The Compositional Structure of Active Inference

Toby St Clere Smithe

Topos Institute
&
University of Oxford

17 March 2021

Preview: the structure of active inference

I will introduce the concept of **active inference doctrine**, a functorial way of packaging up everything needed for a free energy framework:

a category of **parameterized statistical games** which come with contextual fitness functions, along with a **dynamical semantics**, *i.e.* an assignment of systems that 'play' those games.

At the end, I will discuss some open problems and work-in-progress:
in particular, to deal with nested and nonstationary systems, and intervention.

Nonetheless, we can already capture any classical generative model,
and we get a notion of "Markov blanket" for free!

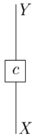
First, we will begin with a quick overview of compositional probability theory ...

Overview

Markov categories for (conditional) probability

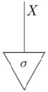
We work in a *Markov* or *copy-delete* category – **canonical example**: $\mathcal{K}\ell(\mathcal{D})$

Objects: spaces X, Y — sets X, Y



Morphisms: "stochastic channels" — *ie.* functions from points to 'beliefs'

$$X \nrightarrow Y \qquad X \to \mathcal{D}Y \;\cong\; X \times Y \to [0,1]$$



States: channels out of the monoidal unit — *ie.* probability distributions (formal convex sums)

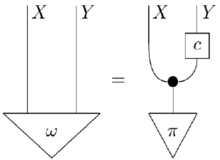
$$I \nrightarrow X \qquad X \to [0,1] \qquad \textstyle\sum_{x:X} \boxed{p(x)}\, |x\rangle$$

so general channels are like 'conditional' probability distributions,
and we adopt the standard notation $p(y|x) := p(x)(y)$

Composition: given $p : X \nrightarrow Y$ and $q : Y \nrightarrow Z$, "average over" Y — for example:

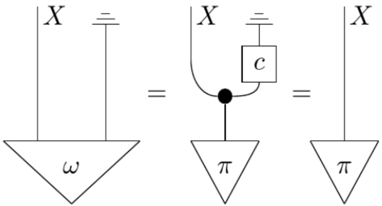
$$q \bullet p : X \to \mathcal{D}Z := x \mapsto \sum_{z:Z} \boxed{\sum_{y:Y} q(z|y) \cdot p(y|x)}\, |z\rangle$$

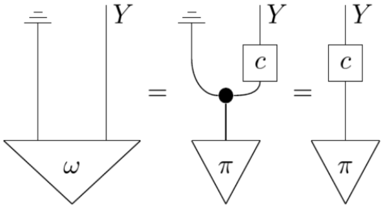
Joint states and generative models



$$P_\omega(A, B) = P_c(B|A) \cdot P_\pi(A)$$

With two *marginals* given by discarding:

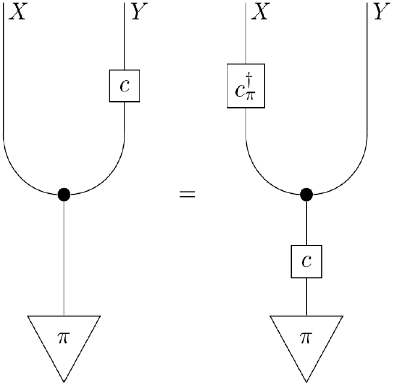




Bayes' rule, categorically

As a 'dagger' operation on stochastic channels:

$$\mathrm{P}_c(B|A) \cdot \mathrm{P}_\pi(A) = \mathrm{P}_{c_\pi^\dagger}(A|B) \cdot \mathrm{P}_{c \bullet \pi}(B)$$



Note that the Bayesian inverse channel depends on the prior!

The bidirectionality of Bayesian inference

- Given a 'generative' channel $c : X \to \mathcal{D}Y$, the corresponding 'recognition' channel has a **state-dependent** type $c^\dagger : \mathcal{D}X \times Y \to \mathcal{D}X$.
- A pair of a forwards map with a 'dependent' backwards map is called a **lens**.
- Lenses are a common pattern in 'bidirectional' systems (*e.g.* economic games, databases).
- I proved that the inverse of a composite channel (hierarchical generative model) is the lens composite of its components [1].
- This explains formally the bidirectional structure of predictive coding.
- And it turns out to be rather useful for our purposes today!

