Journal of Automated Reasoning (2022) 66:175–213
https://doi.org/10.1007/s10817-021-09612-0

A Coq Formalization of Lebesgue Integration of Nonnegative Functions

Sylvie Boldo[1] · François Clément[2,3] · Florian Faissole[1] · Vincent Martin[4] · Micaela Mayero[5]

Received: 15 April 2021 / Accepted: 9 October 2021 / Published online: 23 November 2021
© The Author(s), under exclusive licence to Springer Nature B.V. 2021

Abstract
Integration, just as much as differentiation, is a fundamental calculus tool that is widely used in many scientific domains. Formalizing the mathematical concept of integration and the associated results in a formal proof assistant helps in providing the highest confidence on the correctness of numerical programs involving the use of integration, directly or indirectly. By its capability to extend the (Riemann) integral to a wide class of irregular functions, and to functions defined on more general spaces than the real line, the Lebesgue integral is perfectly suited for use in mathematical fields such as probability theory, numerical mathematics, and real analysis. In this article, we present the Coq formalization of σ-algebras, measures, simple functions, and integration of nonnegative measurable functions, up to the full formal proofs of the Beppo Levi (monotone convergence) theorem and Fatou's lemma. More than a plain formalization of the known literature, we present several design choices made to balance the harmony between mathematical readability and usability of Coq theorems. These results are a first milestone toward the formalization of L^p spaces such as Banach spaces.

Keywords Formal proof · Coq · Measure theory · Lebesgue integration

1 Introduction

This paper is dedicated to the Coq [25] formalization of Lebesgue integration theory. Among many applications in mathematics, we focus on the objective of building Sobolev spaces

✉ Sylvie Boldo
sylvie.boldo@inria.fr

1 Université Paris-Saclay, CNRS, ENS Paris-Saclay, Inria, Laboratoire Méthodes Formelles, 91190 Gif-sur-Yvette, France

2 Inria, 2 rue Simone Iff, 75589 Paris, France

3 CERMICS, École des Ponts, 77455 Marne-la-Vallée, France

4 LMAC (Laboratory of Applied Mathematics of Compiègne), CS 60319, Université de technologie de Compiègne, 60203 Compiègne Cedex, France

5 LIPN, CNRS UMR 7030, Université Paris 13, 93430 Villetaneuse, France

[2] that are used in numerous fields: in functional analysis [18,62,69], and in statistical and probabilistic mathematics [6,29,35,38,67], to name just a few.

Our main application is on the numerical resolution of partial differential equations (PDEs), using the Finite Element Method (FEM). Our final and long-term goal is to formally prove the correctness of the FEM and of parts of a library implementing it. The FEM can be applied to compute numerical approximations to solutions of many problems arising in physics, mechanics, and biology, for just a few examples. The success of the FEM is in large part due to its sound mathematical foundation, see, for instance, [22,33,61,70] among the extensive literature. Prior to this work, we established in [15] a formalization of the proof of the Lax–Milgram theorem, that is a relatively simple way of proving the existence and uniqueness of the PDE solution and their FEM approximations for a wide range of problems. The Lax–Milgram theorem is set on a general Hilbert space (a complete vector space with an inner product). In the context of PDEs, the next stage is then the application of the Lax–Milgram theorem: typically, for the Poisson equation, one takes as Hilbert space a subspace of the H^1 Sobolev space, see, for instance, [33, Sect. 3.2]. The L^p Lebesgue space is the space of functions whose absolute value to the power $p \geq 1$ is integrable, and H^1 is defined as functions in L^2 having a weak derivative also in L^2. We recall that L^p is a Banach space (a complete normed vector space), and L^2 and H^1 are Hilbert spaces. This paper deals with the construction of the Lebesgue integral for nonnegative measurable functions, a first step toward the formalization of L^p, H^1, and other Sobolev spaces. Future work will include the formal definition and the proof that they are indeed complete normed vector spaces.

As far as the integral is concerned, several options are available, e.g., see [19]. The choice must be driven by the properties required for our future developments. As mentioned before, we are more interested in the completeness of the considered functional spaces (like L^p), than in the ability to integrate the most exotic irregular functions. On the one hand, the Riemann integral is thus clearly not satisfactory as it is not compatible with limit: the limit of Riemann-integrable functions is not necessarily a Riemann-integrable function. On the other hand, the gauge (Henstock–Kurzweil) integral [4,42,47] has attractive properties, e.g., it is often considered as the easiest powerful integral to teach. Unfortunately, its main drawback is that defining a complete normed vector space of HK-integrable functions is not as obvious as with the Lebesgue L^p spaces [39,57]. This led us to choose the Lebesgue integral, which has the additional desirable property of being very general: it is neither limited to functions defined on Euclidean spaces, nor to the Lebesgue measure on $\mathbb{R}^n$.

There are also several ways to build the integral of real-valued, or complex-valued, functions for the Lebesgue measure. First, the Daniell approach [26,36] allows the extension of an *elementary* integral defined for *elementary* functions to a larger class of functions by means of continuity and linearity. When applied to the Riemann integral for continuous real-valued functions with compact support, it yields an integral equivalent to the Lebesgue integral for the Lebesgue measure. Second, a not so different alternate path consists in the completion of the normed vector space of continuous functions with compact support, and the extension of the Riemann integral which is uniformly continuous [16,28]. Third, and the option we chose to follow, is a modern form of the original works of Lebesgue [48]. The Riemann integral is based on subdivisions of the domain of the function to integrate. In contrast, the Lebesgue approach focuses on the codomain. For each preimage, we need to provide its *measure*, whatever its irregularity.

This article covers the main concepts of measure theory such as the definitions of σ-algebra, measurability of functions, measure, and simple functions. Then, the integral is built following the Lebesgue scheme: first for nonnegative simple functions, then extended to all nonnegative measurable functions by taking the supremum. The definition of the integral of

a function with arbitrary sign can be made by the difference, when possible, of the integrals of the positive and negative parts of the function; this is out of the scope of this paper and will be tackled in future work. The objective of this paper is to formally prove the main results on nonnegative measurable functions: the Beppo Levi (monotone convergence) theorem, and Fatou's lemma.

From a mathematical point of view, given a measure space defined by a set X, a σ-algebra Σ, and a measure μ, the two statements can be expressed in a mathematical setting as follows.

Textbook Theorem 1 (Beppo Levi, monotone convergence).
Let $(f_n)_{n\in\mathbb{N}}$ be a sequence of nonnegative measurable functions that is pointwise nondecreasing. Then, the pointwise limit $\lim_{n\to\infty} f_n$ is nonnegative and measurable, and we have in $\overline{\mathbb{R}}_+$

$$\int \lim_{n\to\infty} f_n \,\mathrm{d}\mu = \lim_{n\to\infty} \int f_n \,\mathrm{d}\mu. \tag{1}$$

Textbook Theorem 2 (Fatou's lemma).
Let $(f_n)_{n\in\mathbb{N}}$ be a sequence of nonnegative measurable functions. Then, the pointwise limit $\liminf_{n\to\infty} f_n$ is nonnegative and measurable, and we have in $\overline{\mathbb{R}}_+$

$$\int \liminf_{n\to\infty} f_n \,\mathrm{d}\mu \leqslant \liminf_{n\to\infty} \int f_n \,\mathrm{d}\mu. \tag{2}$$

These are the cornerstones of our intended future work, such as the building of the L^p Lebesgue spaces as Banach spaces, the proofs of Lebesgue's dominated convergence theorem and of the Tonelli–Fubini theorems, and also the construction of the Lebesgue measure (for instance, through Carathéodory's extension theorem [20,29]). As a consequence, we do not yet need technical results on subset systems such as the Dynkin π–λ theorem [29], or the monotone class theorem [24], that are popular tools for the extension of some property to the whole σ-algebra (e.g., the uniqueness of a measure).

Interactive theorem proving is more and more being used and adapted for formalizing real and numerical analysis. Real-life applications, such as hybrid systems or cyber-physical systems, are critical and rely on advanced analysis results. Until now, only the Riemann integral was available in Coq. As useful as the Riemann integral is, the Lebesgue integral is necessary for the numerical analysis we are examining. In addition, even though the Lebesgue integral exists in other theorem provers (see Sect. 10), we have decided to formalize it in Coq. Indeed, it is crucial for our future work to be able to merge results both from numerical analysis and from computer arithmetic (to bound rounding errors for instance). For that, we plan to rely on the Flocq library, which does not have a comparable equivalent in other theorem provers.

We use the Coquelicot library [13], a modernization of the real standard library of Coq, including a formalization of $\overline{\mathbb{R}}$, described in more detail in Sect. 2.1. This library provides classical real numbers which correspond to the real analysis we deal with. For this reason, we have also decided, as basic choices of our formalization, to use classical logic and to rely on the following axioms: strong excluded middle and functional extensionality. These choices are described in Sect. 2.2 and discussed in Sect. 9.

The mathematical definitions and proofs were mainly taken from textbooks [37,40,51], detailed and compiled in a research report [23] in order to ease the formalization in Coq. The Coq code is available at

http://lipn.univ-paris13.fr/MILC/CoqLIntp/index.php,
or in the public repository
https://lipn.univ-paris13.fr/coq-num-analysis/tree/LInt_p.1.0/Lebesgue,
where the tag `LInt_p.1.0` corresponds to the code of this article.

The paper is organized as follows. Section 2 presents the main basic Coq choices on which our formalization is based. The sequel is our own contribution. Section 3 details auxiliary results on reals. The concept of measurability is discussed in Sect. 4, and that of measure in Sect. 5. Section 6 is devoted to simple functions, and Sect. 7 to integration of nonnegative functions and the main theorems. The case of the Dirac measure is studied in Sect. 8. Concerns about proof engineering are discussed in Sect. 9. Section 10 presents some state of the art of the formalization of the integral. Section 11 concludes and gives some perspectives.

2 Coquelicot Library and Other Basic Coq Choices

We first briefly review the few proof packages used in this work, and some technical and logical choices we made. These are discussed further in Sects. 9.1 and 9.2.

2.1 The Coquelicot Library and $\overline{\mathbb{R}}$

The Coquelicot library is a conservative extension of the Coq real standard library (Reals), with total functions for limit, derivative, and Riemann integral [13,49,50]. The features used here are the generic topology, the hierarchy of algebraic structures based on canonical structures, and the extended real numbers.

Generic topology. The Coquelicot topology is defined using filters [17,21]. Intuitively, filters can be seen as sets of neighborhoods. For instance, the filter `eventually` on type `nat` corresponds to the most intuitive neighborhoods of ∞.

```
Definition eventually : (nat → Prop) → Prop := fun P ⇒ ∃ N, ∀ n, N ⩽ n → P n.
```

It is used to define the convergence of sequences.

Algebraic hierarchy. Coquelicot also defines an algebraic hierarchy based on canonical structures. A useful level here is `UniformSpace`, that formalizes the mathematical concept of uniform space [17,68]: it is a generalization of metric space with an abstraction of balls. In a uniform space `E`, the property `open : (E → Prop) → Prop` characterizes its open subsets.

Extended real numbers. Coquelicot provides a definition of the extended real numbers $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$. The formal definition contains three constructors: `Finite` for real numbers, `p_infty` for ∞ and `m_infty` for $-\infty$. Conversely, the function `real` returns the real number for finite numbers and 0 for $\pm\infty$.

```
Inductive Rbar :=
 | Finite : R → Rbar
 | p_infty : Rbar
 | m_infty : Rbar.

Definition real : Rbar → R :=
 fun x ⇒ match x with
  | Finite r ⇒ r
  | _ ⇒ 0
  end.
```

In addition to this definition, coercions from `R` to `Rbar` and *vice versa*, an order with `Rbar_lt` and `Rbar_le`, total operations such as `Rbar_opp`, `Rbar_plus`, `Rbar_minus`, `Rbar_inv`, `Rbar_mult`, `Rbar_min` and `Rbar_abs` with their properties are provided.

In particular, this means that addition on $\overline{\mathbb{R}}$ is a total function [13] that always returns a value. For instance, $\infty + (-\infty)$ (i.e., $\infty - \infty$) is 0, making some statements unintuitive, see also Sect. 4.5. However, the case of multiplication is not an issue as the convention $0 \times \pm\infty = \pm\infty \times 0 = 0$ is widely adopted for measure theory and Lebesgue integration, because it yields more compact statements.

2.2 Axioms

Real analysis, as most mathematics, uses classical logic, and measure theory and Lebesgue integration are no exception. For this reason, we chose to conduct this formalization in a full-flavored classical framework.

We did not add our own axioms. In addition to the axioms defining `R`, we require some classical properties from the standard library, listed here with the theorems we use.

```
Require Import ClassicalDescription.
Require Import PropExtensionality.
Require Import FunctionalExtensionality.
Require Import ClassicalChoice.

Check excluded_middle_informative.
      : ∀ (P : Prop), {P} + {¬ P}

Check propositional_extensionality.
      : ∀ (P Q : Prop), P ↔ Q → P = Q

Check functional_extensionality.
      : ∀ {A B : Type} (f g : A → B), (∀ x, f x = g x) → f = g

Check choice.
      : ∀ (A B : Type) (R : A → B → Prop),
        (∀ x, ∃ y, R x y) → ∃ (f : A → B), ∀ x, R x (f x)
```

We rely on `excluded_middle_informative` many times, including for instance the definition of the characteristic function in Sect. 4.1. We have a brief use of dependent types in Sect. 6.2 related to simple functions, and we then rely on `propositional_extensionality` and `functional_extensionality`. Last, we rely on `choice` at a single point in the proof of Lemma `negligible_union_countable`, and this is explained in Sect. 5.3.

3 Auxiliary Results About the Reals

From now on, we present our own contributions. A global dependency graph of our Coq files and results is given in Fig. 1 page 38, with links back to the appropriate sections.

The theorems described in this section are not dedicated to the Lebesgue integral and could be part of a support library. In Sect. 3.1, we show the expression of open subsets of $\mathbb{R}$ and $\mathbb{R}^2$ with a countable topological basis. Section 3.2 deals with sums on $\overline{\mathbb{R}}$. Section 3.3 presents some additional results about limits.

3.1 Second-Countability of Real Numbers

In Sect. 4.4, we need to characterize and decompose the open subsets of $\mathbb{R}$. More precisely, we build generators of the σ-algebras of $\mathbb{R}$ and $\mathbb{R}^2$ that contain the open subsets. Such generators need to be of countable size to comply with the properties of σ-algebras. Thus, the concepts of topological basis and second-countability appeared necessary.

Recall that a topological basis allows to express any open subset of a topological space as the union of a subfamily of the basis. A topological space is called *second-countable* when it admits a countable topological basis. Euclidean spaces $\mathbb{R}^n$ are second-countable. Indeed, the open boxes with rational boundaries form such a countable topological basis. In the case of $\mathbb{R}$, this is expressed as follows. For any open subset A of $\mathbb{R}$, there exists a sequence of pairs of rationals $(q_1^n, q_2^n)_{n\in\mathbb{N}}$ such that A is the union of the corresponding open intervals, $A = \bigcup_{n\in\mathbb{N}}(q_1^n, q_2^n)$.

The mathematical proof is well known, but the road to formalization was tedious.

Countability. We define bijections from $\mathbb{N}$ to $\mathbb{N}^2$, $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{Q}^2$. It is not enough to prove they have the same size, we need "perfect" bijections, meaning inverse functions from one type to the other, handling correctly special cases such as zero.

Connected components. Given a subset A of $\mathbb{R}$ and a real x, we define the bounds of the largest possible interval included in A and containing x (a.k.a. the connected component of x in A).

```
Definition bottom_interv : (R → Prop) → R → Rbar :=
  fun A x ⇒ Glb_Rbar (fun z ⇒ ∀ y, z < y < x → A y).

Definition top_interv : (R → Prop) → R → Rbar :=
  fun A x ⇒ Lub_Rbar (fun z ⇒ ∀ y, x < y < z → A y).
```

The functions `Glb_Rbar` and `Lub_Rbar` are total functions from the Coquelicot library that compute the greatest lower bound and the least upper bound of a subset of reals.

