Univalent Foundations

Martín Hötzel Escardó

University of Birmingham, UK

UniMath School, Birmingham, UK, December 2017

Rosetta stone

`https://github.com/UniMath/UniMath/blob/master/USAGE.md`
Scroll down to see a notation table.

Exercises and self-study

- There are lots of hidden exercises disguised as claims in these notes.
- There is also important material which we won't have time to cover in the lecture.

Univalent mathematics in outline

1. Developed in a univalent type theory.
2. Crucially relies on identity types or type of paths between two points.
3. Certain types behave like sets (e.g. natural numbers).
4. But there are more general types (e.g. the type of groups, the type of categories).
5. Certain types behave like propositions (e.g. the empty type and the unit type.)
   - Logical connectives and quantifiers become constructions of types.
   - Proofs become constructions of elements of types.
   - Univalent logic is constructive by default,
     but compatible with non-constructive principles (excluded middle, choice).

What a univalent type theory is for

1. General mathematics performed in a certain way (univalent mathematics).

2. Synthetic homotopy theory (homotopy type theory, or HoTT).

This lecture is about *general mathematics* in univalent type theory:

(1) rather than (2).

What a univalent type theory is

As a first approximation, it is a mathematical language for expressing definitions, theorems and proofs that is invariant under isomorphism:

> If you can say something about the object X, and Y is an object isomorphic to X, then you can say the same thing about Y:
>
> *$P(X)$ and $X \simeq Y$ together imply $P(Y)$.*

A precursor

1. It may also happen for a language for mathematics that it makes impossible to distinguish isomorphic objects, even if it wasn't designed with that purpose in mind.

2. Such a language is intensional Martin-Löf type theory (MLTT).

3. For example, in MLTT we can't exhibit any property P of $\mathbb{N}$ that $\mathbb{N} \times \mathbb{N}$ hasn't. (Metatheorem.)

4. In *set theory*, they can be distinguished by the property $P(X) \equiv (0 \in X)$.

   (There isn't a global membership relation in MLTT.)

Some univalent type theories

1. A spartan MLTT + univalence axiom + resizing rules.

   (UniMath)

2. A more generous MLTT + univalence axiom + higher-inductive types.

   (Homotopy type theory book)

3. Cubical Type Theory.

   - Has univalence as a theorem.
   - Has some higher-inductive types.
   - Extends a spartan MLTT.
   - Has intrinsic computational content.

I will formulate the univalence axiom after we have developed enough material.

A spartan (intensional) Martin-Löf type theory

1. Base types $\mathbb{0}$, $\mathbb{1}$, $\mathbb{2}$, $\mathbb{N}$.
2. Cartesian products $A \times B$ and $\Pi(x : X), A(x)$.
3. Function types $X \to A$ also written A^X.
4. Sums $A + B$ and $\Sigma(x : X), A(x)$.
5. A large type $\mathcal{U}$ of small types (universe).

   Sometimes an extra large type $\mathcal{V}$ of large types with $\mathcal{U} : \mathcal{V}$.
6. An identity type $\mathsf{Id}_X(x, y)$ for any X in $\mathcal{U}$ or $\mathcal{V}$ and $x, y : X$.

Identity type discussion

1. The type $\mathsf{Id}_X(x, y)$ collects the ways in which the points $x, y : X$ are identified.
2. For some types the identity type can be defined from the other types.
3. E.g. for the type of natural numbers, it can be constructed by double induction:

$$\begin{aligned} \mathsf{Id}_{\mathbb{N}}(0, 0) &\equiv \mathbb{1}, \\ \mathsf{Id}_{\mathbb{N}}(0, n+1) &\equiv \mathbb{0}, \\ \mathsf{Id}_{\mathbb{N}}(m+1, 0) &\equiv \mathbb{0}, \\ \mathsf{Id}_{\mathbb{N}}(m+1, n+1) &\equiv \mathsf{Id}_{\mathbb{N}}(m, n). \end{aligned}$$

4. In this example, $\mathsf{Id}_X(x, y)$ is always a subsingleton.
5. In general, there may be more identifications of x and y in the identity type.
6. We often write $x =_X y$ or simply $x = y$ to denote $\mathsf{Id}_X(x, y)$.
7. We write $\equiv$ for definitions.

