Nonsmooth numerical optimal control

Armin Nurkanović

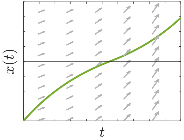
Systems Control and Optimization Laboratory
Department of Microsystems Engineering
University of Freiburg, Germany

based on joint work with
Moritz Diehl, Jonathan Frey, Mario Sperl, Sebastian Albrecht, Bernard Brogliato

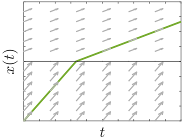
2nd Seasonal School of the ELO-X project
University of Freiburg
6th – October, 2022

Nonsmooth Dynamics (NSD) - a classification

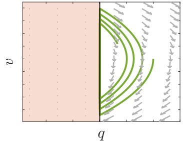
Regard ordinary differential equation (ODE) with a **nonsmooth** right-hand side (RHS).
Distinguish three cases:



NSD1: non-differentiable RHS, e.g., $\dot{x} = 1 + |x|$

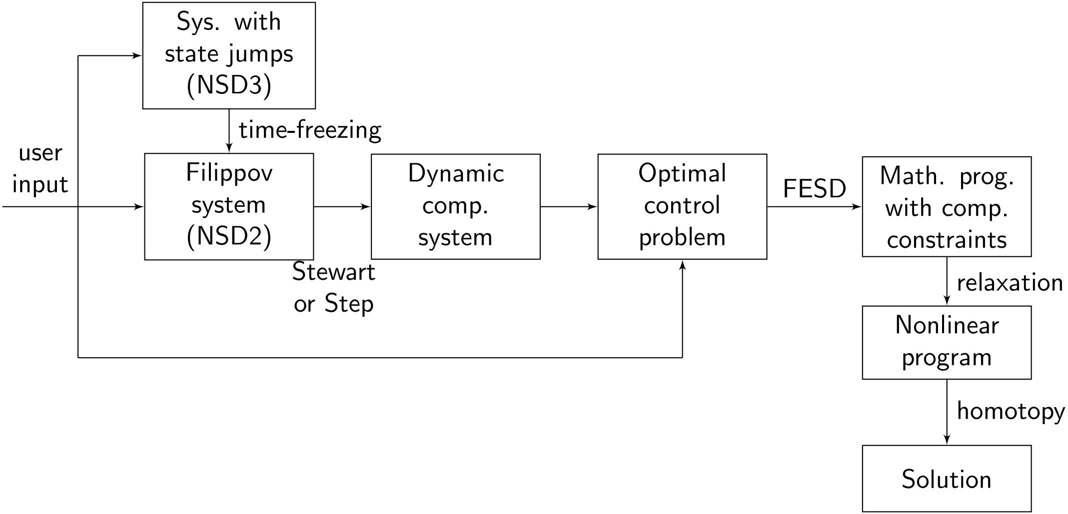


NSD2: state dependent switch of RHS, e.g., $\dot{x} = 2 - \mathrm{sign}(x)$

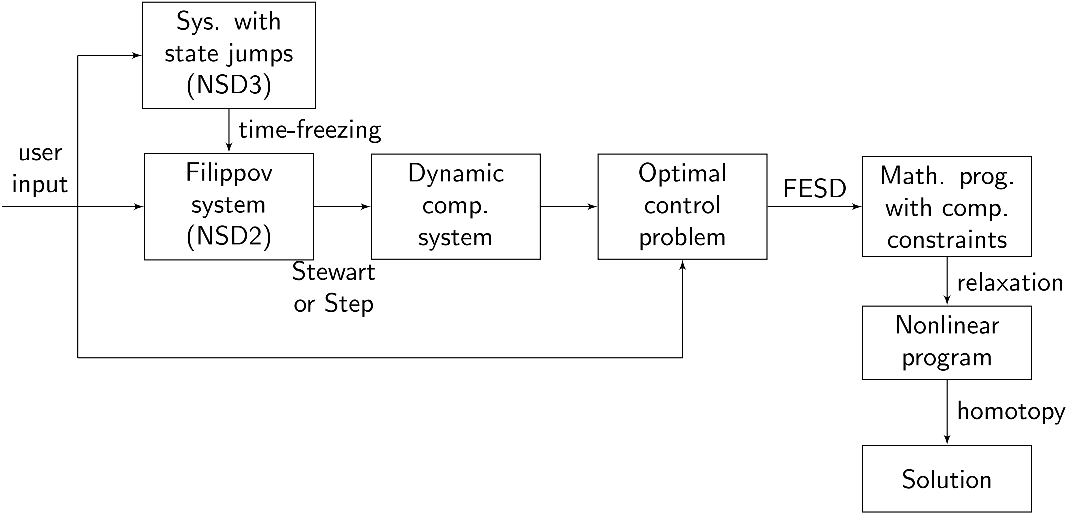


NSD3: state dependent jump, e.g., bouncing ball, $v(t_+) = -0.9\, v(t_-)$

Nonsmooth numerical optimal control - overview



Nonsmooth numerical optimal control - overview



Toolchain implemented in our open-source package NOSNOC

NOSNOC: NOnSmooth Numerical Optimal Control

Open-source package based on MATLAB, CasADi and IPOPT

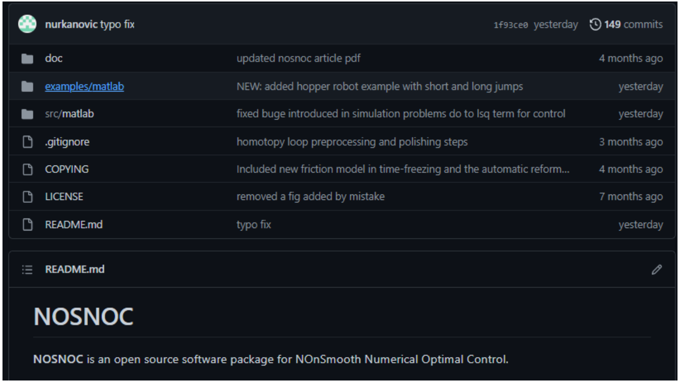
Key features

1. automatic reformulation of systems with state jumps into switched systems via the time-freezing reformulation
2. automatic discretization of the OCP via FESD (high accuracy)
3. solution methods for the resulting discrete-time OCP via continuous optimization in a homotopy (no integers)

NOSNOC: `https://github.com/nurkanovic/nosnoc`

NOSNOC: NOnSmooth Numerical Optimal Control

Open-source package based on MATLAB, CasADi and IPOPT



NOSNOC: `https://github.com/nurkanovic/nosnoc`

NOSNOC: NOnSmooth Numerical Optimal Control

Open-source package based on MATLAB, CasADi and IPOPT

nurkanovic NEW: added hopper robot example with short and long jumps — f81e7d0 yesterday — History

..		
a_simple_tutorial	fixed a minor bug due to a experimental option that is not avilable a...	20 days ago
cart_pole_with_friction	added support for least square cost terms with time variable refrences	2 days ago
different_switching_cases	changed some header info and removed outdated functions	5 months ago
friction_example_stewart	fixed small bug with autogenerating models for elastic mechanical imp...	2 months ago
hopper_robot_ocp	NEW: added hopper robot example with short and long jumps	yesterday
ivp_problem	support for automatic reformulation of time-freezing, several sanity ...	5 months ago
ocp_motor_with_friction	Added example from Optimal Control of a Voice–Coil–Motor with Coulomb...	2 months ago
oscilator_integrator_order	fixed some minor bugs in the vectorized version of create_nlp	2 months ago
robot_ocp	modify plotting script	3 days ago
sliding_mode_ocp	polishing of some aux. scripts	3 days ago
three_cart_problem	made more generic gif generation for the three cart example	yesterday
time_freezing_hystheresis	fixed some minor bugs in the vectorized version of create_nlp	2 months ago
time_freezing_inelastic_tutorial	minor edits in example	2 months ago
time_freezing_throwing_ball	implemented vectorized version of create_nlp_nosnoc	2 months ago
time_optimal_car_benchmark	changed some header info and removed outdated functions	5 months ago

NOSNOC: `https://github.com/nurkanovic/nosnoc`

NOSNOC: NOnSmooth Numerical Optimal Control

Open-source package based on MATLAB, CasADi and IPOPT

nurkanovic NEW: added hopper robot example with short and long jumps		f81e7d0 yesterday History
..		
a_simple_tutorial	fixed a minor bug due to a experimental option that is not avilable a...	20 days ago
cart_pole_with_friction	added support for least square cost terms with time variable refrences	2 days ago
different_switching_cases	changed some header info and removed outdated functions	5 months ago
friction_example_stewart	fixed small bug with autogenerating models for elastic mechanical imp...	2 months ago
hopper_robot_ocp	NEW: added hopper robot example with short and long jumps	yesterday
ivp_problem	support for automatic reformulation of time-freezing, several sanity ...	5 months ago
ocp_motor_with_friction	Added example from Optimal Control of a Voice–Coil–Motor with Coulomb...	2 months ago
oscilator_integrator_order	fixed some minor bugs in the vectorized version of create_nlp	2 months ago
robot_ocp	modify plotting script	3 days ago
sliding_mode_ocp	polishing of some aux. scripts	3 days ago
three_cart_problem	made more generic gif generation for the three cart example	yesterday
time_freezing_hystheresis	fixed some minor bugs in the vectorized version of create_nlp	2 months ago
time_freezing_inelastic_tutorial	minor edits in example	2 months ago
time_freezing_throwing_ball	implemented vectorized version of create_nlp_nosnoc	2 months ago
time_optimal_car_benchmark	changed some header info and removed outdated functions	5 months ago

NOSNOC: `https://github.com/nurkanovic/nosnoc`

Videos and gifs.

NSD2 - Piecewise smooth / Filippov systems

Motivating examples - crossing a discontinuity

Consider the ODE

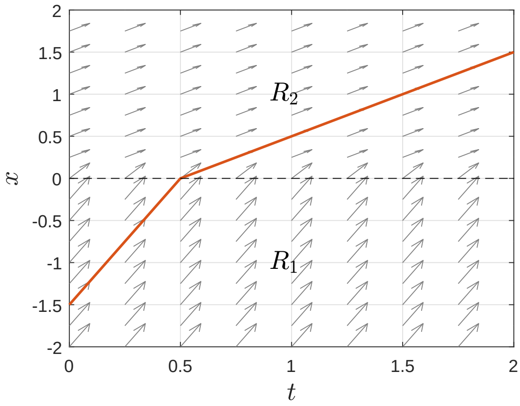
$$\dot{x} = 2 - \mathrm{sign}(x)$$

Consider the ODE

$$\dot{x} = 2 - \operatorname{sign}(x)$$

More explicitly...

$$\dot{x} = \begin{cases} 3, & \text{if } x < 0 \\ 1, & \text{if } x > 0 \end{cases}$$



Motivating examples - sliding mode (simpler)

Consider the ODE

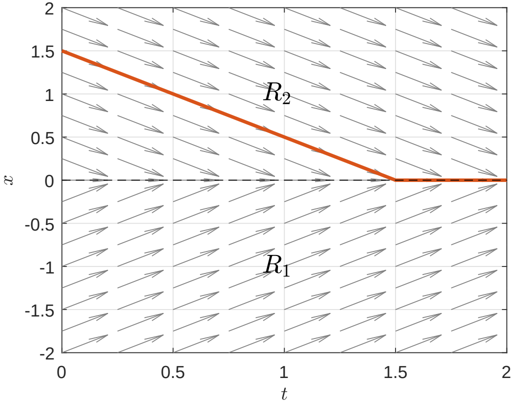
$$\dot{x} = -\text{sign}(x)$$

And let

$$\text{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

Then...

$$\dot{x} = \begin{cases} 1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ -1, & \text{if } x > 0 \end{cases}$$



Motivating examples - sliding mode (simpler)

Consider the ODE

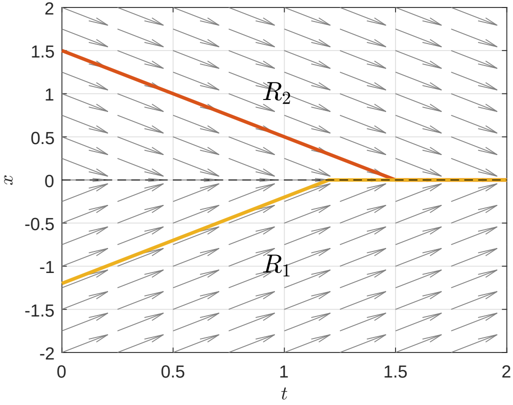
$$\dot{x} = -\text{sign}(x)$$

And let

$$\text{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

Then...

$$\dot{x} = \begin{cases} 1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ -1, & \text{if } x > 0 \end{cases}$$