We prove many properties such as belonging to the closure of its own connected component, `bottom_interv A x ⩽ x ⩽ top_interv A x`, and belonging to the interior of its own connected component for points of open subsets, that is `open A → A x → bottom_interv A x < x < top_interv A x`.

Using density of rational numbers. Given an open subset A of $\mathbb{R}$, we prove through density of $\mathbb{Q}$ in $\mathbb{R}$ that A contains at most a countable number of connected components.

```
Lemma open_R_charac_Q :
  ∀ (A : R → Prop), open A →
    ∀ x, A x ↔ (∃ q : Q, let y := Q2R q in
      A y ∧ Rbar_lt (bottom_interv A y) x ∧ Rbar_lt x (top_interv A y)).
```

Countability appears in this lemma through the rationality of y (otherwise, the theorem would be trivial by using x).

In addition, using again the density of $\mathbb{Q}$ in $\mathbb{R}$, we can take rational bounds for these intervals (by taking countable unions of intervals with rational bounds to recover each initial interval with real bounds). And then, using countability of $\mathbb{Q}^2$, we have a bijection from the integers to the rational bounds of the open intervals, and these serve as topological basis.

```
Definition topo_basis_R : nat → R → Prop :=
  fun n x ⇒ Q2R (fst (bij_NQ2 n)) < x < Q2R (snd (bij_NQ2 n)).
```

Second-countability. Given an open subset A of $\mathbb{R}$, we want to exhibit the q_i^n's such that $A = \bigcup_{n\in\mathbb{N}}(q_1^n, q_2^n)$. This means we need to choose among the possible intervals of the

topological basis the useful ones by relying on a property P. Then, A is equivalent to the countable union of the `topo_basis_R n` such that `P n` holds.

```
Lemma R_second_countable :
 ∀ (A : R → Prop), open A →
  ∃ (P : nat → Prop), (∀ x, A x ↔ ∃ n, P n ∧ topo_basis_R n x).
```

The same property holds for $\mathbb{R}^2$. We can define a topological basis for $\mathbb{R}^2$ (from the tensor product of the topological basis of $\mathbb{R}$) and prove

```
Lemma R2_second_countable :
 ∀ (A : R * R → Prop), open A →
  ∃ (P : nat → Prop), (∀ x, A x ↔ ∃ n, P n ∧ topo_basis_R2 n x).
```

3.2 About Sums of Extended Real Numbers

Integrals of simple functions are defined in Sect. 6.2 as sums of extended reals. Even if we only sum nonnegative extended reals, we decided to use only `Rbar` as discussed in Sect. 9.1. But as in mathematics, the addition on `Rbar` as defined by Coquelicot is not associative. Indeed, $\infty + (\infty - \infty) = \infty$, while $(\infty + \infty) - \infty = 0$. Our design choice therefore implies that big operators [5] cannot be used.

Let us begin with sums of a finite number of values. The definition goes as expected, with an equivalent alternative using `fold_right` for lists instead of functions.

```
Fixpoint sum_Rbar n (f : nat → Rbar) : Rbar :=
 match n with
 | 0 ⇒ f 0%nat
 | S n1 ⇒ Rbar_plus (f (S n1)) (sum_Rbar n1 f)
 end.

Definition sum_Rbar_l : list Rbar → Rbar := fun l ⇒ fold_right Rbar_plus 0 l.
```

In addition, we found it useful to define an "applied" sum that takes a function f and a list ℓ and returns the sum of the images $\Sigma_{i \in \ell} f(i)$.

```
Definition sum_Rbar_map : ∀ {E : Type}, list E → (E → Rbar) → Rbar :=
 fun E l f ⇒ sum_Rbar_l (map f l).
```

The curly brackets around `E` mean that this argument is implicit and need not be specified, as Coq can guess it from the type of the list `l`.

This definition allows us to use extensionality either on the list `l`, on the function `f`, or on the application `map f l`, which turned out to be more practical than what this obvious definition seems. Examples of use are the following lemmas (that do not need nonnegativity). The first one mixes two applications.

```
Lemma sum_Rbar_map_map :
 ∀ {E F : Type} (f : E → F) (g : F → Rbar) (l : list E),
  sum_Rbar_map (map f l) g = sum_Rbar_map l (fun x ⇒ g (f x)).
```

The second one focuses on the statement $\Sigma_{i \in \ell_1} f(i) = \Sigma_{i \in \ell_2} f(i)$. Such result is obvious when $\ell_1 = \ell_2$, but it is also possible to prove it when the lists are identical except for items i of the lists such that $f(i) = 0$, as these do not impact the final sums. Indeed, the sums may be the same even if the two lists are different (and of different lengths for instance). The function `select` is defined later in Sect. 6.1. It has type `(E → Prop) → list E → list E` and selects the elements of a list that have a given property, without changing otherwise the order in the lists.

```
Lemma sum_Rbar_map_ext_l :
 ∀ {E : Type} (l1 l2 : list E) (f : E → Rbar),
  select (fun x ⇒ (f x ≠ 0)) l1 = select (fun x ⇒ f x ≠ 0) l2 →
  sum_Rbar_map l1 f = sum_Rbar_map l2 f.
```

When values are nonnegative, associativity is back and we have the expected theorems on sums.

```
Lemma sum_Rbar_end :
 ∀ f n, (∀ i, (i ⩽ S n)%nat → Rbar_le 0 (f i)) →
  (sum_Rbar (S n) f = Rbar_plus (f 0%nat) (sum_Rbar n (fun i ⇒ f (S i)))).

Lemma sum_Rbar_l_concat :
 ∀ (l1 l2 : list Rbar), non_neg_l l1 → non_neg_l l2 →
  sum_Rbar_l (l1 ++ l2) = Rbar_plus (sum_Rbar_l l1) (sum_Rbar_l l2).
```

For the sake of brevity, we have defined the properties `non_neg` and `non_neg_l` for non-negative functions and lists.

The most interesting theorem about sums of lists is the ability to swap the order of a double summation,

$$\sum_{i_1 \in \ell_1} \sum_{i_2 \in \ell_2} f(i_1, i_2) = \sum_{i_2 \in \ell_2} \sum_{i_1 \in \ell_1} f(i_1, i_2).$$

```
Lemma sum_Rbar_map_switch :
 ∀ {E : Type} (f : E → E → Rbar) l1 l2,
  (∀ x y, In x l1 → In y l2 → Rbar_le 0 (f x y)) →
  sum_Rbar_map l1 (fun x ⇒ sum_Rbar_map l2 (fun y ⇒ f x y)) =
   sum_Rbar_map l2 (fun y ⇒ sum_Rbar_map l1 (fun x ⇒ f x y)).
```

3.3 About Limits

We also need some additional results on limits and suprema.

First of all, the sums defined in Sect. 3.2 have a finite number of terms. But the main theorems to come rely on infinite sums (i.e., series). The most common definition is the limit of the finite partial sums, i.e., `Lim_seq` in Coquelicot [13]. Nevertheless, by virtue of the least-upper-bound property in $\mathbb{R}$ and $\overline{\mathbb{R}}$, when a sequence is increasing (which happens when adding only nonnegative values), the supremum is also the limit, and we may equivalently use `Sup_seq` instead. This has proved more convenient and more suited to our needs. So theorems of Sect. 7 such as the Beppo Levi theorem rely on `Sup_seq`.

Next, we are interested in the limit inferior of sequences in $\overline{\mathbb{R}}$. But, Coquelicot only provides `LimInf_seq` of type `(nat → R) → Rbar`, and nothing for `nat → Rbar` sequences. Therefore, we defined a minor variant of the desired type and proved a few lemmas by directly copying their proofs from those for `LimInf_seq` in Coquelicot.

```
Definition LimInf_seq' : (nat → Rbar) → Rbar :=
 fun u ⇒ Sup_seq (fun m ⇒ Inf_seq (fun n ⇒ u (n + m)%nat)).
```

4 Measurability

We present now the formalization of σ-algebras, which are defined as an inductive type. They characterize *measurable* subsets, and particular attention is paid to $\mathbb{R}$, $\overline{\mathbb{R}}$ and $\mathbb{R}^2$, where the open subsets generate the *Borel* measurable subsets.

The issue of subsets is briefly addressed in Sect. 4.1. Section 4.2 is devoted to the measurability of subsets, and Sect. 4.3 to Cartesian products. The Borel subsets of $\mathbb{R}$ and $\overline{\mathbb{R}}$ are detailed in Sect. 4.4. And Sect. 4.5 deals with the measurability of functions.

4.1 Subsets and Characteristic Functions

We consider a generic set `E` defined in Coq as `E : Type`. Usually, subsets of `E` are defined in Coq as having type `E → Prop`, or `E → bool`. We choose `Prop`, and this is discussed in Sect. 9.2. Then, the power set of `E` has type `(E → Prop) → Prop`.

Given a subset A, we define its characteristic function (or indicator function) $\mathbb{1}_A$ that maps elements of A to 1, and others to 0.

```
Context {E : Type}.

Definition charac : (E → Prop) → E → R :=
 fun A x ⇒ match (excluded_middle_informative (A x)) with
  | left _ ⇒ 1
  | right _ ⇒ 0
  end.
```

Indeed, it is very convenient for direct use in arithmetic expressions without exhibiting the membership conditional in a dependent type or an assumption. It is used a lot in the context of simple functions in Sect. 6.

The characteristic function is also convenient to simulate the restriction of a numerical function to a subset, for instance, in Sect. 4.5. More precisely, the mathematical function $f_{|A}$ could be formalized either as a record with a dependent type or as a total function. We have explored the first way which became impractical as proofs creep into our statements and prevent some rewritings. The total function is then $f \times \mathbb{1}_A$, which is the correct value when needed and 0 elsewhere. This is perfectly suited to our context, as integrating zero has no impact.

4.2 Measurability of Subsets

The design choice for the measurability of subsets, i.e., the definition of σ-algebra, is a central issue for this paper. Even though several equivalent definitions are possible, the use of an inductive type has proved successful, with several proofs done by induction.

Among several possible informal definitions [23, Section 8.6], a σ-algebra is a subset of the power set that contains the empty set and is closed under complement and countable unions. In fact, a σ-algebra can be really huge and it is very convenient to represent it with a smaller collection G of so-called generators, and to consider the smallest σ-algebra containing G. This corresponds to the informal concept of *generated* σ-algebra. Indeed, in many situations, it is sufficient to establish a property on G to have it on the whole σ-algebra generated by G.

While this may suggest the use of a record, we rely on an inductive type. More precisely, we "start" with a collection of generators `genE : (E → Prop) → Prop`. Then, a subset is measurable if it is either a generator, empty, the complement of a measurable subset, or

the countable union of measurable subsets. This design choice is discussed in Sect. 9.3. Note that the issue of generators is at the center of Sect. 4.3 for Cartesian products and discussed in Sect. 4.4 for the Borel subsets of real numbers.

```
Variable genE : (E → Prop) → Prop.

Inductive measurable : (E → Prop) → Prop :=
 | measurable_gen : ∀ A, genE A → measurable A
 | measurable_empty : measurable (fun _ ⇒ False)
 | measurable_compl : ∀ A, measurable (fun x ⇒ ¬ A x) → measurable A
 | measurable_union_countable :
    ∀ (A : nat → E → Prop), (∀ n, measurable (A n)) → measurable (fun x ⇒ ∃ n, A n x).
```

From this definition, we then prove various lemmas, relying on our classical setting, such as measurability of the full set, and of countable intersections.

```
Lemma measurable_inter_countable :
 ∀ (A : nat → E → Prop), (∀ n, measurable (A n)) → measurable (fun x ⇒ ∀ n, A n x).
```

A mathematically unexpected, but quite useful theorem is the following.

```
Lemma measurable_Prop : ∀ P, measurable (fun _ ⇒ P).
```

Constant properties (that do not depend on a variable), be they true or false, are measurable as both `True` (the full set) and `False` (the empty set) are measurable. When decomposing a subset to prove its measurability, this comes in handy.

In many situations, several collections of generators are possible, and switching between them may be convenient for the proof at hand. In fact, if G_1 is included in the σ-algebra generated by G_2, and *vice versa*, then both generated σ-algebras are the same. This yields the following extensionality result.

```
Lemma measurable_gen_ext :
 ∀ genE1 genE2,
  (∀ A, genE1 A → measurable genE2 A) → (∀ A, genE2 A → measurable genE1 A) →
  (∀ A, measurable genE1 A ↔ measurable genE2 A).
```

We now define what is a σ-algebra, but this definition is hardly used later on as we rely mostly on the previous inductive. A σ-algebra is formally defined as a subset of the power set that is equal to the σ-algebra induced by itself as generator.

```
Definition is_sigma_algebra: ((E → Prop) → Prop) → Prop :=
 fun calS ⇒ calS = measurable calS.
```

We have the equivalence with one of the commonly used mathematical definitions: $\mathcal{S}$ is a σ-algebra when it contains the empty set and is closed under complement and countable unions.

```
Lemma is_sigma_algebra_correct :
 ∀ calS, is_sigma_algebra calS ↔
  (calS (fun _ ⇒ False) ∧
  (∀ A, calS (fun x ⇒ ¬ A x) → calS A) ∧
  (∀ (A : nat → E → Prop), (∀ n, calS (A n)) → calS (fun x ⇒ ∃ n, A n x))).
```

We can of course prove that the basic σ-algebras are indeed compliant with our definition, be it the discrete σ-algebra (the whole power set), or the trivial σ-algebra (reduced to $\{\varnothing, E\}$).

```
Lemma is_sigma_algebra_discrete : is_sigma_algebra (fun _ ⇒ True).
Lemma is_sigma_algebra_trivial : is_sigma_algebra (fun A ⇒ (∀ x, ¬ A x) ∨ (∀ x, A x)).
```

An immediate consequence of the extensionality result about generators is the idempotence of σ-algebra generation. Indeed, the σ-algebra generated by a given generated σ-algebra is the very same σ-algebra. This may be expressed in Coq as `is_sigma_algebra (measurable genE)`, showing that our definition is indeed a σ-algebra in the mathematical sense. In addition, the definition by induction gives us for free that our definition represents the smallest generated σ-algebra.

To sum up, in our development, the measurability of subsets of `E : Type` is built by induction from a *generator* `genE : (E → Prop) → Prop`, providing a σ-algebra.

4.3 Cartesian Product and Measurability

Although we do not deal with the Tonelli–Fubini theorems in this paper, the Cartesian product is used in Sect. 4.5 to establish measurability of the addition and multiplication of two measurable numerical functions.

Given two measurable spaces, i.e., two sets E and F and their associated generators G_E and G_F, it is natural to ask the question of measurability on the Cartesian product $E \times F$; but with which σ-algebra? Among other possibilities, the tensor product of the two σ-algebras is of paramount interest, since it makes both canonical projections (the maps $((x_E, x_F) \mapsto x_E)$ and $((x_E, x_F) \mapsto x_F)$) measurable. It is the σ-algebra generated by the Cartesian products of measurable subsets of E and F.

Unfortunately, on the matter of generator, simply taking the Cartesian products of elements of G_E and G_F is not correct: in this case, for instance, one cannot prove the measurability of $A_E \times F$, for $A_E \in G_E$. We need to add the full sets to the initial generator, using the following definition.

```
Definition gen2 : (E * F → Prop) → Prop :=
  fun A ⇒ ∃ AE AF, (genE AE ∨ AE = fun _ ⇒ True) ∧ (genF AF ∨ AF = fun _ ⇒ True) ∧
    (∀ X, A X ↔ AE (fst X) ∧ AF (snd X)).
```

And we prove this satisfies the desired property.

```
Lemma gen2_is_product_measurable :
  ∀ AE AF, measurable genE AE → measurable genF AF →
    measurable (gen2 genE genF) (fun X ⇒ AE (fst X) ∧ AF (snd X)).
```

4.4 Borel Subsets of Real Numbers

We specify now an important class of σ-algebras. When the measurable space has also a topological space structure (e.g., `UniformSpace` in Coquelicot, see Sect. 2.1), one usually selects the Borel σ-algebra. It is generated by all the open subsets, or equivalently by all the closed subsets, and has the nice property of providing measurability for continuous functions (see Sect. 4.5).

