Formalisation Tools for Classical Analysis: A Case Study in Control Theory

Damien Rouhling

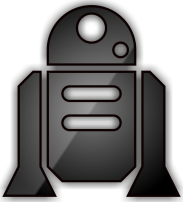
Université Côte d'Azur, Inria, France

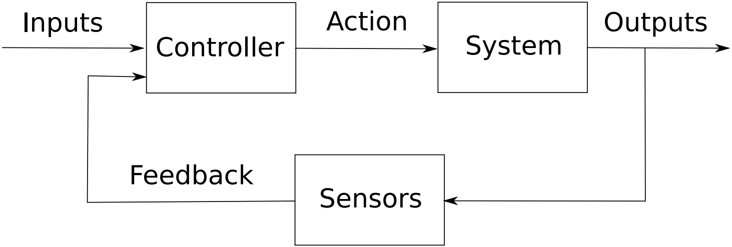
September 30, 2019

Dissecting the title

Formalisation Tools for Classical Analysis:
A Case Study in Control Theory

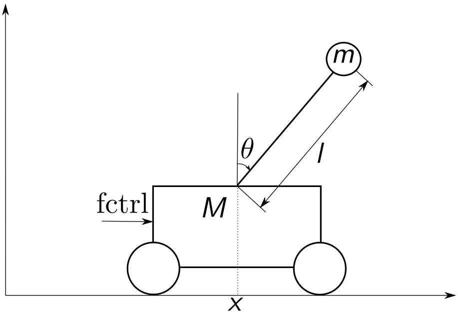
Formalisation Tools for Classical Analysis: A Case Study in Control Theory





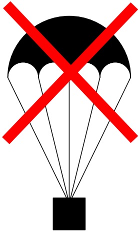
Formalisation Tools for Classical Analysis: A Case Study in Control theory

The inverted pendulum is a standard example for testing control techniques.



- Goal: stabilize the pendulum on its unstable equilibrium.
- Control function: force fctrl applied to the cart.

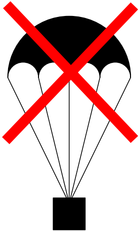
Free fall

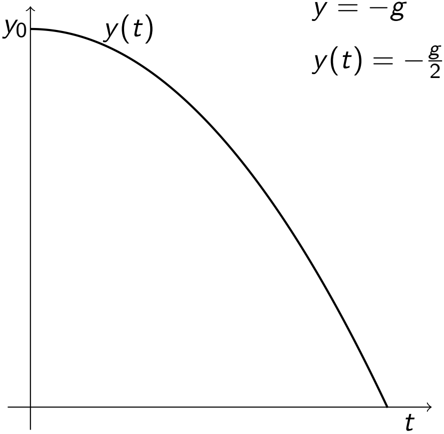


$$\ddot{y} = -g$$

Formalisation Tools for Classical Analysis: A Case study in Control theory

Free fall

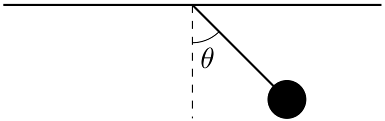




$$\ddot{y} = -g$$

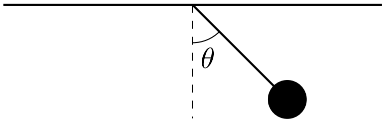
$$y(t) = -\frac{g}{2}t^2 + y_0$$

Pendulum



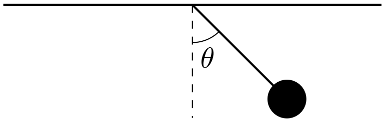
$$\ddot{\theta} + \frac{g}{l} \sin \theta = 0$$

Pendulum



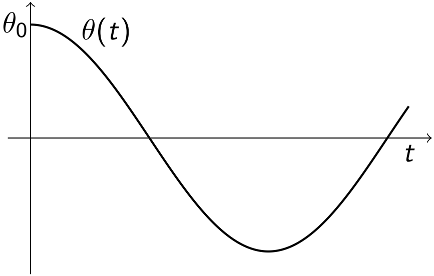
$$\ddot{\theta} + \frac{g}{l}\theta = 0$$

Pendulum



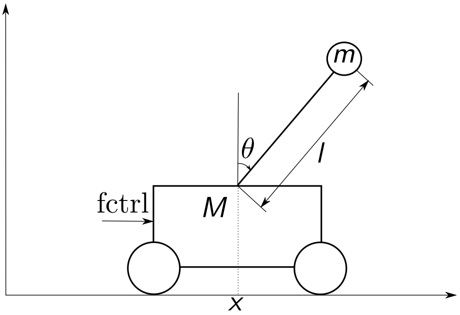
$$\ddot{\theta} + \frac{g}{l}\theta = 0$$

$$\theta(t) = \theta_0 \cos\left(\sqrt{\frac{g}{l}}t\right)$$



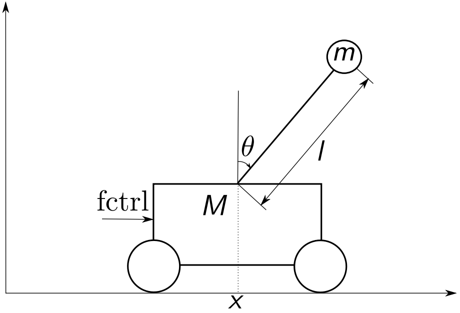
Inverted Pendulum

$$M(q)\ddot{q} + C\left(q, \dot{q}\right)\dot{q} + G\left(q\right) = \tau$$



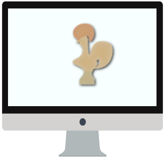
Inverted Pendulum

$$M(q)\ddot{q} + C\left(q, \dot{q}\right)\dot{q} + G\left(q\right) = \tau$$

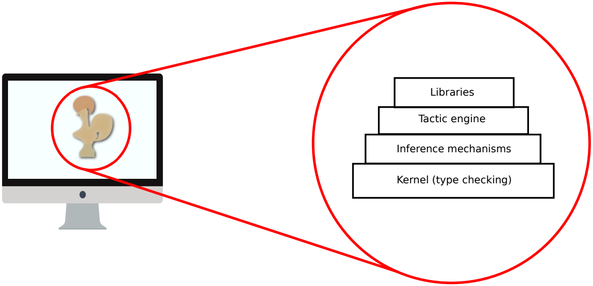


Lozano, Fantoni, Block (2000)

