An overview of the real numbers in theorem provers: an application with real analysis in Coq

Micaela Mayero

LIPN - UMR CNRS - USPN (Paris 13) - France

Febrary 2023
MAP 2023

Motivations: what am I going to talk about?

Some results can be critical. What confidence?

- Computation and proof errors:
  - modelling (physics, biology...),
  - mathematics (schemes, framework...),
  - incomplete convergence (numerical method, linear/nonlinear resolution, time step, mesh size, polynomial degree, truncation...),
  - input data, mesh,
  - programming, implementation,
  - rounding / floating points errors, ...
- Validation:
  - confront with experiment / experience,
  - tests with known, analytical solutions,
  - compare various codes and methods (numerical benchmarks),
- Formal Methods:
  - verification
  - formal proof

Generalities, families

About proof assistants: not automatic provers, to deal with humans (interactive way), type checkers

- Propositional logic, Peano integers: Automath (de Bruijn, 1968)
- + Arithmetic (integers): Mizar (Trybulec, 1970), HOL (Milner, 1970), Isabelle (Paulson, 1990), Coq (Thierry Coquand, 1984), PVS (1992), Agda (Catarina Coquand,1999), ...
- + reals numbers: Mizar (Hryniewiecki, 1989), HOL-Light (Harrison, 1996), Coq (Mayero, Niqui, 1998, 2000), PVS (Dutertre, 1996), Isabelle/HOL (Fleuriot, 2000), Lean (2020), ...
- + floating numbers: PVS (1995), HOL-Light (2000), Coq (2003), ...

An example: Coq

Coq is based on CIC (type theory, inductive types, Curry-Howard isomorphism, intuitionistic logic built-in,...).

```
Theorem plus_n_O :
    ∀ n:nat, n + 0 = n.
Proof.
induction n;simpl.
reflexivity.
rewrite IHn.
reflexivity.
Qed.
```

```
Proof.
easy.
Qed.
```

Proof λ-term:

```
plus_n_O = fun n : nat ⇒
nat_ind (fun n0 : nat ⇒ n0 + 0 = n0)
  (eq_refl : 0 + 0 = 0)
  (fun (n0 :  nat) (IHn :  n0 + 0 = n0) ⇒
   eq_ind_r (fun n1 : nat ⇒ S n1 = S n0)
     eq_refl IHn : S n0 + 0 = S n0) n
```

What is the problem with real numbers ?

To compute with "real numbers" (floating points) in a computer (processor):

A = (1003+ -1000)+7.501 = 10.5010000000000012
B = 1003+ (-1000+7.501) = 10.5009999999999764

But what about a proof of A = B ?

In maths, A = B means that the addition is associative for real numbers. And it's true!

In theorem provers: we want "true" real numbers, to keep the mathematical properties.

Some methods to define real numbers

Mathematics:

Remark: real numbers are inumerable $\hookrightarrow$ no inductive type

- axiomatisation/construction
- Cauchy, Cantor, Dedekind cut
- 1st ordre/2nd order

Logic:

- classical logic / intuitionnistic logic
- “usual” analysis / Bishop analysis

Some examples (1/5)

Mizar: the logic is based on the axioms of the Tarski-Grothendieck set theory (an extension of the Zermelo-Fraenkel set theory).
While originally an axiomatization, the formalization of real numbers in Mizar was later changed to a construction based on Dedekind cuts (Trybulec 1998):

```
func DEDEKIND_CUTS -> Subset - Family o f RAT + equals
{ A where A i s Subset o f RAT + :
f o r r being Element o f RAT + s t r in A holds
( ( f o r s being Element o f RAT + s t s <= ' r holds s in A ) &
ex s being Element o f RAT + s t ( s in A & r < s ) )
} \ { RAT +};
```

A is a Dedekind cut if it is a strict subset of $\mathbb{Q}^+ \cup \{0\}$ such that $\forall r \in A$,
$(\forall s \in \mathbb{Q}^+ \cup \{0\}, s \leq r \Rightarrow s \in A) \wedge (\exists s \in \mathbb{Q}^+ \cup \{0\}, r < s \wedge s \in A)$

```
func A + B -> Element o f DEDEKIND_CUTS equals
{ ( r + s ) where r , s i s Element o f RAT + : ( r in A & s in B ) };
```

Some examples (2/5)

HOL Light: based on classical higher-order logic with axioms of infinity, extensionality, and choice in the form of Hilbert's ϵ operator. It follows the LCF approach.
Rather than using fully-featured sequences of rational numbers, real numbers are based on nearly-additive sequences of natural numbers.
The predicate `is_nadd` characterizing such a sequence `x` is given below, with `dist` the function returning the distance between two natural numbers.

```
let is_nadd = new definition
`is_nadd x <=> (?B. !m n. dist (m*x(n), n*x(m))<=B*(m+n))`;;

let nadd_le = new definition
`x <<= y <=> ?B. !n. x(n) <= y(n)+B`;;
```

- existence of the least upper bound of any nonempty bounded set of nearly-additive sequences;
- this theorem is later extended to give the completeness of real numbers.
- addition of two nearly-additive sequences is performed pointwise
- the multiplication is the composition of sequences
- whole real line is built as a succesive quotient types.

Some examples (3/5)

PVS: the logic is based on classical higher-order logic.
Thanks to its pervasive support for subtyping, PVS takes a top-down approach. PVS starts from a number type that is a superset of all numerals. This set encompasses integers (from Lisp), rationals, real numbers. Below is an excerpt of the pvs reals.

```
number : NONEMPTYTYPE
number_field : NONEMPTYTYPE FROM number
n0x : VAR nonzero_number
inverse_mult : AXIOM n0x * (1/ n0x ) = 1

real : NONEMPTYTYPE FROM number_field
nonzero_real : NONEMPTYTYPE = { r : real | r /= 0} CONTAINING 1
nzreal_is_nznum : JUDGEMENT nonzero_real SUBTYPE OF nonzero_number
x , y : VAR real
posreal_add_closed : POSTULATE x > 0 AND y > 0 IMPLIES x + y > 0
trichotomy : POSTULATE x > 0 OR x = 0 OR 0 > x
```

(POSTULATE means that PVS decision procedure are able to prove these properties: it is the combination of the explicit axioms from theories and the implicite ones from decision procedures that defines what real numbers are in PVS.)

Some examples (4/5)

L∃∀N: the logic is based on Calculus of Inductives Constructions extended with quotient types.
The type of real numbers constructed as equivalence classes of Cauchy sequences of rational numbers.

```
structure real := of_cauchy ::
(cauchy : cau_seq.completion.Cauchy (abs : Q → Q))

def Cauchy := @quotient (cau_seq _ abv) cau_seq.equiv

instance equiv : setoid (cau_seq B abv) :=...
```

Mathlib: a unified library of mathematics formalized. It is distinguished by dependently typed foundations, focus on classical mathematics, extensive hierarchy of structures.

