EPiC Series in Computing

Volume 61, 2019, Pages 211–219

ARCH19. 6th International Workshop on Applied Verification of Continuous and Hybrid Systems

Under the Hood of a Stand-Alone Lagrangian Reachability Tool

S. Gruenbacher[1], J. Cyranka[2], M.A. Islam[3], M. Tschaikowski[1], S.A. Smolka[4], and R. Grosu[1]

[1] Technische Universität Wien, Vienna, Austria
{sophie.gruenbacher,max.tschaikowski,radu.grosu}@tuwien.ac.at
[2] University California at San Diego, San Diego, CA, USA
JCyranka@gmail.com
[3] Texas Tech University, Lubbock, TX, USA
ariful.islam@ttu.edu
[4] Stony Brook University, Stony Brook, NY, USA
{sas@cs.stonybrook.edu}

Abstract

Tool presentation: We present work in progress on a stand-alone implementation of Lagrangian reachability, a recently introduced over-approximation technique for nonlinear continuous systems. Unlike the previous prototype, the current implementation does not depend on the over-approximation tool CAPD, and invokes an improved Lohner's QR method to tame the infamous wrapping effect.

1 Introduction

Nonlinear ordinary differential equations (ODEs) are ubiquitous in the formal modeling of cyber-physical and biological systems [15, 18, 28, 8]. Unfortunately, exact solutions of ODEs are limited to linear systems. Thus, the verification of safety-critical nonlinear systems requires one to compute an over-approximation of their reachable states, in as tight a fashion as possible.

This work continues the line of research on Lagrangian Reachtube analysis (LRT) [11, 12], a formal over-approximation approach for nonlinear continuous ODEs, based on their associated variational equations [13, 14, 19, 9]. Roughly speaking, variational equations determine tangents along a given family of nonlinear ODEs solutions. The underlying information, in particular the Cauchy-Green stretching factor (SF), is then used to construct a reachtube that tightly over-approximates the set of reachable states at each point in time.

Our main goal is to make LRT a stand-alone tool that scales to large systems of nonlinear ODEs, such as the ones associated with deep (recurrent) neural networks or our deep neural regulatory networks [20]. To this end, we first aimed to remove the dependence of the LRT tool we presented in [11, 12], on the verification tool CAPD [7, 29, 30]. This required us to replace the CAPD routines with verified integration schemes [22, 24, 4], an approach also taken by other tools, including CAPD, CORA [6] and Flow* [10]. Additionally, we take advantage of

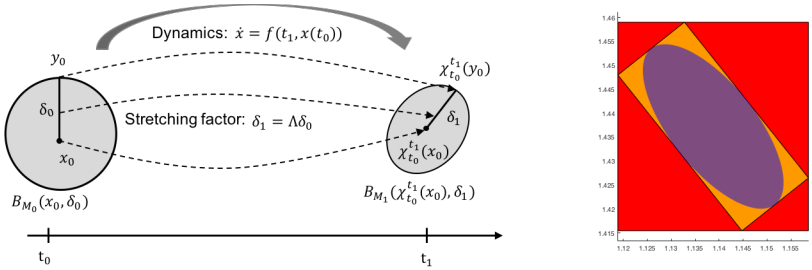

Figure 1: (Left) Visualization of one step of the Lagrangian Reachtube (LRT). The dashed arrows reflect the solution flow χ of Eq. (1)(left). In this figure, $\delta_0 = \eta\xi_0$ and $\delta_1 = \eta\xi_1$. (Right) Brusselator snapshot of the wrapping deficiency of Line 3 in Algorithm 1.

an improved Lohner's QR method [21, 24] to account for the infamous wrapping effect, which is intrinsically connected to interval arithmetic [27]. For ease of presentation, we discuss our approach in the context of the Euler method, the simplest integration scheme available. Our prototype, however, supports also higher-order Runge-Kutta schemes.

This paper presents work in progress. The performance of our current implementation is still below that of one of our previous LRT tools on the benchmarks in [11, 12]. The reasons for the lower performance and various ways of improving it are discussed in Section 5.

Notation. We denote by I the identity matrix in $\mathbb{R}^{n\times n}$, and by ∂_x the partial derivative with respect to variable x. We denote by $\|\cdot\|_2$ the *Euclidean norm*, and by $\|\cdot\|_\infty$ the maximum norm. A similar notation is used for the induced operator norms. Following standard notation, the symbol $\succ 0$ stands for positive definiteness. For a given square matrix $M \succ 0$, let $B_M(x,\delta)$ be the closed ball with center x and radius $\delta > 0$ with respect to the metric $\|x\|_M := \sqrt{x^T M x}$. The notation $[x]$ is used for a box in $\mathbb{R}^n$, i.e., a product of compact intervals. We denote by $\mathrm{mid}([x])$ and $\mathrm{rad}([x])$ the midpoint and the radius of $[x]$, respectively. We call an $n \times n$ matrix $[\mathcal{F}]$ an interval matrix, if for all $1 \le i, j \le n$, the entries $[\mathcal{F}](i,j)$ of $[\mathcal{F}]$ are intervals.

