

Continuity and Differentiability in ACL2

Ruben Gamboa
Logical Information Machines, Inc.
ruben@lim.com

March 29, 1999

Ruben Gamboa

Abstract

This case study shows how ACL2 can be used to reason about the real and complex numbers, using non-standard analysis. It describes some modifications to ACL2 that include the irrational real and complex numbers in ACL2's numeric system. It then shows how the modified ACL2 can prove classic theorems of analysis, such as the intermediate-value and mean-value theorems.

Introduction

Readers familiar with ACL2 will know that ACL2 supports the rational and complex-rational numbers, but it does not include the irrationals. It is not too difficult to show that the exclusion of the irrationals is complete. For example, it is easy to prove that

`(equal (* x x) 2)`

is false for all values of x . Such readers may also fear that the classical way to introduce the irrationals will not work well with ACL2. In particular, axioms such as the least upper bound axiom are naturally expressed in set theory, but they do not appear to have a similar expression in the language of ACL2.

There is a way out, however. Non-standard analysis, first formalized by Robinson [8], gave a rigorous foundation to the informal reasoning about infinitesimal quantities used by Leibniz when he co-invented calculus and by engineers and scientists ever since. A convenient feature of many arguments using non-standard analysis is that mathematical induction and recursion replace the usual limit and compactness arguments. This suggests that non-standard analysis can be formalized in ACL2. Such a formalization is outlined in the next section; a more complete presentation can be found in [3, 2]. This section is followed with proofs of the intermediate- and mean-value theorems in ACL2.

Exercise 1 *Using ACL2 (not ACL2(r)) prove the theorem*
`(not (equal (* x x) 2))`.

1 Non-Standard Analysis

From the viewpoint of non-standard analysis, the integers can be divided into two groups. The *standard* integers include $0, \pm 1, \pm 2, \dots$. There is at least one non-*standard* integer, say N . The sum or product of two *standard* integers is also *standard*, so $N \pm 1, N \pm 2$ and so on are non-*standard*. This implies there is no least non-*standard* integer.

If a number is smaller in magnitude than all positive *standard* numbers, it is called *i-small*. Similarly, a number is *i-large* if it is larger in magnitude than all *standard* numbers; otherwise, it is called *i-limited*. Two numbers are *i-close* to each other if their difference is *i-small*. These properties have simple interpretations over the integers: The *i-limited* integers are precisely the *standard* integers, the only *i-small* integer is zero, and no two integers are *i-close* to each other.

The reals have a richer structure. Besides zero, there are an infinite number of *i-small* numbers or infinitesimals. These include rationals like $1/N$, where N is the *i-large* integer seen above. It also includes irrationals, such as π/N . It is not the case that all *limited* reals are *standard*. For example, the only *standard* number that is also *i-small* is zero, and we have already seen two other examples of *i-small* numbers. But any *limited* real x is *i-close* to a *standard* number *x ; that is, x can be written as the sum of the unique *standard* number *x and an *i-small* number ϵ , $x = {}^*x + \epsilon$. The number *x is called the *standard-part* of x .

One of the benefits of non-standard analysis, particularly in the context of automated theorem proving, is that the new predicates possess simple algebraic properties. For example, $x + y$ is *i-small* (*i-limited*) if both x and y are *i-small* (*i-limited*). If x is *i-limited* and ϵ is *i-small*, $\epsilon \cdot x$ is *i-small* and for x not *i-small* and $\epsilon \neq 0$, x/ϵ is *i-large*. If x is *i-close* to y and y is *i-close* to z , then x is *i-close* to z . As all of these properties illustrate, the predicates really do capture the intuitive notion of “infinitely small,” “finite,” “infinitely large,” and “infinitely close.”

Standard-part also obeys simple algebraic rules. For example ${}^*(x + y) = {}^*x + {}^*y$ and ${}^*(x \cdot y) = {}^*x \cdot {}^*y$ for *i-limited* x and y . The restrictions to *i-limited* numbers is not necessary for the addition rule, but it is necessary for the product rule. The inverse operations follow similar rules: ${}^*(-x) = -{}^*x$ and ${}^*(1/x) = 1/{}^*x$, but the division rule is only true when x is *i-limited* and not *i-small*. Finally, when $x < y$, it follows that ${}^*x \leq {}^*y$. Notice the last inequality is not strict. Although this list is far from complete, it shows that the rules for the non-standard predicates can easily be captured in an automated theorem prover, such as ACL2(r).

