# Computable analysis and verified exact real computation in Coq

Michal Konečný, Aston University, UK
Florian Steinberg, INRIA Saclay, France
Holger Thies, Kyushu University, Japan

March 23, 2021

Fourth Workshop on Mathematical Logic and its Applications
MLA 2021 (Online)

# Computable analysis in Coq

- Incone is a library for doing computable analysis in the Coq proof assistant.
- Definitions closely follow those from computable analysis.
- The internal logic of Coq is constructive.
- Axioms can be added to allow e.g. classical reasoning.
- Coq is often used for classical program verification
  Program $\rightarrow$ Specification $\rightarrow$ Correctness Proof

# Real analysis in Coq

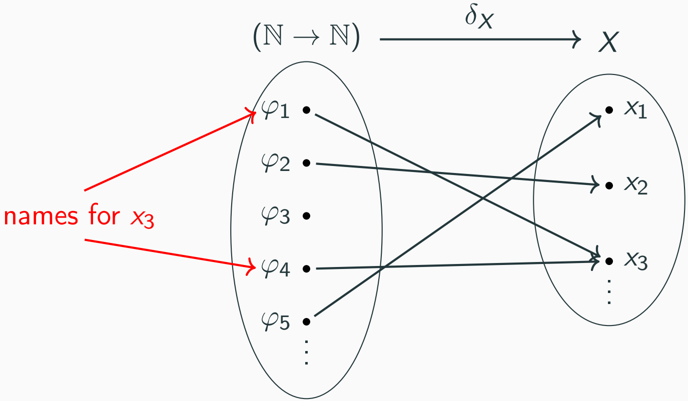
- The Coq standard library
  - Axiomatic definition of the reals
  - Not computational
  - Assumes existence of non-computable functions e.g. $up : \mathbb{R} \to \mathbb{N}$.
- CoRN
  - Completely constructive
  - Executable inside Coq

## The incone library

- The incone library formalizes results from computable analysis in Coq.
- Definitions closely follow those from computable analysis.
- Distinguishes between an algorithm and its specification.
- Correctness proofs can use classical mathematics (e.g. use the real numbers from the Coq standard library).
- Computability is not formalized and only reasoned about on the meta-level.

# Computable analysis and representations

# Representations




Represented space $\mathsf{X} := (X, \delta_X)$.

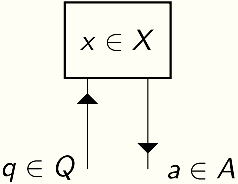
# Represented spaces

Computable analysis uses explicit encodings by natural numbers or finite binary strings.
However, here we consider more general encodings by "basic types".

## Represented space

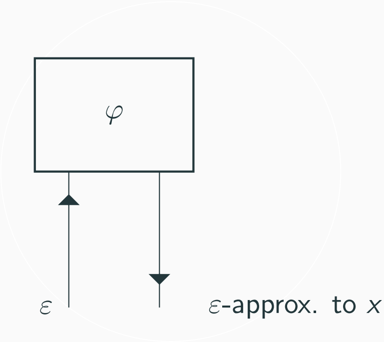
A represented space X consists of

- An abstract base type $X$.
- A space of names $\mathcal{B}_{\mathsf{X}} := Q \to A$ with questions $Q$ and answers $A$ and proofs that they are countable.
- A partial, surjective function $\delta :\subseteq \mathcal{B}_{\mathsf{X}} \to X$ called representation.



# Example: $\mathbb{R}_{\mathbb{Q}}$

A representation for the reals is given by choosing questions and answers to be $\mathbb{Q}$ and $\varphi : \mathbb{Q} \to \mathbb{Q}$ is a name for $x \in \mathbb{R}$ if $\forall \varepsilon > 0, |\varphi(\varepsilon) - x| \leq \varepsilon$.



