Inverse Functions in ACL2(r)

The Nine Billion Names of $\sqrt{2}$

R. Gamboa J. Cowles

University of Wyoming

ACL2 Workshop 2009

Outline

$\sqrt{2}$ in ACL2: Nasty, Brutish, Short

```
(defthm there-is-no-sqrt-2
  (not (equal (* x x) 2)))
```

Proof: By case analysis.

- $\sqrt{2}$ must be numeric
- $\sqrt{2}$ cannot be rational
- $\sqrt{2}$ cannot be complex-rational
- All numbers in ACL2 are rational or complex-rational

Approximating $\sqrt{x}$ in ACL2

```
(defthm convergence-of-iter-sqrt
  (implies (and (rationalp x)
                (rationalp epsilon)
                (< 0 epsilon)
                (≤ 0 x))
           (and (≤ (* (iter-sqrt x epsilon)
                      (iter-sqrt x epsilon))
                   x)
                (< (- x (* (iter-sqrt x epsilon)
                           (iter-sqrt x epsilon)))
                   epsilon))))
```

$\sqrt{x}$ in ACL2(r)

$\sqrt{x}$ can be introduced in ACL2(r), because

- ACL2(r) adds the irrationals to ACL2's number system
- The completeness of the real numbers is established via standard-part, part of an axiomatization of the reals based on non-standard analysis

The Goal of ACL2(r)

- ACL2(r) uses non-standard analysis to introduce notions from calculus into ACL2, e.g., the Intermediate Value Theorem
- Eventually, it should know about all results from first-year calculus (but that's in the future)
- Today, we take a step forward, by introducing inverse functions, including $\ln x$

Inverse Functions

Suppose $f : D \rightarrow R$ has the following properties:

- f is 1-1: $\forall x_1, x_2 \in D. f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
- f is onto: $\forall y \in R. \exists x \in D. f(x) = y$

Inverse Functions

Suppose $f : D \rightarrow R$ has the following properties:

- f is 1-1: $\forall x_1, x_2 \in D. f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
- f is onto: $\forall y \in R. \exists x \in D. f(x) = y$

Then there is a function $f^{-1} : R \rightarrow D$ such that

- $\forall y \in R. f(f^{-1}(y)) = y$

Inverse Functions

Suppose $f : D \to R$ has the following properties:

- f is 1-1: $\forall x_1, x_2 \in D. f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
- f is onto: $\forall y \in R. \exists x \in D. f(x) = y$

Then there is a function $f^{-1} : R \to D$ such that

- $\forall y \in R. f(f^{-1}(y)) = y$
- $\forall x \in D. f^{-1}(f(x)) = x$

Inverse Functions

Suppose $f : D \to R$ has the following properties:

- f is 1-1: $\forall x_1, x_2 \in D. f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
- f is onto: $\forall y \in R. \exists x \in D. f(x) = y$

Then there is a function $f^{-1} : R \to D$ such that

- $\forall y \in R. f(f^{-1}(y)) = y$
- $\forall x \in D. f^{-1}(f(x)) = x$
- $\forall y_1, y_2 \in R. f^{-1}(y_1) = f^{-1}(y_2) \Rightarrow y_1 = y_2$

Inverse Functions

Suppose $f : D \to R$ has the following properties:

- f is 1-1: $\forall x_1, x_2 \in D. f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
- f is onto: $\forall y \in R. \exists x \in D. f(x) = y$

Then there is a function $f^{-1} : R \to D$ such that

- $\forall y \in R. f(f^{-1}(y)) = y$
- $\forall x \in D. f^{-1}(f(x)) = x$
- $\forall y_1, y_2 \in R. f^{-1}(y_1) = f^{-1}(y_2) \Rightarrow y_1 = y_2$
- $\forall x \in D, y \in R. f(x) = y \Rightarrow f^{-1}(y) = x$

Inverse Functions in ACL2(r)

- The invertible function f can be introduced using encapsulate
- The constraints include "f is 1-1" (easy) and "f is onto" (hard)
- The domain D and range R also need to be introduced (as unary boolean functions)
- The function f^{-1} can be defined using defchoose