However, one major problem remains. We have seen that the sum of two *i-small* numbers is also *i-small*. Using induction, it would appear that $n \cdot \epsilon$ is *i-small* for any positive integer n and *i-small* ϵ , but letting $n = N$ and $\epsilon = 1/N$ for N any *i-large* integer provides a simple counter-example.

What this illustrates is that induction must be restricted in the presence of non-standard predicates. ACL2(r) contains a modified induction principle that deals with this problem, but for the purposes of this chapter, it is sufficient

to simply disallow induction on formulas using any of the predicates of non-standard analysis. Similarly, we disallow the use of recursion to define a function using any of the new predicates. From now on, we will refer to formulas using any of the non-standard predicates as *classical* formulas. Any function defined using a non-classical function is automatically non-classical. Our restriction limits the use of induction and recursion to the classical formulas only.

But we began this chapter by saying that non-standard analysis makes heavy use of mathematical induction and recursion, where these often replace limit and compactness arguments. How can this be when induction and recursion are disallowed for non-classical formulas? The answer is simple. In the next section, we will see an inductive proof of the intermediate value theorem. A key lemma there will be that for any partition $\{a, a + \epsilon, \dots, a + n\epsilon = b\}$ of the interval $[a, b]$ if $f(a) < 0$ and $f(b) > 0$ there must be some i , $0 \leq i \leq n$, such that $f(a + i\epsilon) < 0$ and $f(a + (i + 1)\epsilon) \geq 0$. This lemma can be established with induction on n , the size of the partition. The intermediate value theorem follows by applying this key lemma when n is *i-large*, or equivalently when ϵ is *i-small*.

Non-standard analysis has one more pleasant surprise. Given a classical formula $F(x)$, if $F(x)$ is true for all *standard* values of x , it is also true for *all* values of x . This is known as the *transfer principle*, and it can be explicitly invoked in ACL2(r) by using the event `defthm-std` in place of `defthm`. Similarly, a *classical* function $f(x)$ can be defined by specifying its values only for the *standard* values of x . This is only permissible when the value of $f(x)$ is *standard* for any *standard* x . We say the function $f(x)$ is implicitly defined by its values on the *standard* elements. Notice that the newly defined function $f(x)$ is *classical*, even though its explicit definition for *standard* values of x may not be. A common technique is to define $f(x)$ as the *standard-part* of some value. Theorems about $f(x)$ can be proved by using `defthm-std`, since this requires that only the *standard* values of x be considered.

This section was a whirlwind tour through non-standard analysis in ACL2(r). These concepts will become clearer in the next section, where we will show how non-standard analysis can be used in ACL2(r) to prove some basic results from calculus. Readers interested in a more thorough introduction to non-standard analysis should consult [7, 1, 6].

2 Continuity

Intuitively, a function f is continuous if $f(x)$ is close to $f(y)$ whenever x is sufficiently close to y . This intuition is muddled using standard analysis, but it is represented faithfully in non-standard analysis, where the formal notion *i-close* replaces the informal notion of “sufficiently close.”

In ACL2, we can introduce an arbitrary continuous function using `encapsulate` as follows:

```
(encapsulate
 ((rcfn (x) t))
```

```

(local (defun rcfn (x) x))
(defthm rcfn-standard
  (implies (standard-numberp x)
    (standard-numberp (rcfn x)))
  :rule-classes (:rewrite :type-prescription))
(defthm rcfn-real
  (implies (realp x)
    (realp (rcfn x)))
  :rule-classes (:rewrite :type-prescription))
(defthm rcfn-continuous
  (implies (and (standard-numberp x)
    (realp x)
    (realp y)
    (i-close x y)
    (i-close (rcfn x) (rcfn y))))
  ).

```

The function `rcfn` is axiomatized as a real, continuous function. These constraints are made in `rcfn-real` and `rcfn-continuous`. Notice the natural expression of continuity in non-standard analysis. Its only surprise is the hypothesis that `x` should be *standard*. Without this hypothesis the constraint on `rcfn` would be stronger: `rcfn` would become uniformly continuous.

All *standard* functions map *standard* arguments to *standard* values. However, the contemporary version of ACL2(r) does not contain this axiom, which is why we added `rcfn-standard` as a constraint. This may not be necessary in a newer version of ACL2(r).

