Hybrid vs. nonsmooth dynamical systems Why can it be so difficult to model physics in hybrid modelers? [1]

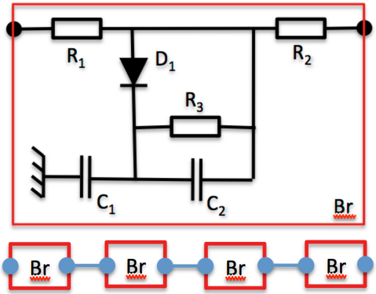
Benoît Caillaud

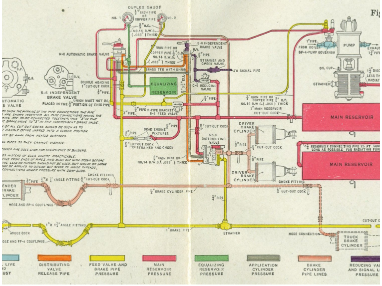
Inria
Rennes, France

Aussois, 1–5 December 2014

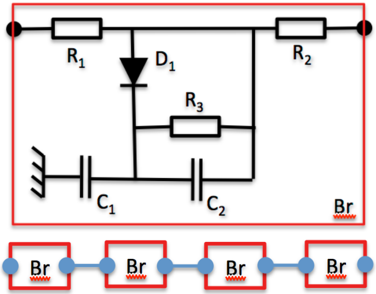
[1] Collaboration with A. Aljarbooh, A. Benveniste, T. Bourke et M. Pouzet.

Example: the Westinghouse brake

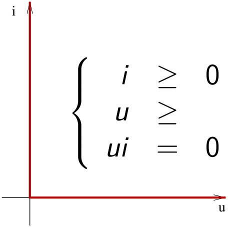




Example: the Westinghouse brake

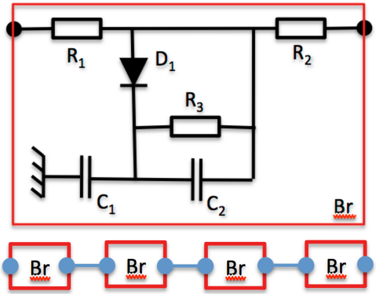


$$\left\{\begin{array}{rcl} N\mathbf{i} & = & \mathbf{0} \\ K\mathbf{u} & = & \mathbf{0} \\ \mathbf{u} - R\mathbf{i} & = & \mathbf{0} \\ C\dot{\mathbf{u}} - \mathbf{i} & = & \mathbf{0} \\ \forall i \quad 0 \leq \mathbf{i}_{D_i} & \perp & \mathbf{u}_{D_i} \geq 0 \end{array}\right.$$



$$\left\{\begin{array}{rcl} i & \geq & 0 \\ u & \geq & \\ ui & = & 0 \end{array}\right.$$

Example: the Westinghouse brake



$$
\left\{
\begin{array}{rcl}
N\mathbf{i} & = & \mathbf{0} \\
K\mathbf{u} & = & \mathbf{0} \\
\mathbf{u} - R\mathbf{i} & = & \mathbf{0} \\
C\dot{\mathbf{u}} - \mathbf{i} & = & \mathbf{0} \\
\forall i \quad 0 \leq \mathbf{i}_{D_i} & \perp & \mathbf{u}_{D_i} \geq 0
\end{array}
\right.
$$

- 2^n modes
- Orientation and scheduling are mode dependent
- Dymola/OpenModelica : need to introduce leakage and impedance in diodes

Nonsmooth dynamical systems

Dynamical system:

$$\dot{q} = A.q + r$$

Nonsmooth perturbation r, solution of (for instance) a linear complementarity problem (LCP) :

$$\left\{ \begin{array}{rcl} r & = & B.x \\ y & = & N.q + M.x \\ 0 \leq x & \perp & y \geq 0 \end{array} \right.$$

Other nonsmooth optimisation problems: MLCP, NLCP, QP

Siconos numerical library [1, 2]