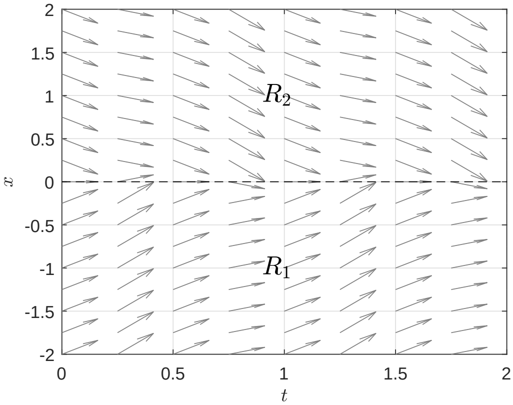
Motivating examples - sliding mode

Consider the ODE

$$\dot{x} = -\mathrm{sign}(x) + 0.5\sin(t)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$



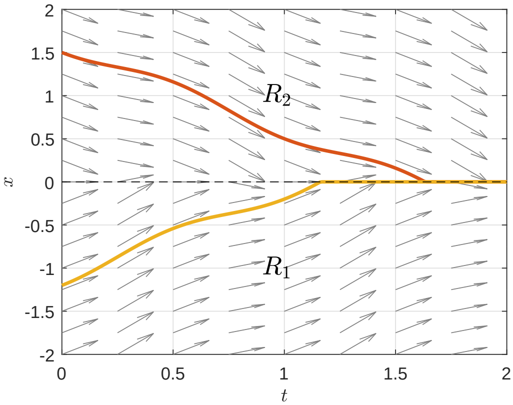
Motivating examples - sliding mode

Consider the ODE

$$\dot{x} = -\mathrm{sign}(x) + 0.5\sin(t)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$



Motivating examples - sliding mode

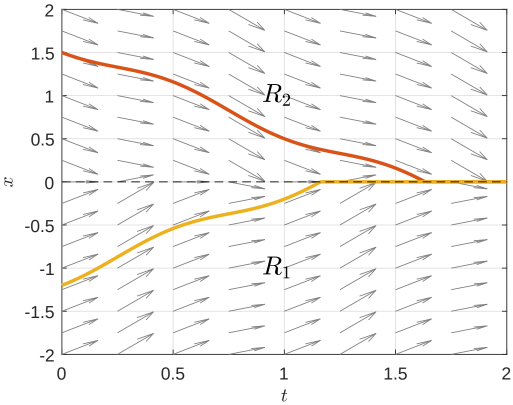
Consider the ODE

$$\dot{x} = -\mathrm{sign}(x) + 0.5\sin(t)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

We have for some $t > t^*$ that $x(t) = 0$ and $\dot{x}(t) = 0$



Motivating examples - sliding mode

Consider the ODE

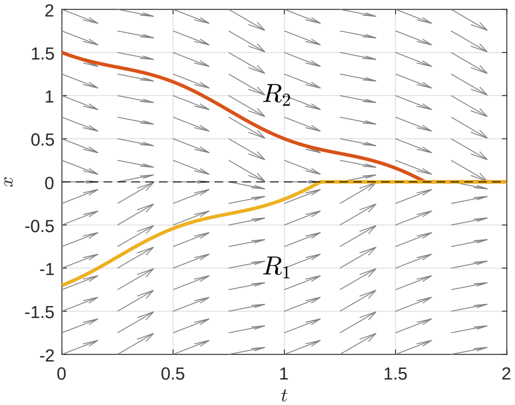
$$\dot{x} = -\mathrm{sign}(x) + 0.5\sin(t)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

We have for some $t > t^*$ that $x(t) = 0$ and $\dot{x}(t) = 0$

That is $\mathrm{sign}(0) = 0 = 0.5\sin(t)$



Motivating examples - sliding mode

Consider the ODE

$$\dot{x} = -\mathrm{sign}(x) + 0.5\sin(t)$$

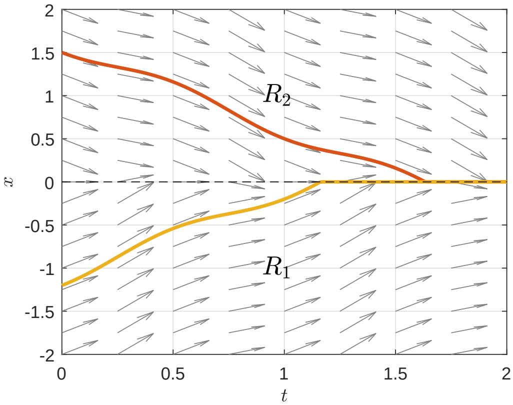
And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

We have for some $t > t^*$ that $x(t) = 0$ and $\dot{x}(t) = 0$

That is $\mathrm{sign}(0) = 0 = 0.5\sin(t)$

WHAT HAPPEND?



Motivating examples - sliding mode - fixed

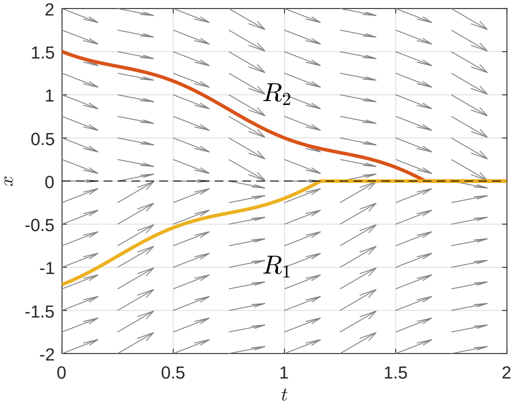
Consider the ODE

$$\dot{x} \in -\mathrm{sign}(x) + 0.5\sin(t)$$

And let

$$\mathrm{sign}(x) \in \begin{cases} \{-1\}, & \text{if } x < 0 \\ [-1, 1], & \text{if } x = 0 \\ \{1\}, & \text{if } x > 0 \end{cases}$$

We have for some $t > t^*$ that $x(t) = 0$ and $\dot{x}(t) = 0$



Motivating examples - sliding mode - fixed

Consider the ODE

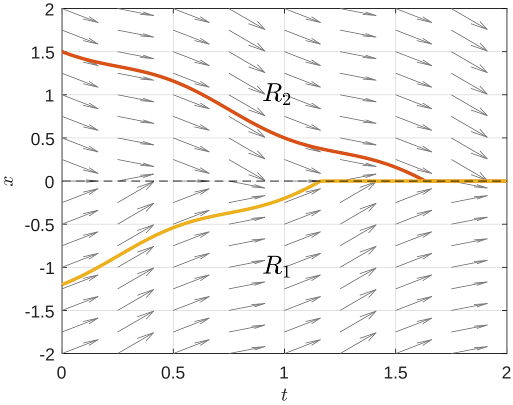
$$\dot{x} \in -\mathrm{sign}(x) + 0.5\sin(t)$$

And let

$$\mathrm{sign}(x) \in \begin{cases} \{-1\}, & \text{if } x < 0 \\ [-1, 1], & \text{if } x = 0 \\ \{1\}, & \text{if } x > 0 \end{cases}$$

We have for some $t > t^*$ that $x(t) = 0$ and $\dot{x}(t) = 0$

That is $\mathrm{sign}(0) = [-1, 1] \ni 0.5\sin(t)$



Motivating examples - sliding mode - fixed

Consider the ODE

$$\dot{x} \in -\mathrm{sign}(x) + 0.5\sin(t)$$

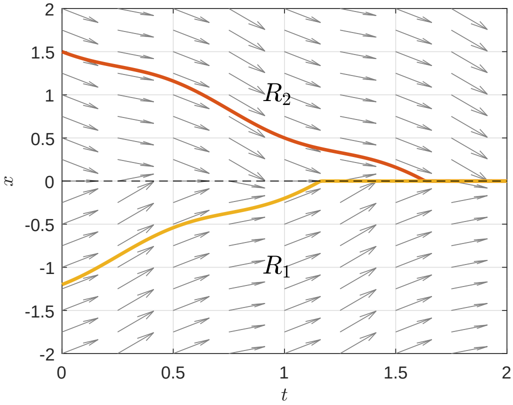
And let

$$\mathrm{sign}(x) \in \begin{cases} \{-1\}, & \text{if } x < 0 \\ [-1, 1], & \text{if } x = 0 \\ \{1\}, & \text{if } x > 0 \end{cases}$$

We have for some $t > t^*$ that $x(t) = 0$ and $\dot{x}(t) = 0$

That is $\mathrm{sign}(0) = [-1, 1] \ni 0.5\sin(t)$

It works! Thanks to A.F. Filippov

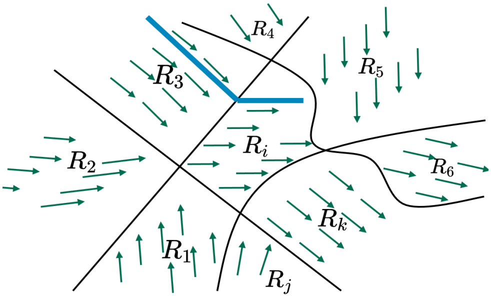


NSD2 Systems - state dependent switches

Regard **discontinuous** right-hand side, piecewise smooth on disjoint open regions $R_i \subset \mathbb{R}^{n_x}$

Discontinuous ODE (NSD2)

$$\dot{x} = f_i(x, u), \text{ if } x \in R_i, \quad i \in \{1, \dots, n_f\}$$



Numerical aims:

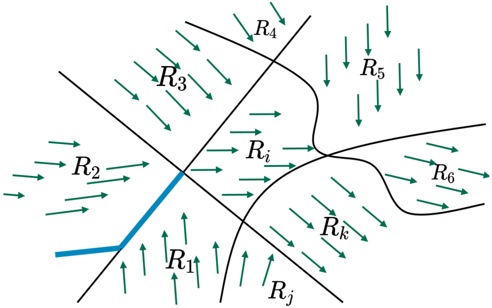
1. exactly detect switching times
2. obtain exact sensitivities across regions

NSD2 Systems - state dependent switches

Regard **discontinuous** right-hand side, piecewise smooth on disjoint open regions $R_i \subset \mathbb{R}^{n_x}$

Discontinuous ODE (NSD2)

$$\dot{x} = f_i(x,u), \text{ if } x \in R_i,$$
$$i \in \{1, \ldots, n_f\}$$



Numerical aims:

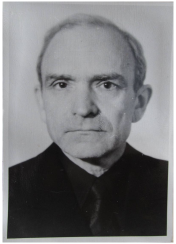
1. exactly detect switching times
2. obtain exact sensitivities across regions
3. appropriately treat evolution on boundaries (sliding mode $\rightarrow$ Filippov convexification)

Filippov Convexification

Dynamics not yet well-defined on region boundaries ∂R_i. Idea by A.F. Filippov (1923-2006): replace ODE by differential inclusion, using convex combination of neighboring vector fields.

Filippov Differential Inclusion

$$\dot{x} \in F_{\mathrm{F}}(x,u) := \Big\{ \sum_{i=1}^{n_f} f_i(x,u)\,\theta_i \;\Big|\; \sum_{i=1}^{n_f} \theta_i = 1, \\ \theta_i \geq 0, \quad i = 1, \dots n_f, \\ \theta_i = 0, \quad \text{if } x \notin \overline{R_i} \Big\}$$



Aleksei F. Filippov
(1923-2006)
image source: wikipedia

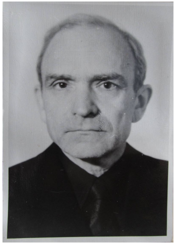
Filippov Convexification

Dynamics not yet well-defined on region boundaries ∂R_i. Idea by A.F. Filippov (1923-2006): replace ODE by differential inclusion, using convex combination of neighboring vector fields.

Filippov Differential Inclusion

$$\dot{x} \in F_{\mathrm{F}}(x,u) := \Big\{ \sum_{i=1}^{n_f} f_i(x,u)\,\theta_i \;\Big|\; \sum_{i=1}^{n_f} \theta_i = 1, \quad \theta_i \geq 0, \quad i = 1,\dots n_f, \quad \theta_i = 0, \quad \text{if } x \notin \overline{R_i} \Big\}$$

- for interior points $x \in R_i$ nothing changes: $F_{\mathrm{F}}(x,u) = \{f_i(x,u)\}$
- Provides meaningful generalization on region boundaries. E.g. on $\overline{R_1} \cap \overline{R_2}$ both θ_1 and θ_2 can be nonzero



Aleksei F. Filippov
(1923-2006)
image source: wikipedia

How to compute convex multipliers θ?

Answer in a remarkable paper by David E. Stewart from 1990

Numer. Math. 58, 299–328 (1990)

Numerische Mathematik

A high accuracy method for solving ODEs with discontinuous right-hand side

David Stewart

Department of Mathematics, University of Queensland, St. Lucia, Australia 4067

Received August 1, 1987/January 16, 1990

Summary. Ordinary Differential Equations with discontinuities in the state variables require a differential inclusion formulation to guarantee existence [8]. From this formulation a high accuracy method for solving such initial value problems is developed which can give any order of accuracy and "tracks" the discontinuities. The method uses an "active set" approach, and determines appropriate active sets from solutions to Linear Complementarity Problems. Convergence results are established under some non-degeneracy assumptions. The method has been implemented, and results compare favourably with previously published methods [7, 21].

How to compute convex multipliers θ?