Lebesgue integration theory is essentially meant for real-valued functions (or with codomain $\overline{\mathbb{R}}$, $\mathbb{R}^n$, or $\mathbb{C}^n$). Thus, we need to equip $\mathbb{R}$, and $\overline{\mathbb{R}}$, with their Borel σ-algebras, and we have some leeway in choosing the generators, instead of all open subsets. Now, we present our choice and also prove that other possibilities define the same σ-algebras.

Borel subsets on $\mathbb{R}$. Among many possibilities, we pick the closed intervals (of the form $[a, b]$, with $a \leqslant b$ reals) for $\mathbb{R}$.

```
Definition gen_R_cc : (R → Prop) → Prop := fun A ⇒ ∃ a b, (∀ x, A x ↔ a ⩽ x ⩽ b).
```

This choice of `gen_R_cc` is somewhat arbitrary and could be changed. Thus, we introduce an anonymous `gen_R` that will be used in the sequel of the paper for the definition of measurable subsets of $\mathbb{R}$.

```
Definition gen_R := gen_R_cc.
Definition measurable_R : (R → Prop) → Prop := measurable gen_R.
```

Other choices for $\mathbb{R}$ include the open intervals, or of the form $[a, b)$, or the open left-rays of the form $(-\infty, b)$. We proved, for instance, that measurability on $\mathbb{R}$ generated by closed intervals (our definition) is the same as measurability generated by open intervals.

```
Definition gen_R_oo : (R → Prop) → Prop := fun A ⇒ ∃ a b, ∀ x, A x ↔ a < x < b.
Lemma measurable_R_equiv_oo : ∀ A, measurable_R A ↔ measurable gen_R_oo A.
```

The proof is a call to the generator extensionality lemma and then relies on basic properties of measurability (closedness under complement and countable union), and on the definition of a nested sequence of closed intervals (from `gen_R_cc`) whose union is an open interval (from `gen_R_oo`), thanks to the Archimedean property of $\mathbb{R}$. Moreover, from the density of $\mathbb{Q}$ in $\mathbb{R}$, we may only consider open intervals with rational endpoints.

```
Definition gen_R_Qoo : (R → Prop) → Prop := fun A ⇒ ∃ a b, ∀ x, A x ↔ Q2R a < x < Q2R b.
Lemma measurable_R_equiv_Qoo : ∀ A, measurable_R A ↔ measurable gen_R_Qoo A.
```

And finally, more interestingly from a mathematical viewpoint, we prove that our measurable subsets on $\mathbb{R}$ (based on closed intervals) are indeed the Borel subsets generated by `open` from `UniformSpace`.

```
Lemma measurable_R_open : ∀ A, measurable_R A ↔ measurable open A.
```

The proof is simply an application of lemma `R_second_countable` from Sect. 3.1 stating that any open subset is the countable union of open intervals with rational endpoints. This is needed in Sect. 4.5 where the measurability of the addition of two measurable real-valued functions relies on the continuity of the addition in $\mathbb{R}$.

Borel subsets on $\mathbb{R}^2$. Combining the generator for a Cartesian product of Sect. 4.3 and the second-countability of $\mathbb{R}^2$ of Sect. 3.1, we have an equivalence result for the Borel subsets of $\mathbb{R}^2$.

```
Definition gen_R2 : (R * R → Prop) → Prop := gen2 gen_R gen_R.
Definition measurable_R2 : (R * R → Prop) → Prop := measurable gen_R2.
Lemma measurable_R2_open : ∀ (A : R * R → Prop), measurable_R2 A ↔ measurable open A.
```

Here, `open` stands for the open subsets of $\mathbb{R}^2$. The canonical structures of Coquelicot deduce that $\mathbb{R}^2$, as product of two `UniformSpace`s, is a `UniformSpace`.

Borel subsets on $\overline{\mathbb{R}}$. For $\overline{\mathbb{R}}$, the generators we choose are the closed right-rays (of the form $[a, \infty]$, $a \in \overline{\mathbb{R}}$), but we also define an anonymous `gen_Rbar`.

```
Definition gen_Rbar_cu : (Rbar → Prop) → Prop := fun A ⇒ ∃ a, ∀ x, A x ↔ Rbar_le a x.
Definition gen_Rbar := gen_Rbar_cu.
Definition measurable_Rbar : (Rbar → Prop) → Prop := measurable gen_Rbar.
```

We proved the equivalence with the measurability defined by closed left-rays (of the form $[-\infty, a]$). Unlike $\mathbb{R}$, the measurability of the addition of two measurable $\overline{\mathbb{R}}$-valued functions does not rely on continuity anymore (see Sect. 4.5), and we did not prove that our measurable subsets on $\overline{\mathbb{R}}$ (based on closed rays) are indeed the Borel subsets generated by the open subsets of $\overline{\mathbb{R}}$, as we do not need it for now.

Next, we proved that measurability is compatible with scaling.

```
Lemma measurable_scal_Rbar :
  ∀ A l, measurable_Rbar A → measurable_Rbar (fun x ⇒ A (Rbar_mult l x)).
```

Note that ℓ may be any extended real, even 0 or $\pm\infty$. So one may imagine the numerous subcases to ensure this lemma.

4.5 Measurability of Functions

From the measurability of subsets defined above, we can now define the measurability of a function.

General case. Given two sets E and F and associated generators G_E and G_F, a function $f : E \to F$ is measurable when for every measurable subset A, the subset $f^{-1}(A)$ is measurable, i.e., $\{x \mid f(x) \in A\}$ is measurable. Note that f^{-1} is obviously understood as a function from the power set of F to the one of E. The measurability is then defined in Coq as follows.

```
Definition measurable_fun : (E → F) → Prop :=
  fun f ⇒ ∀ A, measurable genF A → measurable genE (fun x ⇒ A (f x)).
```

We then prove some basic properties. For instance, it is enough to consider the generators to ensure the measurability of a function.

```
Lemma measurable_fun_gen :
  ∀ (f : E → F), measurable_fun f ↔ (∀ A, genF A → measurable genE (fun x ⇒ A (f x))).
```

When `E` and `F` are also `UniformSpace` (from Coquelicot, see Sect. 2.1), the use of Borel σ-algebras (generated by the open subsets) ensures that continuous functions are measurable. As explained in Sect. 2.1, the continuity definition is based on filters.

```
Lemma measurable_fun_continuous :
  ∀ f, (∀ x, continuous f x) → measurable_fun open open f.
```

This is simply due to the fact that the inverse image of an open subset by a continuous function is an open subset.

Case of numerical functions. Now let us consider the case of numerical functions, with codomain $\mathbb{R}$, or $\overline{\mathbb{R}}$. The definition relies on the generators `gen_R` and `gen_Rbar` defined above.

```
Definition measurable_fun_R : (E → R) → Prop := measurable_fun genE gen_R.
Definition measurable_fun_Rbar : (E → Rbar) → Prop := measurable_fun genE gen_Rbar.
```

Later on, we have to deal with piecewise-defined functions, and in such a situation, it is interesting to treat each piece separately, and to use the restriction defined in Sect. 4.1 as the multiplication by the characteristic function. The following result, simple but useful, states that given a measurable subset A and a measurable function g, given a function f equal to g on A, then $f \times \mathbb{1}_A$ is measurable. Its proof is rather easy given the proved properties of the measurability of subsets.

```
Lemma measurable_fun_when_charac :
  ∀ (f g : E → Rbar) A, measurable genE A →
    (∀ x, A x → f x = g x) → measurable_fun_Rbar g →
    measurable_fun_Rbar (fun x ⇒ Rbar_mult (f x) (charac A x)).
```

The main mathematical result of the rest of this section is the compatibility of measurability of functions with algebraic operations (addition, scalar multiplication and multiplication); the most complex one being the addition. From the mathematical standpoint, when extended

real values are involved, it is assumed that these operations are well defined. In Coq, when using operations on `Rbar` from Coquelicot that are total functions, the situation is different, and somewhat more complex as explained below.

Functions to $\mathbb{R}$. Let us prove first the measurability of the sum of two measurable real-valued functions.

```
Lemma measurable_fun_Rplus :
 ∀ f1 f2, measurable_fun_R f1 → measurable_fun_R f2 →
  measurable_fun_R (fun x ⇒ f1 x + f2 x).
```

The proof uses the compatibility of measurability with the composition of functions: if both f and g are measurable, then so is $f \circ g$. This is applied to $f \stackrel{\text{def.}}{=} ((x, y) \mapsto x + y)$ of type $\mathbb{R}^2 \to \mathbb{R}$ and $g \stackrel{\text{def.}}{=} (x \mapsto (f_1(x), f_2(x)))$ of type $\mathbb{R} \to \mathbb{R}^2$.

Measurability on $\mathbb{R}^2$ relies on `gen_R2`, the generator of the Borel subsets of $\mathbb{R}^2$ defined in Sect. 4.4. The proof is based on the generator equivalence between `gen_R2` and `open`, and on the continuity of addition. This proof was not difficult, but happened to be much higher-level than expected. The multiplication of real-valued functions is treated exactly in the same way.

Scalar multiplication for measurable functions is deduced from a similar theorem about scalar multiplication for measurable subsets. In the end, the measurable real-valued functions form an algebra (over the field $\mathbb{R}$); however, we have not stated it (with canonical structures for instance) as we have no use for it, but all the needed lemmas are proved.

Functions to $\overline{\mathbb{R}}$. Let us consider now the addition of measurable extended real-valued functions. The semantics of $+_{\overline{\mathbb{R}}}$ is more complex, as it raises the question of what is $\infty - \infty$. We rely on the Coquelicot definition of `Rbar_plus`. As a total function, it returns 0 in this special case, see Sect. 2.1. The proof for $\mathbb{R}$ was based on the continuity of $+$; but that cannot be used here, as $+_{\overline{\mathbb{R}}}$ is not continuous on the whole set $\overline{\mathbb{R}}^2$ (there are problems at infinity, even for the total function).

In order to stick closely to the mathematics, we rely on a way to express the legality of addition: the property `ex_Rbar_plus` that basically prevents adding ∞ and $-\infty$. Thus, we prove the following theorem.

```
Lemma measurable_fun_plus :
 ∀ f1 f2, measurable_fun_Rbar f1 → measurable_fun_Rbar f2 →
  (∀ x, ex_Rbar_plus (f1 x) (f2 x)) →
   measurable_fun_Rbar (fun x ⇒ Rbar_plus (f1 x) (f2 x)).
```

The proof is a little tedious as it splits E into all the possible cases using `measurable_fun_when_ charac`: when both $f_1(x)$ and $f_2(x)$ are finite, the previous theorem on $\mathbb{R}$ is used. Otherwise, the preimages of $\pm\infty$ are measurable since singletons are (as closed subsets). Thus, we are able to finish all the cases.

Among the peculiarities of Coq compared to mathematics, note that a simpler theorem can be devised.

```
Lemma measurable_fun_plus' :
 ∀ f1 f2, measurable_fun_Rbar f1 → measurable_fun_Rbar f2 →
  measurable_fun_Rbar (fun x ⇒ Rbar_plus (f1 x) (f2 x)).
```

It states the same conclusion, but without assuming the legality of addition. Indeed, the total function $(x \mapsto f_1(x) +_{\overline{\mathbb{R}}} f_2(x))$, with value 0 when both operands are infinite opposites, is actually measurable. This subtlety when considering $\infty - \infty$ is related to total functions, a design choice that prevents dependent types but may give strange results when out of the domain of the function. This strangeness also exists in the Coq standard library of reals [54]

when considering the division as a total function, making 1/0 a valid real. This hard question would be solved more naturally in other provers, for instance, in PVS relying on TCCs (Type-Correctness Conditions) [60]. To conclude, the main problem with this theorem is that it does not state what the mathematicians read in it, so we have decided not to use it.

The multiplication of two functions taking their values in $\overline{\mathbb{R}}$ is treated similarly. However, it does not raise the same issues as addition, because Coquelicot and mathematics for measure theory use the same convention $\pm\infty \times 0 = 0$, see Sect. 2.1. Multiplication by a scalar is deduced from a similar theorem on measurable subsets. Note that in contrast to the case of $\mathbb{R}$, measurable functions with values in `Rbar` do not form an algebra, as `Rbar_plus` is not associative, see Sect. 3.2.

5 Measure

A measurable space with a σ-algebra can be equipped with a measure. A measure is a mapping from measurable subsets to nonnegative extended real values that satisfies additivity properties. Some well-known measures are the Lebesgue measure, the counting measure, the Dirac measure (see Sect. 8), and numerous probability measures (that take values in the interval [0, 1]).

Measure theory is a general abstract setting that applies to any measure, and the axiomatization of their fundamental properties is formalized here with an instantiation in Sect. 8.

5.1 Specification and Basic Properties

Given a measurable space defined by a set `E : Type` and a generator `genE` of type `(E → Prop) → Prop` (see Sect. 4.2), our design choice is to specify measures as a `Record` type containing a map `meas : (E → Prop) → Rbar` together with the fundamental properties making this map a measure.

```
Record measure := mk_measure {
  meas :> (E → Prop) → Rbar;
  meas_False : meas (fun _ ⇒ False) = 0;
  meas_ge_0 : ∀ A, Rbar_le 0 (meas A);
  meas_sigma_additivity : ∀ A : nat → E → Prop,
    (∀ n, measurable genE (A n)) → (∀ n m x, A n x → A m x → n = m) →
    meas (fun x ⇒ ∃ n, A n x) = Sup_seq (fun n ⇒ sum_Rbar n (fun m ⇒ meas (A m)))}.
```

The measure is defined as a record. For the sake of brevity, we want to use it directly as a function, so we have a coercion (hence the symbol `:>`) between the type `measure` and `(E→ Prop)→ Rbar`.

The first two properties `meas_False` and `meas_ge_0` are self-explanatory. Using standard mathematical notations ($\uplus$ denotes the disjoint union), the σ-additivity of a map μ means that for any sequence $(A_n)_{n\in\mathbb{N}}$ of pairwise disjoint measurable subsets of E, we have $\mu\left(\biguplus_{n\in\mathbb{N}} A_n\right) = \sum_{n\in\mathbb{N}} \mu(A_n)$. Note that infinite summations in $\overline{\mathbb{R}}_+$ are formalized as the supremum of partial sums (see Sect. 3.3).

From these fundamental axioms, we prove several other properties of measures among which monotony (i.e., $A \subseteq B \Rightarrow \mu(A) \leqslant \mu(B)$, for measurable subsets A and B), and the weakening of σ-additivity into (finite) additivity, for finite unions of pairwise disjoint subsets. For instance, the special case of the union of two disjoint subsets simplifies into

```
Lemma measure_additivity :
 ∀ (μ : measure) A B, measurable genE A → measurable genE B →
  (∀ x, A x → B x → False) → μ (fun x ⇒ A x ∨ B x) = Rbar_plus (μ A) (μ B).
```

Another interesting result is the following decomposition of the measure of a measurable subset `A : E → Prop` using a countable partition `B : nat → (E → Prop)` of the set `E`.

```
Lemma measure_decomp :
 ∀ (μ : measure) A (B : nat → E → Prop),
  measurable genE A → (∀ n, measurable genE (B n)) →
  (∀ x, ∃ n, B n x) → (∀ n p x, B n x → B p x → n = p) →
  μA = Sup_seq (fun N ⇒ sum_Rbar N (fun n ⇒ μ (fun x ⇒ A x ∧ B n x))).
```

The proof derives directly from σ-additivity. A weakened version for finite partitions is useful to establish additivity of the integral of nonnegative simple functions in Sect. 6.3.

5.2 Boole's Inequality and Continuity from Below

The σ-additivity and additivity properties described in Sect. 5.1 deal with the union of pairwise disjoint measurable subsets. When the union is not disjoint, the equality becomes an inequality, and the resulting subadditivity property is called Boole's inequality. The proof path we have followed first addresses the finite case, then establishes an important intermediate result known as continuity from below, and finally deals with the infinite case of σ-subadditivity.

