Lenses: applications and generalizations

David I. Spivak

Department of Mathematics
Massachusetts Institute of Technology

Outline

1 **Introduction**
- An agent in an environment
- Lenses organize interactions
- Lenses in CT

2 Some applications of lenses

3 Generalizing lens categories

4 Conclusion

An agent in an environment

We always hear of an *agent in an environment*. What's that?

An agent in an environment

We always hear of an *agent in an environment*. What's that?

- The agent has an effect on the environment and vice versa.
- What does that mean?

An agent in an environment

We always hear of an *agent in an environment*. What's that?

- The agent has an effect on the environment and vice versa.
- What does that mean?
- It means agent and environment are communicating somehow.
  - The agent *observes* the environment and *acts* on it.

An agent in an environment

We always hear of an *agent in an environment*. What's that?

- The agent has an effect on the environment and vice versa.
- What does that mean?
- It means agent and environment are communicating somehow.
  - The agent *observes* the environment and *acts* on it.
  - The agent's state affects that of the environment and vice versa.
    - Agent affects environment through *action*.
    - Environment affects agent through *observation*.
    - Each is affected in that it undergoes a change of *state*.

How shall we model this mathematically?

A formalization of agent/environment interaction

Setup:

- Agent affects environment through *action*.
- Environment affects agent through *observation*.
- Each is affected in that it undergoes a change of *state*.

A formalization of agent/environment interaction

Setup:

- Agent affects environment through *action*.
- Environment affects agent through *observation*.
- Each is affected in that it undergoes a change of *state*.

Let's model states and communications as sets:

- a set S_{Ag} for the possible states of the agent,
- a set S_{En} for the possible states of the environment,
- a set A for the possible actions, and
- a set O for the possible observations.

A formalization of agent/environment interaction

Setup:

- Agent affects environment through *action*.
- Environment affects agent through *observation*.
- Each is affected in that it undergoes a change of *state*.

Let's model states and communications as sets:

- a set S_{Ag} for the possible states of the agent,
- a set S_{En} for the possible states of the environment,
- a set A for the possible actions, and
- a set O for the possible observations.

These change in time. At every time step, what happens?

- Action is dictated by agent's state via some $S_{\text{Ag}} \to A$.

A formalization of agent/environment interaction

Setup:

- Agent affects environment through *action*.
- Environment affects agent through *observation*.
- Each is affected in that it undergoes a change of *state*.

Let's model states and communications as sets:

- a set S_{Ag} for the possible states of the agent,
- a set S_{En} for the possible states of the environment,
- a set A for the possible actions, and
- a set O for the possible observations.

These change in time. At every time step, what happens?

- Action is dictated by agent's state via some $S_{\text{Ag}} \to A$.
- Agent's state is updated by the observation via some $S_{\text{Ag}} \times O \to S_{\text{Ag}}$.

A formalization of agent/environment interaction

Setup:

- Agent affects environment through *action*.
- Environment affects agent through *observation*.
- Each is affected in that it undergoes a change of *state*.

Let's model states and communications as sets:

- a set S_{Ag} for the possible states of the agent,
- a set S_{En} for the possible states of the environment,
- a set A for the possible actions, and
- a set O for the possible observations.

These change in time. At every time step, what happens?

- Action is dictated by agent's state via some $S_{\text{Ag}} \to A$.
- Agent's state is updated by the observation via some $S_{\text{Ag}} \times O \to S_{\text{Ag}}$.
- Observation is dictated by environment's state via $S_{\text{En}} \to O$.

A formalization of agent/environment interaction

Setup:

- Agent affects environment through *action*.
- Environment affects agent through *observation*.
- Each is affected in that it undergoes a change of *state*.

Let's model states and communications as sets:

- a set S_{Ag} for the possible states of the agent,
- a set S_{En} for the possible states of the environment,
- a set A for the possible actions, and
- a set O for the possible observations.

These change in time. At every time step, what happens?

- Action is dictated by agent's state via some $S_{\text{Ag}} \to A$.
- Agent's state is updated by the observation via some $S_{\text{Ag}} \times O \to S_{\text{Ag}}$.
- Observation is dictated by environment's state via $S_{\text{En}} \to O$.
- Environment's state is updated by the action via $S_{\text{En}} \times A \to S_{\text{En}}$.

How to organize all this stuff?

We have sets $S_{\mathsf{Ag}}, S_{\mathsf{En}}, A, O$ and functions

$$S_{\mathsf{Ag}} \to A \qquad S_{\mathsf{En}} \to O$$
$$S_{\mathsf{Ag}} \times O \to S_{\mathsf{Ag}} \qquad S_{\mathsf{En}} \times A \to S_{\mathsf{En}}$$

How to organize all this stuff?

How to organize all this stuff?

We have sets $S_{\text{Ag}}, S_{\text{En}}, A, O$ and functions

$$S_{\text{Ag}} \to A \qquad S_{\text{En}} \to O$$
$$S_{\text{Ag}} \times O \to S_{\text{Ag}} \qquad S_{\text{En}} \times A \to S_{\text{En}}$$

How to organize all this stuff?

- Each pair of functions is a special case of what are called *lenses*.
- Lenses are the morphisms in a cat **Lens**, whose objects are pairs $\binom{X}{Y}$.
  - The lenses from our agent/environment setup would be denoted:
  - $\binom{S_{\text{Ag}}}{S_{\text{Ag}}} \to \binom{A}{O}$ and $\binom{S_{\text{En}}}{S_{\text{En}}} \to \binom{O}{A}$

How to organize all this stuff?

We have sets $S_{\text{Ag}}, S_{\text{En}}, A, O$ and functions

$$S_{\text{Ag}} \to A \qquad S_{\text{En}} \to O$$
$$S_{\text{Ag}} \times O \to S_{\text{Ag}} \qquad S_{\text{En}} \times A \to S_{\text{En}}$$

How to organize all this stuff?

- Each pair of functions is a special case of what are called *lenses*.
- Lenses are the morphisms in a cat **Lens**, whose objects are pairs $\binom{X}{Y}$.
  - The lenses from our agent/environment setup would be denoted:
  - $\binom{S_{\text{Ag}}}{S_{\text{Ag}}} \to \binom{A}{O}$ and $\binom{S_{\text{En}}}{S_{\text{En}}} \to \binom{O}{A}$

Lenses have been coming up in the ACT community a lot lately.

Applications of lenses

There have been many uses of lens-like things over the years.

- Bidirectional transformations (Oles),
- dialectica categories and linear logic (de Paiva),
- the view-update problem in databases (Hoffman, Pierce),
- functional programming (Gibbons, Oliveira, Palmer, Kmett),
- wiring diagrams, discrete and continuous dynamical systems (Spivak),
- open economic games (Ghani, Hedges),
- supervised learning (Fong, Spivak, Tuyéras).

I'll explain a few of these as we go.

Bringing lenses into the fold

The formula lenses and their composition is kinda weird:

$$\mathbf{Lens}\left(\begin{pmatrix}A\\A'\end{pmatrix},\begin{pmatrix}B\\B'\end{pmatrix}\right) := \left\{ (f, f^\sharp) \;\middle|\; \begin{array}{r} f : A \to B \\ f^\sharp : A \times B' \to A' \end{array} \right\}.$$

Bringing lenses into the fold

The formula lenses and their composition is kinda weird:

$$\mathbf{Lens}\left(\begin{pmatrix}A\\A'\end{pmatrix},\begin{pmatrix}B\\B'\end{pmatrix}\right) := \left\{(f, f^\sharp) \;\middle|\; \begin{array}{r} f : A \to B \\ f^\sharp : A \times B' \to A' \end{array}\right\}.$$

Can we understand **Lens** in another way that's more comfortable?

- Today: we'll first see **Lens** as part of a larger category that
  - provides a geometrical perspective,
  - is more familiar to category theorists, and
  - has better formal properties.

Bringing lenses into the fold

The formula lenses and their composition is kinda weird:

$$\mathbf{Lens}\left(\begin{pmatrix}A\\A'\end{pmatrix},\begin{pmatrix}B\\B'\end{pmatrix}\right) := \left\{(f, f^\sharp) \;\middle|\; \begin{array}{r} f : A \to B \\ f^\sharp : A \times B' \to A' \end{array}\right\}.$$

Can we understand **Lens** in another way that's more comfortable?

- Today: we'll first see **Lens** as part of a larger category that
  - provides a geometrical perspective,
  - is more familiar to category theorists, and
  - has better formal properties.
- We then generalize further to pick up some close cousins of lenses.

Other generalizations

There are other generalizations possible.

Other generalizations

There are other generalizations possible.

- Kmett, Riley, etc. have generalized lenses to *optics*.
  - Briefly: for any monoidal category $(\mathcal{C}, I, \otimes)$, ...
  - an optic $\binom{A}{A'} \to \binom{B}{B'}$ can be identified with an element of

$$\int^{M \in \mathcal{C}} C(A, M \otimes B) \times C(M \otimes B', A').$$

Other generalizations

There are other generalizations possible.

- Kmett, Riley, etc. have generalized lenses to *optics*.
  - Briefly: for any monoidal category $(\mathcal{C}, I, \otimes)$, ...
  - an optic $\binom{A}{A'} \to \binom{B}{B'}$ can be identified with an element of

$$\int^{M \in \mathcal{C}} C(A, M \otimes B) \times C(M \otimes B', A').$$

- This can be generalized even further using Tambara modules.
- However, it's not the direction I want to go today.

Plan of the talk

Plan for the rest of the talk:

- Some applications of lenses
- Generalizing lens categories

Outline

1 Introduction

2 **Some applications of lenses**

3 Generalizing lens categories

4 Conclusion

Agent in an environment

We began with an agent in an environment.

$$S_{\mathsf{Ag}} \to A \qquad S_{\mathsf{En}} \to O$$
$$S_{\mathsf{Ag}} \times O \to S_{\mathsf{Ag}} \qquad S_{\mathsf{En}} \times A \to S_{\mathsf{En}}$$

Agent in an environment

We began with an agent in an environment.

$$S_{\mathsf{Ag}} \to A \qquad S_{\mathsf{En}} \to O$$
$$S_{\mathsf{Ag}} \times O \to S_{\mathsf{Ag}} \qquad S_{\mathsf{En}} \times A \to S_{\mathsf{En}}$$

These are lenses $\binom{S_{\mathsf{Ag}}}{S_{\mathsf{Ag}}} \to \binom{A}{O}$ and $\binom{S_{\mathsf{En}}}{S_{\mathsf{En}}} \to \binom{O}{A}$. What's up with the flip?

Agent in an environment

We began with an agent in an environment.

$$S_{\mathrm{Ag}} \to A \qquad S_{\mathrm{En}} \to O$$
$$S_{\mathrm{Ag}} \times O \to S_{\mathrm{Ag}} \qquad S_{\mathrm{En}} \times A \to S_{\mathrm{En}}$$

These are lenses $\binom{S_{\mathrm{Ag}}}{S_{\mathrm{Ag}}} \to \binom{A}{O}$ and $\binom{S_{\mathrm{En}}}{S_{\mathrm{En}}} \to \binom{O}{A}$. What's up with the flip?

- Idea: if we multiply these lenses we get:

$$\binom{S_{\mathrm{Ag}} \times S_{\mathrm{En}}}{S_{\mathrm{Ag}} \times S_{\mathrm{En}}} \to \binom{A \times O}{O \times A}$$

and there's an "symmetry" lens morphism $\binom{A \times O}{O \times A} \to \binom{1}{1}$.

Agent in an environment

We began with an agent in an environment.

$$S_{\mathrm{Ag}} \to A \qquad S_{\mathrm{En}} \to O$$
$$S_{\mathrm{Ag}} \times O \to S_{\mathrm{Ag}} \qquad S_{\mathrm{En}} \times A \to S_{\mathrm{En}}$$

These are lenses $\binom{S_{\mathrm{Ag}}}{S_{\mathrm{Ag}}} \to \binom{A}{O}$ and $\binom{S_{\mathrm{En}}}{S_{\mathrm{En}}} \to \binom{O}{A}$. What's up with the flip?

- Idea: if we multiply these lenses we get:

$$\binom{S_{\mathrm{Ag}} \times S_{\mathrm{En}}}{S_{\mathrm{Ag}} \times S_{\mathrm{En}}} \to \binom{A \times O}{O \times A}$$

  and there's an "symmetry" lens morphism $\binom{A \times O}{O \times A} \to \binom{1}{1}$.
- Composing, we get a single lens $\binom{S}{S} \to \binom{1}{1}$, where $S = S_{\mathrm{Ag}} \times S_{\mathrm{En}}$.

Agent in an environment

We began with an agent in an environment.

$$S_{\mathrm{Ag}} \to A \qquad S_{\mathrm{En}} \to O$$
$$S_{\mathrm{Ag}} \times O \to S_{\mathrm{Ag}} \qquad S_{\mathrm{En}} \times A \to S_{\mathrm{En}}$$

These are lenses $\binom{S_{\mathrm{Ag}}}{S_{\mathrm{Ag}}} \to \binom{A}{O}$ and $\binom{S_{\mathrm{En}}}{S_{\mathrm{En}}} \to \binom{O}{A}$. What's up with the flip?

- Idea: if we multiply these lenses we get:

$$\binom{S_{\mathrm{Ag}} \times S_{\mathrm{En}}}{S_{\mathrm{Ag}} \times S_{\mathrm{En}}} \to \binom{A \times O}{O \times A}$$

  and there's an "symmetry" lens morphism $\binom{A \times O}{O \times A} \to \binom{1}{1}$.
- Composing, we get a single lens $\binom{S}{S} \to \binom{1}{1}$, where $S = S_{\mathrm{Ag}} \times S_{\mathrm{En}}$.