Assume sets R_i given by [cf. Stewart, 1990]

$$\boxed{R_i = \left\{ x \in \mathbb{R}^n \middle| g_i(x) < \min_{j \neq i} g_j(x) \right\}}$$

How to compute convex multipliers θ?

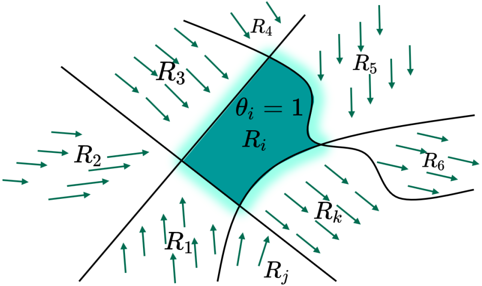
Assume sets R_i given by [cf. Stewart, 1990]

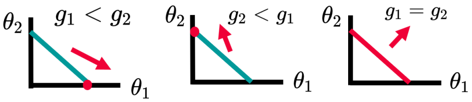
$$\boxed{R_i = \left\{x \in \mathbb{R}^n \middle| g_i(x) < \min_{j \neq i} g_j(x) \right\}}$$

Linear program (LP) Representation

$$\dot{x} = \sum_{i=1}^{n_f} f_i(x,u)\,\theta_i \quad \text{with}$$

$$\begin{aligned}
\theta \in \arg \min_{\tilde{\theta} \in \mathbb{R}^{n_f}} \quad & \sum_{i=1}^{n_f} g_i(x)\,\tilde{\theta}_i \\
\text{s.t.} \quad & \sum_{i=1}^{n_f} \tilde{\theta}_i = 1 \\
& \tilde{\theta} \geq 0
\end{aligned}$$





Note that the boundary between R_i and R_j is defined by $\{x \in \mathbb{R}^n \mid 0 = g_i(x) - g_j(x)\}$.

LP representation

$$\dot{x} = F(x,u)\,\theta$$

$$\text{with} \quad \theta \in \underset{\tilde{\theta} \in \mathbb{R}^{n_f}}{\operatorname{argmin}} \quad g(x)^\top \tilde{\theta} \qquad \text{s.t.} \quad 0 \le \tilde{\theta}, \quad 1 = e^\top \tilde{\theta}$$

where

$$\begin{aligned} F(x,u) &:= [f_1(x,u), \ldots, f_{n_f}(x,u)] \in \mathbb{R}^{n_x \times n_f} \\ g(x) &:= [g_1(x), \ldots, g_{n_f}(x)]^\top \in \mathbb{R}^{n_f} \\ e &:= [1,1,\ldots,1]^\top \in \mathbb{R}^{n_f} \end{aligned}$$

From Filippov to dynamic complementarity systems

Using the KKT conditions of the parametric LP

LP representation

$$\dot{x} = F(x,u)\,\theta$$

$$\text{with} \quad \theta \in \underset{\tilde{\theta} \in \mathbb{R}^{n_f}}{\operatorname{argmin}} \quad g(x)^\top \tilde{\theta} \quad \text{s.t.} \quad 0 \leq \tilde{\theta}, \quad 1 = e^\top \tilde{\theta}$$

where

$$F(x,u) := [f_1(x,u), \ldots, f_{n_f}(x,u)] \in \mathbb{R}^{n_x \times n_f}$$
$$g(x) := [g_1(x), \ldots, g_{n_f}(x)]^\top \in \mathbb{R}^{n_f}$$
$$e := [1, 1, \ldots, 1]^\top \in \mathbb{R}^{n_f}$$

Express equivalently by optimality conditions:

Dynamic Complementarity System (DCS)

$$\dot{x} = F(x,u)\,\theta \tag{1a}$$
$$0 = g(x) - \lambda - e\mu \tag{1b}$$
$$0 \leq \theta \perp \lambda \geq 0 \tag{1c}$$
$$1 = e^\top \theta \tag{1d}$$

Compact notation

$$\dot{x} = F(x,u)\,\theta$$
$$0 = G_{\mathrm{LP}}(x, \theta, \lambda, \mu),$$

- $\mu \in \mathbb{R}$ and $\lambda \in \mathbb{R}^{n_f}$ are Lagrange multipliers
- (1c) $\Leftrightarrow \min\{\theta, \lambda\} = 0 \in \mathbb{R}^{n_f}$
- Together, (1b), (1c), (1d) determine the $(2n_f + 1)$ variables θ, λ, μ uniquely

Optimal control needs to solve Nonlinear Programs (NLPs)

Original optimal control problem in continuous time

$$
\begin{aligned}
\min_{\substack{x(\cdot),u(\cdot),\\ \theta(\cdot),\lambda(\cdot),\mu(\cdot)}} \quad & \int_0^T L(x,u)\mathrm{d}t + E(x(T)) \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = \textstyle\sum_{i=1}^{n_f} f_i(x(t),u(t))\ \theta_i(t) \\
& 0 = G_{\mathrm{LP}}(x(t),\theta(t),\lambda(t),\mu(t)) \\
& 0 \geq h(x(t),u(t)),\ t \in [0,T] \\
& 0 \geq r(x(T))
\end{aligned}
$$

Assume smooth (convex) L, E, h, r
Nonsmooth dynamics make problem nonconvex
Direct methods discretize, then optimize
E.g., collocation or multiple shooting

Optimal control needs to solve Nonlinear Programs (NLPs)

Original optimal control problem in continuous time

$$\begin{aligned}
\min_{\substack{x(\cdot),u(\cdot),\\ \theta(\cdot),\lambda(\cdot),\mu(\cdot)}} \quad & \int_0^T L(x,u)\mathrm{d}t + E(x(T)) \\
\text{s.t.} \quad & x(0) = \bar{x}_0 \\
& \dot{x}(t) = \textstyle\sum_{i=1}^{n_f} f_i(x(t),u(t))\ \theta_i(t) \\
& 0 = G_{\mathrm{LP}}(x(t),\theta(t),\lambda(t),\mu(t)) \\
& 0 \geq h(x(t),u(t)),\ t \in [0,T] \\
& 0 \geq r(x(T))
\end{aligned}$$

Assume smooth (convex) L, E, h, r

Nonsmooth dynamics make problem nonconvex

Direct methods discretize, then optimize

E.g., collocation or multiple shooting

Goal: discretized optimal control problem (an NLP)

$$\begin{aligned}
\min_{x,z,u} \quad & \textstyle\sum_{k=0}^{N-1} \Phi_L(x_k,z_k,u_k) + E(x_N) \\
\text{s.t.} \quad & x_0 = \bar{x}_0 \\
& x_{k+1} = \Phi_f^{\mathrm{dif}}(x_k,z_k,u_k) \\
& 0 = \Phi_f^{\mathrm{alg}}(x_k,z_k,u_k) \\
& 0 \geq \Phi_h(x_k,z_k,u_k),\ k = 0,\dots,N-1 \\
& 0 \geq r(x_N)
\end{aligned}$$

Smooth convex Φ_L, E, Φ_h, r

Variables $x = (x_0,\dots)$, $z = (z_0,\dots)$ and $u = (u_0,\dots,u_{N-1})$ summarized in vector $w \in \mathbb{R}^{n_w}$

Nonsmooth Φ_f^{alg}

Why not use standard discretization methods?

Motivating examples - sliding mode - explicit Euler

Consider the ODE

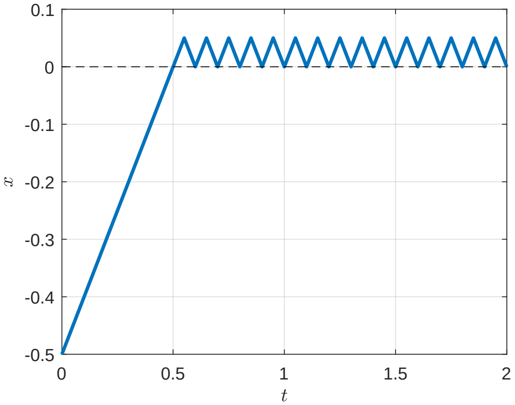
$$\dot{x} = -\mathrm{sign}(x)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

We use

$$x_{k+1} = x_k - h \cdot \mathrm{sign}(x_k)$$



Motivating examples - sliding mode - explicit Euler

Consider the ODE

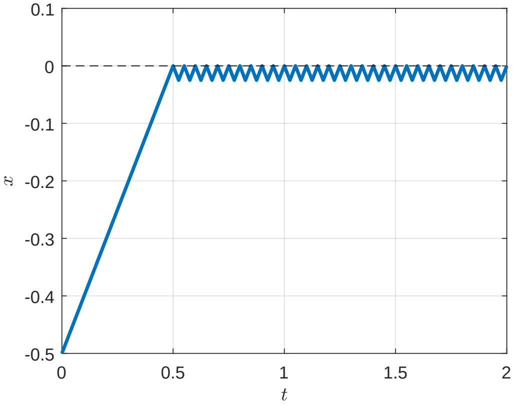
$$\dot{x} = -\mathrm{sign}(x)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

We use

$$x_{k+1} = x_k - h \cdot \mathrm{sign}(x_k)$$



Consider the ODE

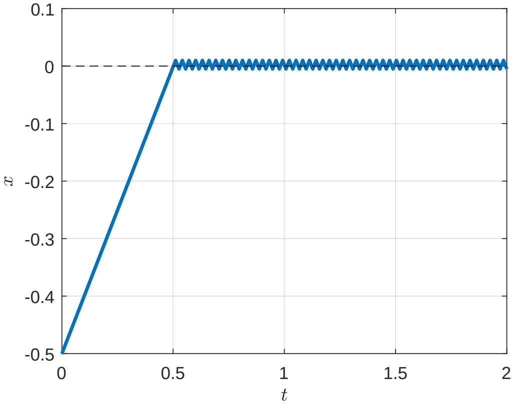
$$\dot{x} = -\mathrm{sign}(x)$$

And let

$$\mathrm{sign}(x) = \begin{cases} -1, & \text{if } x < 0 \\ 0, & \text{if } x = 0 \\ 1, & \text{if } x > 0 \end{cases}$$

We use

$$x_{k+1} = x_k - h \cdot \mathrm{sign}(x_k)$$



Numerical simulation example: unstable switched oscillator

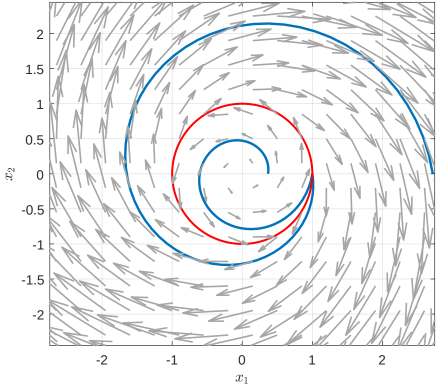
Regard an unstable nonsmooth oscillator

$$\dot{x}(t) = \begin{cases} A_1 x, & c(x) < 0, \\ A_2 x, & c(x) > 0, \end{cases}$$

with

$$A_1 = \begin{bmatrix} 1 & \omega \\ -\omega & 1 \end{bmatrix}, \; A_2 = \begin{bmatrix} 1 & -\omega \\ \omega & 1 \end{bmatrix},$$

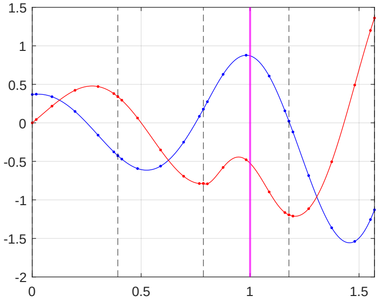
$$c(x) = x_1^2 + x_2^2 - 1, \; \omega = 2\pi, \; x(0) = [e^{-1} \; 0]^\top$$



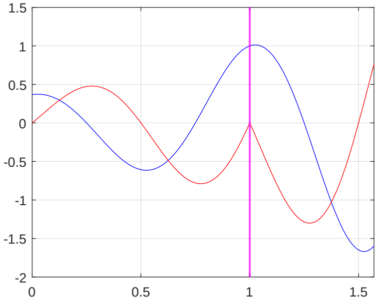
Numerical simulation example: importance of switch detection

- Use Implicit Runge-Kutta Gauss Legendre method of order **10**!
- If implicit Euler has an accuracy of $\approx 10^{-1}$, then this method has for the same step-size an accuracy of $\approx 10^{-10}$
- 4 integration steps over $T = \frac{\pi}{2}$, $h = 0.3928...$

Standard IRK vs FESD IRK



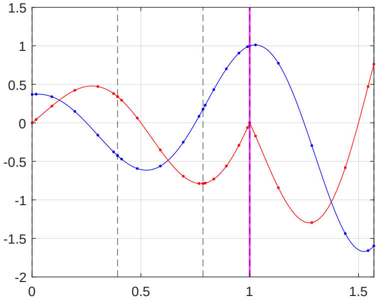
Analytic solution



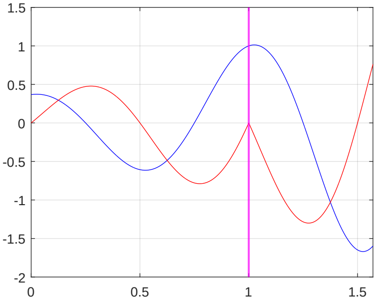
Numerical simulation example: importance of switch detection

