TYPES2019

A Formal Classical Proof of Hahn-Banach in Coq

Marie Kerjean & Assia Mahboubi

Inria Nantes , LS2N

Based Mathcomp *and* MathComp Analysis *libraries,*

developed by Reynald Affeldt, Cyril Cohen, Assia Mahboubi, Damien Rouhling,

Pierre-Yves Strub

Disclaimer

- I am not an expert in Type Theory and new to Formalisation of Mathematics.

```
case: z {zmax} gP => [c [_ _ bp _]] /= gP; apply/bp/gP .
```

- This proof is a test for the Mathematical Components Analysis libraries.

  https://github.com/math-comp/analysis/blob/hb/hahn_banach.v

- This talk : "a user experience of Mathematical Components Analysis".

Lemma 001 of functional analysis

Hahn-Banach Theorem

Consider V a normed space, F a sub-vector space of V, and $f : V -> \mathbb{R}$ a continuous linear form on F. Then there exists a linear continuous form $g : V -> \mathbb{R}$ that extends f.

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : V -> R) (linf : linear_on F f) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Textbook Proof:

- Extending f to a linear function $F \oplus \mathbb{R}v$ bounded by $p$is follows from the convexity of p and the linearity required for the extension.
- Extending f to the whole space V is done through Zorn's lemma.

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : V -> R) (linf : linear_on F f) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Textbook Proof: [Linear Algebra]

- Extending f to a linear function $F \oplus \mathbb{R}v$ bounded by $p$is follows from the convexity of p and the linearity required for the extension.
- Extending f to the whole space V is done through Zorn's lemma.

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : V -> R) (linf : linear_on F f) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Textbook Proof: [Linear Algebra]

- Extending f to a linear function $F \oplus \mathbb{R}v$ bounded by $p$is follows from the convexity of p and the linearity required for the extension.

  [real analysis and classical reasoning]

- Extending f to the whole space V is done through Zorn's lemma.

  [Axiome of Choice]

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : scalar V) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

This is my favorite **existence theorem**, with countless applications.

Separation theorems.
Duality Theory for locally convex vector spaces.
Fundamental solutions to certain differential equations.

Existing Formalisations

- Existing Formalisations in Mizar [1993], PVS and HoL/Isabelle [2000]

- Investigation on a constructive version in point-free topology by Coquand, Negri and Cederquist.

Mathematical-Components

A library in Coq constructed for the formalization of Feit-Thompson theorem [Gonthier and al., 2012].

Libraries for algebra with a strong taste for finite dimension:

- Finite Group Theory.
- Ring and modules.
- Finites dimensional vector spaces.
- Matrixes and Polynomials

Ssreflect : un peu, beaucoup, à la folie

- Ssreflect is a set of tactics and notations, used extensively in the Mathcomp libraries.
- MathComp Proofs are often written in an **imperative minimal style** : easier to maintain.
-

Ssreflect : un peu, beaucoup, à la folie

- Ssreflect is a set of tactits and notations, used extensively in the Mathcomp libraries.
- MathComp Proofs are often written in an **imperative minimal style** : easier to maintain.
- The user can choose to use it **as much as she likes**.

```
Lemma linrel_00 x r : f x r -> f 0 0.
Proof.
suff -> : f 0 0 = f (x + (-1) *: x) (r + (-1) * r) by move=> h; apply: lrf.
by rewrite scaleNr mulNr mul1r scale1r !subrr.
Qed.

Lemma long_linrel_00 x r : f x r -> f 0 0.
Proof.
have H : f 0 0 = f (x + (-1) *: x) (r + (-1) * r).
  rewrite scaleNr
  rewrite mulNr
  by rewrite mul1r scale1r subrr subrr. (* unfold if you want *)
intro h. (* move => h*)
apply: lrf.
by [].
Qed.
```

Mathematical-Components- Analysis

Enough of Algebra.

Analysis !

Why ?

- Because it's fun.
- Because it is needed for verification.

  [P.-Y. Strub - EasyCrypt - probabilistic computation]
- Because it is needed for verifying robotics .

  [R. Affeldt, C. Cohen, D. Rouhling -CoqRobot - Lassalle Invariance]

Mathematical-Components- Analysis

Fact

- Formalisation in Coq has been influenced a lot by the constructive point of vue on mathematics - because it can.

Mathematical-Components- Analysis

Opinion

- Formalisation in Coq has been **very much** influenced by the constructive point of vue on mathematics - because it can.