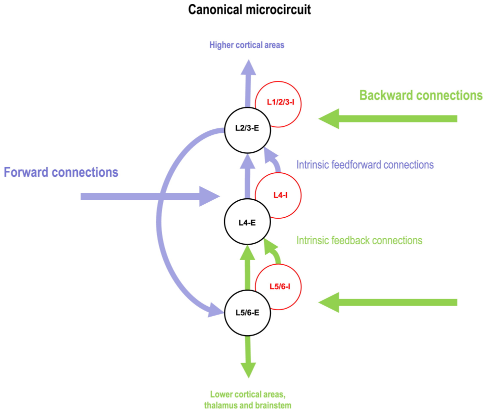
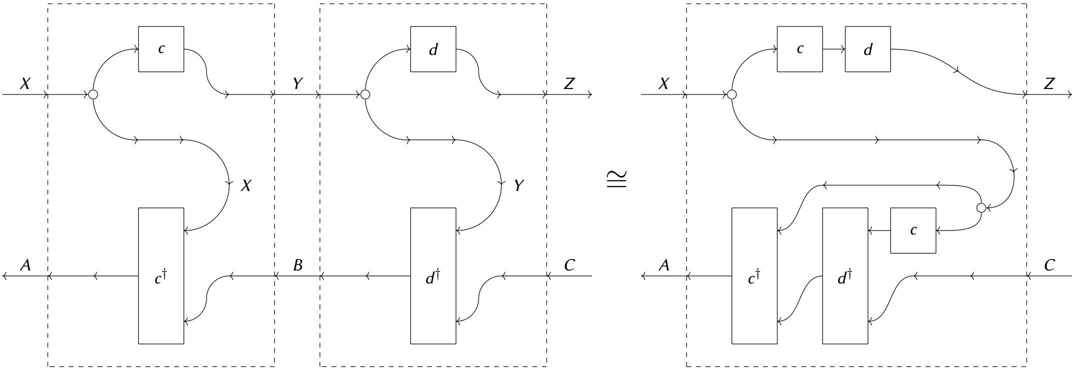


Figure: Bastos et al. [2]

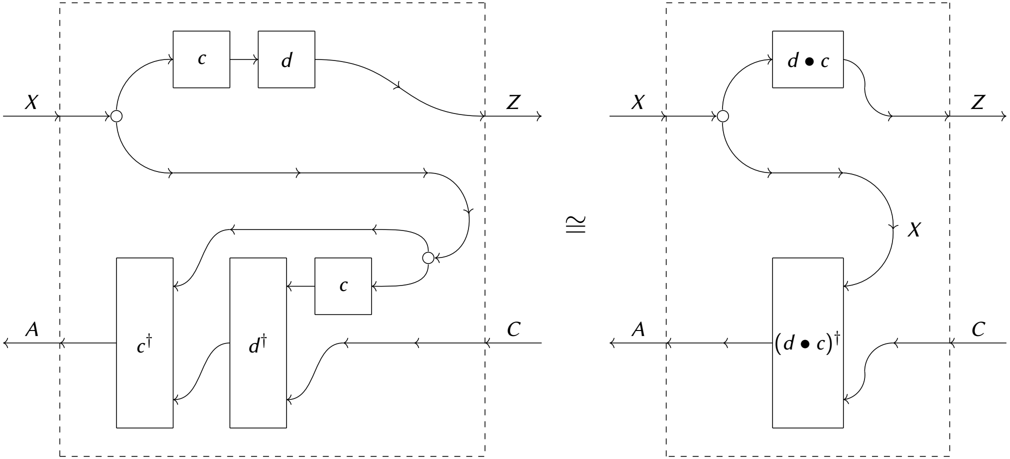
Bayesian lenses

These *Bayesian lenses* form the morphisms of a category whose objects are measurable spaces.
These morphisms (the lenses) compose like this:

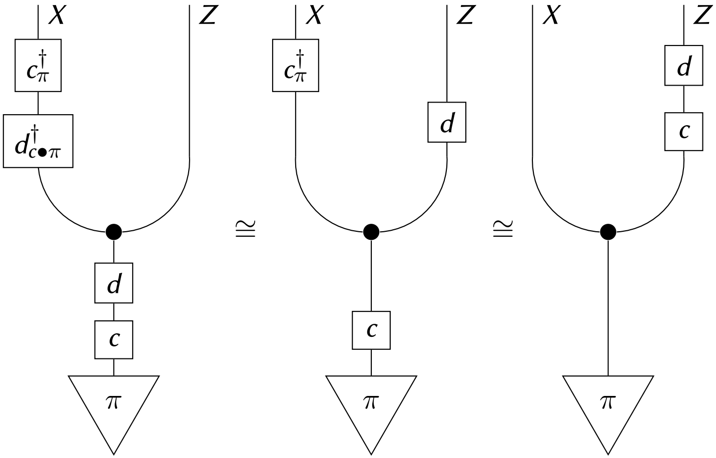


But how does the right-hand side backwards part relate to the inverse of the composite $d \bullet c$?
Answer: they are the same! So we have a nice compositional structure here.

Bayesian updates compose 'optically'



The graphical proof



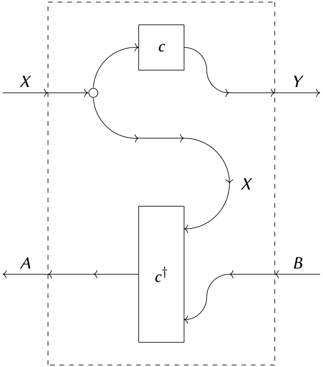
Overview

Statistical games: motivation

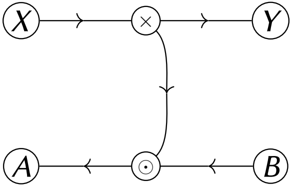
- Given a stochastic channel $c : X \to \mathcal{D}Y$ and a prior $\pi : \mathcal{D}X$, we can form the inversion $c^\dagger_\pi : Y \to \mathcal{D}X$. And we've seen that these pairs form lenses.
  Why is this useful?
- Typically, obtaining $c^\dagger_\pi$ is computationally difficult: we usually need to approximate it.
- This gives us a lot of freedom. Often, one approximation scheme might be 'better' than another, and we should like to quantify this.
- And, often, the fitness of our approximation depends on how it interacts with the world: the prior we choose, and the dataset we have.
- So the approximation is typically *context-dependent* and *parameterized*.
- We can formalize both aspects using ideas from 'open' game theory.

Depicting the ‘type’ of a lens

First, it will be useful to compress our depictions somewhat.
The lens on the left is an element of the set on the right:



$\in$



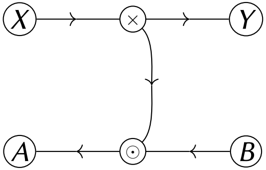
(Really, this just hides the channel labels...)

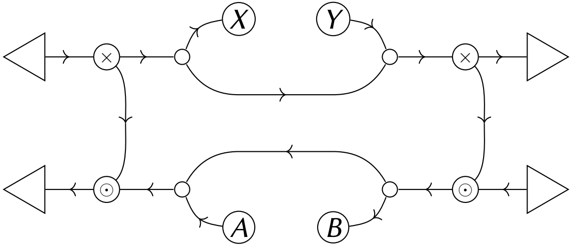
Context for inference problems

To formalize the context-dependence of the approximation fitness, we need to formalize *context*.

A context is a way of making an 'open' system into a 'closed' one: after all, closed systems don't have any context to depend on.

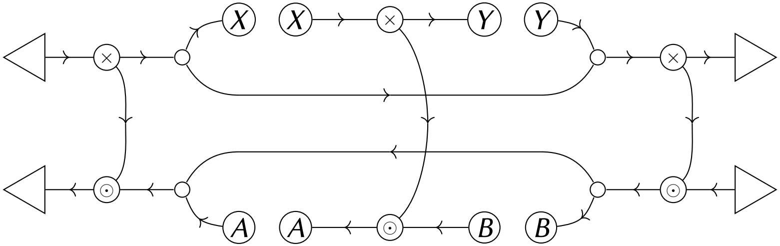
Given a lens with the type on the left, its context will have the type on the right:



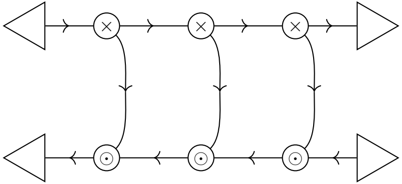


For abstract-nonsense reasons, this will close the lens ...

