SReach: Combining Statistical Tests and Bounded Model Checking for Nonlinear Hybrid Systems with Parametric Uncertainty

Qinsi Wang[1], Paolo Zuliani[2], Soonho Kong[1], Sicun Gao[1], and Edmund Clarke[1]

[1] Computer Science Department, Carnegie Mellon University, USA
[2] School of Computing Science, Newcastle University, UK

Abstract. We present a novel approach for solving the probabilistic bounded reachability problem of hybrid systems with parameter uncertainty. Standard approaches to this problem require numerical solutions for large optimization problems, and become unfeasible for systems involving nonlinear dynamics over the reals. Our approach combines randomized sampling of probabilistic system parameters, SMT-based bounded reachability analysis, and statistical tests. We utilize $\delta-$complete decision procedures to solve reachability analysis in a sound way, i.e., we always decide correctly if, for a given combination of parameters, the system actually reaches the unsafe region. Compared to standard simulation-based analysis methods, our approach supports nondeterministic branching, increases the coverage of simulation, and avoids the zero-crossing problem. We demonstrate that our method is feasible for general hybrid systems with parametric uncertainty by applying the implemented tool **SReach** to various nonlinear hybrid systems with parametric uncertainty.

1 Introduction

Hybrid systems, as models exhibiting both continuous and discrete dynamic behavior, have become a widely used modeling formalism for real-world safety-critical systems, including for example cyber-physical systems, chemical-physical process control, and biomedical systems. The core of hybrid system analysis studies is to construct accurate computational models for real-world systems, and to verify that they meet their design requirements. Simulation-based testing is the most used approach regardless of its incompleteness. Reachability analysis computes instead set over-approximations that cover all system behaviors, which lead to a better input coverage but are naturally harder to scale. It is important to note that reachability analysis can handle models with nondeterminism, and in many cases aim for an infinite time horizon.

We consider the *probabilistic bounded reachability problem*, which is to decide whether a hybrid system with probabilistic system parameters reaches an unsafe region of the state space within a finite number of steps with a probability greater than a given threshold. Although uncertainty raises naturally and the cause for the parametric uncertainty is multifaceted, two factors are critical when building hybrid models for real-world systems. First, probabilistic parameters are needed

when the physics controlling the system is known, but some parameters are either not known precisely, or are expected to vary from individual to individual, or may change by the end of the system operational lifetime. Second, system uncertainty may occur when the model is constructed or learned directly from experimental data. Due to imprecise experimental measurements, the values of system parameters may have ranges of variation with some associated likelihood of occurrence. In both cases we assume that, for probabilistic system parameters, their probability distributions are known, and it is desired to design models which achieve specified performance for these variations.

We describe our tool **SReach** that integrates the existing δ-complete bounded reachability analysis technique [7] with statistical testing, in order to support bounded probabilistic reachability analysis for hybrid systems with parametric uncertainty. We show experiment results of **SReach** for the probabilistic reachability analysis of many realistic hybrid systems that are highly nonlinear and nondeterministic, including a prostate cancer treatment model and a cardiac atrial fibrillation model.

Related Work. To verify the correctness of stochastic hybrid models, there exist several techniques and tools, such as simulation-based verification [1, 19], logic-based verification [14], constraint solving [6], and the probabilistic model checker PRISM [15]. However, it is still difficult to analyze formally stochastic hybrid systems with nonlinear dynamics and complex discrete controls [2, 10]. Theoretically, it is well known that the safety verification problem for hybrid systems with simple dynamics is highly undecidable. Consequently, a unified framework for solving the reachability problem seems impossible, especially for nonlinear hybrid systems. In details, a major difficulty stems from the need of solving logic formulae with nonlinear functions over the reals. Recently, [8, 9] have defined a sound relaxation of this problem and associated decision procedures that always decide correctly if, for a given combination of parameters, the system actually reaches the unsafe region (in the opposite case false positives may be generated, but this can be controlled by a precision parameter $\delta > 0$). The *δ-decision problem* is decidable for bounded first-order formulae over the reals with arbitrary Type 2 computable functions, which include almost all functions used in realistic hybrid systems.

2 Probabilistic bounded reachability

Hybrid automata combine finite automata and dynamical systems [10]. We can extend hybrid automata to allow probabilistic parameters in the following way.