2 Preliminaries

We study a system of nonlinear ODEs in unknown $x \in \mathbb{R}^n$, as given in Eq. 1(left), with the corresponding system of variational equations in unknown $\xi \in \mathbb{R}^n$, as given in Eq. 1(right), where the field $f : \mathbb{R}^n \to \mathbb{R}^n$ is assumed to be a sufficiently smooth, time-invariant function (at least twice continuously differentiable):

$$\partial_t x = f(x), \quad x_0 = x(t_0) \qquad \text{and} \qquad \partial_t \xi = (\partial_x f)(x) \cdot \xi, \quad \xi_0 = \xi(t_0). \tag{1}$$

Since time dependence can be incorporated by adding the auxiliary variable $\partial_t x_0 = 1$ to the system, our discussion naturally extends to time-varying systems of the form $\partial_t x = f(t, x)$.

The approach presented in this paper is based on our recent work on Lagrangian reachability (LRT), where the main idea is to use the Variational Eq. (1)(right), for obtaining a tight over-approximation [11, 12]. More specifically, let $\chi_{t_0}^{t_1}$ be the solution flow induced by Eq. (1)(left), i.e., $\chi_{t_0}^{t_1}(x_0) = x(t_1)$ when $\partial_t x = f(x)$ and $x(t_0) = x_0$. The gradient of the solution flow with respect to initial conditions, known as the sensitivity matrix [13, 14], is defined by the function $x \mapsto (\partial_x \chi_{t_0}^{t_1})(x)$ and satisfies, for $\xi_0 \in \mathbb{R}^n$ and small $\eta \in \mathbb{R}$, the relation:

$$\chi_{t_0}^{t_1}(x_0 + \eta\xi_0) = \chi_{t_0}^{t_1}(x_0) + \eta \cdot (\partial_x \chi_{t_0}^{t_1})(x_0) \cdot \xi_0 + \mathcal{O}(\eta^2). \tag{2}$$

The above equation allows one to approximate $\chi_{t_0}^{t_1}(x_0+\eta\xi_0)$ using $\chi_{t_0}^{t_1}(x_0)$ and $(\partial_x\chi_{t_0}^{t_1})(x_0)$, when η is small. Moreover, the sensitivity matrix $(\partial_x\chi_{t_0}^{t_1})(x_0)$ can be related to the variational equations via $(\partial_x\chi_{t_0}^{t_1})(x_0)\cdot\xi_0=\xi(t_1)$, provided that ξ satisfies $\partial_t\xi=(\partial_x f)(x)\cdot\xi$, with $\xi(t_0)=\xi_0$, and that x obeys $\partial_t x=f(x)$ for $x(t_0)=x_0$ (see Chapter V, Theorem 3.1 in [16]). This motivates us to consider the gradient of the flow matrix (called simply the gradient) $F(t,x_0)$ that is given by the variational matrix equation $\partial_t F(t,x_0)=(\partial_x f)(x(t))\cdot F(t,x_0)$. This is because $F(t_1,x_0)\cdot\xi_0=\xi(t_1)$, when $\partial_t\xi=(\partial_x f)(x)\cdot\xi$ for $\xi(t_0)=\xi_0$ and $\partial_t x=f(x)$ for $x(t_0)=x_0$. Note that the two equations in (1) in particular imply that $F(t_1,x_0)=(\partial_x\chi_{t_0}^{t_1})(x_0)$ holds.[1]

Next, we formally relate the over-approximation of (1) to its variational equations. To this end, we fix two matrices $M_0, M_1\succ 0$ and consider as the initial region $B_{M_0}(x_0,\delta_0)$, i.e., the ball in metric space M_0 centered at x_0 with radius δ_0. Moreover, we chose y_0 to be a point on the surface of $B_{M_0}(x_0,\delta_0)$ and set $x_0'=\chi_{t_0}^{t_1}(x_0)$ and $y_0'=\chi_{t_0}^{t_1}(y_0)$. Let δ_1 be the distance between y_0' and x_0' in the metric space defined by matrix M_1, as shown in Figure 1(left).