We can see this as part of a bigger picture.

The agent-environment system

So what were we doing when we:

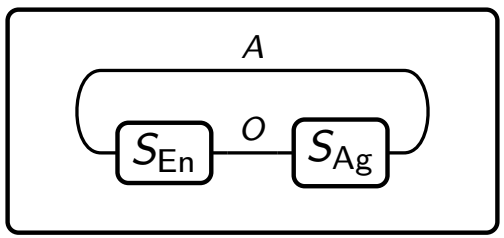
- started with lenses $\binom{S}{S} \to \binom{A}{O}$ and $\binom{S'}{S'} \to \binom{O}{A}$,
- multiplied them together to get a map $\binom{S \times S'}{S \times S'} \to \binom{A \times O}{O \times A}$, and then
- composed the result with a canonical map to $\binom{1}{1}$?

The agent-environment system

So what were we doing when we:

- started with lenses $\binom{S}{S} \to \binom{A}{O}$ and $\binom{S'}{S'} \to \binom{O}{A}$,
- multiplied them together to get a map $\binom{S \times S'}{S \times S'} \to \binom{A \times O}{O \times A}$, and then
- composed the result with a canonical map to $\binom{1}{1}$?

It turns out we were doing this:

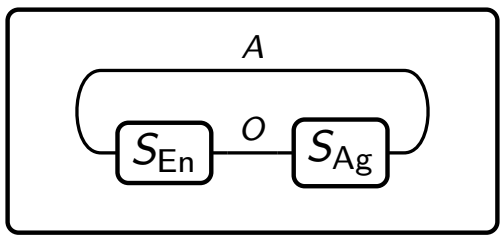


The agent-environment system

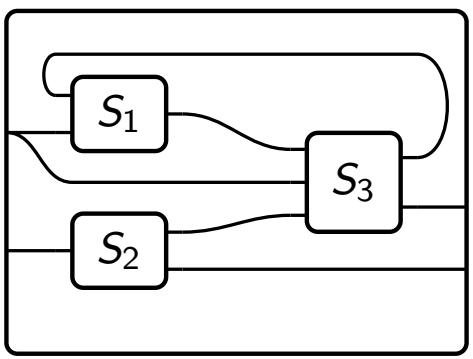
So what were we doing when we:

- started with lenses $\binom{S}{S} \to \binom{A}{O}$ and $\binom{S'}{S'} \to \binom{O}{A}$,
- multiplied them together to get a map $\binom{S \times S'}{S \times S'} \to \binom{A \times O}{O \times A}$, and then
- composed the result with a canonical map to $\binom{1}{1}$?

It turns out we were doing this:

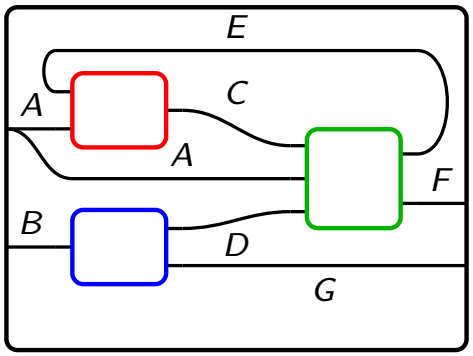


More generally we can consider open systems with many interacting agents



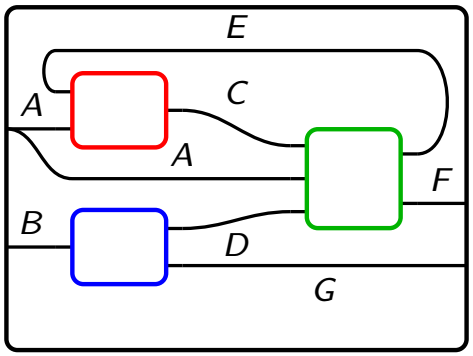
Wiring diagrams

What is going on in this picture mathematically:



Wiring diagrams

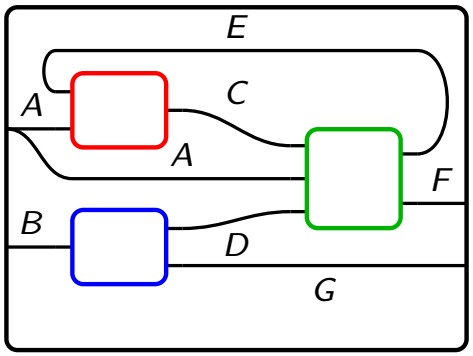
What is going on in this picture mathematically:



For each box, we have a pair $\binom{\text{outputs}}{\text{inputs}}$.

Wiring diagrams

What is going on in this picture mathematically:

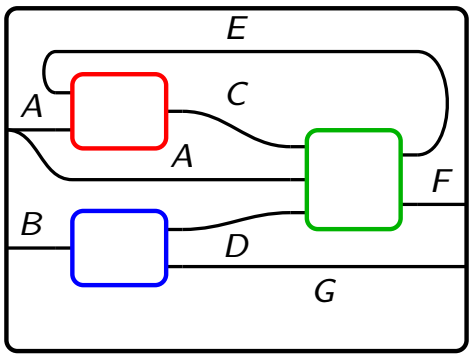


For each box, we have a pair $\binom{\text{outputs}}{\text{inputs}}$.

- We have three interior boxes: $\binom{C}{E \times A}$, $\binom{D \times G}{B}$, $\binom{E \times F}{C \times A \times D}$.
- We have one exterior box: $\binom{F \times G}{A \times B}$.

Wiring diagrams

What is going on in this picture mathematically:

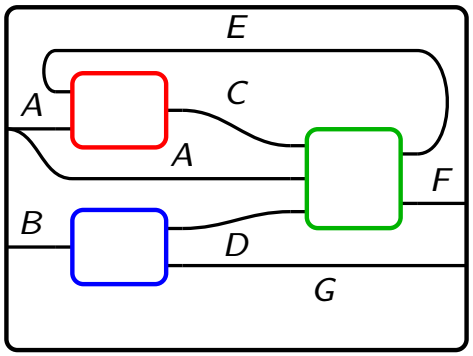


For each box, we have a pair $\binom{\text{outputs}}{\text{inputs}}$.

- We have three interior boxes: $\binom{C}{E \times A}$, $\binom{D \times G}{B}$, $\binom{E \times F}{C \times A \times D}$.
- We have one exterior box: $\binom{F \times G}{A \times B}$.
- The wiring diagram induces a lens $\binom{C \times D \times G \times E \times F}{E \times A \times B \times C \times A \times D} \to \binom{F \times G}{A \times B}$

Wiring diagrams

What is going on in this picture mathematically:



For each box, we have a pair $\binom{\text{outputs}}{\text{inputs}}$.

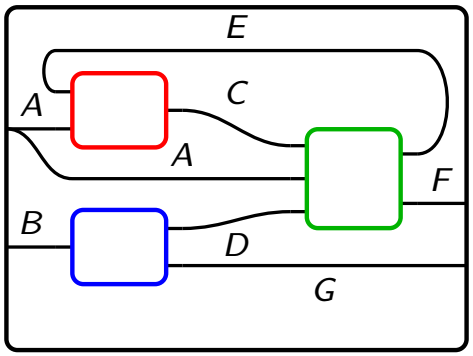
- We have three interior boxes: $\binom{C}{E \times A}$, $\binom{D \times G}{B}$, $\binom{E \times F}{C \times A \times D}$.
- We have one exterior box: $\binom{F \times G}{A \times B}$.
- The wiring diagram induces a lens $\binom{C \times D \times G \times E \times F}{E \times A \times B \times C \times A \times D} \to \binom{F \times G}{A \times B}$
- Both maps are just projections and diagonals:

$$C \times D \times G \times E \times F \to F \times G$$

$$C \times D \times G \times E \times F \times A \times B \to E \times A \times B \times C \times A \times D$$

Wiring diagrams

What is going on in this picture mathematically:



For each box, we have a pair $\binom{\text{outputs}}{\text{inputs}}$.

- We have three interior boxes: $\binom{C}{E \times A}$, $\binom{D \times G}{B}$, $\binom{E \times F}{C \times A \times D}$.
- We have one exterior box: $\binom{F \times G}{A \times B}$.
- The wiring diagram induces a lens $\binom{C \times D \times G \times E \times F}{E \times A \times B \times C \times A \times D} \to \binom{F \times G}{A \times B}$
- Both maps are just projections and diagonals:

$$C \times D \times G \times E \times F \to F \times G$$

$$C \times D \times G \times E \times F \times A \times B \to E \times A \times B \times C \times A \times D$$

Every wiring diagram gives a lens made of projections and diagonals.

WDs and discrete dynamical systems

A discrete dynamical system of type $\binom{A}{A'}$ consists of

- A set S
- A function $f^{\mathrm{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathrm{upd}} \colon S \times A' \to S$ called "update"
- Optional: an element $s_0 \in S$ called "initial state".

WDs and discrete dynamical systems

A discrete dynamical system of type $\binom{A}{A'}$ consists of

- A set S
- A function $f^{\mathrm{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathrm{upd}} \colon S \times A' \to S$ called "update"
- Optional: an element $s_0 \in S$ called "initial state".

This is just a lens $\binom{f^{\mathrm{rdt}}}{f^{\mathrm{upd}}} \colon \binom{S}{S} \to \binom{A}{A'}$, with optional $\binom{s_0}{!} \colon \binom{1}{1} \to \binom{S}{S}$.

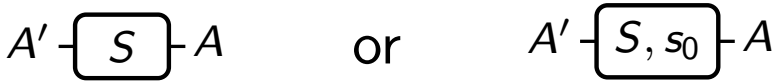
WDs and discrete dynamical systems

A discrete dynamical system of type $\binom{A}{A'}$ consists of

- A set S
- A function $f^{\mathrm{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathrm{upd}} \colon S \times A' \to S$ called "update"
- Optional: an element $s_0 \in S$ called "initial state".

This is just a lens $\binom{f^{\mathrm{rdt}}}{f^{\mathrm{upd}}} \colon \binom{S}{S} \to \binom{A}{A'}$, with optional $\binom{s_0}{!} \colon \binom{1}{1} \to \binom{S}{S}$.

- We'll denote this setup by writing S, or (S, s_0) inside the box



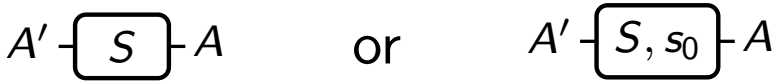
WDs and discrete dynamical systems

A discrete dynamical system of type $\binom{A}{A'}$ consists of

- A set S
- A function $f^{\text{rdt}} \colon S \to A$ called "readout"
- A function $f^{\text{upd}} \colon S \times A' \to S$ called "update"
- Optional: an element $s_0 \in S$ called "initial state".

This is just a lens $\binom{f^{\text{rdt}}}{f^{\text{upd}}} \colon \binom{S}{S} \to \binom{A}{A'}$, with optional $\binom{s_0}{!} \colon \binom{1}{1} \to \binom{S}{S}$.

- We'll denote this setup by writing S, or (S, s_0) inside the box



- A wiring diagram is a lens $\binom{A_1}{A_1'} \otimes \cdots \otimes \binom{A_n}{A_n'} \to \binom{B}{B'}$, and
- Each dynamical system is a lens $\binom{S_i}{S_i} \to \binom{A_i}{A_i'}$. So composing...
- We get a dynamical system $\binom{S_1 \times \cdots \times S_n}{S_1 \times \cdots \times S_n} \to \binom{B}{B'}$ in outer box.

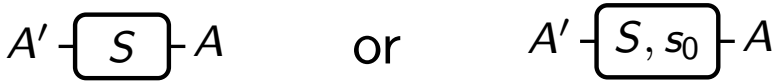
WDs and discrete dynamical systems

A discrete dynamical system of type $\binom{A}{A'}$ consists of

- A set S
- A function $f^{\text{rdt}} \colon S \to A$ called "readout"
- A function $f^{\text{upd}} \colon S \times A' \to S$ called "update"
- Optional: an element $s_0 \in S$ called "initial state".

This is just a lens $\binom{f^{\text{rdt}}}{f^{\text{upd}}} \colon \binom{S}{S} \to \binom{A}{A'}$, with optional $\binom{s_0}{!} \colon \binom{1}{1} \to \binom{S}{S}$.

- We'll denote this setup by writing S, or (S, s_0) inside the box



- A wiring diagram is a lens $\binom{A_1}{A_1'} \otimes \cdots \otimes \binom{A_n}{A_n'} \to \binom{B}{B'}$, and
- Each dynamical system is a lens $\binom{S_i}{S_i} \to \binom{A_i}{A_i'}$. So composing...
- We get a dynamical system $\binom{S_1 \times \cdots \times S_n}{S_1 \times \cdots \times S_n} \to \binom{B}{B'}$ in outer box.

This story of DS's and WD's existed years before I knew about lenses.

Learners

Similarly, the story of learners existed before we knew about lenses.

Learners

Similarly, the story of learners existed before we knew about lenses.

- A learner is something that approximates a function $A' \to A$.

Learners

Similarly, the story of learners existed before we knew about lenses.

- A learner is something that approximates a function $A' \to A$.
  - It consists of a function $P \times A' \to A$, where P is a set
  - It also has an update-backprop function $P \times A' \times A \to P \times A'$.

Learners

Similarly, the story of learners existed before we knew about lenses.

- A learner is something that approximates a function $A' \to A$.
  - It consists of a function $P \times A' \to A$, where P is a set
  - It also has an update-backprop function $P \times A' \times A \to P \times A'$.
  - So it's just a lens $\binom{\text{implement}}{\text{upd-backprop}} : \binom{P}{P} \otimes \binom{A'}{A'} \to \binom{A}{A}$

Learners

Similarly, the story of learners existed before we knew about lenses.