Definition 1. (Hybrid Automata with Parametric Uncertainty) *A hybrid automaton with probabilistic parameters is a tuple $H_p = \langle Q, X, RX, \mathsf{jump}, \mathsf{inv}, \mathsf{init}\rangle$ with the following components. $X = \{x_1, ..., x_n\}$ is a finite set of real variables and $Q = \{q_1, ..., q_m\}$ a finite set of discrete modes. $RX = \{u_1, \cdots, u_k\}$ is a finite set of random variables, where the distribution of u_i is denoted by P_i. The other components are jumps, invariants, and initial condition predicates over $X \cup RX$, as standardly defined for hybrid automata.*

Definition 2. (Probabilistic Bounded Reachability) *Let Ω_{RX} to be the sample space of the probabilistic parameters RX, and $S : RX \to \Omega_{RX}$ to be a sampler. The probabilistic bounded reachability for H_p estimates the maximum probability, over Ω_{RX}, of the bounded reachability for $H_p[s_i/RX]$, where $s_i \in S(RX)$.*

To solve probabilistic bounded reachability, we sample parameters in RX according to their distributions and obtain a hybrid model M_i with no probabilistic parameter. **SReach** then calls **dReach** [7] with the desired precision δ and unfolding steps k. **dReach** returns either unsat or δ-sat for M_i, and this information is then used by statistical tests to decide whether stopping or repeat the procedure. The full procedure is illustrated in Algorithm 1. **SReach** can answer two types of questions: (1) Does MP satisfy ϕ with probability greater than a certain threshold? and (2) What is the probability that MP satisfies ϕ? These two types of questions can be answered by hypothesis testing and statistical estimation methods, respectively. Both methods produce answers up to some correctness precision that can be set arbitrarily by the user. We have implemented in **SReach** a number of hypothesis testing and statistical estimation techniques including: Lai's test [13], Bayes factor test [12], Bayes factor test with indifference region [18], Sequential probability ratio test (SPRT) [17], Chernoff-Hoeffding bound [11], and Bayesian Interval Estimation with Beta prior [19]. (See Appendix A for more details.)

Algorithm 1 SReach

```
1: function SReach(MP, ST, δ, k)
2:     Succ ← 0                                  ▷ number of δ-sat samples
3:     N ← 0                                     ▷ total number of samples
4:     RV ← ExtractRV(MP)                        ▷ get the RVs from the probabilistic model
5:     repeat
6:         S_i ← Sim(RV)                         ▷ sample the parameters
7:         M_i ← Gen(MP, S_i)                    ▷ generate a dReach model
8:         Res ← dReach(M_i, δ, k)               ▷ call dReach to solve k-step δ-reachability
9:         if Res = δ-sat then
10:            Succ ← Succ + 1
11:        end if
12:        N ← N + 1
13:    until ST.done(Succ, N)                    ▷ perform statistical test
14:    return ST.output
15: end function
```

3 Experiments

Our method is implemented in the open-source tool **SReach** (`https://github.com/dreal/SReach`). See Appendix B for its usage. All benchmarks and data shown below are on the tool website. All experiments were conducted on a machine with 2.9GHz Intel Core i7 processor and 8GB RAM, running OS X 10.9.2. In our experiments we used 0.001 as the precision for the δ-decision problem; and Bayesian sequential estimation with 0.01 half-interval width, coverage probability 0.99, and uniform prior ($\alpha = \beta = 1$). The detailed description of the following models in Appendix C demonstrates their highly nonlinear characteristics.

Prostate cancer treatment. We modified the model of the intermittent androgen suppression (IAS) therapy in [16] by adding parametric uncertainty. The IAS therapy switches between treatment-on, and treatment-off with respect to the serum level thresholds of prostate-specific antigen (PSA) - r_0 and r_1. As suggested by the clinical trials [3], an effective IAS therapy highly depends on the individual patient. Thus, we modified the model by taking the parametric variation caused by the personalized differences into account. In details, according to the clinic data from hundreds of patients [4], we replaced 6 system parameters with random variables with appropriate (continuous) distributions, including α_x (proliferation rate of AD cells), α_y (proliferation rate of AI cells), β_x (apoptosis rate of AD cells), β_y (apoptosis rate of AI cells), m_1 (mutation rate from AD to AI cells), and z_0 (normal androgen level).