Nonsmooth dynamical systems

Dynamical system:

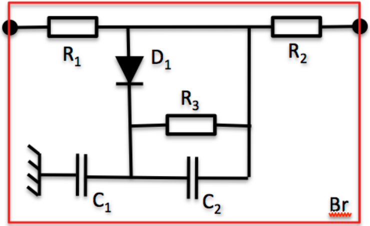
$$\dot{q} = A.q + r$$

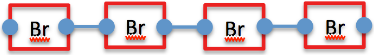
Nonsmooth perturbation r, solution of (for instance) a linear complementarity problem (LCP) :

$$\begin{cases} r &= B.x \\ y &= N.q + M.x \\ 0 \leq x &\perp y \geq 0 \end{cases}$$

Other nonsmooth optimisation problems: MLCP, NLCP, QP

Siconos numerical library [1, 2]





$$\begin{cases} N\mathbf{i} &= \mathbf{0} \\ K\mathbf{u} &= \mathbf{0} \\ \mathbf{u} - R\mathbf{i} &= \mathbf{0} \\ C\dot{\mathbf{u}} - \mathbf{i} &= \mathbf{0} \\ \forall i \quad 0 \leq \mathbf{i}_{D_i} &\perp \mathbf{u}_{D_i} \geq 0 \end{cases}$$

Siconos

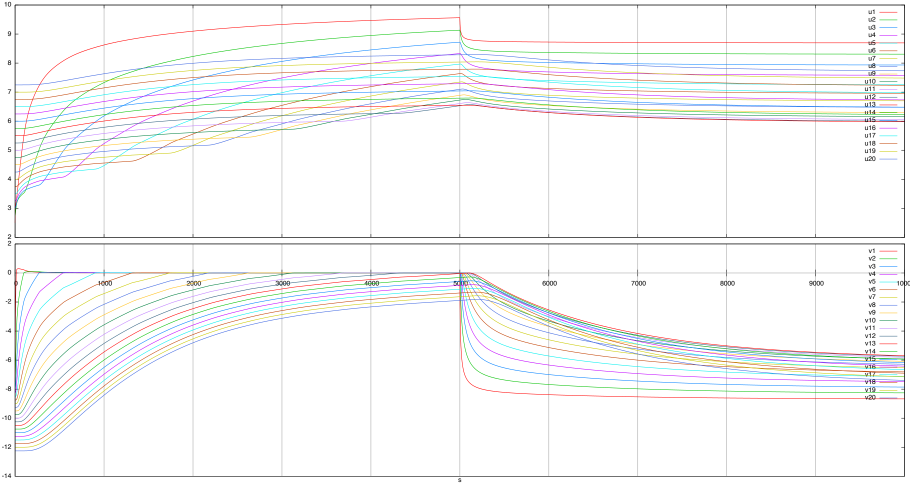
A software for modeling and simulation of nonsmooth dynamical systems

- C++ numerical library
- Several layers:
  - Kernel: nonsmooth optimization, LCP, MLCP (relay), NLCP, Newton impact law, QP
  - Numerics: system structure, discretization schemes, (N)LCS, MDI
  - Mechanics: high level multibody mechanics
- `http://siconos.gforge.inria.fr`