The Cauchy Green stretching factor (SF) Λ measures the deformation of the ball $B_{M_0}(x_0,\delta_0)$ into the ball $B_{M_1}(x_0',\delta_1)$, i.e., $\Lambda=\delta_1/\delta_0$. One can thus use the SF to bound the infinite set of reachable states at time t_1 with the ball-overestimate $B_{M_1}(\chi_{t_0}^{t_1}(x_0),\delta_1)$ in an appropriate metric $M_1\succ 0$, which may differ from $M_0\succ 0$. If $M_1=M_0$ we refer to the computed SF as M_0-SF or M_1-SF. Instead, if $M_0\neq M_1$, we refer to the computed SF as $M_{0,1}$-SF.

The correctness of the LRT approach is rooted in the following theorem.

Theorem 1 (Thm. 1 in [12]). *Let $t_0\le t_1$ be time points, and $\chi_{t_0}^{t_1}(x)$ be the solution at t_1 of the Cauchy problem (1), with initial condition (t_0,x). Let $M_0,M_1\in\mathbb{R}^{n\times n}$ with $M_0,M_1\succ 0$, and let $A_0^TA_0=M_0$, $A_1^TA_1=M_1$ be their respective decompositions. Let the ball in the M_0-norm with center x_0 and radius δ_0, $\mathcal{B}=B_{M_0}(x_0,\delta_0)\subseteq\mathbb{R}^n$, be a set of initial states for Eq. (1). Assume that there exists a compact enclosure $[\mathcal{F}]\subseteq\mathbb{R}^{n\times n}$ for the gradients F such that:*

$$(\partial_x\chi_{t_0}^{t_1})(x)\in[\mathcal{F}],\quad \forall x\in\mathcal{B}. \tag{3}$$

Suppose $\Lambda>0$ is an upper bound of the whole set of $M_{0,1}$ SFs [11], that is:

$$\Lambda\ge\sqrt{\lambda_{max}\left((A_0^T)^{-1}F^TM_1FA_0^{-1}\right)}=\|A_1FA_0^{-1}\|_2,\quad\forall F\in[\mathcal{F}], \tag{4}$$

where λ_{max} represents the maximum eigenvalue. Then,

$$\chi_{t_0}^{t_1}(x)\in B_{M_1}(\chi_{t_0}^{t_1}(x_0),\Lambda\cdot\delta_0), \tag{5}$$

meaning that the reachable states at time t_1 are contained in the ball $B_{M_1}(\chi_{t_0}^{t_1}(x_0),\Lambda\cdot\delta_0)$.

In order to obtain a tight over-approximation, we wish to minimize Λ, the upper bound for the $M_{0,1}$-SF given in Theorem 1. Since finding the minimal value across all $F\in[\mathcal{F}]$ is a difficult task, we use the following heuristics. Given that the set $[\mathcal{F}]$ is represented by an interval matrix in our algorithm, we pick the M_1 that minimizes the M_1-SF with respect to $\mathrm{mid}\,([\mathcal{F}])$, where $\mathrm{mid}\,([\mathcal{F}])$ is the value of F at the center of $[\mathcal{F}]$. In [12], the following formula for such an M_1, conveniently denoted by $\hat{M}_1$, has been obtained.

Theorem 2 (Thm. 2 in [12]). *Let (gradient) $F\in\mathbb{R}^{n\times n}$ have full rank and $\hat{A}_1$, $\hat{M}_1$ be such that*

$$\hat{A}_1(F)=V(F)^{-1}\quad\textit{and}\quad\hat{M}_1(F)=\hat{A}_1(F)^T\hat{A}_1(F), \tag{6}$$

[1] Proof sketch: $\partial_t(\partial_x\chi_{t_0}^t)(x)=\partial_x(\partial_t\chi_{t_0}^t)(x)=\partial_x(f(\chi_{t_0}^t))(x)=(\partial_x f)(\chi_{t_0}^t(x))\cdot(\partial_x\chi_{t_0}^t)(x)$.

Algorithm 1 Lagrangian Reachability Algorithm [11, 12].

Require: Reals $C_M, \eta > 0$, horizon $T > 0$, maximal step $h > 0$, time $t_0 \geq 0$, set $B_{M_0}(x_0, \delta_0)$