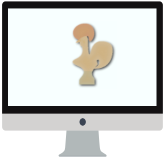
Formalisation Tools for Classical Analysis: A Case study in Control theory

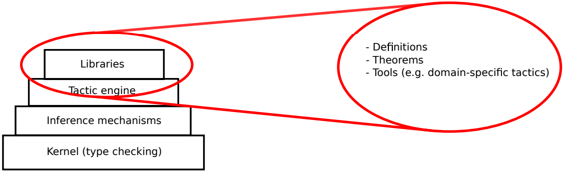


Formalisation Tools for Classical Analysis: A Case study in Control theory



Formalisation Tools for Classical Analysis: A Case study in Control theory





Formalisation tools for Classical Analysis: A Case study in Control theory

- We obtain different logics by selecting different allowed reasoning steps.
- Classical reasoning allows for standard reasoning steps in mathematics: proof by contradiction, excluded middle, the axiom of choice are allowed.

Formalisation tools for Classical Analysis: A Case study in Control theory

- We obtain different logics by selecting different allowed reasoning steps.
- Classical reasoning allows for standard reasoning steps in mathematics: proof by contradiction, excluded middle, the axiom of choice are allowed.

LaSalle (1960):

> We say also that $x(t)$ *approaches a set* M as t approaches infinity, if each $\epsilon > 0$ there is a $T > 0$ with the property that for each $t > T$ there is a p in M with $||x(t) - p|| < \epsilon$; that is, for all $t > T$ the points $x(t)$ are within a distance ϵ of M. For instance, if $x(t)$ is bounded for $t > 0$, then $x(t)$ approaches its positive limiting set Γ^+ as $t \to \infty$. If this were not so, there would be an $\epsilon > 0$ with the property that for each $T > 0$ there is a $t > T$, such that $||x(t) - p|| \geq \epsilon$ for all p in Γ^+. Hence, there would be a sequence t_n tending to infinity with n and such that $||x(t_n) - p|| \geq \epsilon$ for all p in Γ^+. But since $x(t)$ is bounded for $t \geq 0$, the sequence $x(t_n)$ has a limit point which is in Γ^+, which is a contradiction. This proves the proposition. The way in

Contributions

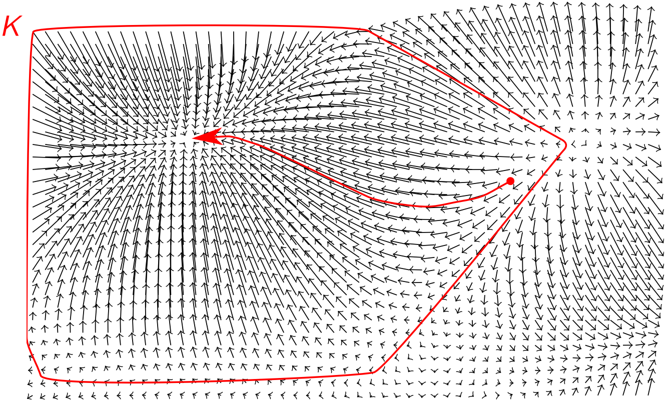
- Formal proof of soundness of a control function for the inverted pendulum.
  - Formal proof of a generalised version of LaSalle's invariance principle.
    In collaboration with Cyril Cohen.
  - Application of the formal version of LaSalle's invariance principle to the inverted pendulum.
- Reusable tools for formal proofs in: topology, asymptotic reasoning, analysis in higher dimensions.
  - A new library for classical analysis in Coq: Mathematical Components Analysis.
    In collaboration with Reynald Affeldt, Cyril Cohen, Assia Mahboubi and Pierre-Yves Strub.
- A modular methodology for proofs by computation.
  In collaboration with Cyril Cohen.

Contributions

- Formal proof of soundness of a control function for the inverted pendulum.
  - Formal proof of a generalised version of LaSalle's invariance principle.
    In collaboration with Cyril Cohen.
  - Application of the formal version of LaSalle's invariance principle to the inverted pendulum.
- Reusable tools for formal proofs in: topology, asymptotic reasoning, analysis in higher dimensions.
  - A new library for classical analysis in Coq: Mathematical Components Analysis.
    In collaboration with Reynald Affeldt, Cyril Cohen, Assia Mahboubi and Pierre-Yves Strub.
- A modular methodology for proofs by computation.
  In collaboration with Cyril Cohen.

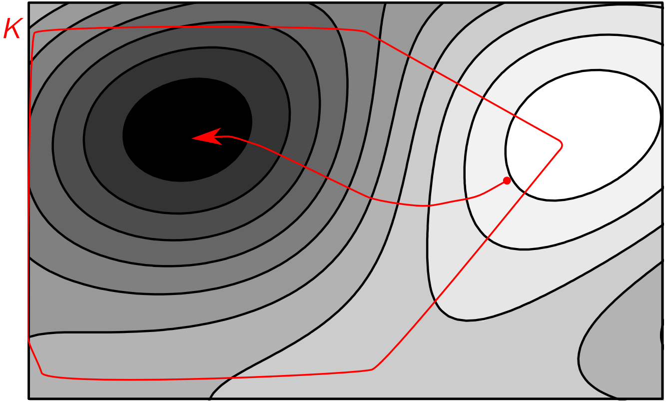
Intuition of LaSalle's invariance principle

The differential equation as a vector field: $\dot{y} = F(y)$.



Intuition of LaSalle's invariance principle

Contour map of a Lyapunov function V:



Intuition of LaSalle's invariance principle

Requirements:

- Invariant compact set K.
- Lyapunov function V.
- Regularity assumptions.

