A Formal Verification for Kantorovitch's Theorem

Ioana Pasca

A Formal Verification for Kantorovitch's Theorem

Ioana Paşca [1]

1: INRIA Sophia Antipolis
Ioana.Pasca@sophia.inria.fr

Abstract

Kantorovitch's theorem gives sufficient conditions for the convergence of Newton's method. We present a full formalization of this theorem in the case of a real function. The work is accomplished inside the Coq proof assistant and it is based on the `Reals` library provided by the theorem prover. For the general case of the theorem we first describe a way to represent concepts from multivariate analysis, as Coq does not offer such a library. We then discuss the proof of Kantorovitch's theorem based on this representation.

1. Introduction

The main purpose of formal methods is to guarantee that software satisfies its formal specification. We are interested in extending such techniques to areas of computer science that rely on numerical methods. This is of interest because of the safety critical domains that rely on these methods like on-board software for planes, trains, real time programs used in medicine etc. Formal verifications of numerical methods are rare. We can mention the work of Micaela Mayero [19] in which real analysis concepts are formalized in order to prove the correctness of a differentiation algorithm.

At a slightly different level, we are interested in having mathematical concepts formalized inside proof assistants. In the long run, the aim would be to use them for extensive mathematical developments. At present, computer algebra systems like Maple or Matlab are more widely used than proof assistants, for representing mathematical concepts on machines. However there are cases where the level of accuracy provided by these systems is not sufficient and we would like to have a certification that the computation is indeed correct [16, 5].

In this context, we are interested in Newton's process, which approximates the root of a given function. Kantorovitch's theorem gives sufficient conditions for the convergence of this process.

A formal verification of such a theorem is of interest for computer scientists who use numerical methods in their developments. Newton's method is widely used because of its quadratic convergence rate.

Applying formal methods to a domain means encoding the concepts of that domain inside a proof assistant and verifying the desired properties within this setting. It is thus obvious that the success of formal verification for numerical methods depends on having an extensive and practical formalization of real analysis concepts inside the proof assistant. For the formal proof of Kantorovitch's theorem, we organized our work by starting with the case of a single variate real function and then generalizing for several dimensions. In section 2 we provide a survey of formal real analysis concepts in various theorem provers. We also present an outline of the mathematical proof of Kantorovitch's theorem. In section 3, we present the proof of the theorem in one dimension. Section 4 describes the multivariate analysis concepts we formalized and section 5 discusses the proof of the theorem in $\mathbb{R}^p$. The conclusions and possible extensions of this work are detailed in section 6.

2. Survey of related work

Mathematical proofs cannot be entirely discovered by mechanical means, but the correctness of a formal proof can be verified by a machine. Work in this domain has uncovered several approaches to representing the proofs and each of these approaches has led to a different system. Some of the most important proof systems at present are HOL [13], PVS [25], ACL2 [17], Isabelle[23] and Coq [2, 1]. Each of them offers a library of formalized mathematical concepts and verified theorems.

2.1. Formalizing real analysis concepts

As stated before we are interested in the way real analysis concepts are encoded to have a representation that is both feasible and corresponds to the mathematical concept being formalized. We are also interested in seeing what sort of theorems and results have been obtained using these concepts.

The first thing is to find a way to represent the real numbers. They can either be given an axiomatic definition as a complete ordered field satisfying the least upper bound principle or they can be defined constructively and proved the right properties. There are several constructions for the reals, the most famous being the Dedekind model, based on the notion of cut, the Cantor model, which uses Cauchy sequences of rational numbers and the Weierstrass model which uses decimal fractions.

The next step is to see how real analysis concepts can be efficiently formalized inside the proof assistant. There are two main approaches. One is to follow classical analysis where concepts are expressed using the usual $\varepsilon - \delta$ definitions. The other is to use non-standard analysis as first introduced by Robinson [24].

Non-standard analysis works with ideally small and ideally large numbers, so it formalizes the concepts of infinite and infinitesimal. It introduces numbers systems such as hyper-reals and the infinitely close relation, which help express concepts such as limit, continuity etc. The main argument in favor of using non-standard analysis is that, by using this "infinitely close relation", the manipulation of objects (e.g. derivatives, limits of functions) becomes algebraic and therefore theorems are easier to automate.

For HOL, the main work can be found in [14]. The real numbers are constructed using an adaptation of Cantor's method. Real analysis is dealt with in a classical way. One interesting fact is the way limits are implemented. As opposed to other systems, which treat independently the limit of a sequence and that of a function, HOL describes them using one concept by relying on the theory of nets. Results are achieved around continuity, differentiation, integrability and transcendental functions. We mention that a quantifier elimination procedure has also been implemented for this theory. Some applications of the developed theory involve verifications for floating point algorithms.

In Isabelle one can actually find most of the concepts formalized in both classical and non-standard analysis. What is interesting is the proof of equivalence between the concepts in the two approaches[8].

The ACL2 proof assistant has a non-standard approach to real analysis. [9] offers an introduction to non-standard analysis techniques and shows how they can be used to reason mechanically about concepts like transcendental functions. The formalization of continuity, differentiability etc. allows proving theorems such as intermediate value theorem and Rolle's theorem.

[6] presents an implementation of basic real analysis building on the axiomatic definition of the reals in the PVS theorem prover. This implementation includes definitions of convergence, continuity, differentiability of real-valued functions and proofs for theorems around this concepts (e.g. the mean value theorem).

