Dynamic networks and learning

David I. Spivak

Grant FA9550-20-1-0348. Thanks to Fred Leve, Tristan Nguyen, & Doug Riecken

Outline

Working foundations for complex systems theory

Here's my basic viewpoint:

- The world is becoming increasingly complex.
- Complex systems theory lacks a working foundation.
- Consider: what mathematical abstractions simultaneously model
  - dynamical,
  - decision-making,
  - database,
  - deep learning, ...

  systems, and provide for the links and interactions between them?

Working foundations for complex systems theory

Here's my basic viewpoint:

- The world is becoming increasingly complex.
- Complex systems theory lacks a working foundation.
- Consider: what mathematical abstractions simultaneously model
  - dynamical,
  - decision-making,
  - database,
  - deep learning, ...

  systems, and provide for the links and interactions between them?
- Set theory is a foundation, but it's not helpful; not "working".

Finding working abstractions is what I spend my time on.

Finding the right abstractions

One can do work without good abstractions.

- Long ago, there were different *number systems* for cheese, goats, etc.
- Before the civil war, each bank had its own currency.
- Until 1200s, Roman numerals were used for all math (e.g. $\times$, $\sqrt{2}$, etc).

Finding the right abstractions

One can do work without good abstractions.

- Long ago, there were different *number systems* for cheese, goats, etc.
- Before the civil war, each bank had its own currency.
- Until 1200s, Roman numerals were used for all math (e.g. $\times$, $\sqrt{2}$, etc).

Mathematics provides accounting systems.

- Accounting systems, broadly construed, let us track what's going on.
- An account of my bank balance, an account of how nature works, etc.
- We need math systems in which to record events, do our reckoning.

Finding the right abstractions

One can do work without good abstractions.

- Long ago, there were different *number systems* for cheese, goats, etc.
- Before the civil war, each bank had its own currency.
- Until 1200s, Roman numerals were used for all math (e.g. $\times$, $\sqrt{2}$, etc).

Mathematics provides accounting systems.

- Accounting systems, broadly construed, let us track what's going on.
- An account of my bank balance, an account of how nature works, etc.
- We need math systems in which to record events, do our reckoning.

The mathematical accounting used today for complex systems are poor.

- Dynamics, decisions, data, and learning each use their own system.
- I'm betting that these can be unified by *finding the right abstractions*.
- Category theory is known to provide interoperable (multi-context) accounting systems.

Toward foundations for complex systems

We want a common foundation for dynamics, decisions, data, and learning.

- It turns out there's a really elegant formalism available.

Toward foundations for complex systems

We want a common foundation for dynamics, decisions, data, and learning.

- It turns out there's a really elegant formalism available.
  - It goes under the unenlightening name *polynomial functor theory*.
  - It's been well-studied by eminent mathematicians.
  - It generalizes the use of *lenses* in functional programming.
  - I noticed that it also applies to the four subjects above.
  - Unified foundations facilitates knowledge transfer among them.

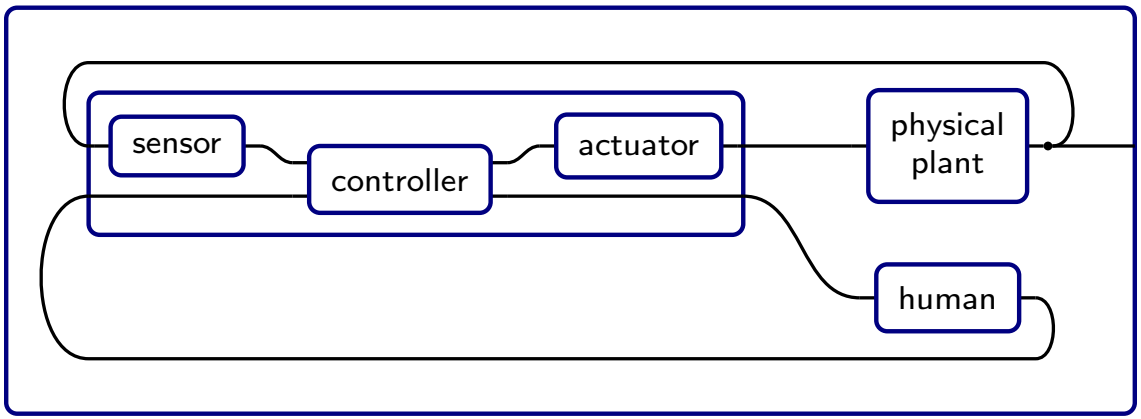
Toward foundations for complex systems

We want a common foundation for dynamics, decisions, data, and learning.

- It turns out there's a really elegant formalism available.
  - It goes under the unenlightening name *polynomial functor theory*.
  - It's been well-studied by eminent mathematicians.
  - It generalizes the use of *lenses* in functional programming.
  - I noticed that it also applies to the four subjects above.
  - Unified foundations facilitates knowledge transfer among them.
- Today I'll just tell one relatively simple story about *dynamic networks*.