Universe discussion

- We assume a large type $\mathcal{U}$ whose elements are small types.
- This has many uses, e.g.:

1. Define type families as functions $X \to \mathcal{U}$, for example by induction if X is the type of natural numbers.

   (Like in the previous slide.)

2. Define properties of elements of types, e.g. $\mathsf{isEven} : \mathbb{N} \to \mathcal{U}$.

3. Define the type of groups:

   $$\mathit{Group} \equiv \Sigma(G : \mathcal{U}), \mathsf{isSet}(G) \times \Sigma(\cdot : G \times G \to G), \Sigma(e : G), (\Pi(x : G), x \cdot e = x) \times \cdots$$

   Define the type of categories. Or of topological spaces. Etc.

Two kinds of type-theoretic logic

- Curry–Howard logic.
  1. Propositions are types.
  2. Proofs are elements of types.

- Univalent logic.
  1. Propositions are subsingleton types. (As in topos logic.)
  2. Proofs are again elements of types.

- Both are constructive by default.

- But we can consistently postulate excluded middle and choice if we wish.
  (At the expense of losing implicit computational content.)

Curry–Howard propositional logic

Given two propositions (that is, types) A and B, we define

1. Conjunction. $A \wedge B \equiv A \times B$.

   "A proof of $A \wedge B$ is a pair (a, b) consisting of a proof a of A and a proof b of B."

2. Disjunction. $A \vee B \equiv A + B$.

   "A proof of $A \vee B$ is either a proof of A or a proof of B."

3. Implication. $A \to B$ is the function space, which has the same notation.

   "A proof of $A \to B$ transforms any proof of A into a proof of B."

4. Negation $\neg A \equiv A \to \mathbb{0}$.

Curry–Howard quantifiers

Given a family A of propositions (that is, types) indexed by a type X, we have:

1. Universal quantification: $\forall(x : X), A(x) \equiv \Pi(x : X), A(x)$.

   "A proof of $\forall(x : X), A(x)$ is a function that gives a proof of $A(x)$ for any given $x : X$."

2. Existential quantification: $\exists(x : X), A(x) \equiv \Sigma(x : X), A(x)$.

   "A proof of $\exists(x : X), A(x)$ is a pair (x, a) consisting of a witness $x : X$ and a proof a of $A(x)$."

Curry–Howard logic

false	$\bot$	$\mathbb{0}$	empty type
true	$\top$		any type for which a point can be exhibited
and	$\wedge$	$\times$	cartesian product of two types
or	$\vee$	$+$	disjoint sum of two types
implies	$\rightarrow$	$\rightarrow$	function space
for all	$\forall$	$\prod$	product of a type family
for some	$\exists$	$\sum$	disjoint sum of a type family
equals	$=$	Id	identity type

Martin-Löf introduced the identity type precisely to extend Curry-Howard logic with equality.

Example: there are infinitely many prime numbers

$$f : \Pi(n : \mathbb{N}), \Sigma(p : \mathbb{N}), (p > n) \times \mathrm{isPrime}(p).$$

1. A point of this type is a function f that for any $n : \mathbb{N}$ gives a prime $p > n$.

2. The type $n > p$ can be defined by induction on n and p like we defined the identity type $m = n$.

   Or in many other equivalent ways, e.g. $(n > p) \equiv \Sigma(k : \mathbb{N}), p + k + 1 = n$, after we have defined addition by induction.

3. After we define multiplication by induction, we define $\mathrm{isPrime}(p)$ in the usual way.

A lot of mathematics can be formulated and proved in this way.

Exercise

1. Define $>$ of type $\mathbb{N} \to \mathbb{N} \to \mathcal{U}$ by double induction on $\mathbb{N}$.
2. Prove that $n > p$ implies $\Sigma(k : \mathbb{N}), p + k + 1 = n$ and conversely.
3. Define isPrime.

Definition of the identity type (or path type)

- Given a type $X : \mathcal{U}$ and any two points $x, y : X$, we have a type

$$\mathsf{Id}_X\ x\ y : \mathcal{U},$$

whose elements are called paths from x to y.

We also use the following two synonyms (often with the subscript X elided):

$$\mathsf{Path}_X\ x\ y, \qquad\qquad x =_X y.$$

- This comes with a specified path $\mathsf{idp}\, x : \mathsf{Id}\ x\ x$, called identity path, for every point $x : X$.
- And with an induction principle.