- A learner is something that approximates a function $A' \to A$.
  - It consists of a function $P \times A' \to A$, where P is a set
  - It also has an update-backprop function $P \times A' \times A \to P \times A'$.
  - So it's just a lens $\binom{\text{implement}}{\text{upd-backprop}} : \binom{P}{P} \otimes \binom{A'}{A'} \to \binom{A}{A}$
- For any monoidal category $\mathcal{C}$, there is a monoidal category $\mathbf{Para}(\mathcal{C})$
  - Objects in $\mathbf{Para}(\mathcal{C})$ are objects in $\mathcal{C}$
  - Morphisms $A' \to A$ in $\mathbf{Para}(\mathcal{C})$ consist of pairs (P, f) where
    - P is an object of $\mathcal{C}$,
    - $f : P \otimes A' \to A$
  - Composition is "multiply parameters and compose"

Learners

Similarly, the story of learners existed before we knew about lenses.

- A learner is something that approximates a function $A' \to A$.
  - It consists of a function $P \times A' \to A$, where P is a set
  - It also has an update-backprop function $P \times A' \times A \to P \times A'$.
  - So it's just a lens $\binom{\text{implement}}{\text{upd-backprop}} : \binom{P}{P} \otimes \binom{A'}{A'} \to \binom{A}{A}$
- For any monoidal category $\mathcal{C}$, there is a monoidal category $\mathbf{Para}(\mathcal{C})$
  - Objects in $\mathbf{Para}(\mathcal{C})$ are objects in $\mathcal{C}$
  - Morphisms $A' \to A$ in $\mathbf{Para}(\mathcal{C})$ consist of pairs (P, f) where
    - P is an object of $\mathcal{C}$,
    - $f : P \otimes A' \to A$
  - Composition is "multiply parameters and compose"

Our category **Learn** is just **Para**(**Lens**).

Hierarchical planning

Imagine a high-level planner is negotiating with a low-level planner.

Hierarchical planning

Imagine a high-level planner is negotiating with a low-level planner.

- You say you're in the office and want to leave.
- This is a high-level state (office) and action (leave).
- It's passed to a low-level planner to actually execute the action.

Hierarchical planning

Imagine a high-level planner is negotiating with a low-level planner.

- You say you're in the office and want to leave.
- This is a high-level state (office) and action (leave).
- It's passed to a low-level planner to actually execute the action.
  - The low-level planner needs to know the low-level state
  - Not just "in the office" but "sitting in a chair"

Hierarchical planning

Imagine a high-level planner is negotiating with a low-level planner.

- You say you're in the office and want to leave.
- This is a high-level state (office) and action (leave).
- It's passed to a low-level planner to actually execute the action.
  - The low-level planner needs to know the low-level state
  - Not just "in the office" but "sitting in a chair"
- The low-level state determines the high-level state.

Hierarchical planning

Imagine a high-level planner is negotiating with a low-level planner.

- You say you're in the office and want to leave.
- This is a high-level state (office) and action (leave).
- It's passed to a low-level planner to actually execute the action.
  - The low-level planner needs to know the low-level state
  - Not just "in the office" but "sitting in a chair"
- The low-level state determines the high-level state.

This amounts to a lens $\binom{S_{\text{low}}}{A_{\text{low}}} \to \binom{S_{\text{high}}}{A_{\text{high}}}$.

- There's more to the story than this.
- E.g. "monadic Markov decision processes" are also lenses.

Hierarchical planning

Imagine a high-level planner is negotiating with a low-level planner.

- You say you're in the office and want to leave.
- This is a high-level state (office) and action (leave).
- It's passed to a low-level planner to actually execute the action.
  - The low-level planner needs to know the low-level state
  - Not just "in the office" but "sitting in a chair"
- The low-level state determines the high-level state.

This amounts to a lens $\binom{S_{\text{low}}}{A_{\text{low}}} \to \binom{S_{\text{high}}}{A_{\text{high}}}$.

- There's more to the story than this.
- E.g. "monadic Markov decision processes" are also lenses.

View update?

The view-update problem is a widely-cited example of lenses.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

The usual view-update formulation is kinda weird from my perspective.

- It treats the instances on $\mathcal{C}$ as a set.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

The usual view-update formulation is kinda weird from my perspective.

- It treats the instances on $\mathcal{C}$ as a set.
- Consider a lens $\binom{\text{view}}{\text{update}} : \binom{\mathcal{C}\text{-Inst}}{\mathcal{C}\text{-Inst}} \to \binom{\mathcal{D}\text{-Inst}}{\mathcal{D}\text{-Inst}}$.
- Isn't this quite floppy? Totally not functorial, anything goes.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

The usual view-update formulation is kinda weird from my perspective.

- It treats the instances on $\mathcal{C}$ as a set.
- Consider a lens $\binom{\text{view}}{\text{update}} : \binom{\mathcal{C}\text{-Inst}}{\mathcal{C}\text{-Inst}} \to \binom{\mathcal{D}\text{-Inst}}{\mathcal{D}\text{-Inst}}$.
- Isn't this quite floppy? Totally not functorial, anything goes.

People use lens laws to try to mitigate the floppiness.

- Lens laws: get-put, put-get, and put-put.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

The usual view-update formulation is kinda weird from my perspective.

- It treats the instances on $\mathcal{C}$ as a set.
- Consider a lens $\binom{\text{view}}{\text{update}} : \binom{\mathcal{C}\text{-Inst}}{\mathcal{C}\text{-Inst}} \to \binom{\mathcal{D}\text{-Inst}}{\mathcal{D}\text{-Inst}}$.
- Isn't this quite floppy? Totally not functorial, anything goes.

People use lens laws to try to mitigate the floppiness.

- Lens laws: get-put, put-get, and put-put.
- These together are equivalent to "constant complement" condition.
- That implies that $\mathcal{C}\text{-Inst} \cong \mathcal{D}\text{-Inst} \times M$ for some M
- Basically this only happens if $\mathcal{D}$ is a totally disjoint component of $\mathcal{C}$!

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

The usual view-update formulation is kinda weird from my perspective.

- It treats the instances on $\mathcal{C}$ as a set.
- Consider a lens $\binom{\text{view}}{\text{update}} : \binom{\mathcal{C}\text{-Inst}}{\mathcal{C}\text{-Inst}} \to \binom{\mathcal{D}\text{-Inst}}{\mathcal{D}\text{-Inst}}$.
- Isn't this quite floppy? Totally not functorial, anything goes.

People use lens laws to try to mitigate the floppiness.

- Lens laws: get-put, put-get, and put-put.
- These together are equivalent to "constant complement" condition.
- That implies that $\mathcal{C}$-Inst $\cong \mathcal{D}$-Inst $\times M$ for some M
- Basically this only happens if $\mathcal{D}$ is a totally disjoint component of $\mathcal{C}$!
- The lens laws are too strong, but without them it's too floppy.

View update?

The view-update problem is a widely-cited example of lenses.

- A database *instance* is a bunch of tables filled with data.
- The tables interlock according to a certain pattern, called a *schema*.
- Instances for a given schema $\mathcal{C}$ form a category $\mathcal{C}$-Inst.

The usual view-update formulation is kinda weird from my perspective.

- It treats the instances on $\mathcal{C}$ as a set.
- Consider a lens $\binom{\text{view}}{\text{update}} : \binom{\mathcal{C}\text{-Inst}}{\mathcal{C}\text{-Inst}} \to \binom{\mathcal{D}\text{-Inst}}{\mathcal{D}\text{-Inst}}$.
- Isn't this quite floppy? Totally not functorial, anything goes.

People use lens laws to try to mitigate the floppiness.

- Lens laws: get-put, put-get, and put-put.
- These together are equivalent to "constant complement" condition.
- That implies that $\mathcal{C}$-Inst $\cong \mathcal{D}$-Inst $\times M$ for some M
- Basically this only happens if $\mathcal{D}$ is a totally disjoint component of $\mathcal{C}$!
- The lens laws are too strong, but without them it's too floppy.

Can we do better?

Continuous dynamical systems?

Recall that a discrete dynamical system with inputs A' and outputs A is:

- A set S
- A function $f^{\mathrm{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathrm{upd}} \colon S \times A' \to S$ called "update"

Continuous dynamical systems?

Recall that a discrete dynamical system with inputs A' and outputs A is:

- A set S
- A function $f^{\mathsf{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathsf{upd}} \colon S \times A' \to S$ called "update"

Replacing sets A', A by manifolds, a *continuous dynamical system* is:

- A manifold S, (denote its tangent bundle TS),

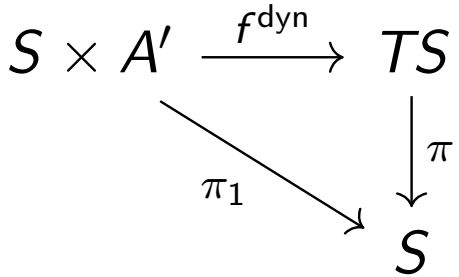
Continuous dynamical systems?

Recall that a discrete dynamical system with inputs A' and outputs A is:

- A set S
- A function $f^{\mathrm{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathrm{upd}} \colon S \times A' \to S$ called "update"

Replacing sets A', A by manifolds, a *continuous dynamical system* is:

- A manifold S, (denote its tangent bundle TS),
- A differentiable map $f^{\mathrm{rdt}} \colon S \to A$,
- A differentiable map $f^{\mathrm{dyn}} \colon S \times A' \to TS$ satisfying:



In other words, for every input and state s, a tangent vector at s.

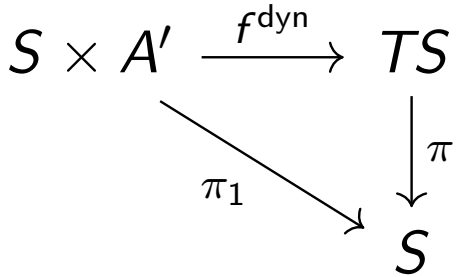
Continuous dynamical systems?

Recall that a discrete dynamical system with inputs A' and outputs A is:

- A set S
- A function $f^{\mathrm{rdt}} \colon S \to A$ called "readout"
- A function $f^{\mathrm{upd}} \colon S \times A' \to S$ called "update"

Replacing sets A', A by manifolds, a *continuous dynamical system* is:

- A manifold S, (denote its tangent bundle TS),
- A differentiable map $f^{\mathrm{rdt}} \colon S \to A$,
- A differentiable map $f^{\mathrm{dyn}} \colon S \times A' \to TS$ satisfying:



In other words, for every input and state s, a tangent vector at s.

The two notions are quite similar, but can we see the latter as a lens?

Outline

1 Introduction

2 Some applications of lenses

3 **Generalizing lens categories**

4 Conclusion

So how should I think about an object in Lens?

How should we think about $\binom{A}{A'}$?

- Is it just a pair of sets?
- Why are maps $\binom{A}{A'} \to \binom{B}{B'}$ the way they are?

So how should I think about an object in Lens?

How should we think about $\binom{A}{A'}$?

- Is it just a pair of sets?
- Why are maps $\binom{A}{A'} \to \binom{B}{B'}$ the way they are?

$$\mathbf{Lens}\left(\binom{A}{A'}, \binom{B}{B'}\right) := \left\{ (f, f^\sharp) \;\middle|\; \begin{array}{r} f : A \to B \\ f^\sharp : A \times B' \to A' \end{array} \right\}.$$

So how should I think about an object in Lens?

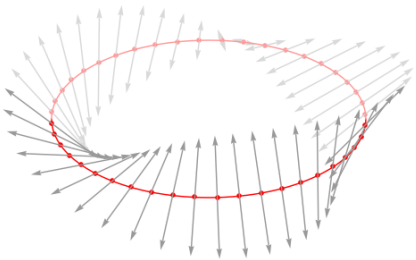
How should we think about $\binom{A}{A'}$?

- Is it just a pair of sets?
- Why are maps $\binom{A}{A'} \to \binom{B}{B'}$ the way they are?

$$\mathbf{Lens}\left(\binom{A}{A'}, \binom{B}{B'}\right) := \left\{ (f, f^\sharp) \;\middle|\; \begin{array}{r} f : A \to B \\ f^\sharp : A \times B' \to A' \end{array} \right\}.$$

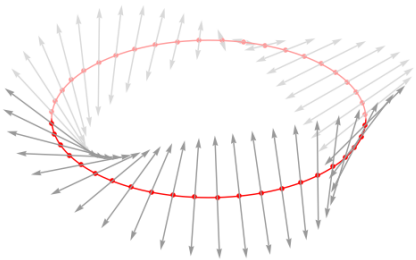
Suggestion: think of objects as "bundles."

What are bundles?



The term bundle is most used in algebraic topology and geometry.

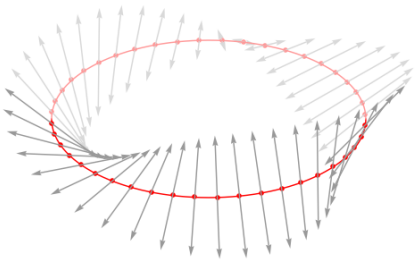
What are bundles?



The term bundle is most used in algebraic topology and geometry.

- A *bundle* is a special kind of morphism $p\colon E \to B$ in a category.
  - The *base space* B consists of "locations" or contexts.
  - For any context $b : B$, the *fiber* $E(b) := p^{-1}(b)$ are possibilities.