Let us first consider finite subadditivity. It states that for any finite sequence $(A_n)_{n\in[0..N]}$ of measurable subsets of E, we have

$$\mu\left(\bigcup_{n\in[0..N]} A_n\right) \leqslant \sum_{n\in[0..N]} \mu(A_n).$$

The proof is performed by induction on the parameter N and uses several previously proved results, such as additivity and monotony of measures, and compatibility of measurability with finite union and intersection. A specialization for the case $N = 2$, called `measure_union`, will be handy in the sequel.

The next step is technical, it allows to transform any countable union of subsets into a pairwise disjoint union, while keeping equal the partial unions. When the input sequence $(A_n)_{n\in\mathbb{N}}$ is nondecreasing, the new sequence of pairwise disjoint subsets somehow corresponds to "nested onion peels": $B_0 \stackrel{\text{def.}}{=} A_0$, and for all $n \in \mathbb{N}$, $B_{n+1} \stackrel{\text{def.}}{=} A_{n+1} \setminus A_n$. The Coq formalization is quasiliteral.

```
Definition layers : (nat → E → Prop) → nat → E → Prop :=
  fun A n ⇒ match n with
   | O ⇒ A O
   | S n ⇒ fun x ⇒ A (S n) x ∧ ¬ A n x
   end.
```

We prove several properties of layers, such as compatibility with partial and countable union (i.e., $\biguplus_{n\in I} B_n = \bigcup_{n\in I} A_n$ with `B := layers A`, for $I = [0..N]$ and $I = \mathbb{N}$), and compatibility with measurability (i.e., the layers of a sequence of measurable subsets are measurable).

Our main application of layers and their properties is the continuity from below of measures. This results states that for any nondecreasing sequence $(A_n)_{n\in\mathbb{N}}$ of measurable subsets of E (i.e., $A_n \subseteq A_{n+1}$), we have

$$\mu\left(\bigcup_{n\in\mathbb{N}} A_n\right) \leqslant \lim_{n\to\infty} \mu(A_n).$$

Note that monotony of measures allows to replace the limit by a supremum (see Sect. 3.3). Again, the Coq formalization is straightforward.

```
Definition continuous_from_below : ((E → Prop) → Rbar) → Prop :=
  fun μ⇒ ∀ A : nat → E → Prop,
    (∀ n, measurable genE (A n)) → (∀ n x, A n x → A (S n) x) →
    μ (fun x ⇒ ∃ n, A n x) = Sup_seq (fun n ⇒ μ (A n)).

Lemma measure_continuous_from_below : ∀ (μ : measure), continuous_from_below μ.
```

The proof simply stems from finite additivity and σ-additivity of measures and from careful use of the properties of layers.

Finally, let us consider σ-subadditivity, i.e., Boole's inequality. It states that for any sequence $(A_n)_{n\in\mathbb{N}}$ of measurable subsets of E, we have

$$\mu\left(\bigcup_{n\in\mathbb{N}} A_n\right) \leqslant \sum_{n\in\mathbb{N}} \mu(A_n).$$

It is formalized using `Sup_seq`.

```
Lemma measure_Boole_ineq :
  ∀ (μ : measure) (A : nat → E → Prop), (∀ n, measurable genE (A n)) →
    Rbar_le (μ (fun x ⇒ ∃ n, A n x)) (Sup_seq (fun n ⇒ sum_Rbar n (fun m ⇒ μ (A m)))).
```

The proof is an application of continuity from below to the sequence of partial unions ($B_N \stackrel{\text{def.}}{=} \bigcup_{n\in[0..N]} A_n$). In Coq, partial unions are defined using existential quantification that makes the proof process convenient and fluid.

```
Definition partial_union : (nat → E → Prop) → nat → E → Prop :=
  fun A n x ⇒ ∃ m, (m ⩽ n)%nat ∧ A m x.
```

Then, the proof resumes by applying finite subadditivity to the nondecreasing sequence `partial_union A`, and using properties of the supremum.

5.3 Negligible Subsets

The concepts of *negligible* subset and property satisfied *almost everywhere* are of major importance in Lebesgue integration theory. They are the key ingredients to obtain the positive definiteness property (i.e., $\|u\| = 0 \Rightarrow u = 0$) of the norm in L^p Lebesgue spaces, which will be the subject of future developments.

A subset A of E is said to be negligible for the measure μ, or simply μ-negligible, when it is included in a measurable subset B of measure 0.

```
Definition negligible : (E → Prop) → Prop :=
  fun A ⇒ ∃ B, (∀ x, A x → B x) ∧ measurable genE B ∧ μ B = 0.
```

We prove several simple results about negligible subsets. For instance, measurable subsets of measure 0 are negligible, and subsets of negligible subsets are negligible. The negligibility of the countable union of negligible subsets is a bit more challenging; it is a consequence of Boole's inequality.

```
Lemma negligible_union_countable :
  ∀ (A : nat → E → Prop), (∀ n, negligible (A n)) → negligible (fun x ⇒ ∃ n, A n x).
```

This lemma is the only one where we rely on the choice property, see Sect. 2.2. The reason is as follows. Given a natural number n, as we have `negligible (A n)`, we deduce the existence of a B containing A_n that is both measurable and of measure 0. But the use of Boole's inequality and of the measurability of a countable union of sets require a *sequence* of these B. So we rely on `choice` to go from "for each n, we have a B" to a sequence of type `nat → E → Prop` with the expected properties.

A property is said to hold μ-almost everywhere (`ae`) when its complement is μ-negligible.

```
Definition ae : (E → Prop) → Prop := fun A ⇒ negligible (fun x ⇒ ¬ A x).
```

We prove some simple results about μ-almost everywhere properties. For instance, the countable intersection of properties holding μ-almost everywhere holds μ-almost everywhere. This derives from `negligible_union_countable`. An important instantiation of `ae` is the equality μ-almost everywhere, used in Sect. 7.4.

```
Definition ae_eq : (E → F) → (E → F) → Prop := fun f g ⇒ ae (fun x ⇒ f x = g x).
```

6 Simple Functions

Simple functions are real-valued functions that attain only a finite number of values. But, unlike step functions used for Riemann sums, each value may be taken here on a nonconnected subset.

This is a very simple mathematical definition, but it will require some proof engineering to have a usable formal definition. Another mathematical equivalent definition is that a simple function is a finite linear combination of indicator functions, and can be expressed as

$$f = \sum_{y \in f(E)} y \times \mathbb{1}_{f^{-1}(\{y\})}, \tag{3}$$

where $\mathbb{1}$ is the characteristic function (see Sect. 4.1). This definition is impractical in Coq as it sums over $f(E)$ that may be infinite in general. Only the property of f makes this subset finite. We choose to have a data structure that allows us to access the possible values, in order to be able to compute the integral of simple functions, and we choose to have them as a list. Indeed, the Coq List library is rather comprehensive, even if not perfectly suited for our use. We also finally choose to have simple functions of type `E → R` and not `E → Rbar`; this is discussed in Sect. 9.4.

We consider an ambient set `E` now required to be inhabited. The empty case is not of interest here, and it would mean empty lists that make the following functions fail. Instead of having additional hypotheses on the lists, it was easier not to consider empty types. Given a function and a list, the property `finite_vals` states that the values taken by the function belong to the list.

```
Definition finite_vals : (E → R) → list R → Prop := fun f l ⇒ ∀ x, In (f x) l.
```

Note that this list is far from unique: the elements may be in any order, can be taken several times, and useless values may be in the list. Hence, the need for a canonical list that is computed in Sect. 6.1, in order to integrate nonnegative simple functions, as described in Sect. 6.2. The positive linearity of the integral is shown in Sect. 6.3.

6.1 Canonical Representation

As explained above, the property `finite_vals` does not specify a unique list. To enforce uniqueness, we need a strictly sorted list, with only the useful values.

```
Definition finite_vals_canonic : (E → R) → list R → Prop :=
  fun f l ⇒ (LocallySorted Rlt l) ∧ (∀ y, In y l → ∃ x, f x = y) ∧ (∀ x, In (f x) l).
```

where `LocallySorted P` (from the Coq standard library) is the inductive definition of a sorted list using the `P` relation. Here, we require the strict order `Rlt` to prevent duplicates.

The related proofs are then threefold. First, we need to prove that only one list fits the requirement.

```
Lemma finite_vals_canonic_unique :
  ∀ f l1 l2, finite_vals_canonic f l1 → finite_vals_canonic f l2 → l1 = l2.
```

The proof is not difficult, but slightly tedious. An intermediate lemma states that if two lists have the same elements (using `In`) and are both `LocallySorted` with `Rlt`, then they are equal.

Second, to recover the fact that our simple functions are indeed a finite linear combination of indicator functions, we also prove that `finite_vals_canonic` $f\ \ell$

implies the same equality as (3), but for y in the list: $f = \sum_{y \in \ell} y \times \mathbb{1}_{f^{-1}(\{y\})}$.

```
Lemma finite_vals_sum_eq :
  ∀ (f : E → R) l, finite_vals_canonic f l →
    ∀ x, f x = sum_Rbar_map l (fun y ⇒ y ∗ (charac (fun z ⇒ f z = y) x)).
```

Last but not least, we need to be able to build this canonical list using several intermediate steps.

```
Fixpoint select {E : Type} (P : E → Prop) (l : list E) : list E :=
  match l with
  | nil ⇒ nil
  | y :: l1 ⇒ match (excluded_middle_informative (P y)) with
    | left _ ⇒ y :: select P l1
    | right _ ⇒ select P l1
    end
  end.

Definition RemoveUseless : ∀ {E F : Type}, (E → F) → list F → list F :=
  fun E F f l ⇒ select (fun y ⇒ ∃ x, f x = y) l.

Definition canonizer : (E → R) → list R → list R :=
  fun f l ⇒ sort Rle (RemoveUseless (nodup Req_EM_T l) f).
```

Let us explain these functions. The `select` function takes advantage of the `excluded_middle_informative` axiom to select the values of a list having a given property. The `RemoveUseless` function then allows us to select only the useful values of the list (such that there exists a preimage to it). The `nodup` function from the Coq standard library removes duplicates (the decidability of equality on real numbers is given). We redefined the sort function and call it with the nonstrict order `Rle`.

The canonizer function is then a successive call to `nodup`, `RemoveUseless` and `sort`. Note that these operations are actually commuting, thus any ordering would have been correct. We choose the one that eases the proofs. In particular, `sort` is the last called function as it will imply an easy proof that the final list is sorted. The function `nodup` is called first as few lemmas are available on it.

The correctness of this canonizer is then proved.

```
Lemma finite_vals_canonizer :
 ∀ f l, finite_vals f l → finite_vals_canonic f (canonizer f l).
```

For instance, consider the case of the characteristic function of some proper subset A (distinct from the empty set and from the full set, thus assuming that type `E` is neither empty, nor representing a singleton). This function actually takes the values 0 and 1 (see Sect. 4.1): it is a simple function. Starting with any list of real numbers containing 0 and 1, e.g., `l` := [1; 0; 0; 2], we may show the property `finite_vals (charac A) l`. And thus, from the previous lemma, we have `finite_vals_canonic (charac A) (canonizer (charac A) l)`, where the canonized list is simply [0; 1].

6.2 Integration of Nonnegative Simple Functions

Following the definition of simple functions, we retain those for which preimages of singletons are measurable and thus admit a measure, possibly infinite. Those measurable simple functions are collected into the set $\mathcal{SF}$, and the subset of nonnegative ones is denoted $\mathcal{SF}_+$. The needed tools for integrating in $\mathcal{SF}_+$ are sums on $\overline{\mathbb{R}}$ as defined in Sect. 3.2, and a measure μ as defined in Sect. 5. The integral of $f \in \mathcal{SF}_+$ is defined by

$$\int_{\mathcal{SF}_+} f \, \mathrm{d}\mu \stackrel{\text{def.}}{=} \sum_{y \in f(E)} y \times \mu\left(f^{-1}(\{y\})\right). \tag{4}$$

We first need to specify simple functions of $\mathcal{SF}$ that have measurable preimages.

```
Definition SF_aux : ((E → Prop) → Prop) → (E → R) → list R → Prop :=
 fun genE f lf ⇒ finite_vals_canonic f lf ∧ (∀ y, measurable genE (fun x ⇒ f x = y)).

Definition SF : ((E → Prop) → Prop) → (E → R) → Set :=
 fun genE f ⇒ {lf | SF_aux genE f lf}.
```

Note that the list is in `Set` as we need to get a hand on it to compute the integral. A weak existential is not strong enough for our purpose. In Coq, the notation `{x | P x}` means *there exists* `x` *such that* `P x`, but the type is `Set`. This means it is a strong existential, i.e., it is possible to pick up the witness `x`.

Note also that since singletons are Borel subsets of $\mathbb{R}$ (as closed subsets), we are able to prove measurability of functions in $\mathcal{SF}$.

```
Lemma SF_aux_measurable_fun: ∀ genE f l, SF_aux genE f l → measurable_fun_R f.
```

Then, the definition of the integral in $\mathcal{SF}_+$ is straightforward from a proof of type `SF`.

```
Definition af1 : (E → R) → Rbar → Rbar :=
 fun f y ⇒ Rbar_mult y (μ (fun x ⇒ Finite (f x) = y))).

Definition LInt_SFp : ∀ genE, ∀ (f : E → R), SF genE f → Rbar :=
 fun f Hf ⇒ let lf := proj1_sig Hf in sum_Rbar_map lf (af1 f).
```

Note the required hypothesis `Hf` that encompasses both the proof that f is a valid simple function, and the list witness ℓ on which the definition depends. Then, `proj1_sig` returns the first part of this proof, that is the list `lf`, in order to sum on it. This dependent type is only inside the library and is not to be used outside: final users will make use only of a total function for the Lebesgue integral. This limited use of dependent types has not proved inconvenient.

We first prove that the value of the integral does not depend on the chosen list/proof (`Hf1` and `Hf2` in the next lemma).

```
Lemma LInt_SFp_correct :
  ∀ f (Hf1 Hf2 : SF genE f), non_neg f → LInt_SFp f Hf1 = LInt_SFp f Hf2.
```

A first easy example of integration is the relationship between this integral and the characteristic function. The characteristic function has two possible values (0 and 1), so it is a simple function. When the subset A is measurable, $\mathbb{1}_A$ belongs to $\mathcal{SF}_+$, and its integral is, as expected, the measure of A.

```
Lemma LInt_SFp_charac :
  ∀ A (HA : measurable genE A), LInt_SFp (charac A) (SF_charac A HA) = μ A.
```

6.3 Linearity of the Integral of Simple Functions

Then comes a proof that is unexpectedly complex, the additivity of the integral in $\mathcal{SF}_+$: $\int_{\mathcal{SF}_+}(f+g)\,\mathrm{d}\mu = \int_{\mathcal{SF}_+} f\,\mathrm{d}\mu + \int_{\mathcal{SF}_+} g\,\mathrm{d}\mu$.

Alternate proofs are available, e.g., see [37,62], but were not considered in this work. We choose a proof using a change of variable, from the sum of values taken by each function f and g to the values taken by their sum $f+g$. The main difficulty is that the canonical list ℓ_{f+g} associated with $f+g$ has nothing to do with any kind of "addition" of the lists ℓ_f and ℓ_g associated with f and g.

The first stage is a lemma stating that $\mathcal{SF}$ is closed under addition.

```
Lemma SF_plus : ∀ f (Hf : SF genE f) g (Hg : SF genE g), SF genE (fun x ⇒ f x + g x).
```

For that, we rely on

```
Definition cartesian_Rplus : list R → list R → list R :=
  fun l1 l2 ⇒ flat_map (fun a1 ⇒ (map (fun a2 ⇒ a1 + a2) l2)) l1.
```

that gathers all possible sums from two lists. When applied to ℓ_f and ℓ_g, the result may be too large a list, but no useful value is missing. Thus, we may strip unwanted values by applying the previously defined canonizer(see Sect. 6.1).

