Reachability for weakly nonlinear systems using Carleman linearization

Presentation at:

OCTOBER 25-27 2021, LIVERPOOL, UK

15th International Conference on Reachability Problems

Marcelo Forets

Christian Schilling

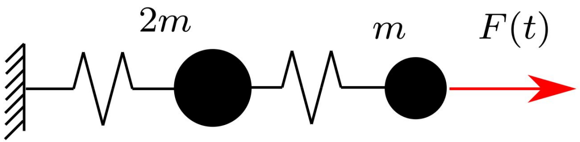
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

CURE Centro Universitario Regional del Este

JuliaReach

AALBORG UNIVERSITY DENMARK

What is reachability for continuous dynamical systems?

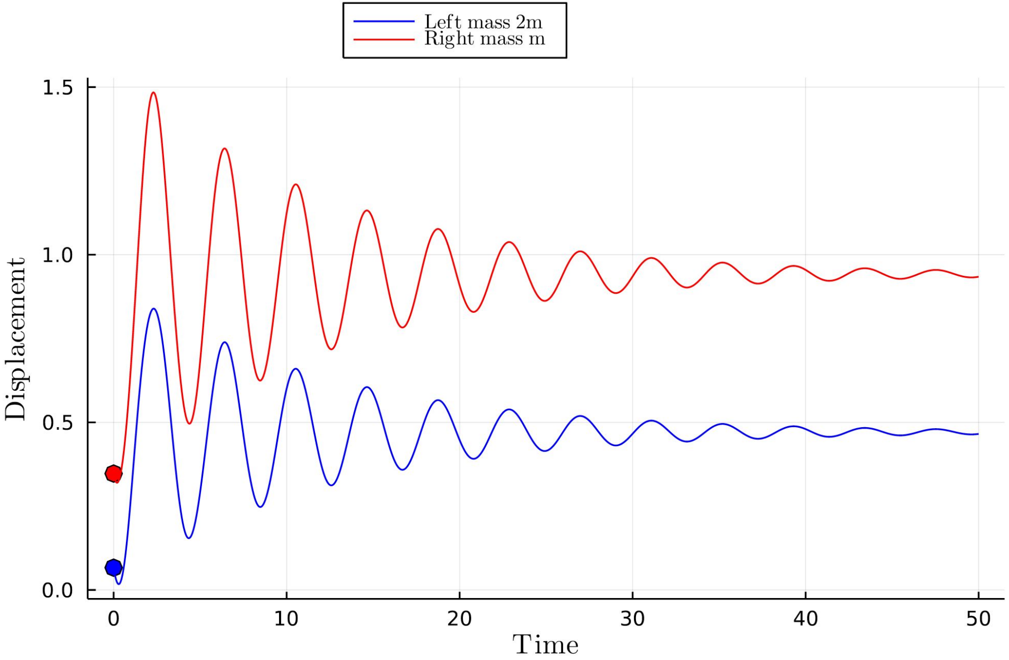


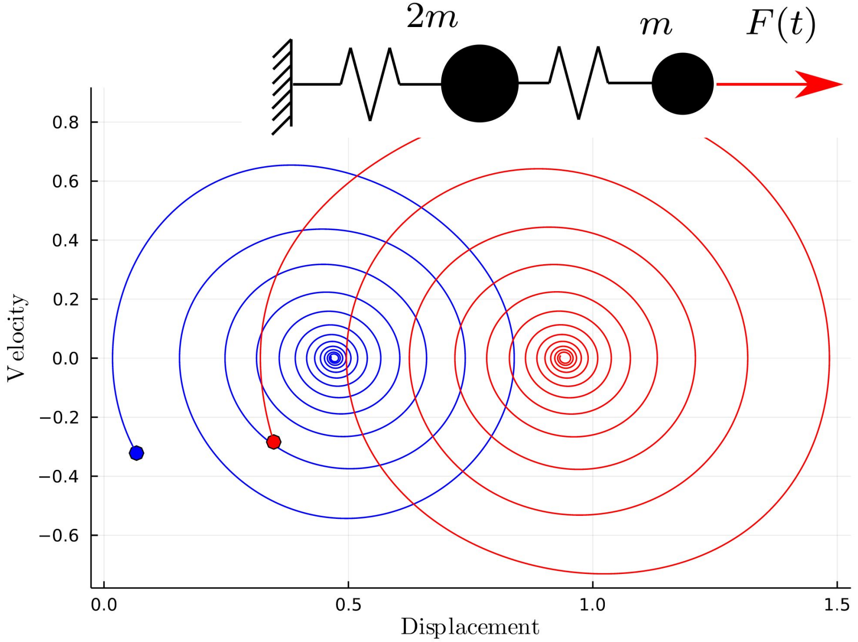
Example (linear differential equation):

$$Mx''(t) + Cx'(t) + Kx(t) = F(t)$$

Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

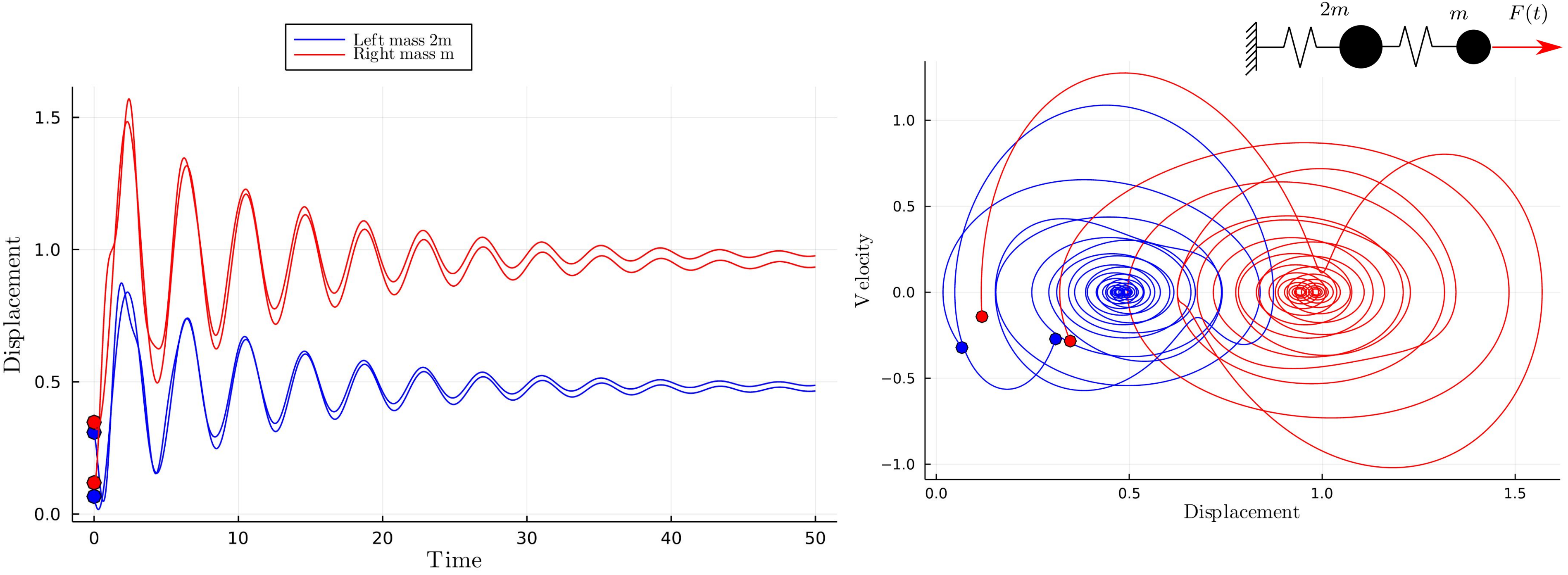
What is reachability for continuous dynamical systems?





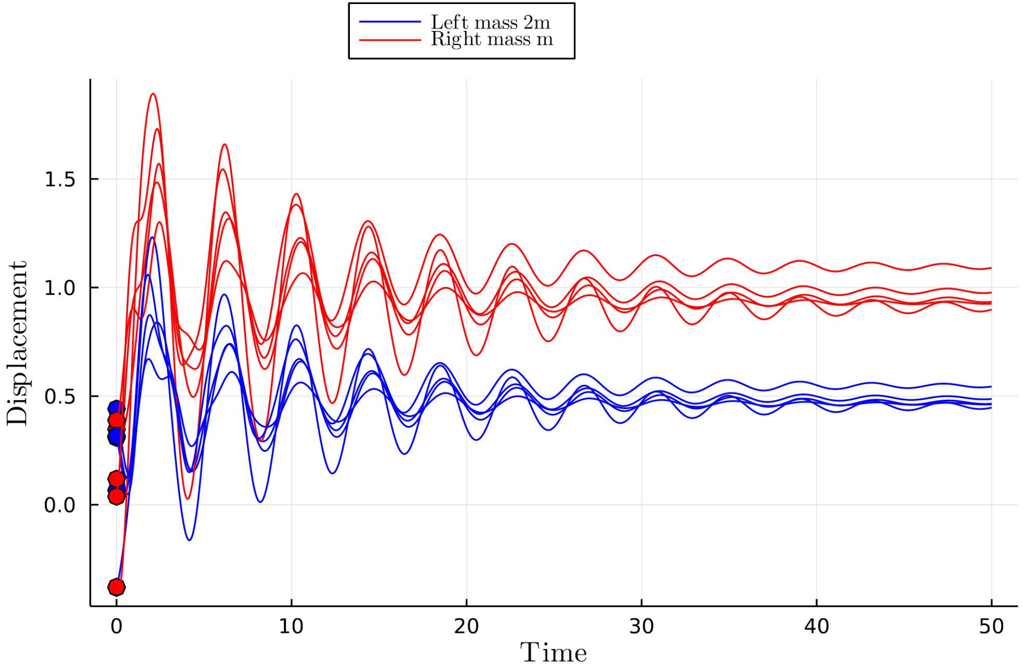
Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

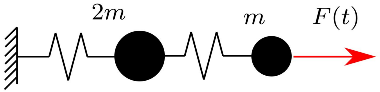
What is reachability for continuous dynamical systems?