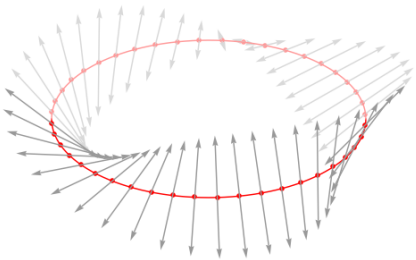
What are bundles?



The term bundle is most used in algebraic topology and geometry.

- A *bundle* is a special kind of morphism $p\colon E \to B$ in a category.
  - The *base space* B consists of "locations" or contexts.
  - For any context $b : B$, the *fiber* $E(b) := p^{-1}(b)$ are possibilities.
- For example, vector bundles, circle bundles, submersions, etc.
  - The tangent bundle TB of a manifold B:
  - At each $b : B$, the fiber $TB(b)$ are the possible velocities at b.

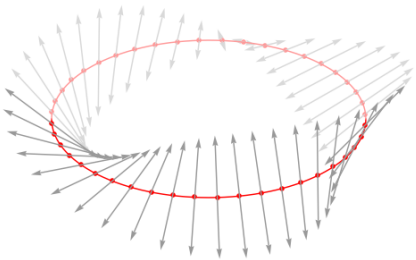
What are bundles?



The term bundle is most used in algebraic topology and geometry.

- A *bundle* is a special kind of morphism $p\colon E \to B$ in a category.
  - The *base space* B consists of "locations" or contexts.
  - For any context $b : B$, the *fiber* $E(b) := p^{-1}(b)$ are possibilities.
- For example, vector bundles, circle bundles, submersions, etc.
  - The tangent bundle TB of a manifold B:
  - At each $b : B$, the fiber $TB(b)$ are the possible velocities at b.
- A database instance can be thought of as a bundle over its schema.
  - A discrete opfibration of categories $p\colon I \to B$.
  - At each table $b : B$, the fiber $I(b)$ is its set of rows.

What are bundles?



The term bundle is most used in algebraic topology and geometry.

- A *bundle* is a special kind of morphism $p\colon E \to B$ in a category.
  - The *base space* B consists of "locations" or contexts.
  - For any context $b : B$, the *fiber* $E(b) := p^{-1}(b)$ are possibilities.
- For example, vector bundles, circle bundles, submersions, etc.
  - The tangent bundle TB of a manifold B:
  - At each $b : B$, the fiber $TB(b)$ are the possible velocities at b.
- A database instance can be thought of as a bundle over its schema.
  - A discrete opfibration of categories $p\colon I \to B$.
  - At each table $b : B$, the fiber $I(b)$ is its set of rows.

A *trivial bundle* is one of the form $\pi_1\colon B \times B' \to B$ for some B'.

Pullbacks of bundles

Suppose that $p\colon E \to B$ is a bundle.

- We haven't said what that means exactly, just given examples.
- But whatever bundles are, you should be able to pull them back.

Pullbacks of bundles

Suppose that $p\colon E \to B$ is a bundle.

- We haven't said what that means exactly, just given examples.
- But whatever bundles are, you should be able to pull them back.
  - That is, given a bundle $E_2 \xrightarrow{p_2} B_2$ and a map $B_1 \to B_2$
  - The pullback should exist and be a bundle over B_1.

$$\begin{array}{ccc} f^*(E_2) & \longrightarrow & E_2 \\ {\scriptstyle p_1}\downarrow & \lrcorner & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array}$$

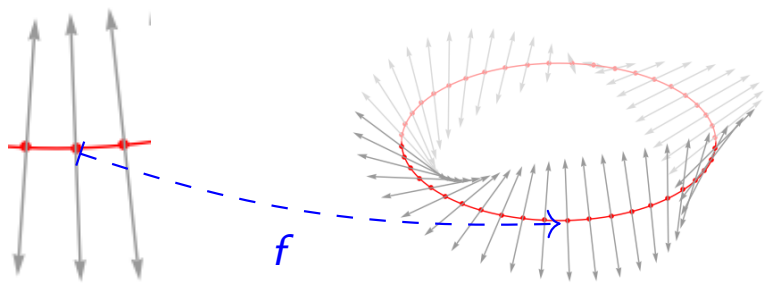
Pullbacks of bundles

Suppose that $p\colon E \to B$ is a bundle.

- We haven't said what that means exactly, just given examples.
- But whatever bundles are, you should be able to pull them back.
  - That is, given a bundle $E_2 \xrightarrow{p_2} B_2$ and a map $B_1 \to B_2$
  - The pullback should exist and be a bundle over B_1.

$$\begin{array}{ccc} f^*(E_2) & \longrightarrow & E_2 \\ {\scriptstyle p_1}\downarrow & \lrcorner & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array}$$

The fiber over any $b_1 \in B_1$ is the same as that over its image $f(b_1)$.



Morphisms of bundles

The usual sort of bundle morphism is just a commutative square

$$\mathrm{Hom}\left(\begin{array}{cc} E_1 & E_2 \\ {\scriptstyle p_1}\downarrow & \downarrow{\scriptstyle p_2} \\ B_1 & B_2 \end{array}\right) = \left\{ (f,g) \;\middle|\; \begin{array}{ccc} E_1 & \xrightarrow{g} & E_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array} \right\}$$

Morphisms of bundles

The usual sort of bundle morphism is just a commutative square

$$\mathrm{Hom}\left(\begin{array}{cc} E_1 & E_2 \\ {\scriptstyle p_1}\downarrow & \downarrow{\scriptstyle p_2} \\ B_1 & B_2 \end{array}\right) = \left\{ (f,g) \;\middle|\; \begin{array}{ccc} E_1 & \xrightarrow{g} & E_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array} \right\}$$

- The pullback $f^*E_2 \cong B_1 \times_{B_2} E_2$ has a universal property by which...
- ... the map g can be identified with a map $E_1 \to f^*E_2$.

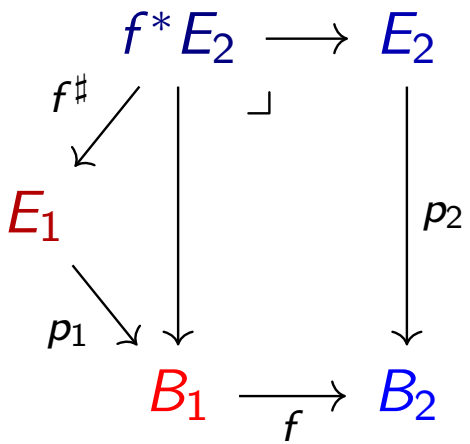
Morphisms of bundles

The usual sort of bundle morphism is just a commutative square

$$\mathrm{Hom}\left(\begin{array}{cc} E_1 & E_2 \\ {\scriptstyle p_1}\downarrow & \downarrow{\scriptstyle p_2} \\ B_1 & B_2 \end{array}\right) = \left\{ (f,g) \;\middle|\; \begin{array}{ccc} E_1 & \xrightarrow{g} & E_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array} \right\}$$

- The pullback $f^*E_2 \cong B_1 \times_{B_2} E_2$ has a universal property by which...
- ... the map g can be identified with a map $E_1 \to f^*E_2$.

But in algebraic geometry, the arrow $E_1 \to f^*(E_2)$ is often reversed:



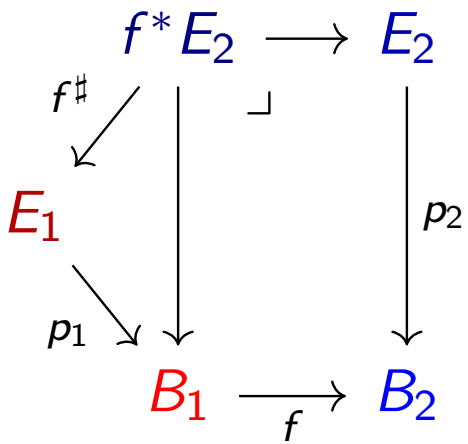
Morphisms of bundles

The usual sort of bundle morphism is just a commutative square

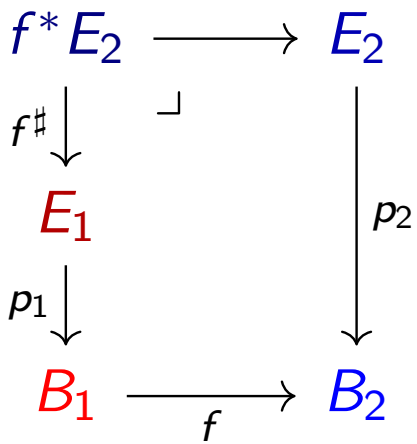
$$\mathrm{Hom}\left(\begin{array}{cc} E_1 & E_2 \\ {\scriptstyle p_1}\downarrow & \downarrow{\scriptstyle p_2} \\ B_1 & B_2 \end{array}\right) = \left\{ (f,g) \;\middle|\; \begin{array}{ccc} E_1 & \xrightarrow{g} & E_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array} \right\}$$

- The pullback $f^*E_2 \cong B_1 \times_{B_2} E_2$ has a universal property by which...
- ... the map g can be identified with a map $E_1 \to f^*E_2$.

But in algebraic geometry, the arrow $E_1 \to f^*(E_2)$ is often reversed:



or simply



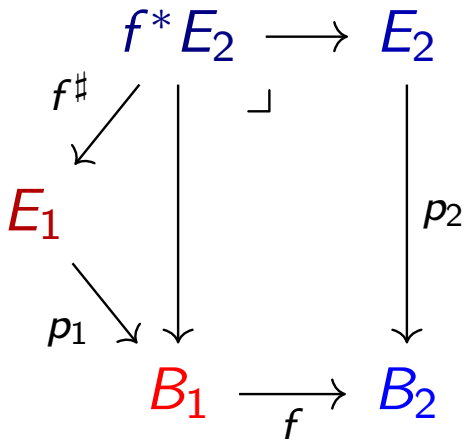
Morphisms of bundles

The usual sort of bundle morphism is just a commutative square

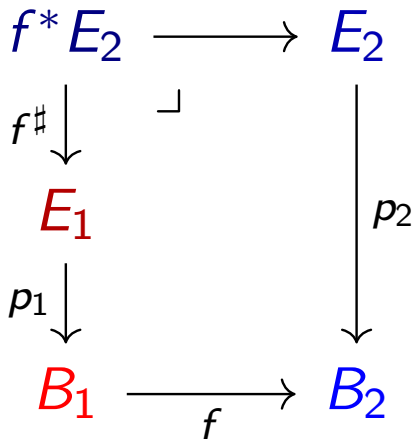
$$\mathrm{Hom}\left(\begin{array}{cc} E_1 & E_2 \\ {\scriptstyle p_1}\downarrow & \downarrow{\scriptstyle p_2} \\ B_1 & B_2 \end{array}\right) = \left\{ (f,g) \;\middle|\; \begin{array}{ccc} E_1 & \xrightarrow{g} & E_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array} \right\}$$

- The pullback $f^*E_2 \cong B_1 \times_{B_2} E_2$ has a universal property by which...
- ... the map g can be identified with a map $E_1 \to f^*E_2$.

But in algebraic geometry, the arrow $E_1 \to f^*(E_2)$ is often reversed:



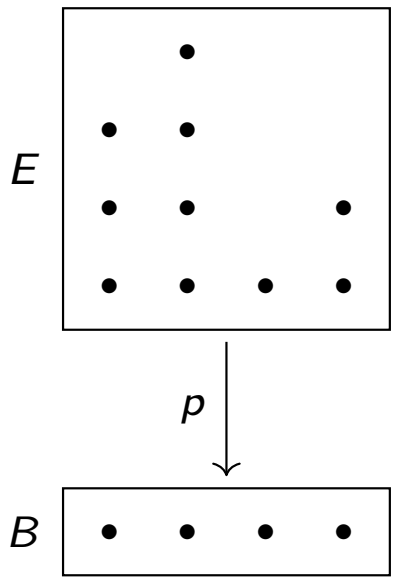
or simply



There's a strong relationship between the AG-style maps and lenses.

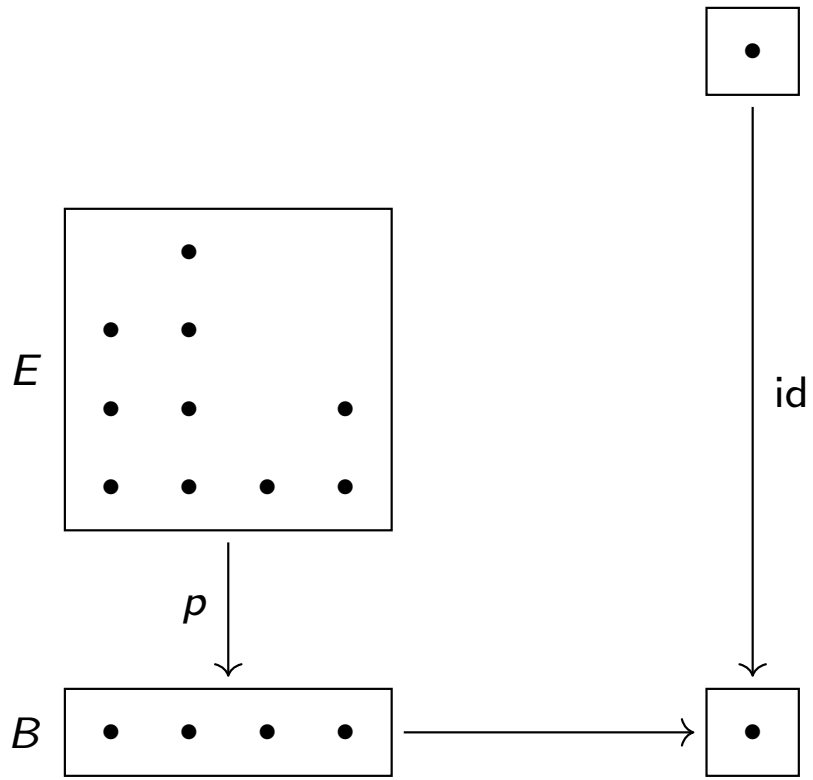
Example

Given a bundle $p\colon E \to B$, let's visualize a map to the bundle $1 \to 1$.