Introduction to dynamic networks

Wiring diagrams provide a very useful abstraction for systems of systems:



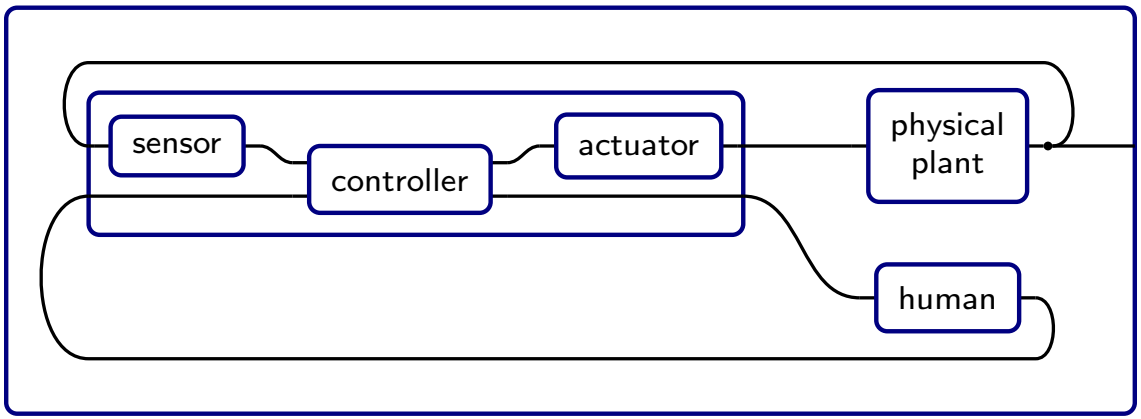
Each block  represents an open dynamical system:

$$x \in \mathbb{R}^n, a \in A, b \in B \qquad \dot{x} = f^{\text{dyn}}(x, a), \qquad b = f^{\text{out}}(x)$$

- The systems can be discrete, continuous, hybrid, whatever you want.
- The wires say how outputs of one are fed as parameters to another.

Introduction to dynamic networks

Wiring diagrams provide a very useful abstraction for systems of systems:



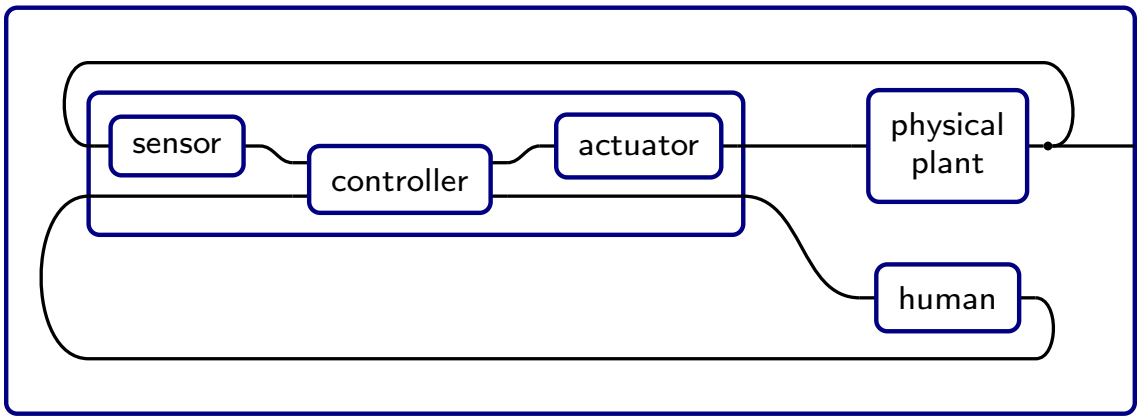
Each block  represents an open dynamical system:

$$x \in \mathbb{R}^n, a \in A, b \in B \qquad \dot{x} = f^{\text{dyn}}(x, a), \qquad b = f^{\text{out}}(x)$$

- The systems can be discrete, continuous, hybrid, whatever you want.
- The wires say how outputs of one are fed as parameters to another.
- A *dynamic network* is one where the connectivity changes in time.

Introduction to dynamic networks

Wiring diagrams provide a very useful abstraction for systems of systems:



Each block  represents an open dynamical system:

$$x \in \mathbb{R}^n, a \in A, b \in B \qquad \dot{x} = f^{\text{dyn}}(x, a), \qquad b = f^{\text{out}}(x)$$

- The systems can be discrete, continuous, hybrid, whatever you want.
- The wires say how outputs of one are fed as parameters to another.
- A *dynamic network* is one where the connectivity changes in time.

Today's talk: I'll explain this and how it generalizes deep learning.

Outline

Bundles

Fix a convenient category $\mathcal{S}$ of sets or spaces.

