Applied Category Theory:
Towards a science of interdisciplinarity

David I. Spivak

TOPOS INSTITUTE

ACT4E, at ETH
2021 January 14

Outline

A road to true interdisciplinarity

- Scientific disciplines are conceptual analogies of the world.
    - Science: a schematic, conceptual account of phenomena.
    - Engineering is using these accounts to channel world events.
    - But how do different disciplines and accounts cohere?
    - To solve big problems, we need to connect different approaches.

A road to true interdisciplinarity

- Scientific disciplines are conceptual analogies of the world.
  - Science: a schematic, conceptual account of phenomena.
  - Engineering is using these accounts to channel world events.
  - But how do different disciplines and accounts cohere?
  - To solve big problems, we need to connect different approaches.
- We need a shared fabric, a substrate for interdiscipinarity.
  - Interdisciplinarity consists of effective analogy-making.
  - To go further, we need to formalize the analogies themselves.

A road to true interdisciplinarity

- Scientific disciplines are conceptual analogies of the world.
  - Science: a schematic, conceptual account of phenomena.
  - Engineering is using these accounts to channel world events.
  - But how do different disciplines and accounts cohere?
  - To solve big problems, we need to connect different approaches.
- We need a shared fabric, a substrate for interdiscipinarity.
  - Interdisciplinarity consists of effective analogy-making.
  - To go further, we need to formalize the analogies themselves.
- Better yet: we need a conceptual stem-cell.
  - Something that can differentiate into huge variety of forms.
  - Find the analogies between forms as aspects within the stem cell.

Category theory as conceptual stem-cell

Category theory (CT) can differentiate into many forms:

- All forms of pure math... (we'll briefly discuss this)

Category theory as conceptual stem-cell

Category theory (CT) can differentiate into many forms:

- All forms of pure math... (we'll briefly discuss this)
- Databases and knowledge representation (categories and functors)
- Functional programming languages (cartesian closed categories)
- Universal algebra (finite-product categories)
- Dynamical systems and fractals (operad-algebras, co-algebras)
- Hierarchical planning (lenses and monads)
- Shannon Entropy (operad of simplices)
- Partially-ordered sets and metric spaces (enriched categories)
- Higher order logic (toposes = categories of sheaves)
- Measurements of diversity in populations (magnitude of categories)
- Collaborative design (enriched categories and profunctors)
- Petri nets and chemical reaction networks (monoidal categories)
- Quantum processes and NLP (compact closed categories)

Popper's objection

"A theory that explains everything explains nothing." – Karl Popper

Popper's objection

"A theory that explains everything explains nothing." – Karl Popper

We counter this objection in two ways:

- Couldn't the same objection be made about mathematics?
  - Mathematics is the basis of hard science, used everywhere.
  - CT—like math—explains, models, formalizes many many things.
  - Conclude that math/CT explains everything and hence nothing?

Popper's objection

"A theory that explains everything explains nothing." – Karl Popper

We counter this objection in two ways:

- Couldn't the same objection be made about mathematics?
  - Mathematics is the basis of hard science, used everywhere.
  - CT—like math—explains, models, formalizes many many things.
  - Conclude that math/CT explains everything and hence nothing?
- Stem cells don't do work until they differentiate.
  - "Adult-level" work requires differentiation and optimization.
  - But the unified origins lead to impressive interoperability.

CT is the gateway to pure mathematics

CT is humanity's most powerful thought-compression language.

CT is the gateway to pure mathematics

CT is humanity's most powerful thought-compression language.

- Designed to transport theorems from one area of math to another.
  - Example: from topology (shapes) to algebra (equations).
  - This isn't mere analogy, it's analogy made rigorous.

CT is the gateway to pure mathematics

CT is humanity's most powerful thought-compression language.

- Designed to transport theorems from one area of math to another.
    - Example: from topology (shapes) to algebra (equations).
    - This isn't mere analogy, it's analogy made rigorous.
- It's revolutionized pure math since its inception in 1940s.
    - It's touched or greatly influenced all corners of mathematics.
    - It's become a gateway to learning mathematics.

CT is the gateway to pure mathematics

CT is humanity's most powerful thought-compression language.

- Designed to transport theorems from one area of math to another.
    - Example: from topology (shapes) to algebra (equations).
    - This isn't mere analogy, it's analogy made rigorous.
- It's revolutionized pure math since its inception in 1940s.
    - It's touched or greatly influenced all corners of mathematics.
    - It's become a gateway to learning mathematics.
- And it's branched out from math in a big way.
    - Databases and knowledge representation (categories and functors)
    - Functional programming languages (cartesian closed categories)
    - Hierarchical planning (lenses and monads)
    - Dynamical systems and fractals (operad-algebras, co-algebras)
    - Shannon Entropy (operad of simplices)
    - Measurements of diversity in populations (magnitude of categories)
    - Collaborative design (enriched categories and profunctors)
    - Petri nets and chemical reaction networks (monoidal categories)
    - Quantum processes and NLP (compact closed categories)

Our historical moment

Compare the information revolution to the industrial revolution:

Our historical moment

Compare the information revolution to the industrial revolution:

- Industrial revolution
  - Sewage in streets, runoff in rivers, smog in skies.
  - Uncontrolled flows of material.

Our historical moment

Compare the information revolution to the industrial revolution:

- Industrial revolution
  - Sewage in streets, runoff in rivers, smog in skies.
  - Uncontrolled flows of material.
- Information revolution
  - Big data is messy: it's gleaned, not channeled.
  - Break Humpty Dumpty into a thousand pieces, then reconstruct?
  - Uncontrolled flows of information.

Our historical moment

Compare the information revolution to the industrial revolution:

- Industrial revolution
  - Sewage in streets, runoff in rivers, smog in skies.
  - Uncontrolled flows of material.
- Information revolution
  - Big data is messy: it's gleaned, not channeled.
  - Break Humpty Dumpty into a thousand pieces, then reconstruct?
  - Uncontrolled flows of information.

I propose that CT can really help with information hygiene.

Getting to specifics

The category-theoretic stem cell is about compositional design patterns.