In Coq, two approaches have been explored. The Coq Standard Library called `Reals` provides an axiomatic definition of the reals. The library contains a lot of results for real analysis: sequences and series, transcendental function, concepts of limit, continuity, differentiation, integration, calculus

theorems like the mean value theorem,the fundamental theorem of calculus etc.

There also exists a constructive formalization of real analysis in Coq. In C-CoRN (Coq Constructive Repository at Nijmegen [4]) the reals are build as a Cauchy completion of the rationals [22]. Among the most important work in this setting we can mention the constructive formalization of the fundamental theorem of algebra [10] and the fundamental theorem of calculus [3].

Closely related to our work, we can mention the existence in C-CoRN of a formalization for a root finding algorithm. The result is part of a larger project that dealt with the formalization of the Fundamental Theorem of Algebra [10] using a proof given by Kneser.

Our work was done using the Coq Proof Assistant and the Standard Library, i.e. the axiomatic, classical definition for real analysis concepts. For the second part of our work we also used the `ssreflect` extension and its libraries for reasons to be discussed in Section 4.

2.2. Mathematical proof of Kantorovitch's theorem

As stated above, we are interested in having sufficient conditions for the convergence of Newton's process in the case of an equation system. The problem was studied by Willers, Sténine, Ostrowski, Kantorovitch and others. In what follows, we present a special case of Kantorovitch's theorem that expresses the convergence of Newton's method for a finite system of non-linear equations. It establishes the existence and uniqueness of a solution for the initial equation system.

The statement of the theorem according to [7] is as follows:

Theorem 1 *Consider a system of non-linear algebraic or transcendent equations $f(x) = 0$, where the vector function $f : \mathbb{R}^p \to \mathbb{R}^p$ has continuous first and second partial derivatives in a certain domain ω, i.e. $f(x) \in C^{(2)}(\omega)$. Let $x^{(0)}$ be a point contained in ω with its closed ε-neighborhood $\overline{U_\varepsilon}(x^{(0)}) = \{\|x - x^{(0)}\| \leq \varepsilon\}$ also included in ω. If the following conditions hold:*

1. *the Jacobian matrix $W(x) = [\frac{\partial f_i(x)}{\partial x_j}]$ has an inverse for $x = x^{(0)}$, $\Gamma_0 = W^{-1}(x^{(0)})$ with $\|\Gamma_0\| \leq A_0$;*
2. *$\|\Gamma_0 f(x^{(0)})\| \leq B_0 \leq \frac{\varepsilon}{2}$;*
3. *$\sum_{k=1}^{p} |\frac{\partial^2 f_i(x)}{\partial x_j \partial x_k}| \leq C$ pour $i, j = 1, 2, ..., p$ and $x \in \overline{U_\varepsilon}(x^{(0)})$;*
4. *the constants A_0, B_0, C satisfy the inequality $2pA_0B_0C \leq 1$.*

then, for the initial approximation $x^{(0)}$, the Newton process

$$x^{(n+1)} = x^{(n)} - W^{-1}(x^{(n)}) f(x^{(n)}) \tag{1}$$

$(n = 1, 2, ...)$ converges and the limit vector $x^* = \lim_{n\to\infty} x^{(n)}$ is a solution of the initial system, so that $\|x^* - x^{(0)}\| \leq 2B_0 \leq \varepsilon$. Moreover, in the domain $\{\|x - x^{(0)}\| \leq 2B_0\}$ the solution is unique.

Here is the outline of the proof for the theorem as presented in [7]:

- prove a collection of properties for each element of the Newton sequence;
- infer that it is a Cauchy sequence;
- use the completeness of $\mathbb{R}^p$ to prove the convergence;
- prove that the limit of the sequence is a root of the given function;
- prove that in a certain interval the root is unique.

For the first part, the properties verified for each element of the sequence are similar to those for $x^{(0)}$. Resoning by induction we get the following:

$$W(x^{(n)}) \text{ is invertible}$$
$$\|W^{-1}(x^{(n)})\| \leq A_n \tag{2}$$
$$\|W^{-1}(x^{(n)})f(x^{(n)})\| \leq B_n \leq \frac{\varepsilon}{2^{n+1}} \tag{3}$$
$$2pA_nB_nC \leq 1$$

where

$$A_n = 2A_{n-1}$$
$$B_n = pA_{n-1}B_{n-1}^2C$$

We are able to establish:

$$\overline{U}_{\varepsilon}(x^{(0)}) \supset \overline{U}_{\frac{\varepsilon}{2}}(x^{(1)}) \supset \ldots \supset \overline{U}_{\frac{\varepsilon}{2^n}}(x^{(n)}) \supset \ldots \tag{4}$$

From (4) we can infer that $x^{(n)}$ is a Cauchy sequence:

$$x^{(n+m)} \in \overline{U}_{\frac{\varepsilon}{2^n}}(x^{(n)}) \Rightarrow \|x^{(n+m)} - x^{(n)}\| \leq \frac{\varepsilon}{2^n}$$

The latter quantity can be made arbitrary small for $n > N$ and $m \in \mathbb{N}$, which is equivalent to Cauchy's criterion. We use the result that $\mathbb{R}^p$ is a complete metric space to deduce that the sequence converges. By taking the limit in (1) we get that the limit of the sequence is a root of function f.

To prove the uniqueness of the solution, we suppose that there exists another solution of the equation and prove that it is also the limit of the sequence. By uniqueness of this limit we have the desired result.

