DynBEX: a Differential Constraint Library for Studying Dynamical Systems

Julien Alexandre Dit Sandretto, Alexandre Chapoutot

DynIBEX: a Differential Constraint Library for Studying Dynamical Systems

Julien Alexandre dit Sandretto
U2IS, ENSTA ParisTech, Université Paris-Saclay,
828 bd des Maréchaux,
91762 Palaiseau Cedex, France
alexandre@ensta.fr

Alexandre Chapoutot
U2IS, ENSTA ParisTech, Université Paris-Saclay,
828 bd des Maréchaux,
91762 Palaiseau Cedex, France
chapoutot@ensta.fr

ABSTRACT

Interval analysis tools have been used in many techniques and methods to study dynamical systems. Nevertheless, custom homemade libraries are usually developed for these purposes. DynIBEX is a free open-source library combining validated numerical integration methods with a constraint programming library named IBEX. Its purpose is to provide all the basic interval operators which may be used into more complex algorithms for studying dynamical systems.

CCS Concepts

•Mathematics of computing → **Solvers; Interval arithmetic;** **•Theory of computation** → *Timed and hybrid models;*

Keywords

Interval constraints; Validated numerical integration; Runge-Kutta methods

1. INTRODUCTION

An important amount of research deals with cyber-physical systems to prove safety or study stability properties. One of the mathematical foundation of cyber-physical systems are (hybrid) dynamical systems which are complex objects to study, due to their non trivial temporal evolutions.

Studying dynamical systems to prove safety or stability properties refers to consider the set of all the trajectories. Interval analysis is an appealing approach because it is tractable for high dimensional systems and combining with space decomposition it is possible to represent complex geometric forms. Interval analysis tools are found in many work such as for verification methods based on simulation or on reachable set computation, [9, 5, 4, 3], or in verification techniques based on SMT solver, mainly to define decision procedures over the reals, as in [8] or in *SMT modulo ODE* [6]. Control synthesis problem have also been considered in [7, 1].

HSCC '16 Vienna, Austria
© 2016 ACM. ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

In all the previous cited works a common set of interval operators are used which could be gathered into a library. DynIBEX[1] is such a library providing operator to deal with interval constraints (IBEX library[2]) and differential equations. Note that DynIBEX follows the same purpose than CORA [2] with an emphasis on the interval constraint programming approach. We report the main features of DynIBEX in this poster.

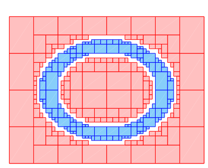
2. FEATURES OF DYNIBEX

2.1 Features coming from IBEX library

The most basic notion used in interval analysis is the *inclusion function* $[f] : \mathbb{IR}^n \to \mathbb{IR}^k$ for a function $f : \mathbb{R}^n \to \mathbb{R}^k$ which satisfies for all $[\mathbf{x}] \in \mathbb{IR}^n$, $f([\mathbf{x}]) = \{f(\mathbf{x}) \mid \mathbf{x} \in [\mathbf{x}]\} \subseteq [f]([\mathbf{x}])$. $\mathbb{R}$ stands for the set of real values while $\mathbb{IR}$ stands for the set of interval values. The *natural* inclusion function is the simplest to obtain: all occurrences of the real variables are replaced by their interval counterpart and all arithmetic operations are evaluated using interval arithmetic [10]. To avoid some limitations with interval arithmetic, other kinds of arithmetic may bu used such as affine arithmetic [3]. Both arithmetic are available in IBEX.