Some examples (5/5)

Coq: the logic is based on Calculus of Inductives Constructions.
Reals are purely axiomatized. The operations are total functions. The comparison between reals is decidable (classical aspect). Some axioms:

```
Parameter ℝ: Set.
Parameter Rplus: ℝ→ ℝ→ ℝ.
Parameter Rlt: ℝ→ ℝ→ Prop.
Parameter Rinv: ℝ→ ℝ.
Axiom Rplus_com: ∀ r1 r2:ℝ, r1 + r2 = r2 + r1.
Axiom Rplus_lt_compat_l: ∀ r r1 r2 :ℝ , r1 < r2 → r + r1 < r + r2.
Axiom Rinv_l : ∀ r :ℝ , r ≠ 0 → / r * r = 1.
Axiom total_order_T: ∀ r1 r2 :ℝ , {r1 < r2} + {r1 = r2} + {r1 > r2}.
```

In Coq there also exists a constructive real numbers library: C-CoRN (Niqui, 2000).

In 2013, the Coquelicot (Boldo, Lelay, Melquiond) has been designed: a conservative user-friendly extension of stdlib Reals, easier way of writing formulas and theorem statements, by relying on total functions in place of dependent types for limits, derivatives, integrals, power series,...

In 2020, the strandard Reals library has been conservatively updated (V. Semeria) introducing a commun constructive and classical part.

Disclaimers

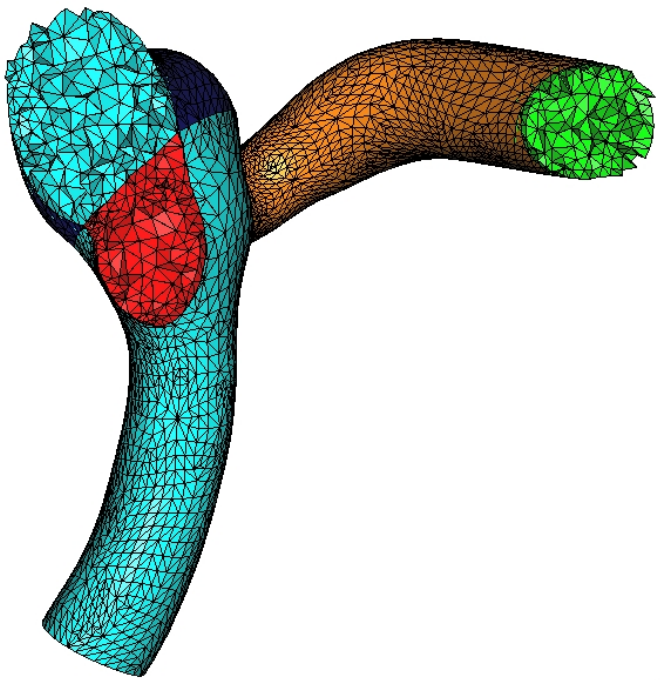
Presentation of a joint work and some colleagues's works.
computer scientists and mathematician.

- *1D wave equation (2010, 2013, 2014)* :
  - S.Boldo, F.Clément, J.C.Filliâtre, M.Mayero, G.Melquiond, P.Weis.

coq-num-analysis library:

- *Lax–Milgram* and *Lebesgue Integration of Nonnegative Functions* (2017, 2022):
  - SB, FC, F.Faissole, V.Martin, MM.
- *Bochner Integral* (2022):
  - SB, FC, L.Leclerc.
- *Lebesgue Induction, Tonelli's Theorem* (2023):
  - SB, FC, VM, MM, H.Mouhcine.

Motivations (1/2)



mesh of a cerebral aneurysm

Finite Element Method (FEM): widely used, sound mathematical foundations, broad spectrum of PDE, complex 3D geometries + real life applications...

Objective

- Prove correctness of the mathematical Finite Element method and of (parts of) a C++ FEM code, in Coq.

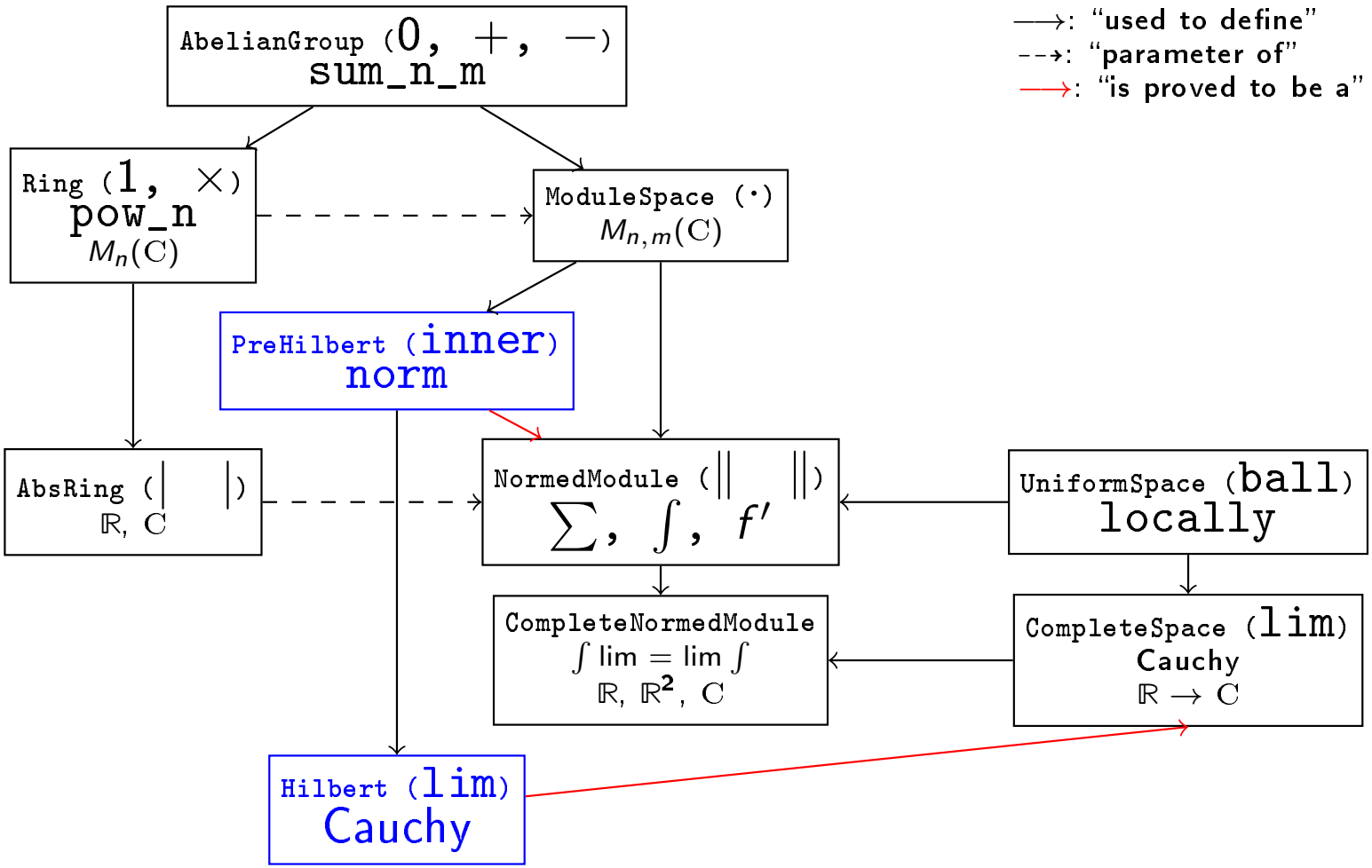
Motivations (2/2)