- A *bundle* is just a map $E \xrightarrow{\pi} B$; call E *total space* and B *base space*.
- Given $b \in B$, the preimage $E[b] := \pi^{-1}(b)$ is called the *fiber over* b.

Bundles

Fix a convenient category $\mathcal{S}$ of sets or spaces.

- A *bundle* is just a map $E \xrightarrow{\pi} B$; call E *total space* and B *base space*.
- Given $b \in B$, the preimage $E[b] := \pi^{-1}(b)$ is called the *fiber over* b.

Examples:

- The *tangent bundle*, $T\mathbb{R}^n \to \mathbb{R}^n$. The fiber over $x \in \mathbb{R}^n$ is $T_x\mathbb{R}^n$.
- The *trivial bundle*, $F \times B \to B$. The fiber over $b \in B$ is always F.
- $\{(x, y) \mid 0 \leq x \leq y\} \to \mathbb{R}$. The fiber changes as $y \in \mathbb{R}$ varies.

Bundles

Fix a convenient category $\mathcal{S}$ of sets or spaces.

- A *bundle* is just a map $E \xrightarrow{\pi} B$; call E *total space* and B *base space*.
- Given $b \in B$, the preimage $E[b] := \pi^{-1}(b)$ is called the *fiber over* b.

Examples:

- The *tangent bundle*, $T\mathbb{R}^n \to \mathbb{R}^n$. The fiber over $x \in \mathbb{R}^n$ is $T_x\mathbb{R}^n$.
- The *trivial bundle*, $F \times B \to B$. The fiber over $b \in B$ is always F.
- $\{(x, y) \mid 0 \le x \le y\} \to \mathbb{R}$. The fiber changes as $y \in \mathbb{R}$ varies.

Given $f : B' \to B$, denote by $f^*E \to B'$ the *pullback bundle*.

$$\begin{array}{ccc} f^*E & \longrightarrow & E \\ {\scriptstyle \pi_f}\downarrow & \lrcorner & \downarrow{\scriptstyle \pi} \\ B' & \xrightarrow[f]{} & B \end{array}$$

- π' has same fibers as π, i.e. $(f^*E)[b'] = E[f(b')]$ for each $b' \in B'$.

The category of bundles and lenses

It's often convenient to denote bundles as generating fun's (polynomials):

$$\begin{array}{c} E \\ \downarrow \\ B \end{array} \qquad \leftrightsquigarrow \qquad \sum_{b \in B} y^{E[b]}$$

- Here y is just a formal variable; e.g., the tangent bundle is $\sum_{m \in M} y^{T_m M}$.

A certain kind of map between bundles, called a *lens*, is defined as follows:

$$(f, f^\sharp): \quad \begin{array}{ccc} E' & \longrightarrow & E \\ \downarrow{\scriptstyle \pi'} & & \downarrow{\scriptstyle \pi} \\ B' & & B \end{array} \qquad \text{means} \qquad \begin{array}{ccccc} E' & \xleftarrow{f^\sharp} & f^*E & \longrightarrow & E \\ {\scriptstyle \pi'}\downarrow & & \downarrow & \lrcorner & \downarrow{\scriptstyle \pi} \\ B' & = & B' & \xrightarrow[f]{} & B \end{array}$$

It looks confusing at first, but we'll see a familiar example next slide.

Outline

Open dynamical systems

Let A, B be spaces. Define an (A, B)-*dynamical system* to be

- a natural number n, the *dimension*;
- a continuous function $f^{\text{dyn}} : A \times \mathbb{R}^n \to T\mathbb{R}^n$, called *dynamics*, with

$$\begin{array}{ccc} A \times \mathbb{R}^n & \xrightarrow{f^{\text{dyn}}} & T\mathbb{R}^n \\ & \searrow & \downarrow \\ & & \mathbb{R}^n \end{array}$$

  i.e. a vector field on $\mathbb{R}^n$, parameterized by A; and
- a continuous function $f^{\text{rdt}} : \mathbb{R}^n \to B$, called *readout*.

An (A, B)-dynamical system is denoted $\dot{x} = f^{\text{dyn}}(a, x)$ and $b = f^{\text{rdt}}(x)$.

Open dynamical systems

Let A, B be spaces. Define an (A, B)-*dynamical system* to be

- a natural number n, the *dimension*;
- a continuous function $f^{\text{dyn}} : A \times \mathbb{R}^n \to T\mathbb{R}^n$, called *dynamics*, with

$$\begin{array}{ccc} A \times \mathbb{R}^n & \xrightarrow{f^{\text{dyn}}} & T\mathbb{R}^n \\ & \searrow & \downarrow \\ & & \mathbb{R}^n \end{array}$$

  i.e. a vector field on $\mathbb{R}^n$, parameterized by A; and
- a continuous function $f^{\text{rdt}} : \mathbb{R}^n \to B$, called *readout*.