Getting to specifics

The category-theoretic stem cell is about compositional design patterns.

Let's focus on one.

- Operads:

Getting to specifics

The category-theoretic stem cell is about compositional design patterns.

Let's focus on one.

- Operads: the category-theoretic formalization of "operations".
- It's a branch of category theory that nicely represents the spirit.
- "Operadic": a *theme* of design patterns.

Plan of the talk

I'll discuss CT as mathematics for organizing our thinking.

- Operads, a general framework for compositional operations.
  - I'll sketch a definition and give a lot of examples.
  - Functors between operads connect different operation domains.
  - Application: solving simultaneous systems of nonlinear equations.

Plan of the talk

I'll discuss CT as mathematics for organizing our thinking.

- Operads, a general framework for compositional operations.
  - I'll sketch a definition and give a lot of examples.
  - Functors between operads connect different operation domains.
  - Application: solving simultaneous systems of nonlinear equations.
- Consider as you watch:
  - Is a hard science of interdisciplinarity possible?

Plan of the talk

I'll discuss CT as mathematics for organizing our thinking.

- Operads, a general framework for compositional operations.
  - I'll sketch a definition and give a lot of examples.
  - Functors between operads connect different operation domains.
  - Application: solving simultaneous systems of nonlinear equations.
- Consider as you watch:
  - Is a hard science of interdisciplinarity possible?
  - What sort of mathematical theory would it require?

Plan of the talk

I'll discuss CT as mathematics for organizing our thinking.

- Operads, a general framework for compositional operations.
  - I'll sketch a definition and give a lot of examples.
  - Functors between operads connect different operation domains.
  - Application: solving simultaneous systems of nonlinear equations.
- Consider as you watch:
  - Is a hard science of interdisciplinarity possible?
  - What sort of mathematical theory would it require?
  - Does Category Theory fit the bill?

Plan of the talk

I'll discuss CT as mathematics for organizing our thinking.

- Operads, a general framework for compositional operations.
  - I'll sketch a definition and give a lot of examples.
  - Functors between operads connect different operation domains.
  - Application: solving simultaneous systems of nonlinear equations.
- Consider as you watch:
  - Is a hard science of interdisciplinarity possible?
  - What sort of mathematical theory would it require?
  - Does Category Theory fit the bill?

Let's get into it with *operads*, a framework for compositional operations.

Outline

What are compositional operations?

Operations take arrangements of many sorts and produce one sort.

What are compositional operations?

Operations take arrangements of many sorts and produce one sort.

An operad consists of:

- A collection of sorts $X, Y, \ldots,$
- And ways to arrange them, $\varphi\colon X_1, \ldots, X_k \to Y,$
- Such that arrangements can be nested inside each other.

What are compositional operations?

Operations take arrangements of many sorts and produce one sort.

An operad consists of:

- A collection of sorts $X, Y, \ldots,$
- And ways to arrange them, $\varphi \colon X_1, \ldots, X_k \to Y$,
- Such that arrangements can be nested inside each other. (That last part is the compositionality.)

What are compositional operations?

Operations take arrangements of many sorts and produce one sort.

An operad consists of:

- A collection of sorts $X, Y, \ldots$,
- And ways to arrange them, $\varphi\colon X_1, \ldots, X_k \to Y$,
- Such that arrangements can be nested inside each other.
  (That last part is the compositionality.)

Slightly more formal definition to come.

Operads are everywhere

Operads are implicitly being used in many fields.

- Electrical engineering: “wiring diagrams”
- Design: “set-based design”
- Computer programming: “data flow”
- Natural language processing: “grammars”
- Materials science: “hierarchical materials”
- Information theory: “Shannon entropy”

Operads are everywhere

Operads are implicitly being used in many fields.

- Electrical engineering: “wiring diagrams”
- Design: “set-based design”
- Computer programming: “data flow”
- Natural language processing: “grammars”
- Materials science: “hierarchical materials”
- Information theory: “Shannon entropy”

We want to bring operads to the fore.

- There’s a common theme in the way we think.
- Operads structure this sort of thinking.
- With mathematical structure, we can go much further.

Operads are everywhere

Operads are implicitly being used in many fields.

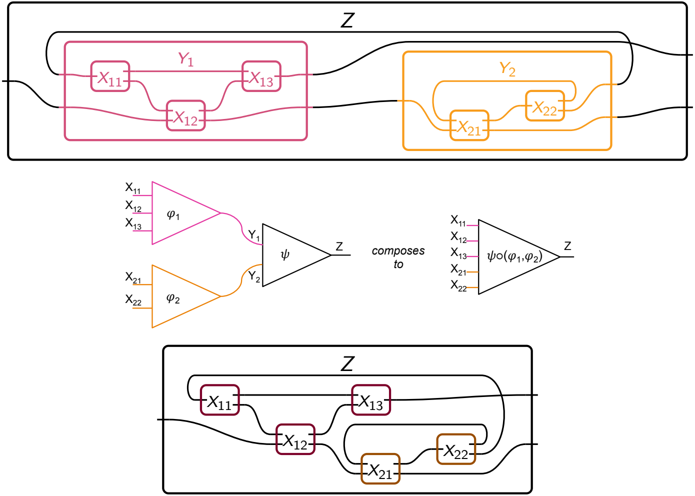
- Electrical engineering: "wiring diagrams"
- Design: "set-based design"
- Computer programming: "data flow"
- Natural language processing: "grammars"
- Materials science: "hierarchical materials"
- Information theory: "Shannon entropy"

We want to bring operads to the fore.

- There's a common theme in the way we think.
- Operads structure this sort of thinking.
- With mathematical structure, we can go much further.