Mathematical Components Analysis : CIC + + Axiome of Choice + Excluded middle + Functional Extensionality + Propositional Equality + Propositional Irrelevance

This library reinterprets and extends the **Coquelicot** project.

[Boldo and al, 2015]

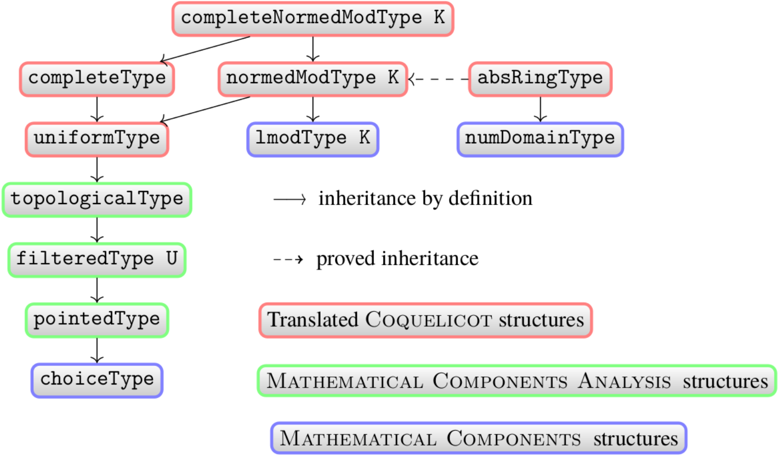

Figure: MATHEMATICAL COMPONENTS ANALYSIS hierarchy

[Cohen et al. 2018]

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : scalar V) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Textbook Proof:

- Extending f to a linear function $F \oplus \mathbb{R}v$ bounded by p follows from the convexity of p and the linearity required for the extension.
- Extending f to the whole space V is done through Zorn's lemma.

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : scalar V) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Textbook Proof: [Linear Algebra]

- Extending f to a linear function $F \oplus \mathbb{R}v$ bounded by p follows from the convexity of p and the linearity required for the extension.
- Extending f to the whole space V is done through Zorn's lemma.

Hahn-Banach before normed spaces

```
Variables (R : realFieldType) (V : lmodType R)
 (p : convex R) (F : submod V).

Theorem HahnBanach (f : scalar V) :
 ( forall x, F x -> ( f x <= p x )) ->
 exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Textbook Proof: [Linear Algebra]

- Extending f to a linear function $F \oplus \mathbb{R}v$ bounded by p follows from the convexity of p and the linearity required for the extension.

  [real analysis and classical reasoning]

- Extending f to the whole space V is done through Zorn's lemma.

  [Axiome of Choice]

Hahn-Banach

Partial functions: reasoning on the graphs of functions.

```
f : V -> R -> Prop
```

```
Definition spec (g : V -> R -> Prop) :=
  [/\ functional g, linear_rel g, maj_by p g & forall v, F v -> g v (f v) ].

Record zorn_type : Type := ZornType
  {carrier : V -> R -> Prop; specP : spec carrier}.

Lemma domain_extend (z : zorn_type) v :
    exists2 ze : zorn_type, (zorn_rel z ze) & (exists r, (carrier ze) v r).

Theorem HahnBanach : exists g : {scalar V},
 (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

Hahn-Banach in normed spaces

The theorem is formalized, but questionable until it is not used somewhere:

https://github.com/math-comp/analysis/blob/hb/hahn_banach_applications.v

```
Variable (V : normedModType R)

Lemma continuousR_bounded0 (f : {scalar V}) :
  (continuousR_at 0 f) -> ( exists r , (r > 0 ) /\ (forall x : V, ( `|f x| ) <=
       (`|[x]| ) * r ) ) .

Theorem HB_geom_normed ( F : pred V ) (H : submod_closed F) (f : {scalar V}) :
 continuousR_on F f ->
  exists g : {scalar V} , ( continuous g ) /\ ( forall x, F x -> (g x = f x)).
```

- The tools are rewriting lemmas of convergence in terms of filters, neighborhoods or norms.
- What's missing is a good theory of sub-vector spaces and induced topologies.

Looking for Lemmas

```
Search (exists _ , _) "Hahn".
```

- Searching via patterns.

```
Search _ (exists _ , _) (continuous _) in topology.
```

- Searchin via names (next slide).

```
Search "HB".
Search "my_favorite_thm".
Search "why_on_earth_isnt_this_automated".
```

Looking for Lemmas

```
Search (exists _ , _) "Hahn".
```

- Searching via patterns.

  ```
  Search _ (exists _ , _) (continuous _) in topology.
  ```

- Searchin via names (next slide).

  ```
  Search "HB".
  Search "my_favorite_thm".
  Search "why_on_earth_isnt_this_automated".
  ```

- Combine the two.

