UMBC

Chemical Biochemical and Environmental Engineering

LEAN THEOREM PROVER

Lean for Scientists and Engineers

Tyler R. Josephson

AI & Theory-Oriented Molecular Science (ATOMS) Lab

University of Maryland, Baltimore County

Bubble Tea ft. juu & cinders
dark cat

Twitter: @trjosephson
Email: tjo@umbc.edu

Lean for Scientists and Engineers 2024

1. Logic and proofs for scientists and engineers
    1. Introduction to theorem proving
    2. Writing proofs in Lean
    3. Formalizing derivations in science and engineering
2. Functional programming in Lean 4
    1. Functional vs. imperative programming
    2. Numerical vs. symbolic mathematics
    3. Writing executable programs in Lean
3. Provably-correct programs for scientific computing

Schedule (tentative)

Logic and proofs for scientists and engineers
Functional programming in Lean 4
Provably-correct programs for scientific computing

July 9, 2024	Introduction to Lean and proofs
July 10, 2024	Equalities and inequalities
July 16, 2024	Proofs with structure
July 17, 2024	Proofs with structure II
July 23, 2024	Proofs about functions; types
July 24, 2024	Calculus-based-proofs
July 30-31, 2024	Prof. Josephson traveling
August 6, 2024	Functions, definitions, structures, recursion
August 8, 2024	Polymorphic functions for floats and reals, compiling Lean to C
August 13, 2024	Input / output, lists, arrays, and indexing
August 14, 2024	Lists, arrays, indexing, and matrices
August 20, 2024	LeanMD & BET Analysis in Lean
August 21, 2024	SciLean tutorial, by Tomáš Skřivan

Content inspired by:
Mechanics of Proof, by Heather Macbeth
Functional Programming in Lean, by David Christiansen

Guest instructor: Tomáš Skřivan

Schedule for today

1. Survey for attendees
2. Recap Lecture 1
3. Syntax vs. Semantics
4. Solving equalities and inequalities in Lean

Survey for attendees

https://forms.gle/pg5JGpTgD1aSCshY6

Slide from Lecture 1

Schedule for today

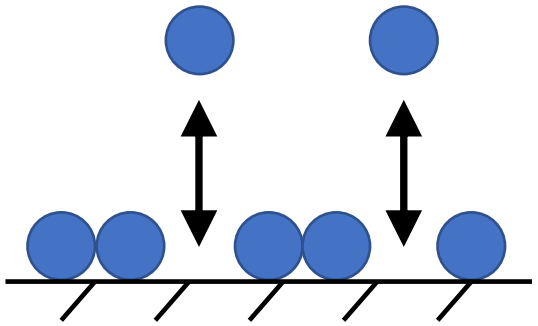
1. Provably-correct scientific computing
2. Derivations in science and engineering are math proofs
3. Formalizing mathematics with computers
4. Lean 4 and Mathlib
5. Case studies in proofs: adsorption and gas law thermodynamics
6. Case study in programming: bug-free BET analysis
7. Outlook
    1. LeanMD
    2. LLMs for theorem proving
    3. SciLib