Context for inference problems

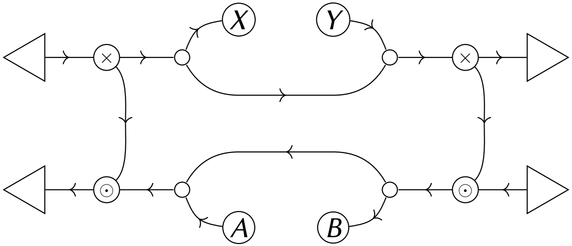


Context for inference problems



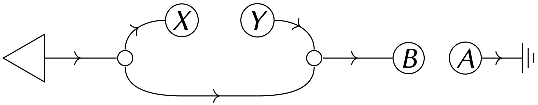
Context for inference problems

But what is this weird context thing anyway?



For categorical reasons, this thing simplifies into the following ...

Context for inference problems



And we can read this off as follows (you can trust me on this!):
a context for a Bayesian lens $(X, A) \nrightarrow (Y, B)$ is
a prior $\pi : \mathcal{D}X$ on X and a *continuation* channel $Y \to \mathcal{D}B$.

The continuation encodes how the lens' environment turns a prediction about Y into an (uncertain) observation in B.

We will call the set of these contexts (these prior-continuation pairs) $C\big((X, A), (Y, B)\big)$.

The category of statistical games

We can now define the basic category **SGame** of statistical games.

The objects of **SGame** will be the objects (X, A) of **BayesLens**.

Then a **statistical game** is a morphism $(X, A) \to (Y, B)$ in **SGame**:
a lens $(X, A) \nrightarrow (Y, B)$ paired with a *fitness function* $C\big((X, A), (Y, B)\big) \to \mathbb{R}$.

Composition of statistical games is lens composition paired with the sum of the 'local' fitnesses.
Identities are identity lenses (which just pass on information) with 0 fitness.

But note that there's not much freedom here! Where are the parameters?

Parameterized statistical games

In order that a system might have some control over its performance, there needs to be some freedom built into the morphisms themselves.

We therefore define the category **Para(SGame)** of **parameterized statistical games**.

It has the same objects (X, A) as before, but now each morphism $(X, A) \xrightarrow{\Sigma} (Y, B)$ comes with a parameter space Σ.

Concretely, the morphisms are parameterized Bayesian lenses $(\Sigma \times X, \Sigma \times A) \nrightarrow (Y, B)$ paired with fitness functions $C\big((\Sigma \times X, A), (\Sigma \times Y, B)\big) \to \mathbb{R}$.

Composing lenses $(\Sigma \times X, \Sigma \times A) \nrightarrow (Y, B)$ and $(\Gamma \times Y, \Gamma \times B) \nrightarrow (Z, C)$ gives a lens $(\Sigma \times \Gamma \times X, \Sigma \times \Gamma \times X) \nrightarrow (Z, C)$ parameterized in both spaces.

We are finally ready for some examples!

Example: maximum likelihood estimation

A Bayesian lens of the form $(1, 1) \nrightarrow (X, X)$ is fully specified by a state $\pi : 1 \to X$.

A context for such a lens is given by a trivial 'prior' on 1 and $k : X \to X$ is any endochannel on X. (Idea: "given a prediction, obtain a random observation".)

A **maximum likelihood game** is any game of type $(1, 1) \nrightarrow (X, X)$ for any $X : \mathcal{C}$, and whose loss function is $\mathbb{E}_{k \bullet \pi}\left[-\log p_\pi\right]$, where p_π is a density function for π.

NB: $\mathbb{E}_{x \sim \pi}[f] = \mathbb{E}(f \bullet \pi) = \int_{x:X} f(x)\,\pi(\mathrm{d}x)$ when $f : X \to \mathbb{R}$ and $\pi : \mathcal{C}(1, X)$.

Thinking of density as a measure of likelihood, note that an optimal strategy for an ML game is one that maximises the likelihood of the state obtained from the context.

Example: Bayesian inference

By generalizing the forwards part of the lens from states to channels, we obtain the following.