- Use Implicit Runge-Kutta Gauss Legendre method of order **10**!
- If implicit Euler has an accuracy of $\approx 10^{-1}$, then this method has for the same step-size an accuracy of $\approx 10^{-10}$
- 4 integration steps over $T = \frac{\pi}{2}$, $h = 0.3928...$

Standard IRK vs FESD IRK



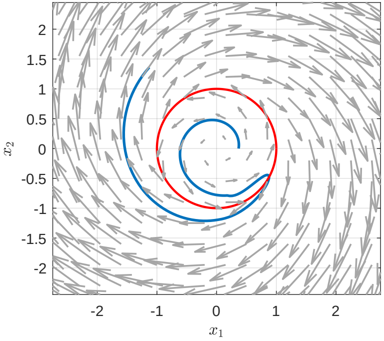
Analytic solution



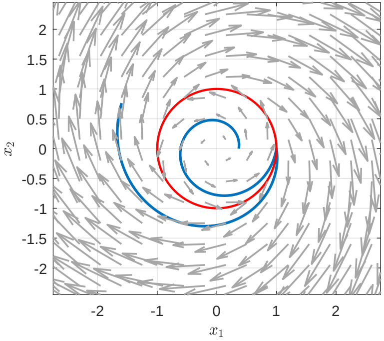
Numerical simulation example: importance of switch detection

- Use Implicit Runge-Kutta Gauss Legendre method of order **10**!
- If implicit Euler has an accuracy of $\approx 10^{-1}$, then this method has for the same step-size an accuracy of $\approx 10^{-10}$
- 4 integration steps over $T = \frac{\pi}{2}$, $h = 0.3928...$

Standard IRK vs FESD IRK



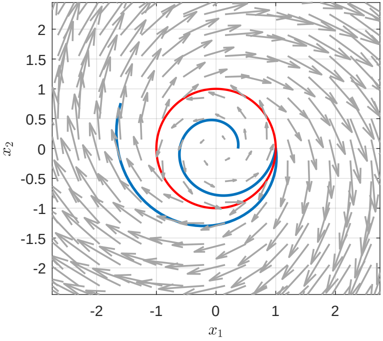
Analytic solution



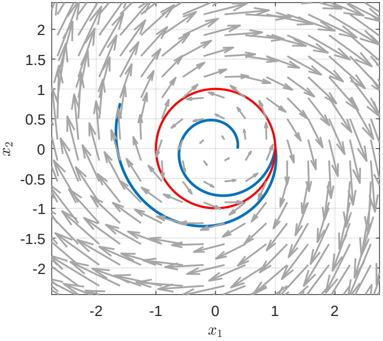
Numerical simulation example: importance of switch detection

- Use Implicit Runge-Kutta Gauss Legendre method of order **10**!
- If implicit Euler has an accuracy of $\approx 10^{-1}$, then this method has for the same step-size an accuracy of $\approx 10^{-10}$
- 4 integration steps over $T = \frac{\pi}{2}$, $h = 0.3928...$

Standard IRK vs FESD IRK



Analytic solution



Why not smooth everything?

Direct optimal control with a standard IRK discretization

Tutorial example inspired by [Stewart & Anitescu, 2010]

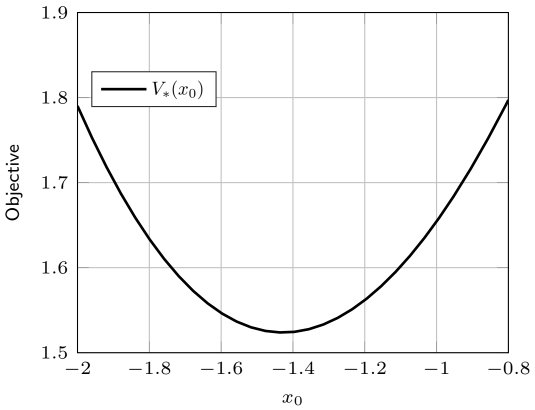
Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$
$$\text{s.t.} \quad \dot{x}(t) = 2 - \mathrm{sign}(x(t)), \quad t \in [0,2]$$

Free initial value $x(0)$ is the effective degree of freedom.
Denote by $V_*(x_0)$ the nonsmooth objective value for the unique feasible trajectory starting at $x(0) = x_0$.

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_*(x_0)$$



Direct optimal control with a standard IRK discretization - smoothing

Tutorial example inspired by [Stewart & Anitescu, 2010]

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2)-5/3)^2$$

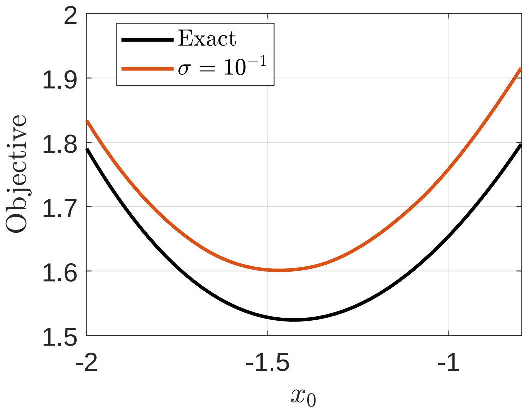
$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh\left(\frac{x(t)}{\sigma}\right), \quad t \in [0,2]$$

Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with $h = 0.1$; i.e., $N = 20$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - smoothing

Tutorial example inspired by [Stewart & Anitescu, 2010]

Continuous-time OCP

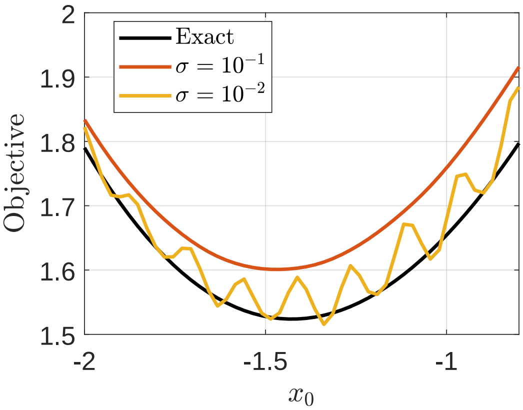
$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh\left(\frac{x(t)}{\sigma}\right), \quad t \in [0, 2]$$

Free initial value $x(0)$ is the effective degree of freedom.
IRK Radau of order 5 with
$h = 0.1$; i.e., $N = 20$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - smoothing

Tutorial example inspired by [Stewart & Anitescu, 2010]

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

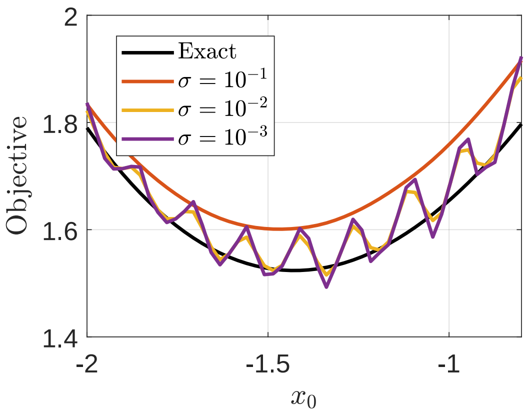
$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh\left(\frac{x(t)}{\sigma}\right), \quad t \in [0,2]$$

Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with $h = 0.1$; i.e., $N = 20$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - smoothing

Tutorial example inspired by [Stewart & Anitescu, 2010]

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

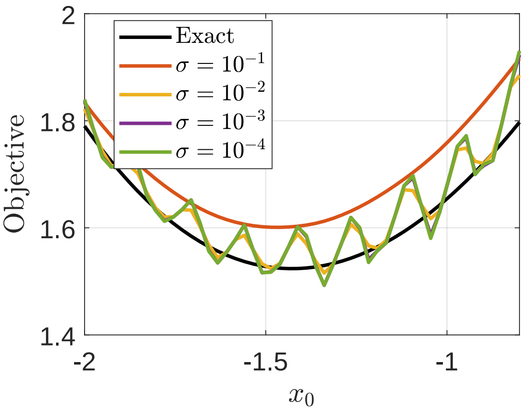
$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh\left(\frac{x(t)}{\sigma}\right), \quad t \in [0,2]$$

Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with $h = 0.1$; i.e., $N = 20$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - Let us try again

See "Limits of MPCC formulations in direct optimal control with nonsmooth differential equations", N., Albrecht, Diehl, ECC 2020

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2)-5/3)^2$$

$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh\left(\frac{x(t)}{\sigma}\right), \quad t \in [0,2]$$

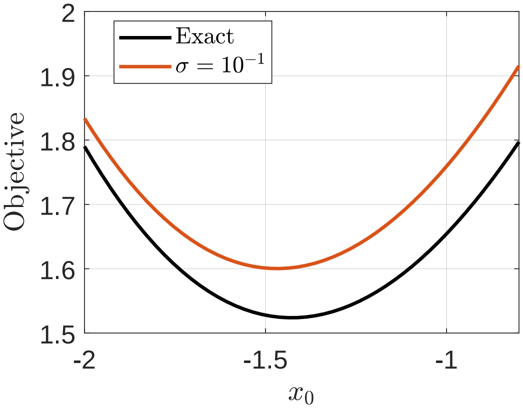
Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with

$h = 0.01$; i.e., $N = 200$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - Let us try again

See "Limits of MPCC formulations in direct optimal control with nonsmooth differential equations", N., Albrecht, Diehl, ECC 2020

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh\left(\frac{x(t)}{\sigma}\right), \quad t \in [0,2]$$

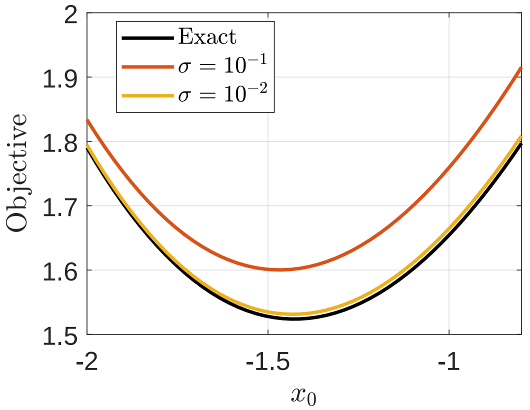
Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with

$h = 0.01$; i.e., $N = 200$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - Let us try again

See "Limits of MPCC formulations in direct optimal control with nonsmooth differential equations", N., Albrecht, Diehl, ECC 2020

Continuous-time OCP

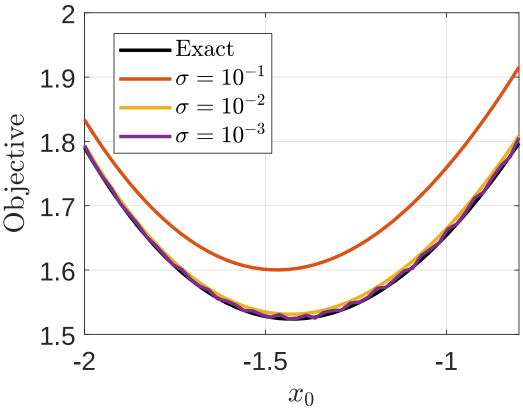
$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh(\frac{x(t)}{\sigma}), \quad t \in [0,2]$$

Free initial value $x(0)$ is the effective degree of freedom.
IRK Radau of order 5 with
$h = 0.01$; i.e., $N = 200$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - Let us try again

See "Limits of MPCC formulations in direct optimal control with nonsmooth differential equations", N., Albrecht, Diehl, ECC 2020

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

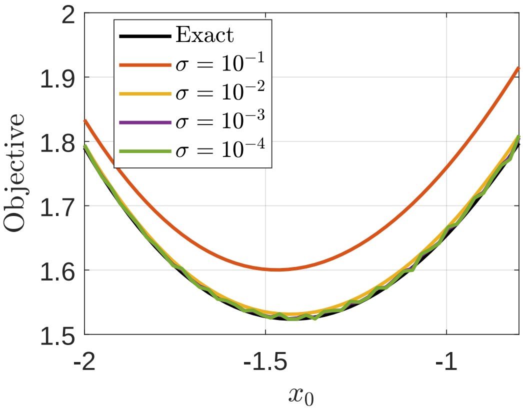
$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh(\frac{x(t)}{\sigma}), \quad t \in [0,2]$$

Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with $h = 0.01$; i.e., $N = 200$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization - Let us try again

See "Limits of MPCC formulations in direct optimal control with nonsmooth differential equations", N., Albrecht, Diehl, ECC 2020

Continuous-time OCP

$$\min_{x(\cdot)\in\mathcal{C}^0([0,2])} \int_0^2 x(t)^2 \mathrm{d}t + (x(2) - 5/3)^2$$

$$\text{s.t.} \quad \dot{x}(t) = 2 - \tanh(\frac{x(t)}{\sigma}), \quad t \in [0,2]$$

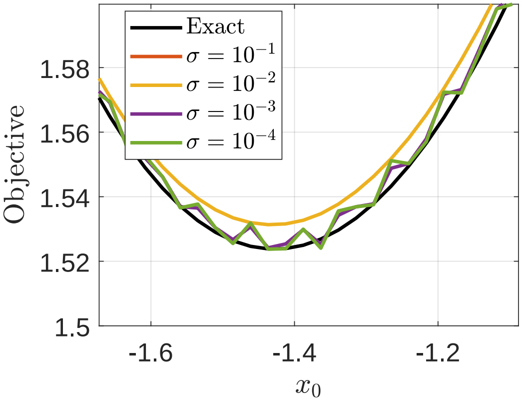
Free initial value $x(0)$ is the effective degree of freedom.