Model	#RVs	r_0	r_1	Est_P	#S_S	#T_S	Avg_T(s)	Tot_T(s)
PCT1	6	5.0	10.0	0.04	0	227	0.145	32.915
PCT2	6	7.0	11.0	0.591	2144	3628	432.491	1569077.348
PCT3	6	10.0	15.0	0.996	227	227	692.861	157279.446

Table 1: #RVs = number of random variables in the model, #S_S = number of δ-sat samples, #T_S = total number of samples, r_0 = lower threshold of the serum PSA level, r_1 = upper threshold, Est_P = estimated probability of the property, Avg_T(s) = average CPU time of each sample in seconds, and Tot_T(s) = total CPU time for all samples in seconds.

To describe the variations due to individual difference, we assigned α_x to be $U(0.0193, 0.0214)$, α_y to be $U(0.0230, 0.0254)$, β_x to be $U(0.0072, 0.0079)$, β_y to be $U(0.0160, 0.0176)$, m_1 to be $U(0.0000475, 0.0000525)$, and z_0 to be $N(30.0, 0.001)$. We used **SReach** to estimate the probabilities of the model preventing the relapse of the prostate cancer with three distinct pairs of treatment thresholds (*i.e.*, combinations of r_0 and r_1). In the experiments, we chose 2 as the unfolding steps. For each sample generated, **SReach** dealt with 41 variables, and 10 ODEs. As shown in Table 1, the model with thresholds $r_0 = 10$, and $r_1 = 15$ has the probability approaching to 1, indicating that these thresholds may be considered for the general treatment.

Atrial Fibrillation. The minimum resistor model (MRM) reproduces experimentally measured characteristics of human ventricular cell dynamics [5]. The MRM reduces the complexity of existing models by representing channel gates of different ions with one fast channel, and two slow gates. However, due to this reduction, for most model parameters, it becomes impossible to obtain their values through measurements. With this application, we will show that **SReach** can also be adopted to identify appropriate ranges and distributions for model parameters, *i.e.*, parameter estimation.

To illustrate the way that **SReach** is used to conduct parameter estimation, we chose two system parameters - EPI_TO1, and EPI_TO2, and varied their distributions to see with which distributions for these two system parameters, the model can present the desired pattern. The model has 4 modes. In the experiments, we chose 3 as the unfolding steps. For each sample generated, **SReach** dealt with 62 variables, and 24 ODEs. As in the Table 2, when EPI_TO1 is either close to 400, or between 0.0061 and 0.007, and EPI_TO2 is close to 6, the

model can satisfy the given bounded reachability property with a probability very close to 1.

Model	#RVs	EPI_TO1	EPI_TO2	#S_S	#T_S	Est_P	A_T(s)	T_T(s)
Cd_to1_s	1	U(6.1e-3, 7e-3)	6	227	227	0.996	0.362	82.174
Cd_to1_uns	1	U(5.5e-3, 5.9e-3)	6	0	227	0.004	0.124	28.148
Cd_to2_s	1	400	U(0.131, 6)	227	227	0.996	0.361	81.947
Cd_to2_uns	1	400	U(0.1, 0.129)	0	227	0.004	0.139	31.552
Cd_to12_s	2	N(400, 1e-4)	N(6, 1e-4)	227	227	0.996	0.373	84.671
Cd_to12_uns	2	N(5.5e-3, 10e-6)	N(0.11, 10e-5)	0	227	0.004	0.131	29.737

Table 2: #RVs = number of random variables in the model, #S_S = number of δ-sat samples, #T_S = total number of samples, Est_P = estimated probability of property, A_T(s) = average CPU time of each sample in seconds, and T_T(s) = total CPU time for all samples in seconds.

Additional benchmarks. To further demonstrate the feasibility of **SReach**, we also applied it to the following benchmarks. Table 3 shows the results of experiments. BB refers to the bouncing ball models, Tld the thermostat model with linear temperature decrease, Ted the thermostat model with exponential decrease, DT the dual thermostat models, W the watertank models, DW the dual watertank models, Que the model for queuing system which has both nonlinear functions and nondeterministic jumps, 3dOsc the model for 3d oscillator, and QuadC the model for quadcopter stabilization control.