# Cauchy Representation in Coq

Easy to define in Coq using the axiomatization of the reals in the standard library:
$\varphi$ is name for $x$: $\forall \varepsilon > 0, |x - \varphi(\varepsilon)| \leq \varepsilon$.

```
(* A name for x encodes x by rational approximations *)
Definition is_name (phi : (Q -> Q)) (x : R) :=
      forall eps, (0 < (Q2R eps)) ->
        Rabs (x - (phi eps)) <= eps.
```

Example: $\varepsilon \mapsto \varepsilon$ is a name for 0.

```
(* A name for zero *)
Lemma zero_name : (is_name (fun eps => eps) 0).
```

# Realizers

**Definition (Realizer)**

Let X, Y be represented spaces. $F :\subseteq \mathcal{B}_{\mathsf{X}} \to \mathcal{B}_{\mathsf{Y}}$ is a realizer for $f :\subseteq \mathsf{X} \to \mathsf{Y}$ if

$$\delta_{\mathsf{Y}} \circ F(\varphi) = f(x) \text{ for all } \varphi \in \delta_{\mathsf{X}}^{-1}(x) \text{ and } x \in X.$$

$$\begin{tikzcd} X \arrow[r, "f"] & Y \\ \mathcal{B}_{\mathsf{X}} \arrow[u, "\delta_{\mathsf{X}}"] \arrow[r, "F"'] & \mathcal{B}_{\mathsf{Y}} \arrow[u, "\delta_{\mathsf{Y}}"'] \end{tikzcd}$$

# Realizers for reals in coq

```
(* A realizer maps names to names *)
Definition is_realizer
         (F: (Q -> Q) -> Q -> Q) (f : R -> R) :=
           forall phi x, (is_name phi x) ->
               (is_name (F phi) (f x)).
```

- Realizers specify algorithms
- Correctness can be proven using classical mathematics

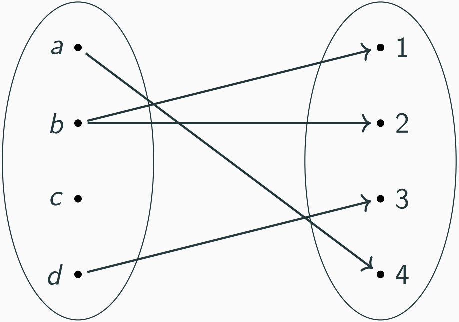
# Basic constructions on represented spaces

For represented spaces $\mathsf{X}$ and $\mathsf{Y}$ we can automatically define representations for

- The product space $\mathsf{X} \times \mathsf{Y}$.
- The co-product space $\mathsf{X} + \mathsf{Y}$.
- Infinite sequences $\mathsf{X}^\omega$.
- Spaces of subsets $A \subseteq \mathsf{X}$.
- The space of (continuous) functions $\mathsf{X} \to \mathsf{Y}$.

# Multivalued functions

A (partial) multivalued function $f :\subseteq X \rightrightarrows Y$ is just a relation or a set-valued function. The intuitive meaning is that several valid values exist.




A realizer for a multivalued function $f : X \rightrightarrows Y$ has to return a name for any $y \in f(x)$ when given a name for $x \in X$.

# Finite spaces and operations on multifunctions

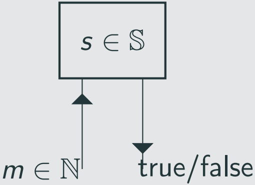
# Example: Sierpiński space

**Definition (Sierpiński space)**

Sierpiński space is the topological space with point set $\{\top, \bot\}$ and open sets $\emptyset$, $\{\top\}$ and $\{\top, \bot\}$.

**Example (Representation for Sierpiński space)**

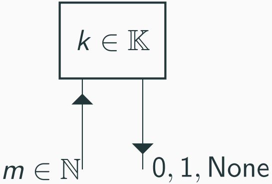
- Question space $Q_{\mathbb{S}} := \mathbb{N}$.
- Answer space $A_{\mathbb{S}} :=$ `bool`.
- $\delta_{\mathbb{S}}(\varphi) = \top \iff \exists n, \varphi(n) = \texttt{true}$.



# The Kleeneans

Kleene's three-valued logic is the logic on $\{0, 1, \perp\}$. It can be turned into a represented space as follows.

