Probability theory can be fun and simple with dependent types (Yet another formal theory of probabilities in Coq)

Reynald Affeldt, Alessandro Bruni, Pierre Roux, Takafumi Saikawa

30th International Conference on Types for Proofs and Programs
10 - 14 June 2024

An overview of existing formalizations of probabilities in Coq [1]

InfoTheo (2009–ongoing)

- Formalizes *finite probabilities*; used for information theory [JAR 2014], error-correcting codes [JAR 2020], robust statistics [ITP 2024]

coq-proba [Tassarotti, 2023]

- Used to verify a compiler for probabilistic programming languages [PLDI 2023]

FormalML [The FormalML development team, 2023]

- Contains *advanced theorems* in probability theory, e.g., a stochastic approximation theorem [ITP 2022]

[1] Isabelle/HOL and Mathlib have extensive libraries for probabilities, this talk focuses on Coq

A proof engineering effort

Mathematical Components

29 followers https://math-comp.github.io/math-...

math-comp Public
Mathematical Components
Coq 541 109 98 34 Updated 7 minutes ago

analysis Public
Mathematical Components compliant Analysis Library
Coq 176 40 72 (1 issue needs help) 38 Updated 2 hours ago

real-closed Public
Theorems for Real Closed Fields
Coq 12 10 5 3 Updated 5 hours ago

hierarchy-builder Public
High level commands to declare a hierarchy based on packed classes
Prolog 90 MIT 19 64 13 Updated 10 hours ago

Applications of MathComp-Analysis to probabilities?

MathComp-Analysis timeline

- Asymptotic reasoning + Landau notations $\rightarrow$ differentiability [JFR 2018]
- Lebesgue integral [JAR 2023]
- Fundamental theorem of calculus [Affeldt and Stone, 2024]
- Probability theory (2023–ongoing)

Applications to probabilities

- Verified probabilistic programming languages [CPP 2023, APLAS 2023]
- Verified worst-case failure probability of real-time systems [Markovic et al., 2023]

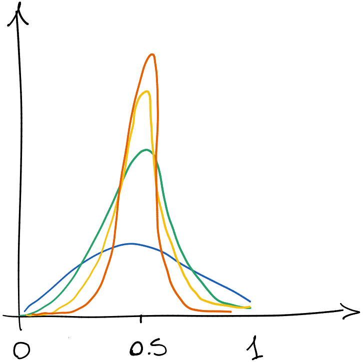
Other planned applications

- Verified robust statistics [PPDP 2021, ITP 2024]
- Verified machine learning [Ślusarz et al., ITP 2024]

An example: Bernoulli sampling [Rajani, 2019]

Bernoulli sampling

Given n independent 0-1 random variables X_i, $p \in (0, 1]$, $\theta \in (0, p)$, $\delta \in (0, 1]$ with $Pr(X_i = 1) = p$, $X = \sum_{i=1}^{n} X_i$, and $\bar{X} = \frac{X}{n}$, then $Pr(|\bar{X} - p| \leq \theta) \geq 1 - \delta$ when $n \geq \frac{3}{\theta^2} \ln(\frac{2}{\delta})$.

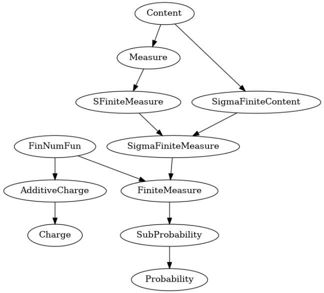


Simple and general: inherit from measure theory with Hierarchy Builder

Definition (Measure)

A measure $\mu : \mathcal{P}(T) \to \overline{\mathbb{R}}$ satisfies:

1. $\mu(\emptyset) = 0$ (measure-0)
2. $0 \leq \mu(A)$ for any set A (non-negativity)
3. $\mu(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mu(A_i)$ (σ-additivity)