IRK Radau of order 5 with

$h = 0.01$; i.e., $N = 200$ steps

Equivalent reduced problem

$$\min_{x_0\in\mathbb{R}} V_\sigma(x_0)$$



Direct optimal control with a standard IRK discretization

Tutorial example inspired by [Stewart & Anitescu, 2010]

Numer. Math. (2010) 114:653–695
DOI 10.1007/s00211-009-0262-2

Numerische Mathematik

Optimal control of systems with discontinuous differential equations

David E. Stewart · Mihai Anitescu

(another remarkable paper by D. Stewart)

- discretize the OCP with standard IRK for DCS
- numerical sensitivities wrong independent of the step-size
- smoothing works only if step-size smaller than smoothing parameter

Numer. Math. (2010) 114:653–695
DOI 10.1007/s00211-009-0262-2

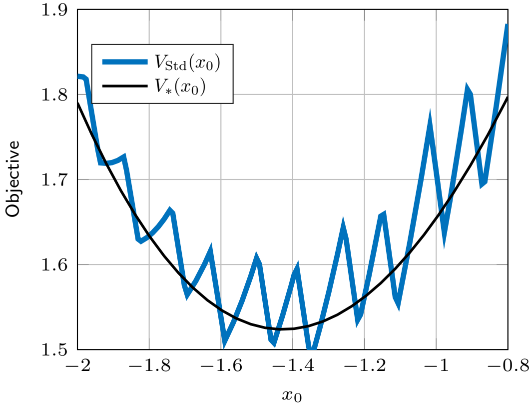
Numerische Mathematik

Optimal control of systems with discontinuous differential equations

David E. Stewart · Mihai Anitescu

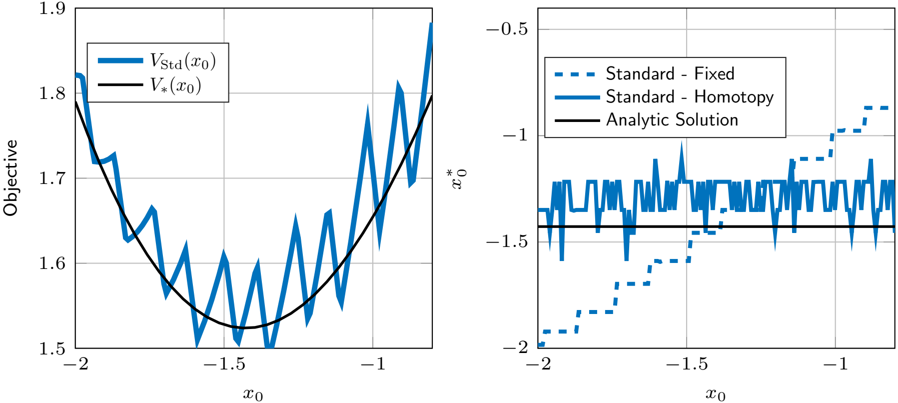
(another remarkable paper by D. Stewart)

- discretize the OCP with standard IRK for DCS
- numerical sensitivities wrong independent of the step-size
- smoothing works only if step-size smaller than smoothing parameter



Direct optimal control with a standard IRK discretization

Tutorial example inspired by [Stewart & Anitescu, 2010]



- Spurious local minima, optimizer gets trapped close to initialization
- Sensitivity correct if step-sizes smaller than smoothing parameter [Stewart & Anitescu, 2010] $\implies$ homotopy improves convergence
- Integrator NOT differentiable even when they appear to be so!
- Still, at best $O(h)$ accuracy can be expected

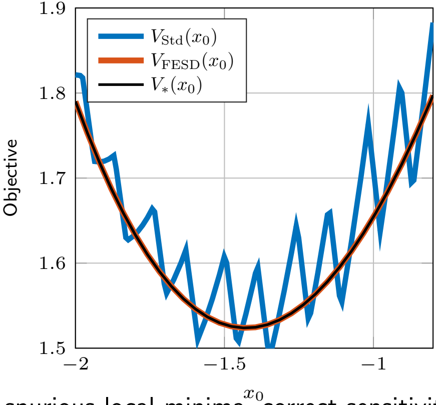
Summary of theoretical results

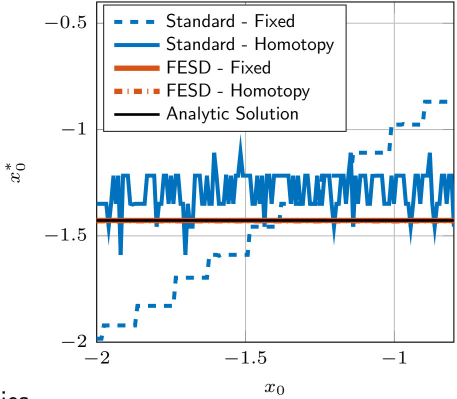
Submitted to Num. Mat, arXiv:2205.05337

1. Convergence of the FESD method to a Filippov solution of the underlying system with accuracy $O(h^p)$ is proven. Here, p is the order of the underlying smooth IRK scheme.
2. Convergence of numerical sensitivities to the true value with $O(h^p)$ is given. The Stewart & Anitescu problem is resolved.
3. An FESD problem needs to solve a nonlinear complementarity problem (NCP) to advance the integration. The solutions of these NCP are locally unique.

Revisiting the OCP example - now with FESD

Tutorial example inspired by [Stewart & Anitescu, 2010]

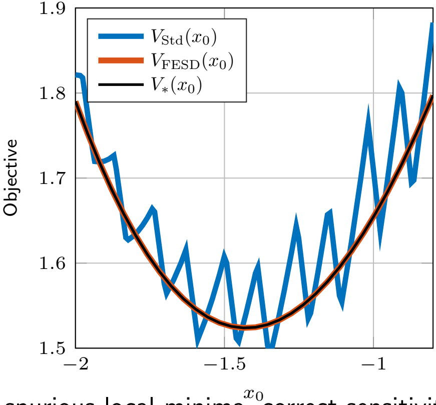


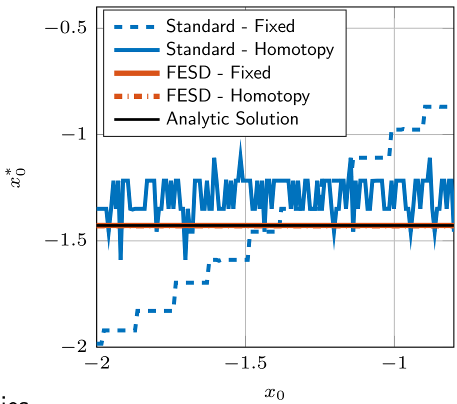


- No spurious local minima, correct sensitivities
- Convergence to the "true" local minima, both with homotopy and without it
- In contrast to the standard approach with accuracy $O(h)$, now we have $O(h^p)$

Revisiting the OCP example - now with FESD

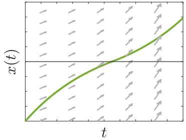
Tutorial example inspired by [Stewart & Anitescu, 2010]



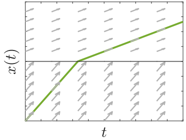


- No spurious local minima, correct sensitivities
- Convergence to the "true" local minima, both with homotopy and without it
- In contrast to the standard approach with accuracy $O(h)$, now we have $O(h^p)$
- FESD resolves the accuracy and convergence issues

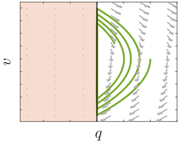
Regard ordinary differential equation (ODE) with a **nonsmooth** right-hand side (RHS).
Distinguish three cases:



NSD1: non-differentiable RHS, e.g., $\dot{x} = 1 + |x|$



NSD2: state dependent switch of RHS, e.g., $\dot{x} = 2 - \mathrm{sign}(x)$



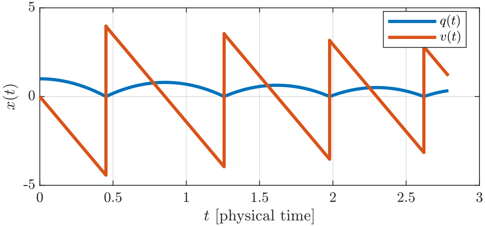
NSD3: **state dependent jump**, e.g., bouncing ball, $v(t_+) = -0.9\, v(t_-)$

NSD3 state jump example: bouncing ball

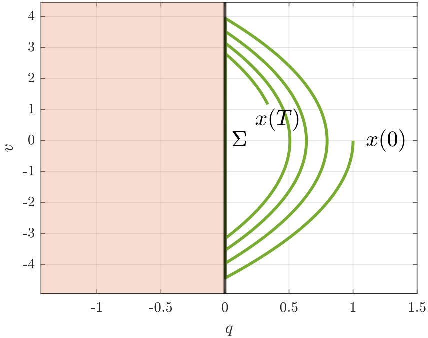
Bouncing ball with state $x = (q, v)$:

$$\begin{aligned} \dot{q} = v,\ m\dot{v} = -mg, \quad & \text{if } q > 0 \\ v(t^+) = -0.9\, v(t^-), \quad & \text{if } q(t^-) = 0 \text{ and } v(t^-) < 0 \end{aligned}$$

Time plot of bouncing ball trajectory:



Phase plot of bouncing ball trajectory:



Question: could we transform NSD3 systems into (easier) NSD2 systems?

Three ideas:

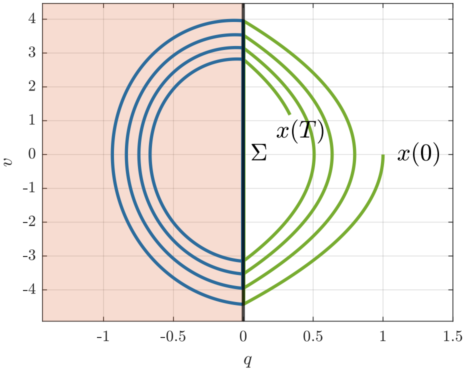
1. mimic state jump by **auxiliary dynamic system** $\dot{x} = f_{\mathrm{aux}}(x)$ on prohibited region
2. introduce a **clock state** $t(\tau)$ that stops counting when the auxiliary system is active
3. adapt speed of time, $\frac{\mathrm{d}t}{\mathrm{d}\tau} = s$ with $s \geq 1$, and **impose terminal constraint** $t(T) = T$

The time-freezing reformulation

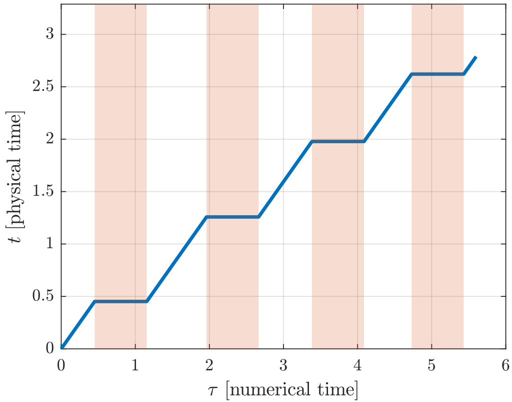
Augmented state $(x, t) \in \mathbb{R}^{n+1}$ evolves in **numerical time** τ. Augmented system is nonsmooth, of NSD2 type:

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\begin{bmatrix} x \\ t \end{bmatrix} = \begin{cases} s \begin{bmatrix} f(x) \\ 1 \end{bmatrix}, & \text{if } c(x) \geq 0 \\ \begin{bmatrix} s f_{\mathrm{aux}}(x) \\ 0 \end{bmatrix}, & \text{if } c(x) < 0 \end{cases}$$

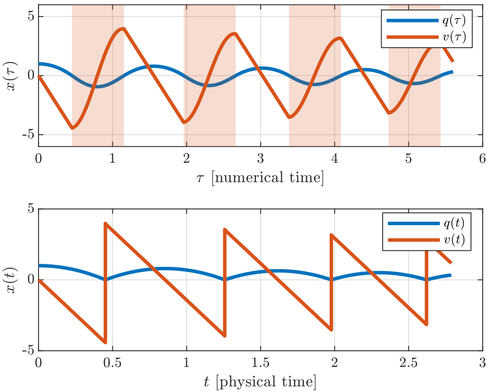
- During normal times, system and clock state evolve with adapted speed $s \geq 1$.
- Auxiliary system $\frac{\mathrm{d}x}{\mathrm{d}\tau} = f_{\mathrm{aux}}(x)$ mimics state jump while time is frozen, $\frac{\mathrm{d}t}{\mathrm{d}\tau} = 0$.