Definition of the identity type (or path type)

Notice that $\mathsf{Id} : X \to X \to \mathcal{U}$ is a type family and $\mathsf{idp} : \Pi(x : X), \mathsf{Id}\ x\, x$.

- Recursion principle. Given any other type family $Id' : X \to X \to \mathcal{U}$ together with a function $\mathsf{idp}' : \Pi(x : X), \mathsf{Id}'\ x\, x$, we have a function

$$h : \Pi(x, y : X), \mathsf{Id}\ x\, y \to Id'\, x\, y$$

with

$$h\, x\, x\, (\mathsf{idp}\, x) \equiv idp'\, x.$$

This doesn't say that $\mathsf{idp}\ x$ is the only path.

What this says is that h is uniquely determined by its effect on identity paths.

(Compare to the Yoneda Lemma, if you know it, which says that a natural transformation of a certain kind is uniquely determined by its effect on identity morphisms.)

Definition of the identity type (or path type)

The recursion principle allows to define many useful things, including:

1. Path inverse: $\Pi(x, y : X), x = y \to y = x$.
2. Path composition: $\Pi(x, y, z : X), x = y \to y = z \to x = z$.
3. Map-on-path: $\Pi(f : X \to Y), \Pi(x, y : X), x = y \to f\,x = f\,y$.
4. Transport: $\Pi(A : X \to \mathcal{U}), \Pi(x, y : X), x = y \to A\,x \to A\,y$.

But the recursion principle is not enough to e.g. show that

1. Path inversion is an involution.
2. Path composition is associative with identity paths as a left- and right-neutral elements.
3. Map-on-path is functorial with respect to path identity and composition.
4. A transport along any path is invertible.

Exercise

Define path inversion, composition, map-on-path, and transport.

Definition of the identity type (or path type)

- Induction principle. If $P : \Pi(x, y), x = y \to \mathcal{U}$ is any type family and $b : \Pi(x : X), P\,x\,x\,(\mathrm{idp}\,x)$ is any function, then we have a function

$$h : \Pi(x, y : X), \Pi(p : x = y), P\,x\,y\,p$$

with

$$h\,x\,x\,(\mathrm{idp}\,x) = b\,x.$$

This says that to "prove" that P "holds" for all paths p between any two points x and y, it suffices to prove the "base case" b that it holds for all identity paths.

The recursion principle is the particular case with $P\,x\,y\,p \equiv \mathrm{Id}'\,x\,y$ that depends on the end-points of p, but not on p itself.

Exercise

Formulate and prove the above properties of inversion, composition, map-on-path, and transport.

Definition of the identity type (or path type)

Summary:

- Type former Id (also written $=$ or Path).
- Identity paths idp (also written refl in the original literature).
- Induction principle (also called J).

  To prove something for all paths between any two points, it is enough to prove it for all identity paths of all points.

Preparation: Univalence gives function extensionality (funext)

Pointwise equal functions are equal:

$$\Pi(X : \mathcal{U}), \Pi(A : X \to \mathcal{U}), \Pi(f, g : \Pi(x : X), A(x)), (\Pi(x : X), f(x) = g(x)) \to f = g.$$

- Something we don't have in intensional Martin-Löf type theories.
- We don't have univalence yet, so we assume funext when needed, for the moment.

Univalent propositions

Or *propositions*, for short (also known as *h-propositions* or *mere propositions*).

1. A type X is a proposition if any two of its elements are equal:

$$\mathsf{isProp}(X) \equiv \Pi(x, y : X), x = y.$$

   We also say that X is a subsingleton.

$$\mathsf{Prop} \equiv \Sigma(P : \mathcal{U}), \mathsf{isProp}(P).$$

2. We have $\mathsf{funext} \to \mathsf{isProp}(\mathsf{isProp}(X))$.

   Being a proposition is itself a proposition.

3. Assuming funext, also any two maps into the same proposition are equal.

Exercise

1. funext $\to$ isProp(isProp(X)).
2. Any two functions into a proposition are equal, assuming funext.