FEM requires a wide spectrum of mathematical results:

- Lax–Milgram:
  - Banach fixed point thm.
  - Riesz representation thm.
  - Hilbert spaces.
  - Continuous linear mappings.
- Sobolev spaces:
  - Lebesgue integral.
    - Measure theory.
    - Simple functions.
    - Lebesgue measure.
    - Results on $\mathbb{R}$ and $\overline{\mathbb{R}}$.
    - ...
  - Distributions.
  - Approximation results.
- Finite elements:
  - Polynomials.
  - Algebra on finite dimensional spaces.
  - Interpolation.
  - Geometry on polygons / polyhedra (mesh).
  - ...
- Solvers:
  - Linear solvers.
  - Non linear solvers.
  - Optimization.
  - Parallelism.
- ...

Enriched Coquelicot Hierarchy

Coquelicot: conservative extension of stdlib Reals.



Lax-Milgram Theorem and Céa's Lemma

Theorem (Lax-Milgram)

Let E : `Hilbert`, $f \in E'$, $C, \alpha \in \mathbb{R}_+^*$. Let $\varphi : E \to$ `Prop`, φ `ModuleSpace`-compatible and complete. Let a be a bilinear form of E bounded by C and α-coercive. Then:

$$\exists! u \in E, \varphi(u) \wedge \forall v \in E, \varphi(v) \Longrightarrow f(v) = a(u,v) \wedge \|E\|u \leq \frac{1}{\alpha}|||f|||_\varphi.$$

Lemma (Céa)

Let E : `Hilbert`, $f \in E'$, $0 < \alpha$. Let $\varphi : E \to$ `Prop`, φ `ModuleSpace`-compatible and complete. Let a be a bilinear form of E, bounded by $C > 0$ and α-coercive. Let u and u_φ be the solutions given by Lax–Milgram Theorem respectively on E and on the subspace φ. Then:

$$\forall v_\varphi \in E, \varphi(v_\varphi) \Longrightarrow \|E\|u - u_\varphi \leq \frac{C}{\alpha}\|E\|u - v_\varphi.$$

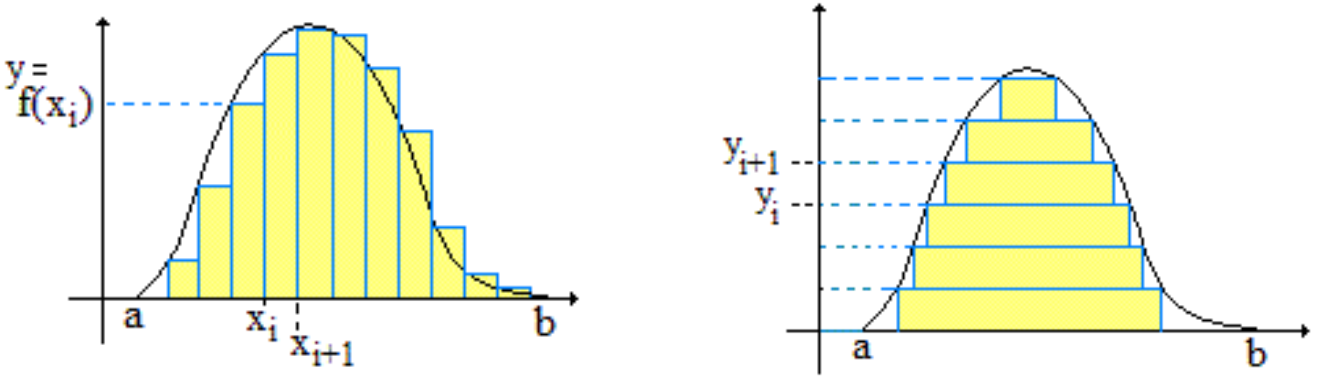
Lax-Milgram: summary of the work done

- results about functional spaces, linear and bilinear mappings
- fixed-point theorem in a sub-complete normed module
- decide if a space ($\geq$ NormedModule) is only zero
- norm on functions (`operator_norm`) on $\mathbb{R} \cup \{+\infty\}$
- 8 equivalences of continuity of linear mappings
- definitions of pre-Hilbert and Hilbert spaces in Coquelicot hierarchy
- define `clm`: the set of the continuous linear mappings
- prove it is a `NormedModule`, to consider `clm` E (`clm` E $\mathbb{R}$)
- prove Lax-Milgram theorem and Céa's lemma
- for a total of about 10k lines of Coq and 430 lemmas/definitions

Integration theories: short review

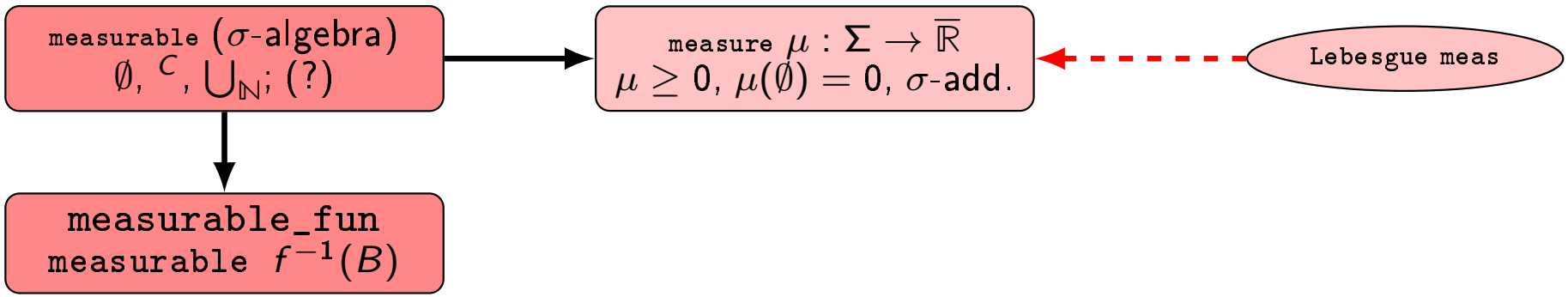
Several choices:

- Riemann integral: exists already in Coq.
- Henstock–Kurzweil (gauge) integral.
- Daniell integral. (no measure needed)
  Possible constructive approach (Bishop, Cheng, 1972, Coq: Semeria, 2021).
- Lebesgue integral (broader class of functions).
- Bochner integral (functions to Banach space).