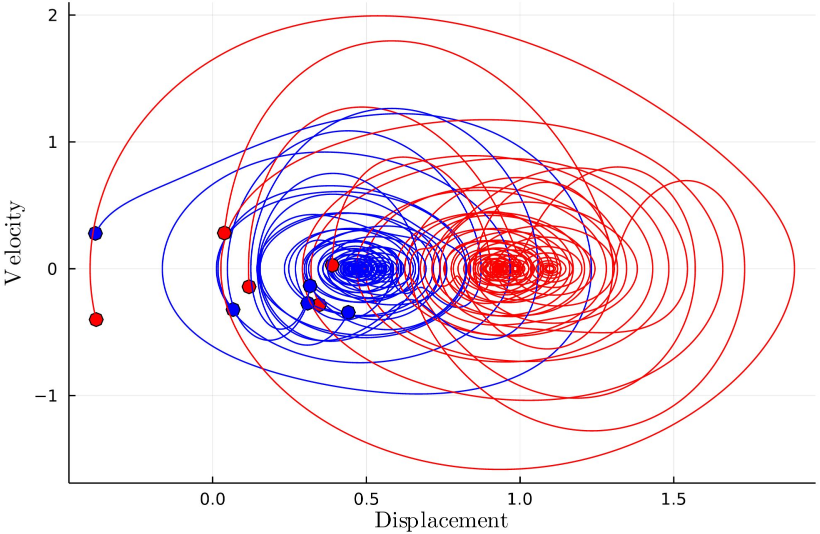


Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

What is reachability for continuous dynamical systems?

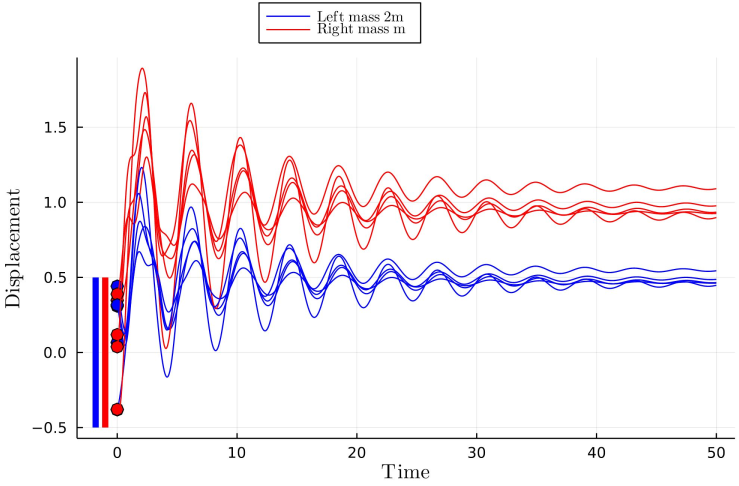


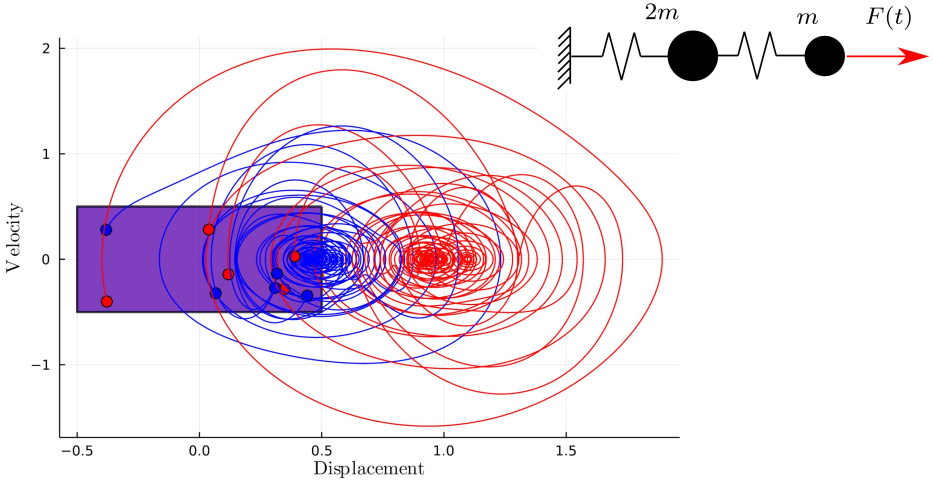




Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

What is reachability for continuous dynamical systems?

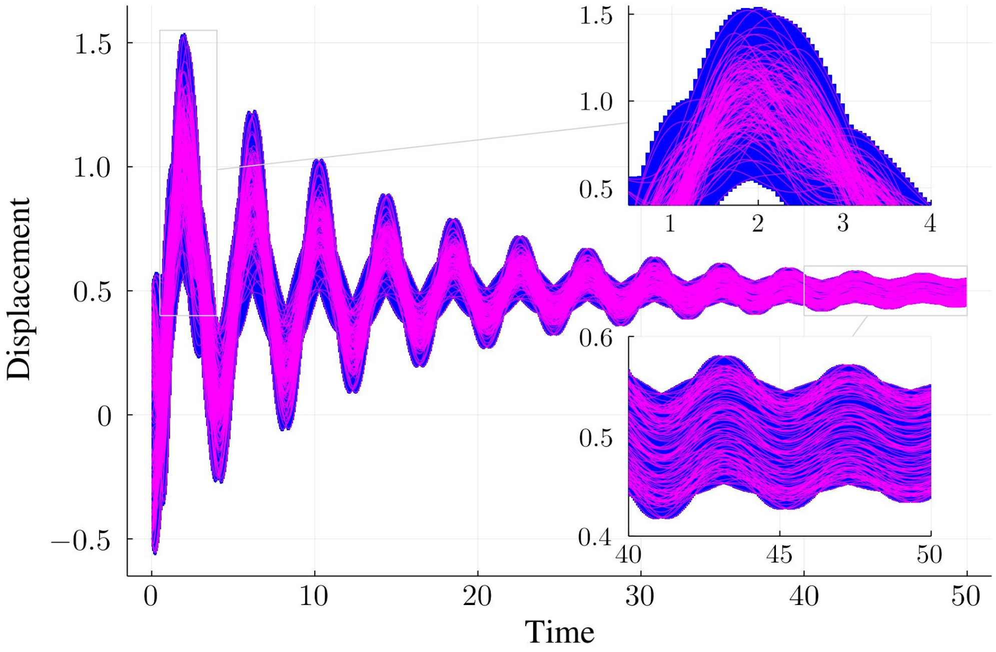




Reachability is a numerical method to compute sets of states reachable by dynamical systems for *all initial states* and all admissible *parameters* and *inputs*

Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

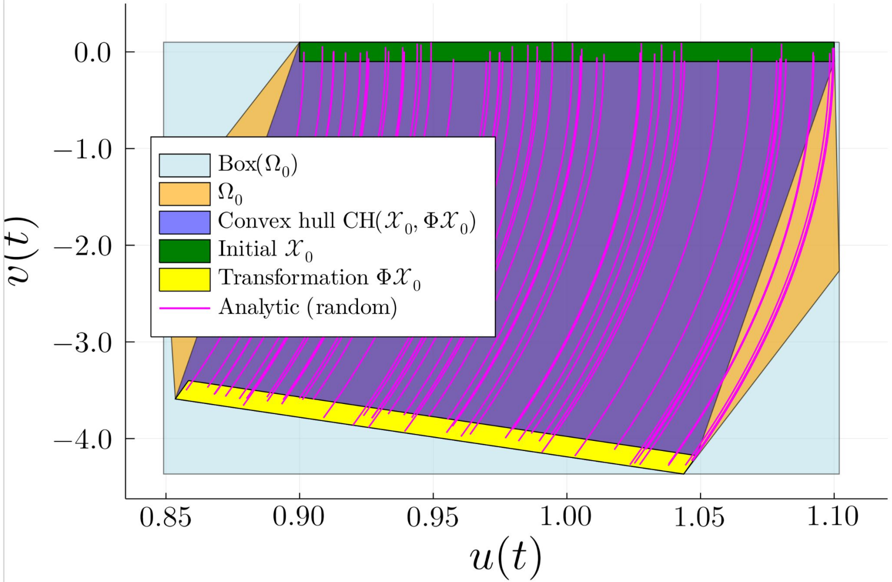
What is reachability for continuous dynamical systems?



