ACL2(r) Mechanized Proof of the Orthogonality Relations of Trigonometric Functions Using Non-Standard Analysis

Cuong Chau

Agenda

- Introduction
- Warm-Up Exercise
- ACL2(r) Mechanized Proof of the Orthogonality Relations of Trigonometric Functions
- Conclusion and Future Directions

Agenda

- Introduction
- Warm-Up Exercise
- ACL2(r) Mechanized Proof of the Orthogonality Relations of Trigonometric Functions
- Conclusion and Future Directions

Introduction

- The orthogonality relations of trigonometric functions.

$$\int_{-L}^{L} \sin\left(m\frac{\pi}{L}x\right)\sin\left(n\frac{\pi}{L}x\right)dx = \begin{cases} 0, \text{ if } m \neq n \vee m = n = 0 \\ L, \text{ if } m = n \neq 0 \end{cases}$$

$$\int_{-L}^{L} \cos\left(m\frac{\pi}{L}x\right)\cos\left(n\frac{\pi}{L}x\right)dx = \begin{cases} 0, \text{ if } m \neq n \\ L, \text{ if } m = n \neq 0 \\ 2L, \text{ if } m = n = 0 \end{cases}$$

$$\int_{-L}^{L} \sin\left(m\frac{\pi}{L}x\right)\cos\left(n\frac{\pi}{L}x\right)dx = 0$$

Introduction

- The orthogonality relations of trigonometric functions (when L = π).

$$\int_{-\pi}^{\pi} \sin(mx)\sin(nx)dx = \begin{cases} 0, \text{ if } m \neq n \vee m = n = 0 \\ \pi, \text{ if } m = n \neq 0 \end{cases}$$

$$\int_{-\pi}^{\pi} \cos(mx)\cos(nx)dx = \begin{cases} 0, \text{ if } m \neq n \\ \pi, \text{ if } m = n \neq 0 \\ 2\pi, \text{ if } m = n = 0 \end{cases}$$

$$\int_{-\pi}^{\pi} \sin(mx)\cos(nx)dx = 0$$

Introduction

- The orthogonality relations of trigonometric functions play an important role in Fourier series analysis.
- They are often used to determined the Fourier coefficients of periodic functions.
- Lack of ACL2 mechanized proofs of these properties limits ACL2 for reasoning about Fourier series properties.

Introduction

- We present an ACL2(r) mechanized proof of these orthogonality relations using the Second Fundamental Theorem of Calculus (FTC-2).
- The proof procedure can also be applied to compute the definite integral of any real-valued continuous function f defined on an interval [a, b], even when f contains free variables.

Defun-std

- Syntax is like defun:
  (defun-std f (x1 ... xn)
     <body>)
- Proof obligation for the above defun-std form:
  (implies (and (standardp x1) ... (standardp xn))
              (standardp <body>))
  Note that <body> does not need to be classical!
- Axiom added for the above defun-std form:
  (implies (and (standardp x1) ... (standardp xn))
              (equal (f x1 ... xn)
                        <body>))

Defthm-std

- The transfer principle is implemented in ACL2(r) with defthm-std.

  (defthm-std name <body>) ; optionally, :hints etc.

  – Apply if the <body> is classical. Before attempting the proof, ACL2(r) adds a hypothesis of (standardp x) for all variables x in the <body>:

    (implies (and (standardp x1) (standardp x2) ... (standardp xk))
    <body>)

  – Also apply to prove that a classical function returns standard values with standard inputs. Formally, if f is classical,

    (defthm-std name
    (implies (and (standardp x1) (standardp x2) ... (standardp xk))
    (standardp (f x1 x2 ... xk))))

Agenda

- Introduction
- Warm-Up Exercise
- ACL2(r) Mechanized Proof of the Orthogonality Relations of Trigonometric Functions
- Conclusion and Future Directions

Warm-Up Exercise

- Theorem: For all infinitesimal real numbers x, sin(x)/x is infinitesimally close (i-close) to 1.
- Approach: Using Taylor series expansion of the sine function.
- This exercise helped me learn how to apply the transfer principle in non-standard analysis into my proof.