Fix a channel $c : Z \nrightarrow X$ with density function $p_c : X \times Z \to \mathbb{R}_+$ and a measure of divergence between states on Z, $D : \mathcal{C}(1, Z) \times \mathcal{C}(1, Z) \to \mathbb{R}$. A corresponding (generalized) **simple Bayesian inference game** is any game of type **BayesLens**$((Z, Z), (X, X))$ with loss function

$$\mathop{\mathbb{E}}_{x \sim k \bullet c \bullet \pi} \left[\mathop{\mathbb{E}}_{z \sim c'_\pi(x)} [-\log p_c(x|z)] + D(c'_\pi(x), \pi) \right]$$
$$= \mathop{\mathbb{E}}_{z \sim c'_\pi \bullet k \bullet c \bullet \pi} \left[-\int_X \log p_c(\mathrm{d}k \bullet c \bullet \pi | z) \right] + D(c'_\pi \bullet k \bullet c \bullet \pi, \pi)$$

where $\pi : I \nrightarrow Z$ and $k : X \nrightarrow X$, and where the second line follows from the first by linearity.

NB: since Bayesian updates compose optically, these games are closed under composition, giving hierarchical Bayesian inference games.

Example: variational autoencoder

By letting the choice of 'forwards' channel vary, we obtain **autoencoder** games.
Fix a family $\mathcal{F} \hookrightarrow \mathcal{C}(Z, X)$ of forward channels and a family $\mathcal{P} \hookrightarrow \mathcal{C}(X, Z)$ of backward channels such that each $c : \mathcal{F}$ admits a density function $p_c : X \otimes Z \to \mathbb{R}_+$ and each $d : \mathcal{P}$ admits a density function $q : Z \otimes X \to \mathbb{R}_+$. (This determines the lens parameterization: $\Sigma = \mathcal{F} \times \mathcal{P}$.)

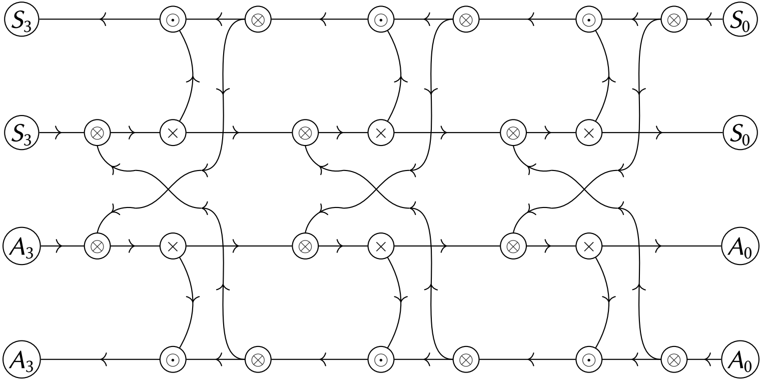
Then a **simple variational autoencoder game** is any game of type $(Z, Z) \to (X, X)$ with loss function given by the **free energy** $\phi(x)$ expected in the context (π, k):

$$\mathop{\mathbb{E}}_{x \sim k \bullet c \bullet \pi} [\phi(x)] = \mathop{\mathbb{E}}_{z \sim c'_\pi(x)} \left[\log \frac{q(z|x)}{p_c(x|z) p_\pi(z)} \right]$$

where $\pi : I \mathrel{\bullet\!\!\rightarrow} Z$ admits a density function $p_\pi : Z \to \mathbb{R}_+$, $q : Z \otimes X \to \mathbb{R}_+$ is a density function associated to c'_π, and k has type $X \mathrel{\bullet\!\!\rightarrow} X$.

Example: hierarchical active inference

And we can form not just ‘sequential’ hierarchies, but ‘parallel’ ones too, as in the following depiction of (the lens part of) a ‘hiearchical active inference’ game:



Overview

Dynamical semantics: motivation

OK, but we're interested in *models of living things*, not just abstract games!

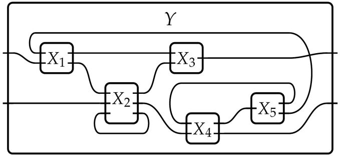
The free energy framework is an informal "process theory" of cognition: a specification of the functions that a brain should implement, along with a model of how it might implement them.

Such models are what I call **dynamical semantics**: the abstract games are like an algebraic description of the 'program' that is implemented by a system, and the dynamical semantics is what gives the system life.

I call a subcategory of (parameterized) statistical games along with a dynamical semantics an **active inference doctrine**.