According to the intermediate value theorem, if a , b and z are reals such that $a < b$, $rcfn(a) < z$, and $rcfn(b) > z$, there is some c between a and b such that $rcfn(c) = z$. We can prove this theorem by finding a suitable value of c . Consider the points $x_0 = a$, $x_1 = a + \epsilon$, \dots , $x_n = a + n\epsilon = b$ for a fixed but arbitrary positive integer n . Using a simple induction, it is easy to show that for some index i , $0 \leq i < n$, $rcfn(x_i) < z$ and $rcfn(x_{i+1}) \geq z$. So far neither continuity nor non-standard analysis have been used in the proof, but they will take center stage from this point. First, notice that since `rcfn` is continuous, $rcfn(*x) = *rcfn(x)$ for *i-limited* x . This follows because $*x$ and x are *i-close* and $*x$ is *standard*, so by continuity their `rcfn` values are *i-close*. Moreover, since $rcfn(*x)$ is *standard* it must be equal to $*rcfn(x)$, as no two *standard* numbers are *i-close* to each other. Now, consider what happens when ϵ is *i-small* and a , b , and z are *standard*. Then x_i and x_{i+1} are *i-close*, so $*x_i$ and $*x_{i+1}$ are equal and also *i-close* to x_i and x_{i+1} . Moreover, $rcfn(*x_i) = *rcfn(x_i) \leq *z = z$ and similarly $rcfn(*x_{i+1}) = *rcfn(x_{i+1}) \geq *z = z$, therefore $rcfn(*x_i) = z$. To complete the proof, it is only necessary to show that $a \leq *x_i \leq b$, but this trivially follows from the fact that $a \leq x_i \leq b$ and a and b are *standard*. This proves the intermediate value theorem for *standard* values of a , b , and z . Using the transfer principle, it follows that the intermediate value theorem is true for all real values of a , b , and z .

This argument can be formalized in ACL2(r). We begin with the lemma that $rcfn(*x) = *rcfn(x)$ for *i-limited* n :

```
(defthm rcfn-standard-part
  (implies (and (realp x)
                (i-limited x))
            (equal (rcfn (standard-part x))
                   (standard-part (rcfn x))))
  :hints ...).
```

This lemma is a simple consequence of the continuity of `rcfn` at $*x$ and the fact that two numbers that are *i-close* have the same *standard-part* provided at least one of them is *i-limited*.

The second key lemma is the discovery of the number x_i with its specific properties. This number can be found using the following function:

```
(defun find-zero-n (a z i n eps)
  (declare (xargs :measure (nfix (1+ (- n i))))))
  (if (and (realp a) (integerp i) (integerp n) (< i n)
          (realp eps) (< 0 eps)
          (< (rcfn (+ a eps)) z))
      (find-zero-n (+ a eps) z (1+ i) n eps)
      (realfix a))).
```

This function recurses “from i to n ” looking for the first index i so that $rcfn(a + (i+1) \cdot \epsilon) \geq z$, and it returns $a + i \cdot \epsilon$ (for simplicity, the value of a is incremented each time through the recursion).

It is trivial to show that `find-zero-n` returns a value of x_i so that $rcfn(x_i) < z$:

```
(defthm rcfn-find-zero-n-<-z
  (implies (and (realp a) (< (rcfn a) z))
            (< (rcfn (find-zero-n a z i n eps)) z)))
```

This follows directly from the fact that $rcfn(a) < z$. It is also easy to prove that the value of x_i satisfies $rcfn(x_i + \epsilon) \geq z$, from the fact that $rcfn(b) > z$:

```
(defthm rcfn-find-zero-n+eps->=z
  (implies (and (realp a) (integerp i) (integerp n) (< i n)
                (realp eps) (< 0 eps)
                (< (rcfn a) z)
                (< z (rcfn (+ a (* (- n i) eps)))))
            (<= z (rcfn (+ (find-zero-n a z i n eps)
                           eps)))))
```

Notice the term $(+ a (* (- n i) eps))$ is always equal to b . Without too much difficulty, it is also possible to prove that $a \leq x_i \leq b$. Notice that all these theorems use induction to prove properties of `find-zero-n`, a recursive function.