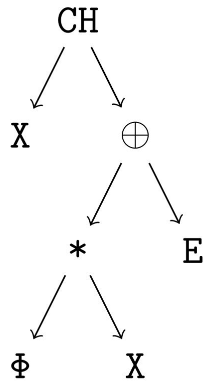

```julia
using ReachabilityAnalysis

# model parameters
m = 0.25; k = 2.0

# finite-element method assembled matrices
M = [2m 0; 0 m]; K = [2k -k; -k k]; C = (M+K)/20
F = [0.0, 1.0]; ΔF0 = Interval(0.9, 1.1)

# initial-value problem with uncertain initial conditions
U0 = BallInf(zeros(4), 0.5)
sys = SecondOrderLinearContinuousSystem(M, C, K, F)
prob = InitialValueProblem(homogeneize(sys), U0 × ΔF0)

# solve using support function method (box directions)
solA = solve(prob, 50, LGG09(δ=5e-2, dirs=:box, dim=5))
```

Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

Conservative time discretization (step 1/2)



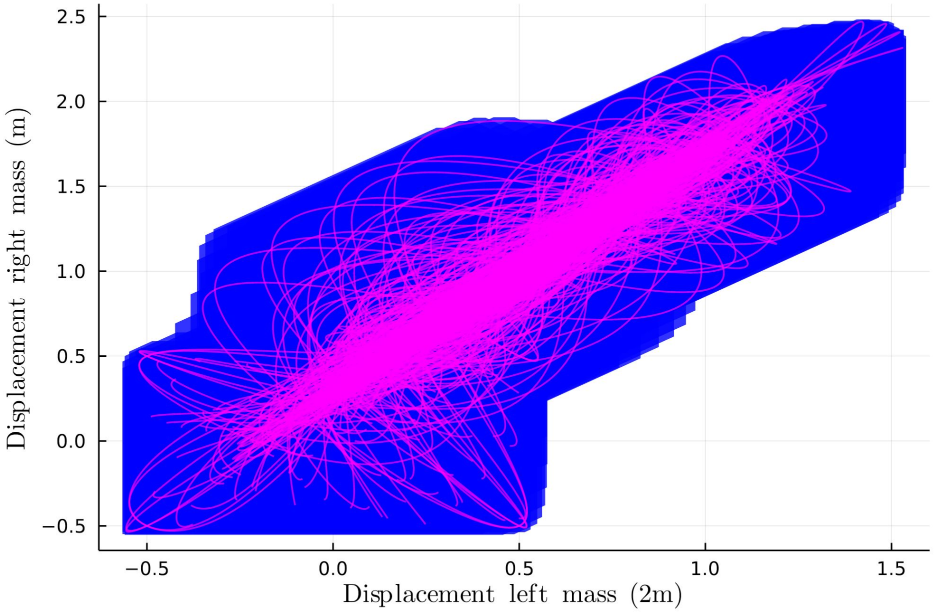

```
1  julia> Ω₀ = CH(X₀, Φ*X₀ ⊕ E₊)
```



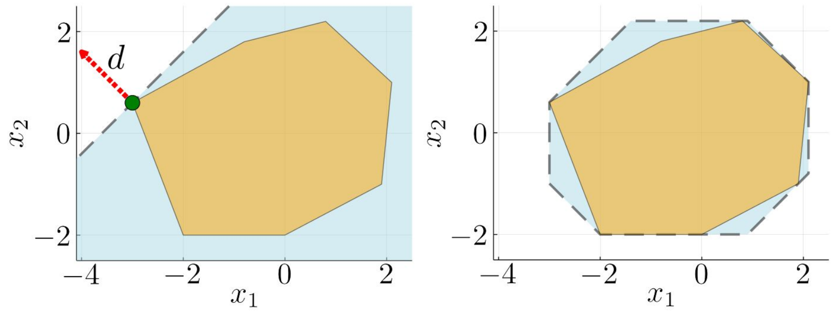
Efficient set computations require specialized algorithms based on different combinations of *set type representations* and *operations* involved.

See: *LazySets.jl: Scalable Symbolic-Numeric Set Computations.* Marcelo Forets and Christian Schilling. arXiv preprint arXiv:2110.01711 (2021). **Submitted to JuliaCon'2021 (full paper).**

Set propagation (step 2/2)

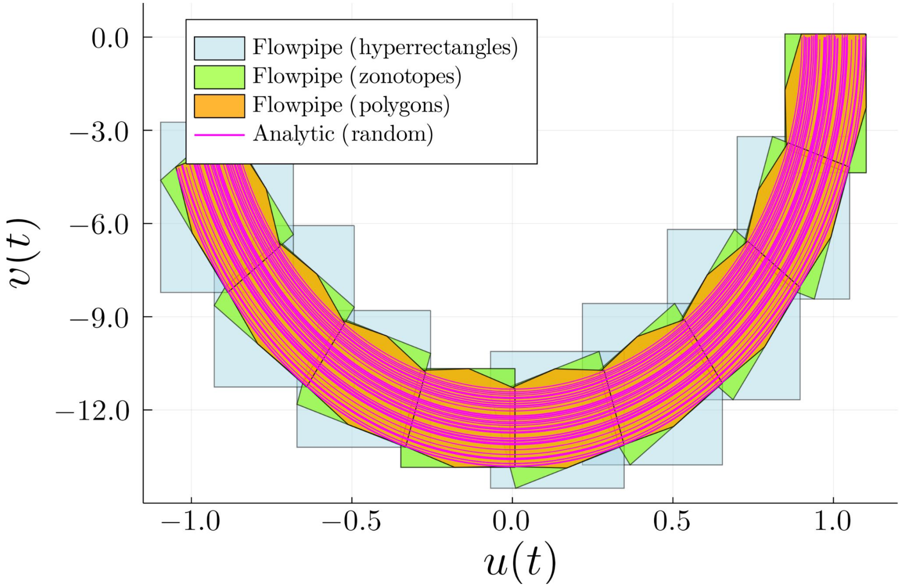


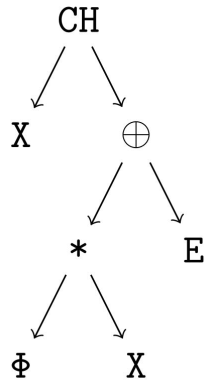

```
# solve using octagonal template directions
solB = solve(prob, 50, LGG09(δ=5e-2, dirs=:oct, dim=5))
```



Example taken from https://github.com/JuliaReach/SetPropagation-FEM-JuliaCon21, presented at the JuliaCon'2021 Set Propagation Methods in Julia: Techniques and Applications. **Submitted to JuliaCon'21 Proceedings (extended abstract).**

Set propagation (step 2/2)

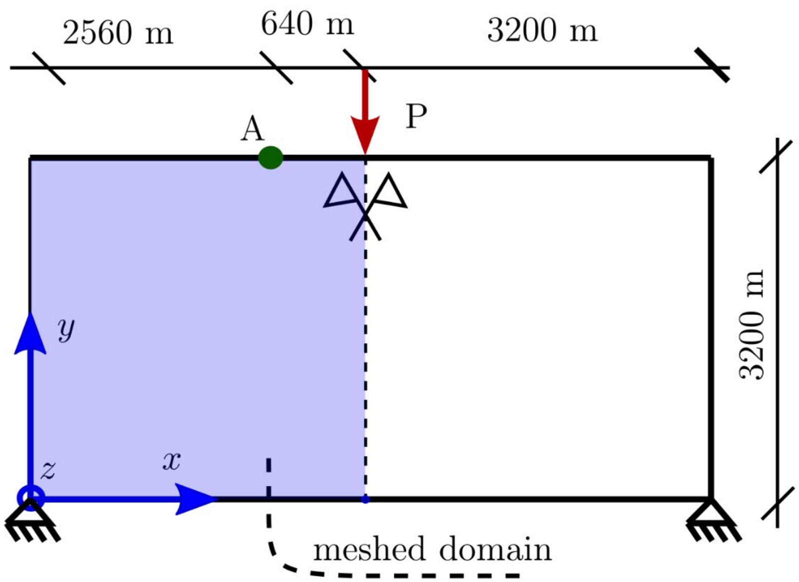




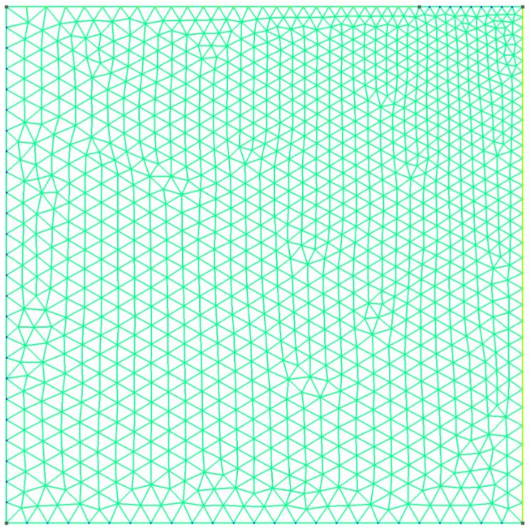
Efficient set computations require specialized algorithms based on different combinations of *set type representations* and *operations* involved.

See: *LazySets.jl: Scalable Symbolic-Numeric Set Computations.* Marcelo Forets and Christian Schilling. arXiv preprint arXiv:2110.01711 (2021). **Submitted to JuliaCon'2021 (full paper).**

Example in higher dimensions



(a) Diagram of domain and boundary conditions considered.



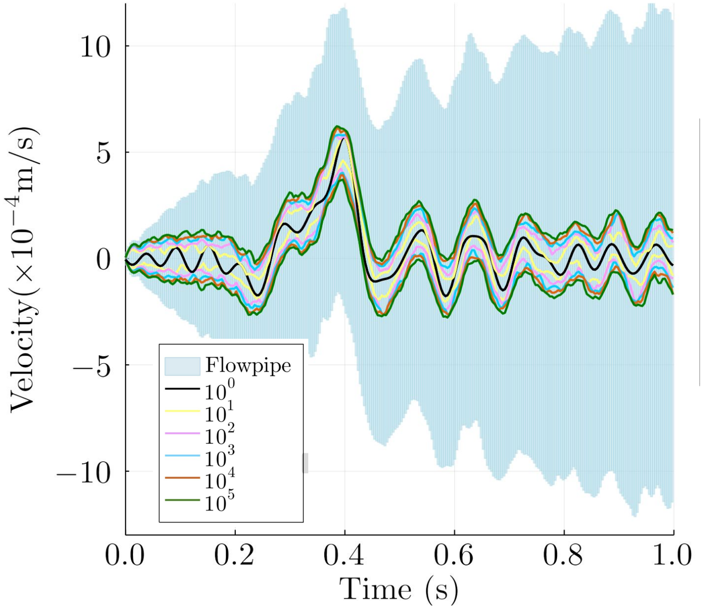
(b) Finite Element Method mesh used, formed by triangular elements.

