547–550). ACM.

Kupferman, O., & Vardi, M. (1999). Model checking of safety properties. *Formal Methods in System Design*, *19*.

Lyapunov, A. M. (1992). The general problem of the stability of motion. *International Journal of Control*, *55*(3), 531–534.

MacKay, D. J. C. (2003). *Information theory, inference, and learning algorithms*. Cambridge, UK ; New York: Cambridge University Press.

Manna, Z., & Pnueli, A. (1990). A hierarchy of temporal properties. In *Proceedings of the ninth annual ACM symposium on principles of distributed computing* (pp. 377–410).

Masti, D., Fabiani, F., Gnecco, G., & Bemporad, A. (2023). Counter-example guided inductive synthesis of control Lyapunov functions for uncertain systems. *IEEE Control Systems Letters*.

Meng, Y., Li, Y., Fitzsimmons, M., & Liu, J. (2021). Smooth converse Lyapunov-barrier theorems for asymptotic stability with safety constraints and reach-avoid-stay specifications.

Meng, Y., Li, Y., & Liu, J. (2021). Control of nonlinear systems with reach-avoid-stay specifications: A Lyapunov-barrier approach with an application to the Moore-Greizer model. In *2021 American control conference* (pp. 2284–2291).

Noroozi, N., Karimaghaee, P., Safaei, F., & Javadi, H. (2008). Generation of Lyapunov functions by neural networks. (p. 5).

Papachristodoulou, A., Anderson, J., Valmorbida, G., Prajna, S., Seiler, P., & Parrilo, P. (2013). SOSTOOLS version 3.00 sum of squares optimization toolbox for MATLAB. arXiv:1310.4716 [cs, math].

Papachristodoulou, A., & Prajna, S. (2002). On the construction of Lyapunov functions using the sum of squares decomposition. *Vol. 3*, In *Proceedings of the 41st IEEE conference on decision and control, 2002.* (pp. 3482–3487). Las Vegas, NV, USA: IEEE.

Peruffo, A., Ahmed, D., & Abate, A. (2021). Automated and formal synthesis of neural barrier certificates for dynamical models. In *International conference on tools and algorithms for the construction and analysis of systems* (pp. 370–388). Springer.

Pnueli, A. (1977). The temporal logic of programs. In *18th annual symposium on foundations of computer science* (pp. 46–57).

Prajna, S. (2006). Barrier certificates for nonlinear model validation. *Automatica (Journal of IFAC)*, *42*(1), 117–126.

Prajna, S., Jadbabaie, A., & Pappas, G. (2004). Stochastic safety verification using barrier certificates. *Vol. 1*, In *2004 43rd IEEE conference on decision and control (CDC) (IEEE cat. no.04CH37601)* (pp. 929–934). Nassau, Bahamas: IEEE.

Prajna, S., & Rantzer, A. (2005). On the necessity of barrier certificates. *IFAC Proceedings Volumes*, *38*(1), 526–531.

Ratschan, S. (2017). Simulation based computation of certificates for safety of dynamical systems. In *Formal modeling and analysis of timed systems: 15th international conference, FORMATS 2017, Berlin, Germany, September 5–7, 2017, proceedings 15* (pp. 303–317). Springer.

Ratschan, S. (2018). Converse theorems for safety and barrier certificates. *IEEE Transactions on Automatic Control*, *63*(8), 2628–2632.

Ratschan, S., & She, Z. (2010). Providing a basin of attraction to a target region of polynomial systems by computation of Lyapunov-like functions. *SIAM Journal on Control and Optimization*, *48*(7), 4377–4394.

Ravanbakhsh, H., & Sankaranarayanan, S. (2015a). Counter-example guided synthesis of control Lyapunov functions for switched systems. In *2015 54th IEEE conference on decision and control* (pp. 4232–4239).

Ravanbakhsh, H., & Sankaranarayanan, S. (2015b). Counterexample guided synthesis of switched controllers for reach-while-stay properties. arXiv:1505.01180 [cs].