Benchmark	#Ms	K	#ODEs	#Vs	#RVs	δ	Est_P	#S_S	#T_S	A_T(s)	T_T(s)
BBK1	1	1	2	14	3	0.001	0.754	5372	7126	0.086	612.836
BBK5	1	5	2	38	3	0.001	0.059	209	3628	0.253	917.884
BBwDv1	2	2	4	20	4	0.001	0.208	2206	10919	0.080	873.522
BBwDv2K2	2	2	4	20	3	0.001	0.845	7330	8669	0.209	1811.821
BBwDv2K8	2	8	4	56	3	0.001	0.207	2259	10901	0.858	9353.058
Tld	2	7	2	33	4	0.001	0.996	227	227	0.213	48.351
Ted	2	7	4	50	4	0.001	0.996	227	227	12.839	2914.448
DTldK3	2	3	4	26	2	0.001	0.996	227	227	0.382	86.714
DTldK5	2	5	4	38	2	0.001	0.161	1442	8961	0.280	2509.078
W4mv1	4	3	8	26	6	0.001	0.381	5953	15639	0.238	3722.082
W4mv2K3	4	3	8	26	6	0.001	0.996	227	227	0.673	152.771
W4mv2K7	4	7	8	50	6	0.001	0.004	0	227	0.120	27.240
DWK1	2	1	4	14	5	0.001	0.996	227	227	0.171	38.817
DWK3	2	3	4	26	5	0.001	0.996	227	227	0.215	48.806
DWK9	2	9	4	62	5	0.001	0.996	227	227	5.144	1167.688
Que	3	2	3	13	4	0.001	0.228	2662	11677	0.095	1109.315
3dOsc	3	2	18	48	2	0.001	0.996	227	227	8.273	1877.969
QuadC	1	0	14	44	6	0.001	0.996	227	227	825.641	187420.507

Table 3: #Ms = number of modes, K indicates the unfolding steps, #ODEs = number of ODEs in the model, #Vs = number of total variables in the unfolded formulae, #RVs = number of random variables in the model, δ = precision used in **dReach**, #S_S = number of δ-sat samples , #T_S = total number of samples, Est_P = estimated probability of the property, A_T(s) = average CPU time of each sample in seconds, and T_T(s) = total CPU time for all samples in seconds.

4 Conclusions and future work

We presented a probabilistic reachability analysis tool. The tool combines δ-decision procedures [7–9] and statistical testing techniques. It supports bounded

reachability analysis and parameter estimation for hybrid systems with parametric uncertainty. This tool was used for the reachability analysis of two representative examples - a prostate cancer treatment control and a cardiac model - which are currently out of the reach of other formal (SMT-based) tools. In the near future, we plan to extend support for more general stochastic hybrid models that include probabilistic jumps with discrete or continuous distributions. Also, we plan to develop a parallel version of our tool.

References

1. A. Abate. Probabilistic reachability for stochastic hybrid systems: Theory, computations, and applications. Technical Report UCB/EECS-2007-132, University of California, Berkeley, 2007.
2. R. Alur. Formal verification of hybrid systems. In *EMSOFT*, pages 273–278, 2011.
3. N. Bruchovsky et al. Final results of the Canadian prospective phase ii trial of intermittent androgen suppression for men in biochemical recurrence after radiotherapy for locally advanced prostate cancer. *Cancer*, 107(2):389–395, 2006.
4. N. Bruchovsky, L. Klotz, J. Crook, and S. L. Goldenberg. Locally advanced prostate cancerbiochemical results from a prospective phase ii study of intermittent androgen suppression for men with evidence of prostate-specific antigen recurrence after radiotherapy. *Cancer*, 109(5):858–867, 2007.
5. A. Bueno-Orovio, E. M. Cherry, and F. H. Fenton. Minimal model for human ventricular action potentials in tissue. *J. of Theor. Biology*, 253(3):544–560, 2008.
6. M. Fränzle, H. Hermanns, and T. Teige. Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems. In *Hybrid Systems: Computation and Control*, pages 172–186. Springer, 2008.
7. S. Gao, S. Kong, W. Chen, and E. M. Clarke. δ-complete analysis for bounded reachability of hybrid systems. Technical report, Carnegie Mellon University, 2014.
8. S. Gao, S. Kong, and E. M. Clarke. dReal: An SMT solver for nonlinear theories over the reals. In *Automated Deduction–CADE-24*, pages 208–214. Springer, 2013.
9. S. Gao, S. Kong, and E. M. Clarke. Satisfiability modulo ODEs. In *FMCAD*, pages 105–112, 2013.
10. T. A. Henzinger. *The theory of hybrid automata*. Springer, 2000.
11. W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
12. R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.
13. T. L. Lai. Nearly optimal sequential tests of composite hypotheses. *The Annals of Statistics*, pages 856–886, 1988.
14. A. Platzer. Stochastic differential dynamic logic for stochastic hybrid programs. In *Automated Deduction–CADE-23*, pages 446–460. Springer, 2011.
15. PRISM. `https://www.prismmodelchecker.org`.
16. G. Tanaka, Y. Hirata, S. L. Goldenberg, N. Bruchovsky, and K. Aihara. Mathematical modelling of prostate cancer growth and its application to hormone therapy. *Phil. Tran. Roy. Soc. A: Math., Phys. and Eng. Sci.*, 368(1930):5029–5044, 2010.
17. A. Wald. Sequential tests of statistical hypotheses. *Annals of Mathematical Statistics*, 16(2):117–186, 1945.
18. H. L. Younes. Verification and planning for stochastic processes with asynchronous events. Technical report, DTIC Document, 2005.
19. P. Zuliani, A. Platzer, and E. M. Clarke. Bayesian statistical model checking with application to Simulink/Stateflow verification. In *HSCC*, pages 243–252, 2010.