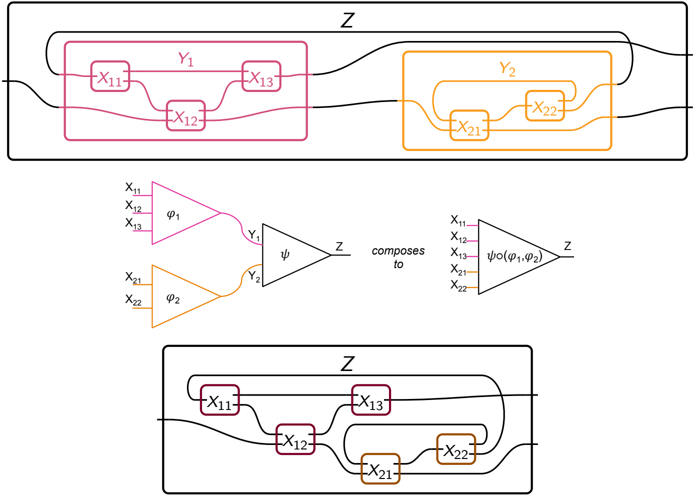
Let's look for sorts, arrangements, and nesting in some examples.

Operad 1: wiring diagrams



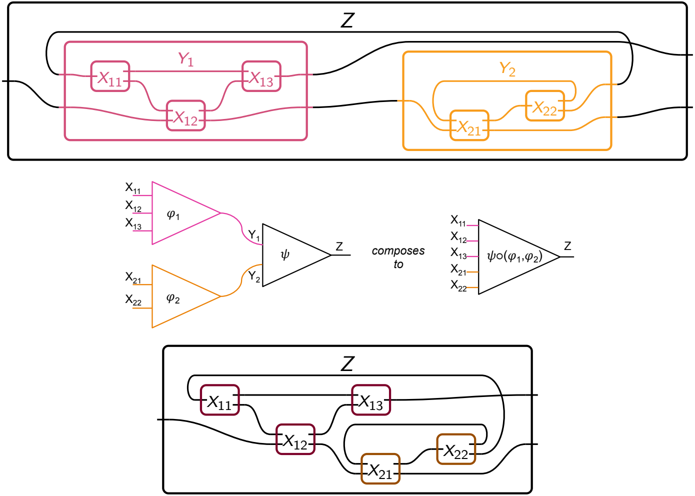
Sorts: boxes with ports.

Operad 1: wiring diagrams



Sorts: boxes with ports. Arrangements: wiring diagrams.

Operad 1: wiring diagrams



Sorts: boxes with ports. Arrangements: wiring diagrams. Nesting: nesting.

Formal definition of operad

An operad $\mathcal{O}$ consists of

- A set $\mathrm{Ob}(\mathcal{O})$, elements of which are called *sorts*.
- For sorts $X_1, \ldots, X_k, Y \in \mathrm{Ob}(\mathcal{O})$, a set

  $$\mathrm{Mor}_{\mathcal{O}}(X_1, \ldots, X_k; Y)$$

  Its elements are called *morphisms* or arrangements of $X_1, \ldots, X_k$ in Y. A k-ary arrangement $\varphi \in \mathrm{Mor}_{\mathcal{O}}(X_1, \ldots, X_k; Y)$ may be denoted

  $$\varphi \colon (X_1, \ldots, X_k) \to Y.$$

- For each sort $X \in \mathrm{Ob}(\mathcal{O})$, an identity arrangement $\mathrm{id}_X \colon (X) \to X$.
- A composition, or nesting formula, e.g.,

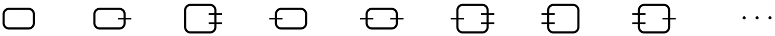
  $$\psi \circ (\varphi_1, \ldots, \varphi_k) \colon (X_{i,j}) \xrightarrow{\varphi_i} (Y_i) \xrightarrow{\psi} Z.$$

These are required to satisfy well-known “unital” and “associative” laws.

Operad 1: WDs again

An operad $\mathcal{W}$ for composing wiring diagrams:

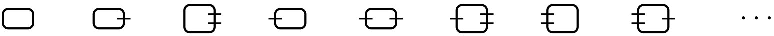
- Sort $X \in \mathcal{W}$: any possible box-with-ports.



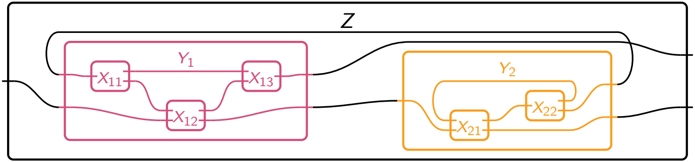
Operad 1: WDs again

An operad $\mathcal{W}$ for composing wiring diagrams:

- Sort $X \in \mathcal{W}$: any possible box-with-ports.



- Arrangement $\varphi \colon X_1, \ldots, X_k \to Y$ in $\mathcal{W}$: any wiring of X's in Y.
- Nesting: the facts about this fractal of wiring possibilities.



$\mathcal{W}$ provides fine-grained control of flow operators and operations.

Operad 2: hierarchical protein materials

There is an operad $\mathcal{M}$ for composing hierarchical protein materials.

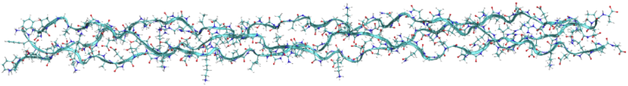
- Why protein materials?
  - Protein materials include your skin: stretchable, breathable, waterproof.
  - Eat hamburgers, make amazing material.
  - Materials scientists would *love* to make materials like this.

[1] Giesa, T.; Jagadeesan, R.; Spivak, D.I.; Buehler, M.J. (2015) "Matriarch: a Python library for materials architecture." *ACS Biomaterials Science & Engineering*.

Operad 2: hierarchical protein materials

There is an operad $\mathcal{M}$ for composing hierarchical protein materials.

- Why protein materials?
  - Protein materials include your skin: stretchable, breathable, waterproof.
  - Eat hamburgers, make amazing material.
  - Materials scientists would *love* to make materials like this.
- A protein is an arrangement of simpler proteins.
  - There are "atomic" proteins: amino acids.
  - arrange in series or parallel (H-bonds), or
  - arrange in helices, double helices, any conceivable curve, etc.