set $\delta \leftarrow \delta_0, \quad \delta_x \leftarrow 0, \quad b \leftarrow 0, \quad [\mathcal{F}_0] \leftarrow \{I\}$
while $t_0 < T$ **do**
  compute over-approximation $[\mathcal{X}]$ of $B_{M_0}(x_0, \delta_0)$ in Cartesian coordinates
  set $t_1 \leftarrow t_0 + 2h$
  repeat
    set $t_1 \leftarrow t_0 + \frac{1}{2}(t_1 - t_0)$
    compute over-approximation $[\mathcal{F}_1^*]$ of $\left(\partial_x \chi_{t_0}^{t_1}\right)([\mathcal{X}])$
  until $\lambda < \eta$, where λ estimates $\|[\mathcal{F}_1^*] - [\mathcal{F}_0]\|_2$
  compute tighter over-approximation $[\mathcal{F}_1]$ of $\left(\partial_x \chi_{t_0}^{t_1}\right)([\mathcal{X}])$ for final t_1
  compute Λ_0 as a bound of the M_0-SF
  compute optimal metric $\hat{M}_1(F) = \hat{A}_1(F)^T \hat{A}_1(F)$ for $F = \mathrm{mid}\,([\mathcal{F}_1])$ (Thm. 2) and Λ_1 as a bound of the M_1-SF
  if $\Lambda_0 > C_M \cdot \Lambda_1$ **then**
    set Λ to a bound of the $M_{0,1}$-SF (Thm. 1)
    set $M_1 \leftarrow \hat{M}_1, \quad b \leftarrow 1, \quad [\mathcal{F}_0] \leftarrow \{I\}$
  else
    set $M_1 \leftarrow M_0, \quad \Lambda \leftarrow \Lambda_0, \quad [\mathcal{F}_0] \leftarrow [\mathcal{F}_1]$
  end if
  compute over-approximation $[x_1]$ of $\chi_{t_0}^{t_1}(x_0)$ and **set** $x_1 \leftarrow \mathrm{mid}([x_1])$, $\delta_{x_1} \leftarrow \mathrm{rad}\,([x_1])$
  set $\delta_x \leftarrow \delta_x + \delta_{x_1}$
  set $\delta_1 \leftarrow \Lambda \cdot \delta + \delta_x$ (this specifies $B_{M_1}(x_1, \delta_1)$ in full)
  if $b = 1$ **then**
    set $\delta \leftarrow \delta_1, \quad \delta_x \leftarrow 0, \quad b \leftarrow 0$ (only if the norm is changed)
  end if
  set $(M_0, x_0, \delta_0, t_0) \leftarrow (M_1, x_1, \delta_1, t_1)$ (prepare for next iteration)
end while

where matrices $V(F)$ and $\hat{A}_1(F)^{-1}$ contain the normalized eigenvectors of F. Then:

$$\min_{\substack{A_1 \in \mathbb{R}^{n\times n} \\ A_1 \text{ is invertible}}} \Lambda(A_1, F) = \Lambda(\hat{A}_1, F),$$

where the M_1-SF is given by $\Lambda(A_1, F) = \sqrt{\lambda_{max}\left((A_1^T)^{-1} F^T M_1 F A_1^{-1}\right)} = \left\|A_1 F A_1^{-1}\right\|_2$. In other words, the symmetric positive-definite matrix $\hat{M}_1 = \hat{A}_1^T \hat{A}_1$ minimizes the M_1-SF.

3 Lagrangian Reachability Algorithm

Algorithm 1 summarizes the overall approach. Following [11, 12], the choice of the current step size $t_1 - t_0$ is adaptive and based on infinitesimal strain theory (IST). Specifically, the time-step is successively halved until the displacement gradient is sufficiently small. This is to help ensure that the over-approximation at the next time step is tight. The algorithm does not specify how the over-approximation of $[\mathcal{X}]$ and $[\mathcal{F}]$ is accomplished. This is discussed below.

By invoking low-order schemes, we aim to improve the scalability of [11, 12]. Previously, we appealed to the CAPD verification tool [7, 29, 30], which utilizes high-order Taylor numerical schemes. To simplify the presentation, we next discuss only the Euler's method. The wrapping effect is addressed by an improved version of the Lohner's QR method [21, 24] and a careful choice of the next norm during the computation of the SF.

Computation of $[\mathcal{X}]$ *in Algorithm 1.* The dynamical system specified by Eq. (1) is defined with respect to Cartesian coordinates (CC). In contrast, the over-approximation of the reachable states $B_{M_0}(x_0, \delta_0)$ from the previous ball $B_{M_i}(x_i, \delta_i)$ in one integration step is given in norm M_0. To work in CC, we first estimate $B_{M_0}(x_0, \delta_0)$ by a rectangle $[\mathcal{X}]$, as shown in line 3 of Algorithm 1 and Figure 1(right). To this end, we first compute a rectangle $[\mathcal{X}_{M_0}]$ enclosing the ball $B_{M_0}(x_0, \delta_0)$ in the A_0^{-1} coordinate system, where $M_0 = A_0^T \cdot A_0$. Afterwards, we transform $[\mathcal{X}_{M_0}]$ to CC (the orange box) to finally obtain $[\mathcal{X}]$ (the enclosing red box of [11, 12]).