The Onto Constraint

- The problem with onto is that it uses existential quantifiers ($\forall y \exists x \ldots$)
- Normally, we would define a function $g(y)$ to remove the quantifier $\exists x$
- But that would be the inverse function!
- Instead, we use ACL2's support for quantifiers

The Onto Constraint with Quantifiers

1. Name the property "ontoness"

```
(defun-sk ifn-is-onto-predicate (y)
  (exists (x)
    (and (ifn-domain-p x)
         (equal (ifn x) y))))
```

2. Assert that the property holds

```
(defthm ifn-is-onto
  (implies (ifn-range-p y)
           (ifn-is-onto-predicate y)))
```

Defining the Inverse

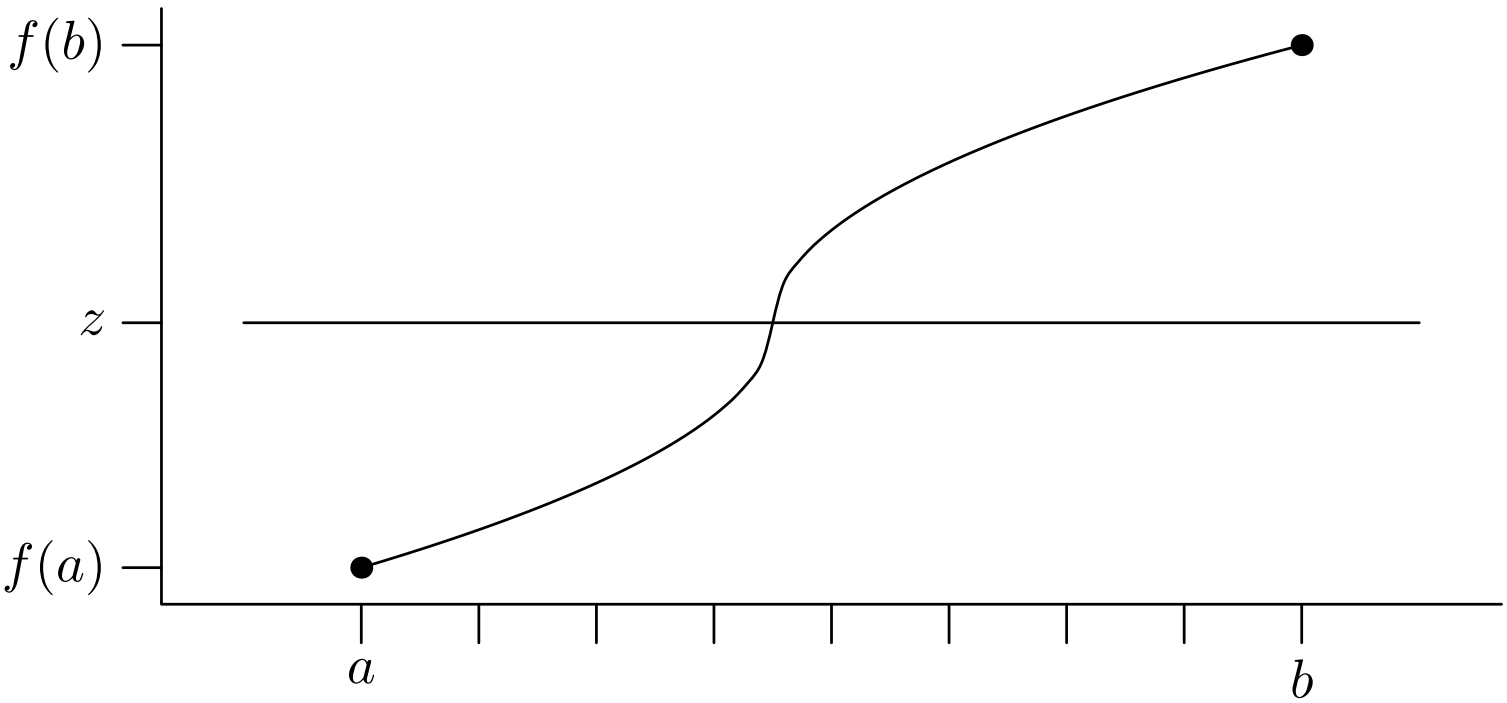
```
(defchoose ifn-inverse (x) (y)
  (and (ifn-domain-p x)
       (equal (ifn x) y)))
```

- By itself, the defchoose simply states that if any function can be an inverse of f, then f^{-1} can be that inverse
- That the inverse function exists is guaranteed by the constraints on the function f (aka ifn)

$\sqrt{x} \equiv$ Inverse of Square?