2235 linear triangular elements

1180 nodes

See: *Combining Set Propagation with Finite Element Methods for Time Integration in Transient Solid Mechanics Problems*. Forets, Marcelo, Daniel F. Caporale, and Jorge M. Pérez Zerpa. arXiv preprint arXiv:2105.05841.
Accepted in Computers & Structures Journal (2021).

Example in higher dimensions

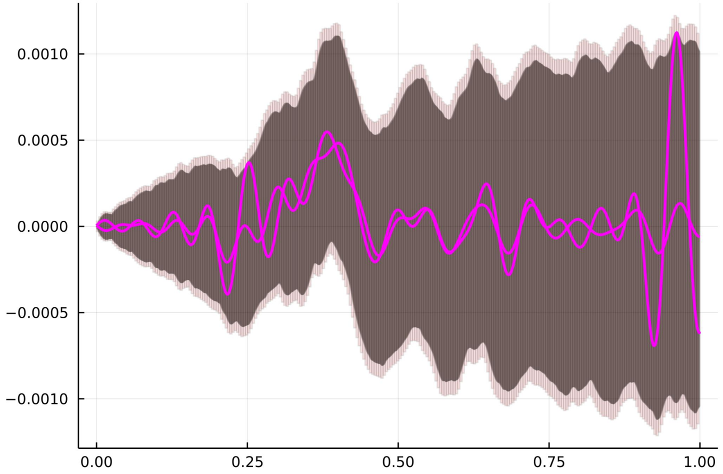


Method	# Trajectories	Time (s)	$\|v_{env}\|_{L_1}$ (10^{-5})	$\|v_{env}\|_{L_\infty}$ (10^{-5})
Newmark	1	0.3	9.27	56.98
Newmark	10	2.0	13.52	57.53
Newmark	100	17.7	16.61	57.59
Newmark	1000	175.5	18.52	58.22
Newmark	10000	1771.4	19.98	61.18
Newmark	100000	17796.1	21.42	62.21
Set Propagation	-	8.5	81.33	122.25

The initial set is propagated exponentially faster than using simulations over randomly sampled initial states

See: *Combining Set Propagation with Finite Element Methods for Time Integration in Transient Solid Mechanics Problems*. Forets, Marcelo, Daniel F. Caporale, and Jorge M. Pérez Zerpa. arXiv preprint arXiv:2105.05841 (2021).
Accepted in Computers & Structures Journal.

Example in higher dimensions



Method	# Trajectories	Time (s)	$\|v_{env}\|_{L_1}$ (10^{-5})	$\|v_{env}\|_{L_\infty}$ (10^{-5})
Newmark	1	0.3	9.27	56.98
Newmark	10	2.0	13.52	57.53
Newmark	100	17.7	16.61	57.59
Newmark	1000	175.5	18.52	58.22
Newmark	10000	1771.4	19.98	61.18
Newmark	100000	17796.1	21.42	62.21
Set Propagation	-	8.5	81.33	122.25

The sequence of sets obtained (flowpipe) converges to the exact reachable states when the time-step decreases

See: *Combining Set Propagation with Finite Element Methods for Time Integration in Transient Solid Mechanics Problems***.**
Forets, Marcelo, Daniel F. Caporale, and Jorge M. Pérez Zerpa. arXiv preprint arXiv:2105.05841 (2021).
Accepted in Computers & Structures Journal.

JuliaReach project

JuliaReach

To advance state-of-the-art working on *fundamental* problems

To build comprehensive, efficient, correct, reproducible, well documented libraries

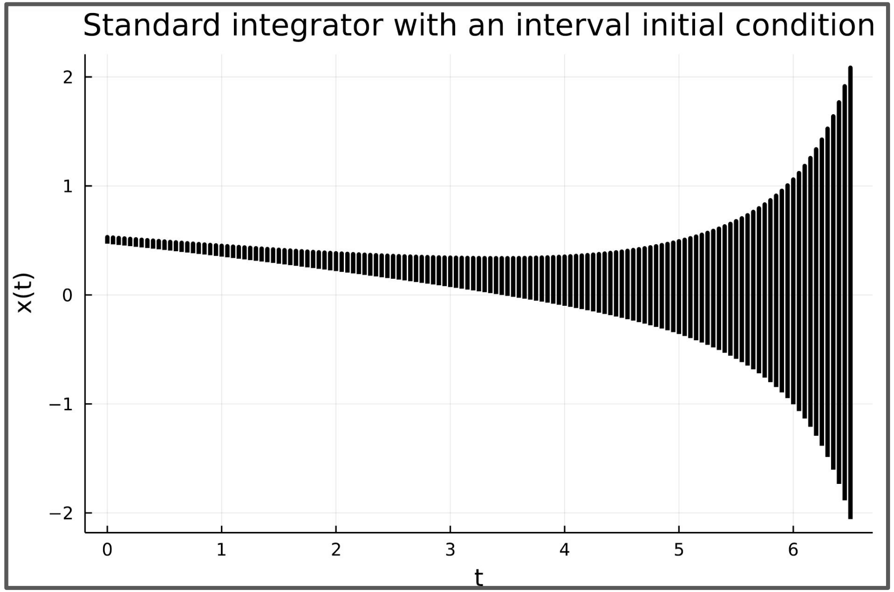
To widen the applicability of reachability analysis for scientists & engineers

Reachability for a nonlinear differential equation

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = rx(1 - x/K)$$

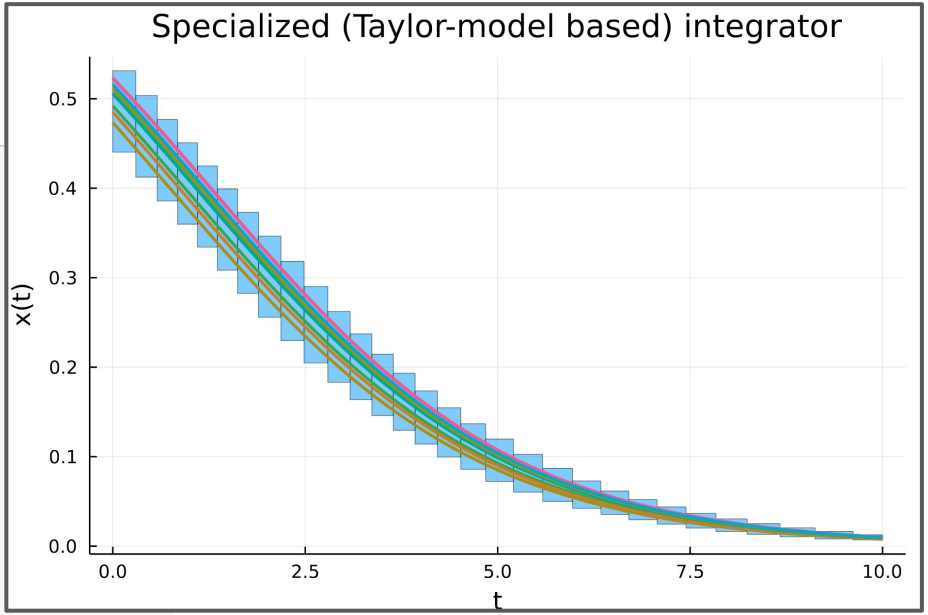
Logistic equation (population dynamics)

```
using DifferentialEquations,
      IntervalArithmetic

# define the problem
function f(dx, x, p, t)
    r = -0.5
    K = 0.8
    dx[1] = r*x[1]*(1 - x[1]/K)
end

x₀ = [0.47 .. 0.53]  # initial condition
prob = ODEProblem(f, x₀, (0.0, 6.5))
sol = solve(prob, RK4(), adaptive=false,
            dt=0.05, reltol=1e-6)
```

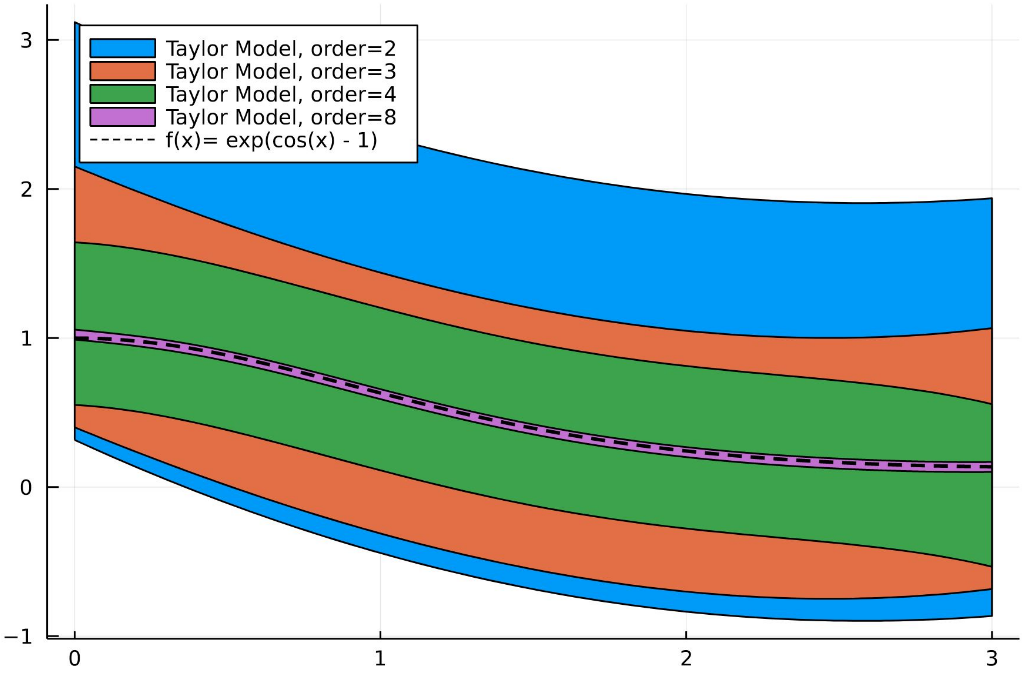


Reachability for a nonlinear differential equation

```julia
using ReachabilityAnalysis, Plots

# define the model (same as before)
function f(dx, x, p, t)
    r = -0.5
    K = 0.8
    dx[1] = r*x[1]*(1 - x[1]/K)
end

X0 = 0.47 .. 0.53 # interval initial states
# initial-value problem
prob = @ivp(x' = f(x), x(0) ∈ X0, dim=1)

# solve it
sol = solve(prob, alg=TMJets21a(abstol=1e-10), T=10.0);
```