Computation of $[\mathcal{F}]$ *in Algorithm 1.* Let $\phi_{t_i}^{t_{i+1}}$ be the solution flow induced by the variational matrix equations, i.e., $\phi_{t_i}^{t_{i+1}}(F_i, x_i) = F_{i+1}$ when $\partial_t F = (\partial_x f)(x) \cdot F$, $F(t_i) = F_i$ and $x(t_i) = x_i$. Given an interval matrix $[\mathcal{F}_i] \subseteq \mathbb{R}^{n \times n}$, our goal is to compute the interval matrix $[\mathcal{F}_{i+1}]$ such that $\phi_{t_i}^{t_{i+1}}([\mathcal{F}_i], [\mathcal{X}_i]) \subseteq [\mathcal{F}_{i+1}]$. Due to the linearity of the variational equations, it holds that $\phi_{t_i}^{t_{i+1}}([\mathcal{F}_i], [\mathcal{X}_i]) \subseteq \phi_{t_0}^{t_0+h_i}(I, [\mathcal{X}_i]) \cdot [\mathcal{F}_i]$ with $h_i := t_{i+1} - t_i$ [30]. This breaks down to finding an enclosure $[J_{i+1}]$ such that $\phi_{t_0}^{t_0+h_i}(I, [\mathcal{X}_i]) \subseteq [J_{i+1}]$. Using the first order enclosure method [24], we compute rough enclosures $[\tilde{\mathcal{X}}_i]$ and $[\tilde{J}_i]$ that satisfy:

$$\Big[[\mathcal{X}_i] + [0, h_i] \cdot f([\tilde{\mathcal{X}}_i])\Big] \subseteq [\tilde{\mathcal{X}}_i] \qquad \text{with } \chi_{t_i}^{t_{i+1}}([\mathcal{X}_i]) \subseteq [\tilde{\mathcal{X}}_i],$$
$$\Big[I + [0, h_i] \cdot \partial_x f([\tilde{\mathcal{X}}_i]) \cdot [\tilde{J}_i]\Big] \subseteq [\tilde{J}_i] \qquad \text{with } \phi_{t_0}^{t_0+h_i}(I, [\mathcal{X}_i]) \subseteq [\tilde{J}_i].$$

As proposed in [30], we compute $[\tilde{J}_i]$ by setting each element of the matrix to the interval $[\pm \max(e^{l[0,h_i]})]$ and refine this enclosure by using the variational matrix equations related to (1), where l denotes the logarithmic Euclidean norm of $\partial_x f([\tilde{\mathcal{X}}_i])$. Afterwards, the tight enclosure $[J_{i+1}]$ is obtained by exploiting the relation:

$$[J_{i+1}] \subseteq I + h_i \partial_x f([\mathcal{X}_i]) \cdot I + \underbrace{\frac{h_i^2}{2} \partial_t \Big((\partial_x f)(x(t)) \cdot F(t, x) \Big) \Big([\tilde{\mathcal{X}}_i], I \Big) \cdot [\tilde{J}_i]}_{\text{local truncation error}},$$

which underlies Euler's method. By recalling that $\partial_t F(t, x_0) = (\partial_x f)(x(t)) \cdot F(t, x_0)$, it holds that $F(t + h_i, x_0) = F(h_i, x(t)) \cdot F(t, x_0)$, as has been observed in [30]. Hence:

$$[\mathcal{F}_{i+1}] = [J_{i+1}] \cdot [\mathcal{F}_i]. \tag{7}$$

To allow for tight estimation, we do not evaluate the right-hand side of (7) using interval arithmetics. Instead, we pick a point matrix $\hat{\mathcal{F}}_i \in [\mathcal{F}]$, set $[\Delta\mathcal{F}_i] := [\mathcal{F}_i] - \hat{\mathcal{F}}_i$ and write (7) as:

$$[\Delta\mathcal{F}_{i+1}] = [J_{i+1}] \cdot [\Delta\mathcal{F}_i] + [\Delta Z_{i+1}] \tag{8}$$

where we set:

$$[Z_{i+1}] = [J_{i+1}] \cdot \hat{\mathcal{F}}_i - \hat{\mathcal{F}}_{i+1}, \qquad \hat{\mathcal{F}}_{i+1} = \mathrm{mid}([Z_{i+1}]), \qquad [\Delta Z_{i+1}] = [Z_{i+1}] - \hat{\mathcal{F}}_{i+1}.$$