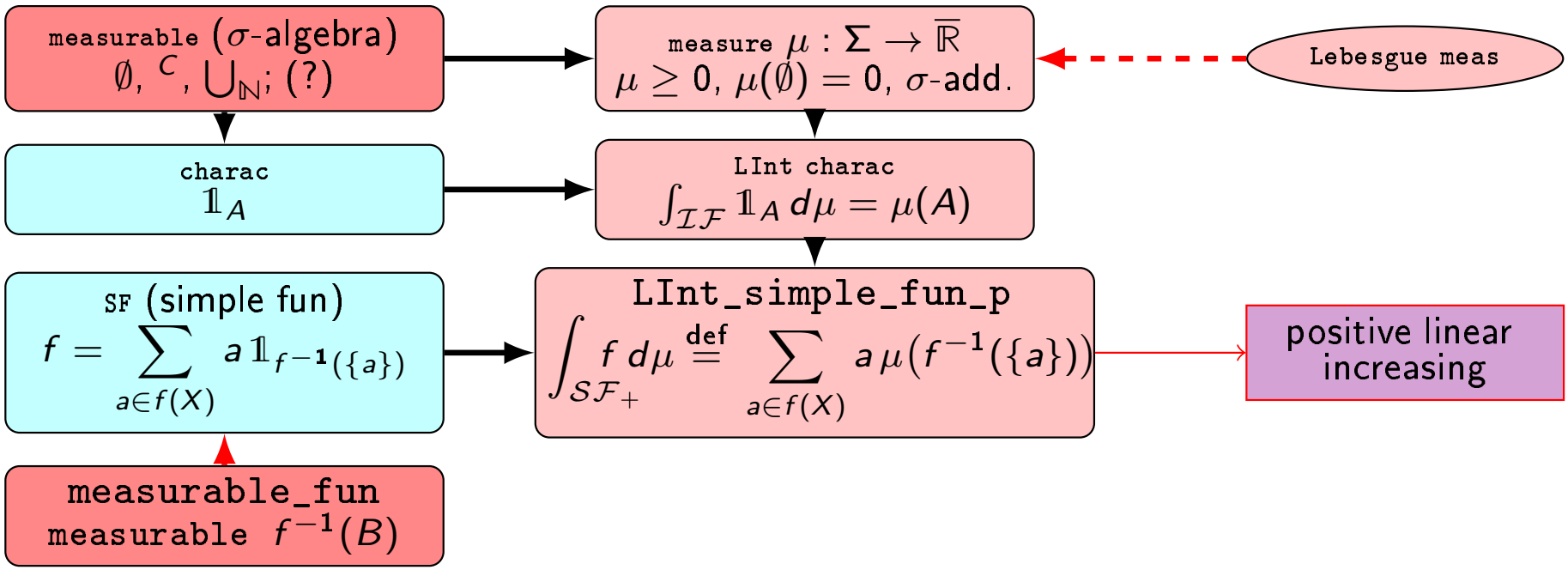


Riemann integral *vs* Lebesgue integral

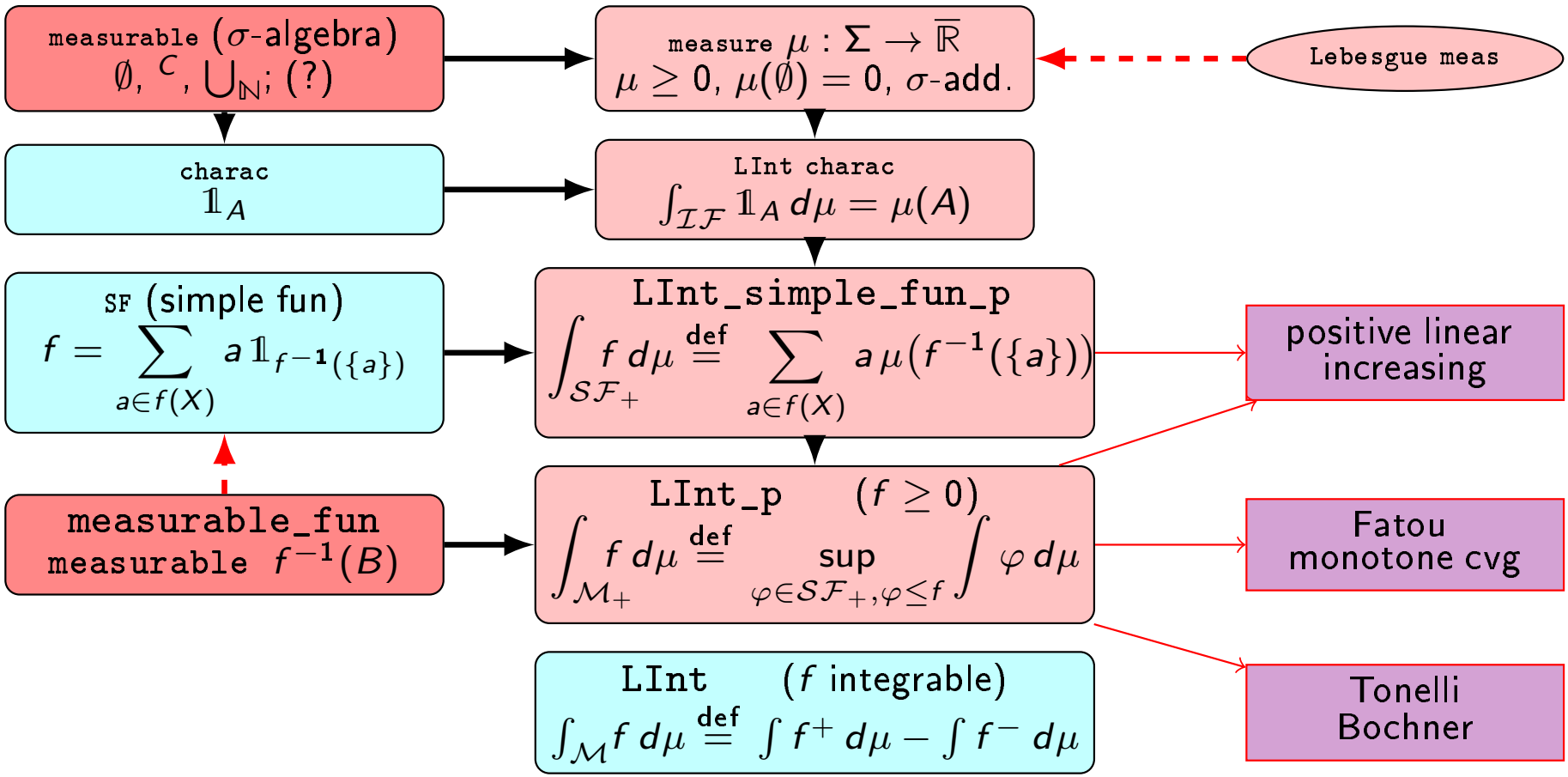
Building Lebesgue integral



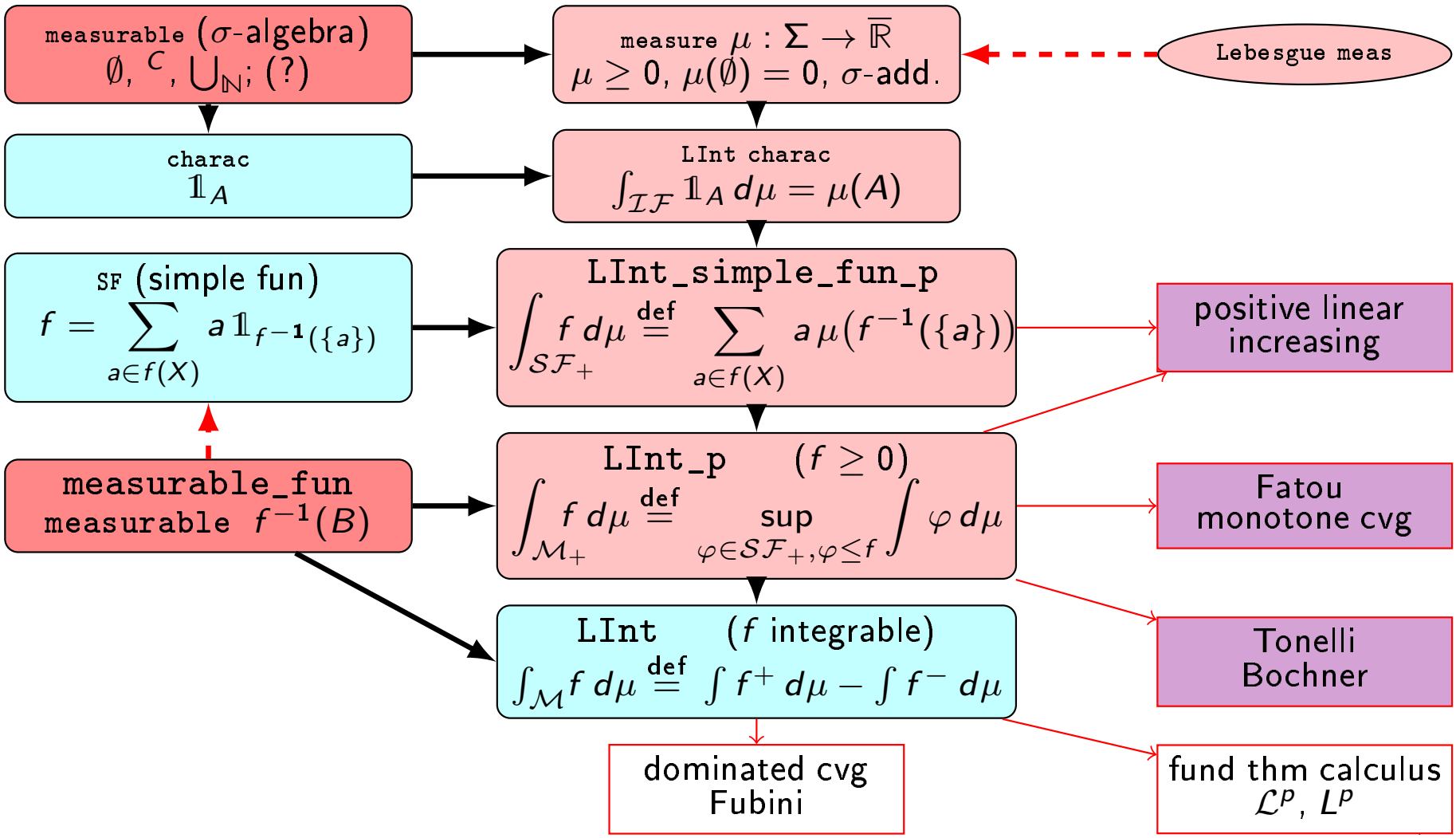
Building Lebesgue integral



Building Lebesgue integral



Building Lebesgue integral



Formalization of Lebesgue Integral: some focuses

Long work, well known mathematical results.

Short focus on some points of the formalization in Coq:

- Measurability and σ-algebra
- Dealing with "μ-almost everywhere"
- Integration of simple functions
- Lebesgue integral definition

Measurability in Coq

Given a set $A \in X$ (A : X→ Prop), is it measurable?

Definition from the generator sets:

```
Context {X : Type}.

Variable gen : (X → Prop) → Prop.

Inductive measurable : (X→ Prop) → Prop :=
  | measurable_gen : ∀ A, gen A→ measurable A
  | measurable_empty : measurable (fun _ ⇒ False)
  | measurable_compl : ∀ A,
        measurable A → measurable (fun x ⇒ not (A x))
  | measurable_union_countable :
   ∀ A :nat → (X→ Prop),
      (∀ n, measurable (A n)) → measurable (fun x ⇒ ∃ n, A n x).
```

Not the usual mathematical definition.

Proof that `measurable` sets = σ-algebras.