Taylor models

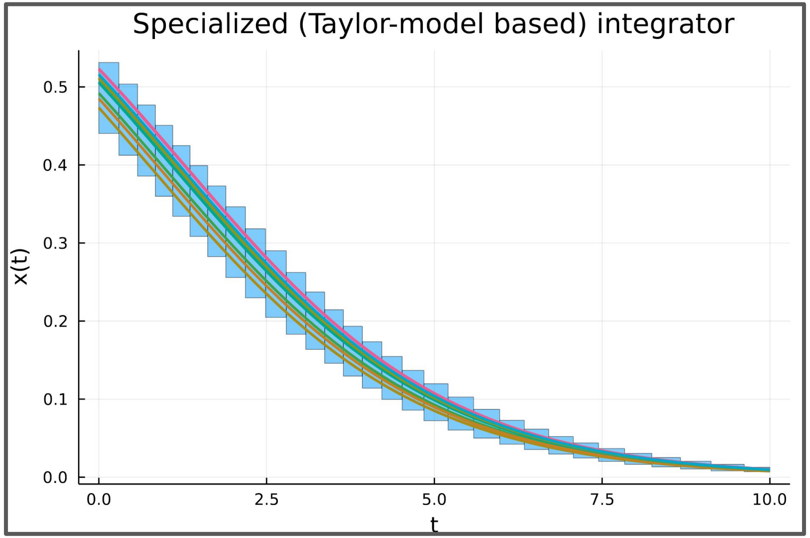


fT_3 = $0.39484468326455857 - 0.393855592043663\, t + 0.18246938526101011\, t^2 + 0.014258580485865105\, t^3 + [-0.946548, 0.803778]$

fT_8 = $0.39484468326455857 - 0.393855592043663\, t + 0.18246938526101011\, t^2 + 0.014258580485865105\, t^3 - 0.054727422711077306\, t^4 + 0.024706768559030104\, t^5 + 0.0007709727046973829\, t^6 - 0.00520692104599662\, t^7 + 0.0019470348640809945\, t^8 + [-0.0410741, 0.0258577]$

Reachability for a nonlinear differential equation

Example:
Taylor model reach-set at
the final computed interval



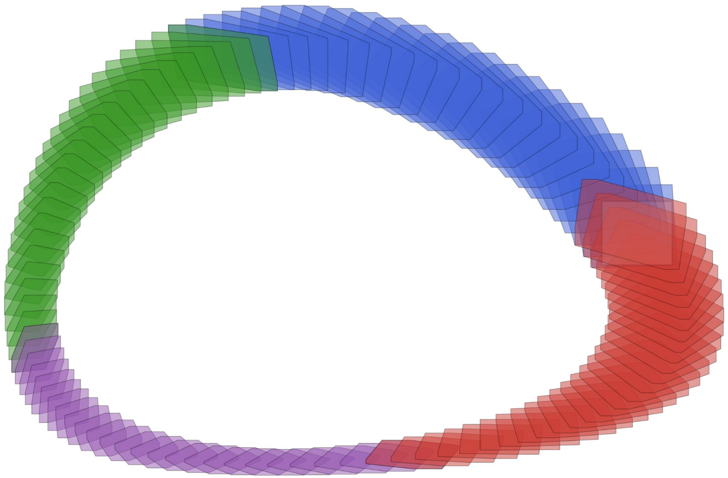

```
tspan(sol[end])
```

```
[9.61803, 10]
```

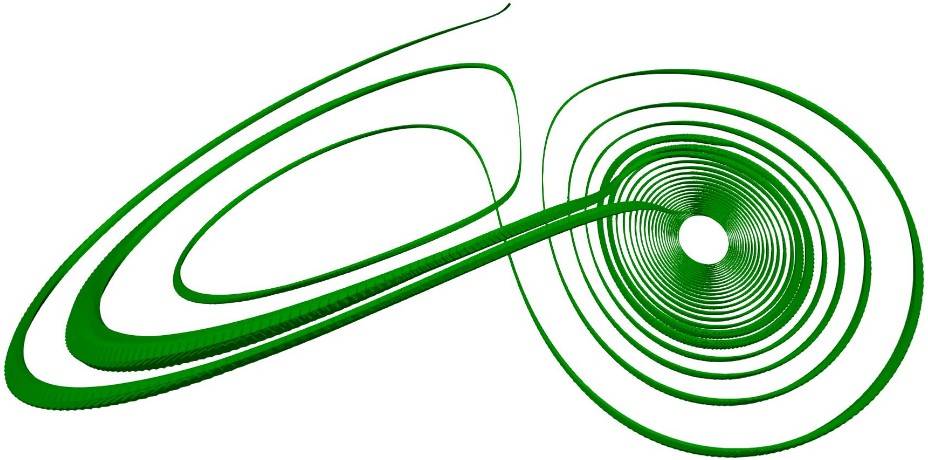
```
sol[end]
```

```
TaylorModelReachSet{Float64}(TaylorModels.TaylorModel1{TaylorN{Float64}, Float64}[  0.010728640000425796 + 0.0016935
616569102134 x₁ + 0.0001657222430356068 x₁² + ( - 0.0052923801775511875 - 0.0008240688117857138 x₁ - 7.8846059232215
5e-5 x₁²) t + ( 0.0012876075183438367 + 0.00019488963434660352 x₁ + 1.7762397657259186e-5 x₁²) t² + ( - 0.0002030100
35766464 - 2.888459368553032e-5 x₁ - 2.339217437712025e-6 x₁²) t³ + ( 2.2566088815570262e-5 + 2.752383693482914e-6 x
₁ + 1.474683203606484e-7 x₁²) t⁴ + ( - 1.7202398083315093e-6 - 1.1552580809843272e-7 x₁ + 1.1261821093858127e-8 x₁²)
t⁵ + ( 6.016969171459992e-8 - 1.413785130768101e-8 x₁ - 4.570779684951191e-9 x₁²) t⁶ + ( 6.311540762007901e-9 + 3.82
8591659395892e-9 x₁ + 7.063749944165518e-10 x₁²) t⁷ + ( - 1.4955436168686581e-9 - 4.920337196748677e-10 x₁ - 6.83032
0243727477e-11 x₁²) t⁸ + [-1.13178e-10, 1.0917e-10]], [9.61803, 10])
```

Nonlinear reachability



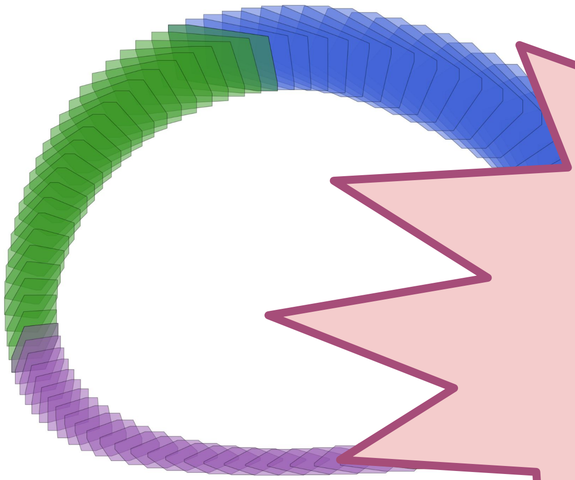
Van der pol system



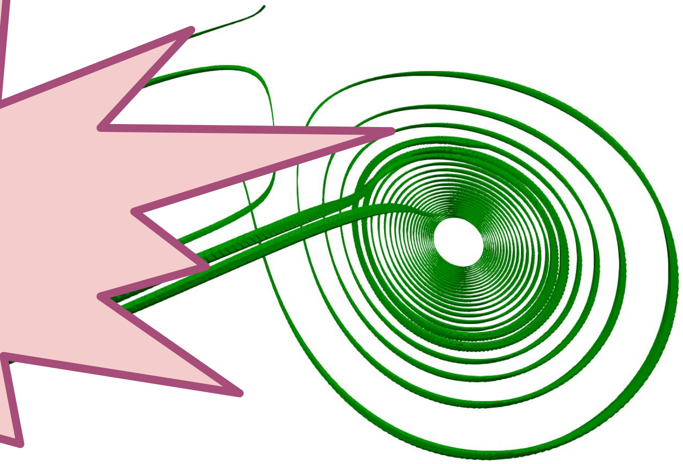
Lorenz system

Main nonlinear reachability approaches are: invariant generation, optimization based-approaches, solution-space abstractions, and state-space abstractions.

Nonlinear reachability



Open problem: nonlinear systems in high dimensions!



Van der pol system

Lorenz system

Main nonlinear reachability approaches are: invariant generation, optimization based-approaches, solution-space abstractions, and state-space abstractions.