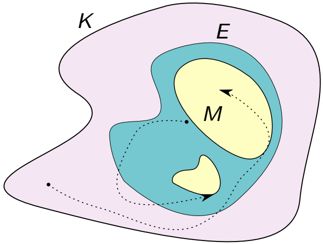
Conclusion:
The solutions of $\dot{y} = F(y)$ starting in K converge to a set where $\dot{V} = 0$.

My work on LaSalle's invariance principle

- Generalise the principle:
  - Weaken the hypotheses: $F(0) = 0$, regularity assumptions, $\mathbb{R}^n$.

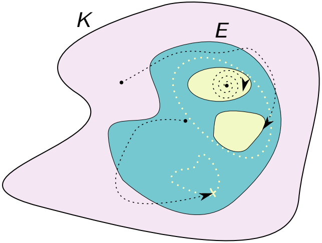
My work on LaSalle's invariance principle

- Generalise the principle:
  - Weaken the hypotheses: $F(0) = 0$, regularity assumptions, $\mathbb{R}^n$.
  - Strengthen the conclusion.



My work on LaSalle's invariance principle

- Generalise the principle:
  - Weaken the hypotheses: $F(0) = 0$, regularity assumptions, $\mathbb{R}^n$.
  - Strengthen the conclusion.



My work on LaSalle's invariance principle

- Generalise the principle:
  - Weaken the hypotheses: $F(0) = 0$, regularity assumptions, $\mathbb{R}^n$.
  - Strengthen the conclusion.
- Fill the gaps in the proof.

LaSalle (1960):

> infinity with n and such that $x(t_n) \to p$ as $n \to \infty$. One of the fundamental properties of limiting sets is the following:
> *If $x(t)$ is bounded for $t \geq 0$, then its positive limiting set Γ^+ is a nonempty, compact, invariant set.*

My work on LaSalle's invariance principle

- Generalise the principle:
  - Weaken the hypotheses: $F(0) = 0$, regularity assumptions, $\mathbb{R}^n$.
  - Strengthen the conclusion.
- Fill the gaps in the proof.
- Formalise topological notions: compact sets and closed sets.

LaSalle (1960):

Proof. Let $x(t)$ be a solution initially in Ω. Since $\dot{V}(x) \leq 0$ in Ω, $V(x(t))$ is a nonincreasing function of t. $V(x)$, being continuous on the compact set Ω, is bounded from below on Ω. Therefore, $V(x(t))$ has a limit c as $t \to \infty$. Note also that the positive limiting set Γ^+ is in Ω (because Ω is a closed set), and since V is continuous on Ω, $V(x) \equiv c$ on Γ^+. Γ^+ is an invariant set, and hence $\dot{V}(x) = 0$ on Γ^+. Thus, Γ^+ is in M. This implies, as was pointed out above, that $x(t) \to M$ as $t \to \infty$. All solutions starting in Ω approach M as t approaches infinity.

My work on LaSalle's invariance principle

- Generalise the principle:
  - Weaken the hypotheses: $F(0) = 0$, regularity assumptions, $\mathbb{R}^n$.
  - Strengthen the conclusion.
- Fill the gaps in the proof.
- Formalise topological notions: compact sets and closed sets.
- Develop notations for limits.

LaSalle (1960):

> infinity with n and such that $x(t_n) \to p$ as $n \to \infty$. One of the fundamental properties of limiting sets is the following:
> *If $x(t)$ is bounded for $t \geq 0$, then its positive limiting set Γ^+ is a nonempty, compact, invariant set.*

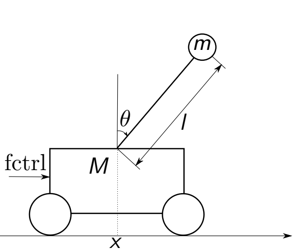
`f @ x --> y, lim (f @ x), cvg (f @ +oo), u --> -oo`

`x \o t --> p` or `(x \o t) @ \oo --> p`

Contributions

- Formal proof of soundness of a control function for the inverted pendulum.
  - Formal proof of a generalised version of LaSalle's invariance principle.
    In collaboration with Cyril Cohen.
  - Application of the formal version of LaSalle's invariance principle to the inverted pendulum.
- Reusable tools for formal proofs in: topology, asymptotic reasoning, analysis in higher dimensions.
  - A new library for classical analysis in Coq: Mathematical Components Analysis.
    In collaboration with Reynald Affeldt, Cyril Cohen, Assia Mahboubi and Pierre-Yves Strub.
- A modular methodology for proofs by computation.
  In collaboration with Cyril Cohen.

Soundness property for a control function



- Goal: the pendulum converges to its unstable equilibrium.
- Properties proven by Lozano, Fantoni and Block:
  - The pendulum converges to a homoclinic orbit:

$$\frac{1}{2}ml^2\dot{\theta}^2 = mgl\,(1-\cos\theta)\,.$$

  - The cart converges to its initial position: $x = 0$ and $\dot{x} = 0$.

Proof of soundness

Property: the pendulum converges to a set where

$$\frac{1}{2}ml^2\dot{\theta}^2 = mgl\,(1-\cos\theta) \quad \text{and} \quad x = 0 \quad \text{and} \quad \dot{x} = 0.$$

Proof: LaSalle's invariance principle with a well-chosen Lyapunov function V and a well-chosen compact set K.

Proof of soundness (cont.)

- The laws of Physics give a second-order differential equation. We transform the equation on (x, θ) into a first-order equation on

$$p = (p_0, p_1, p_2, p_3, p_4) = \left(x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}\right).$$

Proof of soundness (cont.)

- The laws of Physics give a second-order differential equation. We transform the equation on (x, θ) into a first-order equation on

$$p = (p_0, p_1, p_2, p_3, p_4) = \left(x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}\right).$$

$\Rightarrow$ Take into account the relation between the variables, e.g.:

$$\dot{p}_0 = p_1.$$

Proof of soundness (cont.)