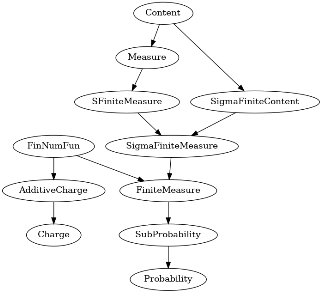


Simple and general: inherit from measure theory with Hierarchy Builder

Definition (Measure)

A measure $\mu : \mathcal{P}(T) \to \overline{\mathbb{R}}$ satisfies:

1. $\mu(\emptyset) = 0$ (measure-0)
2. $0 \leq \mu(A)$ for any set A (non-negativity)
3. $\mu(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mu(A_i)$ (σ-additivity)



Definition (Probability measure)

A probability measure additionally implements the following interface:

```
HB.factory Record Measure_isProbability d (T : measurableType d)
    (R : realType) (P : set T -> \bar R) of isMeasure _ _ _ P :=
  { probability_setT : P setT = 1%E }.
```

…and fun: random variables and expectations

```
Context d (T : measurableType d) (R : realType) (P : probability T R).
```

Definition (Random variables)

A random variable is neither random, nor a variable. It's a measurable function from `T` to `R`.

```
Definition random_variable := {mfun T >-> R}.
Notation "{ 'RV' P >-> R }" := (@random_variable _ _ R P).
```

…and fun: random variables and expectations

```
Context d (T : measurableType d) (R : realType) (P : probability T R).
```

Definition (Random variables)

A random variable is neither random, nor a variable. It's a measurable function from `T` to `R`.

```
Definition random_variable := {mfun T >-> R}.
Notation "{ 'RV' P >-> R }" := (@random_variable _ _ R P).
```

Definition (Expectation)

Expectation of X with the measure P can be expressed as the Lebesgue integral $\int X\, dP$:

```
Definition expectation (X : {RV P >-> R}) := \int[P]_w (X w)%:E.
```

Recovering discreteness

Discrete (random) variables

Discrete random variables additionally implement the following interface:

```
HB.mixin Record MeasurableFun_isDiscrete d (T : measurableType d) (R : realType)
  (X : T -> R) of @MeasurableFun d T R X := { countable_range : countable (range X) }.
```

Recovering discreteness

Discrete (random) variables

Discrete random variables additionally implement the following interface:

```
HB.mixin Record MeasurableFun_isDiscrete d (T : measurableType d) (R : realType)
  (X : T -> R) of @MeasurableFun d T R X := { countable_range : countable (range X) }.
```

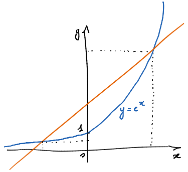
Discrete sums

When `X : {dRV P >-> R}` (the type of discrete random variables),
we build a function a_k to enumerate its values, and c_k to enumerate the probabilities,
so that the distribution can be written as $\sum_k c_k \delta_{a_k}$:

```
Lemma distribution_dRV A : measurable A ->
  distribution P X A = \sum_(k <oo) a X k * \d_(c X k) A.
```

(More) formal adventures in convex spaces

[Saikawa et al., CICM 2020] shows that probability theory benefits from a theory of *convex spaces*.
We are porting it to MathComp-Analysis to define convex functions:



Convex function

```
Definition convex_function (R : realType) (D : set R) (f : R -> R) :=
  forall t : {i01 R}, {in D &, forall (x y : R), f (x <| t |> y) <= f x <| t |> f y}.
```

Exponentials are convex

```
Lemma convex_expR : convex_function setT expR.
Lemma convex_powR p : 1 <= p -> convex_function `[0, +oo[ (fun x : R => powR x p).
```

Moments: exponential expectations

```
Definition mmt_gen_fun (X : {RV P >-> R}) (t : R) := 'E_P[expR \o t \o* X].
```

Applications of convexity: Hölder and Minkowski and L_p -spaces

We are building a theory of L_p -spaces. For that purpose we prove Hölder's and Minkowski's inequalities, which are also generally applicable to probabilities:

Hölder

```
Lemma hoelder (f g : T -> R) (p q : R) : measurable_fun setT f -> measurable_fun setT g ->
  0 < p  ->  0 < q  ->  p^-1 + q^-1 = 1 (* Hoelder conjugates *) ->
    'N_1 [f \* g] <= 'N_p [f] * 'N_q [g].
```

(Here `\+` and `\*` are pointwise addition and multiplication, and `N_p [f]` is the p-norm of `f`)

Applications of convexity: Hölder and Minkowski and L_p -spaces

We are building a theory of L_p -spaces. For that purpose we prove Hölder's and Minkowski's inequalities, which are also generally applicable to probabilities:

Hölder

```
Lemma hoelder (f g : T -> R) (p q : R) : measurable_fun setT f -> measurable_fun setT g ->
  0 < p  ->  0 < q  ->  p^-1 + q^-1 = 1 (* Hoelder conjugates *) ->
    'N_1 [f \* g] <= 'N_p [f] * 'N_q [g].
```

Minkowski

```
Lemma minkowski f g p : measurable_fun setT f -> measurable_fun setT g -> 1 <= p ->
  'N_p%:E[f \+ g] <= 'N_p%:E[f] + 'N_p%:E[g].
```

(Here `\+` and `\*` are pointwise addition and multiplication, and `N_p [f]` is the p-norm of `f`)

More useful lemmas: Markov, Chernoff, Chebyshev and Cantelli

```
Lemma markov (X : {RV P >-> R}) (f : R -> R) (eps : R) : (0 < eps) ->
    measurable_fun [set: R] f -> (forall r, 0 <= r -> 0 <= f r) ->
    {in Num.nneg &, {homo f : x y / x <= y}} ->
  (f eps)%:E * P [set x | eps%:E <= `| (X x)%:E | ] <=
    'E_P[f \o (fun x => `| x |) \o X].

Lemma chernoff (X : {RV P >-> R}) (r a : R) : (0 < r) ->
  P [set x | X x >= a] <= mmt_gen_fun X r * (expR (- (r * a)))%:E.

Lemma chebyshev (X : {RV P >-> R}) (eps : R) : (0 < eps) ->
  P [set x | (eps <= `| X x - fine ('E_P[X])|) ] <= (eps ^- 2)%:E * 'V_P[X].

Lemma cantelli (X : {RV P >-> R}) (lambda : R) :
    P.-integrable setT (EFin \o X) -> P.-integrable setT (EFin \o (X ^+ 2)) ->
    (0 < lambda) ->
  P [set x | lambda%:E <= (X x)%:E - 'E_P[X]] <=
    (fine 'V_P[X] / (fine 'V_P[X] + lambda^2))%:E.
```

Our experiment (WIP): Bernoulli sampling [Rajani, 2019]

Theorem

Given n independent 0-1 random variables X_i, $p \in (0,1]$, $\theta \in (0,p)$, $\delta \in (0,1]$ with $Pr(X_i = 1) = p$, $X = \sum_{i=1}^{n} X_i$, and $\bar{X} = \frac{X}{n}$, then $Pr(|\bar{X} - p| \leq \theta) \geq 1 - \delta$ when $n \geq \frac{3}{\theta^2} \ln(\frac{2}{\delta})$.

becomes:

```
Theorem sampling (X : seq {RV P >-> R}) (theta delta p : R) :
  let n := size X in let X' x := ((\sum_(Xi in X) Xi) x) / n%:R in
  is_bernoulli_trial X n -> 0 < p <= 1 -> 0 < delta <= 1 ->
  0 < theta < p -> 0 < n -> 3 / theta^+2 * ln(2 / delta) <= n%:R
  -> P [set i | `| X' i - p | <= theta] >= 1 - delta%:E.
```

Conclusions

- We are generalizing Infotheo theories by porting them to MathComp-Analysis (future work: conditional probabilities, information theory, etc.)
- We are working on the verification of probabilistic programs by equational reasoning
- We aim to have a rich and general library that can be reused
- We are looking for contributors!