The reasoning steps behind (2) or (3) use non trivial results from matrix theory and differential calculus in several dimensions (e.g. Taylor's formula) as we shall point out again later on.

3. Proof in one dimension

3.1. Informal presentation

Kantorovitch's theorem gives the conditions of convergence in the general, multidimensional case. Naturally, it can be reduced to the unidimensional case. This entails some simplifications in the conditions imposed as well as in the proof. This adapted proof of Kantorovitch's theorem will later serve as a guideline in the general proof.

We also made an adjustment in the statement of the theorem by loosening the constraint that the function should be twice derivable. We just impose a bounding condition on the variation of the first derivative, which is trivially verified if the function does have a continuous second derivative (condition 3 in Theorem 2).

The modified statement of the theorem is as follows:

Theorem 2 *Consider an equation $f(x) = 0$, where $f : [a, b] \to \mathbb{R}$, a, $b \in \mathbb{R}$ $f(x) \in C^{(1)}([a, b])$. Let $x^{(0)}$ be a point contained in $[a, b]$ with its closed ε-neighbourhood $\overline{U_{\varepsilon}}(x^{(0)}) = \{|x - x^{(0)}| \leq \varepsilon\} \subset [a, b]$. If the following conditions hold:*

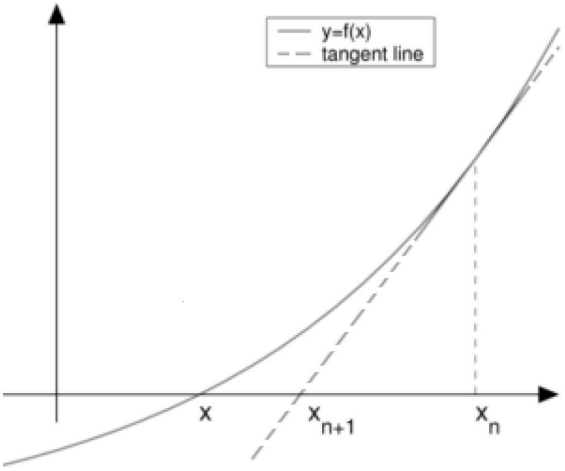

Figure 1: Newton's method

1. $f'(x^{(0)}) \neq 0$ *and* $|\frac{1}{f'(x^{(0)})}| \leq A_0$;
2. $|\frac{f(x^{(0)})}{f'(x^{(0)})}| \leq B_0 \leq \frac{\varepsilon}{2}$;
3. $\forall x, y \in [a, b], |f'(x) - f'(y)| \leq C|x - y|$
4. *the constants* A_0, B_0, C *satisfy the inequality* $2A_0B_0C \leq 1$.

then, for an initial approximation $x^{(0)}$*, the Newton process*

$$x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})} \tag{5}$$

$(n = 1, 2, ...)$ *converges and the limit vector* $x^* = \lim_{n\to\infty} x^{(n)}$ *is a solution of the initial system, so that* $|x^* - x^{(0)}| \leq 2B_0 \leq \varepsilon$. *Moreover, in the domain* $\{|x - x^{(0)}| \leq 2B_0\}$ *the solution is unique.*

The intuition behind these conditions can be more easily understood by analysing Figure 1. The conditions say that if the slope of the function is sufficiently steep and if the variation of this slope (i.e. the first derivative) is bounded sufficiently tight, then the successive approximations will get closer to the root each time, moreover, the process will converge to the root.

3.2. The formal proof

As stated before, Coq contains a library with results about real numbers. The formalization of reals in this standard library is axiomatic and done in classical logic. The library contains most of the results needed in the proof. The work accomplished was mainly conducted in two directions: providing results not found in the standard library `Reals` and proving the structure of the theorem. During this work, we also tried to understand where difficulties arise in such a development and what could be done to facilitate such proofs. An interesting issue was working with derivatives.

3.2.1. Derivation

Let $f : \mathbb{R} \to \mathbb{R}$ be a derivable function on $[a, b]$. "On paper" we can write the corresponding Newton's sequence $x^{(n+1)} = x^{(n)} - \frac{f(x^{(n)})}{f'(x^{(n)})}$, without worrying whether the term $x^{(n)}$ is in the interval $[a, b]$ (so

that we know the derivative actually exists). However, the formalization of derivatives in Coq requires that when one talks about the derivative of a function in a point, one has to provide a term that states that the function is actually derivable at that point. The goal is to ensure that derivatives are properly used. Nevertheless, this is an inconvenience when manipulating derivatives.

The way we worked around this particular impediment is by defining a total function to use in the definition of the sequence and then say that on interval $[a, b]$ this function is equal to the derivative of f. This enables us to define our concepts and at the same time prevents us from using properties of the derivative in a point before proving the function is derivable there.

3.2.2. Higher order derivatives

Another issue that for the moment remains unresolved is that of higher order derivatives. This is one of the reasons for which we chose to generalize the statement of the theorem. We did not want to talk about the second derivative because at this moment in Coq it can be referred to only as the derivative of the first derivative. This renders its use rather tedious. We would prefer having a general mechanism for dealing with derivatives.

3.2.3. Statement of the theorem

For the actual proof of the theorem we were able to follow rather accurately the "proof on paper". The Coq proof assistant possesses some tactics that help automatize steps in the proof dealing with equality relations (e.g `ring`, `field`), with inequalities on the real numbers (`fourier`) or with differentiation rules (`reg`). They made a big difference in the amount of work that had to be accomplished.