Since $a \leq x_i \leq b$ for *standard* a and b , it follows that x_i is *i-limited*, hence $*x_i$ is *standard*. We can therefore use `defun-std` to implicitly define a function

that agrees with $*x_i$ on all *standard* arguments. This function should return the right choice of c to satisfy the intermediate value theorem:

```
(defun-std find-zero (a b z)
  (if (and (realp a) (realp b) (realp z) (< a b))
      (standard-part
       (find-zero-n a
                    z
                    0
                    (i-large-integer)
                    (/ (- b a) (i-large-integer))))
      0)).
```

To prove that `find-zero` satisfies the requirements of the intermediate value theorem, we have to transfer the properties already proved about `find-zero-n` to `find-zero`. The properties about `find-zero-n` were proved using induction; now we will use `defthm-std` to transfer these properties to `find-zero`. The lemma `rcfn-find-zero-n-<-z` can be transferred as follows:

```
(defthm-std rcfn-find-zero-<=-z
  (implies (and (realp a) (realp b) (< a b)
                (realp z) (< (rcfn a) z))
            (<= (rcfn (find-zero a b z)) z))
  :hints ...).
```

Because `defthm-std` is used to prove `rcfn-find-zero-<=-z`, `ACL2(r)` assumes the extra hypotheses that a , b , and z are all *standard*. Because of this, the definition of `find-zero` can be opened and replaced with $*x_i$. The remainder of the proof follows simply from the fact that if $x < y$, $*x \leq *y$. Notice how the strict inequality has been replaced. The transfer principle can also be used on `rcfn-find-zero-n+eps->=-z` to conclude that:

```
(defthm-std rcfn-find-zero->=-z
  (implies (and (realp a) (realp b) (< a b)
                (realp z) (< (rcfn a) z) (< z (rcfn b)))
            (<= z (rcfn (find-zero a b z))))
  :hints ...).
```

Similarly, we find that `(find-zero a b z)` is somewhere between a and b . All of these lemmas can be proved by applying the transfer principle applied to the corresponding lemmas about `find-zero-n`. Collecting all of these results gives the intermediate value theorem:

```
(defthm intermediate-value-theorem
  (implies (and (realp a) (realp b) (< a b)
                (realp z) (< (rcfn a) z) (< z (rcfn b)))
            (and (realp (find-zero a b z))
                  (< a (find-zero a b z))
                  (< (find-zero a b z) b)
                  (equal (rcfn (find-zero a b z))
                          z))))
```

:hints ...).

Exercise 2 Prove a version of the intermediate value theorem applicable when $(\text{rcfn } a)$ is greater than z and $(\text{rcfn } b)$ is less than z . **Hint.** Functionally instantiate the intermediate value theorem.

Exercise 3 Using $\text{ACL2}(r)$ show the existence of some x such that $(\text{equal } (* x x) 2)$.

Hint. Use the intermediate value theorem.

The proofs of other familiar facts about continuous functions follow the same pattern. For example, the proof of the maximal and minimal theorems — every continuous function achieves its maximum and minimum values over a closed interval — differs from the above only in the choice of the function `find-zero-n`. The remaining details are very similar. Using induction we establish that $\text{rcfn}(x_i) \geq \text{rcfn}(x_j)$, for all the x_j points in an ϵ -grid from a and b . Then $\text{rcfn}(*x_i) = *\text{rcfn}(x_i) \geq *\text{rcfn}(x_j) = \text{rcfn}(*x_j)$. Notice how continuity is used in this last step to justify moving the *standard-part* out of rcfn . For an arbitrary $x \in [a, b]$, we can find the x_j in the ϵ -grid that is closest to x . If x is *standard*, we have that $\text{rcfn}(x) = *\text{rcfn}(x_j)$ since x and x_j are *i-close*. It follows that for an arbitrary *standard* $x \in [a, b]$, $\text{rcfn}(*x_i) \geq \text{rcfn}(x)$. By the transfer principle, this is true for all $x \in [a, b]$, not just the *standard* ones, so rcfn achieves its maximum at $*x_i$. We leave the reader to formalize this argument in $\text{ACL2}(r)$.

Exercise 4 Define a function `find-max-rcfn-x` and prove that `rcfn` achieves a maximum over $[a, b]$ at the point `(find-max-rcfn-x a b)`.