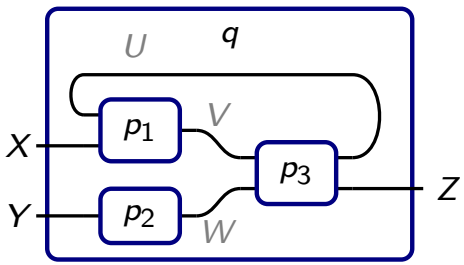
An (A, B)-dynamical system is denoted $\dot{x} = f^{\text{dyn}}(a, x)$ and $b = f^{\text{rdt}}(x)$.

This is exactly the same as a lens $\sum_x y^{T_x} \to By^A$

$$\begin{array}{ccccc} T\mathbb{R}^n & \xleftarrow{f^{\text{dyn}}} & A \times \mathbb{R}^n & \longrightarrow & A \times B \\ \downarrow & & \downarrow & \lrcorner & \downarrow \\ \mathbb{R}^n & = & \mathbb{R}^n & \xrightarrow[f^{\text{rdt}}]{} & B \end{array}$$

Wiring diagrams are also lenses

Not only open dynamical systems but also *wiring diagrams* are lenses.



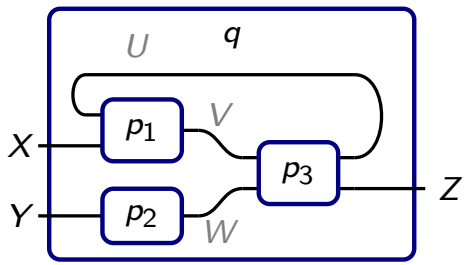
$$p_1 = Vy^{UX}$$
$$p_2 = Wy^{Y}$$
$$p_3 = UZy^{VW}$$
$$q = Zy^{XY}$$

- There are various operations on bundles, e.g. *Dirichlet product* $\otimes$

$$\begin{array}{ccc} E_1 & & E_2 \\ \downarrow & \otimes & \downarrow \\ B_1 & & B_2 \end{array} := \begin{array}{c} E_1 \times E_2 \\ \downarrow \\ B_1 \times B_2 \end{array}$$

Wiring diagrams are also lenses

Not only open dynamical systems but also *wiring diagrams* are lenses.



$$p_1 = Vy^{UX}$$
$$p_2 = Wy^{Y}$$
$$p_3 = UZy^{VW}$$
$$q = Zy^{XY}$$

- There are various operations on bundles, e.g. *Dirichlet product* $\otimes$

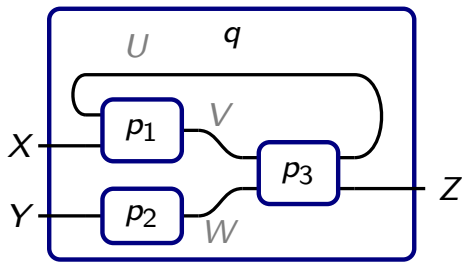
$$\begin{array}{ccc} E_1 & & E_2 \\ \downarrow & \otimes & \downarrow \\ B_1 & & B_2 \end{array} := \begin{array}{c} E_1 \times E_2 \\ \downarrow \\ B_1 \times B_2 \end{array}$$

$$\left(\sum_{b_1 \in B_1} y^{E_1[b_1]}\right) \otimes \left(\sum_{b_2 \in B_2} y^{E_2[b_2]}\right) := \sum_{(b_1,b_2) \in B_1 \times B_2} y^{E_1[b_1] \times E_2[b_2]}$$

e.g. if t is the tangent bundle on $\mathbb{R}$ then $t^{\otimes n}$ is tangent bundle on $\mathbb{R}^n$.

Wiring diagrams are also lenses

Not only open dynamical systems but also *wiring diagrams* are lenses.



$$p_1 = Vy^{UX}$$
$$p_2 = Wy^{Y}$$
$$p_3 = UZy^{VW}$$
$$q = Zy^{XY}$$

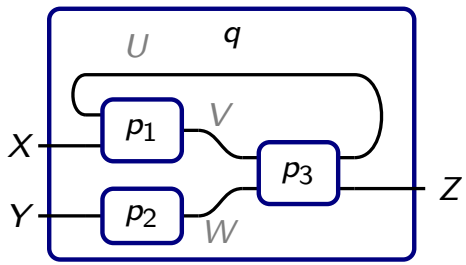
- There are various operations on bundles, e.g. *Dirichlet product* $\otimes$

$$\begin{array}{ccc} E_1 & & E_2 \\ \downarrow & \otimes & \downarrow \\ B_1 & & B_2 \end{array} := \begin{array}{c} E_1 \times E_2 \\ \downarrow \\ B_1 \times B_2 \end{array} \qquad \left(\sum_{b_1 \in B_1} y^{E_1[b_1]}\right) \otimes \left(\sum_{b_2 \in B_2} y^{E_2[b_2]}\right) := \sum_{(b_1,b_2) \in B_1 \times B_2} y^{E_1[b_1] \times E_2[b_2]}$$

  e.g. if t is the tangent bundle on $\mathbb{R}$ then $t^{\otimes n}$ is tangent bundle on $\mathbb{R}^n$.
- The wiring diagram above is a lens $p_1 \otimes p_2 \otimes p_3 \to q$.
- There is a continuum of lenses between wiring diagrams too.