Intermission

1. Getting connected with this course
2. Getting started with Lean
3. Proofs about equality

Slide from Lecture 1

Derivations in science are math proofs



Langmuir Adsorption

Langmuir, *JACS*, 1918

Proposition

5 premises **imply** conjecture

Site balance: $S_0 = S + S_a$

Adsorption rate model: $r_{ads} = k_{ads} \cdot p \cdot S$

Desorption rate model: $r_{des} = k_{des} \cdot S_a$

Equilibrium assumption: $r_{ads} = r_{des}$

Mass balance $q = S_a$

$$\longrightarrow \quad q = \frac{S_0 K_{eq} p}{1 + K_{eq} p}$$

Theorem

Proposition is TRUE

Proof

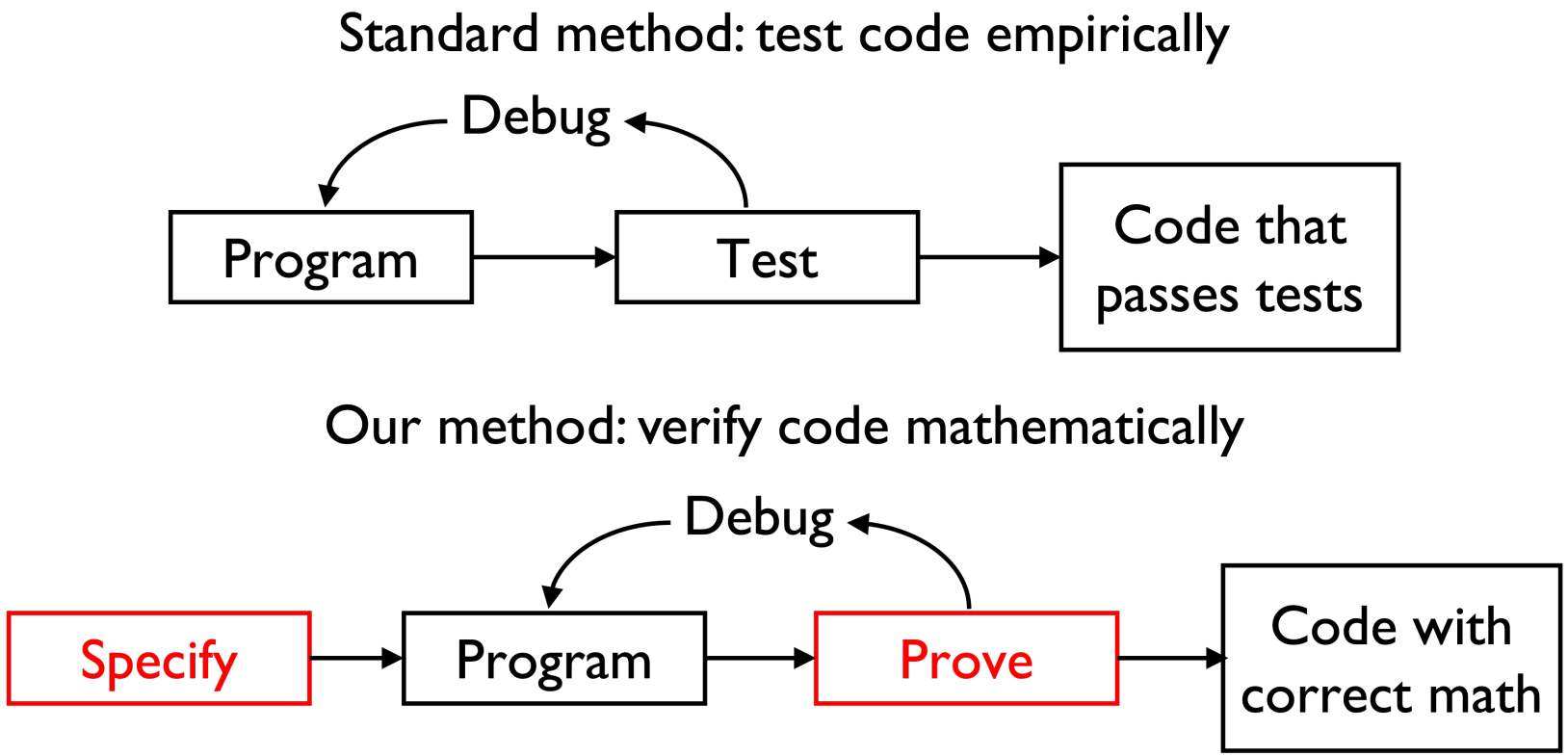
Derivation using algebraic manipulations (substitution, cancelling terms, etc.)

✓ _____
✓ _____
✓ _____
✓ _____

Slide from Lecture 1

A vision for bug-free scientific computing

Selsam, Liang, Dill, "Developing Bug-Free Machine Learning Systems with Formal Mathematics," ICML 2017.



Two kinds of math proofs

Thomas C Hales. Formal proof. Notices of the AMS, 2008.

Handwritten proofs	Formal proofs
Informal syntax	Strict, computer language syntax
Only readable for human	Machine-readable and executable
Might exclude information	Cannot miss assumptions or steps
Might contain mistakes	Rigorously verified by computer
Requires humans to proofread	Automated proof checking
Easy to write	Challenging to write

Lean theorem prover and programming language

Coquand and Huet, PhD thesis, INRIA, 1986.
de Moura, Kong, Avigad, van Doorn, von Raumer, CADE 25, 2015.