Appendix

A Statistical tests

In this section we briefly describe the statistical techniques implemented in **SReach**. To deal with qualitative questions, **SReach** supports the following hypothesis testing methods.

Lai's test [13]. As a simple class of sequential tests, it tests the one-sided composite hypotheses $H_0 : \theta \leq \theta_0$ versus $H_1 : \theta \geq \theta_1$ for the natural parameter θ of an exponential family of distributions under the $0-1$ loss and cost c per observation. [13] shows that these tests have nearly optimal frequentist properties and also provide approximate Bayes solutions with respect to a large class of priors.

Bayes factor test [12]. The use of Bayes factors is a Bayesian alternative to classical hypothesis testing. It is based on the Bayes theorem. Hypothesis testing with Bayes factors is more robust than frequentist hypothesis testing, as the Bayesian form avoids model selection bias, evaluates evidence in favor the null hypothesis, includes model uncertainty, and allows non-nested models to be compared. Also, frequentist significance tests become biased in favor of rejecting the null hypothesis with sufficiently large sample size.

Bayes factor test with indifference region. A hypothesis test has ideal performance if the probability of the Type-I error (respectively, Type-II error) is exactly α (respectively, β). However, these requirements make it impossible to ensure a low probability for both types of errors simultaneously (see [18] for details). A solution is to use an indifference region. The indifference region indicates the distance between two hypotheses, which is set to separate the two hypotheses.

Sequential probability ratio test (SPRT) [17]. The SPRT considers a simple hypothesis $H_0 : \theta = \theta_0$ against a simple alternative $H_1 : \theta = \theta_1$. With the critical region Λ_n and two thresholds A, and B, SPRT decides that H_0 is true and stops when $\Lambda_n < A$. It decides that H_1 is true and terminates if $\Lambda_n > B$. If $A < \Lambda_n < B$, it will collect another observation to obtain a new critical region Λ_{n+1}. The SPRT is optimal, among all sequential tests, in the sense that it minimizes the average sample size.

To offer quantitative answers, **SReach** also supports estimation procedures as below.

Chernoff-Hoeffding bound [11]. To estimate the mean p of a (bounded) random variable, given a precision δ' and coverage probability α, the Chernoff-Hoeffding bound computes a value p' such that $|p' - p| \leq \delta'$ with probability at least α.

Bayesian Interval Estimation with Beta prior [19]. This method estimates p, the unknown probability that a random sampled model satisfies a specified reachability property. The estimate will be in the form of a confidence interval, containing p with an arbitrary high probability. [19] assumes that the unknown p is given by a random variable, whose density is called the prior density, and focuses on Beta priors.

B The SReach tool

B.1 Input format

The inputs to our **SReach** tool are descriptions of hybrid automata with random variables (representing the probabilistic system parameters), and the reachability property to be checked. Following roughly the same format as the above definition of hybrid automata, and adding the declarations of random variables, the description of an automaton is as follows.

Preprocessor. We can use the C language syntax to define constants and macros. It can also be used for defining random variables — see below.

Variable declaration. For a random variable, the declaration specifies its distribution and name. For the variables which are not random variables, they are required to be declared within bounds.

Hybrid automaton. A hybrid automaton is represented by a set of modes. Within each mode declaration, users can specify statements for mode invariant(s), flow function(s), and jump condition(s). For a mode invariant, we can give any logic formula of the variables. For a flow function, it is expressed by an ODE. As for a jump condition, it is written as

```
<logic_formula1>  ==> @<tagert_mode>  <logic_formula2>,
```

where the first logic formula is given as the guard of the jump, and the second one specifies the reset condition after the jump.