Wiring diagrams are also lenses

Not only open dynamical systems but also *wiring diagrams* are lenses.



$$p_1 = Vy^{UX}$$
$$p_2 = Wy^{Y}$$
$$p_3 = UZy^{VW}$$
$$q = Zy^{XY}$$

- There are various operations on bundles, e.g. *Dirichlet product* $\otimes$

$$\begin{array}{ccc} E_1 & & E_2 \\ \downarrow & \otimes & \downarrow \\ B_1 & & B_2 \end{array} := \begin{array}{c} E_1 \times E_2 \\ \downarrow \\ B_1 \times B_2 \end{array} \qquad \left(\sum_{b_1 \in B_1} y^{E_1[b_1]}\right) \otimes \left(\sum_{b_2 \in B_2} y^{E_2[b_2]}\right) := \sum_{(b_1,b_2) \in B_1 \times B_2} y^{E_1[b_1] \times E_2[b_2]}$$

  e.g. if t is the tangent bundle on $\mathbb{R}$ then $t^{\otimes n}$ is tangent bundle on $\mathbb{R}^n$.
- The wiring diagram above is a lens $p_1 \otimes p_2 \otimes p_3 \to q$.
- There is a continuum of lenses between wiring diagrams too.

We said a dynamical system in each p_i is just a lens $t^{\otimes n_i} \to p_i$.

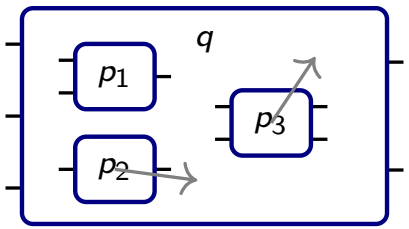
- Thus we get a dyn'l system $t^{\otimes n_1} \otimes t^{\otimes n_2} \otimes t^{\otimes n_3} \to p_1 \otimes p_2 \otimes p_3 \to q$.
- The foundations highlight the compositionality of dynamical systems.

Outline

Dynamic networks

In the real world, interaction patterns often change through time.

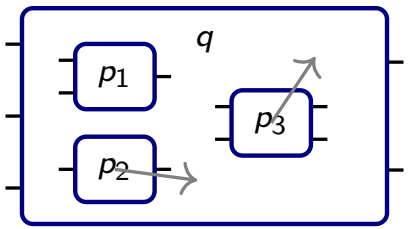
- Heat sources and sinks can be connected and disconnected,
- Airplanes can go in and out of communication range, etc.



Dynamic networks

In the real world, interaction patterns often change through time.

- Heat sources and sinks can be connected and disconnected,
- Airplanes can go in and out of communication range, etc.



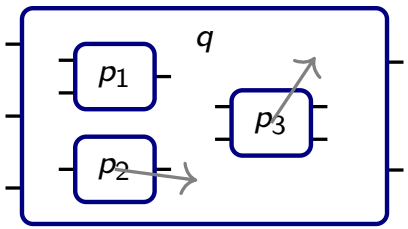
To achieve this, we use the fact that **Poly**, like Vect is *monoidal closed*:

Given $p, q \in \mathbf{Poly}$, get *internal hom* $[p, q] \in \mathbf{Poly}$.

Dynamic networks

In the real world, interaction patterns often change through time.

- Heat sources and sinks can be connected and disconnected,
- Airplanes can go in and out of communication range, etc.



To achieve this, we use the fact that **Poly**, like Vect is *monoidal closed*:

$$\text{Given } p, q \in \mathbf{Poly}, \quad \text{get } \textit{internal hom} \quad [p, q] \in \mathbf{Poly}.$$

Dynamically changing connection pattern for $p_1, \ldots, p_k$ in q is

$$\varphi \colon t^{\otimes n} \to [p_1 \otimes \cdots \otimes p_k, q],$$

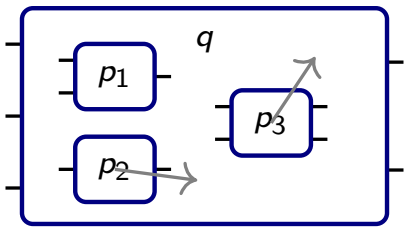
where as before $t^{\otimes n}$ is the tangent bundle $T\mathbb{R}^n \to \mathbb{R}^n$.