- The laws of Physics give a second-order differential equation. We transform the equation on (x, θ) into a first-order equation on

$$p = (p_0, p_1, p_2, p_3, p_4) = \left(x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}\right).$$

$\Rightarrow$ Take into account the relation between the variables, e.g.:

$$\dot{p_0} = p_1.$$

- We still lose pieces of information. The invariant compact set K will help keeping them as invariants.

$$K = \left\{p \in \mathbb{R}^5 \mid p_2^2 + p_3^2 = 1 \text{ and } V(p) \leqslant k_0\right\}.$$

Correcting the proof

Errors encountered:

- Forgotten constant.
- Circular dependency.
- Wrong manipulation of equations:

$$\forall t \in I.\ f(t) = g(t) \quad \Rightarrow \quad \forall t \in I.\ \dot{f}(t) = \dot{g}(t).$$

Correcting the proof

Errors encountered:

- Forgotten constant.
- Circular dependency.
- Wrong manipulation of equations:

$$\forall t \in I.\ f(t) = g(t) \quad \Rightarrow \quad \forall t \in I.\ \dot{f}(t) = \dot{g}(t).$$

If I is not reduced to a point.

Correcting the proof

Errors encountered:

- Forgotten constant.
- Circular dependency.
- Wrong manipulation of equations:

$$\forall t \in I.\ f(t) = g(t) \quad \Rightarrow \quad \forall t \in I.\ \dot{f}(t) = \dot{g}(t).$$

If I is not reduced to a point.

Consequences: minor adaptations and the necessity to find a new proof for some points.

Soundness theorem for the inverted pendulum

$$\frac{1}{2}ml^2\dot{\theta}^2 = mgl\left(1-\cos\theta\right) \quad \text{and} \quad x = 0 \quad \text{and} \quad \dot{x} = 0.$$

$$p = (p_0, p_1, p_2, p_3, p_4) = \left(x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}\right).$$

```
Definition homoclinic_orbit :=
  [set p : 'rV[R]_5 | p[0] = 0 ∧ p[1] = 0 ∧
   (1 / 2) * m * (l ^ 2) * (p[4] ^ 2) = ...].
```

Soundness theorem for the inverted pendulum

$$\frac{1}{2}ml^2\dot{\theta}^2 = mgl\,(1-\cos\theta) \quad \text{and} \quad x = 0 \quad \text{and} \quad \dot{x} = 0.$$

$$p = (p_0, p_1, p_2, p_3, p_4) = \left(x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}\right).$$

```
Definition homoclinic_orbit :=
  [set p : 'rV[R]_5 | p[0] = 0 ∧ p[1] = 0 ∧
   (1 / 2) * m * (l ^ 2) * (p[4] ^ 2) = ...].
```

```
Lemma cvg_to_homoclinic_orbit (p : 'rV[R]_5) :
  p ∈ K -> sol p @ +oo --> homoclinic_orbit.
```

Soundness theorem for the inverted pendulum

$$\frac{1}{2}ml^2\dot{\theta}^2 = mgl\left(1-\cos\theta\right) \quad \text{and} \quad x = 0 \quad \text{and} \quad \dot{x} = 0.$$

$$p = (p_0, p_1, p_2, p_3, p_4) = \left(x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}\right).$$

```
Definition homoclinic_orbit :=
  [set p : 'rV[R]_5 | p[0] = 0 ∧ p[1] = 0 ∧
   (1 / 2) * m * (l ^ 2) * (p[4] ^ 2) = ...].
```

```
Lemma cvg_to_homoclinic_orbit (p : 'rV[R]_5) :
  p ∈ K -> sol p @ +oo --> homoclinic_orbit.
```

Cauchy-Lipschitz / Picard-Lindelöf

A few aspects of the formalisation

- Formalisation of $\mathbb{R}^n$.
  $\Rightarrow$ Combination of the Coquelicot and Mathematical Components libraries.

A few aspects of the formalisation

- Formalisation of $\mathbb{R}^n$.
  $\Rightarrow$ Combination of the Coquelicot and Mathematical Components libraries.
- Topological spaces and Tychonoff's Theorem (extra).

A few aspects of the formalisation

- Formalisation of $\mathbb{R}^n$.
  $\Rightarrow$ Combination of the Coquelicot and Mathematical Components libraries.
- Topological spaces and Tychonoff's Theorem (extra).
- Tools for automatic computation of differentials/derivatives.

Lozano, Fantoni, Block (2000):

The Lyapunov function candidate (13) becomes

$$V = \frac{k_E}{2}E^2 + \frac{k_v}{2}z_2^2 + \frac{k_x}{2}z_1^2. \tag{29}$$

which leads to

$$\dot{V} = -k_{\mathrm{d}x}z_2^2. \tag{32}$$

Automatic computation of differentials/derivatives

Goal: Prove that the derivative at point `x` of `fun y => 1 + y` is 1.

Automatic computation of differentials/derivatives

Goal: Prove that the derivative at point `x` of `fun y => 1 + y` is `1`.

```
==============================
is_derive (fun y => 1 + y) x 1
```

`auto_derive`

Automatic computation of differentials/derivatives

Goal: Prove that the derivative at point `x` of `fun y => 1 + y` is 1.

```
============================
is_derive (fun y => 1 + y) x 1
```

`evar_last`

Automatic computation of differentials/derivatives

Goal: Prove that the derivative at point `x` of `fun y => 1 + y` is 1.

```
2 subgoals

  ?d : R
  ============================
  is_derive (fun y => 1 + y) x ?d

subgoal 2 is:
  ?d = 1
```

```
Lemma is_derive_plus (f g : K -> V) (x : K) (df dg : V) :
  is_derive f x df -> is_derive g x dg ->
  is_derive (fun y => f y + g y) x (df + dg).
```