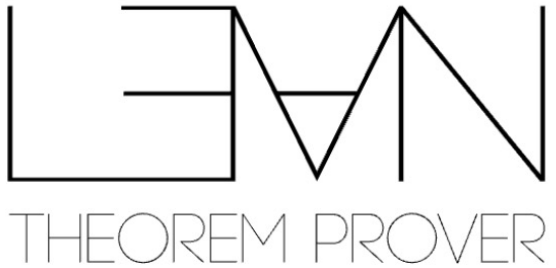
Mathematics constructed from dependent type theory

Trusted kernel with just 6k lines of code

→ >150k theorems
→ >1.5 million lines of verified proofs

Tactics to facilitate proof automation

Compile Lean code to efficient C code



"We're going to digitize mathematics, and it's going to make it better."
– Kevin Buzzard, Imperial College London

Polymorphic functions to bridge floats and reals

Adsorption data → Filter data to focus on "BET regime" → Linearize the raw data → Perform linear regression → Fitted coefficients

Floating point numbers

Polymorphic functions

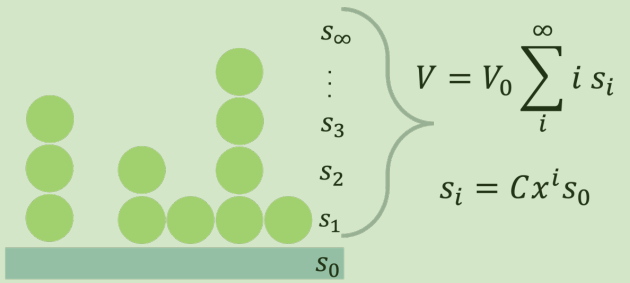
Real numbers

$\mathbb{R}$

Formal proof of BET Theory

$$q = \frac{v_m c p}{(p_0 - p)(1 + (c-1)(p/p_0))}$$

follows from a body of assumptions about



BET Adsorption

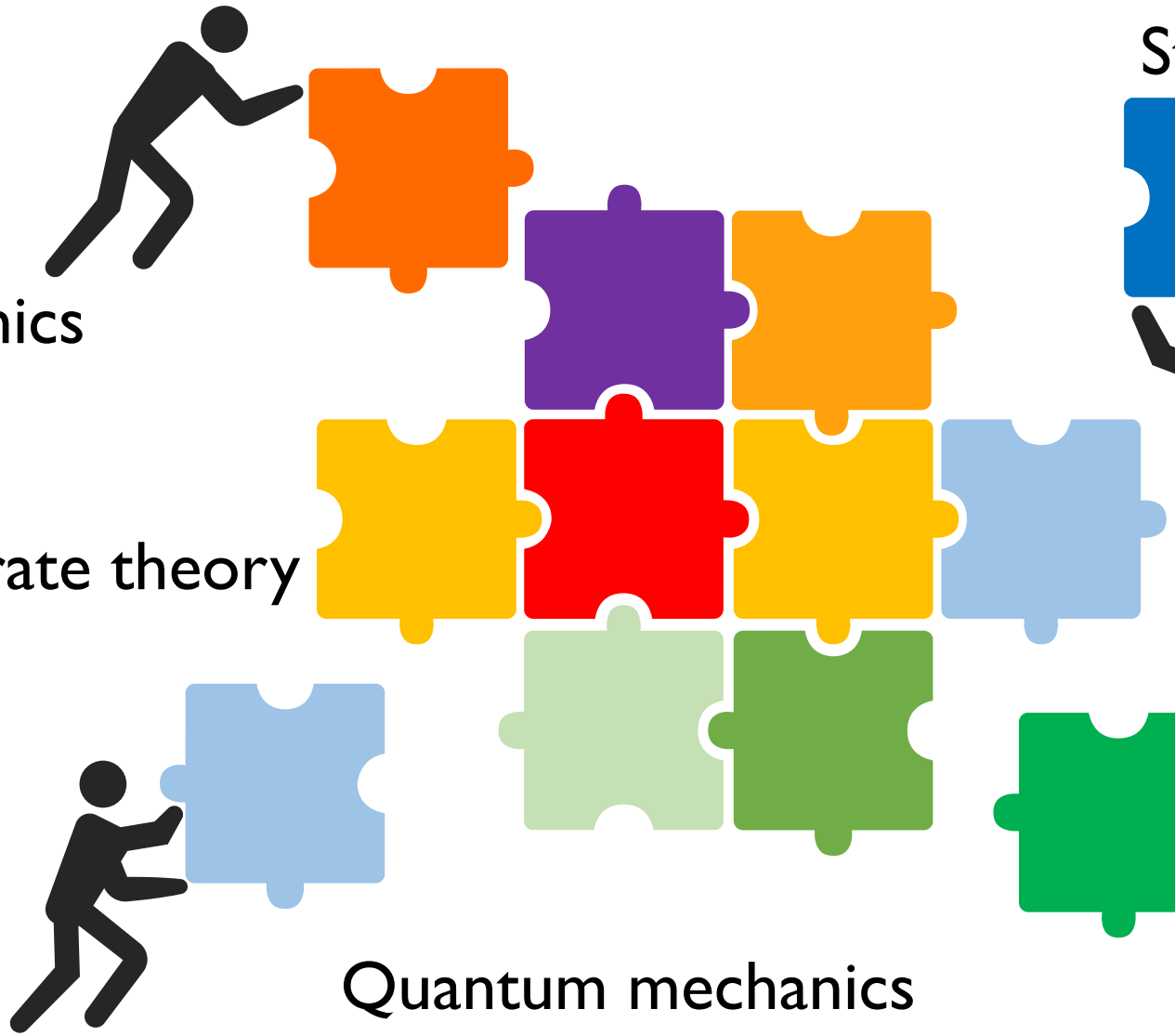
Proof that algebra for linearization is correct