Consider some function $g : \mathbb{R}^n \to \mathbb{R}^k$ and some set $\mathcal{Z} \subset \mathbb{R}^k$. A *contractor* $\mathcal{C}_c : \mathbb{IR}^n \to \mathbb{IR}^n$ associated to the generic constraint $c \equiv g(\mathbf{x}) \in \mathcal{Z}$ is a function taking a box $[\mathbf{x}]$ as input and returning a box $\mathcal{C}_c([\mathbf{x}])$ satisfying *i)* $\mathcal{C}_c([\mathbf{x}]) \subseteq [\mathbf{x}]$ and *ii)* $g([\mathbf{x}]) \cap \mathcal{Z} = g(\mathcal{C}_c([\mathbf{x}])) \cap \mathcal{Z}$. Hence, $\mathcal{C}_c$ provides a box containing the set $\{\mathbf{x} \in [\mathbf{x}] \mid g(\mathbf{x}) \in \mathcal{Z}\}$ of solutions of c included in $[\mathbf{x}]$: *i)* ensures that the returned box is included in $[\mathbf{x}]$ and *ii)* ensures that no solution of $g(\mathbf{x}) \in \mathcal{Z}$ in $[\mathbf{x}]$ is lost. Standard contractor operators are defined in IBEX.

Paving is a common method in interval analysis, it is related to the construction of a set $\mathcal{S} \subset \mathbb{R}^n$ as a list of non-overlapping boxes $[\mathbf{x_i}] \subset \mathcal{S}$ with a non null width. This approach can be used to describe a set, by a list of inner boxes ($[\mathbf{x_i}] \subset \mathcal{S}$), of outer boxes ($[\mathbf{x_i}] \not\subset \mathcal{S}$), and the frontier, *i.e.*, a list of boxes for which we cannot conclude of the membership to $\mathcal{S}$ in an acceptable computation time. For example, if the set $\mathcal{S}$ describes a ring such as $\mathcal{S} = \{(x, y) \mid x^2 + y^2 \in [1, 2]\}$, the paving of $\mathcal{S}$ in the box $[-2, 2] \times [-2, 2]$ is given on the left.



[1]http://perso.ensta-paristech.fr/~chapoutot/dynibex/
[2]http://www.ibex-lib.org

2.2 Validated numerical integration methods

In DynIBEX different classes of differential equations can be treated with interval analysis tools: *Ordinary Differential Equations* (ODE) defining a function $\mathbf{y}(t)$ from $f : \mathbb{R} \times \mathbb{R}^n \times \to \mathbb{R}^n$ such that it is solution of $\dot{\mathbf{y}} = f(t, \mathbf{y})$, where $\dot{\mathbf{y}}$ is the time derivative of $\mathbf{y}(t)$, and *Differential Algebraic Equations* (DAE) of index 1 defined two functions $\mathbf{y}(t)$ and $\mathbf{x}(t)$ from $f : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ and $g : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$ such that they are solutions of $\dot{\mathbf{y}} = f(t, \mathbf{x}, \mathbf{y}) \wedge 0 = g(t, \mathbf{x}, \mathbf{y})$ for all t. Note that $\mathbf{x}(t)$ is implicitly defined by g.

DynIBEX can solve these kinds of equations, with interval initial values and/or interval parameters, using various validated numerical integration schemes based on explicit and implicit Runge-Kutta methods among Heun and Midpoint methods, Radau-IIA (order 3), classical Runge-Kutta, Lobatto-IIIA (order 4), Lobatto-IIIC (order 4).

The combination of validated numerical integration methods for ODEs and DAEs with features of IBEX offers additional capabilities as considering embedded constraints to ODEs and DAEs. Such constraints may represent some physics laws such as energy preservation along trajectories. Basically, DynIBEX can consider constraints of the form $c \equiv \forall t, 0 = h(t, \mathbf{y}, \mathbf{p})$ with $\mathbf{y}$ the solution of an ODE or a DAE and $\mathbf{p}$ stands for a vector of bounded uncertain parameters.