To combat the wrapping effect, we use the Lohner's QR factorization method, where in each step a new basis Q_i for $[\Delta\mathcal{F}_i]$ is chosen such that $[\Delta\mathcal{F}_i] = Q_i \cdot [\Delta\mathcal{F}_i^{Q_i}]$ and that:

$$[\Delta\mathcal{F}_{i+1}^{Q_{i+1}}] = \big([Q_{i+1}^{-1}][J_{i+1}][Q_i]\big)\,[\Delta\mathcal{F}_i^{Q_i}] + [Q_{i+1}^{-1}][\Delta Z_{i+1}].$$

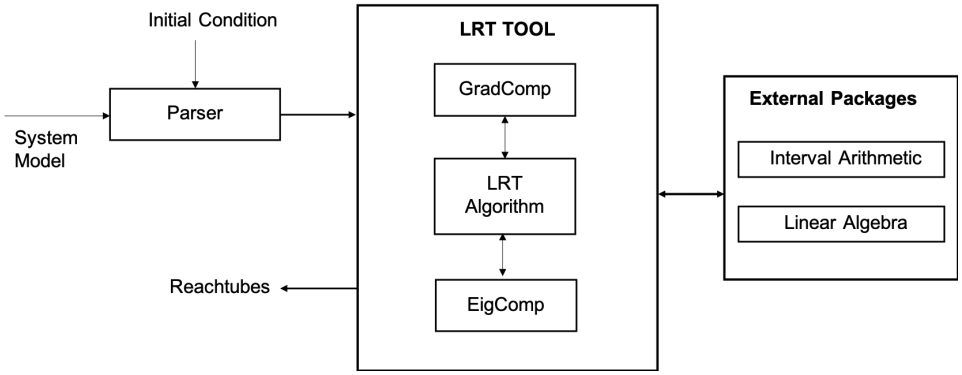

Figure 2: Overall architecture of our new stand-alone LRT tool. GradComp stands for Gradient Computation, and EigComp stands for Eigenvalue Computation.

At the first step, we set $Q_0 := I$ and $\hat{\mathcal{F}}_0 = \text{mid}([\mathcal{F}_0])$, thus ensuring $[\Delta\mathcal{F}_0^{Q_0}] = [\Delta\mathcal{F}_0] = 0$. In each step thereafter, the new coordinate system Q_{i+1} is chosen starting with $U_{i+1} = \text{mid}\left([J_{i+1}][Q_i]\right)$. Following [24], the columns of U_{i+1} are rearranged by a permutation matrix P_{i+1} such that the first column of $\tilde{U}_{i+1}(= U_{i+1} \cdot P_{i+1})$ corresponds to the longest edge of $U_{i+1} \cdot [\Delta\mathcal{F}_i^{Q_i}]$, the second column to the second longest, and so on. To obtain the new coordinate system Q_{i+1} we set it to the orthogonal matrix of the QR-factorization of $\tilde{U}_{i+1}$. For a more detailed discussion of the steps discussed above, please refer to [24] and [30].

Putting everything together. We are now in a position to explain Algorithm 1 in detail. More specifically, $[x_{i+1}]$ in line 18 is computed by applying Runge-Kutta of second order (RK2) to f; instead, $[\mathcal{F}_1^*]$ in line 7 is computed by applying RK2 to the variational matrix equations, without taking the local truncation error into account. This is faster than using Lohner's method. Moreover, the absence of a rigorous bound is not critical because $[\mathcal{F}_1^*]$ is only used to set the next timestep. Instead, $[\mathcal{F}_1]$ in line 9 is obtained by applying Euler's and Lohner's method to the variational matrix equations. Since $[\mathcal{F}_1]$ is stored as an interval matrix, the estimation of Λ_0, Λ_1 and Λ as the bounds for the M_0-, M_1- and $M_{0,1}$-SF in the lines 10, 11 and 14, respectively, amounts to the estimation of the largest eigenvalue; see also Theorem 1. To this end, we rely on the interval matrix algorithms of IBEX [2] and Eigen [1].

4 Tool Architecture

The LRT tool implements Algorithm 1 which is based on the fundamental research we presented in [11, 12]. We ensure soundness of LRT by performing all computations in interval arithmetic [23], where a real-valued variable x is represented as an interval $[\underline{x}, \bar{x}]$ such that $x \geq \underline{x}$ and $x \leq \bar{x}$. The architecture of the stand-alone LRT is illustrated in Figure 2. Below we discuss each of its major components, which gives a complementary view of Algorithm 1:

- **Parser:** The system dynamics and the initial condition are defined in two separate text files. Currently, we are using IBEX's built-in functionality to parse the systems dynamics, and we implemented our own parser for the initial conditions.