The statements of the main lemmas are given bellow. In the code, the `Variable` declarations describe objects that are assumed to exist, the `Hypothesis` declarations describe properties that are assumed to hold and the `Fixpoint` declaration defines the Newton sequence as a recursive function.

```
Variables a b:R. (*the ends of the interval*)
Variables f f':R→R. (*the function and its derivative*)
Variables A0 B0 C:R. (*the constants from the theorem*)
Variable X0:R. (*the initial approximation*)
Fixpoint Xn (n:nat): R := (*the Newton sequence*)
match n with
| 0 => X0
|S n => (Xn n) - (f (Xn n))/(f' (Xn n))
end.

(*the hypothesis on the function and the constants*)
Hypothesis pr: ∀ t:R, c_I a b t → derivable_pt f t.
Hypothesis Hder_f: ∀ (t:R)(H:c_I a b t), derive_pt f t (pr t H)=f' t.
Hypothesis Hcont_f':∀ y, (c_I a b y) → continuity_pt f' y.
Hypothesis Hlim_f':∀ y1 y2:R, c_I a b y1 → c_I a b y2 →
      Rabs (f' y1 - f' y2) ≤ c * (Rabs (y1-y2)).
Hypothesis Hincl:included (c_disc X0 eps) (c_I a b).
Hypothesis Hdif_f':f' X0 ≠ 0 .
Hypothesis Habs_a: Rabs (/(f' X0)) ≤ A0.
Hypothesis Habs_b:Rabs (f X0/(f' X0)) ≤ B0 .
Hypothesis A0_b0_c: 2*A0*B0*C≤1.

(*the theorem stating the existence*)
Theorem kantoro_exist_b:
∃ xs:R, Un_cv Xn xs ∧ c_disc X0 (2*b0) xs ∧ f xs = 0.
```

```
(*uniqueness of the solution*)
Theorem kantoro_unic:
∀ xs2:R, c_disc X0 (2*b0) xs2→ f xs2 = 0 → Un_cv Xn xs2.
```

To clarify the statements above, we mention that $c\_I\ a\ b$ and $c\_disc\ X0\ eps$ denote the closed interval $[a, b]$ and the closed disc centered in $X0$ and of radius esp, respectively. Hypothesis pr says that function f is derivable on interval $[a, b]$, hypothesis $Hder\_f$ states that f' is equal to the derivative of f on the same interval. The continuity and the bounding condition required for the derivative are formalized by $Hcont\_f'$ and $Hlim\_f'$.

The proof is done in classical logic, but we believe that all reasoning used can be made constructive. The development has around 1800 lines and can be found on Internet [1].

4. Multivariate analysis

The main difficulty was to generalize the work done for the real function case to several dimensions.

Mathematically, the elements of $\mathbb{R}^p$ are vectors of length p of real elements. Operations like addition of two vectors (+) and multiplication of a vector by a scalar (*) are defined component wise, using the operations on the real numbers so that $(\mathbb{R}^p, +, *)$ is a real vector space. One can further define various norms on these vectors. An important issue is working with functions $f : \mathbb{R}^p \to \mathbb{R}^p$ and being able to express concepts such as continuity and partial derivatives of such functions. An important amount of work concerns matrices, as some differentiation concepts are expressed using them (e.g. the Jacobian matrix of a function).

4.1. How to work in higher dimensions

4.1.1. Related work

The only proof assistant that contains a formalization of analysis in several dimensions is HOL Light. Among the concepts that can be found in the "Multivariate" library we can mention vectors, matrices, determinants, topology concepts for euclidean spaces, convexity, continuity, differentiability, integration. A documentation of this library can be found in [15]. It discusses the techniques that could be used for expressing concepts in multivariate calculus.

The strategy chosen is to implement vectors as functions $N \to real$, where N is a type with finite cardinality and $real$ is the type of real numbers in HOL. Vectors are of type $(N)finite\_image \to real$, where $(N)finite\_image$ has the same size as N when N is a finite type and 1 otherwise. An indexing operator is defined to use natural numbers as indexes.

Within this formalization, basic operations on vectors are easily defined. The representation of other definitions and properties is described. There are concepts from linear algebra: operators, matrices, determinants; topology: open, closed, compact, convex sets, sequences, continuity, differentiability; basic calculus theorems: mean value theorem, inverse function theorem.

4.1.2. Approaches for Coq

The main challenge is to find an appropriate representation for real vectors inside the Coq proof assistant. It is worth trying to find new solutions and not just copying the HOL version since Coq has a richer type system that supports dependent types and therefore offers more possibilities.

Taking into consideration some key issues of the concepts we need to formalize and the features of our theorem prover, we took a closer look at four approaches that seemed good candidates.

[1] http://www-sop.inria.fr/marelle/Ioana.Pasca/kantoro.v (the code works with version 8.1pl2 of Coq)

1. Elements of $\mathbb{R}^p$ can be viewed as functions of type $nat \to R$, that take as argument a natural number and return the real corresponding to that position in the vector. Then we say we take into consideration only the first p components (i.e. corresponding to the natural numbers smaller than p). Working with such an approach provides a rather natural way of handling the objects. For example, the norm is just a recursive function.

   The proof technique mostly used is induction on natural numbers.