The Westinghouse brake in Siconos

```
#include "SiconosKernel.hpp"

int main(int argc, char * argv[])
{
  ...; // Define constants and initial state

  // --- Model definition ---

  SP::DynamicalSystemsSet NSDS(new DynamicalSystemsSet());

  // NSDS = linear time-invariant dynamical system (LTIDS) +
  //        linear complementarity problem (LCP)

  // LTIDS: q' = A.q + b + r
  SP::SiconosVector q0(new SiconosVector(2*n));
  SP::SimpleMatrix A(new SimpleMatrix(2*n,2*n));
  SP::SiconosVector b(new SiconosVector(2*n));
  ...; // Initializing q0, A and b
  SP::FirstOrderLinearTIDS RC(new FirstOrderLinearTIDS(q0,A,b));

  // LCP = linear time-inv. relation + complementarity cond.
  // y = N.q + M.x and r = B.x and 0 <= y ortho x >= 0
  SP::SimpleMatrix B(new SimpleMatrix(2*n,n));
  SP::SimpleMatrix M(new SimpleMatrix(n,n));
  SP::SimpleMatrix N(new SimpleMatrix(n,2*n));
  ...; // Initializing B, M and N
  SP::FirstOrderLinearTIR LTIR(new FirstOrderLinearTIR(*N,*B));
  LTIR->setDPtr(M);

  SP::NonSmoothLaw CC(new ComplementarityConditionNSL(n));

  // --- Connecting pieces ---

  NSDS->insert(RC);
  SP::Interaction Inter(new Interaction("I",*NSDS,1,n,CC,LTIR));
  SP::Model Circ(new Model(t0,T,Modeltitle));
  Circ->nonSmoothDynamicalSystem()->insertDynamicalSystem(RC);
  Circ->nonSmoothDynamicalSystem()->link(Inter,RC);

  // --- Initialize simulation ---

  SP::Moreau OSI(new Moreau(RC,theta)); // Moreau sweeping
  SP::TimeDiscretisation TiDisc(new TimeDiscretisation(t0,tau));
  SP::LCP OSNSP(new LCP()); // Select LCP solver
  SP::TimeStepping Strat(new TimeStepping(TiDisc,OSI,OSNSP));
  Circ->initialize(Strat); // Initialize scheme

  // --- Simulation ---

  for(k = 1 ; k < Ns ; ++k) // Time stepping
    {
      ...; // Get input
      Strat->computeOneStep(); // Solve
      Strat->nextStep(); // Transfer curr. state into last state
    }
}
```

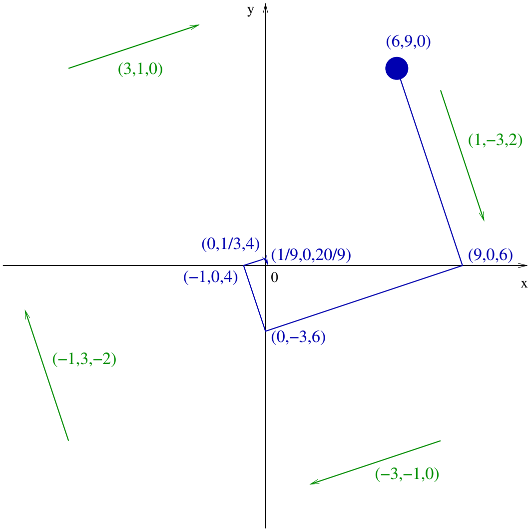
The Westinghouse brake in Siconos



Filippov differential inclusions: Zeno + sliding modes

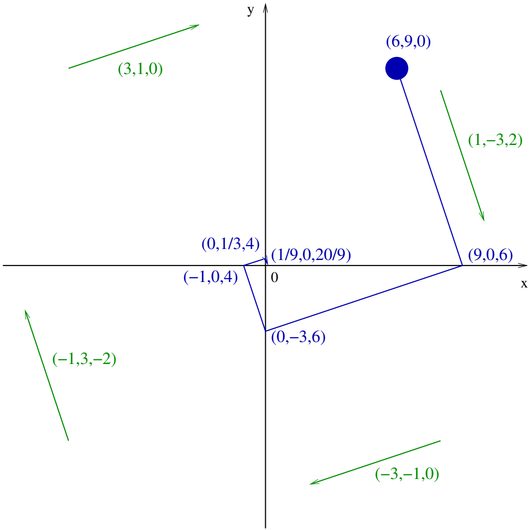
$$\begin{cases} \dot{x} & = & -\mathrm{sgn}(x) + 2\mathrm{sgn}(y) \\ \dot{y} & = & -2\mathrm{sgn}(x) - \mathrm{sgn}(y) \\ \dot{z} & = & \mathrm{sgn}(x) + \mathrm{sgn}(y) \end{cases}$$