- Collagen has a nested structure: it is an array, each fiber of which is a triple helix, each strand of which is a helix, each unit of which is an amino acid.[1]

[1] Giesa, T.; Jagadeesan, R.; Spivak, D.I.; Buehler, M.J. (2015) "Matriarch: a Python library for materials architecture." *ACS Biomaterials Science & Engineering*.

Operad 3: probabilities

Even an operad with one sort can be interesting.

- Consider the operad $\mathcal{P}$ for "probabilities".

Operad 3: probabilities

Even an operad with one sort can be interesting.

- Consider the operad $\mathcal{P}$ for "probabilities".
- Say sorts=$\{$event$\}$, i.e. "event" is the only sort in $\mathcal{P}$.
- Arrangements $=$ probability distributions. Nesting $=$ weighted sums.

Operad 3: probabilities

Even an operad with one sort can be interesting.

- Consider the operad $\mathcal{P}$ for "probabilities".
- Say sorts={event}, i.e. "event" is the only sort in $\mathcal{P}$.
- Arrangements = probability distributions. Nesting = weighted sums.
- Formally: $\mathcal{P}_k := \big\{(x_1, \ldots, x_k) \in \mathbb{R}^k_{\geq 0} \bigm| x_1 + \cdots + x_k = 1\big\}$.

Operad 3: probabilities

Even an operad with one sort can be interesting.

- Consider the operad $\mathcal{P}$ for "probabilities".
- Say sorts={event}, i.e. "event" is the only sort in $\mathcal{P}$.
- Arrangements = probability distributions. Nesting = weighted sums.
- Formally: $\mathcal{P}_k := \left\{(x_1, \dots, x_k) \in \mathbb{R}^k_{\geq 0} \;\middle|\; x_1 + \cdots + x_k = 1\right\}$.

Arrangement: "In this event, there's a distribution on next events."

Operad 3: probabilities

Even an operad with one sort can be interesting.

- Consider the operad $\mathcal{P}$ for "probabilities".
- Say sorts={event}, i.e. "event" is the only sort in $\mathcal{P}$.
- Arrangements = probability distributions. Nesting = weighted sums.
- Formally: $\mathcal{P}_k := \left\{(x_1, \ldots, x_k) \in \mathbb{R}^k_{\geq 0} \;\middle|\; x_1 + \cdots + x_k = 1\right\}$.

Arrangement: "In this event, there's a distribution on next events."

- coin flip: $f = (\frac{1}{2}, \frac{1}{2}) \in \mathcal{P}_2$.
  - In the event coin flip, there's a 50-50 distribution on next events.
- die roll: $r = (\frac{1}{6}, \ldots, \frac{1}{6}) \in \mathcal{P}_6$.
- card selection: $p = (\frac{1}{52}, \ldots, \frac{1}{52}) \in \mathcal{P}_{52}$.

The nesting rule composes distributions by weighted sum:

- Flip a coin: result decides whether to roll a die or pick a card.

$$f \circ (r, p) = \left(\underbrace{\frac{1}{12}, \ldots, \frac{1}{12}}_{\text{6 times}}, \underbrace{\frac{1}{104}, \ldots, \frac{1}{104}}_{\text{52 times}} \right) \in \mathcal{P}_{58}$$

A zoo of operads: Grammars

Any context-free grammar is an operad.

$$
\begin{array}{rcl}
\langle\text{sentence}\rangle & ::= & \langle\text{noun-phrase}\rangle\ \langle\text{verb-phrase}\rangle \\
\langle\text{noun-phrase}\rangle & ::= & \langle\text{pronoun}\rangle \mid \langle\text{proper-noun}\rangle \mid \langle\text{determiner}\rangle\ \langle\text{nominal}\rangle \\
\langle\text{nominal}\rangle & ::= & \langle\text{noun}\rangle \mid \langle\text{nominal}\rangle\ \langle\text{noun}\rangle \\
\langle\text{verb-phrase}\rangle & ::= & \langle\text{verb}\rangle \mid \langle\text{verb}\rangle\ \langle\text{noun-phrase}\rangle \mid \langle\text{verb}\rangle\ \langle\text{prep-phrase}\rangle \\
\langle\text{prep-phrase}\rangle & ::= & \langle\text{preposition}\rangle\ \langle\text{noun-phrase}\rangle
\end{array}
$$

How is this an operad?

- The sorts are the parts of speech.
- The arrangements are the production rules.
- Nesting is nesting.

The operad of sets

There is an operad **Set** of sets and functions.

- The sorts in **Set** are all the sets, e.g. $\{\mathtt{r}, \mathtt{g}, \mathtt{b}\}$, $\mathbb{N}$, $\mathbb{R}^2$, etc.
- An arrangement is a function $f\colon X_1 \times \cdots \times X_k \to Y$.
- Nesting is composition $f \circ (g_1, \ldots, g_k)$.

The operad of sets

There is an operad **Set** of sets and functions.

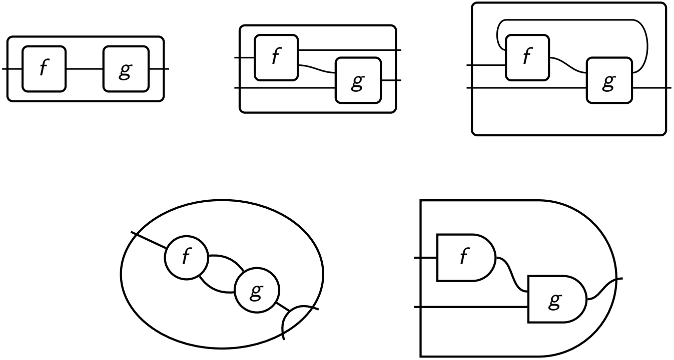
- The sorts in **Set** are all the sets, e.g. $\{\mathtt{r}, \mathtt{g}, \mathtt{b}\}$, $\mathbb{N}$, $\mathbb{R}^2$, etc.
- An arrangement is a function $f\colon X_1 \times \cdots \times X_k \to Y$.
- Nesting is composition $f \circ (g_1, \ldots, g_k)$.

Think of sets as containing features one can sort into.