Typically, we are interested in 'open' systems, away from thermodynamic equilibrium.

What is an ‘open’ dynamical system?



This is a **wiring diagram**.

An ‘open’ dynamical system is any dynamical system that could sensibly fill one of the boxes.

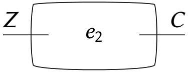
The dynamics typically depend on some inputs or parameters, and the system might produce outputs to be consumed by some other.

In deterministic discrete time, we might have a pair of functions update : $S \times I \to S$ and output : $S \to O$. But we can work quite generally.

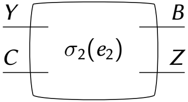
The key thing is that open systems can be wired together thus.
Wiring ‘inner’ boxes into ‘outer’ boxes gives a *multicategory*.

The multicategory (‘operad’) of wiring diagrams

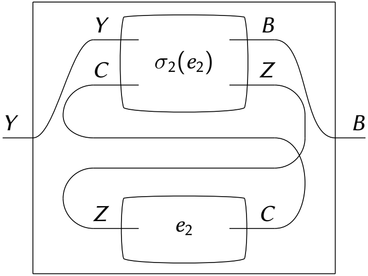
Towards a formal syntax for circuit diagrams: (input to output :: left to right)



Cy^Z



BZy^{YC}

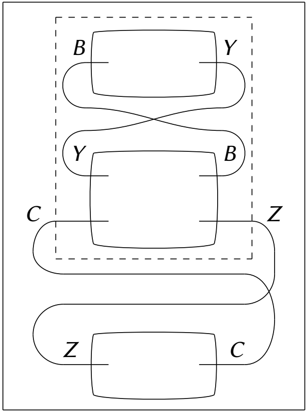


$BZy^{YC} \otimes Cy^Z \to By^Y$

NB: ‘Polynomial’ notation: coefficients are outputs, exponents are inputs

Dynamical 'contexts' and Markov blankets

We also have a notion of 'context' in the dynamical setting, closely related to Markov blankets:



$$Yy^B \otimes BZy^{YC} \otimes Cy^Z \to y$$

Here, the outer box makes a closed system.

For the middle box, its wiring to the top and bottom boxes is its context.

For the composite system in the dashed box, the context is just the bottom box. Note how the inputs and outputs on the context are precisely the inverse of those on the system at hand.

In a wiring diagram, a Markov blanket of a subdiagram is a context.

Question: Given a wiring diagram, can we assign statistical games to its decompositions, "as if" the subsystems were alive?

Dynamical semantics for statistical games

We will write $\mathbf{Sys}(Oy^I)$ for the category of dynamical systems suitable for filling the box Oy^I.

A **game with dynamical semantics** will be just like a statistical game, but instead of a fitness function, we will have a "dynamical semantics" function. Given a lens of type $(X, A) \nrightarrow (Y, B)$, the dynamical semantics map will have type $\mathbf{Sys}(By^Y) \to \mathbf{Sys}(AYy^{XB})$.
(Again, we think of it as a way to wire a system to its environment.)

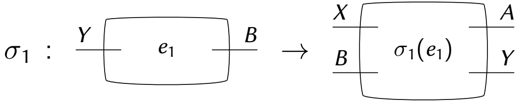
Composition of dynamical games is similar to statistical game composition:
lens composition, plus wiring.

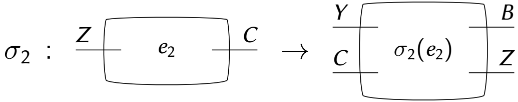
That is, given $(X, A) \nrightarrow (Y, B) \nrightarrow (Z, C)$, we need to obtain a semantics map $\mathbf{Sys}(Cy^Z) \to \mathbf{Sys}(AZy^{XC})$.

We will draw this on the next slides.