The second stage is a couple of lemmas coming from the fact that the subsets $f^{-1}(\{y\})$ for $y \in f(E)$ constitute a partition of the domain E of the function f. First, a specialization of the finite version of the lemma `measure_decomp` (see Sect. 5.1) for preimages by functions f and g provides

```
Lemma SFp_decomp :
  ∀ f g lf lg y, SF_aux genE f lf → SF_aux genE g lg →
    μ(fun x ⇒ f x = y) = sum_Rbar_map lg (fun z ⇒ μ (fun x ⇒ f x = y ∧ g x = z)).
```

Note that this result is first proved with the assumption that y is actually a value taken by f. But this premise can be dropped as for all other values of y, the equality to show simplifies into the trivial $0 = 0$.

Then, the result is lifted to the integral level,

$$\sum_{y\in f(E)} y \times \mu\left(f^{-1}(\{y\})\right) = \sum_{y\in f(E)} y \left(\sum_{z\in g(E)} \mu\left(f^{-1}(\{y\}) \cap g^{-1}(\{z\})\right)\right),$$

which is formalized as

```
Lemma LInt_SFp_decomp :
 ∀ f g lf lg, SF_aux genE f lf → SF_aux genE g lg →
   (* LInt_SFp f H = *) sum_Rbar_map lf (af1 f) = sum_Rbar_map lf
    (fun y ⇒ Rbar_mult y (sum_Rbar_map lg (fun z ⇒ μ (fun x ⇒ f x = y ∧ g x = z)))).
```

The third stage consists in applying the latter lemma to justify the rewritings in the equations below. First, for both integrals in the sum.

$$\begin{aligned}\int_{\mathcal{SF}_+} f\,\mathrm{d}\mu + \int_{\mathcal{SF}_+} g\,\mathrm{d}\mu &= \sum_{y\in f(E)} y\,\mu\left(f^{-1}(\{y\})\right) + \sum_{z\in g(E)} z\,\mu\left(g^{-1}(\{z\})\right)\\ &= \sum_{y\in f(E)}\sum_{z\in g(E)} y\,\mu\left(f^{-1}(\{y\})\cap g^{-1}(\{z\})\right)\\ &\quad + \sum_{z\in g(E)}\sum_{y\in f(E)} z\,\mu\left(g^{-1}(\{z\})\cap f^{-1}(\{y\})\right)\\ &= \sum_{y\in f(E)}\sum_{z\in g(E)} (y+z)\,\mu\left(f^{-1}(\{y\})\cap g^{-1}(\{z\})\right).\end{aligned}$$

And then, for the integral of the sum, where the lemma is applied with $\mathtt{f} = f+g$ and $\mathtt{g} = f$.

$$\begin{aligned}\int_{\mathcal{SF}_+} (f+g)\,\mathrm{d}\mu &= \sum_{t\in(f+g)(E)} t\,\mu\left((f+g)^{-1}(\{t\})\right)\\ &= \sum_{t\in(f+g)(E)} t \sum_{y\in f(E)} \mu\left((f+g)^{-1}(\{t\})\cap f^{-1}(\{y\})\right)\\ &= \sum_{y\in f(E)}\sum_{t\in(f+g)(E)} t\,\mu\left(f^{-1}(\{y\})\cap (f+g)^{-1}(\{t\})\right).\end{aligned}$$

Finally, the last step of the additivity proof is to connect both sets of equalities by establishing the following "horrible" change of variable formula

$$\begin{aligned}&\sum_{z\in g(E)} (y+z)\,\mu\left(f^{-1}(\{y\})\cap g^{-1}(\{z\})\right)\\ &\quad= \sum_{t\in(f+g)(E)} t\,\mu\left(f^{-1}(\{y\})\cap (f+g)^{-1}(\{t\})\right),\end{aligned}$$

that is formalized as

```
Lemma sum_Rbar_map_change_of_variable :
 ∀ f g lf lg y, SF_aux genE f lf → SF_aux genE g lg →
   let lfpg := canonizer (fun x ⇒ f x + g x) (cartesian_Rplus lf lg) in
   sum_Rbar_map lg (fun z ⇒ Rbar_mult (y + z) (μ (fun x ⇒ f x = y ∧ g x = z))) =
    sum_Rbar_map lfpg
     (fun t (* = y + z *) ⇒ Rbar_mult t (μ (fun x ⇒ f x = y ∧ f x + g x = t))).
```

The key ingredient here is that sums may be restricted to only their nonzero terms, which makes the change of variable $z \mapsto t = y + z$ (for fixed y) a bijection.

An interesting point is that this lemma is hardly explicit in mathematical textbooks and we had to puzzle it out to fulfill the proof. We had to write it explicitly as it was a key point in our design choice for simple functions, see Sect. 9.4.

Ultimately, we end up with similar double summations, and we are able to prove the additivity of the integral in $\mathcal{SF}_+$.

```
Lemma LInt_SFp_plus :
  ∀ f (Hf : SF genE f) g (Hg : SF genE g), non_neg f → non_neg g →
   let Hfpg := SF_plus f Hf g Hg in
   LInt_SFp (fun x ⇒ f x + g x) Hfpg = Rbar_plus (LInt_SFp f Hf) (LInt_SFp g Hg).
```

As a break, we establish the compatibility of the integral in $\mathcal{SF}_+$ with nonnegative scaling.

```
Lemma LInt_SFp_scal :
 ∀ f (Hf : SF genE f) a, non_neg f → 0 ⩽ a →
   let Haf := SF_scal f Hf in
   LInt_SFp (fun x ⇒ a ∗ f x) Haf = Rbar_mult a (LInt_SFp f Hf).
```

This calls for a proof `Haf` that a simple function multiplied by a scalar is indeed a simple function. Then, we only need to require that both the scalar and the function are nonnegative to ensure that $\int_{\mathcal{SF}_+} a \times f \, \mathrm{d}\mu = a \times \int_{\mathcal{SF}_+} f \, \mathrm{d}\mu$. Then, monotony of the integral in $\mathcal{SF}_+$ is a direct consequence of additivity, since the relation $g = f + (g - f)$ holds in $\mathcal{SF}_+$ when $f \leqslant g$.

7 Integration of Nonnegative Functions

Let us now consider functions of type `E → Rbar` that may take an infinite number of (possibly infinite) values. The set of measurable functions to $\overline{\mathbb{R}}$ is denoted by $\mathcal{M}$, and its subset of nonnegative functions by $\mathcal{M}_+$. The key idea for the definition of the integral in $\mathcal{M}_+$ is to use approximations from below by simple functions in $\mathcal{SF}_+$, and surprisingly, we benefit from the use of computer arithmetic.

The integral is presented in Sect. 7.1 together with some preliminary properties. Then, Sect. 7.2 is devoted to the crucial Beppo Levi (monotone convergence) theorem. Adapted sequences are defined in Sect. 7.3. Linearity and other properties of the integral are displayed in Sect. 7.4. Finally, Sect. 7.5 is devoted to Fatou's lemma, the other major result on the integral for nonnegative functions.

7.1 Definition and First Properties

We now want to define the Lebesgue integral for nonnegative integrable functions. The mathematical definition is

$$\forall f \in \mathcal{M}_+, \quad \int_{\mathcal{M}_+} f \, \mathrm{d}\mu \overset{\text{def.}}{=} \sup_{\substack{\psi \in \mathcal{SF}_+ \\ \psi \leqslant f}} \int_{\mathcal{SF}_+} \psi \, \mathrm{d}\mu$$

where the supremum is taken for nonnegative measurable simple functions ψ less than or equal to f pointwise, and where the integral in $\mathcal{SF}_+$ is defined in Sect. 6.2.

Keeping on with total functions, we prescribe a value whatever the input function f, with a Coq definition quite similar to the mathematical one.

```
Definition LInt_p : (E → Rbar) → Rbar :=
 fun f ⇒ Rbar_lub (fun z : Rbar ⇒ ∃ (ψ : E → R) (Hψ : SF genE ψ),
   non_neg ψ ∧ (∀ x, Rbar_le (ψ x) (f x)) ∧ LInt_SFp μ ψ Hψ = z).
```

The supremum is taken on a subset of extended reals z such that there exists a simple function ψ less than or equal to f having z for integral.

The first thing to prove is that this newly defined integral is the same as the already-defined integral when the function is a simple function, i.e., $\int_{\mathcal{M}_+} f \, \mathrm{d}\mu$ is equal to $\int_{\mathcal{SF}_+} f \, \mathrm{d}\mu$ for all f in $\mathcal{SF}_+$, and in Coq

```
Lemma LInt_p_SFp_eq :
  ∀ (f : E → R) (Hf : SF genE f), non_neg f → LInt_p f = LInt_SFp μ f Hf.
```

Then comes the monotony of the integral.

$$\forall f, g \in \mathcal{M}_+, \quad f \leqslant g \quad \Longrightarrow \quad \int_{\mathcal{M}_+} f \, \mathrm{d}\mu \leqslant \int_{\mathcal{M}_+} g \, \mathrm{d}\mu.$$

The Coq translation becomes

```
Lemma LInt_p_monotone :
  ∀ (f g : E → Rbar), (∀ x, Rbar_le (f x) (g x)) → Rbar_le (LInt_p f) (LInt_p g).
```

Indeed, the least upper bound (LUB) in the definition of the total function is enough to ensure monotony for any functions f and g, not only for the nonnegative and measurable ones as in the mathematical statement.

Another easy result is about the multiplication by a nonnegative scalar value.

```
Lemma LInt_p_scal_finite :
  ∀ (f : E → Rbar) (a : R), 0 ⩽ a →
    LInt_p (fun x ⇒ Rbar_mult a (f x)) = Rbar_mult a (LInt_p f).
```

As before, there is no assumption on the fact that f is nonnegative.

The following extensionality result instantiated for restricted functions has proved useful. It states that when functions are equal on a measurable subset, then the integral of their restriction to that subset are equal. This is hardly mentioned in mathematics. As before, there is no requirement on the properties of A, f and g. The total function `LInt_p` gives something that is the same in both cases, even for nonmeasurable functions.

```
Lemma LInt_p_when_charac :
  ∀ f g (A : E → Prop), (∀ x, A x → f x = g x) →
    LInt_p (fun x ⇒ Rbar_mult (f x) (charac A x)) =
      LInt_p (fun x ⇒ Rbar_mult (g x) (charac A x)).
```

7.2 The Beppo Levi Theorem

The Beppo Levi theorem (see Textbook Theorem 1, page 3), also known as the monotone convergence theorem, is one of the most fundamental results in measure and integration theories. It states that for any sequence $(f_n)_{n\in\mathbb{N}}$ of pointwise nondecreasing and nonnegative measurable functions (i.e., in $\mathcal{M}_+$), the pointwise limit $\lim_{n\to\infty} f_n$ (which actually equals $\sup_{n\in\mathbb{N}} f_n$, see Sect. 3.3) is also in $\mathcal{M}_+$. This property is proved using standard properties of measurable functions such as monotony and is not particularly challenging. The Beppo Levi theorem also states the following integral-limit exchange formula

$$\int_{\mathcal{M}_+} \sup_{n\in\mathbb{N}} f_n \, \mathrm{d}\mu = \sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} f_n \, \mathrm{d}\mu$$

which is stated in Coq as

```
Lemma Beppo_Levi :
  ∀ (f : nat → E → Rbar), (∀ x n, Rbar_le (f n x) (f (S n) x)) →
    (∀ n, non_neg (f n)) → (∀ n, measurable_fun_Rbar genE (f n)) →
    LInt_p μ (fun x ⇒ Sup_seq (fun n ⇒ f n x)) = Sup_seq (fun n ⇒ LInt_p μ (f n)).
```

The proof of this equality is technical and relies on a wide variety of previously proved results. It can be divided into two inequalities. The easy one, $\sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} f_n \,\mathrm{d}\mu \leqslant \int_{\mathcal{M}_+} \sup_{n\in\mathbb{N}} f_n \,\mathrm{d}\mu$, is proved using monotony of the integral, and fundamental properties of the supremum. The other one,

$$\int_{\mathcal{M}_+} \sup_{n\in\mathbb{N}} f_n \,\mathrm{d}\mu \leqslant \sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} f_n \,\mathrm{d}\mu, \tag{5}$$

is more intricate. The first step of the proof is to show that for any $\psi \in \mathcal{SF}_+$ less than or equal to $\sup_{n\in\mathbb{N}} f_n$, and any number $0 < a < 1$, we have the bound

$$\int_{\mathcal{M}_+} \psi \,\mathrm{d}\mu \leqslant \frac{1}{a} \sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} f_n \,\mathrm{d}\mu. \tag{6}$$

For that purpose, the subsets $A_n = \{x \in E \mid a\psi \leqslant f_n\}$ are first shown to be nondecreasing and measurable, the latter coming from the measurability of functions $a\psi - f_n$. Then, they are shown to cover the full set E, which is stated in Coq as `∀ x, ∃ n, A n x`, and the existential is exhibited as a rank N above which we have $a\psi(x) \leqslant f_n(x)$. Then, the proof of Eq. (6) relies on continuity from below of the measure (see Sect. 5.2), measurability of simple functions (see Sect. 6.2), linearity properties of the integral in $\mathcal{SF}_+$ (see Sect. 6.3), and monotony and compatibility with characteristic functions for the integral in $\mathcal{M}_+$ (see Sect. 7.1).

Finally, the inequality (5) is obtained by taking in (6) the limit as a goes up to 1, and from the definition of the integral in $\mathcal{M}_+$ (see Sect. 7.1) and properties of the supremum.

This concludes the proof of the Beppo Levi theorem.

7.3 Adapted Sequences

As for simple functions, a real difficulty is the additivity property of the integral in $\mathcal{M}_+$: $\int_{\mathcal{M}_+}(f+g)\,\mathrm{d}\mu = \int_{\mathcal{M}_+} f \,\mathrm{d}\mu + \int_{\mathcal{M}_+} g \,\mathrm{d}\mu$. Given the definition of the Lebesgue integral (see Sect. 7.1), the common proof relies on adapted sequences in $\mathcal{SF}_+$.

An adapted sequence for a function f is a pointwise nondecreasing sequence of nonnegative functions that is pointwise converging from below toward f.

```
Definition is_adapted_seq : (E → Rbar) → (nat → E → R) → Prop :=
  fun f ψ ⇒ (∀ n, non_neg (ψ n)) ∧ (∀ x n, ψ n x ⩽ ψ (S n) x) ∧
    (∀ x, is_sup_seq (fun n ⇒ ψ n x) (f x)).
```

In our case, the adapted sequences of interest are the measurable simple functions of $\mathcal{SF}$. We then deduce that the sequence of integrals of such a sequence converges toward the integral of f from below.

```
Definition SF_seq : (nat → E → R) → Set := fun ψ ⇒ ∀ n, SF genE (ψ n).

Lemma LInt_p_with_adapted_seq :
  ∀ f ψ, is_adapted_seq f ψ → SF_seq genE ψ →
    Sup_seq (fun n ⇒ LInt_p μ (ψ n)) = LInt_p μ f.
```

Having this definition and its link to the integral is far from enough. We need to have an adapted sequence corresponding to any function in $\mathcal{M}_+$. This building is quite easy in mathematics: for f nonnegative, the chosen adapted sequence is

$$\varphi_n(x) \stackrel{\text{def.}}{=} \begin{cases} \dfrac{\lfloor 2^n f(x) \rfloor}{2^n} & \text{when } f(x) < n, \\ n & \text{otherwise.} \end{cases} \tag{7}$$

We began by translating literally this definition. Then, we tried to prove that the sequence is nondecreasing, and so on. One of the authors then noticed that such proofs were already done and available in the library Flocq [10] dedicated to computer arithmetic. Flocq is a formalization of floating-point arithmetic that provides a comprehensive library of theorems on a multi-radix multi-precision arithmetic. It also supports efficient numerical computations inside Coq. As it aims at genericity, even computer arithmetic formats are abstract to encompass floating-point and fixed-point arithmetics and many proved results also hold in fixed-point arithmetic. Seen from a computer science point of view, the definition of φ_n in Eq. (7) indeed relies on a fixed-point rounding downward with a least significant bit (lsb) of $-n$. It is formalized in Coq as

```
Definition mk_adapted_seq (* = φn *) : nat → E → R :=
  fun n x ⇒ match (Rbar_le_lt_dec (INR n) (f x)) with
    | left _ ⇒ INR n
    | right _ ⇒ round radix2 (FIX_exp (−Z.of_nat n)) Zfloor (f x)
    end.
```

Many proofs related to inequalities (such as $(\varphi_n)_{n\in\mathbb{N}}$ is bounded and nondecreasing) are really smooth, relying on the support library of Flocq. For instance, the theorem $\varphi_n(x) \leq f(x)$ is a split whether $f(x) < n$, followed by a call to a property of the rounding downward.