   The disadvantage is that, in fact, our functions represent infinite vectors. It is difficult to state equality between such objects.
2. Think of vectors as a function $bound\ p \to R$, where $bound\ p$ is a type of dependent pairs formed from a natural number i and a proof that i is smaller than p. This representation raises problems even at early stages of development. For example, if one tries to define a function that takes as argument a vector and returns its maximum element, the first attempt would be to write it as a recursive function: the maximum between the i^{th} element and the maximum of the vector composed of the first $i-1$ elements. The problem here is that when talking about "vectors of $i-1$ elements" they are a different type than those with i elements and they must be constructed from "vectors of i elements" as must be the proof that the natural number passed as argument is smaller than $i-1$.
3. The third approach is to use the notions on vectors and matrices already developed by Nicolas Magaud [18], where the vectors are implemented as dependent lists. Therefore, $vect\ R\ p$ represents a list having p elements of type R.

   Recursion principles are defined for reasoning on this structure. Also, basic operations such as addition, multiplication by a scalar, scalar product are defined. Matrices are implemented as vectors of vectors and ring structure properties are proved. One of the reasons for which proofs are not easy in this implementation is because even basic functions, like vector addition, require complicated forms of dependent recursion.
4. Another approach is "borrowed" from a development based on Coq with the `ssreflect` extension and library [11, 12]. Among other things, it provides formalizations for various finite types and their properties. A finite type is a record containing a sort, a list of all elements of the finite type and a predicate saying that this list doesn't contain duplicates. We used the finite type $ordinal\ p$ which represents the set of natural numbers smaller than p. The implementation of finite types allows us to interpret this subset of natural numbers both as a type and as a list of elements. The real vectors have type $ordinal\ p \to R$.

   The most important facility offered by this approach is a dual vision of the vectors, more precisely, of the set that indexes the elements of a vector. It combines the views of the previous approaches. So we have a simple way of viewing the vectors as functions (as in the first or second approach), we can define their length (p) and we have a way of reasoning on the data structure: induction on the structure of the list (like in the third approach)enumerating the elements of $ordinal\ p$. Because vectors are just functions, one has easy access at their elements, just by providing the index and equality between two vectors can be stated by assuming extensionality.

Conclusion. We tested and compared the four approaches by defining the basic concepts in each of them. We formalized vectors, matrices, norms, functions and tried to do simple proofs around them so we would be able to understand the advantages and difficulties that occur. We took into consideration how easy it is to work with the concepts and how accurate they represent the mathematical objects and we decided to continue the formalization in the lines of the last approach proposed.

4.2. The formalized concepts

For our work in the multidimensional case we are in the following settings: Coq with the `ssreflect` extension and library, `Reals` library (which implies the use of classical logic), the extensionality

property and the axiom of choice. We have explained above our decision to use `ssreflect` and the libraries developed on it. The main reason for assuming general extensionality is to be able to state equality between two vectors or matrices. The classical logic settings are imposed by the use of the `Reals` library and it allows us to prove more results than in an intuitionistic setting. The axiom of choice has also proved necessary and we shall provide examples later on in this section to motivate its use.

The description of the mathematical concepts follows [20, 21, 7].

Before beginning to describe how concepts are represented, we would like to present in more detail the features of the library developed within the Mathematical Components project [11]. Their work offers descriptions for finite sets. We are interested in *ordinal p* which describes the set $\{0, 1, \ldots, p-1\}$. The type of the finite set *ordinal p* contains an enumeration of all the elements in the set. This enumeration is viewed as a list. As a consequence, we have a means for reasoning on such a data structure, i.e. induction on the structure of the list. Other features of this development will prove useful in our work as we will discuss in this section.

We begin by defining our set of indexes.

```
Definition I := ordinal p.
```

Under the hypothesis $0 < p$, the elements of $\mathbb{R}^p$ would be $I \to R$, i.e. functions from the set of indexes to the reals. This corresponds to the familiar way of viewing vectors as $(x_0, x_1, \ldots, x_{p-1})$, where $x_i \in \mathbb{R}, \forall i \in \{0, 1, \ldots, p-1\}$.

We define the operations on our vectors like addition, subtraction or multiplication by a scalar component wise.

```
Definition add_v (v1 v2:I→R):I→R := (fun i:I => v1 i + v2 i).
Definition dif_v (v1 v2:I→R):I→R := (fun i:I => v1 i - v2 i).
Definition mult_sv (a:R) (v:I→R):I→R := (fun i:I => a * v i).
```

Properties of these operations are proved by simply reducing them to properties on the real numbers. For the latter we benefit from tactics like `ring, field` and `fourier` provided by Coq which automatically solve a large variety of equalities and inequalities on the reals.

Here is a simple example that states the distributivity of scalar multiplication over addition:

```
Lemma mult_add:
∀ (a:R) (v1 v2 :I→R),
mult_sv a (add_v v1 v2) = add_v (mult_sv a v1) (mult_sv a v2).
Proof.
move=> a v1 v2.
rewrite -ext_eq /add_v /mult_sv =>i; ring.
Qed.
```

The tactics are part of the `ssreflect` extension. The first line of the proof just moves the variables a, v_1, v_2 from the goal to the context. The second line says that we want to prove our equality by resorting to the extensionality property (i.e. two vectors are equal iff each of their components for the same index are equal). After that, just rewriting the definition of addition and multiplication by a scalar is enough to obtain an equality between two real numbers. This latter equality is dealt with by Coq's tactic `ring`.

For the norm on vectors we chose the following:

$$\|x\| = \max_i |x_i|$$

to respect the conventions in [7]. Nevertheless, the structure of the proof does not depend on the particular norm used and it can be adapted to any other.