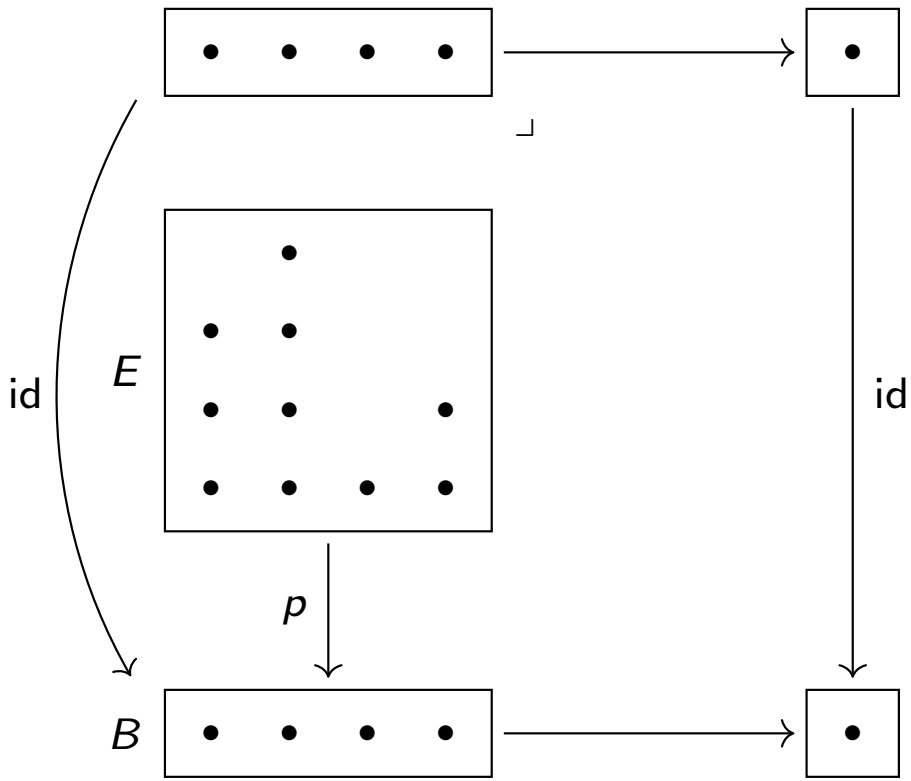
Example

Given a bundle $p\colon E \to B$, let's visualize a map to the bundle $1 \to 1$.



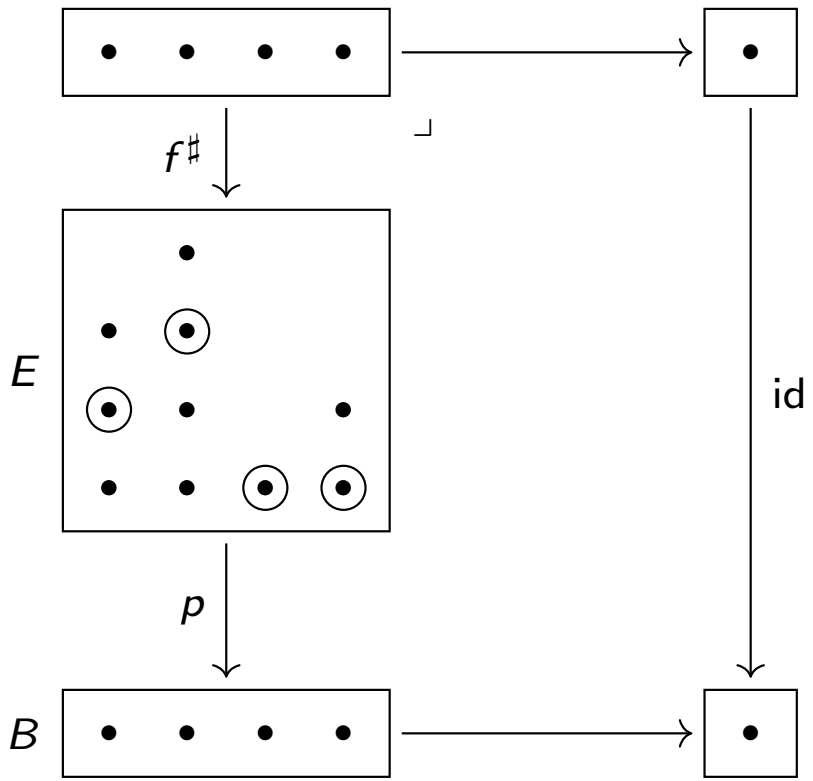
Example

Given a bundle $p\colon E \to B$, let's visualize a map to the bundle $1 \to 1$.



Example

Given a bundle $p\colon E \to B$, let's visualize a map to the bundle $1 \to 1$.



Interpretation of bimorphic lenses as trivial bundles

We will see that **Lens** sits inside this category of bundles.

- That is, there is a fully faithful functor **Lens** $\to$ **Bund**.
- Send lens object $\binom{B}{B'}$ to the trivial bundle (projection) $B \times B' \to B$.

Interpretation of bimorphic lenses as trivial bundles

We will see that **Lens** sits inside this category of bundles.

- That is, there is a fully faithful functor **Lens** $\to$ **Bund**.
- Send lens object $\binom{B}{B'}$ to the trivial bundle (projection) $B \times B' \to B$.
- Note that the pullback of a projection is a projection:

$$\begin{array}{ccc} B_1 \times B' & \longrightarrow & B_2 \times B' \\ \downarrow & \lrcorner & \downarrow \\ B_1 & \xrightarrow[f]{} & B_2 \end{array}$$

Interpretation of bimorphic lenses as trivial bundles

We will see that **Lens** sits inside this category of bundles.

- That is, there is a fully faithful functor **Lens** $\to$ **Bund**.
- Send lens object $\binom{B}{B'}$ to the trivial bundle (projection) $B \times B' \to B$.
- Note that the pullback of a projection is a projection because:

$$\begin{array}{ccccc} B_1 \times B' & \longrightarrow & B_2 \times B' & \longrightarrow & B' \\ \downarrow & \lrcorner & \downarrow & \lrcorner & \downarrow \\ B_1 & \xrightarrow[f]{} & B_2 & \longrightarrow & 1 \end{array}$$

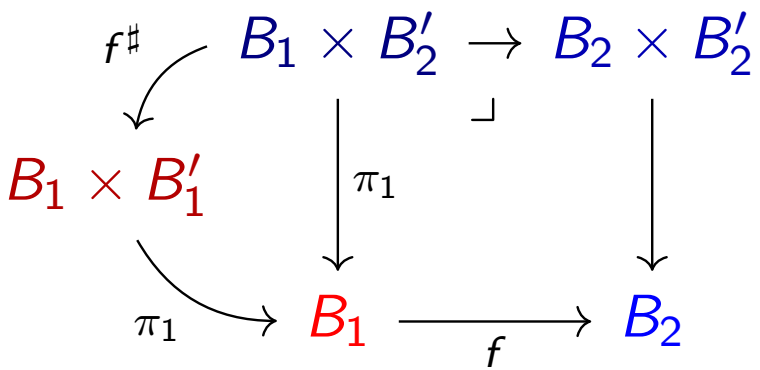
Interpretation of bimorphic lenses as trivial bundles

We will see that **Lens** sits inside this category of bundles.

- That is, there is a fully faithful functor **Lens** $\to$ **Bund**.
- Send lens object $\binom{B}{B'}$ to the trivial bundle (projection) $B \times B' \to B$.
- Note that the pullback of a projection is a projection:

$$\begin{array}{ccc} B_1 \times B_2' & \longrightarrow & B_2 \times B_2' \\ \downarrow & \lrcorner & \downarrow \\ B_1 & \xrightarrow[f]{} & B_2 \end{array}$$

- Send morphism $\binom{f}{f^\sharp} : \binom{B_1}{B_1'} \to \binom{B_2}{B_2'}$ to the bundle morphism:



Such a map $f^\sharp : B_1 \times B_2' \to B_1 \times B_1'$, — in order to commute with π_1 — has no choice on the B_1 factor. Thus it can be identified with a map $f^\sharp : B_1 \times B_2' \to B_1'$.

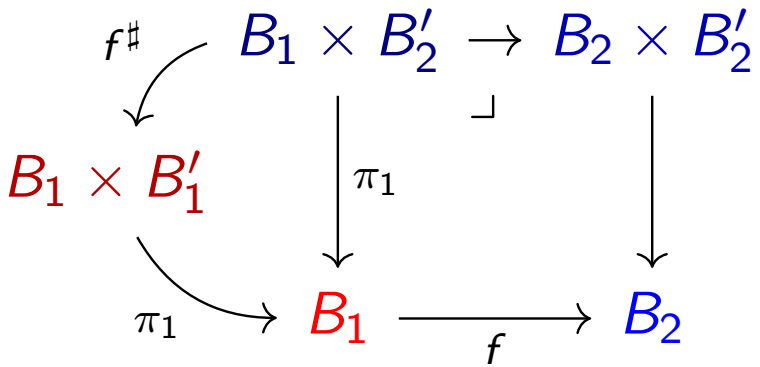
Interpretation of bimorphic lenses as trivial bundles

We will see that **Lens** sits inside this category of bundles.

- That is, there is a fully faithful functor **Lens** $\to$ **Bund**.
- Send lens object $\binom{B}{B'}$ to the trivial bundle (projection) $B \times B' \to B$.
- Note that the pullback of a projection is a projection:

$$\begin{array}{ccc} B_1 \times B_2' & \longrightarrow & B_2 \times B_2' \\ \downarrow & \lrcorner & \downarrow \\ B_1 & \xrightarrow[f]{} & B_2 \end{array}$$

- Send morphism $\binom{f}{f^\sharp} : \binom{B_1}{B_1'} \to \binom{B_2}{B_2'}$ to the bundle morphism:

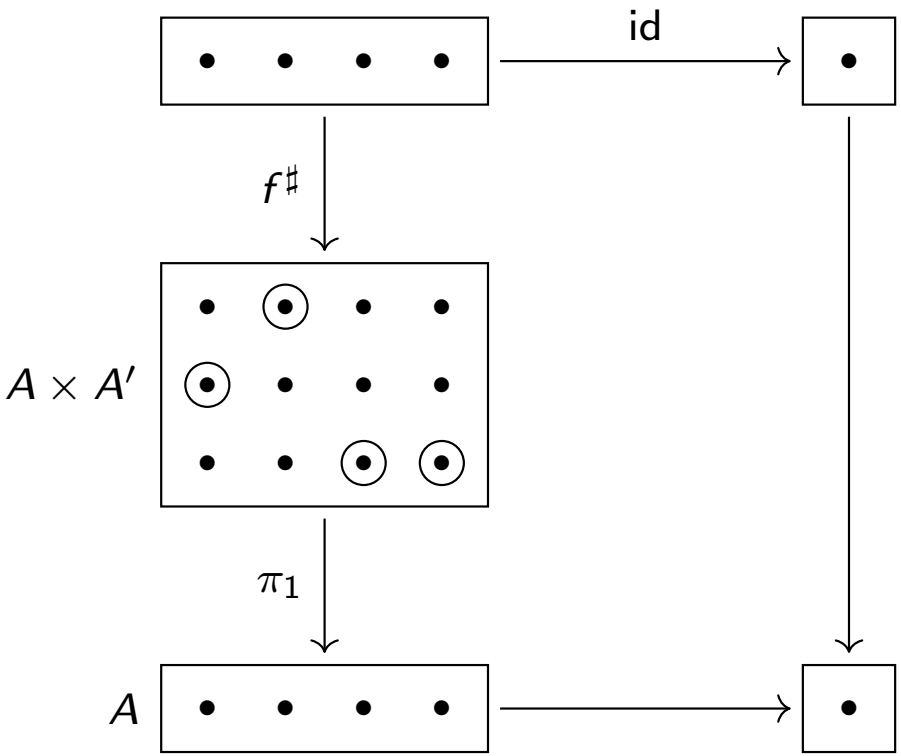


Such a map $f^\sharp : B_1 \times B_2' \to B_1 \times B_1'$, — in order to commute with π_1 — has no choice on the B_1 factor. Thus it can be identified with a map $f^\sharp : B_1 \times B_2' \to B_1'$.

Let's change the name **Bund** to **Lens**$'$.