Exercise 5 Define a function `find-min-rcfn-x` and prove that `rcfn` achieves a minimum over $[a, b]$ at the point `(find-min-rcfn-x a b)`. **Hint.** Instantiate the theorem from the previous exercise.

Exercise 6 Prove that the sum and product of two continuous functions is continuous. **Hint.** Use two constrained functions, `rcfn-1` and `rcfn-2`.

3 Differentiability

Intuitively, we find the derivative of a function f at a point x by taking the slope of the chord between $f(x)$ and $f(x')$ for a point x' infinitely close to x such that $x \neq x'$. In standard analysis we say that as x' approaches x , the slope of the chord approaches the derivative of f at x . The derivative exists if the slope of the chords has a limit. In the language of non-standard analysis, we say that the slope of the chord is *i-close* to the derivative at x if x is *i-close* to x' and $x \neq x'$. In order for the derivative to exist, the slope of the chords between $f(x)$ and $f(x')$ should be *i-close* to the slope of the chord between $f(x)$ and $f(x'')$ whenever x is *i-close* to both x' and x'' and distinct from both of them.

A differentiable function can be introduced into ACL2(r) as follows:

```
(encapsulate
  ((rdfn (x) t))
  (local (defun rdfn (x) x))
  (defthm rdfn-standard
    (implies (standard-numberp x)
              (standard-numberp (rdfn x)))
    :rule-classes (:rewrite :type-prescription))
  (defthm rdfn-real
    (implies (realp x)
              (realp (rdfn x)))
    :rule-classes (:rewrite :type-prescription))
  (defthm rdfn-differentiable
    (implies (and (standard-numberp x)
                  (realp x)
                  (realp y1)
                  (realp y2)
                  (i-close x y1) (not (= x y1))
                  (i-close x y2) (not (= x y2)))
              (and (i-limited (/ (- (rdfn x) (rdfn y1))
                                   (- x y1)))
                    (i-close (/ (- (rdfn x) (rdfn y1))
                                   (- x y1))
                              (/ (- (rdfn x) (rdfn y2))
                                   (- x y2)))))))
  ).
```

We will prove Rolle's theorem for the function `rdfn`: Given reals a, b with $a < b$ so that $\text{rdfn}(a) = \text{rdfn}(b)$, there is a point $c \in [a, b]$ so that the derivative of `rdfn` at c is 0. The proof depends on the maximal and minimal theorems of continuous functions. First, observe that `rdfn` is a continuous function. Therefore, according to exercises 4 and 5, `rdfn` achieves its maximum and minimum. If the maximum is equal to the minimum, `rdfn` is a constant function, and its derivative is zero everywhere on $[a, b]$. Otherwise, either the maximum or minimum occurs inside (a, b) . Assume without loss of generality that the maximum occurs at the point $c \in (a, b)$, and let c' and c'' be two points *i-close* to c with $c' < c < c''$. Then the slope of the chord from $\text{rdfn}(c)$ to $\text{rdfn}(c')$ is positive, but the slope from $\text{rdfn}(c)$ to $\text{rdfn}(c'')$ is negative. Since these slopes must be *i-close*, their *standard-part* — i.e., the derivative at c — must be 0.

So first, we must show that `rdfn` is continuous. Comparing the constraints on `rcfn` and `rdfn`, it is obvious that only `rcfn-continuous` needs to be established for `rdfn`. This follows from the constraint `rdfn-differentiable` by letting y_1 and y_2 be equal to y . Then the difference quotient $\frac{\text{rdfn}(x) - \text{rdfn}(y)}{x - y}$ is *i-limited* and since $x - y$ is *i-small* that follows that $\text{rdfn}(x) - \text{rdfn}(y)$ must be *i-small*, too; i.e., `rdfn` is continuous. This argument can be easily carried out in ACL2(r):

```
(defthm rdfn-continuous
```

```

(implies (and (standard-numberp x)
              (realp x)
              (i-close x y)
              (realp y))
         (i-close (rdfn x) (rdfn y)))
:hints ...).