Proof that linear regression minimizes least squares error

Proof that output corresponds to meaningful parameters

Slide from Lecture 1

SciLib, database of formally verified science



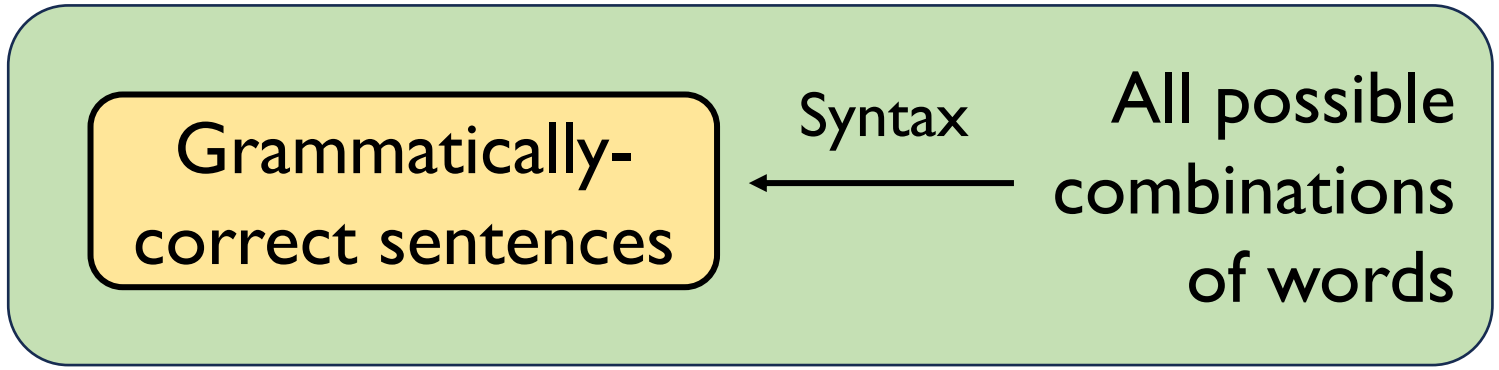
Twitter: @trjosephson
Email: tjo@umbc.edu