- Question space $Q_{\mathbb{K}} := \mathbb{N}$.
- Answer space $A_{\mathbb{K}} := \{0, 1, \text{None}\}$.
- Representation $\delta_{\mathbb{K}}$ such that $\delta_{\mathbb{K}}(\varphi)$ is the first value of $\varphi$ different from None or $\perp$ if no such value exists.
- Can define realizers for logical operations




The Kleeneans can be used to define a computable total real comparison $x <_{\mathbb{K}} y := \begin{cases} (x < y)_{\mathcal{B}} & \text{if } x \neq y \\ \perp_{\mathbb{K}} & \text{otherwise.} \end{cases}$

## Parallelization and Selection

For multifunctions $f : X \rightrightarrows Y$, $f' : X' \rightrightarrows Y'$ we can define $f + f' : X + X' \rightrightarrows Y + Y'$ and $f \times f' : X \times X' \rightrightarrows Y \times Y'$:

$$(f \times f')(x, x') := f(x) \times f'(x').$$

$$(f + f')(p) := \begin{cases} \operatorname{inl} f(x) & \text{if } p = \operatorname{inl} x \\ \operatorname{inr} f'(x') & \text{if } p = \operatorname{inr} x' \end{cases}$$

$f \times f'$ corresponds to running $f$ and $f'$ in parallel, $f + f'$ corresponds to selecting either $f$ or $f'$.

# Multivalued branching

Let $f, g : X \rightrightarrows Y$ and $b : X \rightrightarrows \mathbb{B}$ and consider the expression

$$\text{if } b(x) \text{ then } f(x) \text{ else } g(x).$$

Let $\mathsf{if}_{\mathsf{X}} : \mathbb{B} \times \mathsf{X} \to \mathsf{X} + \mathsf{X}$ be the function defined by

$$\mathsf{if}_{\mathsf{X}}(b, x) := \begin{cases} \mathsf{inl}\, x & \text{if } b = \mathrm{true} \\ \mathsf{inr}\, x & \text{if } b = \mathrm{false}. \end{cases}$$

Then the desired semantics can be expressed by

$$\nabla \circ (f + g) \circ \mathsf{if}_{\mathsf{X}} \circ (b \times \mathrm{id}) \circ \Delta(x).$$

# Exact real computation in Incone

# Verifying exact real computation in Incone

Typically writing and verifying an exact real computation algorithm in Incone consists of the following steps.

- Give an abstract mathematical description of the problem as a multifunction using e.g. the axiomatic real numbers from the Coq standard library.
- Define a representation for the real numbers.
- Define a (computable) function on the naming space.
- Prove that this function is a realizer for the multifunction.

## A structure for computable reals

In exact real computation, basic operations are combined to define more complicated programs. To do something similar in Coq, we define a structure computable reals containing the following basic operations

- Arithmetic operations
- The efficient limit $\lim_{\text{eff}} :\subseteq \mathbb{R}^\omega \to \mathbb{R}$, that maps any sequence $(x_i) \in \mathbb{R}^\omega$ that is efficiently Cauchy to its limit $\lim(x_i)$
- A Kleenean comparison function $<_\mathbb{K}$
- The function $\mathrm{FtoR} : \mathbb{Z} \times \mathbb{Z} \to \mathbb{R}$, $(m, e) \mapsto m \cdot 2^{-e}$
- Rational approximation $\mathrm{approx} : \mathbb{R} \times \mathbb{Q} \rightrightarrows \mathbb{Q}$

More complicated operations can be defined as composition of basic operations independently of the underlying representation.

# Interval Reals

Computing with rationals is not very efficient.
Alternative: approximate real numbers by intervals with dyadic endpoints (numbers of the form $m \cdot 2^e$).

### Definition

Let $\mathbb{ID}$ be the set of intervals with dyadic endpoints. A representation $\mathbb{R}_{\mathbb{ID}}$ of the reals is given by $Q_{\mathbb{ID}} = \mathbb{N}$, $A_{\mathbb{ID}} = \mathbb{ID}$.

$$\delta_{\mathbb{R}_{\mathbb{ID}}}((I_n)_{n\in\mathbb{N}}) = x \quad \iff \quad x \in \bigcap_{n\in\mathbb{N}} I_n \text{ and } \lim_{n\to\infty} |I_n| = 0.$$

The Coq Interval Library already provides many operations on intervals.