Initial conditions and Goals. Following the declaration of modes, we can declare one initial mode with corresponding conditions, and the reachability properties in the end.

Example 3.1. The following is an example input file. Currently, users can specify random variables with Bernoulli distribution, Uniform distribution, Gaussian distribution, and Exponential distribution. (Note: it is easy to include additional distributions if needed.)

```
1  #define pi 3.1416
2  N(1,0.1) mu1;
3  U(10,15) thro;
4  E(0.49) theta1;
5  B(0.75) xinit;
6  [0,5] x;
7  [0,3] time;
8  { mode 1;
9    invt:
10          (x<=1.5);
11          (x>=0);
12   flow:
13          d/dt[x]=thro*(1/(theta1*sqrt(2*pi)))
14                   *exp(0-((x-mu1)^2)/(2*theta1^2));
15   jump:
16         (x>=(thre1+5))==>@2(x'=x);
```

```
17 }
18 init:
19 @1       (x=xinit);
20 goal:
21 @4       (x>=50);
```

B.2 Command line

After building, **SReach** can be simply used through:

```
SReach <statistical_testing_option> <filename> <dReach> <k> <delta>
```

where:

- `statistical_testing_option` is a text file containing a sequence of test specifications. We will introduce the usages of statistical testing options in the following part;
- `filename` is a .pdrh file describing the model of a hybrid system with probabilistic system parameters. It is of the input format described in last subsection;
- `dReach` is a bounded reachability analyzing tool for hybrid systems based on dReal;
- `k` is the number of steps of the model that the tool will explore; and
- `delta` is the precision for the δ-decision problem.

B.3 Statistical testing options

SReach can be used with different statistical testing methods through the following specifications.

Lai's test: `Lai <theta> <cost_per_sample>`, where `theta` indicates the probability threshold.

Bayes factor test: `BFT <theta> <T> <alpha> <beta>`, where `theta` is a probability threshold satisfying `0 < theta < 1`, `T` is a ratio threshold satisfying `T > 1`, and `alpha`, and `beta` are beta prior parameters.

BFT with indifference region: `BFTI <theta> <T> <alpha> <beta> <delta>`, where, besides the parameters used in the above Bayes factor test, `delta` is given to create the indifference region - $[p_0, p_1]$, where p_0 = `theta` + `delta` and p_1 = `theta - delta`. Now, it tests $H_0 : p \geq p_0$ against $H_1 : p \leq p_1$.

Sequential probability ratio test (SPRT): `SPRT <theta> <T> <delta>`.

Chernoff-Hoeffding bound: `CHB <delta1> <coverage_probability>`, where `delta1` is the given precision, and `coverage_probability` indicates the confidence.

Bayesian Interval Estimation with Beta prior:
`BEST <delta1> <coverage_probability> <alpha> <beta>`.

C Model description

Atrial Fibrillation. The model has four discrete control locations, four state variables, and nonlinear ODEs. A typical set of ODEs in the model is:

$$\begin{aligned}
\frac{du}{dt} &= e + (u - \theta_v)(u_u - u)vg_{fi} + wsg_{si} - g_{so}(u) \\
\frac{ds}{dt} &= \frac{g_{s2}}{(1 + \exp(-2k(u - us)))} - g_{s2}s \\
\frac{dv}{dt} &= -g_v^+ \cdot v \qquad \frac{dw}{dt} = -g_w^+ \cdot w
\end{aligned}$$

The exponential term on the right-hand side of the ODE is the sigmoid function, which often appears in modelling biological switches.

Prostate Cancer Treatment. The Prostate Cancer Treatment model exhibits more nonlinear ODEs. The reachability questions are

$$\begin{aligned}
\frac{dx}{dt} &= (\alpha_x(k_1 + (1 - k_1)\frac{z}{z + k_2} - \beta_x((1 - k_3)\frac{z}{z + k_4} + k_3)) - m_1(1 - \frac{z}{z_0}))x + c_1x \\
\frac{dy}{dt} &= m_1(1 - \frac{z}{z_0})x + (\alpha_y(1 - d\frac{z}{z_0}) - \beta_y)y + c_2y \\
\frac{dz}{dt} &= \frac{-z}{\tau} + c_3z \\
\frac{dv}{dt} &= (\alpha_x(k_1 + (1 - k_1)\frac{z}{z + k_2} - \beta_x(k_3 + (1 - k_3)\frac{z}{z + k_4})) \\
&\quad -m_1(1 - \frac{z}{z_0}))x + c_1x + m_1(1 - \frac{z}{z_0})x + (\alpha_y(1 - d\frac{z}{z_0}) - \beta_y)y + c_2y
\end{aligned}$$