- **SF computation:** The SF Λ in Figure 1 is computed based on the Cauchy-Green interval matrix, which depends on the gradient of the solution flow starting at $B_{M_0}(x_0, \delta_0)$. Hence, we need to compute a compact, conservative interval gradient matrix $[\mathcal{F}_1]$ such that:

$$\forall x \in B_{M_0}(x_0, \delta_0) \quad \text{it holds that} \quad (\partial_x \chi_{t_0}^{t_1})(x) \in [\mathcal{F}_1]$$

- **Eigenvalue computation for $[\mathcal{F}_1]$:** Given $[\mathcal{F}_1]$, the next step of LRT is to compute the maximum eigenvalue of the Cauchy-Green (CG) interval matrix $CG = [\mathcal{F}_1]^T \cdot [\mathcal{F}_1]$, which is a symmetric positive-definite interval matrix (tensor).[2] To compute an upper bound of the maximal eigenvalue of all symmetric matrices in some interval bounds, we implemented several algorithms from [17, 26, 25] and use the tightest result available.
- **External Packages:** For interval arithmetic and linear algebra operations, we use IBEX [2] C++ library and Eigen [1] C++ library, respectively.

5 Conclusion

The main motivation of our current work was to scale up (C)LRT to high-dimensional systems, for example, deep neural networks and the deep neural regulatory networks we have introduced in [20]. To this end, we considered the use of lower-order integration schemes, such as the advanced Runge-Kutta methods recently presented in [3], for the numerical computation of the (interval) sensitivity matrix $[\mathcal{F}]$. The main reason behind this consideration is that computing the truncation order for high-dimensional systems is computationally prohibitive [3].

Other tools such as CAPD [7], which we used in [11, 12], also employ a variant of the C1-Lohner method to integrate the variational matrix equations of a given set of ODEs for obtaining $[\mathcal{F}]$. CAPD, however, uses higher-order integration schemes based on the Taylor expansion, whereas we desire to keep the order low. Moreover, CAPD only uses the C1-Lohner method to combat the wrapping effect, while we combine it with a change of norm.

We conducted a preliminary evaluation of our stand-alone LRT tool on the benchmark models from [11, 12]. We were able to recover the tight over-approximation of the 2-dimensional Brusselator, the 2-dimensional Mitchell-Schaeffer cardiac-cell model, and the 12-dimensional Polynomial System. The other models from [12], however, led to blowups in the over-approximation. CAPD [7], CORA [5] and Flow* [10] also seem to suffer from similar blowups if restricted to low-order schemes (or Taylor models). Hence, we will need to increase this order.

We will also explore three ways of improving performance. First, we will seek to reduce the wrapping deficiency of line 3 in Algorithm 1. This can be achieved by working in the M_i norm with both $[\mathcal{F}_i]$ and $B_{M_i}(x_i, \delta_i)$, or by working in the Cartesian norm and employing a QR decomposition to support the use of the orange box shown in Figure 1(right), instead of the red box. Second, we wish to obtain a better Cauchy Green SF. As per Theorem 2, the tightness of the SF depends on the choice of metric M_i, which in turn depends on the eigenvalues of the particular F_i chosen out of $[\mathcal{F}_i]$. Since the F_i's can significantly differ, we would like to explore genetic- or gradient-based techniques for their optimal choice. Third, we will improve the efficiency of the stand-alone LRT tool by tailoring it to specific classes of ODEs, in particular the ones corresponding to deep neural regulatory networks [20].

Acknowledgements. Sophie Gruenbacher is funded by FWF project W1255-N23. Max Tschaikowski is supported by a Lise Meitner Fellowship funded by the Austrian Science Fund (FWF) under grant number M-2393-N32 (COCO).

[2] In fact we compute maximum eigenvalue of the CG matrix with the norm change: $(A_0^T)^{-1}[\mathcal{F}_1^T]M_1[\mathcal{F}_1]A_0^{-1}$.

References

[1] Eigen Linear Algebra Library. http://eigen.tuxfamily.org. Copyright: Mozilla Public License Version 2.0.

[2] IBEX Interval Library. http://ibex-lib.org. Copyright: GNU Lesser General Public License.

[3] J. Alexandre dit Sandretto and A. Chapoutot. Validated explicit and implicit Runge-Kutta methods. *Reliable Computing (Electronic Edition)*, 22, July 2016.

[4] M. Althoff. Reachability analysis of nonlinear systems using conservative polynomialization and non-convex sets. In *International Conference on Hybrid Systems: Computation and Control*, pages 173–182, 2013.

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

[6] M. Althoff, D. Grebenyuk, and N. Kochdumper. Implementation of Taylor models in CORA 2018. In *Proc. of the 5th International Workshop on Applied Verification for Continuous and Hybrid Systems*, pages 145–173, 2018.