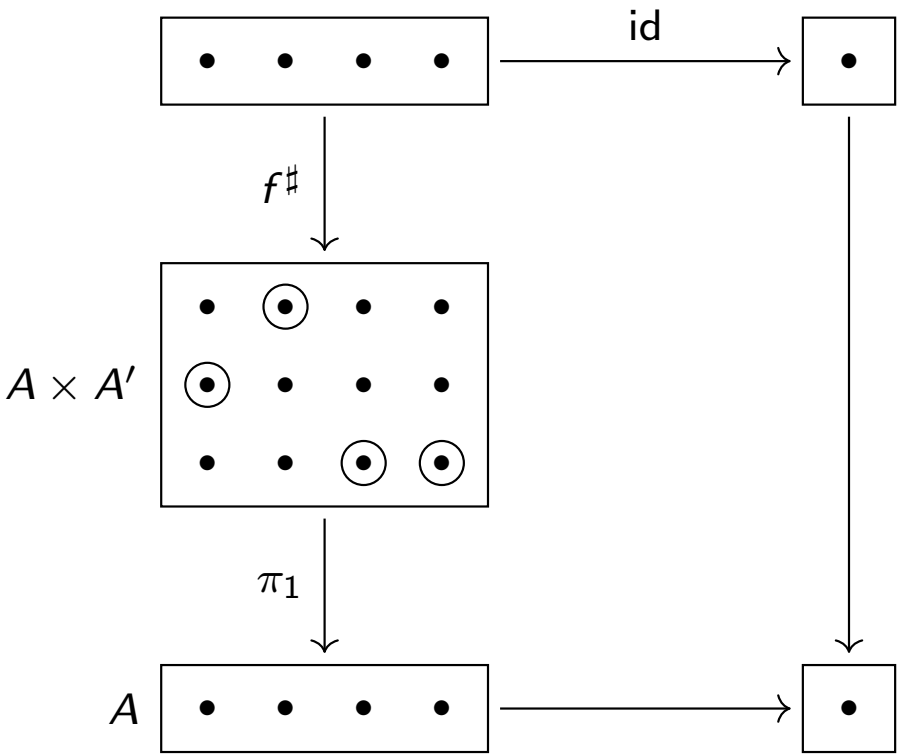
Example

Given a trivial bundle $\binom{A}{A'}$, let's visualize a map to $\binom{1}{1}$.



Example

Given a trivial bundle $\binom{A}{A'}$, let's visualize a map to $\binom{1}{1}$.



This can be identified with a map $f^\sharp \colon A \to A'$.

How to think about Lens

This suggests the following way of thinking of (generalized) lenses.

- An object $\binom{A}{A'}$ consists of contexts and actions
  - A is the contexts; in each $a : A$ there are $A'(a)$ actions available.

How to think about Lens

This suggests the following way of thinking of (generalized) lenses.

- An object $\binom{A}{A'}$ consists of contexts and actions
  - A is the contexts; in each $a : A$ there are $A'(a)$ actions available.
  - Example $\binom{S}{S}$. At each $s : S$, where in S do you want to go next?

How to think about Lens

This suggests the following way of thinking of (generalized) lenses.

- An object $\binom{A}{A'}$ consists of contexts and actions
  - A is the contexts; in each $a : A$ there are $A'(a)$ actions available.
  - Example $\binom{S}{S}$. At each $s : S$, where in S do you want to go next?
  - Example $\binom{S}{TS}$. At each $s : S$, which tangent direction to go in?
- A morphism $\binom{f}{f^\sharp} : \binom{A}{A'} \to \binom{B}{B'}$ is like A giving control to B.
  - Each context $a : A$ is communicated by f to give $fa : B$.
  - For each action $b' : B'(fa)$, there is an action $f^\sharp(b') : A'(a)$.

How to think about Lens

This suggests the following way of thinking of (generalized) lenses.

- An object $\binom{A}{A'}$ consists of contexts and actions
  - A is the contexts; in each $a : A$ there are $A'(a)$ actions available.
  - Example $\binom{S}{S}$. At each $s : S$, where in S do you want to go next?
  - Example $\binom{S}{TS}$. At each $s : S$, which tangent direction to go in?
- A morphism $\binom{f}{f^\sharp} : \binom{A}{A'} \to \binom{B}{B'}$ is like A giving control to B.
  - Each context $a : A$ is communicated by f to give $fa : B$.
  - For each action $b' : B'(fa)$, there is an action $f^\sharp(b') : A'(a)$.
- Example: A map $\binom{S}{S} \to \binom{A}{A'}$ can be identified with a kind of graph.
  - It's a graph with vertex set S, each with a color in A, and
  - $A'(a)$ outgoing arrows for each color $a : A$.

How to think about Lens

This suggests the following way of thinking of (generalized) lenses.

- An object $\binom{A}{A'}$ consists of contexts and actions
  - A is the contexts; in each $a : A$ there are $A'(a)$ actions available.
  - Example $\binom{S}{S}$. At each $s : S$, where in S do you want to go next?
  - Example $\binom{S}{TS}$. At each $s : S$, which tangent direction to go in?
- A morphism $\binom{f}{f^\sharp} : \binom{A}{A'} \to \binom{B}{B'}$ is like A giving control to B.
  - Each context $a : A$ is communicated by f to give $fa : B$.
  - For each action $b' : B'(fa)$, there is an action $f^\sharp(b') : A'(a)$.
- Example: A map $\binom{S}{S} \to \binom{A}{A'}$ can be identified with a kind of graph.
  - It's a graph with vertex set S, each with a color in A, and
  - $A'(a)$ outgoing arrows for each color $a : A$.
  - Special case $A = A' = 1$, a discrete dynamical system!

Notation

We denote by $\binom{B}{E}$ the bundle whose

- base space is B
- fiber over $b : B$ is $E(b)$.
- Right now we're agnostic about what our base category is.

Notation

We denote by $\binom{B}{E}$ the bundle whose

- base space is B
- fiber over $b : B$ is $E(b)$.
- Right now we're agnostic about what our base category is.

Examples:

- If S is a manifold and $TS(s)$ is the tangent space, we write $\binom{S}{TS}$.
- If B' is a set and $E(b) = B'$ for all $b : B$, we'd denote this $\binom{B}{B'}$

Notation

We denote by $\binom{B}{E}$ the bundle whose

- base space is B
- fiber over $b : B$ is $E(b)$.
- Right now we're agnostic about what our base category is.

Examples:

- If S is a manifold and $TS(s)$ is the tangent space, we write $\binom{S}{TS}$.
- If B' is a set and $E(b) = B'$ for all $b : B$, we'd denote this $\binom{B}{B'}$

Note that $\binom{B}{B'}$ really means the trivial bundle $B \times B' \to B$.

- I've recently been denoting $\binom{B}{E}$ by $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$.
- So we have $\left[\begin{smallmatrix} TS \\ S \end{smallmatrix}\right]$. Looks more bundley.
- If we want, we could reserve $\binom{A}{A'}$ for usual lenses (trivial bundles).

Ringed spaces

In algebraic geometry they study *ringed spaces* $(X, \mathcal{O}_X)$.

- Here X is a topological space and $\mathcal{O}_X$ is a sheaf of rings on it.
- We can think of $\mathcal{O}_X$ as a bundle with a fiber-wise ring structure.
- (This is necessary, not sufficient, but pretty close.)

Ringed spaces

In algebraic geometry they study *ringed spaces* $(X, \mathcal{O}_X)$.

- Here X is a topological space and $\mathcal{O}_X$ is a sheaf of rings on it.
- We can think of $\mathcal{O}_X$ as a bundle with a fiber-wise ring structure.
- (This is necessary, not sufficient, but pretty close.)

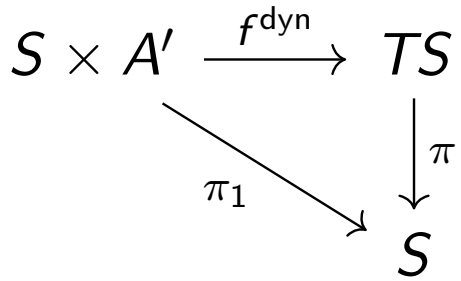
A morphism of ringed spaces $(f, f^\sharp) \colon (X, \mathcal{O}_X) \to (Y, \mathcal{O}_Y)$ is:

- A continuous map $f \colon X \to Y$
- A map of sheaves $f^* \mathcal{O}_Y \to \mathcal{O}_X$.
- That is, it's a map $\left[\begin{smallmatrix} \mathcal{O}_X \\ X \end{smallmatrix}\right] \to \left[\begin{smallmatrix} \mathcal{O}_Y \\ Y \end{smallmatrix}\right]$.

Continuous dynamical systems

Recall: if A', A are manifolds, a *continuous dynamical system* is:

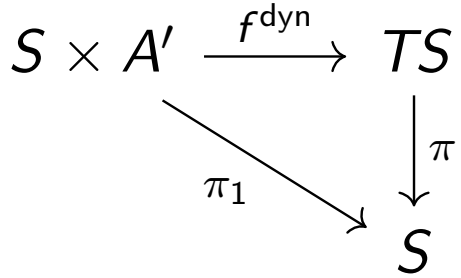
- A manifold S, (tangent bundle TS),
- A differentiable map $f^{\text{rdt}} : S \to A$,
- A differentiable map $f^{\text{dyn}} : S \times A' \to TS$



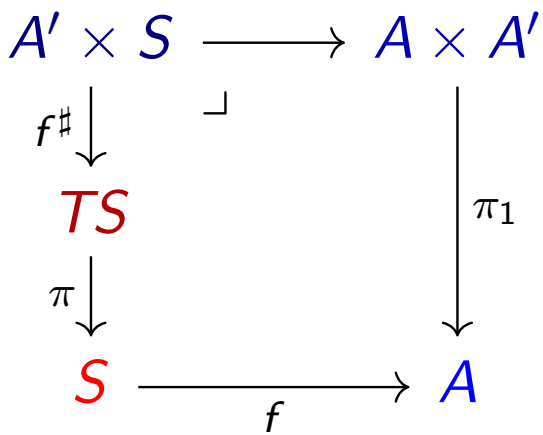
Continuous dynamical systems

Recall: if A', A are manifolds, a *continuous dynamical system* is:

- A manifold S, (tangent bundle TS),
- A differentiable map $f^{\text{rdt}} : S \to A$,
- A differentiable map $f^{\text{dyn}} : S \times A' \to TS$



But from the bundle perspective that commutative diagram is baked in.



In other words the dynamical system is just a lens map $\left[\begin{smallmatrix} TS \\ S \end{smallmatrix}\right] \to \left[\begin{smallmatrix} A' \\ A \end{smallmatrix}\right]$

More principled view update

Here's a principled notion of view-update for databases.

- You have two schemas B_1, B_2 and a profunctor $Q\colon B_1 \nrightarrow B_2$.

More principled view update

Here's a principled notion of view-update for databases.

- You have two schemas B_1, B_2 and a profunctor $Q\colon B_1 \nrightarrow B_2$.
- For any instance $I_1 : B_1$-**Inst**, we query to get an instance $Q_* I_1$.
  - If we insert or deduplicate it, i.e. have $Q_* I_1 \to I_2$...
  - ... we can form the pushout of $(I_1 \leftarrow Q^* Q I_1 \to Q^* I_2)$.
  - Here Q^* is the coquery left adjoint to the query Q_*.
- This is a universal construction. I_1/B_1-**Inst** $\leftrightarrows$ QI_1/B_2-**Inst**.

More principled view update

Here's a principled notion of view-update for databases.

- You have two schemas B_1, B_2 and a profunctor $Q\colon B_1 \nrightarrow B_2$.
- For any instance $I_1 : B_1\text{-}\mathbf{Inst}$, we query to get an instance $Q_* I_1$.
  - If we insert or deduplicate it, i.e. have $Q_* I_1 \to I_2$...
  - ... we can form the pushout of $(I_1 \leftarrow Q^* Q I_1 \to Q^* I_2)$.
  - Here Q^* is the coquery left adjoint to the query Q_*.
- This is a universal construction. $I_1/B_1\text{-}\mathbf{Inst} \leftrightarrows QI_1/B_2\text{-}\mathbf{Inst}$.

$$\begin{array}{ccc}
\displaystyle\sum_{I_1 : B_1\text{-}\mathbf{Inst}} Q_* I_1/B_2\text{-}\mathbf{Inst} & \longrightarrow & \displaystyle\sum_{I_2 : B_2\text{-}\mathbf{Inst}} I_2/B_2\text{-}\mathbf{Inst} \\
\Big\downarrow{\scriptstyle \text{pushout as above}} & \lrcorner & \Big\downarrow{\scriptstyle \pi_1} \\
\displaystyle\sum_{I_1 : B_1\text{-}\mathbf{Inst}} I_1/B_1\text{-}\mathbf{Inst} & & \\
\Big\downarrow{\scriptstyle \pi_1} & & \\
B_1\text{-}\mathbf{Inst} & \xrightarrow[Q_*]{} & B_2\text{-}\mathbf{Inst}
\end{array}$$

What do all these have in common?

Ringed spaces, continuous dynamical systems, and view update.

What do all these have in common?

Ringed spaces, continuous dynamical systems, and view update.

- A category $\mathcal{B}$ where the bases live $f : B_1 \to B_2$.

What do all these have in common?

Ringed spaces, continuous dynamical systems, and view update.

- A category $\mathcal{B}$ where the bases live $f : B_1 \to B_2$.
- For each base B, a category $\mathcal{E}(B)$ of possible "bundles" over B.

What do all these have in common?

Ringed spaces, continuous dynamical systems, and view update.

- A category $\mathcal{B}$ where the bases live $f : B_1 \to B_2$.
- For each base B, a category $\mathcal{E}(B)$ of possible "bundles" over B.
- For each map $f : B_1 \to B_2$ and bundle $E_2 \in \mathcal{E}(B_2)$, ...
- ... a notion of pullback $f^* E_2 \in \mathcal{E}(B_1)$.

What do all these have in common?

Ringed spaces, continuous dynamical systems, and view update.

- A category $\mathcal{B}$ where the bases live $f : B_1 \to B_2$.
- For each base B, a category $\mathcal{E}(B)$ of possible "bundles" over B.
- For each map $f : B_1 \to B_2$ and bundle $E_2 \in \mathcal{E}(B_2)$, ...
- ... a notion of pullback $f^* E_2 \in \mathcal{E}(B_1)$.

That is, a category $\mathcal{B}$ and a functor $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

- Then define $\mathbf{Lens}_{\mathcal{E}}$ as a variant of the Grothendieck construction.

What do all these have in common?

Ringed spaces, continuous dynamical systems, and view update.