Errors in scientific computing software

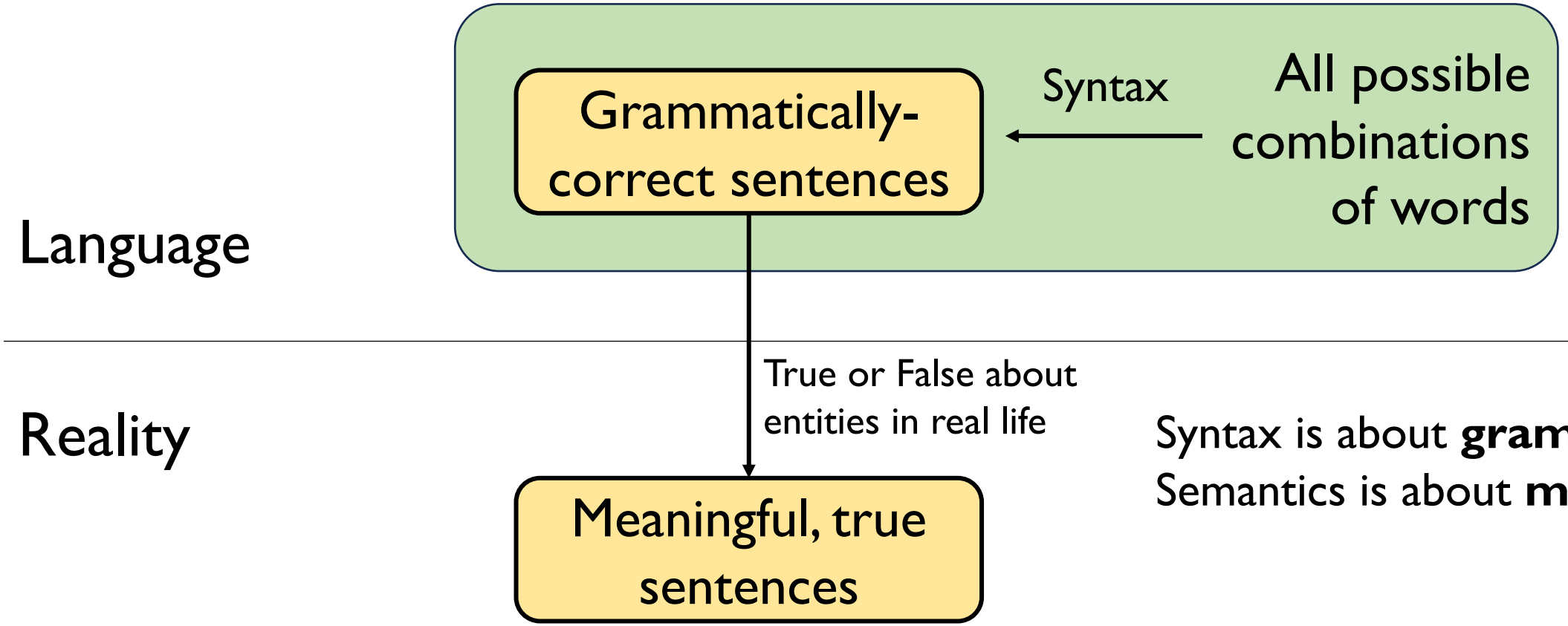
Category of error	Example	Intervention	Lean
Syntax	Not closing parentheses	Editor	Editor
Runtime	Accessing element in list that doesn't exist	Run the program, program gives error message	Editor
Semantic	Missing a minus sign, transposing tensor indices	Human inspection of the code; test-driven development; observing anomalous behavior	Editor
Floating point / Round off	Subtracting small values from large values	Checking energy conservation	