Example of a non-trivial type that is a univalent proposition

$$\Sigma(n : \mathbb{N}), \mathrm{isPrime}(n) \times (\text{n is the difference of two squares of primes}).$$

1. Although propositions are subsingletons, they are not necessarily "proof-irrelevant".
   - They have information content.
   - The number 5 can be extracted from any proof of this proposition.
2. But for the above type to really be a proposition, we additionally need:
   - propositional extensionality (any two logically equivalent propositions are equal).
   - Which is not provable in MLTT, but follows from univalence, like funext.

A subtler example (the mystery of MLTT's identity type)

Although the type $x = y$ need not be a subsingleton for $x, y : X$, the type

$$\Sigma(x : X), x = y$$

is always a subsingleton for any $y : Y$, and in fact even a singleton, or contractible:

$$\mathsf{isSingleton}(A) \equiv \Sigma(a_0 : A), \Pi(a : A), a = a_0.$$

This doesn't require propositional or function extensionality, or anything beyond MLTT.

Theorem of MLTT. $\Pi(y : X), \mathsf{isSingleton}(\Sigma(x : X), x = y)$.

Non-Theorem of MLTT. ~~$\Pi(x, y : X), \mathsf{isSubsingleton}(x = y)$~~.

Exercise

Show that $\Sigma(x : X), x = y$ is contractible by path induction.

An even subtler, crucial example (Voevodsky)

Let $f : X \to Y$ be a function of two types X and Y.

1. The type

$$\mathsf{isIsomorphism}(f) \equiv \Sigma(g : Y \to X), (g \circ f = \mathrm{id}_X) \times (f \circ g = \mathrm{id}_Y)$$

need not be a proposition.

Because of the potential presence, in MLTT, of higher-dimensional types.

2. However, the type

$$\mathsf{isEquivalence}(f) \equiv \Pi(y : Y), \mathsf{isSingleton}(\Sigma(x : X), f(x) = y)$$

always is a proposition, assuming functional extensionality.

(And the latter is a retract of the former.)

Exercise

Let X and Y be types and $f : X \to Y$.

1. Using funext, show that $\mathsf{isProp}(\mathsf{isEquivalence}(f))$ (not hard).
2. Define a map $s : \mathsf{isEquivalence}(f) \to \mathsf{isIsomorphism}(f)$ (not hard).
3. Define a map $r : \mathsf{isIsomorphism}(f) \to \mathsf{isEquivalence}(f)$ (rather hard).
4. Conclude that $r(s(\phi)) = \phi$ for any $\phi : \mathsf{isEquivalence}(f)$ using the fact that $\mathsf{isProp}(\mathsf{isEquivalence}(f))$ (direct).

Exercise

Define idtoeq. You will need to prove that the identity functioni is an equivalence.

Formulation of the univalence axiom, and consequences

1. $X \simeq Y \equiv \Sigma(f : X \to Y), \mathsf{isEquivalence}(f)$.
2. By path recursion, we have a function $\mathsf{idtoeq} : \Pi(X, Y : \mathcal{U}), X = Y \to X \simeq Y$ that maps identity paths to identity equivalences.
3. $\mathsf{UA} \equiv \Pi(X, Y : \mathcal{U}), \mathsf{isEquivalence}(\mathsf{idtoeq}\ X\ Y)$.

- Theorem of MLTT (Voevodsky). The type UA is a proposition.
- Metatheorem (Voevodsky). UA is consistent with MLTT. (Simplicial set model.)
- Theorem of MLTT+UA. $P(X)$ and $X \simeq Y$ imply $P(Y)$ for any $P : \mathcal{U} \to \mathcal{U}$.
- Theorem of spartan MLTT with two universes. The univalence axiom formulated with crude isomorphism rather than equivalence is false.

Some consequences of univalence

- Functional and propositional extensionality.

  Which we already needed.

  Univalence is a generalized extensionality axiom for intensional MLTT.

- In the type of groups, the paths are in one-to-one correspondence with group isomorphisms.

  Therefore isomorphic groups have the same properties (by transport).

Voevodsky's stratification of types by hlevels

0. Contractible types.
1. Types whose path spaces are all contractible.

   Exercise: A type is of hlevel 1 iff it is a proposition in the sense defined above.
2. Types whose path spaces are all propositions.

   These are known as sets. Between any two points, there is at most one path.
3. Types whose path spaces are all sets.

   These are known as 1-groupoids. An example is the type of groups.