- A category $\mathcal{B}$ where the bases live $f : B_1 \to B_2$.
- For each base B, a category $\mathcal{E}(B)$ of possible "bundles" over B.
- For each map $f : B_1 \to B_2$ and bundle $E_2 \in \mathcal{E}(B_2)$, ...
- ... a notion of pullback $f^* E_2 \in \mathcal{E}(B_1)$.

That is, a category $\mathcal{B}$ and a functor $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

- Then define $\mathbf{Lens}_{\mathcal{E}}$ as a variant of the Grothendieck construction.
- The following are equivalent:
  - The opposite of the covariant Grothendieck construction for $\mathcal{E}$.
  - The contravariant Grothendieck for the pointwise opposite of $\mathcal{E}$.
  - The category with
    - objects $\{(B, E) \mid B \in \mathcal{B}, E \in \mathcal{E}(B)\}$
    - morphisms $(B_1, E_1) \to (B_2, E_2)$ given by "lensy" $(f, f^\sharp)$.

Lenses in any symmetric monoidal category

If $(\mathcal{M}, I, \otimes)$ is any SMC, there is a notion of lenses in it.

Lenses in any symmetric monoidal category

If $(\mathcal{M}, I, \otimes)$ is any SMC, there is a notion of lenses in it.

- Objects are pairs $\left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right]$ where m is an object and...
- ... c is a comonoid; i.e. it implicitly has $\epsilon\colon c \to I$ and $\delta\colon c \to c \otimes c$.

Lenses in any symmetric monoidal category

If $(\mathcal{M}, I, \otimes)$ is any SMC, there is a notion of lenses in it.

- Objects are pairs $\left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right]$ where m is an object and...
- ... c is a comonoid; i.e. it implicitly has $\epsilon\colon c \to I$ and $\delta\colon c \to c \otimes c$.
- A morphism $\left[\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right] : \left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right] \to \left[\begin{smallmatrix} m' \\ c' \end{smallmatrix}\right]$ consists of
  - a comonoid homomorphism $f : c \to c'$ and
  - a morphism $f^\sharp\colon c \otimes m' \to m$.

Lenses in any symmetric monoidal category

If $(\mathcal{M}, I, \otimes)$ is any SMC, there is a notion of lenses in it.

- Objects are pairs $\left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right]$ where m is an object and...
- ... c is a comonoid; i.e. it implicitly has $\epsilon\colon c \to I$ and $\delta\colon c \to c \otimes c$.
- A morphism $\left[\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right] : \left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right] \to \left[\begin{smallmatrix} m' \\ c' \end{smallmatrix}\right]$ consists of
  - a comonoid homomorphism $f : c \to c'$ and
  - a morphism $f^\sharp\colon c \otimes m' \to m$.
- Example: $(\mathbf{Set}, 1, \times)$
  - Every object and morphism has a unique comonoid structure.
  - So the above description just reduces to the one we know.

Lenses in any symmetric monoidal category

If $(\mathcal{M}, I, \otimes)$ is any SMC, there is a notion of lenses in it.

- Objects are pairs $\left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right]$ where m is an object and...
- ... c is a comonoid; i.e. it implicitly has $\epsilon\colon c \to I$ and $\delta\colon c \to c \otimes c$.
- A morphism $\left[\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right] : \left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right] \to \left[\begin{smallmatrix} m' \\ c' \end{smallmatrix}\right]$ consists of
  - a comonoid homomorphism $f : c \to c'$ and
  - a morphism $f^\sharp\colon c \otimes m' \to m$.
- Example: $(\mathbf{Set}, 1, \times)$
  - Every object and morphism has a unique comonoid structure.
  - So the above description just reduces to the one we know.

So how can we see this in the general $\mathcal{E}\colon \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$ setup?

Lenses in any symmetric monoidal category

If $(\mathcal{M}, I, \otimes)$ is any SMC, there is a notion of lenses in it.

- Objects are pairs $\left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right]$ where m is an object and...
- ... c is a comonoid; i.e. it implicitly has $\epsilon\colon c \to I$ and $\delta\colon c \to c \otimes c$.
- A morphism $\left[\begin{smallmatrix} f^\sharp \\ f \end{smallmatrix}\right] : \left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right] \to \left[\begin{smallmatrix} m' \\ c' \end{smallmatrix}\right]$ consists of
  - a comonoid homomorphism $f : c \to c'$ and
  - a morphism $f^\sharp\colon c \otimes m' \to m$.
- Example: $(\mathbf{Set}, 1, \times)$
  - Every object and morphism has a unique comonoid structure.
  - So the above description just reduces to the one we know.

So how can we see this in the general $\mathcal{E}\colon \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$ setup?

- Take $\mathcal{B} := \{\text{comonoids } (c, \epsilon, \delta) \text{ in } \mathcal{M}\}$
- Take $\mathcal{E}(c) := \mathbf{coKl}(c \otimes -)$, the coKleisli cat. of comonad $x \mapsto c \otimes x$.
- In $\left[\begin{smallmatrix} m \\ c \end{smallmatrix}\right]$, think of m as the product coalgebra $c \otimes m$, "trivial bundle".

The right generalization of Lens?

We've seen many different lens-like categories $\mathcal{L}$.

- For each, there's a category $\mathcal{B}$ and a functor $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$ ...
- ... whose (mixed-variance) Grothendieck is $\mathbf{Lens}_{\mathcal{E}} \cong \mathcal{L}$.

The right generalization of Lens?

We've seen many different lens-like categories $\mathcal{L}$.

- For each, there's a category $\mathcal{B}$ and a functor $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$ ...
- ... whose (mixed-variance) Grothendieck is $\mathbf{Lens}_{\mathcal{E}} \cong \mathcal{L}$.

But this is perhaps overly general.

- Every split fibration is $\mathbf{Lens}_{\mathcal{E}}$ for some $\mathcal{E}$.
- Should we be looking for something more specific than this?
- Example: the twisted arrow category is $\mathbf{Lens}_{\_/\mathcal{C}}$; too general?

$$\begin{array}{ccc} E_1 & \xleftarrow{f^\sharp} & E_2 \\ {\scriptstyle p_1}\downarrow & & \downarrow{\scriptstyle p_2} \\ B_1 & \xrightarrow[f]{} & B_2 \end{array}$$

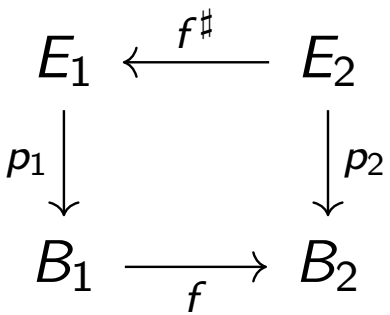
The right generalization of Lens?

We've seen many different lens-like categories $\mathcal{L}$.

- For each, there's a category $\mathcal{B}$ and a functor $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$ ...
- ... whose (mixed-variance) Grothendieck is $\mathbf{Lens}_{\mathcal{E}} \cong \mathcal{L}$.

But this is perhaps overly general.

- Every split fibration is $\mathbf{Lens}_{\mathcal{E}}$ for some $\mathcal{E}$.
- Should we be looking for something more specific than this?
- Example: the twisted arrow category is $\mathbf{Lens}_{\_/\mathcal{C}}$; too general?



A morphism $\begin{bmatrix} E_1 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$ in the twisted arrow category. Looks kinda lensy, but should it count?

Formal properties of $\mathbf{Lens}_{\mathcal{E}}$

The properties of $\mathbf{Lens}_{\mathcal{E}}$ depend on choice of $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

Formal properties of $\mathbf{Lens}_{\mathcal{E}}$

The properties of $\mathbf{Lens}_{\mathcal{E}}$ depend on choice of $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

- Always: get a "vertical-cartesian" factorization system.
  - Each $\begin{bmatrix} f^\sharp \\ f \end{bmatrix} : \begin{bmatrix} E_1 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$ factors as $\begin{bmatrix} E_1 \\ b_1 \end{bmatrix} \to \begin{bmatrix} f^* E_2 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$
- Always: if $\mathcal{B}$ is an SMC and $\mathcal{E}$ is lax monoidal, $\mathbf{Lens}_{\mathcal{E}}$ is an SMC.

Formal properties of $\mathbf{Lens}_{\mathcal{E}}$

The properties of $\mathbf{Lens}_{\mathcal{E}}$ depend on choice of $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

- Always: get a "vertical-cartesian" factorization system.
  - Each $\begin{bmatrix} f^\sharp \\ f \end{bmatrix} : \begin{bmatrix} E_1 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$ factors as $\begin{bmatrix} E_1 \\ b_1 \end{bmatrix} \to \begin{bmatrix} f^* E_2 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$
- Always: if $\mathcal{B}$ is an SMC and $\mathcal{E}$ is lax monoidal, $\mathbf{Lens}_{\mathcal{E}}$ is an SMC.

A nice case: the slice functor: $\mathcal{B}/- : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$

- This works if $\mathcal{B}$ has pullbacks.
  - It sends $B \mapsto \mathcal{B}/B$, the category of bundles.
  - So an object $\begin{bmatrix} E \\ B \end{bmatrix} \in \mathbf{Lens}_{\mathcal{B}/-}$ is just a map $E \to B$.

Formal properties of $\mathbf{Lens}_{\mathcal{E}}$

The properties of $\mathbf{Lens}_{\mathcal{E}}$ depend on choice of $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

- Always: get a "vertical-cartesian" factorization system.
  - Each $\begin{bmatrix} f^\sharp \\ f \end{bmatrix} : \begin{bmatrix} E_1 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$ factors as $\begin{bmatrix} E_1 \\ b_1 \end{bmatrix} \to \begin{bmatrix} f^* E_2 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$
- Always: if $\mathcal{B}$ is an SMC and $\mathcal{E}$ is lax monoidal, $\mathbf{Lens}_{\mathcal{E}}$ is an SMC.

A nice case: the slice functor: $\mathcal{B}/- : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$

- This works if $\mathcal{B}$ has pullbacks.
  - It sends $B \mapsto \mathcal{B}/B$, the category of bundles.
  - So an object $\begin{bmatrix} E \\ B \end{bmatrix} \in \mathbf{Lens}_{\mathcal{B}/-}$ is just a map $E \to B$.
- If $\mathcal{B}$ is locally cartesian closed with disjoint coproducts (e.g. a topos):

Formal properties of $\mathbf{Lens}_{\mathcal{E}}$

The properties of $\mathbf{Lens}_{\mathcal{E}}$ depend on choice of $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

- Always: get a "vertical-cartesian" factorization system.
  - Each $\begin{bmatrix} f^\sharp \\ f \end{bmatrix} : \begin{bmatrix} E_1 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$ factors as $\begin{bmatrix} E_1 \\ b_1 \end{bmatrix} \to \begin{bmatrix} f^* E_2 \\ B_1 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$
- Always: if $\mathcal{B}$ is an SMC and $\mathcal{E}$ is lax monoidal, $\mathbf{Lens}_{\mathcal{E}}$ is an SMC.

A nice case: the slice functor: $\mathcal{B}/- : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$

- This works if $\mathcal{B}$ has pullbacks.
  - It sends $B \mapsto \mathcal{B}/B$, the category of bundles.
  - So an object $\begin{bmatrix} E \\ B \end{bmatrix} \in \mathbf{Lens}_{\mathcal{B}/-}$ is just a map $E \to B$.
- If $\mathcal{B}$ is locally cartesian closed with disjoint coproducts (e.g. a topos):
- Then $\mathbf{Lens}_{\mathcal{B}/-}$ has excellent formal properties.
  - Complete, cocomplete, Cartesian closed.
  - Initial alg's and final coalg's for polynomial endofunctors.
  - Another fact'n system: $\begin{bmatrix} f^\sharp \\ f \end{bmatrix}$ factors as $\begin{bmatrix} E_1 \\ B_1 \end{bmatrix} \to \begin{bmatrix} f_* E_1 \\ B_2 \end{bmatrix} \to \begin{bmatrix} E_2 \\ B_2 \end{bmatrix}$

Polynomial functors

There's a strong connection to polynomial functors, aka *containers*.

Polynomial functors

There's a strong connection to polynomial functors, aka *containers*.

- These are used a lot in functional programming.
- Provide data structures like lists, binary trees, trees, streams, etc.
- Called polynomials because they send x to e.g. x^4+3x^2+2x+1.

Polynomial functors

There's a strong connection to polynomial functors, aka *containers*.

- These are used a lot in functional programming.
- Provide data structures like lists, binary trees, trees, streams, etc.
- Called polynomials because they send x to e.g. x^4+3x^2+2x+1.

Setting: suppose $\mathcal{B}$ is locally cartesian closed with disjoint coproducts.

- We're looking at $\mathbf{Lens}_{\mathcal{B}/\_}$, i.e. objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ are maps $E \to B$ in $\mathcal{B}$.

Polynomial functors

There's a strong connection to polynomial functors, aka *containers*.

- These are used a lot in functional programming.
- Provide data structures like lists, binary trees, trees, streams, etc.
- Called polynomials because they send x to e.g. x^4+3x^2+2x+1.

Setting: suppose $\mathcal{B}$ is locally cartesian closed with disjoint coproducts.

- We're looking at $\mathbf{Lens}_{\mathcal{B}/-}$, i.e. objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ are maps $E \to B$ in $\mathcal{B}$.
- An object $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ is the same data as a polynomial functor!
  - It would denote the functor sending $x \mapsto \sum_{b:B} x^{E(b)}$.