- An arrangement sorts a k-dimensional feature space into one dimension.

Wiring diagram operads, similar to #1

A representative arrangement $\varphi\colon X, Y \to Z$ in five different WD operads.



Outline

Functors translate between operads

We've seen many operads:

- wiring diagrams,
- protein materials,
- sets,
- probabilities,
- grammars.

Functors translate between operads

We've seen many operads:

- wiring diagrams,
- protein materials,
- sets,
- probabilities,
- grammars.

A functor $F\colon \mathcal{O} \to \mathcal{O}'$ is a translator from one operad to another.

- Functors between grammars are like compilers, or elaborations.
- Functors between wiring diagram operads are sub-languages.
- Functors $\mathcal{O} \to$ **Set** are special: they're called $\mathcal{O}$-*algebras*.

Functors translate between operads

We've seen many operads:

- wiring diagrams,
- protein materials,
- sets,
- probabilities,
- grammars.

A functor $F\colon \mathcal{O} \to \mathcal{O}'$ is a translator from one operad to another.

- Functors between grammars are like compilers, or elaborations.
- Functors between wiring diagram operads are sub-languages.
- Functors $\mathcal{O} \to \mathbf{Set}$ are special: they're called *$\mathcal{O}$-algebras*.

Algebras *operationalize* operads.

Operads and their algebras

Rules of composition vs. stuff being composed.

Operads and their algebras

Rules of composition vs. stuff being composed.

- Operad : Group theory :: Algebras : Groups.
- Operad : Ring theory :: Algebras : Rings.

Operads and their algebras

Rules of composition vs. stuff being composed.

- Operad : Group theory :: Algebras : Groups.
- Operad : Ring theory :: Algebras : Rings.

Operad = theory. Algebras = models.

Each operad has many algebras

Each operad $\mathcal{O}$ is a "theory of composition".

- Sorts $X, Y, \ldots$: what *sorts* of elements in this theory?
- Arrangements φ, ψ: what are the operations?
- Nesting: what kind of laws?

Each operad has many algebras

Each operad $\mathcal{O}$ is a "theory of composition".

- Sorts $X, Y, \ldots$: what *sorts* of elements in this theory?
- Arrangements φ, ψ: what are the operations?
- Nesting: what kind of laws?

The $\mathcal{O}$-algebras $A\colon \mathcal{O} \to \mathbf{Set}$ are the models of theory $\mathcal{O}$.

- An $\mathcal{O}$-algebra A says what's actually being composed.
  - To each sort X: a set $A(X)$ of elements.
  - To each arrangement φ: a k-ary operation $A(\varphi)$.
    - If $\varphi\colon X_1, \ldots, X_k \to Y$ is an arrangement,
    - Then $A(\varphi)\colon A(X_1) \times \cdots \times A(X_k) \to A(Y)$ is a function.
  - To each nesting: a law in A.

Each operad has many algebras

Each operad $\mathcal{O}$ is a "theory of composition".

- Sorts $X, Y, \ldots$: what *sorts* of elements in this theory?
- Arrangements φ, ψ: what are the operations?
- Nesting: what kind of laws?

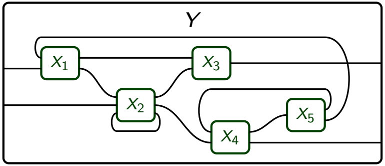
The $\mathcal{O}$-algebras $A\colon \mathcal{O} \to \mathbf{Set}$ are the models of theory $\mathcal{O}$.

- An $\mathcal{O}$-algebra A says what's actually being composed.
    - To each sort X: a set $A(X)$ of elements.
    - To each arrangement φ: a k-ary operation $A(\varphi)$.
        - If $\varphi\colon X_1, \ldots, X_k \to Y$ is an arrangement,
        - Then $A(\varphi)\colon A(X_1) \times \cdots \times A(X_k) \to A(Y)$ is a function.
    - To each nesting: a law in A.

Next, I'll explain what algebras on an operad look like.

Multiple algebras for wiring diagrams operad

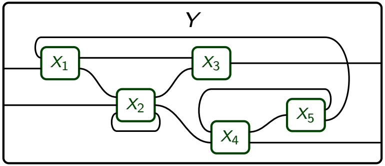
Recall operad $\mathcal{W}$ of all boxes X and WDs, $\varphi\colon X_1, \ldots, X_5 \to Y$.[2]



[2] Imagine each port / wire labeled by a topological space: the signal space.

Multiple algebras for wiring diagrams operad

Recall operad $\mathcal{W}$ of all boxes X and WDs, $\varphi\colon X_1, \ldots, X_5 \to Y$.[2]

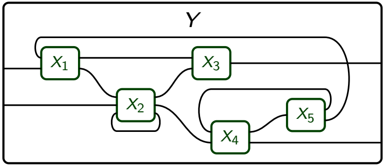


There are many different $\mathcal{W}$-algebras $A\colon \mathcal{W} \to \mathbf{Set}$.

[2] Imagine each port / wire labeled by a topological space: the signal space.

Multiple algebras for wiring diagrams operad

Recall operad $\mathcal{W}$ of all boxes X and WDs, $\varphi\colon X_1, \ldots, X_5 \to Y$.[2]



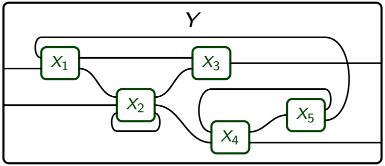
There are many different $\mathcal{W}$-algebras $A\colon \mathcal{W} \to \mathbf{Set}$.

- Open dynamical systems: machines with ports.
  - $A(X)$: the set of all DS's with input-output shape X.
  - $A(\varphi)$: a certain variable-sharing formula.

[2] Imagine each port / wire labeled by a topological space: the signal space.

Multiple algebras for wiring diagrams operad

Recall operad $\mathcal{W}$ of all boxes X and WDs, $\varphi\colon X_1, \ldots, X_5 \to Y$.[2]



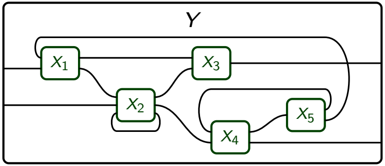
There are many different $\mathcal{W}$-algebras $A\colon \mathcal{W} \to \mathbf{Set}$.