Syntax vs. semantics in *natural* language

Language



Syntax vs. semantics in *natural* language



Syntax is about **grammar**
Semantics is about **meaning**

Syntax vs. semantics in *natural* language

1. Colorless green ideas sleep furiously.

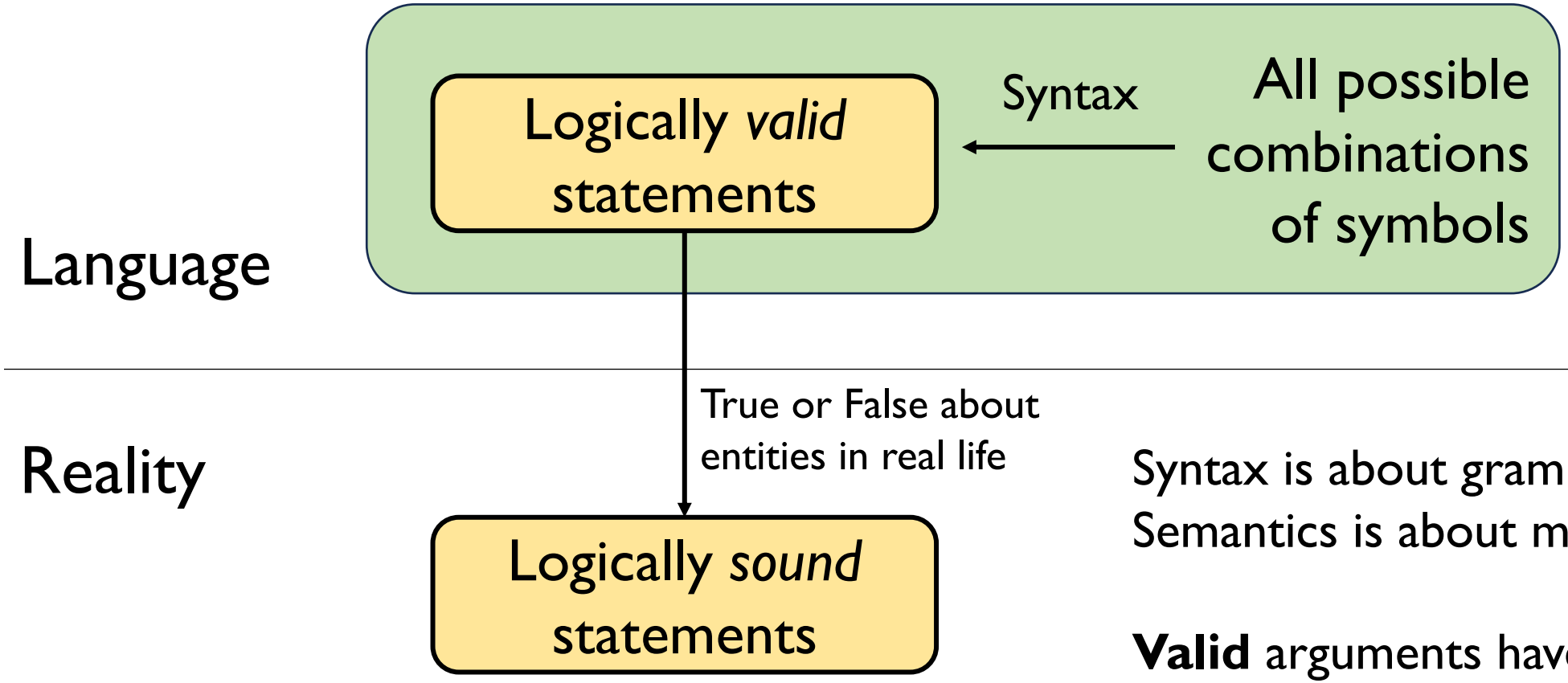
   - Semantic nonsense, while syntax is valid

2. Furiously sleep ideas green colorless.

   - Both semantically and syntactically nonsense

Example from Noam Chomsky,
Syntactic Structure, 1957
Wikipedia article

Syntax vs. semantics in *logic*

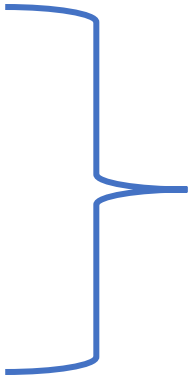


Syntax is about grammar
Semantics is about meaning

Valid arguments have correct logic
Sound arguments are valid and also have true premises