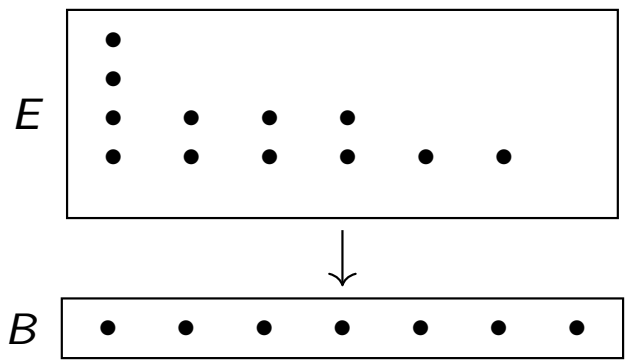
Polynomial functors

There's a strong connection to polynomial functors, aka *containers*.

- These are used a lot in functional programming.
- Provide data structures like lists, binary trees, trees, streams, etc.
- Called polynomials because they send x to e.g. x^4+3x^2+2x+1.

Setting: suppose $\mathcal{B}$ is locally cartesian closed with disjoint coproducts.

- We're looking at $\mathbf{Lens}_{\mathcal{B}/-}$, i.e. objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ are maps $E \to B$ in $\mathcal{B}$.
- An object $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ is the same data as a polynomial functor!
  - It would denote the functor sending $x \mapsto \sum_{b:B} x^{E(b)}$.
  - E.g. the above polynomial corresponds to the following bundle



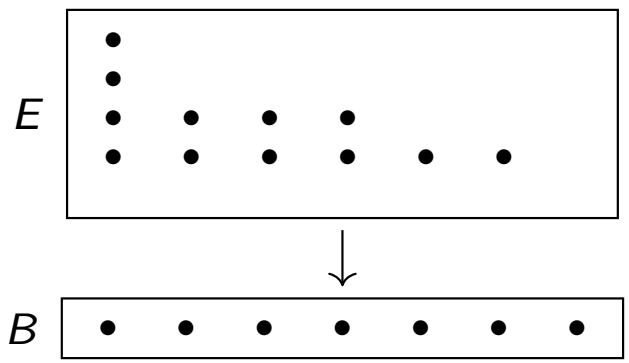
Polynomial functors

There's a strong connection to polynomial functors, aka *containers*.

- These are used a lot in functional programming.
- Provide data structures like lists, binary trees, trees, streams, etc.
- Called polynomials because they send x to e.g. x^4+3x^2+2x+1.

Setting: suppose $\mathcal{B}$ is locally cartesian closed with disjoint coproducts.

- We're looking at $\mathbf{Lens}_{\mathcal{B}/-}$, i.e. objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ are maps $E \to B$ in $\mathcal{B}$.
- An object $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ is the same data as a polynomial functor!
  - It would denote the functor sending $x \mapsto \sum_{b:B} x^{E(b)}$.
  - E.g. the above polynomial corresponds to the following bundle



- And they have the same morphisms too: $\mathbf{Poly}_{\mathcal{B}} \cong \mathbf{Lens}_{\mathcal{B}/-}$.

Polynomial functor interpretation

How should we interpret the isomorphism $\mathbf{Poly}_{\mathbf{Set}} \cong \mathbf{Lens}_{\mathbf{Set}/\_}$?

Polynomial functor interpretation

How should we interpret the isomorphism $\mathbf{Poly}_{\mathbf{Set}} \cong \mathbf{Lens}_{\mathbf{Set}/\_}$?

- How are objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ functors exactly?
- And how are lenses $\left[\begin{smallmatrix} E_1 \\ B_1 \end{smallmatrix}\right] \to \left[\begin{smallmatrix} E_2 \\ B_2 \end{smallmatrix}\right]$ natural transformations?

Polynomial functor interpretation

How should we interpret the isomorphism $\mathbf{Poly}_{\mathbf{Set}} \cong \mathbf{Lens}_{\mathbf{Set}/-}$?

- How are objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ functors exactly?
- And how are lenses $\left[\begin{smallmatrix} E_1 \\ B_1 \end{smallmatrix}\right] \to \left[\begin{smallmatrix} E_2 \\ B_2 \end{smallmatrix}\right]$ natural transformations?

Answer:

- Interpreting $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ as a functor, it sends X to the set $\mathbf{Lens}\left(\left[\begin{smallmatrix} X \\ 1 \end{smallmatrix}\right], \left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]\right)$.

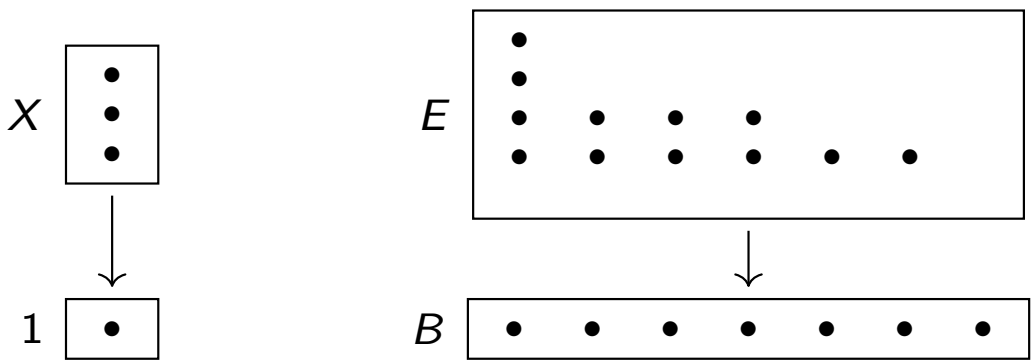
Polynomial functor interpretation

How should we interpret the isomorphism $\mathbf{Poly}_{\mathbf{Set}} \cong \mathbf{Lens}_{\mathbf{Set}/-}$?

- How are objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ functors exactly?
- And how are lenses $\left[\begin{smallmatrix} E_1 \\ B_1 \end{smallmatrix}\right] \to \left[\begin{smallmatrix} E_2 \\ B_2 \end{smallmatrix}\right]$ natural transformations?

Answer:

- Interpreting $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ as a functor, it sends X to the set $\mathbf{Lens}\left(\left[\begin{smallmatrix} X \\ 1 \end{smallmatrix}\right], \left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]\right)$.



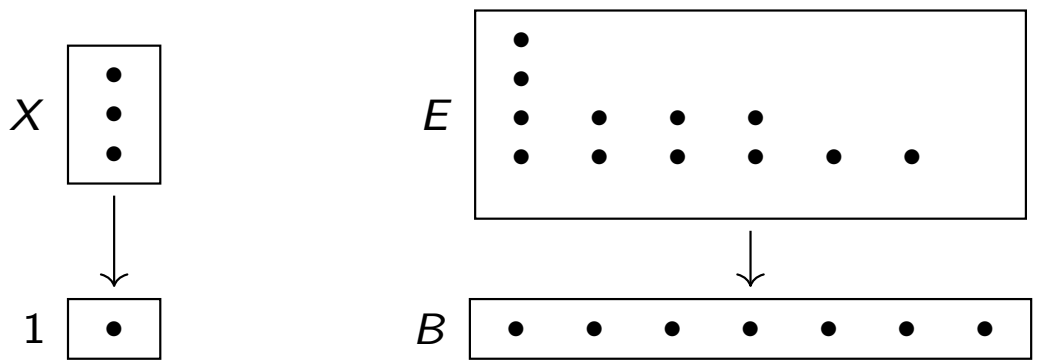
Polynomial functor interpretation

How should we interpret the isomorphism $\mathbf{Poly}_{\mathbf{Set}} \cong \mathbf{Lens}_{\mathbf{Set}/-}$?

- How are objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ functors exactly?
- And how are lenses $\left[\begin{smallmatrix} E_1 \\ B_1 \end{smallmatrix}\right] \to \left[\begin{smallmatrix} E_2 \\ B_2 \end{smallmatrix}\right]$ natural transformations?

Answer:

- Interpreting $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ as a functor, it sends X to the set $\mathbf{Lens}\left(\left[\begin{smallmatrix} X \\ 1 \end{smallmatrix}\right], \left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]\right)$.



- Do you see why this sends X to $X^4 + 3X^2 + 2X + 1$?

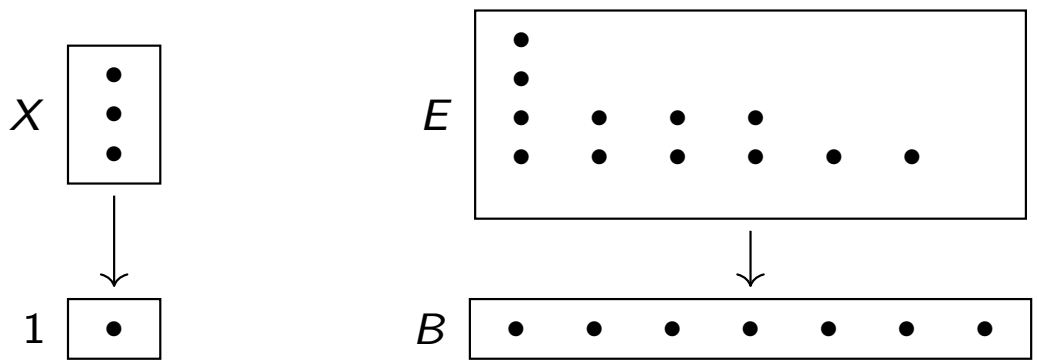
Polynomial functor interpretation

How should we interpret the isomorphism $\mathbf{Poly}_{\mathbf{Set}} \cong \mathbf{Lens}_{\mathbf{Set}/-}$?

- How are objects $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ functors exactly?
- And how are lenses $\left[\begin{smallmatrix} E_1 \\ B_1 \end{smallmatrix}\right] \to \left[\begin{smallmatrix} E_2 \\ B_2 \end{smallmatrix}\right]$ natural transformations?

Answer:

- Interpreting $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$ as a functor, it sends X to the set $\mathbf{Lens}\left(\left[\begin{smallmatrix} X \\ 1 \end{smallmatrix}\right], \left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]\right)$.



- Do you see why this sends X to $X^4 + 3X^2 + 2X + 1$?
- The functor acts on a lens $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right] \to \left[\begin{smallmatrix} E' \\ B' \end{smallmatrix}\right]$ by composing with it.

Outline

1 Introduction

2 Some applications of lenses

3 Generalizing lens categories

4 Conclusion

Summary

Lenses seem to spring up in many different places.

- Functional programming; database transactions;
- Open games; supervised learning;
- Wiring diagrams; discrete, cts dynamic systems; hierarchical planning.

Summary

Lenses seem to spring up in many different places.

- Functional programming; database transactions;
- Open games; supervised learning;
- Wiring diagrams; discrete, cts dynamic systems; hierarchical planning.

We can make sense of their peculiar form $(B_1 \to B_2,\ B_1 \times E_2 \to E_1)$.

- Namely, we think in terms of bundles $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$.
- This perspective puts lenses in a more familiar categorical setting.
  - Used in algebraic geometry and theory of polynomial functors.

Summary

Lenses seem to spring up in many different places.

- Functional programming; database transactions;
- Open games; supervised learning;
- Wiring diagrams; discrete, cts dynamic systems; hierarchical planning.

We can make sense of their peculiar form $(B_1 \to B_2,\ B_1 \times E_2 \to E_1)$.

- Namely, we think in terms of bundles $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$.
- This perspective puts lenses in a more familiar categorical setting.
  - Used in algebraic geometry and theory of polynomial functors.
- The larger category of bundles has better formal properties
  - Coproducts, initial algebras, an extra factorization system, etc.

Summary

Lenses seem to spring up in many different places.

- Functional programming; database transactions;
- Open games; supervised learning;
- Wiring diagrams; discrete, cts dynamic systems; hierarchical planning.

We can make sense of their peculiar form $(B_1 \to B_2,\ B_1 \times E_2 \to E_1)$.

- Namely, we think in terms of bundles $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$.
- This perspective puts lenses in a more familiar categorical setting.
  - Used in algebraic geometry and theory of polynomial functors.
- The larger category of bundles has better formal properties
  - Coproducts, initial algebras, an extra factorization system, etc.
- In fact, one gets a sort of lensy feeling for any $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.

Summary

Lenses seem to spring up in many different places.

- Functional programming; database transactions;
- Open games; supervised learning;
- Wiring diagrams; discrete, cts dynamic systems; hierarchical planning.

We can make sense of their peculiar form $(B_1 \to B_2,\ B_1 \times E_2 \to E_1)$.

- Namely, we think in terms of bundles $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$.
- This perspective puts lenses in a more familiar categorical setting.
  - Used in algebraic geometry and theory of polynomial functors.
- The larger category of bundles has better formal properties
  - Coproducts, initial algebras, an extra factorization system, etc.
- In fact, one gets a sort of lensy feeling for any $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.
- Perhaps overly general. Find a better charicteriz'n of all things lensy?

Summary

Lenses seem to spring up in many different places.

- Functional programming; database transactions;
- Open games; supervised learning;
- Wiring diagrams; discrete, cts dynamic systems; hierarchical planning.

We can make sense of their peculiar form $(B_1 \to B_2,\ B_1 \times E_2 \to E_1)$.

- Namely, we think in terms of bundles $\left[\begin{smallmatrix} E \\ B \end{smallmatrix}\right]$.
- This perspective puts lenses in a more familiar categorical setting.
  - Used in algebraic geometry and theory of polynomial functors.
- The larger category of bundles has better formal properties
  - Coproducts, initial algebras, an extra factorization system, etc.
- In fact, one gets a sort of lensy feeling for any $\mathcal{E} : \mathcal{B}^{\mathrm{op}} \to \mathbf{Cat}$.
- Perhaps overly general. Find a better charicteriz'n of all things lensy?

Thanks; comments and questions welcome!