Dealing with "μ-almost everywhere"

- Many necessary results for $\overset{\mu\,\text{a.e.}}{=}$:
  - $f \overset{\mu\,\text{a.e.}}{=} g \Rightarrow$
    - $|f| \overset{\mu\,\text{a.e.}}{=} |g|$,
    - $f^{\pm} \overset{\mu\,\text{a.e.}}{=} g^{\pm}$,
    - $\alpha f \overset{\mu\,\text{a.e.}}{=} \alpha g$,
    - $f + h \overset{\mu\,\text{a.e.}}{=} g + h$,
    - ...
  - $f_1 \overset{\mu\,\text{a.e.}}{=} g_1,\ f_2 \overset{\mu\,\text{a.e.}}{=} g_2, \Rightarrow$
    - $f_1 + f_2 \overset{\mu\,\text{a.e.}}{=} g_1 + g_2$
    - $f_1 * f_2 \overset{\mu\,\text{a.e.}}{=} g_1 * g_2$
    - ...
  - $\forall i :\ f_i \overset{\mu\,\text{a.e.}}{=} g_i, \Rightarrow$
    - $\sup_i f_i \overset{\mu\,\text{a.e.}}{=} \sup_i g_i$
    - $\lim_i f_i \overset{\mu\,\text{a.e.}}{=} \lim_i g_i$
    - ...
- Many necessary results for $\overset{\mu\,\text{a.e.}}{\leq}$:
  - $f \overset{\mu\,\text{a.e.}}{\leq} g \Rightarrow$
    - $\alpha \geq 0, \Rightarrow \alpha f \overset{\mu\,\text{a.e.}}{\leq} \alpha g$,
    - $\alpha \geq 0, \Rightarrow -\alpha f \overset{\mu\,\text{a.e.}}{\geq} -\alpha g$,
    - $f + h \overset{\mu\,\text{a.e.}}{\leq} g + h$,
    - ...
  - $f_1 \overset{\mu\,\text{a.e.}}{\leq} g_1,\ f_2 \overset{\mu\,\text{a.e.}}{\leq} g_2, \Rightarrow$
    - $f_1 + f_2 \overset{\mu\,\text{a.e.}}{\leq} g_1 + g_2$
    - ...

Simple functions integration: $f = \sum_{y \in f(E)} \mathbb{1}_{f^{-1}(\{y\})}$

$$\int f \, d\mu \stackrel{\text{def}}{=} \sum_{a \in f(X)} a \, \mu\left(f^{-1}(a)\right) \quad \in \overline{\mathbb{R}}$$

```
Definition SF_aux : (E→ R) → (list R) → Prop :=
  fun f l ⇒ finite_vals_canonic f l ∧
    (forall a, measurable gen (fun x ⇒ f x = a)).

Definition SF : (E→ R) → Set := fun f ⇒ { l | SF_aux f l}.

Definition af1 (f:E→ R) :=
  (fun a :  Rbar ⇒ Rbar_mult a (mu (fun (x:E) ⇒ f x = a))).

Definition LInt_simple_fun_p :=
   fun (f:E→ R) (H:SF gen f) ⇒ let l:= (proj1_sig H) in
     sum_Rbar_map l (af1 f).
```

We proved the value does not depends on the proof H.
⇒ theorems about sum, multiplication by a scalar and change of variable

Lebesgue integral definition

$$\int f\,d\mu \overset{\text{def}}{=} \sup_{\varphi \in \mathcal{SF}_+, \varphi \le f} \int \varphi\,d\mu \quad \in \overline{\mathbb{R}}$$

```
Definition LInt_p :(E→ Rbar) → Rbar := fun f ⇒
      Rbar_lub (fun x ⇒ exists (g:E→ R), exists (Hg: SF gen g),
                                 non_neg g ∧
                                 (forall (z:E), Rbar_le (g z) (f z)) ∧
                                 LInt_simple_fun_p mu g Hg = x).
```

Lebesgue integration: summary of the work done

- detailed pen-and-paper proof: 240 pages and 800 lemmas/definitions
- formal proofs: 11 k lines of Coq and 635 lemmas/definitions
- full classical logic.
- should be useful for others.
- formalized:
  - measure theory, Lebesgue measure,
  - Lebesgue integral for measurable nonnegative fun,
  - subset systems (algebra, monotone class, ring, π-system...)
  - Bochner...

Tonelli's Theorem

Theorem

Let (X_1, Σ_1, μ_1) and (X_2, Σ_2, μ_2) be measure spaces. Assume that μ_1 and μ_2 are σ-finite. Let $f \in \mathcal{M}_+(X_1 \times X_2, \Sigma_1 \otimes \Sigma_2)$. Then, we have

$$\int_{X_1 \times X_2} f \, d(\mu_1 \otimes \mu_2) = \int_{X_1} \left(\int_{X_2} f_{x_1} \, d\mu_2 \right) d\mu_1 = \int_{X_2} \left(\int_{X_1} f^{x_2} \, d\mu_1 \right) d\mu_2.$$

Where all the integrals are legitimate.

$\Sigma_1 \otimes \Sigma_2$: product of σ_algebras.
Mplus / $\mathcal{M}_+$: nonnegative measurable functions.

```
Lemma Tonelli : ∀ f, Mplus genX1xX2 f → ... →
  LInt_p muX1xX2 f = LInt_p muX1 (LInt_p_section_fun muX2 f) ∧
  LInt_p muX1xX2 f = LInt_p muX2 (LInt_p_section_fun muX1 (swap f)).
```

Lebesgue Induction Principle

Lemma (Lebesgue induction principle)

Let P be a predicate on functions $X \to \overline{\mathbb{R}}$. Assume that P holds on the set of measurable indicator functions, and that it is compatible on $\mathcal{M}_+$ with positive linear operations and with the supremum of nondecreasing sequences:

$$\begin{aligned} \forall A, \quad & A \in \Sigma \Rightarrow P(\mathbb{1}_A), \\ \forall a \in \mathbb{R}_+,\ \forall f \in \mathcal{M}_+, \quad & P(f) \Rightarrow P(af), \\ \forall f, g \in \mathcal{M}_+, \quad & P(f) \wedge P(g) \Rightarrow P(f+g), \\ \forall (f_n)_{n\in\mathbb{N}} \in \mathcal{M}_+, \quad & (\forall n \in \mathbb{N},\ f_n \le f_{n+1} \wedge P(f_n)) \Rightarrow P\left(\sup_{n\in\mathbb{N}} f_n\right). \end{aligned}$$

Then, P holds on $\mathcal{M}_+$.