   Remember that we defined a group to be a *set* together with a binary operation etc., rather than an arbitrary type.

   ⋮

n+1. Types whose path spaces are all of hlevel n.

Voevodsky's stratification of types by hlevels

- If a type is of hlevel n, then it is of hlevel m for any $m > n$.

  (Contractible types are propositions, propositions are sets, sets are (trivial) 1-groupoids, etc.)

- There is no reason why all types must have a finite hlevel.

  Paths between paths between paths between paths . . . can go on forever in a non-trivial way.

- In the absence of univalence, it is consistent that all types are sets.

Another example of when Curry–Howard goes wrong: image

Define the image of a function $f : X \to Y$ in the usual way, translated to Curry-Howard:

$$\text{image}\, f \equiv \Sigma(y : Y), \Sigma(x : X), f(x) = y.$$

- This is the type of points $y : Y$ for which we have some $x : X$ with $f(x) = y$.
- Trouble: $\text{image}\, f \simeq X$.

  This is not what we expect.
- Example. We don't expect the image of the unique function $\mathbb{2} \to \mathbb{1}$ to be isomorphic to $\mathbb{2}$.

  We expect the image to be a subtype of $\mathbb{1}$.

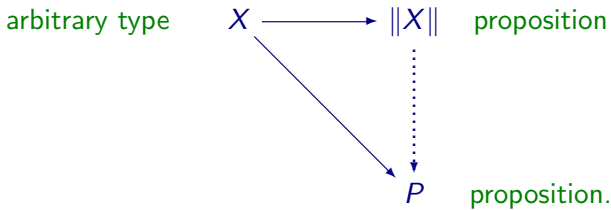
Univalent logic fixes such things in the same way as topos logic.

Exercise

Formulate and prove the troublesome claim.

Propositional truncation (or reflection)

1. A propositional truncation of a type X, if it exists, is the universal solution to the problem of mapping X to a proposition:



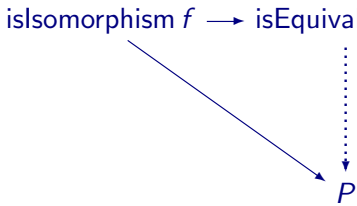
   $\|X\|$ is required to be the smallest proposition X maps into.

2. Several kinds of types can be shown to have truncations in MLTT.

3. There are a number of ways to extend MLTT to get truncations for *all* types. (Such as resizing + funext, or higher inductive types.)

Example

Not always a proposition



always proposition

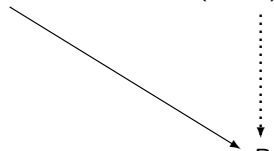
proposition.

Example

Let $f : \mathbb{N} \to \mathbb{N}$ and define

$$\Sigma^{\min}(n : \mathbb{N}), f\ n = 0 \equiv \Sigma(n : \mathbb{N}), (f\ n = 0) \times \Pi(m : \mathbb{N}), f\ m = 0 \to n \leq m.$$

Not always a proposition $\Sigma(n : \mathbb{N}), f\ n = 0 \longrightarrow \Sigma^{\min}(n : \mathbb{N}), f\ n = 0$ a proposition



proposition.

Exercise

Prove that $\Sigma^{\mathsf{min}}(n : \mathbb{N}), f\ n = 0$ is the propositional truncation of $\Sigma(n : \mathbb{N}), (f\ n = 0)$ by showing it has the right universal property. You will need funext.

Theorem (impredicative characterization of propositional truncation)

(Independently of how it is defined concretely, from the universal property alone.)

$$\underbrace{\|X\|}_{\text{small type}} \iff \underbrace{\Pi(P : \mathcal{U}), \mathsf{isProp}(P) \to (X \to P) \to P}_{\text{large type}}$$

Moreover, the rhs is a proposition assuming funext.

Univalent logic

Like Curry-Howard logic, with two differences only:

- $A \vee B \equiv \|A + B\|$.
- $\exists(x : X), A(x) \equiv \|\Sigma(x : X), A(x)\|$.

This turns out to give exactly the same disjunction and existence as topos logic and higher order intuitionistic logic.