```

It is now a simple matter to define the functions `find-max-rdfn-x` and `find-min-rdfn-x` that find the points at which `rdfn` achieves its maximum and minimum respectively. The relevant theorems are functional instances of the theorems in exercises 4 and 5, so they are trivial to establish. It is also simple to prove that if the maximum and minimum are equal, the function is a constant on $[a, b]$:

```

(defthm min=max->-constant-rcfn
  (implies (and (realp a) (realp b) (< a b)
               (realp x) (<= a x) (<= x b)
               (= (rcfn (find-min-rcfn-x a b))
                  (rcfn (find-max-rcfn-x a b))))
          (equal (equal (rcfn (find-min-rcfn-x a b))
                       (rcfn x))
                 t))
:hints ...).

```

Notice this theorem is being proved about `rcfn`, since it holds whether the function is differentiable or not. It is a simple matter to functionally instantiate this theorem to the differentiable function `rdfn`.

The remainder of the argument relies heavily on the slope of the chord from $rdfn(x)$ to $rdfn(y)$, also known as the difference quotient of x and y . We are particularly interested when y is *i-close* to x ; that is, when y can be written as $x + \epsilon$ for some *i-small* ϵ :

```

(defun differential-rdfn (x eps)
  (/ (- (rdfn x) (rdfn (+ x eps))) (- eps))).

```

Similarly, the derivative can be defined as follows:

```

(defun derivative-rdfn (x)
  (standard-part
   (differential-rdfn x (/ (i-large-integer))))).

```

Notice that the choice of ϵ is arbitrary since `rdfn` is differentiable.

The first key lemma is that when `rdfn` is a constant function, any difference quotient of the midpoint of $[a, b]$ must be 0:

```

(defthm rolles-theorem-lemma-1
  (implies (and (realp a) (realp b) (< a b)
               (realp eps)
               (< (abs eps) (/ (- b a) 2))
               (= (rdfn (find-min-rdfn-x a b))
                  (rdfn (find-max-rdfn-x a b))))
          (equal (differential-rdfn (/ (+ a b) 2) eps) 0))
:hints ...).

```

ACL2(r) requires a hint to use the lemma `min=max->-constant-rcfn` to deduce that `rdfn` is constant given that its minimum and maximum values are identical. The condition on `eps` being less than half the width of $[a, b]$ ensures that both endpoints of the chord are inside $[a, b]$.

The remaining lemmas are all similar. For example, when `rdfn` achieves its maximum somewhere in the range (a, b) and ϵ is positive, the difference quotients to the right of the maximum point will be negative:

```
(defthm rolles-theorem-lemma-2a
  (implies (and (realp a) (realp b) (< a b)
                (realp eps) (< 0 eps)
                (< a (- (find-max-rdfn-x a b) eps))
                (< (+ (find-max-rdfn-x a b) eps) b))
            (<= (differential-rdfn (find-max-rdfn-x a b)
                               eps)
                 0))).
```

This is obvious from the geometry, since the value of $rdfn(x + \epsilon)$ is at most equal to $rdfn(x)$ and yet $x + \epsilon$ is greater than x . Hence the chord can not have a positive slope. Similar theorems apply to the other cases; i.e., when ϵ is negative or when the minimum is used instead of the maximum.

There is an important technical point remaining. When we look at a point x in $[a, b]$ we can only look at its difference quotient with respect to ϵ if we are guaranteed that $x + \epsilon$ is also in the range $[a, b]$. When a and b are both *standard* and ϵ is *i-small*, it is clear that $a + \epsilon$ and $b - \epsilon$ are in $[a, b]$. In fact, $x \pm \epsilon$ will be in $[a, b]$ for any *standard* $x \in (a, b)$. Otherwise, x would be *i-close* to either a or b , but no two distinct *standard* numbers can be *i-close* to each other. In particular, this holds for the midpoint $(a + b)/2$ and for the values of x at which `rdfn` achieves its maximum or minimum. To verify this, we need only observe that the maximum and minimum are achieved at *standard* values of x :

```
(defthm standard-find-min-max-rdfn
  (implies (and (standard-numberp a)
                (standard-numberp b))
            (and (standard-numberp (find-min-rdfn-x a b))
                  (standard-numberp (find-max-rdfn-x a b))))
  :hints ...).
```

We also need to prove the claim above that $x \pm \epsilon \in (a, b)$ whenever x , a and b are *standard* and ϵ is *i-small*. The following lemma proves half of this claim:

```
(defthm small-squeeze-standard-1
  (implies (and (realp a) (standard-numberp a)
                (realp x) (standard-numberp x)
                (< a x)
                (realp eps) (< 0 eps) (i-small eps))
            (< a (- x eps)))
  :hints ...).
```

The other half is similar.