- We can apply this theorem to the function $f(x) = x^2$ to find $f^{-1}(x) = \sqrt{x}$
- Clearly, x^2 is 1-1 (over the non-negative reals)
- But how do we know that x^2 is onto (over the non-negative reals)?
- Possible Answer: The Intermediate Value Theorem (IVT)
- This will work for all continuous functions, not just x^2

The Intermediate Value Theorem



Applying the IVT

- We no longer require that f is onto

Applying the IVT

- We no longer require that f is onto
- We still require that f is 1-1

Applying the IVT

- We no longer require that f is onto
- We still require that f is 1-1
- Also, f must be continuous

Applying the IVT

- We no longer require that f is onto
- We still require that f is 1-1
- Also, f must be continuous
- The domain and range should be intervals
  - Aside: We chose to represent intervals explicitly, so that we can quantify over intervals easily (and prove such theorems as $x \in I \wedge z \in I \wedge x < y < z \Rightarrow y \in I$)

Applying the IVT

- We no longer require that f is onto
- We still require that f is 1-1
- Also, f must be continuous
- The domain and range should be intervals
  - Aside: We chose to represent intervals explicitly, so that we can quantify over intervals easily (and prove such theorems as $x \in I \wedge z \in I \wedge x < y < z \Rightarrow y \in I$)
- We need to find $a, b \in D$ such that $f(a) < z < f(b)$ or $f(a) > z > f(b)$

Applying the IVT

- We no longer require that f is onto
- We still require that f is 1-1
- Also, f must be continuous
- The domain and range should be intervals
  - Aside: We chose to represent intervals explicitly, so that we can quantify over intervals easily (and prove such theorems as $x \in I \wedge z \in I \wedge x < y < z \Rightarrow y \in I$)
- We need to find $a, b \in D$ such that $f(a) < z < f(b)$ or $f(a) > z > f(b)$

Once we introduce such a function, we can use functional-instantiate to define the inverse

The Definv Macro

- Using the IVT to justify ontoness is a very powerful tool
- We can use it to define many different inverse functions
- The macro definv automates this process
  - Use defchoose to introduce the inverse function
  - Use functional-instantiate functional-instantiate to prove the inverse properties

Finally! $\sqrt{x}$

```
(defun square (x)
  (realfix (* x x)))
(defun square-interval (y)
  (if (< 1 y)
      (interval 1 y)
      (interval 0 1)))
(definv square
        :domain (interval 0 nil)
        :range  (interval 0 nil)
        :inverse-interval square-interval)
```

Inverse Trigonometric Functions

- The same ideas can be applied to sine and cosine
- The hard part is showing that sine and cosine are 1-1 and continuous over the appropriate domains

Inverse Sine

```
(defun sine-interval (y)
  (declare (ignore y))
  (interval (- (/ (acl2-pi) 2)) (/ (acl2-pi) 2)))

(definv real-sine
        :f-inverse acl2-asin
        :domain    (interval (- (/ (acl2-pi) 2))
                             (/ (acl2-pi) 2))
        :range     (interval -1 1)
        :inverse-interval sine-interval)
```

Inverse Cosine

```
(defun cosine-interval (y)
  (declare (ignore y))
  (interval 0 (acl2-pi)))

(definv real-cosine
        :f-inverse acl2-acos
        :domain    (interval 0 (acl2-pi))
        :range     (interval -1 1)
        :inverse-interval cosine-interval)
```

Application: Polar Form

- Suppose $z \equiv a + bi$ is a non-zero complex number
- Then z can be written as $z = re^{i\theta}$ where
  - $r = ||z|| = \sqrt{a^2 + b^2}$
  - $\theta = \cos^{-1}(a/z)$ or $\theta = 2\pi - \cos^{-1}(a/z)$, depending on the sign of b