- The readout of φ is an interaction pattern $p_1 \otimes \cdots \otimes p_k \to q$
- It dynamically updates $x \in \mathbb{R}^n$ depending on a parameterized ODE.

Dynamic networks

In the real world, interaction patterns often change through time.

- Heat sources and sinks can be connected and disconnected,
- Airplanes can go in and out of communication range, etc.



To achieve this, we use the fact that **Poly**, like Vect is *monoidal closed*:

$$\text{Given } p, q \in \mathbf{Poly}, \quad \text{get } \textit{internal hom} \quad [p, q] \in \mathbf{Poly}.$$

Dynamically changing connection pattern for $p_1, \ldots, p_k$ in q is

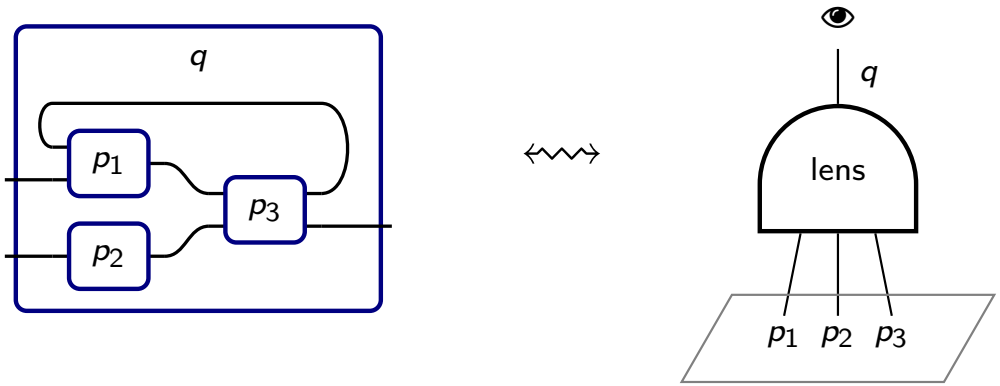
$$\varphi \colon t^{\otimes n} \to [p_1 \otimes \cdots \otimes p_k, q],$$

where as before $t^{\otimes n}$ is the tangent bundle $T\mathbb{R}^n \to \mathbb{R}^n$.

- The readout of φ is an interaction pattern $p_1 \otimes \cdots \otimes p_k \to q$
- It dynamically updates $x \in \mathbb{R}^n$ depending on a parameterized ODE.
- Special case $k = 0$ recovers open dynamical system with interface q.

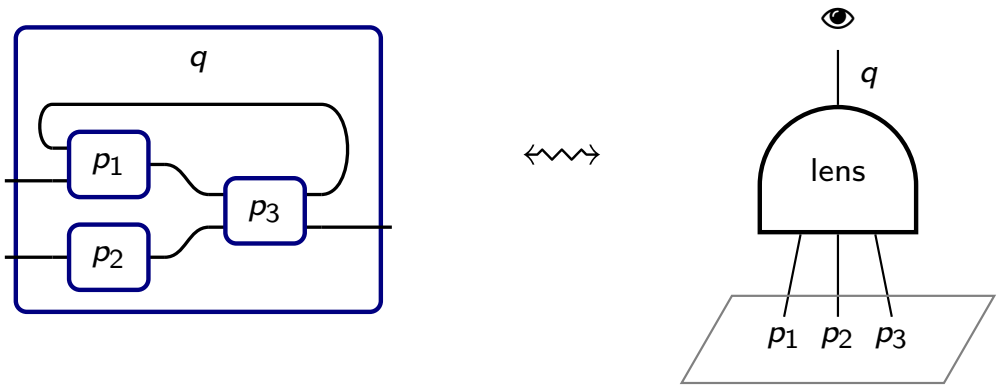
Operadic composition

Imagine the wiring diagram (left) like looking in a microscope (right)



Operadic composition

Imagine the wiring diagram (left) like looking in a microscope (right)

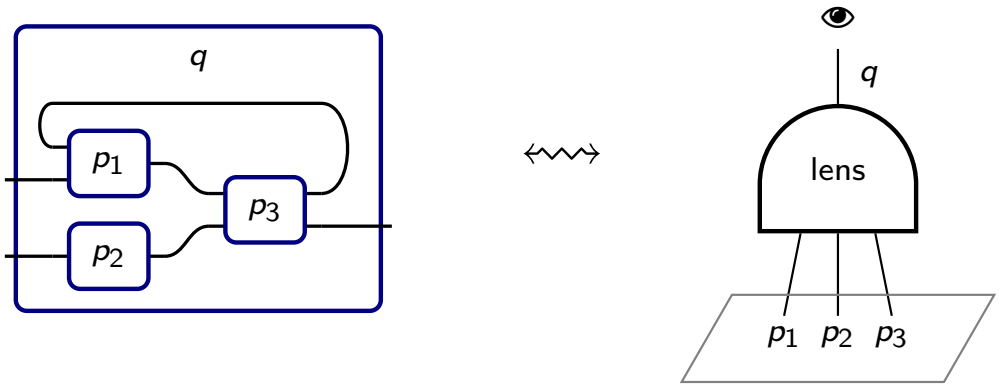


The analogy is useful, but a tiny bit misleading:

- Activity at eyepiece doesn't affect activity on the plate.
- But activity on outer box q does affect activity at inner boxes.