Ravanbakhsh, H., & Sankaranarayanan, S. (2019). Learning control Lyapunov functions from counterexamples and demonstrations. *Autonomous Robots*, *43*(2), 275–307.

Richards, S. M., Berkenkamp, F., & Krause, A. (2018). The Lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems. In *Conference on robot learning* (pp. 466–476). PMLR.

Romdlony, M. Z., & Jayawardhana, B. (2016). Stabilization with guaranteed safety using control Lyapunov–barrier function. *Automatica*, *66*, 39–47.

Samanipour, P., & Poonawala, H. A. (2023). Stability analysis and controller synthesis using single-hidden-layer ReLU neural networks. *IEEE Transactions on Automatic Control*, 1–12.

Sankaranarayanan, S., Chen, X., & Ábrahám, E. (2013). Lyapunov function synthesis using handelman representations. *IFAC Proceedings Volumes*, *46*(23), 576–581.

Sastry, S. (1999). In J. E. Marsden, L. Sirovich, & S. Wiggins (Eds.), *Interdisciplinary applied mathematics: vol. 10*, *Nonlinear systems*. New York, NY: Springer New York.

She, Z., Li, H., Xue, B., Zheng, Z., & Xia, B. (2013). Discovering polynomial Lyapunov functions for continuous dynamical systems. *Journal of Symbolic Computation*, *58*, 41–63.

She, Z., Xia, B., Xiao, R., & Zheng, Z. (2009). A semi-algebraic approach for asymptotic stability analysis. *Nonlinear Analysis: Hybrid Systems*, *3*(4), 588–596.

Solar-Lezama, A., Tancau, L., Bodik, R., Seshia, S., & Saraswat, V. (2006). Combinatorial sketching for finite programs. *SIGOPS Operating Systems Review*, *40*(5), 404–415.

Tabuada, P. (2009). *Verification and control of hybrid systems: A symbolic approach*. Springer.

Tan, X., Cortez, W. S., & Dimarogonas, D. V. (2022). High-order barrier functions: Robustness, safety, and performance-critical control. *IEEE Transactions on Automatic Control*, *67*(6), 3021–3028.

Vannelli, A., & Vidyasagar, M. (1985). Maximal Lyapunov functions and domains of attraction for autonomous nonlinear systems. *Automatica*, *21*(1), 69–80.

Verdier, C. F. (2020). Formal synthesis of analytic controllers: An evolutionary approach.

Verdier, C. F., & Mazo, M., Jr. (2020). Formal controller synthesis for hybrid systems using genetic programming. arXiv:2003.14322 [cs, eess].

Wisniewski, R., & Sloth, C. (2015). Converse barrier certificate theorems. *IEEE Transactions on Automatic Control*, *61*(5), 1356–1361.

Wu, Z., Albalawi, F., Zhang, Z., Zhang, J., Durand, H., & Christofides, P. D. (2019). Control Lyapunov-barrier function-based model predictive control of nonlinear systems. *Automatica*, *109*, Article 108508.

Zhao, Q., Chen, X., Zhang, Y., Sha, M., Yang, Z., Lin, W., et al. (2021). Synthesizing ReLU neural networks with two hidden layers as barrier certificates for hybrid systems. *no. 17*, In *Proceedings of the 24th international conference on hybrid systems: computation and control* (pp. 1–11). New York, NY, USA: Association for Computing Machinery.

Zhao, H., Zeng, X., Chen, T., & Liu, Z. (2020). Synthesizing barrier certificates using neural networks. In *HSCC '20, Proceedings of the 23rd international conference on hybrid systems: computation and control* (pp. 1–11). New York, NY, USA: Association for Computing Machinery.

Zhao, H., Zeng, X., Chen, T., Liu, Z., & Woodcock, J. (2021). Learning safe neural network controllers with barrier certificates. *Formal Aspects of Computing*, *33*(3), 437–455.