In formalizing the norm in Coq, we used once more the facilities provided by the `ssreflect` libraries. Within this development we can find the definition and proofs of various properties for folding a function over a finite set by using the generic function *iprod*. In our case, we fold the function *Rmax* (that gives the maximum of two real numbers) over our set of indexes I in order to retrieve the maximum value of a vector. The absolute value is easily computed component wise. In Coq, the definitions look like this:

```
Definition Rabs_v (v:I→R) (i:I):R := Rabs(v i).
Definition norm (v:I→R):R := iprod R Rmax 0 I (setA I) (Rabs_v v).
```

Proving the good properties for the norm like positive homogeneity, triangle inequality and positive definiteness is done by using properties already proven for *iprod* and induction on the structure of the list of indexes.

We can now define the distance corresponding to our norm:

```
Definition dist_Rp (u v:I→R) :R := norm (dif_v u v).
```

The properties for the distance follow naturally from those of the norm to ensure that, in our representation, $\mathbb{R}^p$, equipped with the above defined distance, is a metric space.

We now define the type of vector sequences as $nat \to I \to R$. The concepts of convergence and Cauchy criterion are formalized in the terms of our distance:

```
Definition conv_Rp (un:nat→I→R) (l:I→R) : Prop :=
   ∀ eps:R, 0 < eps→ ∃ N : nat,
      (∀ n:nat, (N ≤ n)%nat → dist_Rp (un n) l < eps).
Definition Cauchy_crit_Rp (vn:nat→I→R): Prop :=
   ∀ eps:R, 0 < eps → ∃ N : nat, (∀ n m:nat,
      (N ≤ n)%nat → (N ≤ m)%nat → dist_Rp (vn n) (vn m) < eps).
```

We proved a collection of results: the uniqueness of the limit of a sequence, any convergent sequence satisfies Cauchy's criterion, $\mathbb{R}^p$ is a complete metric space (i.e. any sequence satisfying Cauchy's criterion is convergent), the convergence in $\mathbb{R}^p$ is a convergence on components. This is, for example, a place where the axiom of choice turned out to be indispensable.

We then formalized functions $\mathbb{R}^p \to \mathbb{R}^p$ as $(I \to R) \to I \to R$. Operations on functions (addition, multiplication by a scalar etc.) are defined component wise, as we did for vectors.

Having the metric space structure enables us to express that the limit of a function f in a point $v0$ of this space is l:

```
Definition limit_Rp (f:(I→R)→I→R) (v0:I→R) (l:I→R):=
   ∀ eps:R, eps > 0→ ∃ alp:R, alp > 0 ∧
     (∀ v:I→R, 0 < dist_Rp v v0 < alp → dist_Rp (f v) l < eps).
```

To verify the feasibility of the approach we did some proofs around these concepts. For example: if there exists a limit of a function in a point then this limit is unique; the limit of the sum of two functions is the sum of the limits etc. We were rather pleased with the fact that our proofs could be done by following the "proof on paper" and the structure of their equivalent in the unidimensional case.

We can define functions continuous at a point by saying the limit at that point is the value of the function:

```
Definition cont_Rp (f:(I→R)→I→R) (v0:I→R):=
   limit_Rp f v0 (f v0).
```

In order to be able to consider partially derivable functions and their representation as the Jacobian matrix, we first have to introduce concepts on square matrices. They are of type $(I \to I \to R)$. Addition, subtraction and multiplication by a scalar are defined component wise. However, multiplication of a matrix and a vector and of two matrices are trickier to define. Having a square matrix of size p, $M = [m_{ij}]$ and a vector of size p, $v = (v_i)$, their product is $M * v = (\sum_{j=0}^{p-1} m_{ij} * v_j)$. We use a function to compute each term and another one to get the sum:

```
Definition mult1 (m:I→I→R) (v:I→R) (i j :I): R := m i j * v j.
Definition mult_mv (m:I→I→R) (v:I→R) (i:I): R :=
   iprod R Rplus 0 I (setA I) (mult1 m v i).
```

We define the multiplication of two matrices similarly.

Properties of operations on matrices are not difficult to verify once some needed results are obtained for the summation operator.

The norm on matrices is the following:

$$\|M\| = \max_i \sum_j |m_{ij}|$$

Among the properties we were able to prove is the following inequality

$$\|Mv\| \leq \|M\|\|v\|$$

where M is a matrix and v a vector.

```
Lemma ineq_norm_mv:
  ∀ m v, norm (mult_mv m v) ≤ norm_m m * norm v.
```

Other results like $\|M_1 M_2\| \leq \|M_1\|\|M_2\|$ were left unproven. We chose to concentrate on other parts and leave these for later, as it is our belief that they do not require more sophisticated techniques than those already tested.

The part of matrix theory where we encountered some difficulties was the one concerning the inverse of a matrix. The concepts "on paper" need some adaptation before being transposed on a computer. For example, we want to be able to talk about the inverse of a matrix even if we don't know yet whether it is invertible. But we should have the good properties only when we know the matrix is invertible. The solution is similar to the one for division in the `Reals` library and our proposal for derivatives in the one dimensional case. We define a total function on matrices that simulates our inverse, but which acts as the inverse only for the matrices satisfying the property.

Continuing our development with partial derivatives and differentiation, we have managed to formalize some rather important properties. For example, we proved that if a function is partially derivable, then the partial derivative with respect to the variable x_i can be interpreted as the derivative of a function with one argument.