Operadic composition

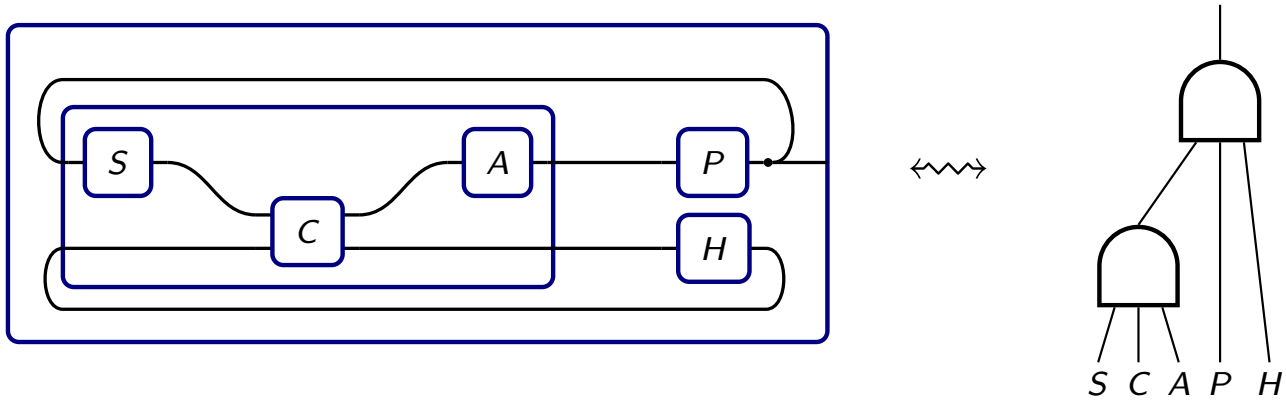
Imagine the wiring diagram (left) like looking in a microscope (right)



The analogy is useful, but a tiny bit misleading:

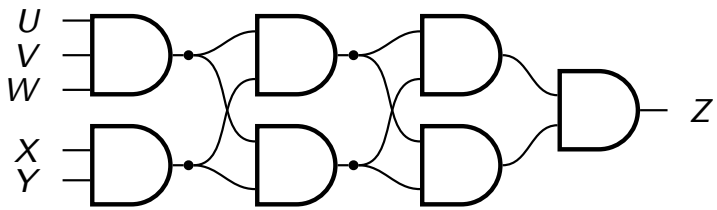
- Activity at eyepiece doesn't affect activity on the plate.
- But activity on outer box q does affect activity at inner boxes.

Operadic composition lets us zoom into the components:



Dynamic networks generalize deep learning

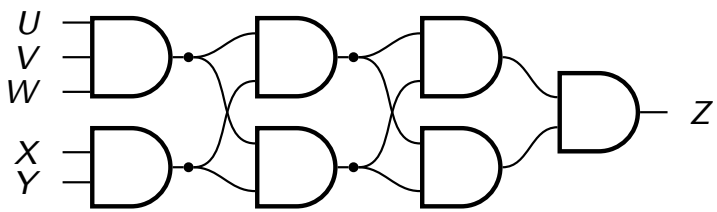
Deep learning involves layers of artificial neurons, e.g.



- The system takes in training data of the form $(u, v, w, x, y; z)$.
- It does a forward pass using current parameters in each neuron.
- It then calculates error and does a backwards pass, ...
- ... updating parameters via gradient descent, and backprop'ing error.

Dynamic networks generalize deep learning

Deep learning involves layers of artificial neurons, e.g.



- The system takes in training data of the form $(u, v, w, x, y; z)$.
- It does a forward pass using current parameters in each neuron.
- It then calculates error and does a backwards pass, ...
- ... updating parameters via gradient descent, and backprop'ing error.