Application: Polar Form

- Suppose $z \equiv a + bi$ is a non-zero complex number
- Then z can be written as $z = re^{i\theta}$ where
  - $r = ||z|| = \sqrt{a^2 + b^2}$
  - $\theta = \cos^{-1}(a/z)$ or $\theta = 2\pi - \cos^{-1}(a/z)$, depending on the sign of b

Note: The following properties are easy to prove in ACL2(r):

- r is a non-negative real
- $r = 0$ only when $a + bi = 0$
- $r = |a|$ when $b = 0$
- $\theta \in [0, 2\pi)$
- if $b = 0$, $\theta = 0$ or $\theta = \pi$, depending on the sign of a

Natural Logarithm

- The macro definv can be used to define $\ln y$ for $y \in [1, \infty)$

Natural Logarithm

- The macro definv can be used to define $\ln y$ for $y \in [1, \infty)$
- This definition can be extended to $y \in (0, \infty)$ by using the property $e^{a-b} = e^a / e^b$, so $\ln(1/y) = -\ln(y)$ when $y \in (0, 1)$

Natural Logarithm

- The macro definv can be used to define $\ln y$ for $y \in [1, \infty)$
- This definition can be extended to $y \in (0, \infty)$ by using the property $e^{a-b} = e^a / e^b$, so $\ln(1/y) = -\ln(y)$ when $y \in (0, 1)$
- Finally, when $z \in \mathbb{C}$ and $z \neq 0$, we can write $z = re^{i\theta}$ so $\ln z = \ln r + i\theta$, where $\ln r$ is as defined previously, since $r \in (0, \infty)$

Natural Logarithm

- The macro definv can be used to define $\ln y$ for $y \in [1, \infty)$
- This definition can be extended to $y \in (0, \infty)$ by using the property $e^{a-b} = e^a / e^b$, so $\ln(1/y) = -\ln(y)$ when $y \in (0, 1)$
- Finally, when $z \in \mathbb{C}$ and $z \neq 0$, we can write $z = re^{i\theta}$ so $\ln z = \ln r + i\theta$, where $\ln r$ is as defined previously, since $r \in (0, \infty)$

Note: It is easy to prove in ACL2(r) that ln satisfies the usual properties, e.g.,

- $\ln(xy) = \ln x + \ln y$
- $\ln \frac{1}{x} = -\ln x$

General Exponentials

- When a and x are numbers and $a \neq 0$, we can define $a^x \equiv e^{x \ln a}$

Again, it is easy to prove that a^x satisfies the usual properties, e.g.,

- $a^{x+y} = a^x a^y$
- $a^{-x} = \frac{1}{a^x}$

It is also easy to show that a^i is equal to the ACL2 built-in function `(expt a i)` when i is an integer

$\sqrt{x}$ Again

From the basic properties of a^x, we can show that

- $x^{1/2} \cdot x^{1/2} = x^1 = x$
- So when $x \in [0, \infty)$, $x^{1/2} = \sqrt{x}$

The last property follows from the uniqueness of inverse functions

Current Work

- For technical reasons (having to do with restrictions on encapsulate, non-classical terms, and free variables), we can only invert functions of a single variable
- However, if we use a classical definition of continuity, we can avoid this restriction
- We are currently working on an ACL2 book that introduces continuity in this way
- In anticipation of that proof, we have some early results with inverses of multi-variable functions (when all but one variable are held fixed)

General Logarithms

- The function a^x can be inverted to yield $\log_a x$
- Since a^x is a function of two variables (a and x), this requires that we specify which variable we are inverting and which is held fixed

General Roots

- The function x^n can be inverted to yield $\sqrt[n]{x}$
- This is the same as inverting a^x, but holding the other variable fixed
- I.e., when we write x^n, we think of n as fixed and x as the free variable

Yet Another $\sqrt{x}$

- Of course, $\sqrt[2]{x} = \sqrt{x}$
- This follows (again) from the uniqueness of inverse functions

Conclusion

- It is often useful to define a function as the inverse of another
  - E.g., $\sqrt{x}$, $\ln x$, $\sin^{-1}(x)$
- ACL2(r) now supports such implicit definitions for many functions
- This takes advantage of ACL2's support for quantifiers, constrained functions, and macros
- Among other things, this mechanism provides us with several new ways to define $\sqrt{x}$ in ACL2(r)