Kronecker products, polynomial ODEs and weak nonlinearity

$$x^{\otimes 2} = x \otimes x = (x_1^2, x_1 x_2, x_2 x_1, x_2^2)^T$$

$$x^{\otimes i} := \underbrace{x \otimes \cdots \otimes x}_{i \text{ times}}, \qquad x \in \mathbb{R}^n$$

$$A \otimes B := \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{pmatrix}$$

Kronecker products, polynomial ODEs and weak nonlinearity

$$x^{\otimes 2} = x \otimes x = (x_1^2, x_1 x_2, x_2 x_1, x_2^2)^T$$

$$x^{\otimes i} := \underbrace{x \otimes \cdots \otimes x}_{i \text{ times}}, \qquad x \in \mathbb{R}^n$$

$$A \otimes B := \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{pmatrix}$$

"linear behavior"

"nonlinear behavior"

$$\boxed{\frac{\mathrm{d}x(t)}{\mathrm{d}t} = F_1 x + F_2 x^{\otimes 2}}$$

$$\|F_2\|_2 / \|F_1\|_2$$

if this ratio is small

the system is said to be <u>weakly nonlinear</u>

Kronecker products, polynomial ODEs and weak nonlinearity

$$x^{\otimes 2} = x \otimes x = (x_1^2, x_1 x_2, x_2 x_1, x_2^2)^T$$

$$x^{\otimes i} := \underbrace{x \otimes \cdots \otimes x}_{i \text{ times}}, \qquad x \in \mathbb{R}^n$$

$$A \otimes B := \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{pmatrix}$$

"linear behavior"

"nonlinear behavior"

$$\boxed{\frac{\mathrm{d}x(t)}{\mathrm{d}t} = F_1 x + F_2 x^{\otimes 2}}$$

General for polynomial ODEs with f(0) = 0; formulation and linearization goes back to Carleman (1932)

Example: logistic equation

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = rx(1 - x/K)$$

Bring to general quadratic form

$$x'(t) = ax(t) + bx^2(t)$$

$$a = r \text{ and } b = -r/K$$

Introduce auxiliary variables

$$\hat{y}_1' = x' = a\hat{y}_1 + b\hat{y}_2$$

$$\hat{y}_2' = 2x'x = 2a\hat{y}_2 + 2b\hat{y}_3$$

$$\hat{y}_j' = ja\hat{y}_j + jb\hat{y}_{j+1}, \qquad j \in \mathbb{N}$$

Truncate to finite order N (here: N = 4)

$$\frac{\mathrm{d}\hat{y}(t)}{\mathrm{d}t} = \begin{pmatrix} a & b & 0 & 0 \\ 0 & 2a & 2b & 0 \\ 0 & 0 & 3a & 3b \\ 0 & 0 & 0 & 4a \end{pmatrix} \hat{y}, \qquad \hat{y}(0) = \begin{pmatrix} x_0 \\ x_0^2 \\ x_0^3 \\ x_0^4 \end{pmatrix}$$

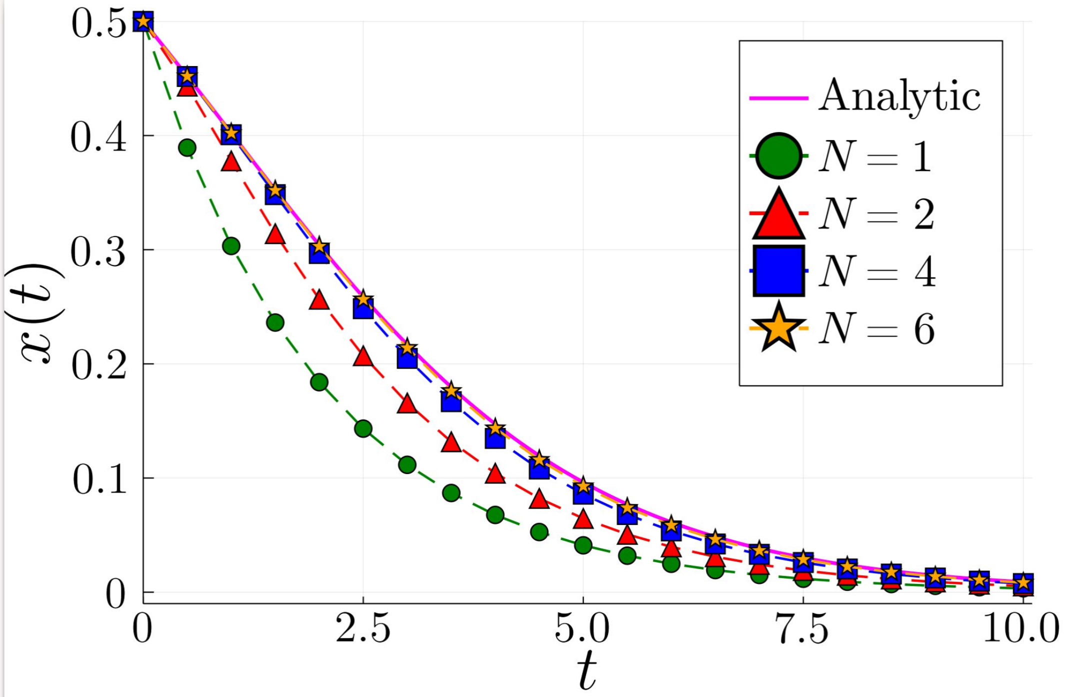
Example: logistic equation

$$\frac{\mathrm{d}\hat{y}(t)}{\mathrm{d}t} = \begin{pmatrix} a & b & 0 & 0 \\ 0 & 2a & 2b & 0 \\ 0 & 0 & 3a & 3b \\ 0 & 0 & 0 & 4a \end{pmatrix} \hat{y}, \qquad \hat{y}(0) = \begin{pmatrix} x_0 \\ x_0^2 \\ x_0^3 \\ x_0^4 \end{pmatrix}$$

The analytic solution in this case is:

$$x(t) = \frac{x_0 e^{at}}{1 + (b/a)(1 - e^{at})x_0}$$

b/a controls the "degree" of nonlinearity of the ODE



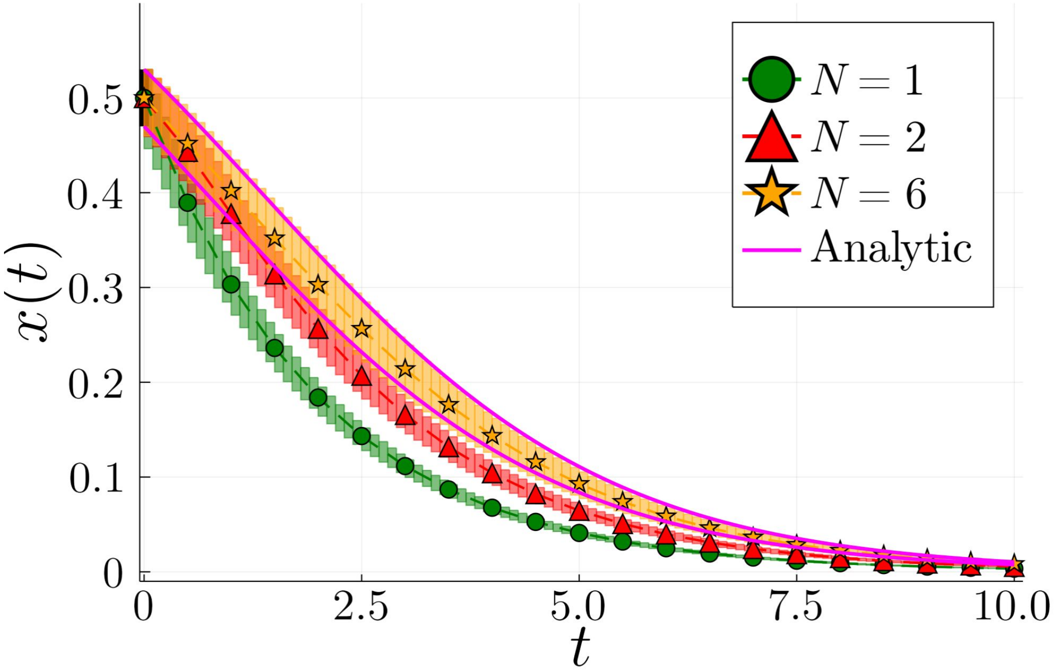
Example: logistic equation (distributed initial conditions)

$$\frac{\mathrm{d}\hat{y}(t)}{\mathrm{d}t} = \begin{pmatrix} a & b & 0 & 0 \\ 0 & 2a & 2b & 0 \\ 0 & 0 & 3a & 3b \\ 0 & 0 & 0 & 4a \end{pmatrix} \hat{y}, \qquad \hat{y}(0) = \begin{pmatrix} x_0 \\ x_0^2 \\ x_0^3 \\ x_0^4 \end{pmatrix}$$

The analytic solution in this case is:

$$x(t) = \frac{x_0 e^{at}}{1 + (b/a)(1 - e^{at})x_0}$$

b/a controls the “degree” of nonlinearity of the ODE



Carleman embedding: general case

$$x'(t) = f(x(t)),\ f : \mathbb{R}^n \to \mathbb{R}^n$$

Bring to general quadratic form

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = F_1 x + F_2 x^{\otimes 2} \qquad (1)$$

Introduce auxiliary variables

$$\hat{y}_j := x^{\otimes j},\ j \in \mathbb{N}$$

Truncate to finite order N

$$A = \begin{pmatrix} A_1^1 & A_2^1 & 0 & 0 & \cdots & 0 \\ 0 & A_2^2 & A_3^2 & 0 & \cdots & 0 \\ 0 & 0 & A_3^3 & A_4^3 & 0 & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & A_{N-1}^{N-1} & A_N^{N-1} \\ 0 & 0 & \cdots & 0 & 0 & A_N^N \end{pmatrix}$$

$$A_{i+i'-1}^i = \sum_{\nu=1}^{i} \overbrace{\mathbb{I}_n \otimes \cdots \otimes \underset{\substack{\uparrow \\ \nu\text{-th position}}}{F_{i'}} \otimes \cdots \otimes \mathbb{I}_n}^{\text{i factors}}$$

$$\boxed{\frac{\mathrm{d}\hat{y}}{\mathrm{d}t} = A\hat{y}, \qquad \hat{y}(0) = \hat{y}_0} \qquad (2)$$

Error bounds

Definition 1. *System* $\boxed{1}$ *is said to be* weakly nonlinear *if the ratio*

$$R := \frac{\|x_0\|\|F_2\|}{|\Re(\lambda_1)|} \tag{5}$$

satisfies $R < 1$.

