`TaylorModels.jl`: Taylor models in Julia and their application to validated solutions of ODEs

Luis Benet*[1], Marcelo Forets[2], David P. Sanders[3], and Christian Schilling[4]

[1]Instituto de Ciencias Físicas, Universidad Nacional Autónoma de México, México
[2]Universidad de la República, Uruguay
[3]Facultad de Ciencias, Universidad Nacional Autónoma de México, México
[4]IST Austria, Klosterneuburg, Austria

Keywords: Taylor models; validated solutions of ODEs; verified numerics

Taylor models, introduced by Berz and Makino [3, 6, 7], define a tool that allows to rigorously bound functions or compute validated solutions of ODEs, among other applications. A Taylor model $M(f) = (p_n, \Delta)$ of an $(n+1)$-continuously differentiable function $f(x)$, $x \in D \subset \mathbb{R}^d$ (defined over an open set containing the domain D of interest), is defined by the n-th order Taylor approximation $p_n(x)$ of $f(x)$ around the point $x_0 \in D$, and an interval $[\Delta]$, such that $f(x) \in p_n(x) + [\Delta]$ for all $x \in D$. In its original form, the coefficients of $p(x)$ are floating-point numbers.

This definition was recently extended by M. Joldes [5], where Taylor models with absolute remainder are defined as above (using interval coefficients for the Taylor polynomial as well as the expansion point), and Taylor models with relative remainder are defined by $f(x) \in [p_n](x) + [\Delta]x^{n+1}$ for all $x \in D$.

Here we present `TaylorModels.jl` [1], a package written in Julia for the rigorous approximation of functions in one and several variables. The package implements both Taylor models with absolute remainder for one and several variables and Taylor models with relative remainder for univariate functions. The polynomial coefficients may be floating-point numbers or intervals, and allow to perform computations using extended precision formats. Figure 1 displays an example from Ref. [6] of a univariate function bounded by Taylor models with absolute or relative remainder. We shall describe examples of its use as well as its application to obtain validated solutions of ODEs. This work is built on other packages developed by us for interval arithmetic [8], Taylor series [2], and set-based

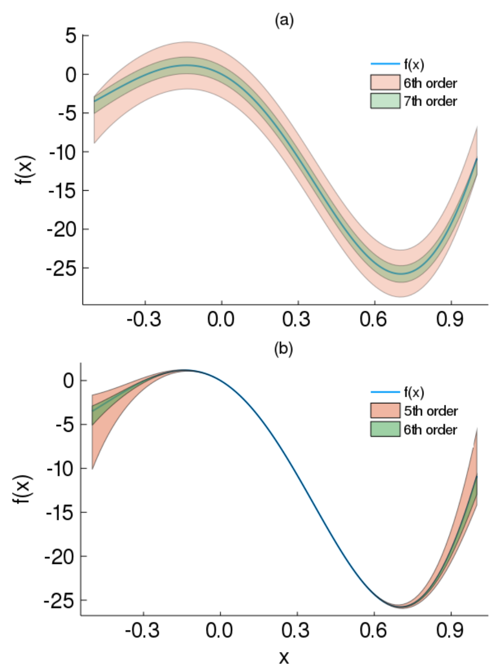

Figure 1: Examples of rigorous bounds for $f(x) = x(x-1.1)(x+2)(x+2.2)(x+2.5)(x+3)\sin(1.7x+0.5)$ in $D = \{x \mid -0.5 \le x \le 1.0\}$ using (a) absolute-remainder Taylor models of order 6 and 7, and (b) relative-remainder Taylor models of order 5 and 6.

*Corresponding author.

reachability [4].

Acknowledgment

LB acknowledges financial support from from the PAPIIT-UNAM project IG-IG100819. CS was supported in part by the Austrian Science Fund (FWF) under grants S11402-N23 (RiSE/SHiNE) and Z211-N23 (Wittgenstein Award) and the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 754411.

References

[1] L. Benet and D. P. Sanders. TaylorModels.jl: Rigorous function approximation with Taylor models in Julia. `https://github.com/JuliaIntervals/TaylorModels.jl`, 2019.

[2] L. Benet and D. P. Sanders. TaylorSeries.jl: Taylor expansions in one and several variables in Julia. *Journal of Open Source Software*, 4(36):1043, 2019.

[3] M. Berz. Differential algebraic description of beam dynamics to very high orders. *Part. Accel.*, 24:109–124, 1989.

[4] S. Bogomolov, M. Forets, G. Frehse, K. Potomkin, and C. Schilling. JuliaReach: A toolbox for set-based reachability. In *HSCC*, pages 39–44, New York, NY, USA, 2019. ACM.

[5] M. M. Joldes. *Rigorous Polynomial Approximations and Applications*. PhD Thesis, École Normale Supérieure de Lyon (ENS Lyon), 2011.

[6] K. Makino. *Rigorous analysis of nonlinear motion in particle accelerators*. PhD Thesis, Michigan State University, 1998.

[7] K. Makino and M. Berz. Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics*, 6(3):239–312, 2003.

[8] D. P. Sanders and L. Benet. IntervalArithmetic.jl: Rigorous floating-point calculations using interval arithmetic in Julia. `https://github.com/JuliaIntervals/IntervalArithmetic.jl`, 2019.