- Open dynamical systems: machines with ports.
  - $A(X)$: the set of all DS's with input-output shape X.
  - $A(\varphi)$: a certain variable-sharing formula.
- In fact, dynamical systems comprise several algebras:
  - A_1: continuous DS's (ODE's with time-varying parameters).
  - A_2: discrete DS's (on a discrete clock).
  - A_3: hybrid DS's (cyber-physical systems).

[2] Imagine each port / wire labeled by a topological space: the signal space.

Multiple algebras for wiring diagrams operad

Recall operad $\mathcal{W}$ of all boxes X and WDs, $\varphi\colon X_1,\dots,X_5 \to Y$.[2]



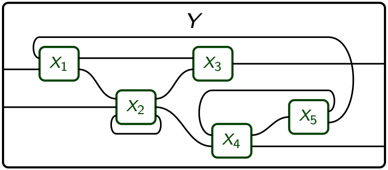
There are many different $\mathcal{W}$-algebras $A\colon \mathcal{W} \to \mathbf{Set}$.

- Open dynamical systems: machines with ports.
  - $A(X)$: the set of all DS's with input-output shape X.
  - $A(\varphi)$: a certain variable-sharing formula.
- In fact, dynamical systems comprise several algebras:
  - A_1: continuous DS's (ODE's with time-varying parameters).
  - A_2: discrete DS's (on a discrete clock).
  - A_3: hybrid DS's (cyber-physical systems).
- But there's a much simpler "algebraic" algebra...

[2] Imagine each port / wire labeled by a topological space: the signal space.

$\mathcal{W}$-algebra of tensor networks

We said $\mathcal{W}$ is modeled by dynamical systems.



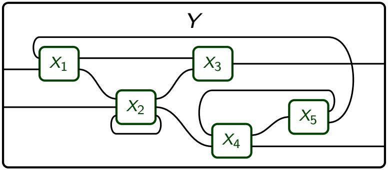
Another algebra $\mathcal{W} \to \mathbf{Set}$: tensors (not dynamic at all).[3]

- Box = tensor format ($T \in V_1 \otimes \cdots \otimes V_n$).
- Wiring diagram = tensor network.
- Contract along shared wires to "compose".

[3] Imagine each wire labeled by a natural number: the dimension.

$\mathcal{W}$-algebra of tensor networks

We said $\mathcal{W}$ is modeled by dynamical systems.



Another algebra $\mathcal{W} \to \mathbf{Set}$: tensors (not dynamic at all).[3]

- Box = tensor format ($T \in V_1 \otimes \cdots \otimes V_n$).
- Wiring diagram = tensor network.
- Contract along shared wires to "compose".

Later: a relationship between dynamical systems and tensors...

- ... which led me to a numerical method for solving systems.
- The pixel array method, up next.

[3] Imagine each wire labeled by a natural number: the dimension.

Outline

Detailed look at a real-world application

Separately plot the solutions to equations: $f(x, w) = 0$ and $g(w, y) = 0$.

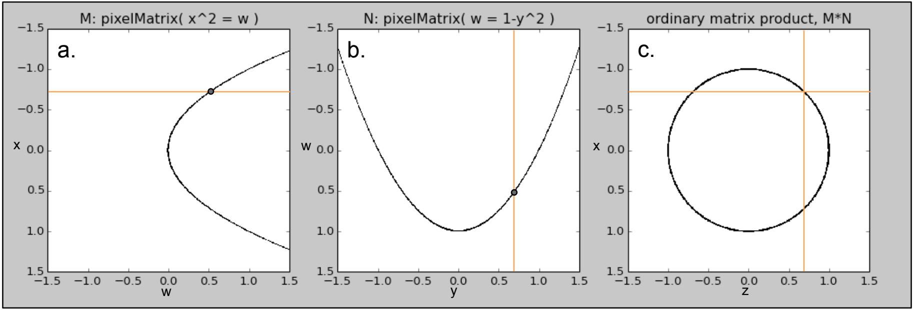
- Plot each in a bounding box, e.g. $[-1.5, 1.5]$.
- Consider plots as matrices of on/off pixels (booleans).
- Multiply matrices to solve system.

Detailed look at a real-world application

Separately plot the solutions to equations: $f(x, w) = 0$ and $g(w, y) = 0$.

- Plot each in a bounding box, e.g. $[-1.5, 1.5]$.
- Consider plots as matrices of on/off pixels (booleans).
- Multiply matrices to solve system.

Example: $x^2 = w$ and $w = 1 - y^2$.



A more complex example

The following eq's are not differentiable, nor even defined everywhere.

$$\cos\left(\ln(z^2 + 10^{-3}x)\right) - x + 10^{-5}z^{-1} = 0 \qquad \text{(Equation 1)}$$

$$\cosh(w + 10^{-3}y) + y + 10^{-4}w = 2 \qquad \text{(Equation 2)}$$

$$\tan(x + y)(x - 2)^{-1}(x + 3)^{-1}y^{-2} = 1 \qquad \text{(Equation 3)}$$

Q: For what values of w and z does a simultaneous solution exist? [4]

[4] Spivak; Dobson; Kumari; Wu (2016) "Pixel Arrays: A fast and elementary method for solving nonlinear systems". `https://arxiv.org/abs/1609.00061`

A more complex example

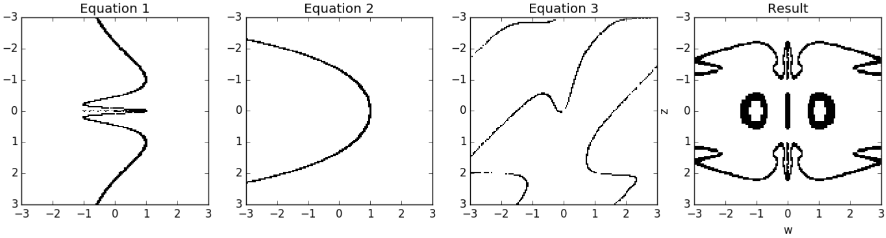
The following eq's are not differentiable, nor even defined everywhere.