[7] M. Capiński, J. Cyranka, Z. Galias, T. Kapela, M. Mrozek, P. Pilarczyk, D. Wilczak, P. Zgliczyński, and M. Zelawski. CAPD - computer assisted proofs in dynamics, a package for rigorous numerics, http://capd.ii.edu.pl/. Technical report, Jagiellonian University, Kraków, 2016.

[8] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Maximal aggregation of polynomial dynamical systems. *Proceedings of the National Academy of Sciences*, 114(38):10029 – 10034, 2017.

[9] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Guaranteed error bounds on approximate model abstractions through reachability analysis. In *International Conference on Quantitative Evaluation of SysTems*, pages 104–121, 2018.

[10] X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *CAV'13, the 25th International Conference on Computer Aided Verification*, pages 258–263, Saint Petersburg, Russia, July 2013. Springer.

[11] J. Cyranka, M. A. Islam, G. Byrne, P. Jones, S. A. Smolka, and R. Grosu. Lagrangian reachabililty. In R. Majumdar and V. Kunčak, editors, *CAV'17, the 29th International Conference on Computer-Aided Verification*, pages 379–400, Heidelberg, Germany, July 2017. Springer.

[12] J. Cyranka, M. A. Islam, S. A. Smolka, S. Gao, and R. Grosu. Tight Continuous-Time Reachtubes for Lagrangian Reachability. In *CDC'18, the 57th IEEE Conference on Decision and Control*, pages 6854–6861, Miami Beach, FL, USA, December 2018. IEEE.

[13] A. Donzé. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *CAV'10, the 22nd International Conference on Computer Aided Verification*, pages 167–170, Edinburgh, UK, July 2010. Springer.

[14] A. Donzé and O. Maler. Systematic simulation using sensitivity analysis. In *International Workshop on Hybrid Systems: Computation and Control*, pages 174–189. Springer, 2007.

[15] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In *CAV'11, the 23rd International Conference on Computer Aided Verification*, pages 379–395, Snowbird, Utah, USA, July 2011. Springer.

[16] P. Hartman. *Ordinary Differential Equations*. SIAM, 1982.

[17] M. Hladik, D. Daney, and E. Tsigaridas. Bounds on real eigenvalues and singular values of interval matrices. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2116–2129, 2010.

[18] M. A. Islam, A. Murthy, A. Girard, S. A. Smolka, and R. Grosu. Compositionality results for cardiac cell dynamics. In *International Conference on Hybrid Systems: Computation and Control*, pages 243–252, 2014.

[19] R. Lal and P. Prabhakar. Bounded error flowpipe computation of parameterized linear systems. In *Proceedings of the 12th International Conference on Embedded Software*, pages 237–246, 2015.

[20] M. Lechner, R. Hasani, M. Zimmer, T. Henzinger, and R. Grosu. Designing worm-inspired neural networks for interpretable robotic control. In *ICRA'19, the 2019 IEEE International Conference on Robotics and Automation*, Montreal, Canada, May 2019. IEEE.

[21] R. Lohner. *Computation of guaranteed enclosures for the solutions of ordinary initial and boundary value problems*, chapter Computational Ordinary Differential Equations. Clarendon Press, Oxford, 1992.

[22] M. Martel and O. Bouissou. GRKLib: a Guaranteed Runge Kutta Library. In *2006 12th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN)*, volume 00, page 8, 2006.

[23] R. E. Moore. Interval analysis. Series in automatic computation. *Englewood Cliff: NJ Prentice-Hall*, 1966.

[24] N. Nedialkov, K. Jackson, and G. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, 105(1):21 – 68, 1999.

[25] J. Rohn. Bounds on eigenvalues of interval matrices. *ZAMMZ.Angew.Math.Mech.*, 78:1049–1050, 1998.

[26] S. M. Rump. Computational error bounds for multiple or nearly multiple eigenvalues. *Linear Algebra and its Applications*, 324(1):209 – 226, 2001. Linear Algebra in Self-Validating Methods.

[27] J. K. Scott and P. I. Barton. Bounds on the reachable sets of nonlinear control systems. *Automatica*, 49(1):93 – 100, 2013.

[28] M. Tschaikowski and M. Tribastone. Approximate reduction of heterogenous nonlinear models with differential hulls. *IEEE Trans. Automat. Contr.*, 61(4):1099–1104, 2016.

[29] D. Wilczak and P. Zgliczyński. Cr-Lohner algorithm. *Schedae Informaticae*, 2011(Volume 20), 2012.

[30] P. Zgliczynski. C1 Lohner Algorithm. *Foundations of Computational Mathematics*, 2(4):429–465, 2002.