Filippov differential inclusions: Zeno + sliding modes



$$
\begin{cases}
\dot{x} &= -\mathrm{sgn}(x) + 2\mathrm{sgn}(y) \\
\dot{y} &= -2\mathrm{sgn}(x) - \mathrm{sgn}(y) \\
\dot{z} &= \mathrm{sgn}(x) + \mathrm{sgn}(y)
\end{cases}
$$

Filippov differential inclusions: Zeno + sliding modes



$$\begin{cases} \dot{x} &= -\mathrm{sgn}(x) + 2\mathrm{sgn}(y) \\ \dot{y} &= -2\mathrm{sgn}(x) - \mathrm{sgn}(y) \\ \dot{z} &= \mathrm{sgn}(x) + \mathrm{sgn}(y) \end{cases}$$

- Square "spiral"
- Attractor : line $x = y = 0$
- Zeno à $t = 7.5$
- Compute z at $t = 10$?

Filippov differential inclusions: Zeno + sliding modes

Filippov differential inclusions:

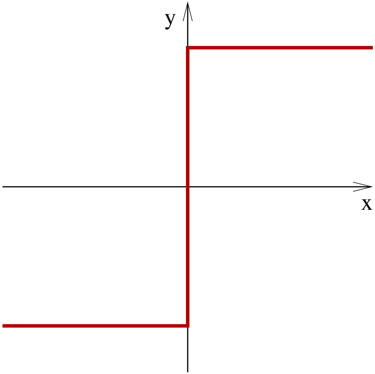
$$\dot{x} = A.x + B.\sigma$$

with:

$$\sigma \in \mathrm{sgn}(C.x)$$

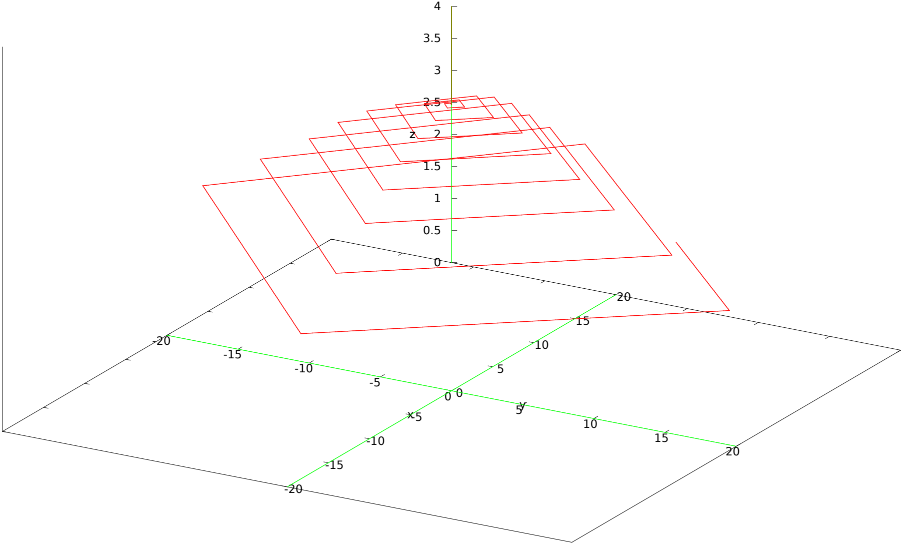
reduction to a Mixed-LCP. Averaging σ over discretization step [3]:

$$\begin{cases} M.z + q = w - v \\ -1 \leq z \leq 1 \\ (1+z).w = 0 \\ (1-z).v = 0 \\ w, v \geq 0 \end{cases}$$