$$\cos\left(\ln(z^2 + 10^{-3}x)\right) - x + 10^{-5}z^{-1} = 0 \qquad \text{(Equation 1)}$$

$$\cosh(w + 10^{-3}y) + y + 10^{-4}w = 2 \qquad \text{(Equation 2)}$$

$$\tan(x + y)(x - 2)^{-1}(x + 3)^{-1}y^{-2} = 1 \qquad \text{(Equation 3)}$$

Q: For what values of w and z does a simultaneous solution exist? [4]



[4] Spivak; Dobson; Kumari; Wu (2016) "Pixel Arrays: A fast and elementary method for solving nonlinear systems". https://arxiv.org/abs/1609.00061

Equations and wiring diagrams

Consider an arbitrary system of equations having the following form:

$$
\begin{aligned}
f_1(\mathbf{t}, u, \mathbf{v}) &= 0 \\
f_2(\mathbf{v}, w, x) &= 0 \\
f_3(u, w, x, y) &= 0 \\
f_4(x, \mathbf{z}) &= 0
\end{aligned}
$$

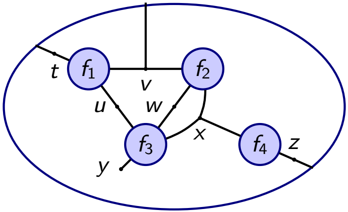
Bold variables are those we want to *expose*; others are *unexposed*.

Equations and wiring diagrams

Consider an arbitrary system of equations having the following form:

$$\begin{aligned} f_1(\mathbf{t}, u, \mathbf{v}) &= 0 \\ f_2(\mathbf{v}, w, x) &= 0 \\ f_3(u, w, x, y) &= 0 \\ f_4(x, \mathbf{z}) &= 0 \end{aligned}$$

Bold variables are those we want to *expose*; others are *unexposed*.

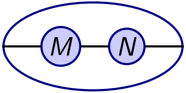


Said another way, we want $\{(t, v, z) \mid \exists u, w, x, y : f_1 = f_2 = f_3 = f_4 = 0\}$.

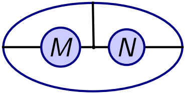
Example wiring diagrams for named operations

Some famous matrix products as wiring diagrams:

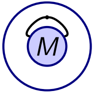
Multiplication: MN



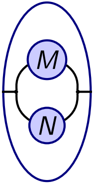
Khatri-Rao: $M \odot N$



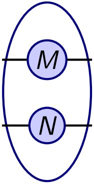
Trace: $\mathrm{Tr}(M)$



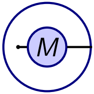
Hadamard: $M \circ N$



Kronecker: $M \otimes N$



Marginalize: $\sum_i M_{i,j}$



Speed and accuracy

- The PA method has good accuracy guarantees.
  - No false negatives, only false positives.
  - As pixel density $\to \infty$, false positive $\to 0$
  - Rate is proportional to size of largest derivative.
- The PA method is faster than Newton-style methods....
  - If you want all solutions in a bounding box...
- Applications: plot steady states of dynamical systems.

PA method for dynamical systems

"We can use the PA method to find the steady states of dynamic systems."

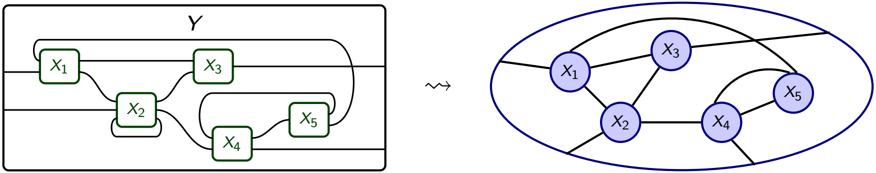
- A statement like that usually means "One can be trained to..."
- We want it to mean there is a *formal, mathematical connection*.

PA method for dynamical systems

"We can use the PA method to find the steady states of dynamic systems."

- A statement like that usually means "One can be trained to..."
- We want it to mean there is a *formal, mathematical connection*.

There is a functor comparing wiring diagram operads.



And we want to translate dynamical systems to pixel arrays.

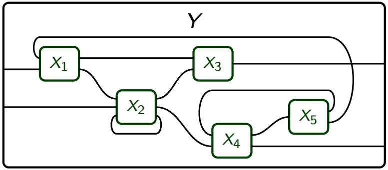
$$\begin{tikzcd}
\mathcal{W}_{\text{box}} \arrow[rr, "\text{as shown above}"] \arrow[dr, "\text{DynSys}"'] & ? & \mathcal{W}_{\text{circle}} \arrow[dl, "\text{PixArray}"] \\
& \mathbf{Set} &
\end{tikzcd}$$

PA method for dynamical systems

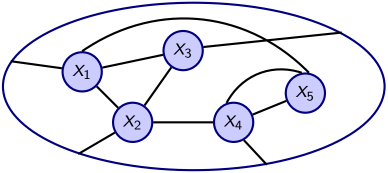
"We can use the PA method to find the steady states of dynamic systems."

- A statement like that usually means "One can be trained to..."
- We want it to mean there is a *formal, mathematical connection*.

There is a functor comparing wiring diagram operads.



$\rightsquigarrow$



And we want to translate dynamical systems to pixel arrays.

$$
\begin{tikzcd}
\mathcal{W}_{\text{box}} \arrow[rr, "\text{as shown above}"] \arrow[dr, "\text{DynSys}"'] \arrow[rr, phantom, shift right=6, "\overset{\text{steady state}}{\Longrightarrow}"] & & \mathcal{W}_{\text{circle}} \arrow[dl, "\text{PixArray}"] \\
& \mathbf{Set} &
\end{tikzcd}
$$

Application: smart grid

NIST has used the PA to solve smart grid power flow problems.