A more involved example lies in the proof of $\varphi_n(x) \leq \varphi_{n+1}(x)$. We first split depending whether $f(x) < n$ or $n+1$, and handle three simple cases. The most difficult case should be to prove that

$$\frac{\lfloor 2^n f(x) \rfloor}{2^n} \leq \frac{\lfloor 2^{n+1} f(x) \rfloor}{2^{n+1}}$$

when $f(x) < n$. Indeed, a direct proof does not seem so straightforward. Taking the computer arithmetic point of view, it becomes $\triangledown_n(x) \leq \triangledown_{n+1}(x)$, with $\triangledown_n$ the rounding down in fixed-point arithmetic with least significant bit (lsb) n. Then, we rely on the following floating-point theorem: if $u \leq v$ and $u \in \mathbb{F}$, then $u \leq \circ(v)$ for any reasonable format $\mathbb{F}$ and rounding $\circ$. Applying it, there is left to prove both that $\triangledown_n(x) \leq x$ (trivial by the properties of the rounding down) and that $\triangledown_n(x)$ fits in the fixed-point format with least significant bit $-n-1$ (easy as it has a lsb of $-n$). The proof of this subcase has been done in 10 lines of standard Coq.

Even the convergence was eased by existing Flocq error lemmas. The main result states that the φ_n are indeed an adapted sequence.

```
Lemma mk_adapted_seq_is_adapted_seq : is_adapted_seq f mk_adapted_seq.
```

The part left to prove is that the φ_n are in $\mathcal{SF}$. The first thing to prove is that the preimages of φ_n are measurable subsets.

```
Lemma mk_adapted_seq_SF_aux : ∀ n y, measurable genE (fun x ⇒ mk_adapted_seq n x = y).
```

There are several proof paths. The chosen one is to prove that the subset is either empty (so measurable), or $y \leq n$. We also prove that y is a fixed-point number with lsb n. Then, we have

$$\varphi_n(x) = y \iff y \leq f(x) < \mathrm{succ}(y)$$

where succ is the successor in fixed-point arithmetic, meaning the next number in the format with lsb n. This subset is measurable as f is measurable, and from the properties of Borel subsets of Sect. 4.4. There are some special cases related to ∞ and to $y = n$ that are tedious but easy.

Last is to prove that the φ_n only take a finite number of values, making them valid simple functions.

```
Lemma mk_adapted_seq_SF : SF_seq genE mk_adapted_seq.
```

The mathematical definition is clear, but this is the first proof of this kind (the only previously proved simple function is the characteristic function). So we have to build by hand the list of all possible values of φ_n: first the list of integers i with $0 \le i \le n$, and then the list of all the $i/2^n$ for $0 \le i \le n2^n$. This kind of proof could clearly be automated, or simplified by dedicated lemmas if need be. From this very generic list, we only have to apply the canonizer defined in Sect. 6.1 to f. Then, we end up the proof, relying on the various properties of the canonizer, the fixed-point rounding, and the measurability above.

To conclude, we have defined explicitly an adapted sequence that we may give to the theorem `LInt_p_with_adapted_seq`, thus providing an explicit formula for the integral in $\mathcal{M}_+$: $\int_{\mathcal{M}_+} f \, \mathrm{d}\mu = \sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} \varphi_n \, \mathrm{d}\mu$.

```
Lemma LInt_p_with_mk_adapted_seq :
 ∀ f, non_neg f → measurable_fun_Rbar genE f →
   Sup_seq (fun n ⇒ LInt_p μ (mk_adapted_seq f n)) = LInt_p μ f.
```

7.4 Linearity and Other Properties of the Integral

We present now some theorems about the integration of nonnegative measurable functions that we consider essential for a library user. They are all consequences of the Beppo Levi (monotone convergence) theorem (see Sect. 7.2). They are gathered in Table 1.

Coq statements	Mathematical statements

`Lemma LInt_p_plus :`
`∀ f g,`
`non_neg f → measurable_fun_Rbar genE f →`
`non_neg g → measurable_fun_Rbar genE g →`
`LInt_p μ (fun x ⇒ Rbar_plus (f x) (g x)) =`
`Rbar_plus (LInt_p μ f) (LInt_p μ g).`
	$\forall f, g \in \mathcal{M}_+$,
	$\int_{\mathcal{M}_+} (f+g)\,\mathrm{d}\mu = \int_{\mathcal{M}_+} f\,\mathrm{d}\mu + \int_{\mathcal{M}_+} g\,\mathrm{d}\mu$.

`Lemma LInt_p_scal :`
`∀ a f, Rbar_le 0 a →`
`non_neg f → measurable_fun_Rbar genE f →`
`LInt_p μ (fun x ⇒ Rbar_mult a (f x)) =`
`Rbar_mult a (LInt_p μ f).`
	$\forall a \in \overline{\mathbb{R}}_+$, $\forall f \in \mathcal{M}_+$,
	$\int_{\mathcal{M}_+} (a \times f)\,\mathrm{d}\mu = a \times \int_{\mathcal{M}_+} f\,\mathrm{d}\mu$.

`Lemma LInt_p_ae_definite :`
`∀ f,`
`non_neg f → measurable_fun_Rbar genE f →`
`(ae_eq μ f (fun _ ⇒ 0) ↔ LInt_p μ f = 0).`
	$\forall f \in \mathcal{M}_+$,
	$f \overset{\mu\,\text{a.e.}}{=} 0 \Leftrightarrow \int_{\mathcal{M}_+} f\,\mathrm{d}\mu = 0$.

`Lemma LInt_p_decomp :`
`∀ f A, measurable genE A →`
`non_neg f → measurable_fun_Rbar genE f →`
`LInt_p μ f = Rbar_plus`
`(LInt_p μ (fun x ⇒ Rbar_mult (f x)`
`(charac A x)))`
`(LInt_p μ (fun x ⇒ Rbar_mult (f x)`
`(charac (fun y ⇒ ¬ A y) x))).`
	$\forall f \in \mathcal{M}_+$, $\forall A$ measurable,
	$\int_{\mathcal{M}_+} f\,\mathrm{d}\mu = \int_{\mathcal{M}_+} (f \times \mathbb{1}_A)\,\mathrm{d}\mu + \int_{\mathcal{M}_+} (f \times \mathbb{1}_{A^c})\,\mathrm{d}\mu$.

`Lemma LInt_p_ae_eq_compat:`
`∀ f g,`
`non_neg f → measurable_fun_Rbar genE f →`
`non_neg g → measurable_fun_Rbar genE g →`
`ae_eq μ f g →`
`LInt_p μ f = LInt_p μ g.`
	$\forall f, g \in \mathcal{M}_+,$
	$f \overset{\mu\,\text{a.e.}}{=} g \Rightarrow$
	$\int_{\mathcal{M}_+} f \,\mathrm{d}\mu = \int_{\mathcal{M}_+} g \,\mathrm{d}\mu.$

Table 1 : Linearity and other properties of the integral in $\mathcal{M}_+$.

The first entry (`LInt_p_plus`) is for additivity of the integral in $\mathcal{M}_+$. The proof is rather smooth, now that additivity of the integral in $\mathcal{SF}_+$ (see Sect. 6.3), the monotone convergence (see Sect. 7.2), and existence of an adapted sequence (see Sect. 7.3) are established. The second entry (`LInt_p_cal`) generalizes the nonnegative scaling property (see Sect. 7.1) to the infinite case too. Both will be significant ingredients of the linearity of the Lebesgue integral for arbitrary sign functions, to build the structure of vector space of the integrable functions (out of the scope of this paper).

The remaining entries actually follow from the first two linearity results. The characterization of zero-integral functions (`LInt_p_ae_definite`) relies on the scaling property and the compatibility result with characteristic functions (see Sect. 7.1), while the decomposition on a partition (`LInt_p_decomp`) relies on the additivity property. Finally, the compatibility result with almost equality (`LInt_p_ae_eq_compat`) relies on the last two. Note that this latter possesses a companion lemma about inequalities, and both share the same proof through the abstraction of their binary relation.

7.5 Fatou's Lemma

Fatou's lemma (see Textbook Theorem 2) is the other fundamental result in Lebesgue integration theory for nonnegative functions. It specifies how the situation deteriorates when the sequence in the Beppo Levi theorem is no longer monotone: the equality becomes an inequality and limits (i.e., suprema) become limits inferior. It states that for any sequence $(f_n)_{n\in\mathbb{N}}$ of nonnegative measurable functions (i.e., in $\mathcal{M}_+$), the pointwise limit $\liminf_{n\to\infty} f_n$ is also in $\mathcal{M}_+$, and the integral of the limit inferior is less than or equal to the limit inferior of the integrals,

$$\int_{\mathcal{M}_+} \liminf_{n\to\infty} f_n \,\mathrm{d}\mu \leqslant \liminf_{n\to\infty} \int_{\mathcal{M}_+} f_n \,\mathrm{d}\mu,$$

which is stated in Coq as

```
Theorem Fatou_lemma :
 ∀ (f : nat → E → Rbar), (∀ n, non_neg (f n)) →
 (∀ n, measurable_fun_Rbar genE (f n)) →
  Rbar_le (LInt_p μ (fun x ⇒ LimInf_seq' (fun n ⇒ f n x)))
        (LimInf_seq' (fun n ⇒ LInt_p μ (f n))).
```

The proof is rather short. The principle is to apply the Beppo Levi theorem (see Sect. 7.2) to the sequence $(\inf_{m\in\mathbb{N}} f_{n+m})_{n\in\mathbb{N}}$ which is shown nondecreasing, nonnegative, and measurable. We get

$$\int_{\mathcal{M}_+} \liminf_{n\to\infty} f_n \,\mathrm{d}\mu = \sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} \inf_{m\in\mathbb{N}} f_{n+m} \mathrm{d}\mu.$$

Then, by the monotony result of Sect. 7.1 and the definitions of infimum and limit inferior, we have

$$\sup_{n\in\mathbb{N}} \int_{\mathcal{M}_+} \inf_{m\in\mathbb{N}} f_{n+m} \mathrm{d}\mu \leqslant \liminf_{n\to\infty} \int_{\mathcal{M}_+} f_n \,\mathrm{d}\mu,$$

proving the result.

Note that a less common proof path is also possible: establish first Fatou's lemma, and then the Beppo Levi theorem.

Fatou's lemma is essential to establish other fundamental results such as the Fatou–Lebesgue theorem that collects a chain of inequalities involving both inferior and superior limits, and above all, Lebesgue's dominated convergence theorem whose result is similar to that of the Beppo Levi theorem, but only through the dominance of the sequence (both left as future work).

8 A Simple Case Study: The Dirac Measure

It makes sense to exhibit an example of measure to test the specifications described in the previous sections and especially the axiomatic definition of Sect. 5.

For instance, the Lebesgue measure, which extends the notion of length of intervals in $\mathbb{R}$, is ubiquitous on Euclidean spaces $\mathbb{R}^n$. And the counting measure, which returns the cardinal, is pertinent on countable sets. But both present formalization issues, and their study is left for future works.

We present the construction and usage of a very simple measure, the Dirac measure. It is used, for instance, in physics to model a point mass.

The Dirac map associated with an element $a \in E$ is the function δ_a mapping any subset $A \subseteq E$ to $\mathbb{1}_A(a)$ (see Sect. 4.1).

```
Definition Dirac : E → (E → Prop) → Rbar := fun a A ⇒ charac A a.
```

As a measure, the total function defined above only makes sense for measurable subsets. But the Dirac measure has the salient property to be a measure for any σ-algebra on `E`, even for the discrete σ-algebra (see Sect. 4.2).

To instantiate the Dirac map δ_a as a measure, we prove that it meets the specification of measures of Sect. 5.1. The first two properties, homogeneity (`Dirac_False`) and non-negativeness (`Dirac_ge_0`), are obvious. The proof of σ-additivity is based on the following argument, that is valid for any pairwise disjoint sequence of subsets $(A_n)_{n\in\mathbb{N}}$ (there is at most one nonzero term in the sum).

$$\delta_a\left(\biguplus_{n\in\mathbb{N}} A_n\right) = \mathbb{1}_{\biguplus_{n\in\mathbb{N}} A_n}(a) = \sum_{n\in\mathbb{N}} \mathbb{1}_{A_n}(a) = \sum_{n\in\mathbb{N}} \delta_a(A_n).$$

Then, the Dirac measure can be built for any generator `genE`.

```
Definition Dirac_measure : E → measure genE :=
  fun a ⇒ mk_measure genE (Dirac a)
    (Dirac_False a) (Dirac_ge_0 a) (Dirac_sigma_additivity a).
```

The integral of any function `f : E → Rbar` with the Dirac measure is

$$\int f \,\mathrm{d}\delta_a = f(a).$$

The formalization in $\mathcal{SF}_+$ for nonnegative measurable simple functions, and any generator `genE`, is (see Sect. 6.2)

```
Lemma LInt_SFp_Dirac :
 ∀ (f : E → R) (Hf : SF genE f) a, LInt_SFp (Dirac_measure genE a) f Hf = f a.
```

The proof is a direct application of lemma `finite_vals_sum_eq` of Sect. 6.1. The version in $\mathcal{M}_+$ for nonnegative measurable functions, and any generator `genE`, is (see Sect. 7.1)

```
Lemma LInt_p_Dirac :
 ∀ (f : E → Rbar) a, non_neg f → measurable_fun_Rbar genE f →
   LInt_p E genE (Dirac_measure genE a) f = f a.
```

Its proof is an application of lemma `LInt_p_with_mk_adapted_seq` of Sect. 7.3.

9 Discussion on Proof-Engineering Concerns

During our development, we had to make several choices regarding logic and the formalization of mathematics.

9.1 Extended Real Numbers

Measure theory and integration of nonnegative functions that are investigated here only manipulate values in $\overline{\mathbb{R}}_+$. Thus arises the question of the most practical Coq formalization for nonnegative extended real numbers among the following three possible choices: either $[0, \infty]$, $\mathbb{R} \cup \{\infty\}$, or $\mathbb{R} \cup \{-\infty, \infty\}$.

From a mathematical point of view, all solutions are acceptable because, in addition, we either prove or require that values are nonnegative. But we also need to keep in mind that eventually we will have to deal with arbitrary sign functions. And despite the absence of algebraic structure, extended real numbers with both infinities are the usual framework often used by mathematicians to allow for simplified expressions of many statements. We have chosen to follow this practice and to use `Rbar` from Coquelicot (see Sect. 2.1). But let us review the other possibilities we considered.

First solution: $[0, \infty]$. For example with a specific type `Rbarplus`. This would be very difficult to use in Coq because it would not be directly related to the type `R`, so we would need a coercion or some subtyping to use this type in formulas with reals. Moreover, it would make ∞ appear explicitly a lot. This would lead to very verbose statements with few automation possibilities.

Second solution: $\mathbb{R} \cup \{\infty\}$. For example with a type with two constructors `R` and `p_infty`. This would keep validity of usual algebraic properties such as associativity and distributivity without the need for additional hypotheses, and would favor a low number of cases in proofs. But it would still be a new type that would lead to Coq coercions. Moreover, when $-\infty$ will enter the picture for functions with arbitrary sign, it would make necessary coercions from/to the three types, which would be as difficult to handle as coercions from/to $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}$.