For example, DynIBEX can solve the following DAE while checking the consistency of the initial values of the state and algebraic variables. It produces at time $t = 0.5$ the enclosure $\mathbf{y} \in ([1.8732, 1.8733], [1.3651, 1.3653], [5.0552, 5.0558])$.

```
int main(){
  Variable y(3); // State variables
  IntervalVector yinit(3);
  yinit[0]= Interval(1.0);
  yinit[1]= 1.0;
  yinit[2]= 3.0;

  Variable x(2); // Algebraic variables
  IntervalVector xinit(2);
  xinit[0]= Interval(0.5);
  xinit[1]= 1.0;

  Function ydot =
       Function(y, x, Return(y[1]+x[0],
                             y[0]-y[1]*x[0],
                             y[0]*y[2]-x[1]));

  Function g = Function(y, x, Return(y[0]-x[1],
                             y[1]-2*x[0]));

  ivp_dae_h1 problem =
      ivp_dae_h1(ydot, g, 0.0, yinit, xinit);

  simulation simu = simulation(&problem, 0.5,
                             RADAU3_DAE, 1e-14);
  simu.run_simulation();

  IntervalVector safe(3);
  safe[0] = Interval(0,9);
  safe[1] = Interval(0,5);
  safe[2] = Interval(0,3);
  bool flag = simu.stayed_in(safe);
  cout << "simu stayed in safe: ";
  cout << flag ? "true" : "false" << endl;

  return 0;
}
```

2.3 Satisfaction problems

Additionally in DynIBEX, a set of operators have been defined to handle constraints of the form $\forall t, \mathbf{y}(t) \in \mathcal{S}$ or $\exists t, \mathbf{y}(t) \cap \mathcal{U} \neq \emptyset$. In the above example, the method `stayed_in` is used to check if the trajectory $\mathbf{y}(t)$ stay in a given box for all t. This feature, added to the contractors and paving capabilities, allow to solve satisfaction problems based on differential equations.

3. CONCLUSION

A quick overview of DynIBEX library has been presented, showing the main features to study dynamical systems. Note also that it is straightforward to simulate or study switched systems such as [7]. Further work has still to be done to take into account more differential problems such as *delay differential equations*.

4. ACKNOWLEDGMENTS

This research benefited from the support of the "Chair Complex Systems Engineering - École polytechnique, THALES, DGA, FX, DASSAULT AVIATION, DCNS Research, ENSTA ParisTech, Télécom ParisTech, Fondation ParisTech, FDO ENSTA"

5. REFERENCES

[1] J. Alexandre dit Sandretto, A. Chapoutot, and O. Mullier. Tuning PI controller in non-linear uncertain closed-loop systems with interval analysis. In *Workshop on Synthesis of Complex Parameters*, volume 44 of *OASIcs*, pages 91–102. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

[2] M. Althoff. An introduction to CORA 2015. In *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015.

[3] O. Bouissou, A. Chapoutot, and S. Mimram. Simulation and verification of hybrid systems using hyson. In *First International Workshop on Applied Verification for Continuous and Hybrid Systems*, 2014.

[4] O. Bouissou, A. Chapoutot, S. Mimram, and B. Strazzulla. Set-based simulation for design and verification of simulink models. In *Embedded Real Time Software and Systems*, 2014.

[5] X. Chen, E. Abrahám, and S. Sankaranarayanan. Taylor model flowpipe construction for non-linear hybrid systems. In *Real-Time Systems Symposium*, pages 183–192. IEEE, 2012.

[6] A. Eggers, N. Ramdani, N. Nedialkov, and M. Fränzle. Improving the SAT modulo ODE approach to hybrid systems analysis by combining different enclosure methods. *Software and Systems Modeling*, 2012.

[7] L. Fribourg and R. Soulat. Stability controllers for sampled switched systems. In *Reachability Problems in Computational Models*, volume 8169 of *LNCS*, pages 135–145. Springer, 2013.

[8] S. Gao, S. Kong, and E. M. Clarke. dReal: An SMT solver for nonlinear theories over the reals. In *Conference on Automated Deduction*, volume 7898 of *LNCS*, pages 208–214. Springer, 2013.

[9] T. A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi. Beyond HYTECH: Hybrid systems analysis using interval numerical methods. In *Hybrid Systems: Computation and Control*, volume 1790 of *LNCS*, pages 130–144. Springer, 2000.

[10] L. Jaulin, M. Kieffer, O. Didrit, and É. Walter. *Applied Interval Analysis*. Springer, 2001.