Automatic computation of differentials/derivatives

Goal: Prove that the derivative at point `x` of `fun y => 1 + y` is 1.

```
3 subgoals

  ?d₁, ?d₂ : R
  ============================
  is_derive (fun _ => 1) x ?d₁

subgoal 2 is:
  is_derive id x ?d₂

subgoal 3 is:
  ?d₁ + ?d₂ = 1
```

```
Lemma is_derive_const (a : V) (x : K) :
  is_derive (fun _ : K => a) x 0.
```

Automatic computation of differentials/derivatives

Goal: Prove that the derivative at point `x` of `fun y => 1 + y` is 1.

```
2 subgoals

  ?d₁, ?d₂ : R
  ==============================
  is_derive id x ?d₂

subgoal 2 is:
  0 + ?d₂ = 1
```

```
Lemma is_derive_id (x : K) :
  is_derive (fun t : K => t) x 1.
```

Automatic computation of differentials/derivatives

```
==============================
0 + 1 = 1
```

Automatic computation of differentials/derivatives

Using a type class `deriv` to store `is_derive_plus`, `is_derive_const` and `is_derive_id` in a data base of differentiation rules, we automatically transform

```
  ==============================
  is_derive (fun y => 1 + y) x 1
```

into

```
  ==============================
  0 + 1 = 1
```

thanks to

```
Lemma deriv_eq (f : K -> V) (x : K) (df' df : V) :
  deriv f x df' -> df' = df -> deriv f x df.
```

Type classes for the computation of differentials/derivatives

- Lightweight implementation.
- Easy to extend with new rules.
- Palliate the lack of an `auto_diff` tactic.
- Possibility to adapt the implementation to avoid giving the value explicitely.

Contributions

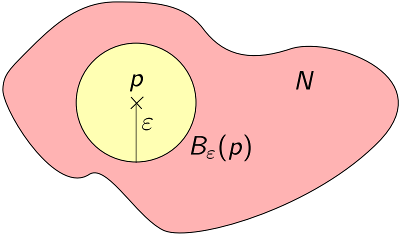
- Formal proof of soundness of a control function for the inverted pendulum.
  - Formal proof of a generalised version of LaSalle's invariance principle.
    In collaboration with Cyril Cohen.
  - Application of the formal version of LaSalle's invariance principle to the inverted pendulum.
- Reusable tools for formal proofs in: topology, asymptotic reasoning, analysis in higher dimensions.
  - A new library for classical analysis in Coq: Mathematical Components Analysis.
    In collaboration with Reynald Affeldt, Cyril Cohen, Assia Mahboubi and Pierre-Yves Strub.
- A modular methodology for proofs by computation.
  In collaboration with Cyril Cohen.

The Mathematical Components Analysis library

- Classical analysis.
- Inspired from Coquelicot.
- Compatible with Mathematical Components.
- Includes various facilities:
  - From our case study:
    - Notations for limits and convergence:
      `f @ x --> y, lim (f @ x), cvg (f @ +oo), u --> -oo`.
    - Automatic computation of differentials/derivatives.
  - Designed for this library:
    - A differential function, together with a notation: `'d f x`.
    - Equational Bachmann-Landau notations:
      `f = g +o_F e, f =O_F e,`
      `f x = g x +O_(x \near F) e x, f x =o_(x \near F) e x`.
    - Automatic proof of positivity.
    - A set of tactics for delayed instantiation of existential witnesses.

Filters

- A set of sets F is a filter if
  - $F \neq \emptyset$.
  - $\forall (P, Q) \in F^2, P \cap Q \in F$.
  - $\forall P \in F, \forall Q \supseteq P, Q \in F$.
- Examples:
  - Neighbourhood filter of a point p, written `locally` p.



Filters

- A set of sets F is a filter if
  - $F \neq \emptyset$.
  - $\forall (P, Q) \in F^2, P \cap Q \in F$.
  - $\forall P \in F, \forall Q \supseteq P, Q \in F$.
- Examples:
  - Neighbourhood filter of a point p, written `locally` p.
  - Neighbourhood filter of $+\infty$, written `Rbar_locally p_infty`.

Filters

- A set of sets F is a filter if
  - $F \neq \emptyset$.
  - $\forall (P, Q) \in F^2, P \cap Q \in F$.
  - $\forall P \in F, \forall Q \supseteq P, Q \in F$.
- Examples:
  - Neighbourhood filter of a point p, written `locally` p.
  - Neighbourhood filter of $+\infty$, written `Rbar_locally p_infty`.
  - Image of a filter F by a function y, written `filtermap` y F.

$$\texttt{filtermap}\ y\ F \ := \left\{ A \mid y^{-1}(A) \in F \right\}.$$

Filters

- A set of sets F is a filter if
  - $F \neq \emptyset$.
  - $\forall (P, Q) \in F^2, P \cap Q \in F$.
  - $\forall P \in F, \forall Q \supseteq P, Q \in F$.
- Examples:
  - Neighbourhood filter of a point p, written `locally` p.
  - Neighbourhood filter of $+\infty$, written `Rbar_locally p_infty`.
  - Image of a filter F by a function y, written `filtermap` y F.
- Convergence:

Before	After
`filterlim y (locally p) (locally q)`	`y @ p --> q`
`filterlim y (locally p) (Rbar_locally p_infty)`	`y @ p --> +oo`
`filterlim y (locally p) (set_locally A)`	`y @ p --> A`
`filterlim u eventually (Rbar_locally m_infty)`	`u --> -oo` or `u @ \oo --> -oo`
`filter_le F (locally p)`	`F --> p`

```
filterlim y F G = filter_le (filtermap y F) G
```

The `near` tactics: motivating example

To prove

$$\lim_a f = l_f \wedge \lim_a g = l_g \Rightarrow \lim_a (f+g) = l_f + l_g$$

Typical ε/δ-reasoning:

$$\forall \varepsilon > 0,\ \exists \delta_f > 0,\ \forall x,\ |x-a| < \delta_f \Rightarrow |f(x) - l_f| < \varepsilon$$
$$\forall \varepsilon > 0,\ \exists \delta_g > 0,\ \forall x,\ |x-a| < \delta_g \Rightarrow |g(x) - l_g| < \varepsilon$$

$$\forall \varepsilon > 0,\ \exists \delta > 0,\ \forall x,\ |x-a| < \delta \Rightarrow |f(x) + g(x) - (l_f + l_g)| < \varepsilon$$