Chosen (third) solution: $\mathbb{R} \cup \{-\infty, \infty\}$. Which is the type `Rbar`. This has the advantage of being already defined in Coquelicot with several lemmas proved, and is related to the type `R`. Of course, for the present developments on nonnegative functions, we have to deal with meaningless negative cases and additional hypotheses. However, this drawback is balanced by the fact that we are ready to treat arbitrary sign functions.

9.2 Classical Logic Aspects

In our previous work, we tried to minimize the use of classical aspects. For example, in the formal proof of the Lax–Milgram theorem [15], we had a few decidability hypotheses and some statements relied on double negations to avoid using a stronger classical property. We consider here that it is no longer worth the effort compared to the theoretical gain. For this reason, we have decided to use the classical theorems listed in Sect. 2.2.

Moreover, as explained in Sect. 4.1, we choose that subsets have the type `E → Prop`. We could have chosen `E → bool` and it would have simplified some proofs as we rely a lot on the excluded middle axiom for deciding whether a point is in a subset or not. Nevertheless, this use of `bool` would not have fully removed the excluded middle axiom. Indeed, when selecting in a list the elements that have a property (function `select` of Sect. 6.1), we need to decide inside this function whether a property holds or not. And this is then applied to the following property: `fun x ⇒ ∃ y, f y = x`. We need, for each $x \in E$ to decide whether it belongs to the image of f and that requires the informative excluded middle axiom. So, as it has no logical impact, we have chosen `Prop` for convenience as it fits better in the libraries we rely on.

9.3 Measurability of Subsets

Implementing the concept of (generated) σ-algebra, i.e., the measurability of subsets, as an inductive type allows to conduct proofs by induction. This is proposed in Isabelle/HOL[1] and Lean,[2] and it is useful in Coq too (see Sect. 4.2). But this design choice can also have an impact on how mathematics results are stated. Of course, there is a constructor for each basic property of σ-algebras. But it is also necessary to add a constructor, `measurable_gen`, that embodies the belonging to the collection of generators. Indeed, this is required to initiate the constructive process of specifying a measurable subset. In other words, our Coq definition also encompasses the mathematical concept of *generated* σ-algebra.

As a consequence, we cannot instantiate a σ-algebra without exhibiting a generator. But fortunately, nothing prevents from setting the whole σ-algebra as the generator, and specify, or prove, that it satisfies `is_sigma_algebra` predicate. A notable effect is that the mathematical result stating that any generated σ-algebra is the smallest σ-algebra containing its generator is already structurally granted by the inductive type `measurable`. Our definition of generated σ-algebra then precedes the definition of σ-algebra, which is not the common order in mathematics. It is, however, common to have several possible orders or equivalent definitions in mathematics. We found that this formalization of generated σ-algebra was easy to use. For example, the Coq proof of lemma `measurable_fun_gen` (see Sect. 4.5) is done very simply by induction on the measurable subset defined inductively and applying directly the three properties `measurable_empty`, `measurable_compl` and `measurable_union_countable`.

Whatever the definitions (or the order of them), the most difficult point was related to the equivalence of generators. More precisely, the equivalence between the σ-algebra generated by compact intervals, `measurable gen_R`, and the Borel σ-algebra generated by open subsets,

[1] The `sigma_sets` inductive set from https://isabelle.in.tum.de/dist/library/HOL/HOL-Analysis/Sigma_Algebra.html.

[2] The `measurable_space.generate_measurable` inductive type from https://leanprover-community.github.io/mathlib_docs/measure_theory/measurable_space_def.html#measurable_space.generate_measurable.

`measurable open`, of Sect. 4.4 has proved tedious, long, and with harder ingredients than expected: many bijections, connected components, density of $\mathbb{Q}$, second-countability.

9.4 Simple Functions

About simple functions, we had difficulties designing them and we tried many Coq definitions for the same mathematical object before deciding for the one described in Sect. 6.

Note, for instance, that we have chosen the total function approach as much as possible in our development to ensure the simplicity in writing formulas. But valid simple functions come as a dependent type `SF` with the function, the list of values and the corresponding proof. This was needed as the list is required to compute the integral. To give the value of this integral, we need to sum over a finite list and therefore we need this list to be given. A solution would be to have an easy mechanism to sum over arbitrary sets (possibly infinite and possibly bigger than $\mathbb{N}$) like done in Lean.[3] This extension of total function would make a practical addition to Coq and may simplify some of our statements but is out of the scope of this paper. Note also that this dependent type `SF` is not supposed to appear to a library user, contrary to measurable functions and subsets, and to the integral.

Another design choice is about the type of simple functions: either `E → R`, or `E → Rbar`. Mathematicians usually consider $\mathbb{R}$ (in fact $\mathbb{R}_+$) as codomain for $\mathcal{SF}_+$, and reserve $\overline{\mathbb{R}}$ (in fact $\overline{\mathbb{R}}_+$) for $\mathcal{M}_+$ (as limits of functions in $\mathcal{SF}_+$). We first tried to have simple functions of type `E → Rbar` for coherence and simplicity, but it failed in the proof of the difficult `sum_Rbar_map_change_of_variable` of Sect. 6.3. We could have kept the same type with an assumption that all values taken are finite, but we found it less convenient than using types for this requirement. As suggested in mathematics, we ended up by having simple functions of type `E → R`.

A surprising successful choice is related to a particular type of simple functions: adapted sequences. Even if the mathematical definitions and proofs are given, we chose to take a computer scientist (or even a computer arithmetic) point of view. The use of Flocq has made trivial many proofs.

10 State of the Art

Interactive theorem provers may be classified into several distinct families with respect to their inherent logic. It may rely on set theory or type theory, on classical logic (e.g., with a built-in axiom of choice) or intuitionistic logic, and so on. Similarly to programming languages, the choice of a proof assistant is driven by the kind of formal proofs to be developed, the support libraries, the ease of the developer, and so on. It is currently either impractical or impossible to automatically "translate" developments done within some proof assistant into another, especially when they do not share the same inherent logic or theory.

This is the main reason for this work: providing to Coq users a practical formalization of Lebesgue integration. As explained below, this already exists in several other provers. We have taken inspiration (within the limits set by the various logics), we have made various design choices and proved the common lemmas and theorems. Our goal is to offer to Coq users (so that they may rely on it for their own Coq development) this library. This also allows this library to be eventually mixed it with Flocq for proving floating-point programs.

[3] The `tsum` operator from https://leanprover-community.github.io/mathlib_docs/topology/algebra/infinite_sum.html#tsum.

Lebesgue measure and integration is known to be an important chapter in mathematics. For instance, it belongs to a "top 100" of mathematical theorems established at the turn of the millennium for which F. Wiedijk is keeping track of the formalization within the main proof assistants.[4] This state of the art focuses on the formalization of the Lebesgue measure and integration; for a larger view about real analysis (in ACL2, Coq, HOL, HOL Light, Mizar, and PVS), we refer the reader to this survey [14].

Regarding intrinsically classical proof assistants, we may cite Mizar [53], Isabelle/HOL [58], and PVS [59].

Mizar libraries have been very advanced since the 1990s on the formalization of mathematics in general. The Lebesgue integral has been addressed continuously during the last two decades, for instance, the integral of simple functions [31], Fatou's lemma [32], and Fubini's theorem [30].

More recently, a large library of results, including many results about measure theory [43] and nonnegative Lebesgue integration,[5] Ordinary differential equations (ODEs) [44–46], and a formalization of Green's theorem [1] have been developed in Isabelle/HOL, and Fourier transform [41] in HOL4. In HOL4 [55] and in PVS,[6] Lebesgue integration theory has also been developed in 2010.

In these provers, the definition of the Lebesgue integral is quite similar to ours. The main difference lies in the definition of simple functions. Leaving measurability aside, they consider the image set and the function is simple when this set is finite. For instance, in PVS, `is_finite(image(f,fullset[T]))`.

Regarding intrinsically intuitionistic proof assistants, we may cite Lean [27], and Coq [25].

About Lean, we only found out in [66] that the Bochner integral is available. Note that the Bochner integral extends the Lebesgue integral to functions taking their values in a Banach space. There is no description on how all this is formalized and the available theorems. We investigated the source, and found the Beppo Levi (monotone convergence) theorem and Fatou's lemma, with some formalization key points. First, Borel spaces[7] are generated from the open sets (only) with a kind of inductive definition, so this ends up being similar to us, even if our genericity in terms of generators has proved useful. Second, as in other provers, simple function definitions[8] rely on a predicate that says whether a set is finite or not (used on the image of the function), which relies on the logical underlying framework that is quite different from the one of Coq.

As mentioned in Introduction, our work relies on the Coq proof assistant. As a previous work about analysis in Coq we can cite formalization of Picard's operator for ODEs [52], which uses the constructive CoRN and the Math Classes libraries. Regarding analysis with classical reals, our previous works in Coq include the full formalization of the discretization of the wave equation [11,12], and the formalization of Lax–Milgram theorem [15]. For both of these works, we paid particular attention to the statements and their proof to avoid the use of classical axioms (we have now chosen the classical side, see Sect. 9.2). Another recent development is the formal proof of the Lax equivalence theorem for finite difference

[4] https://www.cs.ru.nl/~freek/100/.

[5] https://isabelle.in.tum.de/dist/library/HOL/HOL-Analysis/Nonnegative_Lebesgue_Integration.html.

[6] https://shemesh.larc.nasa.gov/fm/ftp/larc/PVS-library/library/measure_integration.html, https://shemesh.larc.nasa.gov/fm/ftp/larc/PVS-library/library/lebesgue.html.

[7] https://leanprover-community.github.io/mathlib_docs/measure_theory/constructions/borel_space.html#borel_space.

[8] https://leanprover-community.github.io/mathlib_docs/measure_theory/integral/lebesgue.html#measure_theory.simple_func.

schemes [65], it is based on the classical standard real numbers, Coquelicot libraries, and our formalization of the Lax–Milgram theorem.

On the other hand, the math-comp/analysis library is currently being developed [3]. It aims at providing numerical analysis results in classical logic, building upon math-comp. This library has been developed in parallel to ours and is still in development, with few documentation and many branches, so it is hard to trace proved theorems.[9] We found out similar definitions for σ-algebra (by induction), connected components and the Lebesgue integral. In one branch, we found out a very experimental generic integral but in another one dedicated to the Lebesgue integral, we found some of the lemmas described in this article. As for the differences, they define simple functions by a dependent type including a unique list. Our adapted sequences are more smooth as based on Flocq, while they handle dyadic intervals by hand. They have a full definition of the Lebesgue measure using Carathéodory's extension theorem.

Last, following the rebuilding of the standard real library [64], in which a constructive and classical basis was built up, constructive analysis lemmas were also introduced by Vincent Semeria, based on the constructive analysis of Bishop [7].

11 Conclusion and Perspectives

This work is a second stone that paves the way toward the formal correctness of the Finite Element Method, the first one being the formal proof of the Lax–Milgram theorem [15]. The contributions are the Coq formalizations and proofs of σ-algebras, measures, simple functions, and Lebesgue integration of nonnegative measurable functions, and the formal proofs of the Beppo Levi (monotone convergence) theorem and Fatou's lemma.

The subset addressed in the present paper is more than 50-page long (6000 lines of code (LOC) of LaTeX, and weighs 220 kB) in its mathematics counterpart [23]. These Coq source files add up to about 11 kLOC, and weigh 370 kB. In total, the cumulative development including the Lax–Milgram theorem [15], finite-dimensional vector spaces [34], and this formalization is about 21 kLOC/650 kB.

As in [12], we observe here again that in-depth pen-and-paper proofs can be an order of magnitude longer than usual proofs from textbooks, and the lengths of formal and detailed pen-and-paper proofs are similar. In the present case, we can notice that the extra effort deployed on `Rbar` (in `Rbar_compl` and `sum_Rbar_nonneg`, roughly 3 kLOC/90 kB) explains part of the gap between the respective sizes of the Coq formalization and the pen-and-paper proof, where $\overline{\mathbb{R}}$ received far less detail.

The hyperlinked dependency graph is detailed in Fig. 1 where our target development, the Lebesgue integral (built on simple functions), is represented in blue. For this purpose, we had to formalize σ-algebras, measurable functions, and measures (represented in green), as well as some preliminary developments on countability, topological bases in $\mathbb{R}$, and the handling of sums in $\overline{\mathbb{R}}$ (represented in yellow). We also had to develop results that were missing both in the standard libraries (in the subdirectories Logic, Lists, Sorting, and Reals) and Coquelicot (represented in brown), including some tactics for `Rbar`. As usual, we can note that a large number of prerequisites are necessary to reach the desired formalization. In our case, this distributes roughly into one third for the complements (either in kLOC or

[9] https://github.com/math-comp/analysis.

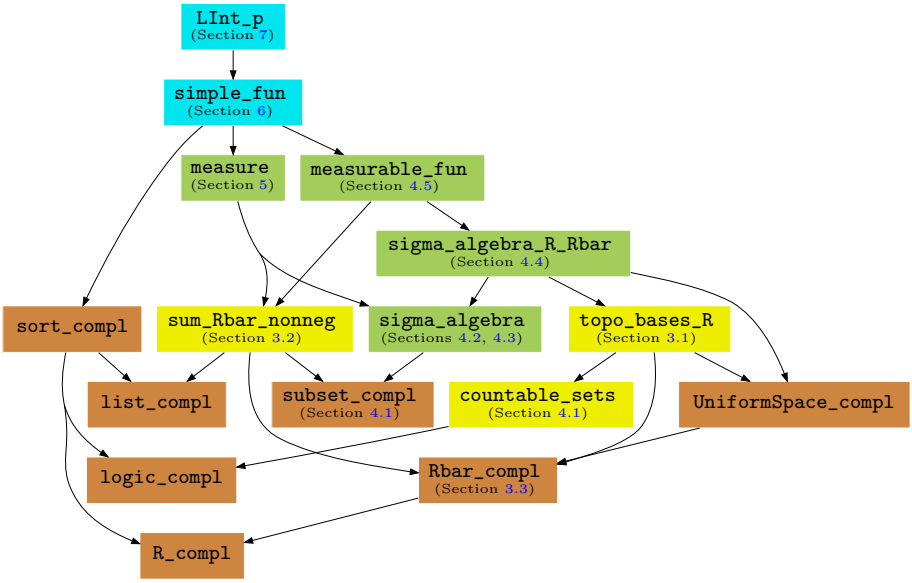

Fig. 1 Dependency graph of our Coq development.
Brown : complements to standard libraries and Coquelicot.
Yellow : new preliminary developments.
Green : new developments in measure theory.
Blue : new developments in Lebesgue integration

in kB), one seventh for the preliminaries, one third for measure theory, and one fifth for the target Lebesgue integral.

As usual, formalization is not just straightforward translation of mathematical texts and formulas. Some design choices have to be made and proof paths may differ, mainly to favor usability of Coq theorems and ease formal developments.

After both the Lax–Milgram theorem [15] and this work, the road is still long to be able to tackle the formal proof of scientific computation programs using the Finite Element Method (FEM).

The next step is to formalize the construction of the Lebesgue measure (for instance, using Carathéodory's extension theorem [20,29]), and the Lebesgue integral for measurable functions with arbitrary sign, with the proofs of Lebesgue's dominated convergence theorem and of the Tonelli–Fubini theorems as the next milestones. Then comes the formalization of the L^p Lebesgue spaces as complete normed vector spaces (a.k.a. Banach spaces), and in particular of L^2 as a complete inner product space (a.k.a. a Hilbert space). We expect the completeness to be the most challenging part of the proof. And finally, the formalization of simple Sobolev spaces H^1 and H_0^1 will need part of the distribution theory [63]. Further steps will include the formalization of parts of interpolation and approximation theories to end up with the FEM.

In parallel, we plan to merge with recent works on constructive reals [64] now included in the Coq standard library, and in particular with the constructive measure theory [8] based on the Daniell integral [26]. We also plan to formalize the Bochner integral [9,56] that

generalizes the Lebesgue integral to the case of functions taking their values in a Banach space, for instance, such as the Euclidean spaces $\mathbb{R}^n$ and the Hermitian spaces $\mathbb{C}^n$.