Another useful result is: given a matrix valued function, $f : \mathbb{R}^p \to M_p(\mathbb{R})$, continuous at a point a, a sequence of vectors $\{x_n\}_{n\in\mathbb{N}}$ converging to a and a sequence $\{y_n\}_{n\in\mathbb{N}}$ converging to zero, then the product of $f(x_n)y_n$ will also be a sequence converging to zero.

The most important result of this field for our work is the proof of Taylor's formula for functions of class $C^{(2)}$. The statement and the proof are as follow:

Theorem 3 *Let $A \subset \mathbb{R}^p$ be a convex and open set. If $f : A \to \mathbb{R}$ is twice partially derivable with continuous first and second partial derivatives, then for all $a \in A$ and $v \in \mathbb{R}^p$ with $[a, a+v] \subset A$, there exists $c \in (a, a+v)$ so that $f(a+v) = f(a) + \sum_{i=1}^{p} \frac{\partial f(a)}{\partial x_i} v_i + \frac{1}{2!} \sum_{i,j=1}^{p} \frac{\partial^2 f(c)}{\partial x_i \partial x_j} v_i v_j$.*

Proof. Consider

$$g : [0,1] \to \mathbb{R},\ g(t) = f(a + tv)$$

then g is twice derivable on $[0,1]$ and

$$g'(t) = \sum_{i=1}^{p} \frac{\partial f(a+tv)}{\partial x_i} v_i \tag{6}$$

$$g''(t) = \sum_{i,j=1}^{p} \frac{\partial^2 f(a+tv)}{\partial x_i \partial x_j} v_i v_j \tag{7}$$

From the Taylor formula in one dimension we get that there exists $\eta \in (0,1)$ so that

$$g(1) = g(0) + g'(0) + \frac{1}{2!} g''(\eta)$$

which gives us the desired result for $c = a + t\eta \in (a, a+v)$.

The proof of this theorem is based on the proof of the Taylor formula in one dimension, which we also formalized. Also, an important issue for this proof is to show some relations between various concepts of differentiability, i.e. to prove equalities (6) and (7).

The development is at this point concentrating on Taylor's formula. Once this result is proved, the only pieces still missing for a complete formalization of the theorem, will be some results from matrix theory. The choice not to focus on them was made because formalization of matrix theory is on-going work for the `ssreflect` libraries and most of the results we need should be available in the near future.

5. Formal proof in the multidimensional case

The proof for the multidimensional case follows the same structure described in Section 2.3. Also, the formalization done for the real case was a good guide in our development. Some results generalize nicely from one to several dimensions. For example, the properties of the absolute value function naturally generalize to those of the norm.

Other results, however, do need considerably more work than their real counterparts. Take for example the proof that

$$|ax| = |a||x|\ \ \forall a, x \in \mathbb{R}$$

This is a trivial property of the absolute value function.

The equivalent result we need in the multidimensional case is:

$$\|Ax\| \leq \|A\|\|x\|\ \forall A \in M_{p\times p}, x \in \mathbb{R}^p$$

The proof of this property, however, is far from trivial. On paper the proof mainly deals with inequalities on sums and absolute values. We managed to obtain a formal proof that follows the same line of reasoning. The technique used is induction on the list enumerating the indexes of a vector (as vectors are represented as functions from the list of indexes to R). Intuitively this corresponds to an induction on the dimension of $\mathbb{R}^p$.

Another example of how things get really complicated in several dimensions is the following:

From the inequality $|1 - t| \leq \frac{1}{2}$ one can easily infer that $t \neq 0$.

But having the relation $\|E_p - A\| \leq \frac{1}{2}$, where E_p is the unit matrix of size p, does not trivially imply that A is invertible. In fact, the proof is rather complicated even on paper as it uses results

from the theory of functional spaces, like the convergence of a particular type of matrix series. This is among the results on matrices we have not formalized yet.

In this setting, we provide a verification of the theorem structure. We were pleasantly surprised by the resemblance between this proof and the one we had already done for the real case. For secondary results we mostly needed a generalization of those in one dimension. Also, the structure of the proof for the main theorem is basically the same.

Provided the appropriate declarations of variables and hypothesis, the formal statement of the existence theorem is the following:

```
Theorem kantoroRp_exist:
∃ xs:I→R,
conv_Rp Xn xs ∧ norm (dif_v xs X0) ≤ 2*b0 ∧ f xs = vect0.
```

The development so far has more than 3000 lines of code [2].

6. Conclusions

The degree of difficulty in providing a formal proof for a theorem depends heavily on how well the basic concepts involved are described. It is important to have concepts that are easy to manipulate and correspond to the intuition about the mathematical object they model. This way, formal theorem proving comes rather naturally to one that is used to doing a proof "on paper". For example, having a new approach to derivatives, a new way to handle them, made our work much easier.

Trying to find an adequate formalization for real vectors revealed once more the difference made by a good representation and good proof techniques. We are rather proud of our choice, as it enabled us to obtain quite a large amount of results in a relatively short time.

The work in formalizing multivariate analysis concepts is new for Coq. Though far from offering an extensive coverage of the properties needed, it is a first step towards having such a library.

6.1. Future work

Formal study of Newton's method and Kantorovitch's theorem is important for computer science domains where solving equation systems occurs frequently. Having a formal description of this theorem should make it easier to study whether better convergence criteria can be found when the studied function satisfies special properties. Working with special cases of input functions is often tedious, and having a formalized proof ensures that no special condition will be overlooked. This means that on one hand, the algorithm that does the computation is certified and on the other, special cases are automatically treated in order to obtain better performances from these algorithms.