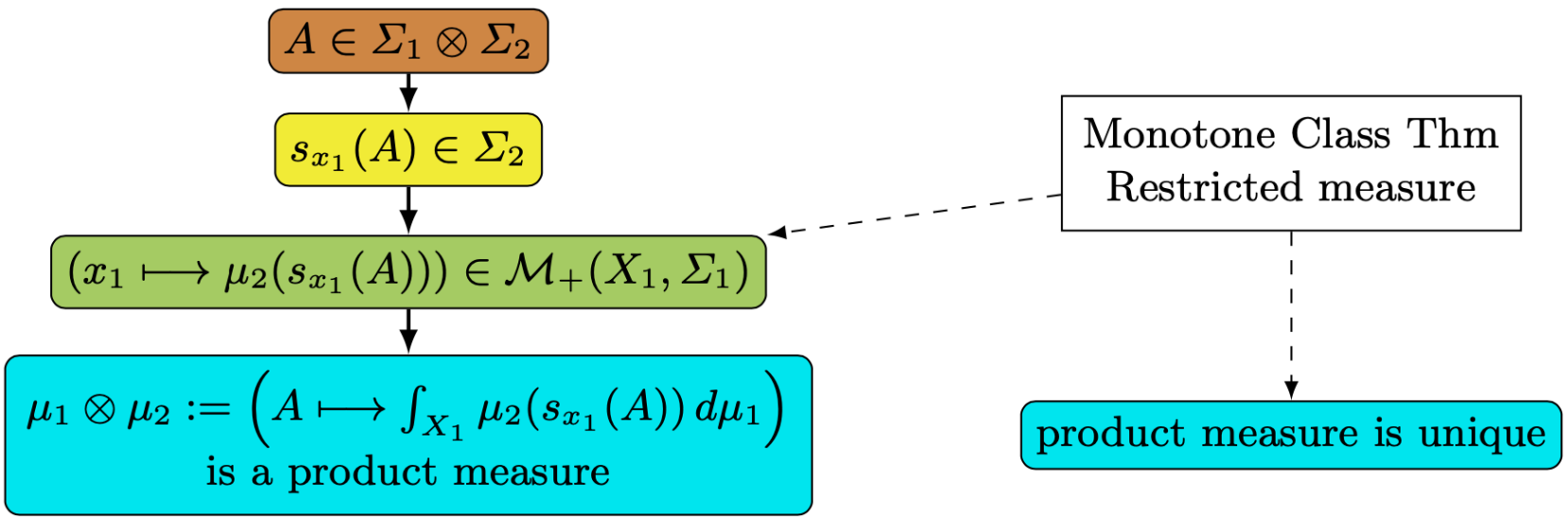
```
Inductive Mp : (X → R̄) → Prop :=
  | Mp_charac : ∀ A, measurable genX A → Mp (charac A)
  | Mp_scal : ∀ a f, 0 ≤ a → Mp f → Mp (fun x ⇒ a *_R̄ f x)
  | Mp_plus : ∀ f g, Mp f → Mp g → Mp (fun x ⇒ f x +_R̄ g x)
  | Mp_sup : ∀ f, incr_fun_seq f → (∀ n, Mp (f n)) →
                Mp (fun x ⇒ Sup_seq (fun n ⇒ f n x)).
```

Product Measure on a Product Space

Product measure

Given two measure spaces (X_1, Σ_1, μ_1) and (X_2, Σ_2, μ_2), a *product measure* on $(X_1 \times X_2, \Sigma_1 \otimes \Sigma_2)$ is a measure μ defined on the product σ-algebra $\Sigma_1 \otimes \Sigma_2$ satisfying the *box property*:

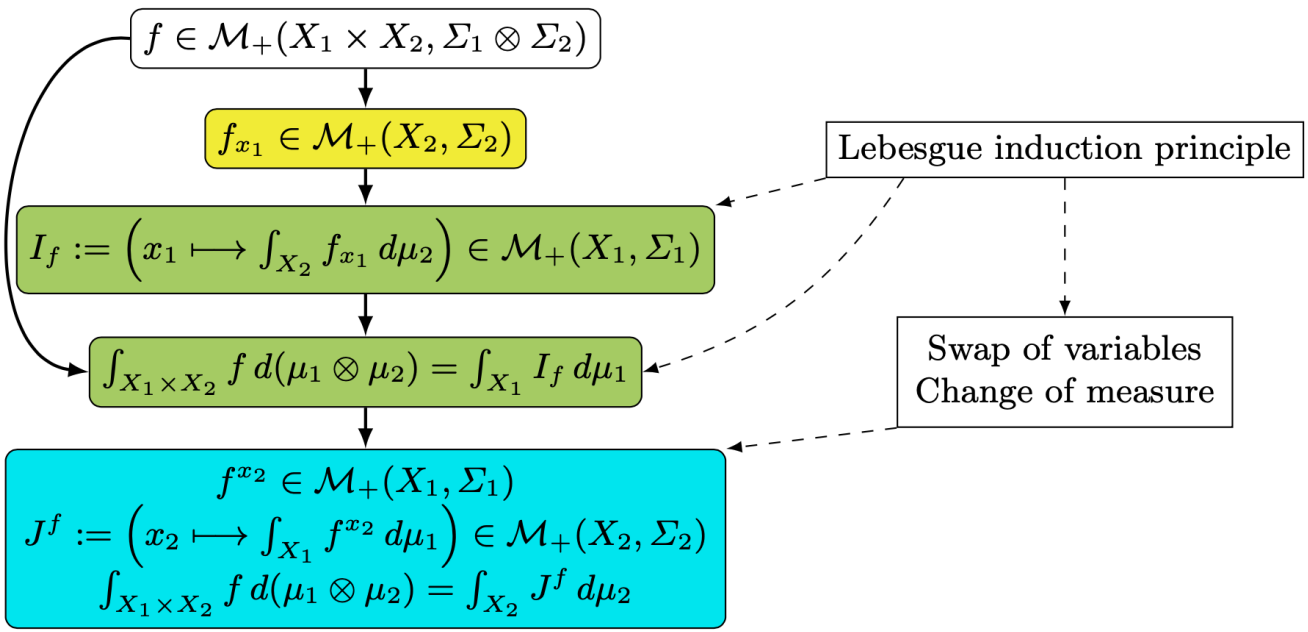
$$\forall A_1 \in \Sigma_1,\ \forall A_2 \in \Sigma_2, \quad \mu(A_1 \times A_2) = \mu_1(A_1)\,\mu_2(A_2).$$