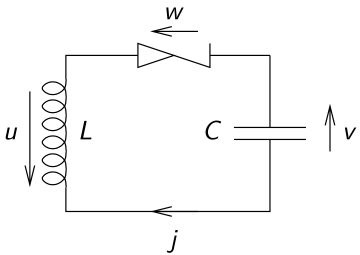
$$\sigma \in \mathrm{sgn}(y)$$
$$\Longleftrightarrow$$
$$y \in N_{[-1,1]}(\sigma) =$$
$$\{y | \forall \alpha \in [-1, 1], y(\sigma - \alpha) \geq 0\}$$

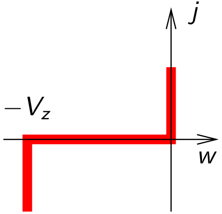
Filippov differential inclusions: Zeno + sliding modes



Detailed example: LZC

$$\begin{cases} j' &= (1/L)u \\ v' &= (1/C)j \\ 0 &= u + v + w \\ j &= j_1 - j_2 \\ w_1 &= -w \\ w_2 &= w + V_z \\ 0 \le j_1 &\perp w_1 \ge 0 \\ 0 \le j_2 &\perp w_2 \ge 0 \end{cases}$$





Detailed example: LZC

Put system into the following form:

$$\begin{cases} q' = Aq + r \\ r = Bx \\ y = Nq + Mx + d \\ 0 \leq x \perp y \geq 0 \end{cases}$$

Define:

$$q = \begin{bmatrix} j \\ v \end{bmatrix} \qquad x = \begin{bmatrix} w_1 \\ j_2 \end{bmatrix} \qquad y = \begin{bmatrix} j_1 \\ w_2 \end{bmatrix}$$

This gives:

$$A = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & 0 \end{bmatrix} \qquad B = \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & 0 \end{bmatrix}$$

$$N = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \qquad M = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad d = \begin{bmatrix} 0 \\ V_z \end{bmatrix}$$

Detailed example: LZC

Discretization: implicit Euler

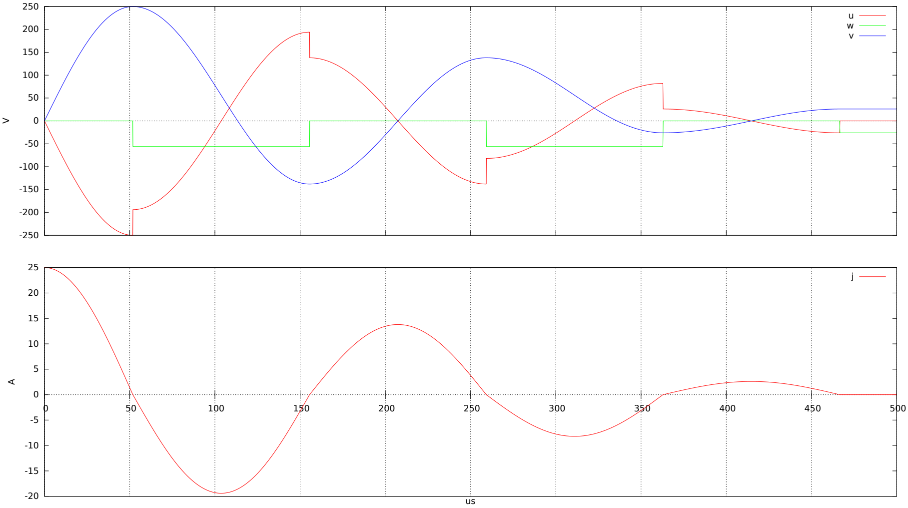
$$\frac{1}{h}(q_{n+1} - q_n) \approx Aq_{n+1} + r_{n+1}$$

With:

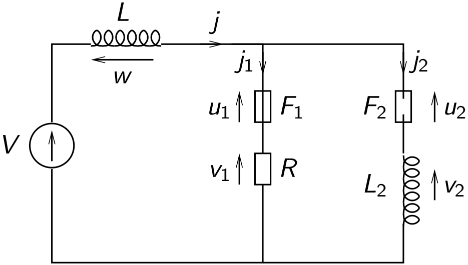
$$\begin{cases} r_{n+1} = Bx_{n+1} \\ y_{n+1} = Nq_{n+1} + Mx_{n+1} + d \\ 0 \leq x_{n+1} \perp y_{n+1} \geq 0 \end{cases}$$

Solve LCP at each step (after elimination of q_{n+1} in system above)

Detailed example: LZC

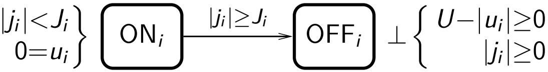


The two fuses example: Relative degree of a LCS



- Invariant:
  $j_1, j_2, u_1, u_2 \geq 0$
- Smooth dynamics:

$$\left\{ \begin{array}{rcl} j' & = & \frac{1}{L} w \\ j_2' & = & \frac{1}{L_2} v_2 \\ j_1 & = & j - j_2 \\ v_1 & = & R_1 j_1 \\ u_1 + v_1 & = & u_2 + v_2 \\ & = & V - w \end{array} \right.$$



$$\left\{ \begin{array}{l} 0 \leq j_1 \quad \perp \quad z_1 - u_1 \geq 0 \\ 0 \leq j_2 \quad \perp \quad z_2 - u_2 \geq 0 \\ \text{when } j_1 \geq J_1 \text{ do } z_1 := U_1 \\ \text{when } j_2 \geq J_2 \text{ do } z_2 := U_2 \end{array} \right.$$

The two fuses example: Relative degree of a LCS

- Put system in LCS form:

$$\begin{cases} \dot{x} & = & Ax + b + r \\ r & = & B\lambda \\ y & = & Cx + D\lambda + e \\ & & 0 \leq y \perp \lambda \geq 0 \end{cases}$$

- Choose:

$$x = \begin{bmatrix} j \\ j_2 \end{bmatrix}$$
$$y = \begin{bmatrix} j_1 \\ j_2 \end{bmatrix}$$
$$\lambda = \begin{bmatrix} z_1 - u_1 \\ z_2 - u_2 \end{bmatrix} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix}$$

- Difficulty: $D = 0$
- LCS is ill-defined: LCP does not have a unique solution
- Derivation of the LCP gives:

$$\begin{cases} \dot{y} = CAx + Cb + CB\lambda \\ y = 0 \ \Rightarrow \ 0 \leq \dot{y} \perp \lambda \geq 0 \\ y > 0 \ \Rightarrow \ \lambda = 0 \end{cases}$$

- Relative degree $= 1$
- CB is non-singular: derived NS law has a unique solution

Conclusion

- Need to design modeling languages with nonsmooth dynamics during continuous-time evolution
- Paradigm shift:

  "Hybrid = Continuous + Discrete" to
  "Hybrid = Nonsmooth + Discrete"
- Time-stepping methods enable simulation:
  - past Zeno points
  - chattering-free sliding modes
  - Main drawback: slow convergence $O(h)$, unlike high order numerical schemes
  - Mixed adaptive event-driven / time-stepping methods
- Solutions of Filippov differential inclusions and (N)LCS are:
  - absolutely continuous
  - almost everywhere differentiable
- zero-Xing detection is as usual
- Relative degree of NSDS $\approx$ DAE index
- Not developed in this talk: Measure differential inclusions (required for Newton impact law)

References

V. Acary and B. Brogliato.
Numerical methods for nonsmooth dynamical systems.
Springer, 2008.

Vincent Acary, Olivier Bonnefon, and Bernard Brogliato.
Nonsmooth modeling and simulation for switched circuits, volume 69 of *Lecture notes in electrical engineering*.
Springer, 2011.

Vincent Acary and Bernard Brogliato.
Implicit euler numerical scheme and chattering-free implementation of sliding mode systems.
Systems and Control Letters, 59:284–293, 2010.