Example concluded: univalent image

The image of a function $f : X \to Y$ is

$$\text{image}\, f \equiv \Sigma(y : Y), \|\Sigma(x : X), f(x) = y\|.$$

Theorem (impredicative characterization of univalent logic)

We get the usual intuitionistic higher-order logic, which reduces everything to *implication* $\to$ and *universal quantification* Π:

- $\bot \iff \Pi(R : Prop), R.$
- $P \wedge Q \iff \Pi(R : \mathrm{Prop}), (P \to Q \to R) \to R.$
- $P \vee Q \iff \Pi(R : \mathrm{Prop}), (P \to R) \to (Q \to R) \to R.$
- $\exists(x : X), P(x) \iff \Pi(R : \mathrm{Prop}), (\Pi(x : X), P(x) \to R) \to R.$
- $\|x = y\| \iff \Pi(P : X \to \mathrm{Prop}), P(x) \to P(y).$ (Leibniz principle.)

(Again, the lhs types are small and the rhs types are large.)

Curry–Howard "excluded middle"

Theorem of MLTT+$\|-\|$. The following are logically equivalent:

1. $\Pi(X : \mathcal{U}), X + \neg X$.
2. $\Pi(X : \mathcal{U}), \neg\neg X \to X$.
3. $\Pi(X : \mathcal{U}), \|X\| \to X$.
4. $\Pi(X : \mathcal{U}), \Sigma(f : X \to X), \Pi(x, y : X), f(x) = f(y)$.

- This is more like global choice than excluded middle.

  We can pick a point of every non-empty type.
- It implies that all types are sets, making univalent type theory trivial.
- False in the presence of two univalent universes.

Univalent excluded middle

The following are equivalent:

1. $\Pi(P : \mathcal{U}), \mathsf{isProp}(P) \to P + \neg P$.

2. $\Pi(P : \mathcal{U}), \mathsf{isProp}(P) \to \neg\neg P \to P$.

They imply $\Pi(X : \mathcal{U}), \|X\| \iff \neg\neg X$.

Which is consistent with univalent type theory.

Myth: propositional truncation erases information

It doesn't. E.g.:

Theorem of MLTT+$\|-\|$. For any $f : \mathbb{N} \to \mathbb{N}$,

$$\|\Sigma(n : \mathbb{N}), f(n) = 0\| \to \Sigma(n : \mathbb{N}), f(n) = 0.$$

If there is a root of f, we can find one.

Exercise

Prove this.

Correct formulation of unique existence

- Not $(\Sigma(x : X), A(x)) \times (\Pi(x, y : X), A(x) \times A(y) \to x = y)$.
- Instead $\mathsf{isContr}(\Sigma(x : X), A(x))$.

  Especially when formulating universal properties.
- A unique $x : X$ such that $A(x)$ is not enough.
- What is really needed is a unique *pair* (x, a) with $x : X$ and $a : A(x)$.

  Like in category theory again.

  Unless all types are sets.

Choice just holds in Curry–Howard logic

Let $X, Y : \mathcal{U}$ be types and $R : X \times Y \to \mathcal{U}$ be a relation.

$$(\Pi(x : X), \Sigma(y : Y), R(x, y)) \to \Sigma(f : X \to Y), \Pi(x : X), R(x, f(x)).$$

Moreover, the implication can be strengthened to a type equivalence.

Exercise

Prove these two claims.

However, univalent choice implies univalent excluded middle

$$(\Pi(x : X), \|\Sigma(y : Y(x)), R(x, y)\|) \to \|\Sigma(f : \Pi(x : X), Y(x)), \Pi(x : X), R(x, f(x))\|.$$

The assumptions are that isSet X and isSet $Y(x)$ for all $x : X$, and we are given $R : (\Sigma(x : X), Y(x)) \to \mathrm{Prop}$.

This form of choice is consistent with univalent type theory.

But we do get unique choice

Summary of univalent logic

1. Σ is used to express given structure or data in general.

   Cf. the type of groups.

2. Truncated Σ is used to express existence. (Still constructive.)

   - But, even better, in practice, one is encouraged to use Σ so that it produces univalent propositions without the need of truncation (if we can).
   - A crucial example is Voevodsky's primary notion of equivalence.

     (But we have seen additional examples.)

3. At the moment, it seems to be an art to decide whether particular mathematical statements should be formulated as giving structure/data or propositions.

4. But the main point is that univalent mathematics allows the distinction.

   (A distinction that is obliterated by the axiom of choice.)