Acknowledgements We are deeply grateful to Stéphane Aubry for some proofs and tactics on `Rbar` properties, and to Assia Mahboubi, Cyril Cohen, Guillaume Melquiond, and Vincent Semeria for fruitful discussions about `Rbar` and `R` in Coq. This work was partly supported by the Paris Île-de-France Region (DIM RFSI MILC). This work was partly supported by Labex DigiCosme (project ANR-11-LABEX-0045-DIGICOSME) operated by ANR as part of the program "Investissement d'Avenir" Idex Paris-Saclay (ANR-11-IDEX-0003-02).
This work was partly supported by the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Programme—Grant Agreement n°810367.

References

1. Abdulaziz, M., Paulson, L.C.: An Isabelle/HOL formalisation of Green's theorem. In: Blanchette, J.C., Merz, S. (eds.) Proceedings of the 7th International Conference on Interactive Theorem Proving (ITP 2016), Volume 9807 of Lecture Notes in Computer Science, pp. 3–19. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-43144-4_1
2. Adams, R.A.: Sobolev Spaces. Pure and Applied Mathematics, vol. 65. Academic Press, New York–San Francisco–London (1975)
3. Affeldt, R., Cohen, C., Mahboubi, A., Rouhling, D., Strub, P.-Y.: Classical analysis with Coq. In: The 9th Coq Workshop (2018). https://easychair.org/smart-slide/slide/n3pK
4. Bartle, R.G.: A Modern Theory of Integration, Volume 32 of Graduate Studies in Mathematics. American Mathematical Society, Providence (2001). https://doi.org/10.1090/gsm/032
5. Bertot, Y., Gonthier, G., Biha, S.O., Pasca, I.: Canonical big operators. In: Mohamed, O.A., Muñoz, C., Tahar, S. (eds.) Proceedings of the 21st International Conference on Theorem Proving in Higher Order Logics (TPHOL 2008), Volume 5170 of Lecture Notes in Computer Science, pp. 86–101. Springer, Berlin (2008). https://doi.org/10.1007/978-3-540-71067-7_11
6. Billingsley, P.: Probability and Measure. Wiley Series in Probability and Mathematical Statistics, 3rd edn. Wiley, New York (1995)
7. Bishop, E.: Foundations of Constructive Analysis. McGraw-Hill Book Co., New York–Toronto–London (1967)
8. Bishop, E., Cheng, H.: Constructive Measure Theory. Number 116 in Memoirs of the American Mathematical Society. American Mathematical Society, Providence (1972)
9. Bochner, S.: Integration von funktionen, deren werte die elemente eines vektorraumes sind. Fundam. Math. **20**, 262–276 (1933). (**In German**)
10. Boldo, S., Melquiond, G.: Flocq: a unified library for proving floating-point algorithms in Coq. In: Proceedings of the IEEE 20th Symposium on Computer Arithmetic (ARITH-20), pp. 243–252. IEEE (2011). https://doi.org/10.1109/ARITH17396.2011
11. Boldo, S., Clément, F., Filliâtre, J.-C., Mayero, M., Melquiond, G., Weis, P.: Wave equation numerical resolution: a comprehensive mechanized proof of a C program. J. Autom. Reason. **50**(4), 423–456 (2013). https://hal.inria.fr/hal-00649240/
12. Boldo, S., Clément, F., Filliâtre, J.-C., Mayero, M., Melquiond, G., Weis, P.: Trusting computations: a mechanized proof from partial differential equations to actual program. Comput. Math. Appl. **68**(3), 325–352 (2014). https://hal.inria.fr/hal-00769201/
13. Boldo, S., Lelay, C., Melquiond, G.: Coquelicot: a user-friendly library of real analysis for Coq. Math. Comput. Sci. **9**(1), 41–62 (2015). https://hal.inria.fr/hal-00860648/
14. Boldo, S., Lelay, C., Melquiond, G.: Formalization of real analysis: a survey of proof assistants and libraries. Math. Struct. Comput. Sci. **26**(7), 1196–1233 (2016). https://hal.inria.fr/hal-00806920/
15. Boldo, S., Clément, F., Faissole, F., Martin, V., Mayero, M.: A Coq formal proof of the Lax–Milgram theorem. In: Proceedings of the 6th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2017), pp. 79–89. Association for Computing Machinery, New York (2017). https://hal.inria.fr/hal-01391578/
16. Bourbaki, N.: Éléments de mathématiques. Livre VI : Intégration. Chapitres 1à 4, 2nd edn. Hermann, Paris (1965). **(In French)**
17. Bourbaki, N.: Éléments de mathématiques. Livre III : Topologie générale. Chapitres 1 à 4, 2nd edn. Hermann, Paris (1971). **(In French)**
18. Brezis, H.: Analyse fonctionnelle—Théorie et applications. Collection Mathématiques Appliquées pour la Maîtrise. Masson, Paris (1983). (**In French**)

19. Burk, F.E.: A Garden of Integrals, Volume 31 of The Dolciani Mathematical Expositions. Mathematical Association of America, Washington (2007)
20. Carathéodory, C.: Algebraic Theory of Measure and Integration. Chelsea Publishing Co., New York (1963)
21. Cartan, H.: Théorie des filtres. C. R. Acad. Sci. **205**, 595–598 (1937). (**In French**)
22. Ciarlet, P.G.: The Finite Element Method for Elliptic Problems, Volume 40 of Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM), Philadelphia (2002). https://doi.org/10.1137/1.9780898719208. Reprint of the 1978 original (North-Holland, Amsterdam)
23. Clément, F., Martin, V.: Lebesgue integration. Detailed proofs to be formalized in Coq. Research Report RR-9386, Inria, Paris, Jan 2021. https://hal.inria.fr/hal-03105815v2. Version 2
24. Cohn, D.L.: Measure Theory. Birkhäuser Advanced Texts: Basler Lehrbücher, 2nd edn. Birkhäuser/Springer, New York (2013). https://doi.org/10.1007/978-1-4614-6956-8
25. Coq-ref. The Coq reference manual. https://coq.inria.fr/refman/
26. Daniell, P.J.: A general form of integral. Ann. Math. (2) **19**(4), 279–294 (1918). https://doi.org/10.2307/1967495
27. de Moura, L., Kong, S., Avigad, J., van Doorn, F., von Raumer, J.: The Lean theorem prover (system description). In: Felty, A.P., Middeldorp, A. (eds.) Proceedings of the 25th International Conference on Automated Deduction (CADE 2015), Volume 9195 of Lecture Notes in Computer Science, pp. 378–388. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21401-6_26
28. Dieudonné, J.: Éléments d'analyse. Chapitres XII à XV. Cahiers Scientifiques, Fasc. XXXI. Gauthier-Villars, Paris, Tome II (1968). (**In French**)
29. Durrett, R.: Probability—Theory and Examples, Volume 49 of Cambridge Series in Statistical and Probabilistic Mathematics, 5th edn. Cambridge University Press, Cambridge (2019). https://doi.org/10.1017/9781108591034
30. Endou, N.: Fubini's theorem. Formaliz. Math. **27**(1), 67–74 (2019). https://doi.org/10.2478/forma-2019-0007
31. Endou, N., Narita, K., Shidama, Y.: Lebesgue integral of simple valued function in Mizar. Formaliz. Math. **13**(1), 67–71 (2005). https://fm.mizar.org/2005-13/pdf13-1/mesfunc3.pdf
32. Endou, N., Narita, K., Shidama, Y.: Fatou's lemma and the Lebesgue's convergence theorem. Formaliz. Math. **16**(4), 305–309 (2008). https://doi.org/10.2478/v10037-008-0037-8
33. Ern, A., Guermond, J.-L.: Theory and Practice of Finite Elements, Volume 159 of Applied Mathematical Sciences. Springer, New York (2004). https://doi.org/10.1007/978-1-4757-4355-5
34. Faissole, F.: Formalization and closedness of finite dimensional subspaces. In: Proceedings of the 19th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2017), pp. 121–128. IEEE (2017)
35. Feller, W.: An Introduction to Probability Theory and Its Applications, vol. I, 3rd edn. Wiley, New York–London–Sydney (1968)
36. Folland, G.B.: Real Analysis—Modern Techniques and Their Applications. Pure and Applied Mathematics (New York), 2nd edn. Wiley, New York (1999)
37. Gallouët, T., Herbin, R.: Mesure, intégration, probabilités. Ellipses Edition Marketing (2013). https://hal.archives-ouvertes.fr/hal-01283567/. **(In French)**
38. Ghosal, S., van der Vaart, A.: Fundamentals of Nonparametric Bayesian Inference, Volume 44 of Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge (2017). https://doi.org/10.1017/9781139029834
39. Gill, T.L., Zachary, W.W.: Functional Analysis and the Feynman Operator Calculus. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-27595-6
40. Gostiaux, B.: Cours de mathématiques spéciales—2. Topologie, analyse réelle. Mathématiques. Presses Universitaires de France, Paris (1993). **(In French)**
41. Guan, Y., Zhang, J., Shi, Z., Wang, Y., Li, Y.: Formalization of continuous Fourier transform in verifying applications for dependable cyber-physical systems. J. Syst. Archit. **106**, 101707 (2020). https://doi.org/10.1016/j.sysarc.2020.101707
42. Henstock, R.: Theory of Integration. Buttherworths, London (1963)
43. Hölzl, J., Heller, A.: Three chapters of measure theory in Isabelle/HOL. In: van Eekelen, M., Geuvers, H., Schmaltz, J., Wiedijk, F. (eds.) Proceedings of the 2nd International Conference on Interactive Theorem Proving (ITP 2011), Volume 6898 of Lecture Notes in Computer Science, pp. 135–151. Springer, Berlin (2011). https://doi.org/10.1007/978-3-642-22863-6_12
44. Immler, F.: Formally verified computation of enclosures of solutions of ordinary differential equations. In: Badger, J.M., Rozier, K.Y. (eds.) Proceedings of the 6th International Symposium NASA Formal Methods (NFM 2014), Volume 8430 of Lecture Notes in Computer Science, pp. 113–127. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06200-6_9

45. Immler, F., Hölzl, J.: Numerical analysis of ordinary differential equations in Isabelle/HOL. In: Beringer, L., Felty, A.P. (eds.) Proceedings of the 3rd International Conference on Interactive Theorem Proving (ITP 2012), Volume 7406 of Lecture Notes in Computer Science, pp. 377–392. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-32347-8_26
46. Immler, F., Traut, C.: The flow of ODEs. In: Blanchette, C.J., Merz, S. (eds.) Proceedings of the 7th International Conference on Interactive Theorem Proving (ITP 2016), Volume 9807 of Lecture Notes in Computer Science, pp. 184–199. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-43144-4_12
47. Kurzweil, J.: Generalized ordinary differential equations and continuous dependence on a parameter. Czechoslov. Math. J. **7**(3), 418–449 (1957). https://doi.org/10.21136/CMJ.1957.100258
48. Lebesgue, H.L.: Leçons sur l'intégration et la recherche des fonctions primitives professées au Collège de France. Cambridge Library Collection. Cambridge University Press, Cambridge (2009). https://doi.org/10.1017/CBO9780511701825. Reprint of the 1904 original [Gauthier-Villars, Paris]. **(In French)**
49. Lelay, C.: Repenser la bibliothèque réelle de Coq : vers une formalisation de l'analyse classique mieux adaptée. Thèse de doctorat, Université Paris-Sud, June (2015). https://tel.archives-ouvertes.fr/tel-01228517/. **(In French)**
50. Lelay, C.: How to express convergence for analysis in Coq. In: The 7th Coq Workshop, June 2015. https://hal.archives-ouvertes.fr/hal-01169321/
51. Maisonneuve, F.: Mathématiques 2: Intégration, transformations, int'egrales et applications—Cours et exercices. Presses de l'École des Mines (2014). **(In French)**
52. Makarov, E., Spitters, B.: The Picard algorithm for ordinary differential equations in Coq. In: Blazy, S., Paulin-Mohring, C., Pichardie, D. (eds.) Proceedings of the 4th International Conference on Interactive Theorem Proving (ITP 2013), Volume 7998 of Lecture Notes in Computer Science, pp. 463–468. Springer, Berlin (2013). https://doi.org/10.1007/978-3-642-39634-2_34
53. Matuszewski, R. (ed.): Formalized Mathematics. Sciendo, Poland. https://fm.mizar.org/
54. Mayero, M.: Formalisation et automatisation de preuves en analyses réelle et numérique. Thèse de doctorat, Université Paris VI, December 2001. http://www-lipn.univ-paris13.fr/~mayero/publis/these-mayero.ps.gz. **(In French)**
55. Mhamdi, T., Hasan, O., Tahar, S.: On the formalization of the lebesgue integration theory in HOL. In: Kaufmann, M., Paulson, L.C. (eds.) Proceedings of the 1st International Conference on Interactive Theorem Proving (ITP 2010), Volume 6172 of Lecture Notes in Computer Science, pp. 387–402. Springer, Berlin (2010). https://doi.org/10.1007/978-3-642-14052-5_27
56. Mikusiński, J.: The Bochner Integral. Academic Press, New York–San Francisco (1978)
57. Musial, P., Tulone, F.: Dual of the class of HK_*r* integrable functions. Minimax Theory Appl. **4**(1), 151–160 (2019)
58. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL—A Proof Assistant for Higher-Order Logic, Volume 2283 of Lecture Notes in Computer Science. Springer, Berlin (2002). https://doi.org/10.1007/3-540-45949-9
59. Owre, S., Rushby, J.M., Shankar, N.: PVS: a prototype verification system. In: Kapur, D. (ed.) Proceedings of the 11th International Conference on Automated Deduction (CADE 1992), Volume 607 of Lecture Notes in Computer Science, pp. 748–752. Springer, Berlin (1992). https://doi.org/10.1007/3-540-55602-8_217
60. Owre, S., Shankar, N., Rushby, J.M., Stringer-Calvert, D.W.J.: PVS system guide. In: SRI International, Computer Science Laboratory, Menlo Park, CA, August 2020. http://pvs.csl.sri.com/doc/pvs-system-guide.pdf. Version 7.1 [1st version in 1999]
61. Quarteroni, A., Valli, A.: Numerical Approximation of Partial Differential Equations. Springer Series in Computational Mathematics, vol. 23. Springer, Berlin (1994)
62. Rudin, W.: Real and Complex Analysis, 3rd edn. McGraw-Hill Book Co., New York (1987)
63. Schwartz, L.: Théorie des distributions, 2nd edn. Hermann, Paris (1966). 1st edition in 1950–1951. **(In French)**
64. Semeria, V.: Nombres réels dans Coq. In: Dargaye, Z., Regis-Gianas, Y. (eds.) Actes des 31es Journées Francophones des Langages Applicatifs (JFLA), pp. 104–111. IRIF (2020). https://hal.inria.fr/hal-02427360/. **(In French)**
65. Tekriwal, M., Duraisamy, K., Jeannin, J.-B.: A formal proof of the Lax equivalence theorem for finite difference schemes. In: 13th International Symposium on NASA Formal Methods (NFM 2021) (2021). **(To appear)**
66. The mathlib Community. The Lean mathematical library. In: Blanchette, J., Hritcu, C. (eds.) Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2020), pp. 367–381. ACM (2020). https://doi.org/10.1145/3372885.3373824

67. Tsybakov, A.B.: Introduction to Nonparametric Estimation. Springer Series in Statistics. Springer, New York (2009). https://doi.org/10.1007/b13794. Revised and extended from the 2004 French original [Springer, Berlin], Zaiats, V. (Trans.)
68. Weil, A.: Sur les espaces à structure uniforme et sur la topologie générale. Hermann, Paris (1937). (**In French**)
69. Yosida, K.: Functional Analysis. Classics in Mathematics. Springer, Berlin (1995). https://doi.org/10.1007/978-3-642-61859-8. Reprint of the 6th (1980) edition (Springer, Berlin)
70. Zienkiewicz, O.C., Taylor, R.L., Zhu, J.Z.: The Finite Element Method: Its Basis and Fundamentals, 7th edn. Elsevier/Butterworth Heinemann, Amsterdam (2013). https://doi.org/10.1016/B978-1-85617-633-0.00001-0

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.