Application: smart grid

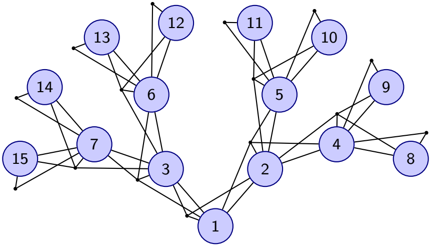
NIST has used the PA to solve smart grid power flow problems.

- Power network model: a binary tree of nodes ("buses")
    - Each node i has four associated variables: P_i, Q_i, V_i, θ_i.
    - For each i, two equations $P_i = \sum_j P_{ij}(V_i, V_j, \theta_i, \theta_j)$, $Q_i = \cdots$
    - The sums are taken over all adjacent nodes.

Application: smart grid

NIST has used the PA to solve smart grid power flow problems.

- Power network model: a binary tree of nodes ("buses")
  - Each node i has four associated variables: P_i, Q_i, V_i, θ_i.
  - For each i, two equations $P_i = \sum_j P_{ij}(V_i, V_j, \theta_i, \theta_j)$, $Q_i = \cdots$
  - The sums are taken over all adjacent nodes.
- This is perfect for the PA method.



- Given parallel resources, runtime is logarithmic in # nodes.

Outline

What just happened?

Was that math or a David Lynch film?

What just happened?

Was that math or a David Lynch film?

Recapping the math:

- We discussed operads.
  - An operad $\mathcal{O}$ is a language for composable operations.
  - Wiring diagrams, grammars, protein materials, probabilities, sets.

What just happened?

Was that math or a David Lynch film?

Recapping the math:

- We discussed operads.
  - An operad $\mathcal{O}$ is a language for composable operations.
  - Wiring diagrams, grammars, protein materials, probabilities, sets.
- We discussed functors $F\colon \mathcal{O} \to \mathcal{O}'$ between operads.
  - Translate from one composition language to another.
  - $\mathcal{O}$-algebras are functors $\mathcal{O} \to \mathbf{Set}$.
  - Dynamical systems, tensors, systems of equations.

What just happened?

Was that math or a David Lynch film?

Recapping the math:

- We discussed operads.
  - An operad $\mathcal{O}$ is a language for composable operations.
  - Wiring diagrams, grammars, protein materials, probabilities, sets.
- We discussed functors $F\colon \mathcal{O} \to \mathcal{O}'$ between operads.
  - Translate from one composition language to another.
  - $\mathcal{O}$-algebras are functors $\mathcal{O} \to \mathbf{Set}$.
  - Dynamical systems, tensors, systems of equations.
- We briefly discussed maps between functors.
  - Steady states of dynamical systems arranged as tensors.
  - (Another example: discretization of cont's dynamical systems.)

Ok, but why did that happen?

The point of all that was to give a glimpse into category theory.

- A simple principle—operations—formalized mathematically.
    - We saw several examples of this principle.
    - We saw a web of interconnections between different examples.

Ok, but why did that happen?

The point of all that was to give a glimpse into category theory.

- A simple principle—operations—formalized mathematically.
  - We saw several examples of this principle.
  - We saw a web of interconnections between different examples.
- And this ‘operations’ stuff is just one part of category theory.
  - CT has formalized the principles of mathematics, in mathematics.
  - Space, measure, operation, symmetry, equivalence, syntax.
  - There is a shared fabric interconnecting all these principles.

Ok, but why did that happen?

The point of all that was to give a glimpse into category theory.

- A simple principle—operations—formalized mathematically.
  - We saw several examples of this principle.
  - We saw a web of interconnections between different examples.
- And this ‘operations’ stuff is just one part of category theory.
  - CT has formalized the principles of mathematics, in mathematics.
  - Space, measure, operation, symmetry, equivalence, syntax.
  - There is a shared fabric interconnecting all these principles.

There's even a category of categories and an operad of operads.

Science of interdisciplinarity

A science of interdisciplinarity can reduce waste.

- Models of different systems need to be independent to optimize.
- They need to be interoperable to work in concert for larger purpose.
- Interdisciplinarity must respect both aspects.

Science of interdisciplinarity

A science of interdisciplinarity can reduce waste.

- Models of different systems need to be independent to optimize.
- They need to be interoperable to work in concert for larger purpose.
- Interdisciplinarity must respect both aspects.

What is the right mathematics to underlie a science of interdisciplinarity?

- Category theory has an impressive record inside of math and out.
- It's been recently highlighted by agencies such as NIST and DARPA.
- We still need to prove it works by using it in practice.

Science of interdisciplinarity

A science of interdisciplinarity can reduce waste.

- Models of different systems need to be independent to optimize.
- They need to be interoperable to work in concert for larger purpose.
- Interdisciplinarity must respect both aspects.

What is the right mathematics to underlie a science of interdisciplinarity?

- Category theory has an impressive record inside of math and out.
- It's been recently highlighted by agencies such as NIST and DARPA.
- We still need to prove it works by using it in practice.

But with courses like ACT4E and your contribution,

Science of interdisciplinarity

A science of interdisciplinarity can reduce waste.

- Models of different systems need to be independent to optimize.
- They need to be interoperable to work in concert for larger purpose.
- Interdisciplinarity must respect both aspects.

What is the right mathematics to underlie a science of interdisciplinarity?

- Category theory has an impressive record inside of math and out.
- It's been recently highlighted by agencies such as NIST and DARPA.
- We still need to prove it works by using it in practice.

But with courses like ACT4E and your contribution,
we can build a hard science of interdisciplinarity.

Science of interdisciplinarity

A science of interdisciplinarity can reduce waste.

- Models of different systems need to be independent to optimize.
- They need to be interoperable to work in concert for larger purpose.
- Interdisciplinarity must respect both aspects.

What is the right mathematics to underlie a science of interdisciplinarity?

- Category theory has an impressive record inside of math and out.
- It's been recently highlighted by agencies such as NIST and DARPA.
- We still need to prove it works by using it in practice.

But with courses like ACT4E and your contribution,
we can build a hard science of interdisciplinarity.

Thanks! Questions and comments welcome.