Warm-Up Exercise

- Theorem: For all infinitesimal real numbers x, sin(x)/x is infinitesimally close (i-close) to 1.
- Approach: Using Taylor series expansion of the sine function.
- This exercise helped me learn how to apply the transfer principle in non-standard analysis into my proof.
- Key lemma: For all real numbers x,

$$\left|\sin(x) - x\right| \le x^2 \text{ when } \left|x\right| \le c \text{ for some constant } c$$

$$(c = 2 \text{ in my proof})$$

Warm-Up Exercise

- Taylor series expansion of the sine function

$$\sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} x^{2k+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

- ACL2(r): For all standard real numbers x,

$$\sin(x) \equiv \text{standard-part}\left(\sum_{k=0}^{\text{i-large-integer}} \frac{(-1)^k}{(2k+1)!} x^{2k+1} \right)$$

$$\equiv \text{standard-part}\left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots \right)$$

Warm-Up Exercise

- For all standard real numbers x,

$$\sin(x) - x \equiv \text{standard-part}\left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots\right) - x$$

$$\equiv \text{standard-part}\left(-\frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots\right)$$

Warm-Up Exercise

- For all standard real numbers x,

$$\sin(x) - x \equiv \text{standard-part}\left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots\right) - x$$

$$\equiv \text{standard-part}\left(-\frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots\right)$$

$$\Rightarrow \left|\sin(x) - x\right| \leq \left|\frac{x^3}{3!}\right| \leq x^2 \text{ when } |x| \leq 2 \;\; (*)$$

- By the transfer principle, (*) is also true for all real numbers x.

Warm-Up Exercise

- For all ~~standard~~ real numbers x,

$$|\sin(x) - x| \le x^2 \text{ when } \quad |x| \le 2 \ (*)$$

$$\Rightarrow \left|\frac{\sin(x)}{x} - 1\right| \le |x| \text{ when } 0 < |x| \le 2$$

$$\Rightarrow -|x| \le \frac{\sin(x)}{x} - 1 \le |x| \text{ when } 0 < |x| \le 2$$

$$\Rightarrow 0 \approx -|x| \le \frac{\sin(x)}{x} - 1 \le |x| \approx 0 \text{ when } x \text{ is infinitesimal}$$

$$\Rightarrow \frac{\sin(x)}{x} \approx 1 \text{ when } x \text{ is infinitesimal (Q.E.D.)}$$

Agenda

- Introduction
- Warm-Up Exercise
- ACL2(r) Mechanized Proof of the Orthogonality Relations of Trigonometric Functions
- Conclusion and Future Directions

FTC-2

- If f' is a real-valued continuous function on [a, b] and the derivative of a real-valued function f is f' on [a, b], then

$$\int_a^b f'(x)\,dx = f(b) - f(a)$$

- Goal: Apply FTC-2 to compute

$$\int_{-\pi}^{\pi} \sin(mx)\sin(nx)dx,$$

$$\int_{-\pi}^{\pi} \cos(mx)\cos(nx)dx,$$

$$\int_{-\pi}^{\pi} \sin(mx)\cos(nx)dx$$

Proof Procedure

- f’ returns real values on [a, b].
- f’ is continuous on [a, b].
- Specifying the real-valued antiderivative f of f’ and proving that f’ is the derivative of f on [a, b].
- Defining the Riemann integral of f’ on [a, b].
- Functionally instantiating the FTC-2 to compute the integral of f’ over [a, b].

Proof Procedure

- f’ returns real values on [a, b].
- f’ is continuous on [a, b].
- Specifying the real-valued antiderivative f of f’ and proving that f’ is the derivative of f on [a, b].
- Defining the Riemann integral of f’ on [a, b].
- Functionally instantiating the FTC-2 to compute the integral of f’ over [a, b].