Electronic Oscillator. The 3dOsc model represents an electronic oscillator model that contains nonlinear ODEs such as the following:

$$\begin{aligned}
\frac{dx}{dt} &= -ax \cdot sin(\omega_1 \cdot \tau) \\
\frac{dy}{dt} &= -ay \cdot sin((\omega_1 + c_1) \cdot \tau) \cdot sin(\omega_2) \cdot 2 \\
\frac{dz}{dt} &= -az \cdot sin((\omega_2 + c_2) \cdot \tau) \cdot cos(\omega_1) \cdot 2 \\
\frac{\omega_1}{dt} &= -c_3 \cdot \omega_1 \quad \frac{\omega_2}{dt} = -c_4 \cdot \omega_2 \quad \frac{d\tau}{dt} = 1
\end{aligned}$$

Quadcopter Control. We developed a model that contains the full dynamics of a quadcopter. We use the model to solve control problems by answering reach-

ability questions. A typical set of the differential equations are the following:

$$\frac{\mathrm{d}\omega_x}{\mathrm{d}t} = L \cdot k \cdot (\omega_1^2 - \omega_3^2)(1/I_{xx}) - (I_{yy} - I_{zz})\omega_y\omega_z/I_{xx}$$
$$\frac{\mathrm{d}\omega_y}{\mathrm{d}t} = L \cdot k \cdot (\omega_2^2 - \omega_4^2)(1/I_{yy}) - (I_{zz} - I_{xx})\omega_x\omega_z/I_{yy}$$
$$\frac{\mathrm{d}\omega_z}{\mathrm{d}t} = b \cdot (\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)(1/I_{zz}) - (I_{xx} - I_{yy})\omega_x\omega_y/I_{zz}$$
$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = \omega_x + \frac{\sin(\phi)\sin(\theta)}{\left(\frac{\sin(\phi)^2\cos(\theta)}{\cos(\phi)} + \cos(\phi)\cos(\theta)\right)\cos(\phi)}\omega_y + \frac{\sin(\theta)}{\frac{\sin(\phi)^2\cos(\theta)}{\cos(\phi)} + \cos(\phi)\cos(\theta)}\omega_z$$
$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = -\left(\frac{\sin(\phi)^2\cos(\theta)}{\left(\frac{\sin(\phi)^2\cos(\theta)}{\cos(\phi)}\omega_y + \cos(\phi)\cos(\theta)\right)\cos(\phi)^2} + \frac{1}{\cos(\phi)}\right)\omega_y - \frac{\sin(\phi)\cos(\theta)}{\left(\frac{\sin(\phi)^2\cos(\theta)}{\cos(\phi)} + \cos(\phi)\cos(\theta)\right)\cos(\phi)}\omega_z$$
$$\frac{\mathrm{d}\psi}{\mathrm{d}t} = \frac{\sin(\phi)}{\left(\frac{\sin(\phi)^2\cos(\theta)}{\cos(\phi)} + \cos(\phi)\cos(\theta)\right)\cos(\phi)}\omega_y + \frac{1}{\frac{\sin(\phi)^2\cos(\theta)}{\cos(\phi)} + \cos(\phi)\cos(\theta)}\omega_z$$
$$\frac{\mathrm{d}xp}{\mathrm{d}t} = (1/m)(\sin(\theta)\sin(\psi)k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) - k \cdot d \cdot xp)$$
$$\frac{\mathrm{d}yp}{\mathrm{d}t} = (1/m)(-\cos(\psi)\sin(\theta)k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) - k \cdot d \cdot yp)$$
$$\frac{\mathrm{d}zp}{\mathrm{d}t} = (1/m)(-g - \cos(\theta)k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2) - k \cdot d \cdot zp$$
$$\frac{\mathrm{d}x}{\mathrm{d}t} = xp, \frac{\mathrm{d}y}{\mathrm{d}t} = yp, \frac{\mathrm{d}z}{\mathrm{d}t} = zp$$