Looking for Lemmas

```
Search (exists _ , _) "Hahn".
```

- Searching via patterns.

```
Search _ (exists _ , _) (continuous _) in topology.
```

- Searchin via names (next slide).

```
Search "HB".
Search "my_favorite_thm".
Search "why_on_earth_isnt_this_automated".
```

- Combine the two.
- Ask by mail / gitter.

Naming Convention

You should expect the name of the main statement in the lemma.

```
normedModType_hausdorff : forall (K : absRingType) (V : normedModType K),
     hausdorff V
```

A list of suffix abbreviation :

E : definition elimination, characteristic properties, P : characteristic propertie, Z : module/vector space scaling. ...

```
Lemma normmZ : forall (K : absRingType) (V : normedModType K) (l : K) (x : V),
       `|[l *: x]| = `|l|%real * `|[x]| .


Lemma flim_normP : forall (K : absRingType) (V : normedModType K) (F :
     classical_sets.set (classical_sets.set V)),
          Filter F -> forall y : V, F --> y <-> (forall eps : R, 0 < eps -> \near
                F, `|[y - F]| < eps).

Lemma near_withinE :
   forall (T : Type) (D : classical_sets.set T) (F : classical_sets.set (
        classical_sets.set T))
     (P : classical_sets.set T), (\forall x \near within D F, P x) = {near F, D
          `<=` P}
```

The Maths should be in Prop

```
Variable Choice : forall T U (P : T -> U -> Prop),
    (forall t : T, exists u : U, P t u) -> { e, forall t, P t (e t) }.

Theorem HahnBanach : exists g : {scalar V},
  (forall x, g x <= p x) /\ (forall x, F x -> g x = f x).
```

However:

- Proving a result in Prop should be done using only axioms in Prop.
- The proof of Zorn in boolp.v used extensively the Choice in Type and the constructive indefinite description.

```
Definition xget {T : choiceType} x0 (P : set T) : T :=
  if pselect (exists x : T, `[<P x>]) isn't left exP then x0
  else projT1 (sigW exP).
```

Fixpoint theorem and Zorn in Prop

Following Lang's Algebra book :

```
Lemma fixpoint_T ( R : {strict_inductive_order T}) (f : T -> T) :
  (forall t, R t (f t))) -> exists t, t = f t.

Lemma Zorn T (R : {order T}) :
  (forall A : set T, total_on A R -> exists t, forall s, A s -> R s t) ->
  exists t, forall s, R t s -> s = t.
```

Thus our formalisation of Hahn-Banach Theorem depends of the following axioms ;

```
Axiom prop_irrelevance : forall (P : Prop) (x y : P), x = y.
Axiom funext : forall (T U : Type) (f g : T -> U), (f =1 g) -> f = g.
Axiom propext : forall (P Q : Prop), (P <-> Q) -> (P = Q).

Axiom choice_prop := ((forall T U (Q : T -> U -> Prop),
 (forall t : T, exists u : U, Q t u) -> (exists e, forall t, Q t (e t))) .
```

Conclusion

Documentation:

- Slides by Cyril Cohen :

  https://perso.crans.org/cohen/CoqWS2018.pdf

- Lessons and exercices on Coq, Ssreflect and Mathcomp libraries:

  https://team.inria.fr/marelle/en/coq-winter-school-2018-2019-ssreflect-mathcomp/

- A Book by Assia Mahboubi and Enrico Tassi :

  https://math-comp.github.io/mcb/

- **Gitter** forums : tell us

  https://gitter.im/math-comp/analysis

Installation: Via git or opam, or soon via Nix.

All about $\mathbb{R}$

- $\mathbb{R}$ in coq/reals.v : an axiomatic definition used by Coquelicot.

  ```
  Variable ( x : R). Check `|x|.
  ```

- $\mathbb{R}$ in analysis/reals.v : a realArchiType of mathcomp with a least upper bound operator.

  ```
  Variable (R : realType) ( x : R). Check `|x|.
  ```

- $\mathbb{R}$ in analysis/normedtype.v : a normed type when seen as R^0.

  ```
  Variable ( x : R^o). Check `|[x]|.
  ```

Some transports lemmas are needed.

```
Lemma absRE : forall x : R, abs x = normrr x
```