Time-freezing for bouncing ball example



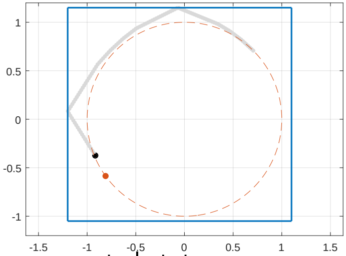
Evolution of physical time (clock state) during augmented system simulation ($s = 1$).



We can recover the true solution by plotting $x(\tau)$ vs. $t(\tau)$ and disregarding "frozen pieces".

A tracking OCP example with Time-Freezing and FESD in NOSNOC

Regard bouncing ball in two dimensions driven by bounded force: $\boxed{\ddot{q} = u}$



- augmented state $x = (q, \dot{q}, t) \in \mathbb{R}^5$
- $n_f = 9$ regions (8 with auxiliary dynamics for state jumps)

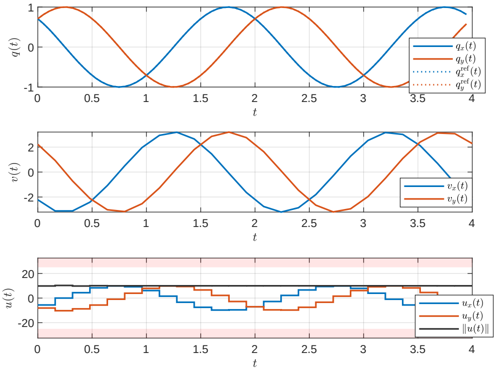
$$
\begin{aligned}
\min_{\substack{x(.),u(.),s(.),\\ \theta(.),\lambda(.),\mu(.)}} \quad & \int_0^T (q - q_{\mathrm{ref}}(\tau))^\top (q - q_{\mathrm{ref}}(\tau))\, s(\tau)\, \mathrm{d}\tau \\
\text{s.t.} \quad & x(0) = x_0, \quad t(T) = T, \\
& x'(\tau) = \sum_{i=1}^{n_f} \theta_i(\tau) f_i(x(\tau), u(\tau), s(\tau)), \\
& 0 = g(x(\tau)) - \lambda(\tau) - \mu(\tau) e, \\
& 0 \leq \lambda(\tau) \perp \theta(\tau) \geq 0, \\
& 1 = e^\top \theta(\tau), \\
& \|u(\tau)\|_2^2 \leq u_{\max}^2, \\
& 1 \leq s(\tau) \leq s_{\max}, \ \tau \in [0, T].
\end{aligned}
$$

$$q_{\mathrm{ref}}(\tau) = (R\cos(\omega\, t(\tau)), R\sin(\omega\, t(\tau))).$$

Results with slowly moving reference

For $\omega = \pi$, tracking is easy: no jumps occur in optimal solution.

- Regard time horizon of two periods
- $N = 25$ equidistant control intervals
- use FESD with $N_{\mathrm{FE}} = 3$ finite elements with Radau 3 on each control interval
- each FESD interval has one constant control u and one speed of time s
- MPCC solved via ℓ_∞ penalty reformulation and homotopy
- For homotopy convergence: in total 4 NLPs solved with `IPOPT` via `CasADi`



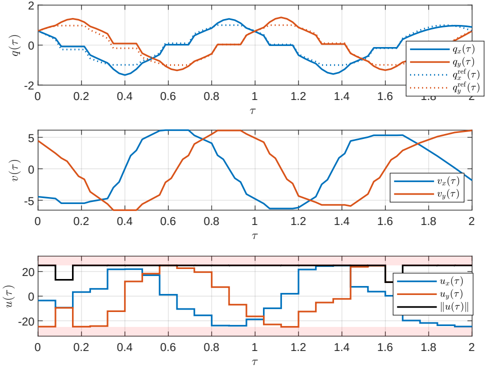
States and controls in physical time.

Results with slowly moving reference - movie

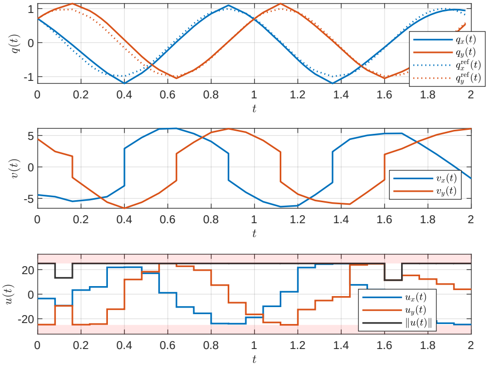
For $\omega = \pi$, tracking is easy: no jumps occur in optimal solution.

Results with fast reference

For $\omega = 2\pi$, tracking is only possible if ball bounces against walls.



States and controls in numerical time.



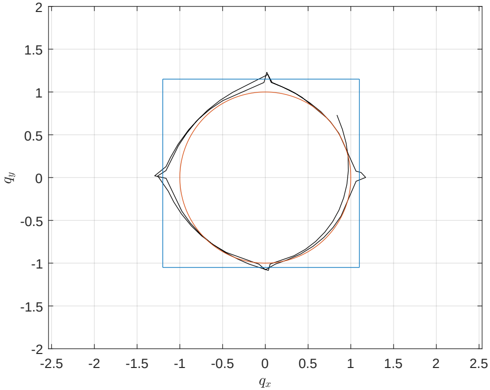
States and controls in physical time.

Results with fast reference - movie

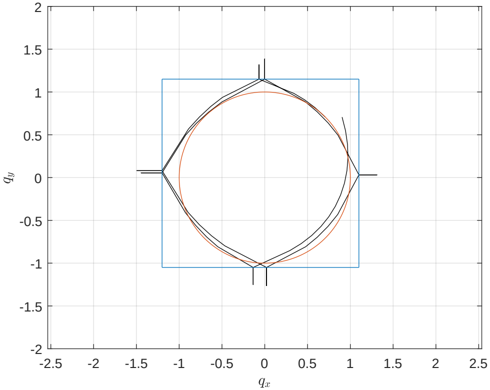
For $\omega = 2\pi$, tracking is only possible if ball bounces against walls.

Homotopy: first iteration vs converged solution

Geometric trajectory



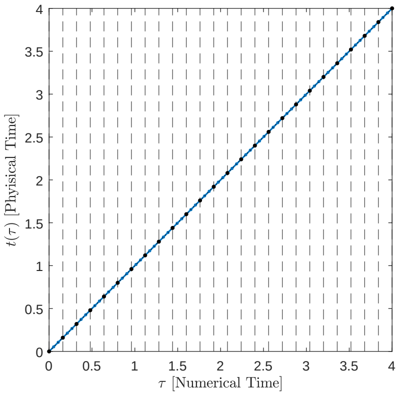
After the first homotopy iteration



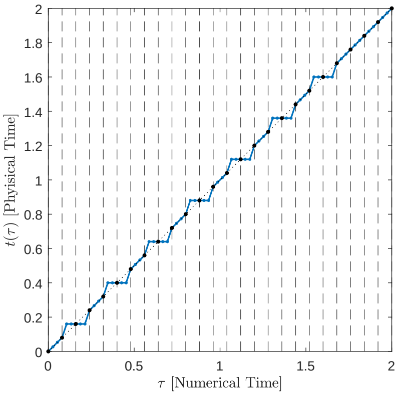
The solution trajectory after convergence

Physical vs. Numerical Time

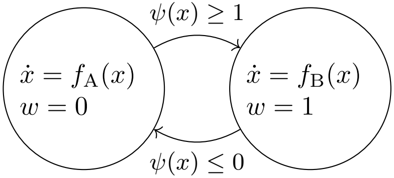
for $\omega = \pi$



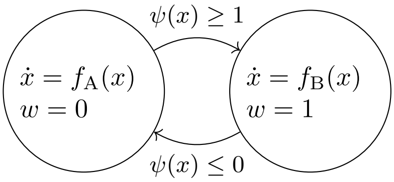
for $\omega = 2\pi$

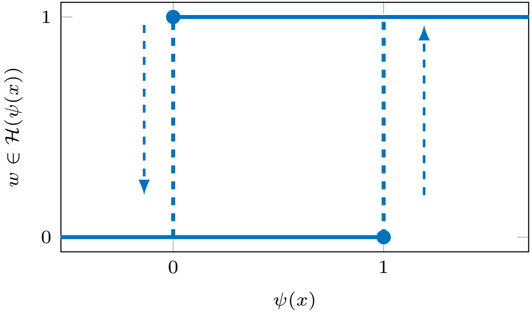


Hybrid systems and finite automaton



Hybrid systems and finite automaton

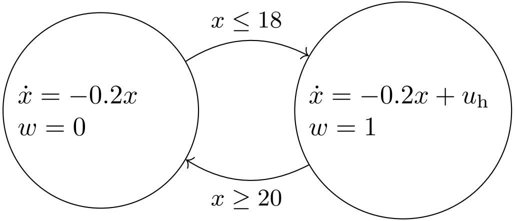




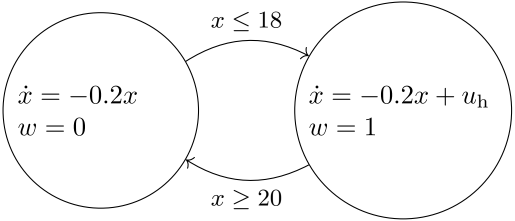
Hybrid system with hysteresis (*incomplete description*)

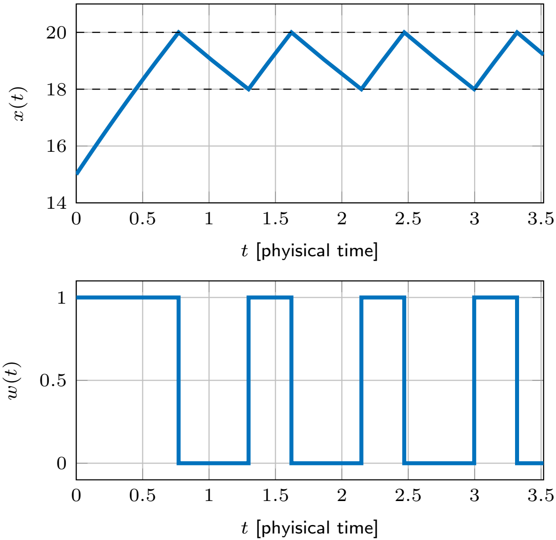
$$\dot{x} = f(x, w) = (1 - w) f_{\mathrm{A}}(x) + w f_{\mathrm{B}}(x)$$

Tutorial example: thermostat with hysteresis



Tutorial example: thermostat with hysteresis

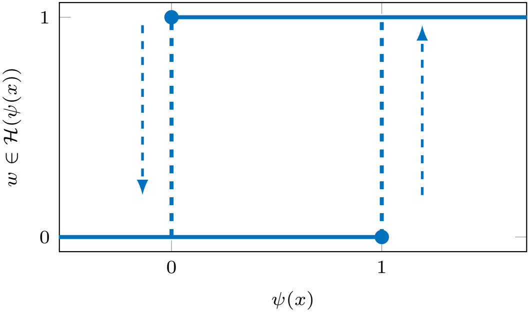




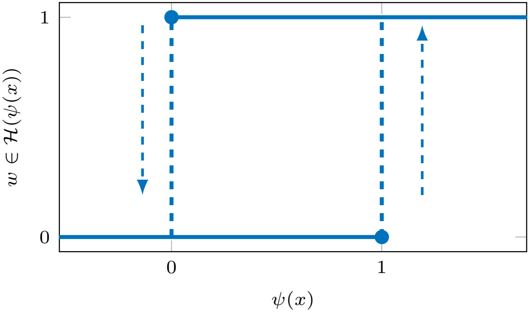
Hysteresis: a system with state jumps

Hybrid system with hysteresis

$$
\begin{aligned}
\dot{x} &= f(x,w) = (1-w)f_{\mathrm{A}}(x) + w f_{\mathrm{B}}(x) \\
\dot{w} &= 0
\end{aligned}
$$