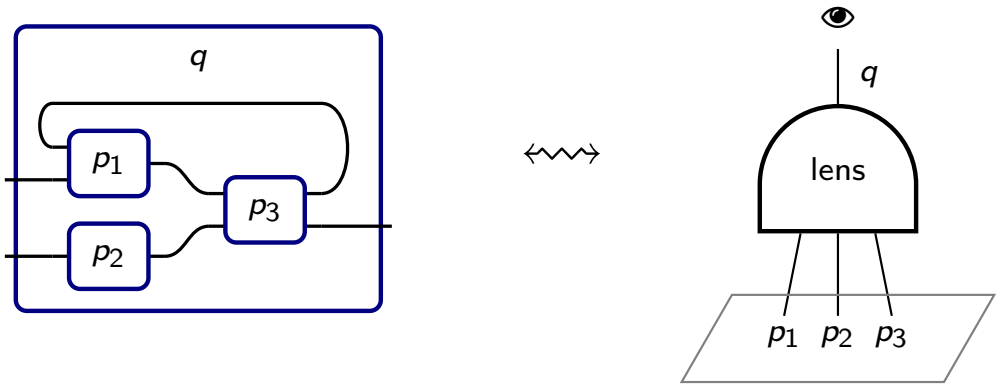
It turns out this gradient descent/backprop algorithm is a lens as above.

- Each neuron  represents a dynamic network: $t^{\otimes(n+1)} \to [t^{\otimes n}, t]$.
- They're composed according to lens composition, as in previous slides.
- The gradient descent / backprop algor'm is very special case of a lens.
- In particular, these lenses have no peer-to-peer message passing.

Outline

Summary

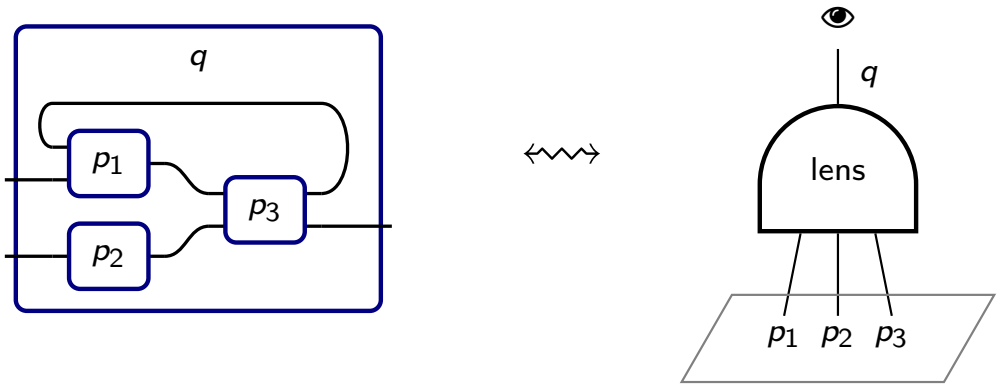
Open dynamical systems can be wired together to form networks.



- Lenses (aka polynomial functors) generalize both ODS's and WD's.
- They give a convenient foundation for considering *dynamic networks*.

Summary

Open dynamical systems can be wired together to form networks.

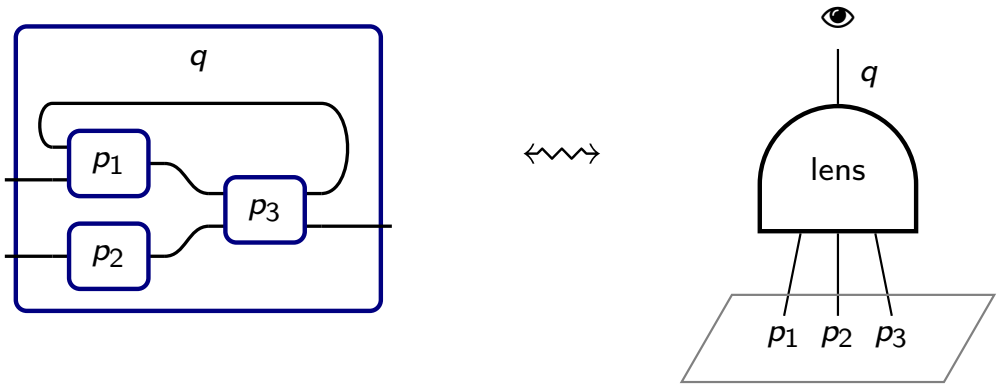


- Lenses (aka polynomial functors) generalize both ODS's and WD's.
- They give a convenient foundation for considering *dynamic networks*.
  - Connection pattern is quite salient in accounting for behavior.
- The poly/lens formalism exposes that dynamic networks generalize:
  - Open dynamical systems, "rewiring diagrams", and DNNs.
  - DNNs are dynamic networks w/o peer-to-peer interaction.

Is there a use for adding peer-to-peer interaction to deep learning?

Summary

Open dynamical systems can be wired together to form networks.



- Lenses (aka polynomial functors) generalize both ODS's and WD's.
- They give a convenient foundation for considering *dynamic networks*.
  - Connection pattern is quite salient in accounting for behavior.
- The poly/lens formalism exposes that dynamic networks generalize:
  - Open dynamical systems, "rewiring diagrams", and DNNs.
  - DNNs are dynamic networks w/o peer-to-peer interaction.

Is there a use for adding peer-to-peer interaction to deep learning?

Thanks; comments and questions welcome!