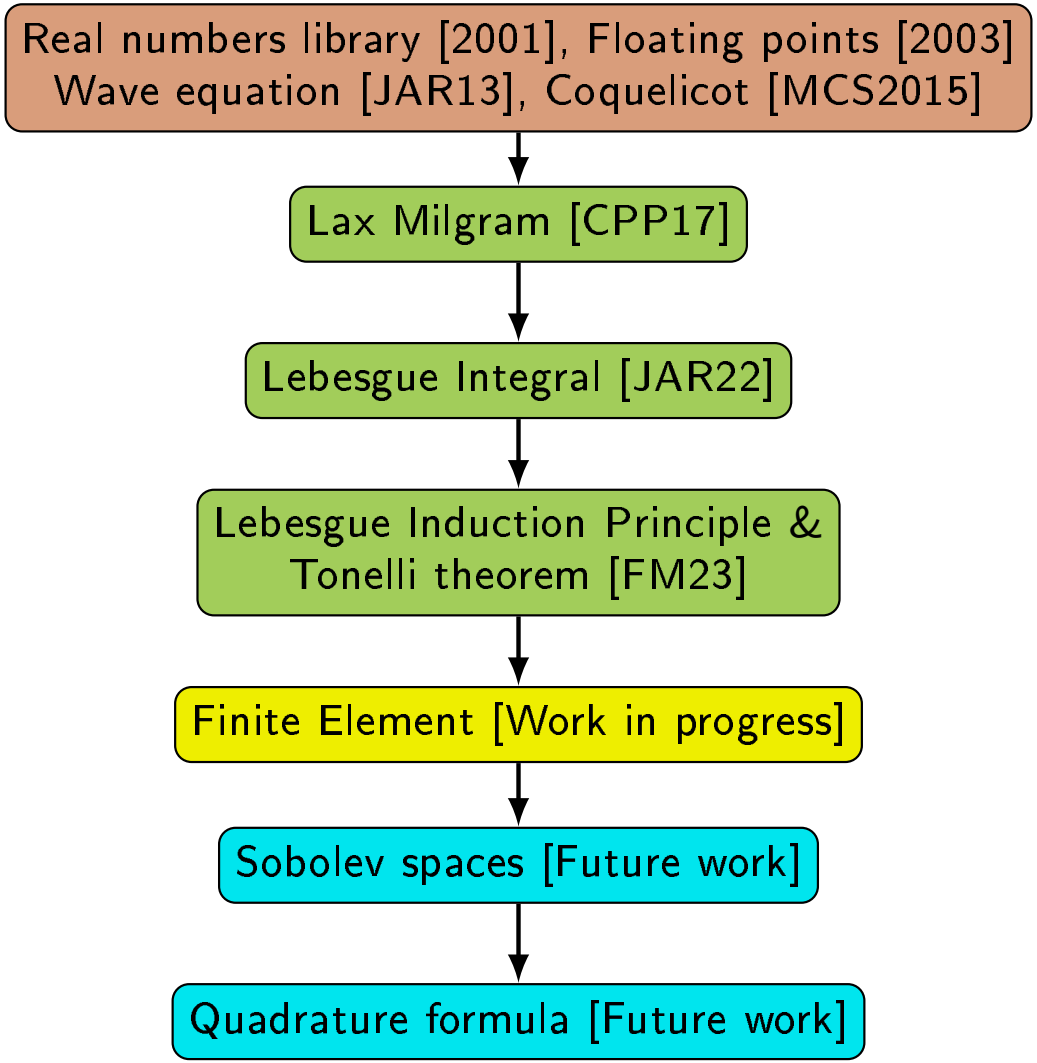
Coq Formalization of the Tonelli's Theorem

$$\int_{X_1 \times X_2} f \, d(\mu_1 \otimes \mu_2) = \int_{X_1} \left(\int_{X_2} f \, d\mu_2 \right) d\mu_1 \tag{1}$$

$$= \int_{X_2} \left(\int_{X_1} f \, d\mu_1 \right) d\mu_2. \tag{2}$$



Conclusion: big picture on our real analysis development



Some other formalization of Lebesgue analysis in some provers

- PVS (2013-)
- HOL-Light (2013-)
- Mizar (2019-)
- Isabelle/HOL (2011-)
- Lean (2021-)
- Coq.mathcomp-analysis (2021-)

Our choices: Coq with Reals, Coquelicot, Flocq

We think Coq is the most suitable tool for our goal to prove properties on the FEM algorithm and program:

- including floating point errors
- Coq provides libraries and results for the mathematical part
- Coq provides the Flocq library for floating-point arithmetic
- adapted if necessary to teaching (easy to understand)
- adapted to researchers from other fields
- continuation of work begun in Coq in the early 2000s.

A great leap forward in 25 or 30 years

- Libraries of real numbers in most provers
- Real analysis too
- Libraries adapted to different logics
- Libraries more and more adapted to the demands: research, math, education
- Automation

Future challenges

- increase user-friendliness
- increase automation
- used by other people, including students and non-logicians
- what is the proper level of abstraction ?

Non-exhaustive bibliography (1/2)

Provers and real numbers history:

1. Herman Geuvers and Rob Nederpelt; Characteristics of de Bruijn's early proof checker Automath; Fundam. Informaticae, 185, 4, 313–336; 2022
2. Freek Wiedijk; The Seventeen Provers of the World; volume 3600 of Lecture Notes in Computer Science. Springer; 2006.
3. Micaela Mayero; Formalisation et automatisation de preuves en analyses réelle et numérique; PhD thesis; Université Paris VI; 2001.
4. Micaela Mayero; Problèmes critiques et preuves formelles; Habilitation thesis; Université Paris 13; 2012.
5. Vincent Semeria; Nombres réels dans Coq; Proceedings of JFLA2020; p105-112; 2020.
6. Sylvie Boldo, Catherine Lelay and Guillaume Melquiond; Formalization of real analysis: a survey of proof assistants and libraries; Math. Struct. Comput. Sci., 26, 7, 1196–1233; 2016
7. Sylvie Boldo and Catherine Lelay and Guillaume Melquiond; Coquelicot: A User-Friendly Library of Real Analysis for Coq; Math. Comput. Sci. 2015.

Non-exhaustive bibliography (2/2)

Real analysis and floating point libraries in Coq:

flocq S.Boldo, G.Melquiond. Flocq: A Unified Library for Proving Floating-Point Algorithms in Coq. IEEE Symposium on Computer Arithmetic 2011: 243-252; 2011.

JAR13 S.Boldo, F.Clément, JC.Filliâtre, M.Mayero, G.Melquiond, and P.Weis. Wave equation numerical resolution: a comprehensive mechanized proof of a c program. J. Autom. Reasoning, 50(4):423–456, 2013.

CAMWA2014 S.Boldo, F.Clément, JC.Filliâtre, M.Mayero, G.Melquiond, and P.Weis. Trusting computations: A mechanized proof from partial differential equations to actual program. Computers & Mathematics with Applications, 68(3):325–352, 2014.

CPP17 S.Boldo, F.Clément, F.Faissole, V.Martin, and M.Mayero. A coq formal proof of the laxmilgram theorem. In Proceedings of the 6th ACM SIGPLAN Conference on Certified Programs and Proofs, CPP 2017, Paris, France, January 16-17, 2017, pages 79–89, 2017.

JAR22 S.Boldo, F.Clément, F.Faissole, V.Martin, and M.Mayero. A coq formalization of lebesgue integration of nonnegative functions. J. Autom. Reasoning, 66(2):175–213, 2022.

FM23 S.Boldo, F.Clément, V.Martin, M.Mayero, and H.Mouhcine. Lebesgue induction and tonelli's theorem in coq. 2023. To appear in Formal Methods (FM 2023).

1. coq-num-analysis: `https://lipn.univ-paris13.fr/coq-num-analysis`
2. mathcom-analysis: `https://github.com/math-comp/analysis`

Thank you