There is a lot of work that still remains to be done in order to attain such a goal. For the moment, the work accomplished offers an insight on the difficulties that occur in such a development, proposes some potential solutions and gives the formalization of some basic results.

For future work, in the short term, there are still some results that need to be verified in order to have a complete formal proof for the multidimensional case. Then we will be able to concentrate on special cases of the theorem in order to improve efficiency of algorithms. In an even longer term, we would like to study and develop techniques and tools that will enable us to address numerical analysis problems from the formal proof point of view.

[2] http://www-sop.inria.fr/marelle/Ioana.Pasca/multidim/

7. Acknowledgments

This work is based on the author's Master thesis. The author thanks Yves Bertot for his supervision, advice and support.

References

[1] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development, Coq'Art:the Calculus of Inductive Constructions*. Springer-Verlag, 2004.

[2] Coq development team. *The Coq Proof Assistant Reference Manual, version 8.1*, 2006.

[3] Luís Cruz-Filipe. A Constructive Formalization of the Fundamental Theorem of Calculus. In H. Geuvers and F. Wiedijk, editors, *Types for Proofs and Programs*, volume 2464 of *LNCS*, pages 108–126. Springer-Verlag, 2003.

[4] Luís Cruz-Filipe, Herman Geuvers, and Freek Wiedijk. C-CoRN, the Constructive Coq Repository at Nijmegen. In *MKM*, pages 88–103, 2004.

[5] D. Delahaye and M. Mayero. Dealing with algebraic expressions over a field in Coq using Maple *Journal of Symbolic Compuations*, 39(5):569-592, May 2005.

[6] B. Duerte. Elements of Mathematical Analysis in PVS. In *Proceedings of the Ninth International Conference on Theorem Proving in Higher-Order Logics (TPHOL '96)*, 1996.

[7] B. Démidovitch et I. Maron. *Éléments de Calcul Numérique*. Mir - Moscou, 1979.

[8] Jacques D. Fleuriot. On the Mechanization of Real Analysis in Isabelle/HOL. In J. Harrison and M. Aagaard, editors, *Theorem Proving in Higher Order Logics: 13th International Conference, TPHOLs 2000*, volume 1869 of *Lecture Notes in Computer Science*, pages 146–162. Springer-Verlag, 2000.

[9] R. Gamboa and M. Kaufmann. Nonstandard Analysis in ACL2. *Journal of automated reasoning*, 27(4):323–428, November 2001.

[10] H. Geuvers, F. Wiedijk, and J. Zwanenburg. A Constructive Proof of the Fundamental Theorem of Algebra without Using the Rationals. In P. Callaghan, Z. Luo, and R. Pollack, editors, *Types for Proofs and Programs, Proc. of the International Workshop TYPES 2000*, volume 2277 of *LNCS*, pages 96–111. Springer, 2001.

[11] Georges Gonthier. Notation of the Four Colour Theorem Proof. Available at http://research.microsoft.com/ gonthier/4colnotations.pdf.

[12] Georges Gonthier, Assia Mahboubi, Laurence Rideau, Enrico Tassi, and Laurent Théry. *A Modular Formalisation of Finite Group Theory*, Rapport de Recherche 6156, INRIA, 2007.

[13] Michael J. C. Gordon and Thomas F. Melham. *Introduction to HOL : A Theorem Proving Environment for Higher-order Logic*. Cambridge University Press, 1993.

[14] John Harrison. *Theorem Proving with the Real Numbers*. Springer-Verlag, 1998.

[15] John Harrison. A HOL Theory of Euclidian Space. In Joe Hurd and Thomas F. Melham, editors, *TPHOLs*, volume 3603 of *LNCS*, pages 114–129. Springer, 2005.

[16] J. Harrison and L. Théry. A Skeptic's Approach to Combining HOL and Maple. *Journal of Automated Reasoning*, 21(3):279-294, December 1998.

[17] Matt Kaufmann, Panagiotis Manolios, and J. Strother Moore. *Computer-aided Reasoning: an Approach*. Kluwer Academic Publishing, 2000.

[18] Nicolas Magaud. *Programming with Dependent Types in Coq: A Study of Square Matrices*, Coq contribution: http://coq.inria.fr/contribs-eng.html.

[19] Micaela Mayero. *Formalisation et Automatisation de Preuves en Analyses Reelle et Numerique.* PhD thesis, Université de Paris VI, 2001.

[20] Mihail Megan. *Analiza Matematica.* Mirton Timisoara, vol. 1-2, 1999.

[21] Mihail Megan. *Calcul Diferential si Integral in Rp.* Mirton Timisoara, 2000.

[22] Milad Niqui. *Formalising Exact Arithmetic: Representations, Algorithms and Proofs.* PhD thesis, Radboud University, Nijmegen, September 2004.

[23] Lawrence C. Paulson and Tobias Nipkow. *Isabelle : A Generic Theorem Prover*, volume 828 of *Lecture Notes in Computer Science.* Springer-Verlag, 1994.

[24] A. Robinson. *Non-Standard Analysis.* Princeton University Press, 1996.

[25] Natarajan Shankar, Sam Owre, and John M. Rushby. *The PVS Proof Checker: A Reference Manual.* Computer Science Laboratory, SRI International, Menlo Park, CA, February 1993. A new edition for PVS Version 2 is expected in 1998.