The `near` tactics: motivating example

To prove

$$\lim_a f = l_f \wedge \lim_a g = l_g \Rightarrow \lim_a (f+g) = l_f + l_g$$

Typical ε/δ-reasoning:

$$\begin{array}{l}
\forall \varepsilon > 0,\ \exists \delta_f > 0,\ \forall x,\ |x - a| < \delta_f \Rightarrow |f(x) - l_f| < \varepsilon \\
\forall \varepsilon > 0,\ \exists \delta_g > 0,\ \forall x,\ |x - a| < \delta_g \Rightarrow |g(x) - l_g| < \varepsilon \\
\varepsilon > 0
\end{array}$$

$$\exists \delta > 0,\ \forall x,\ |x - a| < \delta \Rightarrow |f(x) + g(x) - (l_f + l_g)| < \varepsilon$$

The `near` tactics: motivating example

To prove

$$\lim_a f = l_f \wedge \lim_a g = l_g \Rightarrow \lim_a (f+g) = l_f + l_g$$

Typical ε/δ-reasoning:

$$\begin{array}{l}
\forall \varepsilon > 0,\ \exists \delta_f > 0,\ \forall x,\ |x-a| < \delta_f \Rightarrow |f(x) - l_f| < \varepsilon \\
\forall \varepsilon > 0,\ \exists \delta_g > 0,\ \forall x,\ |x-a| < \delta_g \Rightarrow |g(x) - l_g| < \varepsilon \\
\varepsilon > 0 \\
\delta_f > 0 \\
\forall x,\ |x-a| < \delta_f \Rightarrow |f(x) - l_f| < \frac{\varepsilon}{2} \\
\delta_g > 0 \\
\forall x,\ |x-a| < \delta_g \Rightarrow |g(x) - l_g| < \frac{\varepsilon}{2} \qquad \text{guess} \\
\hline
\exists \delta > 0,\ \forall x,\ |x-a| < \delta \Rightarrow |f(x) + g(x) - (l_f + l_g)| < \varepsilon
\end{array}$$

The `near` tactics: motivating example

To prove

$$\lim_a f = l_f \wedge \lim_a g = l_g \Rightarrow \lim_a (f+g) = l_f + l_g$$

Typical ε/δ-reasoning:

$$\begin{array}{l}
\forall \varepsilon > 0,\ \exists \delta_f > 0,\ \forall x,\ |x-a| < \delta_f \Rightarrow |f(x) - l_f| < \varepsilon \\
\forall \varepsilon > 0,\ \exists \delta_g > 0,\ \forall x,\ |x-a| < \delta_g \Rightarrow |g(x) - l_g| < \varepsilon \\
\varepsilon > 0 \\
\delta_f > 0 \\
\forall x,\ |x-a| < \delta_f \Rightarrow |f(x) - l_f| < \frac{\varepsilon}{2} \\
\delta_g > 0 \\
\forall x,\ |x-a| < \delta_g \Rightarrow |g(x) - l_g| < \frac{\varepsilon}{2}
\end{array}$$

$$\forall x,\ |x-a| < \underbrace{\min(\delta_f, \delta_g)}_{\text{guess}} \Rightarrow |f(x) + g(x) - (l_f + l_g)| < \varepsilon$$

Why ε/δ definitions are not best for formal proofs

A few aspects of typical ε/δ-reasoning:

- The (human) prover has to provide existential witnesses.
- Witnesses are (usually) explicit.
- Witnesses are (usually) given way before they are used.

Why ε/δ definitions are not best for formal proofs

A few aspects of typical ε/δ-reasoning:

- The (human) prover has to provide existential witnesses.
- Witnesses are (usually) explicit.
- Witnesses are (usually) given way before they are used.

$\Rightarrow$ Proof scripts are hard to read and hard to maintain.

Why ε/δ definitions are not best for formal proofs

A few aspects of typical ε/δ-reasoning:

- The (human) prover has to provide existential witnesses.
- Witnesses are (usually) explicit.
- Witnesses are (usually) given way before they are used.

$\Rightarrow$ Proof scripts are hard to read and hard to maintain.

$\Rightarrow$ Use an abstraction like filters.

The `near` tactics: motivating example (cont.)

To prove

$$\lim_a f = l_f \wedge \lim_a g = l_g \Rightarrow \lim_a (f+g) = l_f + l_g$$

Typical ε/δ-reasoning:

$$\forall \varepsilon > 0,\ \exists \delta_f > 0,\ \forall x,\ |x-a| < \delta_f \Rightarrow |f(x) - l_f| < \varepsilon$$
$$\forall \varepsilon > 0,\ \exists \delta_g > 0,\ \forall x,\ |x-a| < \delta_g \Rightarrow |g(x) - l_g| < \varepsilon$$

$$\forall \varepsilon > 0,\ \exists \delta > 0,\ \forall x,\ |x-a| < \delta \Rightarrow |f(x) + g(x) - (l_f + l_g)| < \varepsilon$$

The `near` tactics: motivating example (cont.)