# Interval arithmetic in Coq

Correctness in interval arithmetic:

```
Lemma add_correct prec x y I J:
         x \contained_in I -> y \contained_in J ->
   (x + y) \contained_in (I.add prec I J).
```

To define a realizer we additionally need absolute error bounds to show that intervals get arbitrarily small:

```
Lemma add_error I J n m p x y N:
  diam I <= /2^n ->  diam J <= /2^m ->
  (x \contained_in I) -> (y \contained_in J) ->
  (Rabs x) <=  (2 ^ N) -> (Rabs y) <= (2 ^ N)
  -> diam (I.add p I J) <= /2 ^ n + /2 ^ m +
                          (2 ^ (N+5-p)).
```

# Square root

We give an efficient implementation of $\sqrt{x}$ for $x \in [0, \infty)$.
For this, we define a function $\texttt{sqrt\_approx} : \mathbb{R}^\omega \to \mathbb{R}$ with

$$\left|\texttt{sqrt\_approx}(x, n) - \sqrt{x}\right| \leq 2^{-n}$$

using only operations from the computable reals structure. Then

$$\texttt{sqrt} := \texttt{lim\_eff} \circ \texttt{sqrt\_approx}.$$

For $x \in [0.25, 2]$ the Heron iteration defined by

$$x_0 := 1$$
$$x_{n+1} := \frac{1}{2}\left(x_n + \frac{x}{x_n}\right)$$

converges quadratically, i.e.

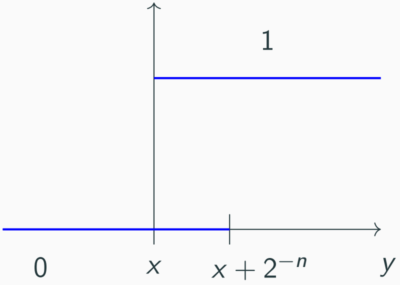
$$\texttt{sqrt\_approx}_{|[0.25,2]}(x, n) = x_{\log_2 n}$$

and then extend to all non-negative reals.

# Soft Comparison

Soft-comparison $sc\colon \mathbb{R} \times \mathbb{R} \times \mathbb{N} \rightrightarrows \mathbb{B}$ is defined by

$\mathrm{true} \in \mathrm{sc}(n, x, y) \Leftrightarrow x < y$ and $\mathrm{false} \in \mathrm{sc}(n, x, y) \Leftrightarrow y < x{+}2^{-n}$.

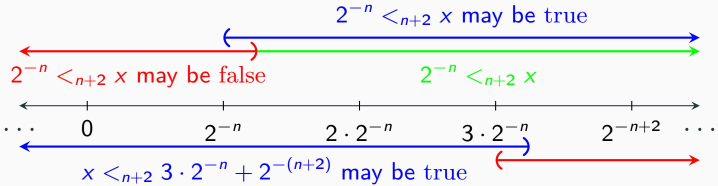



$\mathrm{sc}(n, x, y)$ plotted over $y$ for fixed $x$ and $n$.

# Magnitude and Scaling

We define a multifunction $\texttt{magnitude}: \mathbb{R} \rightrightarrows \mathbb{Z}$ such that

$$z \in \texttt{magnitude}(x) \Leftrightarrow 2^z < x < 2^{z+2}.$$



# Code extraction

For faster execution, Coq can extract Haskell or Ocaml code from proofs and definitions.
The basic extraction mainly performs a straightforward syntactic translation.
It can be improved by telling Coq how to extract:

```
Extract Inlined Constant Z.abs => "(Prelude.abs)".
Extract Inlined Constant Z.geb => "(Prelude.>=)".
Extract Inlined Constant Z.opp => "(Prelude.negate)".
Extract Inlined Constant Z.succ => "(Prelude.succ)".
Extract Inlined Constant Z.pow_pos => "(Prelude.^)".
Extract Inlined Constant Z.pow => "(Prelude.^)".
```

# Conclusion and Future work

- The incone library can be used to implement real number computations in Coq and do proofs in the style of computable analysis.
- Our simple implementation is already quite efficient.
- More complicated spaces and operators, e.g., analytic functions and ODE solving.

Thank you!