It is time to define the point that will satisfy Rolle's theorem, the so-called critical point of *rdfn*:

```
(defun rolles-critical-point (a b)
  (if (equal (rdfn (find-min-rdfn-x a b))
            (rdfn (find-max-rdfn-x a b)))
      (/ (+ a b) 2)
      (if (equal (rdfn (find-min-rdfn-x a b)) (rdfn a))
          (find-max-rdfn-x a b)
          (find-min-rdfn-x a b))))
```

Intuitively, the point x at which *rdfn* achieves its minimum will work. However, it is important to consider the special case when the minimum occurs at either of the endpoints. Notice that *rolles-critical-point* always chooses an interior point in (a, b) . To complete the proof, it is only necessary to consider each of the cases implied by the definition of *rolles-critical-point*. This yields the proof of Rolle's theorem in ACL2(r):

```
(defthm rolles-theorem
  (implies (and (realp a) (standard-numberp a)
                (realp b) (standard-numberp b)
                (= (rdfn a) (rdfn b))
                (< a b))
           (equal (derivative-rdfn
                   (rolles-critical-point a b))
                  0))
  :hints ...).
```

This theorem resembles Rolle's theorem from classical analysis; however, there is an important difference. The theorem *rolles-theorem* explicitly requires that the endpoints a and b of the interval are *standard*, whereas no such restriction is made in Rolle's theorem. Consider, for example, *intermediate-value-theorem* where no such restriction is made. What is the difference between these two? The reader will no doubt realize that at no point in the proof of *rolles-theorem* did we use the transfer principle. In the proof of the intermediate value theorem, the transfer principle was used in the proof of the key lemmas *rcfn-find-zero-<=-z* and *rcfn-find-zero->=-z*. That is why the final statement of the theorem makes no assumption about its parameters being *standard*. A similar technique is required to complete the proof of Rolle's theorem. A first attempt may be to replace *defthm* with *defthm-std* in the proof of *rolles-theorem*. However this will fail, since *derivative-rdfn* is a non-classical function. To correct this problem, it will be necessary to modify key parts of the proof.

Exercise 7 *Modify the proof of rolles-theorem to prove the classical Rolle's theorem. Hint. Consider defun-std and defthm-std. A solution can be found in [4].*

Rolle's theorem is typically used to prove the mean-value theorem. Given a differentiable function f on a closed interval $[a, b]$ such that $f(a) = f(b)$, Rolle's

theorem tells there is a point $c \in [a, b]$ with $f'(c) = 0$. The mean-value theorem is a generalization used when $f(a) \neq f(b)$. In these cases, we can find a point $c \in [a, b]$ so that $f'(c) = m$ where m is the slope of the line from $(a, f(a))$ to $(b, f(b))$. The mean-value theorem can be proved by applying Rolle's theorem to the function $g(x) = f(x) - m \cdot (x - a) - f(a)$. Notice that when $g'(x) = 0$, $f'(x) = m$.

The first step is to define the function $g(x)$ in ACL2(r). We would like this function to be defined solely in terms of x , but notice that the definition given above depends on a and b as well. A simple way to accomplish this is to axiomatize the range $[a, b]$ by introducing the functions (a) and (b) of no arguments:

```
(encapsulate
  ((a () t)
   (b () t))
  (local (defun a () 0))
  (local (defun b () 1))
  (defthm realp-a
    (realp (a))
    :rule-classes (:rewrite :type-prescription))
  (defthm realp-b
    (realp (b))
    :rule-classes (:rewrite :type-prescription))
  (defthm standardp-a
    (standard-numberp (a)))
  (defthm standardp-b
    (standard-numberp (b)))
  (defthm a-<-b
    (< (a) (b)))
  ).
```

The definition of g is now straightforward:

```
(defun rdfn2 (x)
  (+ (rdfn x)
     (- (* (- (rdfn (b)) (rdfn (a)))
           (- x (a))
           (/ (- (b) (a))))))
     (- (rdfn (a))))).
```

Since we intend to apply Rolle's theorem to `rdfn2`, we must show that it is differentiable. I.e., we must show that it satisfies all the constraints of `rdfn`: `rdfn-standard`, `rdfn-real`, and `rdfn-differentiable`. The first two are trivial. The last one is more interesting. It can be proved directly from first principles or indirectly as a consequence of lemmas about the differentiability of composition of functions. This is left as an exercise.