To prove

$$f @ a \to l_f \;\Rightarrow\; g @ a \to l_g \;\Rightarrow\; (f+g) @ a \to (l_f + l_g)$$

Filter reasoning:

$$\frac{\begin{array}{c}\texttt{locally}(l_f) \subseteq f @ a \\ \texttt{locally}(l_g) \subseteq g @ a\end{array}}{\texttt{locally}(l_f + l_g) \subseteq (f+g) @ a}$$

The `near` tactics: motivating example (cont.)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Filter reasoning:

$$\frac{\begin{array}{l}\texttt{locally}(l_f) \subseteq f@a \\ \texttt{locally}(l_g) \subseteq g@a \\ A \in \texttt{locally}(l_f + l_g)\end{array}}{A \in (f+g)@a}$$

The `near` tactics: motivating example (cont.)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Filter reasoning:

$$\begin{array}{l}
\texttt{locally}(l_f) \subseteq f@a \\
\texttt{locally}(l_g) \subseteq g@a \\
\varepsilon > 0 \\
\mathrm{ball}_\varepsilon(l_f + l_g) \subseteq A \qquad \text{unfolding} \Rightarrow \text{introduction of } \varepsilon \\
\hline
A \in (f+g)@a \\
(\text{i.e. } (f+g)^{-1}(A) \in \texttt{locally}(a))
\end{array}$$

The `near` tactics: motivating example (cont.)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Filter reasoning:

$$\begin{array}{ll}
\texttt{locally}(l_f) \subseteq f@a & \\
\texttt{locally}(l_g) \subseteq g@a & \\
\varepsilon > 0 & \\
\mathrm{ball}_\varepsilon(l_f + l_g) \subseteq A & \\
B := (f+g)(f^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_f)) \cap g^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_g))) & \text{guess} \\
\hline
\text{closure by extension} & \begin{array}{l} B \in (f+g)@a \\ B \subseteq A \end{array}
\end{array}$$

The `near` tactics: motivating example (cont.)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Filter reasoning:

$$\begin{array}{l}
\texttt{locally}(l_f) \subseteq f@a \\
\texttt{locally}(l_g) \subseteq g@a \\
\varepsilon > 0 \\
\mathrm{ball}_\varepsilon(l_f + l_g) \subseteq A \\
B := (f+g)(f^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_f)) \cap g^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_g)))
\end{array}$$

$$\forall C,\ f(f^{-1}(C)) \subseteq C \subseteq f^{-1}(f(C))$$

$$\begin{array}{l}
f^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_f)) \cap g^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_g)) \in \texttt{locally}(a) \\
\mathrm{ball}_{\frac{\varepsilon}{2}}(l_f) + \mathrm{ball}_{\frac{\varepsilon}{2}}(l_g) \subseteq \mathrm{ball}_\varepsilon(l_f + l_g)
\end{array}$$

The `near` tactics: motivating example (cont.)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Filter reasoning:

$$\begin{array}{l} \texttt{locally}(l_f) \subseteq f@a \\ \texttt{locally}(l_g) \subseteq g@a \\ \varepsilon > 0 \\ \mathrm{ball}_\varepsilon(l_f + l_g) \subseteq A \\ B := (f+g)(f^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_f)) \cap g^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_g))) \end{array}$$

---

closure by intersection

$$\begin{array}{l} f^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_f)) \in \texttt{locally}(a) \\ g^{-1}(\mathrm{ball}_{\frac{\varepsilon}{2}}(l_g)) \in \texttt{locally}(a) \end{array}$$

The pros and cons of filter reasoning

Improvements:

- The explicit existential witnesses are removed.
- Parts of the arithmetic is hidden thanks to the abstraction.

But:

- There is still a guess: we have to know beforehand how we want to split the epsilons.
- We manipulate sets while (I think) it is more intuitive to reason about points.

The pros and cons of filter reasoning

Improvements:

- The explicit existential witnesses are removed.
- Parts of the arithmetic is hidden thanks to the abstraction.

But:

- There is still a guess: we have to know beforehand how we want to split the epsilons.
- We manipulate sets while (I think) it is more intuitive to reason about points.

$\Rightarrow$ Reintroduce points without breaking the abstraction and use existential variables.

The near tactics: motivating example (conclusion)

Standard filter manipulation:

```
Proof.
move=> /flim_norm limf /flim_norm limg.
move=> A /locally_normP [_/posnumP[e] lim_e_A]; rewrite locally_simpl.
apply: (@filterS _ _ _
  ((f + g) @` ((f @^-1` (ball_ norm lf (e%:num / 2))) `&` (g @^-1` (ball_ norm lg (e%:num / 2)))))).
  move=> _ [x [fx gx] <-]; apply: lim_e_A.
  by rewrite /= opprD addrACA; apply: normm_lt_split.
by apply: filterS (@preimage_image _ _ _ _) _; apply: filterI; [apply: limf|apply: limg].
Qed.
```

With the near tactics:

```
Proof.
move=> /flim_norm limf /flim_norm limg.
apply/flim_normP => _/posnumP[e]; rewrite !near_simpl; near=> x.
by rewrite opprD addrACA normm_lt_split //; near: x; [apply: limf|apply: limg].
Grab Existential Variables. end_near. Qed.
```

Key ingredients

- A lemma to reintroduce points and use existential variables.

```
Lemma filter_near_of F (P : in_filter F) Q :
  Filter F -> (forall x, P(x) -> Q(x)) -> Q ∈ F.
```

- A notation $\forall x \,\mathrm{near}\, F,\ Q(x)$, standing for $Q \in F$, to invite the user to reason about points.
- The fact that filters are closed by intersection, to accumulate properties.

The `near` tactics: motivating example (end)

To prove

$$f @ a \to l_f \;\Rightarrow\; g @ a \to l_g \;\Rightarrow\; (f+g) @ a \to (l_f + l_g)$$

Filter reasoning:

$$\frac{\begin{array}{c} f @ a \to l_f \\ g @ a \to l_g \end{array}}{(f+g) @ a \to (l_f + l_g)}$$

The `near` tactics: motivating example (end)

To prove

$$f @ a \to l_f \;\Rightarrow\; g @ a \to l_g \;\Rightarrow\; (f+g) @ a \to (l_f + l_g)$$

Improved filter reasoning:

$$\begin{array}{c} \forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |f(x) - l_f| < \varepsilon \\ \forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |g(x) - l_g| < \varepsilon \end{array}$$

$$\forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |f(x) + g(x) - (l_f + l_g)| < \varepsilon$$