Definition 2. *System* $\boxed{1}$ *is said to be* dissipative *if* $\Re(\lambda_1) < 0$ *(i.e., the real part of all eigenvalues is negative).*

The conditions $\Re(\lambda_1) < 0$ and $R < 1$ ensure arbitrary-time convergence.

Error bounds

Theorem 1 ([30], Corollary 1]). *Assuming that (1) is weakly nonlinear and dissipative, the error bound associated with the linearized problem (2) truncated at order N satisfies*

$$\|\eta_1(t)\| \leq \varepsilon(t) := \|x_0\| R^N (1 - e^{\Re(\lambda_1)t})^N, \tag{6}$$

with R as defined in (5). This error bound holds for all $t \geq 0$.

[30] Liu, J.P., Kolden, H.Ø., Krovi, H.K., Loureiro, N.F., Trivisa, K., Childs, A.M.:
Efficient quantum algorithm for dissipative nonlinear differential equations. Proceedings of the National Academy of Sciences 118(35) (2021)

Reachability algorithm using Carleman linearization

Algorithm 1: Reachability algorithm

Input: $\mathcal{X}_0 \subseteq \mathbb{R}^n$: hyperrectangular initial states; F_1, F_2: system matrices; N: truncation order; T: time horizon; *post*: algorithm to compute a flowpipe for linear systems

Output: flowpipe overapproximating the reachable states until T

$A, \hat{\mathcal{X}}_0 \leftarrow \mathit{linearize}(\mathcal{X}_0, F_1, F_2, N)$; // Carleman linearization

$(\mathcal{R}_0, \ldots, \mathcal{R}_m) \leftarrow \mathit{post}(y' = Ay, y(0) \in \hat{\mathcal{X}}_0)$; // flowpipe for linear system

for $j \leftarrow 0$ **to** m **do**

$\quad \varepsilon \leftarrow \mathit{error}(\mathcal{R}_j, \mathcal{X}_0, F_1, F_2, N)$; // linearization error

$\quad \mathcal{R}_j^\varepsilon \leftarrow \pi_{1:n}(\mathcal{R}_j) \oplus \mathcal{B}_\varepsilon^n$; // enlarged reach set

end

return $(\mathcal{R}_0^\varepsilon, \ldots, \mathcal{R}_m^\varepsilon)$

Numerical results: Burgers' PDE

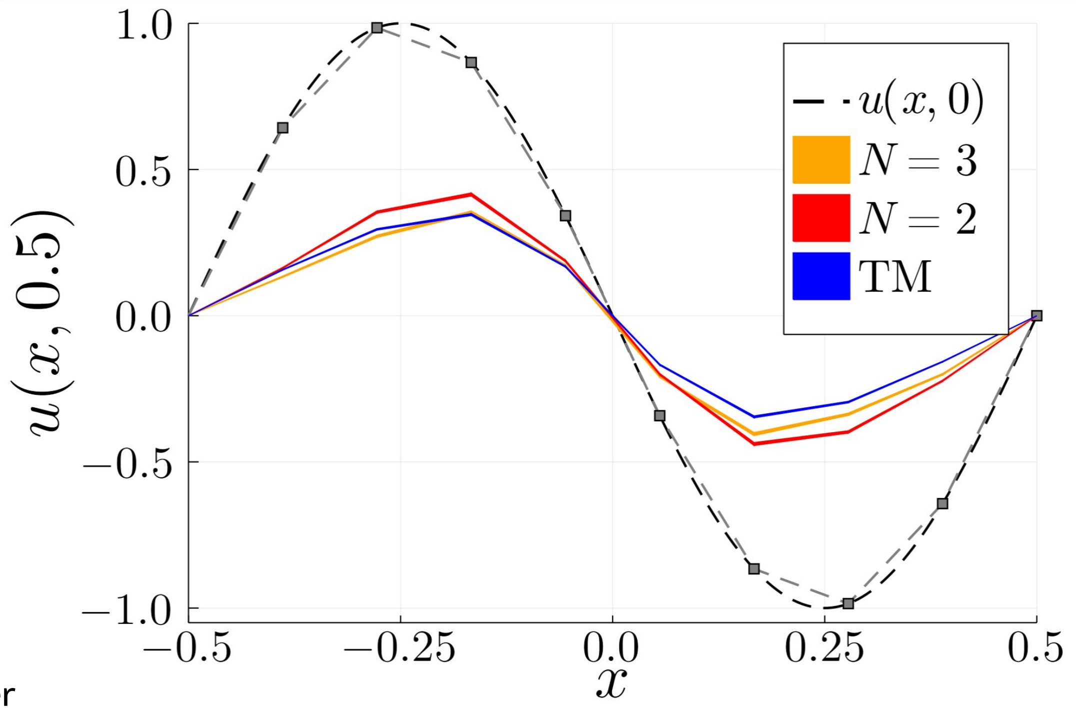
Continuous partial differential equation (space and time)

$$\partial_t u + x \partial_x u = \nu \partial_x^2 u$$

Discretize (space)

$$\partial_t u_i = \nu \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} - \frac{u_{i+1}^2 - u_{i-1}^2}{4\Delta x}$$

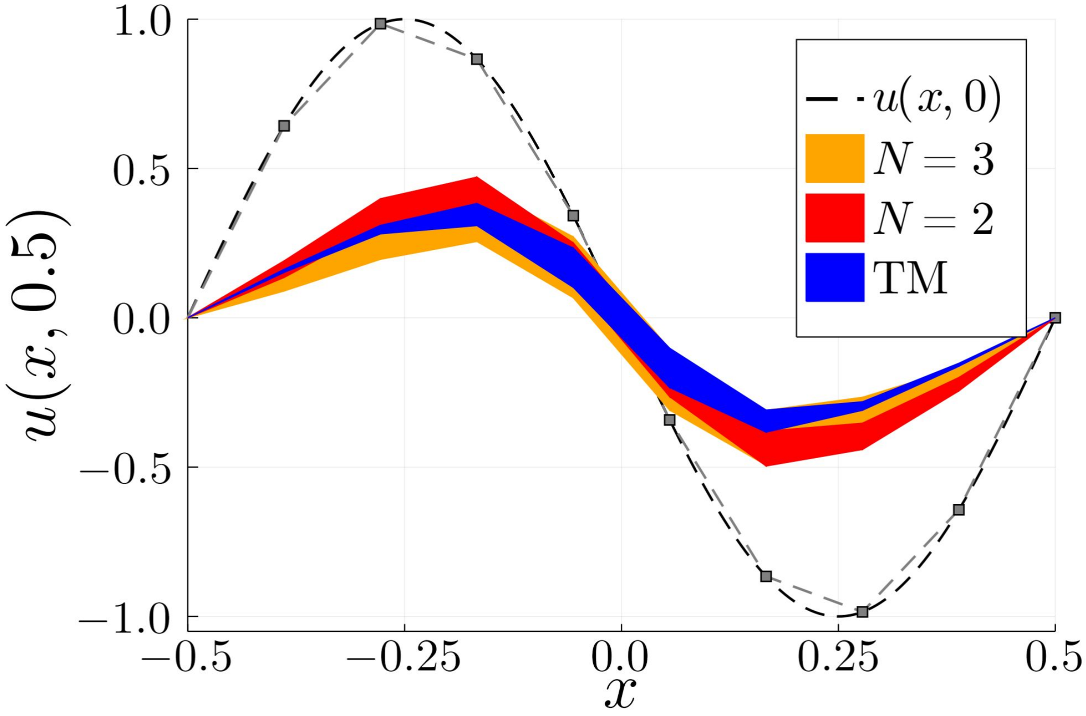
Reachability solver (integrate in time)