Exercise 8 *Prove that the sum and product of two differentiable functions is differentiable.*

It is now possible to invoke Rolle's theorem on `rdfn2`:

```
(defthm rolles-theorem-2
  (implies (and (realp a) (standard-numberp a)
                (realp b) (standard-numberp b)
                (= (rdfn2 a) (rdfn2 b))
                (< a b))
            (equal (derivative-rdfn2
                    (rolles-critical-point-2 a b))
                   0))
  :hints (("Goal"
           :by (:functional-instance rolles-theorem ...))).
```

The theorem `rolles-theorem-2` applies for any *standard* range $[a, b]$. We are particularly interested in the range when a is `(a)` and b is `(b)`. That is, we wish to find a specific instance of `rolles-theorem-2`. Notice that the points `(a)` and `(b)` are *standard* reals by their constraints, and that `(a)` is less than `(b)`. Moreover, from the definition of `rdfn2` it follows that `(rdfn2 (a))` and `(rdfn2 (b))` are both zero. Therefore, when we specialize `rolles-theorem-2` we can remove all of the hypotheses:

```
(defthm rolles-theorem-2-specialized
  (equal (derivative-rdfn2
          (rolles-critical-point-2 (a) (b)))
         0)
  :hints (("Goal" :use ((:instance rolles-theorem-2
                                   (a (a)) (b (b))))
          ...))).
```

The only remaining lemma relates the derivative of `rdfn` to the derivative of `rdfn2`. The following lemma can be proved directly by algebraic manipulation or indirectly by using properties of the difference quotients of sums and products. We chose to follow the direct proof.

```
(defthm mvt-theorem-lemma
  (implies (and (acl2-numberp eps)
                (not (= eps 0)))
            (equal (differential-rdfn x eps)
                    (+ (* (+ (- (rdfn (a))) (rdfn (b)))
                          (/ (+ (- (a)) (b))))
                      (differential-rdfn2 x eps))))
  :hints ...)
```

The mean-value theorem follows from this lemma, the specialized version of Rolle's theorem, and some simple reasoning about *standard-part*:

```
(defthm mvt-theorem
  (equal (derivative-rdfn
          (rolles-critical-point-2 (a) (b)))
         (/ (- (rdfn (b)) (rdfn (a)))
            (- (b) (a))))
```

:hints ...).

4 Conclusions

ACL2(r) extends ACL2 with the capability of reasoning about the real and complex numbers. This chapter illustrates how ACL2(r) can prove some of the classic theorems from elementary analysis, and Kaufmann's article in this book will present more profound theorems from analysis [5]. But the focus of ACL2(r) is not to become a platform for theorem proving about the real numbers. Rather, it is to continue in the tradition of Nqthm and ACL2, to prove theorems about algorithms and mathematical models of circuits. Many algorithms use properties about the real numbers, indirectly by using functions such as the exponential or trigonometric functions or directly, e.g., finding the maximum of a function by finding the roots of its derivative. The point of ACL2(r) is to make these functions and their mathematical properties available to users of ACL2, making ACL2 applicable to even more domains.

References

- [1] F. Diener and M. Diener, editors. *Nonstandard Analysis in Practice*. Springer, 1995.
- [2] Ruben Gamboa. *Mechanically Verifying Real-Valued Algorithms in ACL2*. PhD thesis, The University of Texas at Austin, 1999.
- [3] Ruben Gamboa and Matt Kaufmann. Non-standard analysis in ACL2. in preparation, xxxx.
- [4] M. Kaufmann, P. Manolios, and J S. Moore. Supporting files for "Using the ACL2 Theorem Prover", 1999. <http://www.cs.utexas.edu/users/-moore/acl2-book-99/index.html>.
- [5] Matt Kaufmann. A modular proof methodology for ACL2, 2000.
- [6] Edward Nelson. On-line books: Internal set theory. Available on the world-wide web at <http://www.math.princeton.edu/~nelson/books.html>, xxxx.
- [7] A. Robert. *Non-Standard Analysis*. John Wiley, 1988.
- [8] A. Robinson. Model theory and non-standard arithmetic, infinitistic methods. In *Symposium on Foundations of Mathematics*, 1959.