Hysteresis: a system with state jumps



Hybrid system with hysteresis

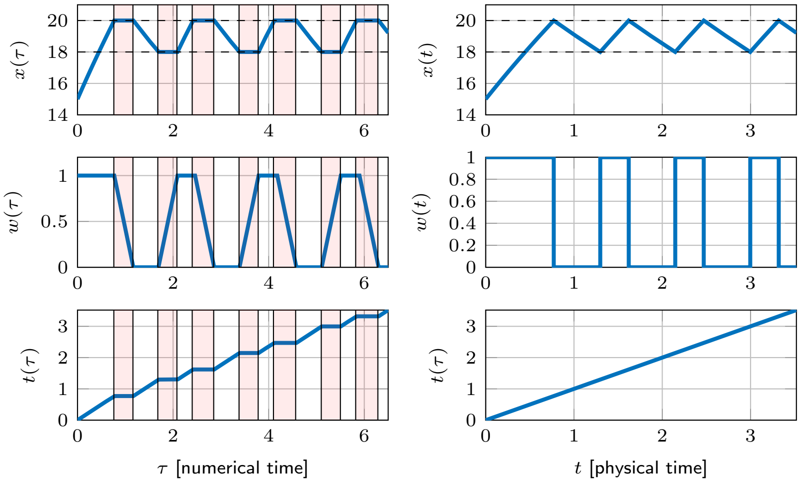
$$\dot{x} = f(x, w) = (1 - w)f_{\mathrm{A}}(x) + w f_{\mathrm{B}}(x)$$
$$\dot{w} = 0$$

The State Jump Law

1. if $w(t_{\mathrm{s}}^-) = 0$ and $\psi(x(t_{\mathrm{s}}^-)) = 1$, then $x(t_{\mathrm{s}}^+) = x(t_{\mathrm{s}}^-)$ and $w(t_{\mathrm{s}}^+) = 1$
2. if $w(t_{\mathrm{s}}^-) = 1$ and $\psi(x(t_{\mathrm{s}}^-)) = 0$, then $x(t_{\mathrm{s}}^+) = x(t_{\mathrm{s}}^-)$ and $w(t_{\mathrm{s}}^+) = 0$

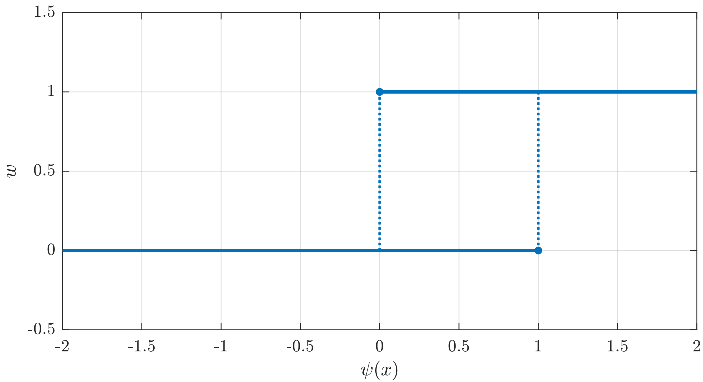
Remember: $w(t)$ is now a discontinuous differential state!

Tutorial example: thermostat and time-freezing



Time-freezing: the state space

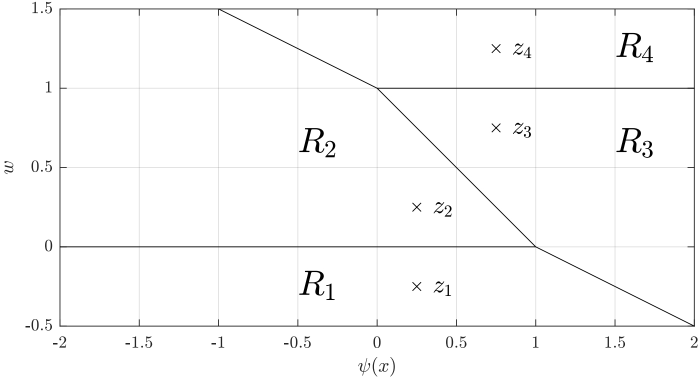
A look at the $(\psi(x), w)-$plane



- Everything except the blue solid curve is prohibited in the $(\psi, w)-$ space (use 1^{st} principle of time-freezing)
- The evolution happens in a lower-dimensional space $\implies$ *sliding mode* (use 4^{th} principle of time-freezing)

Time-freezing: partitioning of the space

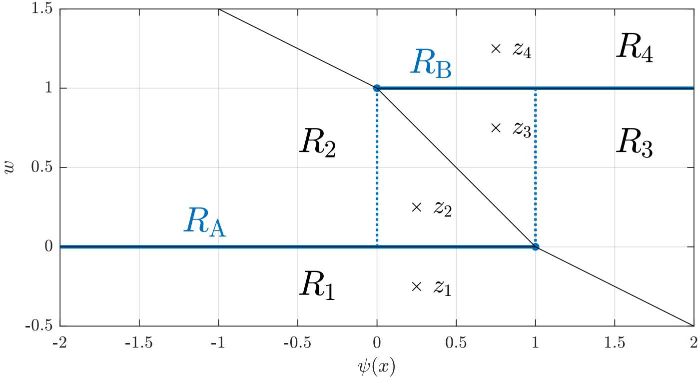
An efficient partition leads to less variables in FESD



- Partition the state space into *Voronoi regions*:
  $R_i = \{z \mid \|z - z_i\|^2 < \|z - z_j\|^2,\ j = 1, \dots, 4, j \neq i\},\ z = (\psi(x), w)$

Time-freezing: partitioning of the space

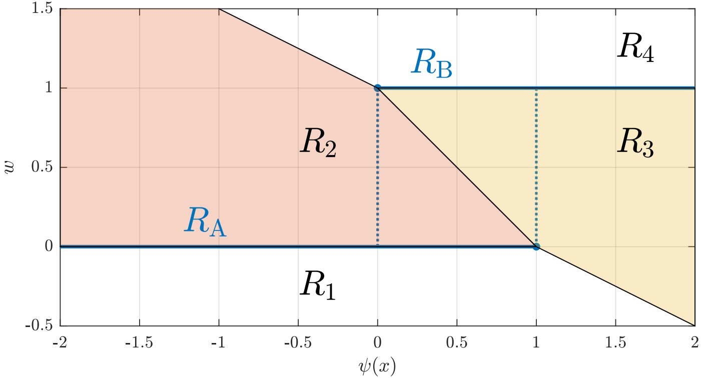
An efficient partition leads to less variables in FESD



- Partition the state space into *Voronoi regions*:
  $R_i = \{z \mid \|z - z_i\|^2 < \|z - z_j\|^2,\ j = 1, \ldots, 4, j \neq i\},\ z = (\psi(x), w)$
- Feasible region for initial *hybrid system with hysteresis* on the region boundaries

Time-freezing: auxiliary dynamics

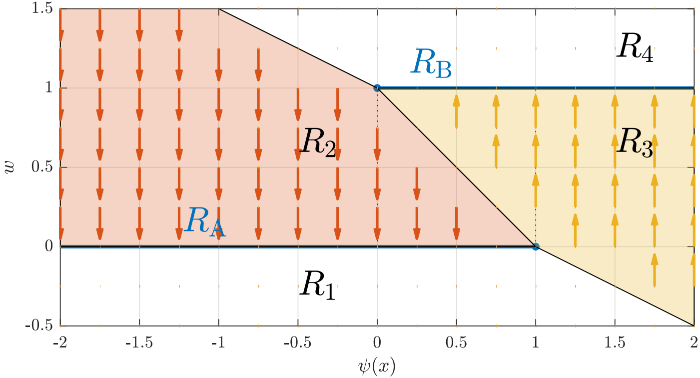
To mimic state jumps in finite numerical time



- Use regions R_2 and R_3 to define auxiliary dynamics for the state jumps of $w(\cdot)$

Time-freezing: auxiliary dynamics

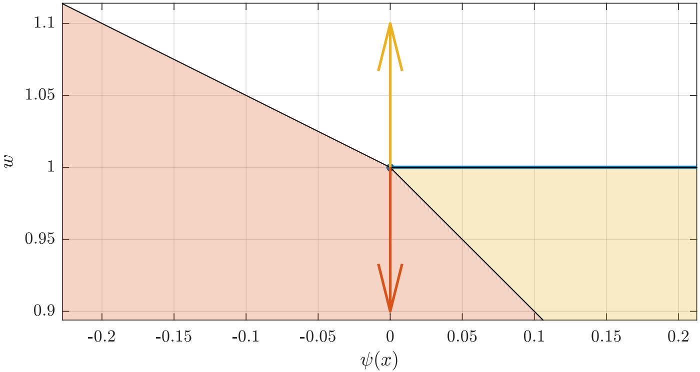
To mimic state jumps in finite numerical time



- Use regions R_2 and R_3 to define auxiliary dynamics for the state jumps of $w(\cdot)$
- Evolution in $w-$direction happens only for $\psi \in \{0, 1\}$

Time-freezing: auxiliary dynamics

To mimic state jumps in finite numerical time



- Use regions R_2 and R_3 to define auxiliary dynamics for the state jumps of $w(\cdot)$
- Evolution in $w-$direction happens only for $\psi \in \{0,1\}$
- Zoom in: with a naive approach one has locally nonunique solutions

Time-freezing: auxiliary dynamics

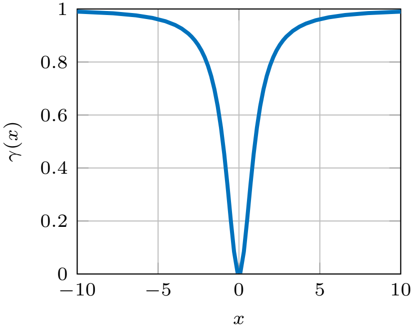
The new state space of the system is $y = (x, w, t) \in \mathbb{R}^{n_x+2}$

Auxiliary dynamics

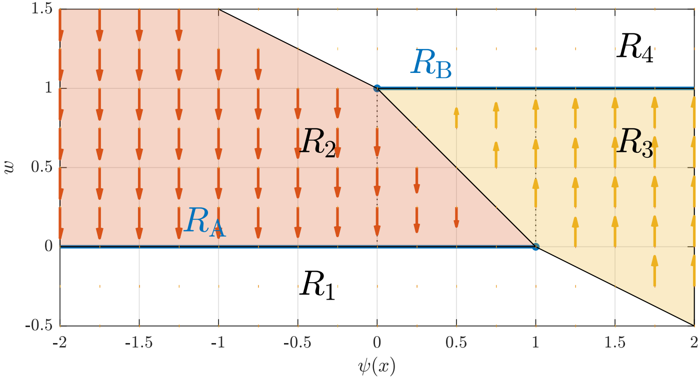
$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{aux,A}}(y) := \begin{bmatrix} 0 \\ -\gamma(\psi(x)) \\ 0 \end{bmatrix}$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{aux,B}}(y) := \begin{bmatrix} 0 \\ \gamma(\psi(x) - 1) \\ 0 \end{bmatrix}$$

$$\gamma(x) = \frac{ax^2}{1 + x^2}$$

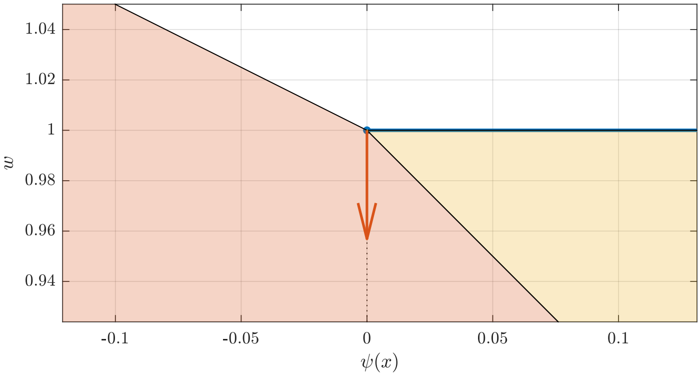


Time-freezing: auxiliary dynamics



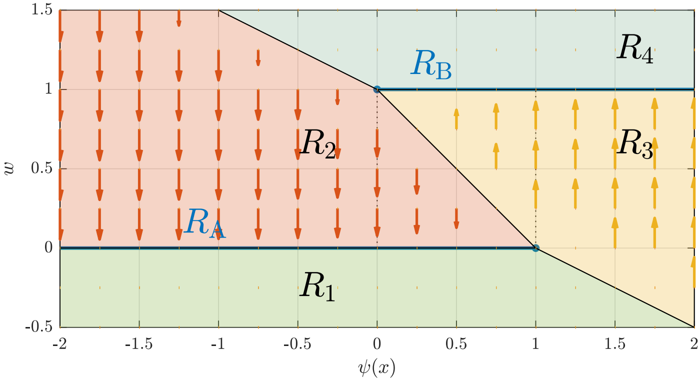
- Smart choice of auxiliary dynamics resolves the nonuniqueness issue

Time-freezing: auxiliary dynamics



- Smart choice of auxiliary dynamics resolves the nonuniqueness issue
- Zoom in: escape only in one direction possible

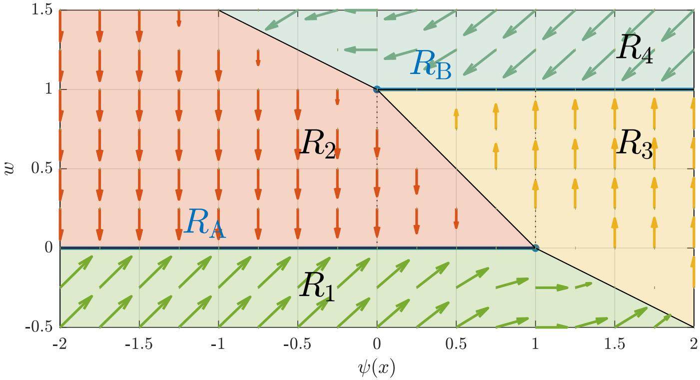
Stop the state jump and construct suitable sliding mode



- Dynamics in R_1 and R_4 stops evolution of auxiliary ODE - similar to inelastic impacts

Time-freezing: DAE forming dynamics

Stop the state jump and construct suitable sliding mode



- Dynamics in R_1 and R_4 stops evolution of auxiliary ODE - similar to inelastic impacts
- Sliding modes on $R_{\mathrm{A}} := \partial R_1 \cap \partial R_2$ and $R_{\mathrm{B}} := \partial R_3 \cap \partial R_4$ match $f_{\mathrm{A}}(y)$ and $f_{\mathrm{B}}(y)$, resp.