Derivative and Antiderivative

- Specifying the antiderivative of a function via some symbolic mathematics system. E.g., Wolfram Alpha.
- In non-standard analysis, showing that f' is the derivative of f is equivalent to prove the following formula:

$$\frac{f(x) - f(y)}{x - y} \approx f'(x)$$

$$\text{where standardp}(x) \wedge x \approx y \wedge x \neq y$$

Defderivative

- The macro defderivative, written by Peter Reid and Ruben Gamboa [1], computes the derivative f' of a function f automatically using symbolic differentiation. It also introduces the theorem showing that f' is, in fact, the derivative of f.

[1] P. Reid and R. Gamboa. Automatic differentiation in ACL2. In *Proc of the Second Conference on Interactive Theorem Proving (ITP-2011)*, 2011.

Defderivative

- Demo.
- Constraints when submitting a defderivative event to ACL2(r):
  - Use the symbol x as the name of the variable with respect to which the (partial) derivative is computed.
  - Do not use the symbol y as the name of any variable in the function. Defderivative already reserved this symbol.

Def-elem-derivative

- Users can register functions with known derivatives to defderivative via the macro def-elem-derivative [1].
- Limitation: def-elem-derivative does not support partial derivative registrations.

[1] P. Reid and R. Gamboa. Automatic differentiation in ACL2. In *Proc of the Second Conference on Interactive Theorem Proving (ITP-2011)*, 2011.

Proof Procedure

- f' returns real values on [a, b].
- f' is continuous on [a, b].
- Specifying the real-valued antiderivative f of f' and proving that f' is the derivative of f on [a, b].
- Defining the Riemann integral of f' on [a, b].
- Functionally instantiating the FTC-2 to compute the integral of f' over [a, b].

Riemann Integral

- The Riemann integral of a function f on an interval [a, b] is the limit of the Riemann sum of f when partitioning [a, b] into extremely small subintervals.
- In non-standard analysis, the Riemann integral is the standard part of the Riemann sum when partitioning [a, b] into infinitesimal subintervals.

```
(defun-std riemann-integral-f-prime (a b)
  (if (< a b)
      (standard-part (riemann-sum-f-prime (small-partition a b)))
    0))
```

- Proof obligation: the Riemann sum is limited when a and b are standard.

Riemann Integral

- Fact: Ruben Gamboa proved that the Riemann sum of any real-valued continuous unary function over a finite interval [a, b] is limited.
- Question: Can we apply this fact to prove for the case of functions of more than one variable (i.e., functions contain free variables) using functional instantiation?

Riemann Integral

- Fact: Ruben Gamboa proved that the Riemann sum of any real-valued continuous unary function over a finite interval [a, b] is limited.
- Question: Can we apply this fact to prove for the case of functions of more than one variable (i.e., functions contain free variables) using functional instantiation? No. Because the theorem we try to prove is non-classical and the functions we try to instantiate are classical [2].

[2] R. Gamboa and J. Cowles. Theory Extension in ACL2(r). In *Journal of Automated Reasoning*, 2007.

Functional Instantiation

Example: Given an arbitrary classical function f(x), it follows that

standardp(x) => standardp(f(x))

If we substitute λ(x).(x + y) into this theorem, we would conclude that

standardp(x) => standardp(x + y)

But this is false!

Riemann Integral

- Fact: Ruben Gamboa proved that the Riemann sum of any real-valued continuous unary function over a finite interval [a, b] is limited.
- Question: Can we apply this fact to prove for the case of functions of more than one variable (i.e., functions contain free variables) using functional instantiation? No. Because the theorem we try to prove is non-classical and the functions we try to instantiate are classical [2].
- Solution: Prove from scratch!

Proof Idea

- Prove that f’ is bounded on [a, b] by limited values.

  f’_min ≤ f’ ≤ f’_max

  where f’_min and f’_max are limited.
- Then, (b – a)*f’_min ≤ Riemann_sum_f’ ≤ (b – a)*f’_max
- Given that a and b are standard, the Riemann_sum_f’ is bounded on [a, b] by limited values. By the squeeze theorem, the Riemann_sum_f’ is also limited.