The `near` tactics: motivating example (end)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Improved filter reasoning:

$$\begin{array}{l} \forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |f(x) - l_f| < \varepsilon \\ \forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |g(x) - l_g| < \varepsilon \\ \varepsilon > 0 \qquad \text{regular intro} \end{array}$$

$$\forall x \,\mathrm{near}\, a,\ |f(x) + g(x) - (l_f + l_g)| < \varepsilon$$

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Improved filter reasoning:

$$\begin{array}{l}
\forall \varepsilon > 0,\ \forall x \text{ near } a,\ |f(x) - l_f| < \varepsilon \\
\forall \varepsilon > 0,\ \forall x \text{ near } a,\ |g(x) - l_g| < \varepsilon \\
\varepsilon > 0 \\
x \text{ near } a, \qquad \texttt{near}\ \text{intro}
\end{array}$$

$$|(f(x) - l_f) + (g(x) - l_g)| < \varepsilon$$

The `near` tactics: motivating example (end)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Improved filter reasoning:

$$\begin{array}{l}
\forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |f(x) - l_f| < \varepsilon \\
\forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |g(x) - l_g| < \varepsilon \\
\varepsilon > 0 \\
x \,\mathrm{near}\, a,
\end{array}$$

$$\begin{array}{l}
|f(x) - l_f| < \frac{\varepsilon}{2} \\
|g(x) - l_g| < \frac{\varepsilon}{2}
\end{array}$$

The `near` tactics: motivating example (end)

To prove

$$f@a \to l_f \;\Rightarrow\; g@a \to l_g \;\Rightarrow\; (f+g)@a \to (l_f + l_g)$$

Improved filter reasoning:

$$\begin{array}{l} \forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |f(x) - l_f| < \varepsilon \\ \forall \varepsilon > 0,\ \forall x \,\mathrm{near}\, a,\ |g(x) - l_g| < \varepsilon \\ \varepsilon > 0 \end{array}$$

`near` revert $\quad \begin{array}{l} \forall x \,\mathrm{near}\, a,\ |f(x) - l_f| < \frac{\varepsilon}{2} \\ \forall x \,\mathrm{near}\, a,\ |g(x) - l_g| < \frac{\varepsilon}{2} \end{array}$

Back to the case study

Lines of code: [1]

	Using Coquelicot	Using our library
LaSalle's invariance principle	~ 370	~ 370
Inverted pendulum	~ 980	~ 900

~ 70 additional lines of code could be removed with a better compatibility between Mathematical Components and tactics such as `ring` and `field`.

[1] Not counting the parts that were integrated to our library.

Conclusion

A case study in control theory:

- Generalisation of LaSalle's invariance principle.
- A corrected proof of soundness for a control function for the inverted pendulum.

A new library for classical analysis:

- Compatible with Mathematical Components.
- New notations and tools (limit notations, Bachmann-Landau notations, `near` tactics).

Some bits of automation:

- Computation of differentials and derivatives.
- A new reflection methodology based on refinements.

Perspectives

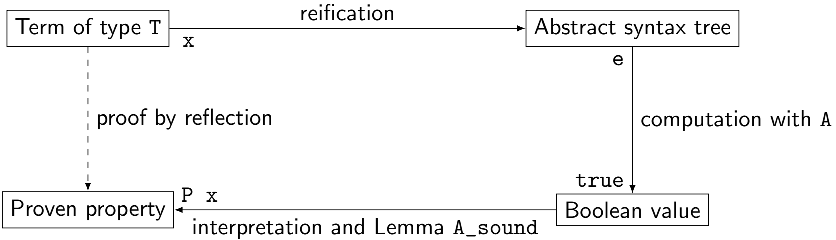
- Towards certified embedded software.
- Integrals and Cauchy-Lipschitz Theorem.
- Better accessibility for non-expert users.

Perspectives

- Towards certified embedded software.
- Integrals and Cauchy-Lipschitz Theorem.
- Better accessibility for non-expert users.

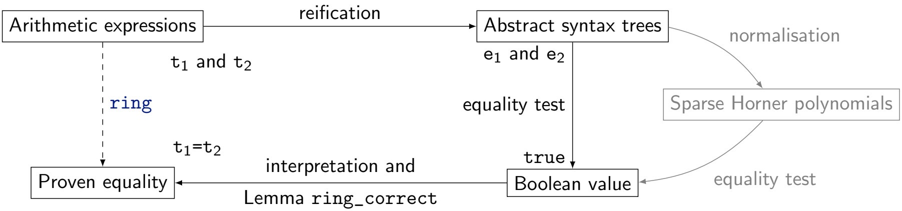
Thank you for your attention!

Reflection



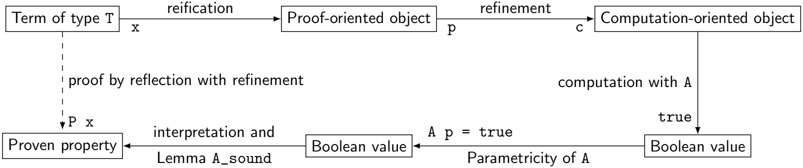

```
Lemma A_sound (e : AST) : A e = true -> P (interp e).
```

Example: the ring tactic




```
Lemma ring_correct (e1 e2 : AST) (l : map) :
  Peq (norm e1) (norm e2) = true ->
  interp l e1 = interp l e2.
```

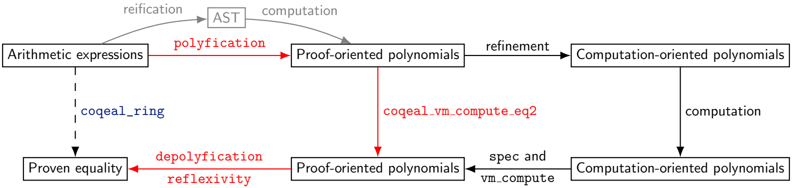
A more modular methodology




```
Lemma A_sound (p : PO_type) : A p = true -> P (interp p).
```

Main ingredients: generic programming and refinement.

Almost example: the coqeal_ring tactic




```
Lemma polyficationP (e : AST) (l : map) :
  interp l e = eval_poly l (ast_to_poly e).
```