Time-freezing: summary

DAE-forming dynamics

$$y = (x, w, t)$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,A}}(y) \coloneqq \begin{bmatrix} 2 f_{\mathrm{A}}(x) \\ \gamma(\psi(x)) \\ 2 \end{bmatrix}$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,B}}(y) \coloneqq \begin{bmatrix} 2 f_{\mathrm{B}}(x) \\ -\gamma(\psi(x) - 1) \\ 2 \end{bmatrix}$$

- In total four regions R_i , $i = 1, 2, 3, 4$ and evolution of original system is the **sliding mode**

Time-freezing: summary

DAE-forming dynamics

$$y = (x, w, t)$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,A}}(y) \coloneqq \begin{bmatrix} 2 f_{\mathrm{A}}(x) \\ \gamma(\psi(x)) \\ 2 \end{bmatrix}$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,B}}(y) \coloneqq \begin{bmatrix} 2 f_{\mathrm{B}}(x) \\ -\gamma(\psi(x) - 1) \\ 2 \end{bmatrix}$$

- In total four regions R_i , $i = 1, 2, 3, 4$ and evolution of original system is the **sliding mode**
- Regions R_2 and R_3 equipped with aux. dynamics $y' = f_2(y) = f_{\mathrm{aux,A}}(y)$ and $y' = f_3(y) = f_{\mathrm{aux,B}}(y)$, resp., to mimic state jump

Time-freezing: summary

DAE-forming dynamics

$$y = (x, w, t)$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,A}}(y) \coloneqq \begin{bmatrix} 2f_{\mathrm{A}}(x) \\ \gamma(\psi(x)) \\ 2 \end{bmatrix}$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,B}}(y) \coloneqq \begin{bmatrix} 2f_{\mathrm{B}}(x) \\ -\gamma(\psi(x) - 1) \\ 2 \end{bmatrix}$$

- In total four regions R_i , $i = 1, 2, 3, 4$ and evolution of original system is the **sliding mode**
- Regions R_2 and R_3 equipped with aux. dynamics $y' = f_2(y) = f_{\mathrm{aux,A}}(y)$ and $y' = f_3(y) = f_{\mathrm{aux,B}}(y)$, resp., to mimic state jump
- Regions R_1 and R_4 equipped with DAE-forming dynamics $y' = f_1(y) = f_{\mathrm{df,A}}(y)$ and $y' = f_4(y) = f_{\mathrm{df,B}}(y)$, resp., to recover original dynamics in sliding mode

Time-freezing: summary

DAE-forming dynamics

$$y = (x, w, t)$$

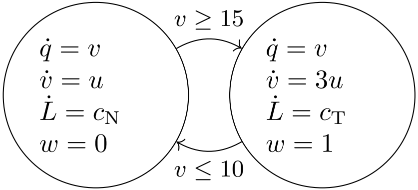
$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,A}}(y) := \begin{bmatrix} 2f_{\mathrm{A}}(x) \\ \gamma(\psi(x)) \\ 2 \end{bmatrix}$$

$$\frac{\mathrm{d}y}{\mathrm{d}\tau} = f_{\mathrm{df,B}}(y) := \begin{bmatrix} 2f_{\mathrm{B}}(x) \\ -\gamma(\psi(x) - 1) \\ 2 \end{bmatrix}$$

- In total four regions R_i , $i = 1, 2, 3, 4$ and evolution of original system is the **sliding mode**
- Regions R_2 and R_3 equipped with aux. dynamics $y' = f_2(y) = f_{\mathrm{aux,A}}(y)$ and $y' = f_3(y) = f_{\mathrm{aux,B}}(y)$, resp., to mimic state jump
- Regions R_1 and R_4 equipped with DAE-forming dynamics $y' = f_1(y) = f_{\mathrm{df,A}}(y)$ and $y' = f_4(y) = f_{\mathrm{df,B}}(y)$, resp., to recover original dynamics in sliding mode
- E.g., $w' = 0 \implies \theta_1 f_{\mathrm{df,A}}(y) + \theta_2 f_{\mathrm{aux,A}}(y) = f_{\mathrm{A}}(y)$ (sliding mode)
- Conclusion: we have a PSS and can treat it with FESD

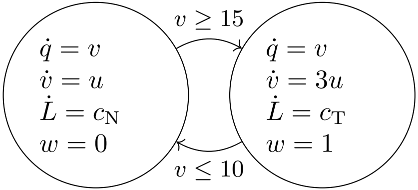
Time optimal control of a car with a turbo accelerator

Example from [Avraam, 2000] solved with NOSNOC



Time optimal control of a car with a turbo accelerator

Example from [Avraam, 2000] solved with NOSNOC

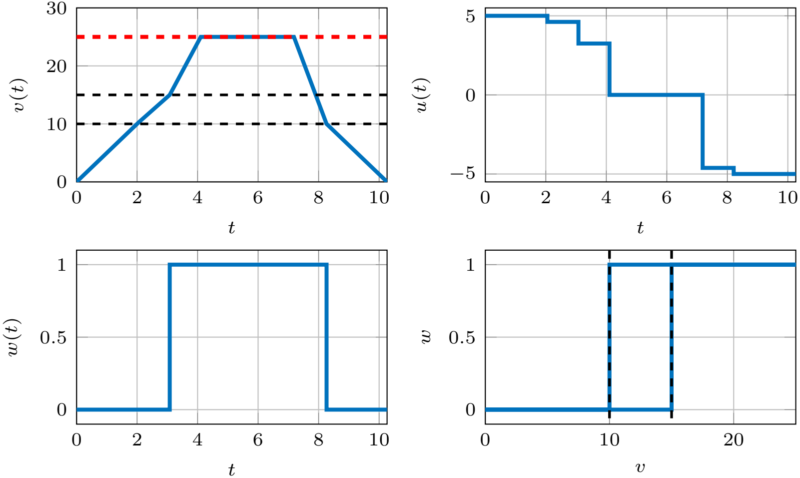


Time optimal control problem

$$
\begin{aligned}
\min_{y(\cdot),u(\cdot),s(\cdot)} \quad & t(\tau_{\mathrm{f}}) + L(\tau_{\mathrm{f}}) \\
\text{s.t.} \quad & y(0) = (z_0, 0) \\
& y'(\tau) \in s(\tau) F_{\mathrm{TF}}(y(\tau), u(\tau)) \\
& -\bar{u} \leq u(\tau) \leq \bar{u} \\
& \bar{s}^{-1} \leq s(\tau) \leq \bar{s} \\
& -\bar{v} \leq v(\tau) \leq \bar{v} \;\; \tau \in [0, \tau_{\mathrm{f}}] \\
& (q(\tau_{\mathrm{f}}), v(\tau_{\mathrm{f}})) = (q_{\mathrm{f}}, v_{\mathrm{f}})
\end{aligned}
$$

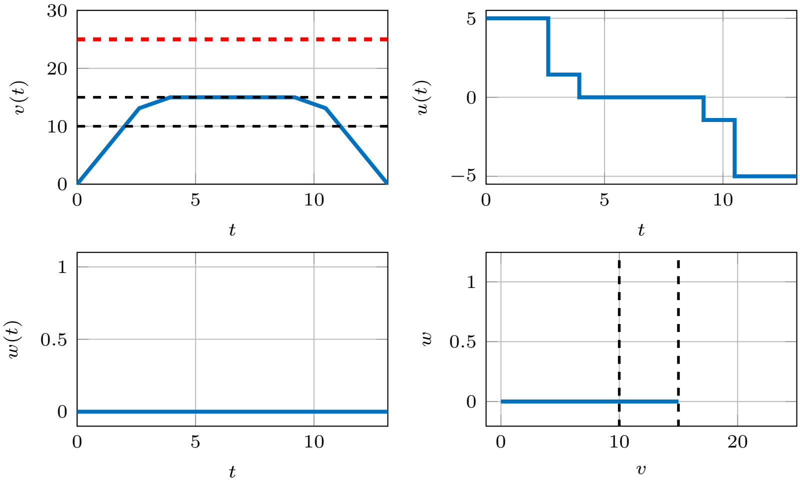
Scenario 1: turbo and nominal cost the same

$c_{\mathrm{N}} = c_{\mathrm{T}}$



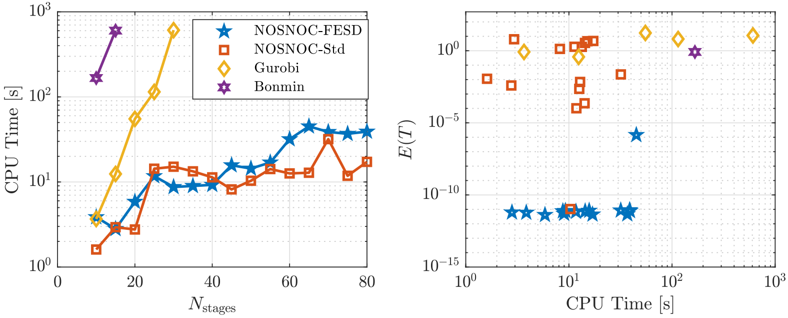
Scenario 2: Turbo is Expensive

$c_{\mathrm{N}} < c_{\mathrm{T}}$



NOSNOC vs MILP/MINLP formulations

Benchmark on time-optimal control problem of a car with turbo



- compare CPU time as function of number of control intervals N (left) and solution accuracy (right)
- MILP (Gurobi): solve problem with fixed T until indefeasibly happens with grid search in T
- MILP/MINLP and NOSNOC-Std no switch detection = low accuracy

Conclusions and outlook

Conclusions

- Finite Elements with Switch Detection (FESD) allow highly accurate simulation and optimal control for nonsmooth systems of level NSD2
- FESD resolves many of the issues that standard methods have: integration accuracy, convergence of sensitivities
- Main difficulty: solving the Mathematical Programs with Complementarity Constraints (MPCC)

Conclusions and outlook

Conclusions

- Finite Elements with Switch Detection (FESD) allow highly accurate simulation and optimal control for nonsmooth systems of level NSD2
- FESD resolves many of the issues that standard methods have: integration accuracy, convergence of sensitivities
- Main difficulty: solving the Mathematical Programs with Complementarity Constraints (MPCC)

Outlook

- Improve on MPCC methods, test other existing relaxation methods (work in progress, soon available in NOSNOC)
- Properties of FESD-MPCC solutions. Are all stationary points strongly stationary points?
- Combinatorial methods for MPCC arising in nonsmooth optimal control
- Efficient NCP solvers for FESD subproblems

References 1 (own, most related to this talk)

- A time-freezing approach for numerical optimal control of nonsmooth differential equations with state jumps.
  A. Nurkanovic, T. Sartor, S. Albrecht, and M. Diehl, IEEE Cont. Sys. Lett., 2021.
- NOSNOC: A software package for numerical optimal control of nonsmooth systems.
  A. Nurkanovic and M. Diehl, IEEE Cont. Sys. Lett., 2022.
- Continuous optimization for control of hybrid systems with hysteresis via time-freezing
  A. Nurkanovic and M. Diehl, IEEE Cont. Sys. Lett., 2022.
- Finite Elements with Switch Detection for Direct Optimal Control of Nonsmooth Systems.
  A. Nurkanovic, M. Sperl, S. Albrecht, and M. Diehl, arXiv preprint, 2022.

References 2 (external, theory and algorithms)

- Differential equations with discontinuous right-hand side.
  A. F. Filippov, Matematicheskii sbornik, 1960.
- MPEC strategies for optimization of a class of hybrid dynamic systems.
  B. Baumrucker and L. Biegler, Journal of Process Control, 2009.
- A high accuracy method for solving ODEs with discontinuous right-hand side.
  D. E. Stewart, Numerische Mathematik, 1990.
- Optimal control of systems with discontinuous differential equations.
  D. E. Stewart and M. Anitescu, Numerische Mathematik, 2010.

References 3 (external, software)

- On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming.
  A. Wächter and L. T. Biegler, Mathematical Programming, 2006.
- CasADi: a software framework for nonlinear optimization and optimal control.
  J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, Mathematical Programming Computation, 2019.

Thank you very much for your attention!