Proof Procedure

- f’ returns real values on [a, b].
- f’ is continuous on [a, b].
- Specifying the real-valued antiderivative f of f’ and proving that f’ is the derivative of f on [a, b].
- Defining the Riemann integral of f’ on [a, b].
- Functionally instantiating the FTC-2 to compute the integral of f’ over [a, b].

Functional Instantiation of FTC-2

- Fact: Ruben Gamboa proved the FTC-2 for any real-valued continuous unary function defined on an interval [a, b].
- Question: Can we apply this fact to prove for the case of functions of more than one variable (i.e., functions contain free variables) using functional instantiation?

Functional Instantiation of FTC-2

- Fact: Ruben Gamboa proved the FTC-2 for any real-valued continuous unary function defined on an interval [a, b].
- Question: Can we apply this fact to prove for the case of functions of more than one variable (i.e., functions contain free variables) using functional instantiation? Yes. Free variables are allowed to appear in functional instantiations if the theorem we try to prove and the functions we try to instantiate are all classical [2].

[2] R. Gamboa and J. Cowles. Theory Extension in ACL2(r). In *Journal of Automated Reasoning*, 2007.

Functional Instantiation of FTC-2

- Fact: Ruben Gamboa proved the FTC-2 for any real-valued continuous unary function defined on an interval [a, b].
- Question: Can we apply this fact to prove for the case of functions of more than one variable (i.e., functions contain free variables) using functional instantiation? Yes. Free variables are allowed to appear in functional instantiations if the theorem we try to prove and the functions we try to instantiate are all classical [2].
- How? Use an "encapsulate trick" with zero-arity functions representing free variables. Demo.

Encapsulate Trick

- Step 1: Define an encapsulate event that introduces zero-arity classical functions representing free variables.
- Step 2: Prove that the zero-arity functions return standard values (use defthm-std).
- Step 3: Prove the main theorem but replacing the free variables with the zero-arity functions introduced in step 1. Without free variables, the functional instantiation can be applied straightforwardly.
- Step 4: Prove the main theorem by functionally instantiating the zero-arity functions in the lemma proved in step 3 with free variables.

Proof Procedure

- f’ returns real values on [a, b].
- f’ is continuous on [a, b].
- Specifying the real-valued antiderivative f of f’ and proving that f’ is the derivative of f on [a, b].
- Defining the Riemann integral of f’ on [a, b].
- Functionally instantiating the FTC-2 to compute the integral of f’ over [a, b].

Agenda

- Introduction
- Warm-Up Exercise
- ACL2(r) Mechanized Proof of the Orthogonality Relations of Trigonometric Functions
- Conclusion and Future Directions

Conclusion and Future Directions

- The proposed proof procedure can also be applied to compute the definite integral of any real-valued continuous function f' defined on an interval [a, b], even when f' contains free variables.
- The derivative can be computed automatically by the automatic differentiator defderivative. However, users cannot register partial derivatives to the automatic differentiator. Could we extend it?
- Still remain a couple of proof obligations in the proposed proof procedure that need to be proved manually. Could we build a symbolic integrator?
- At some point, we would like to use ACL2(r) to verify properties of some physical systems, e.g, computer controlled systems.

References

- [1] P. Reid and R. Gamboa. Automatic differentiation in ACL2. In *Proc of the Second Conference on Interactive Theorem Proving (ITP-2011)*, 2011.
- [2] R. Gamboa and J. Cowles. Theory Extension in ACL2(r). In *Journal of Automated Reasoning*, 2007.
- [3] P. Reid and R. Gamboa. Implementing an Automatic Differentiator in ACL2. In *ACL2 Workshop*, 2011.
- [4] R. Gamboa. Mechanically Verifying Real-Valued Algorithms in ACL2. *Ph.D. thesis, The University of Texas at Austin*, 1999.

Questions!