Numerical results: Burgers' PDE (distributed initial conditions)

$$\Re(\lambda_1) \approx -0.488 < 0$$

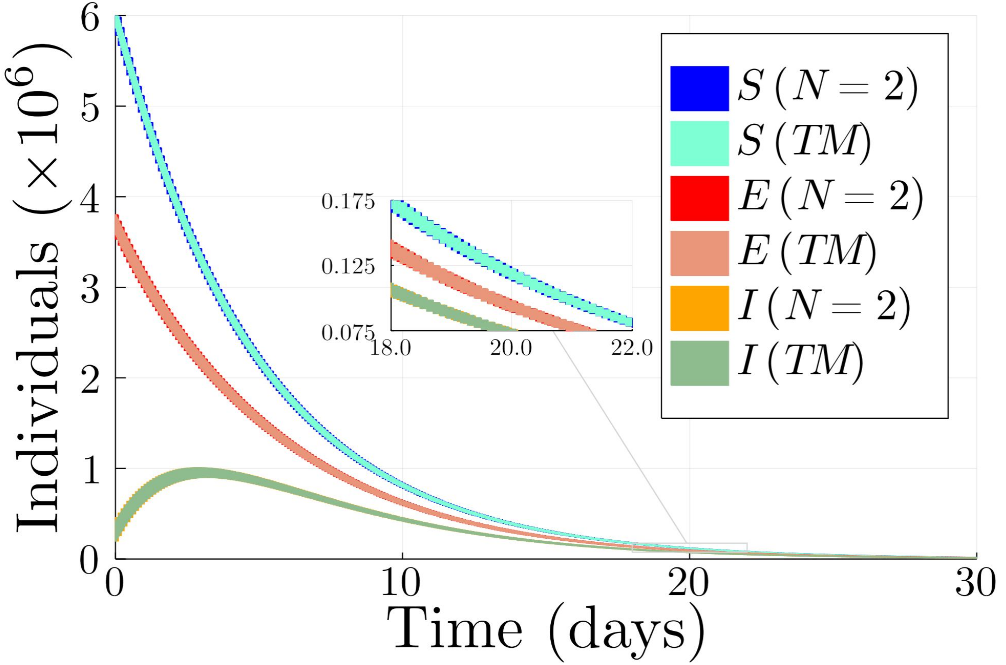
$$R \approx 18.58$$



Numerical results: Epidemic model (SEIR)

$$\frac{\mathrm{d}P_S}{\mathrm{d}t} = -\Lambda\frac{P_S}{P} - r_{\mathrm{vac}}P_S - r_{\mathrm{tra}}P_S\frac{P_I}{P} + \Lambda$$
$$\frac{\mathrm{d}P_E}{\mathrm{d}t} = -\Lambda\frac{P_E}{P} - \frac{P_E}{T_{\mathrm{lat}}} + r_{\mathrm{tra}}P_S\frac{P_I}{P}$$
$$\frac{\mathrm{d}P_I}{\mathrm{d}t} = -\Lambda\frac{P_I}{P} + \frac{P_E}{T_{\mathrm{lat}}} - \frac{P_I}{T_{\mathrm{inf}}}$$
$$\frac{\mathrm{d}P_R}{\mathrm{d}t} = -\Lambda\frac{P_R}{P} + r_{\mathrm{vac}}P_S + \frac{P_I}{T_{\mathrm{inf}}}$$

Data from the early spread of the COVID-19 disease from [P20]



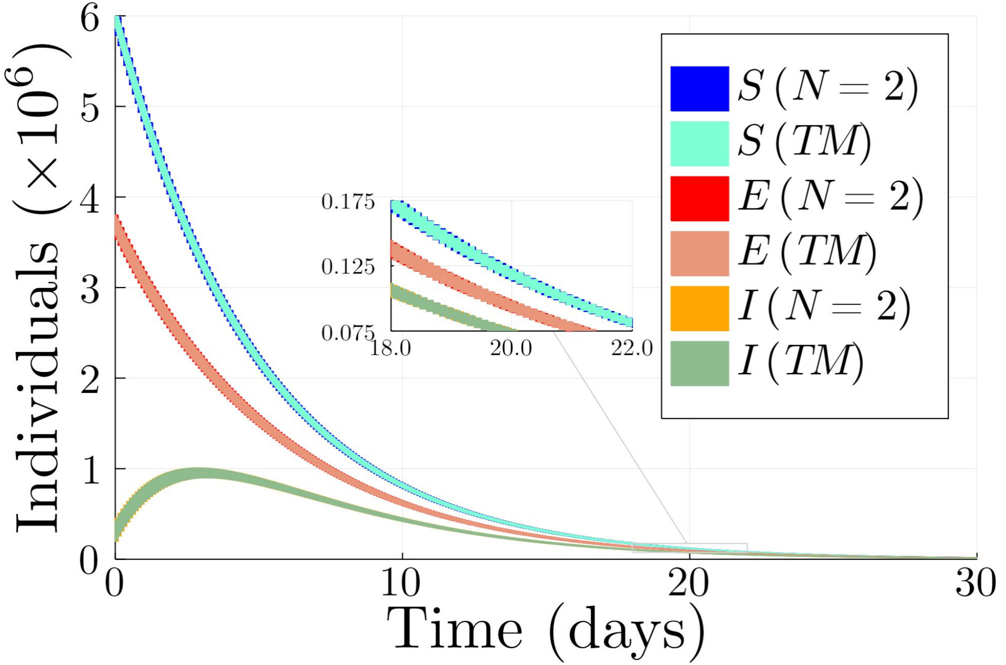
[P20] Pan, A. et al: Association of Public Health Interventions With the Epidemiology of the COVID-19 Outbreak in Wuhan, China. JAMA 323(19), 1915–1923 (2020)

Numerical results: Epidemic model (SEIR)

$$F_1 = \begin{pmatrix} -\frac{\Lambda}{P} - r_{\text{vac}} & 0 & 0 \\ 0 & -\frac{\Lambda}{P} - \frac{1}{T_{\text{lat}}} & 0 \\ 0 & \frac{1}{T_{\text{lat}}} & -\frac{\Lambda}{P} - \frac{1}{T_{\text{inf}}} \end{pmatrix}$$

$$F_2 = \begin{pmatrix} 0 & 0 & -\frac{r_{\text{tra}}}{P} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{r_{\text{tra}}}{P} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$R \approx 0.68 \text{ and } \Re(\lambda_1) \approx -0.19$$



[P20] Pan, A. et al: Association of Public Health Interventions With the Epidemiology of the COVID-19 Outbreak in Wuhan, China. JAMA 323(19), 1915–1923 (2020)

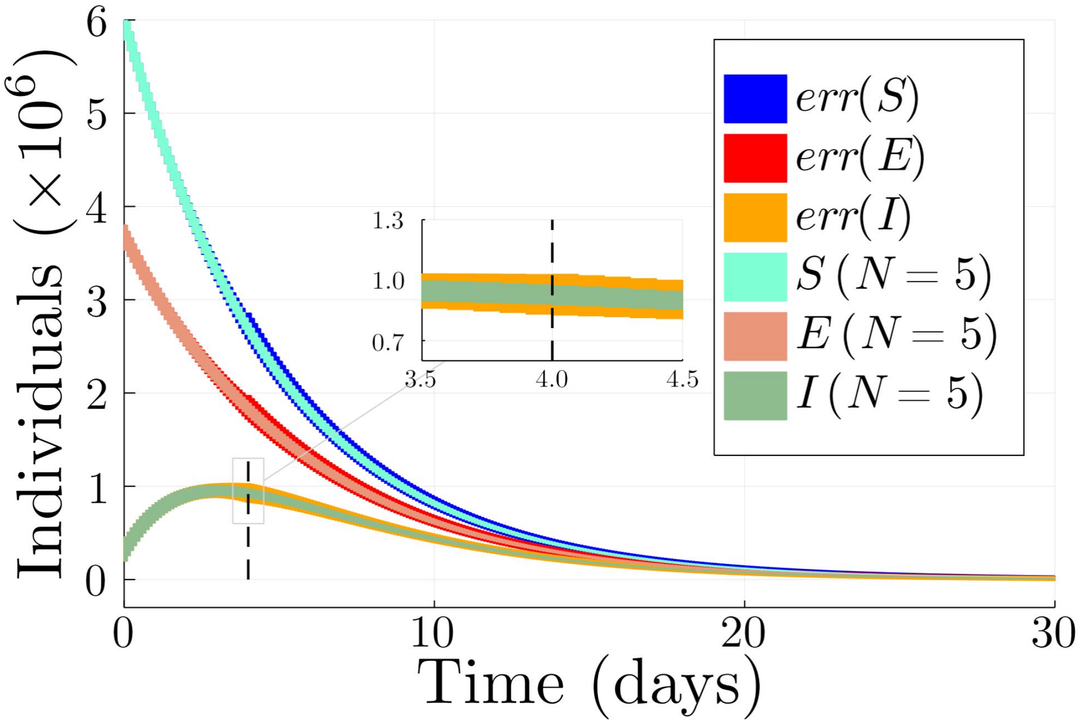
Numerical results: Run-time comparison

	SEIR model		Burgers model	
	no error bound	incl. error bound	initial point	initial set
TM	6.14 s		0.88 s	0.91 s
Carleman	$N = 2$: 0.006 s	$N = 5$: 0.185 s	$N = 2$: 0.0065 s $N = 3$: 0.24 s	$N = 2$: 0.0067 s $N = 3$: 0.29 s

Table 1: Run times for the SEIR model and the Burgers model obtained for the Taylor-model (TM) approach and the Carleman linearization with different truncation orders N.

Idea: time-triggered resets

$$\|\eta_1(t)\| \leq \varepsilon(t) := \|x_0\| R^N (1 - e^{\Re(\lambda_1)t})^N$$



Problem: correction including error term may not converge to zero, even if the true solution does

Idea: re-evaluate the error term at times t* > 0 => reduce error estimate

Question (open): Auomate / optimize when resets happen

Conclusions

We presented a novel reachability method for polynomial ODEs that can leverage established and well-optimized **high-dimensional reachability solvers**

Technique: abstract nonlinear terms into a higher-dimensional space such that the evolution is **approximately linear**

Outperforms other state-of-the-art methods in the set of examples, but requires **weak nonlinearity** and **dissipativity** (for guaranteed bounds)

Still, if the assumptions do not hold, the method returns **relatively good results** (but without guaranteed bounds)

Ideas for possible extensions

Consider time-varying matrices instead of fixed matrices.

Automation of the time-triggered process

How often and when to apply the resets?

Algorithmic specialization for Kronecker products, e.g. Krylov subspace iteration

Can we exploit the structure of the *linearized system*?

Automation and specialization of the quadratic formulation [BP21]

[BP21] Bychkov, Andrey, and Gleb Pogudin. "Optimal monomial quadratization for ODE systems." *ACM Communications in Computer Algebra* 54.3 (2021): 119-123.