Syntax vs. semantics in logic

1. All gimbobs are woozels.
2. All woozels are wingdats.
3. Therefore, all gimbobs are wingdats.

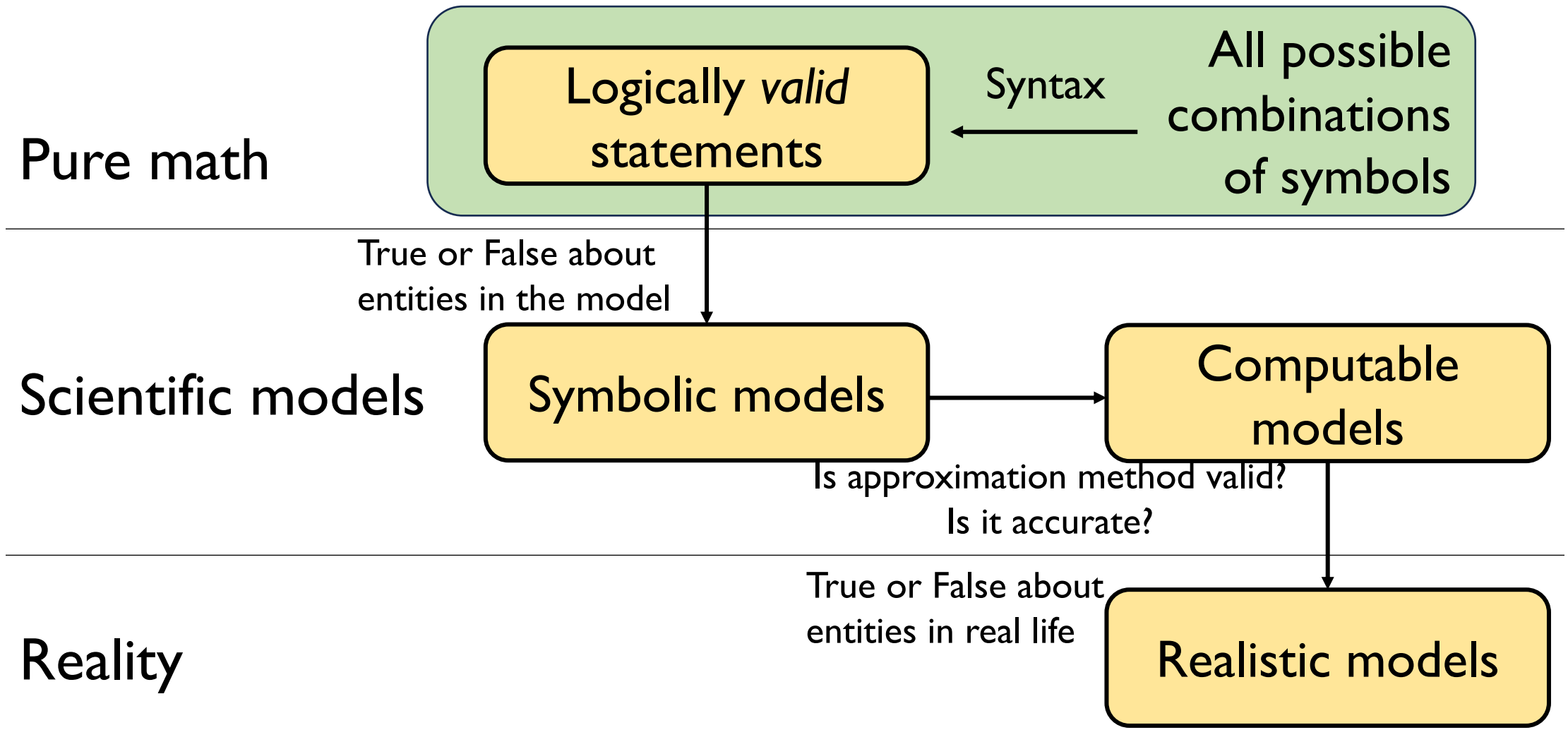


Even if gimbobs, woozels, and wingdats don't exist (semantically nonsense), the *syntax* of the argument is correct.

This is fundamental to mathematics!

When x and y are natural numbers, (x + y) = (y + x) and *it doesn't matter if x and y refer to anything in reality or not.*

Semantic errors in scientific computing



Syntax and Semantics

- To dive deeper:
  - Syntax(logic) on Wikipedia
  - Conclusions and Outlook in our paper
  - Lectures on first-order logic from Percy Liang, Stanford

Slide from Lecture 1

Proofs about equality

Additional reference: Mechanics of Proof, Chapters 1.1 and 1.2

"Calculational"-style proofs

"We solve problems which feel pretty close to high school algebra – deducing equalities/inequalities from other equalities/inequalities – using a technique which is not usually taught in high school algebra: building a single chain of expressions connecting the left-hand side with the right."

– Heather Macbeth, Mechanics of Proof

Intermission

Hypothesis in math vs. science

- Scientific methods says:
  Make hypothesis → Do experiment → Get observations → Attempt to refute the hypothesis
- Math uses the word hypothesis like "assumption" or "premise"