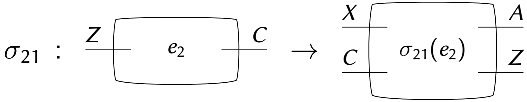
Dynamical semantics: composition law

We are given the semantics functions





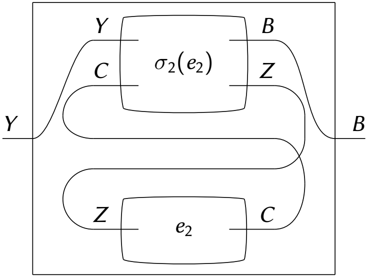
and seek to form a composite semantics function of the type



Dynamical semantics: composition law

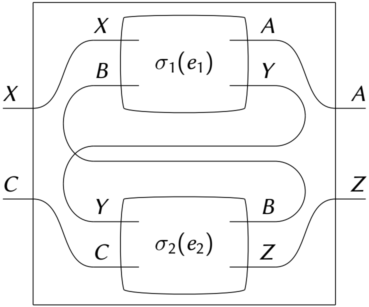
Given some $e_2 : \mathbf{Sys}(Cy^Z)$, we can use the given semantics function σ_2 to obtain a system $\sigma_2(e_2)$, and compose this with e_2 to obtain a system of the type By^Y, which we call e_1:

$e_1 :=$



With this $e_1 : \mathbf{Sys}(By^Y)$, we can use σ_1 to obtain a system $\sigma_1(e_1)$. Compose this with $\sigma_2(e_2)$ to get the composite system we seek:

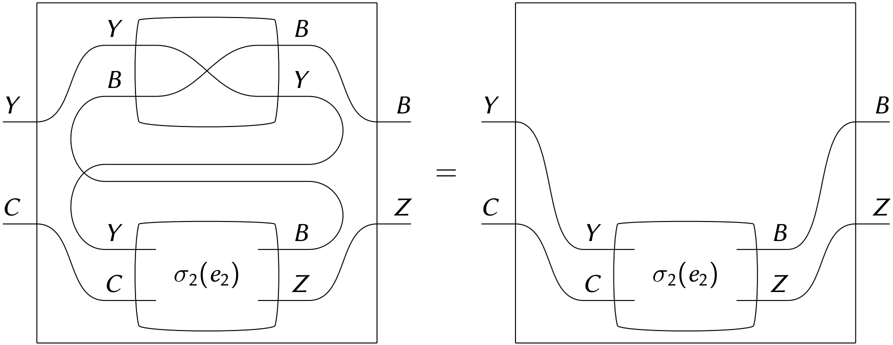
$\sigma_{21}(e_2) :=$



Notice how we start with the 'outermost' system first!
And notice how we have 'internal' (*e.g.* Ay^X) wiring above and 'external' wiring below (*e.g.* Zy^C).

Dynamical semantics: identities

We also need an ‘identity semantics’ to correspond to the identity lens:



This does the job, so we have a category!

Parameterized dynamical semantics

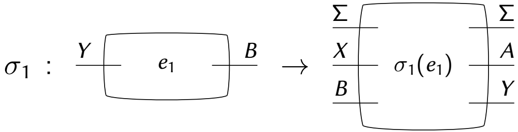
But of course we really need these things to be parameterized to be useful.
We can do that, again using the **Para** construction.

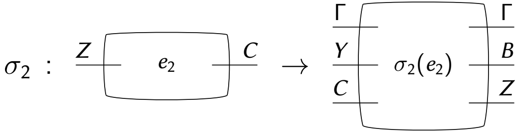
So, a parameterized dynamical game $(X, A) \nrightarrow (Y, B)$ is represented by a parameterized lens $(\Sigma \times X, \Sigma \times A) \nrightarrow (Y, B)$ along with a dynamical semantics map accordingly: $\mathbf{Sys}(By^Y) \to \mathbf{Sys}(\Sigma X B y^{\Sigma A Y})$.

I'll show the composition of these on the next slides.

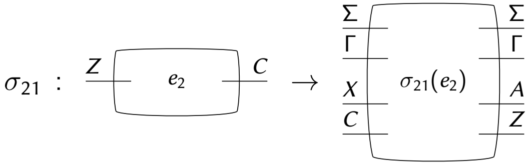
Parameterized dynamical semantics: composition law

We are given the semantics functions



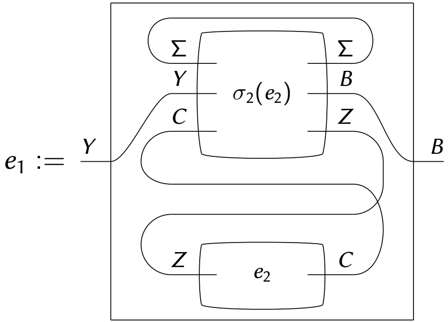


and seek to form a composite semantics function of the type

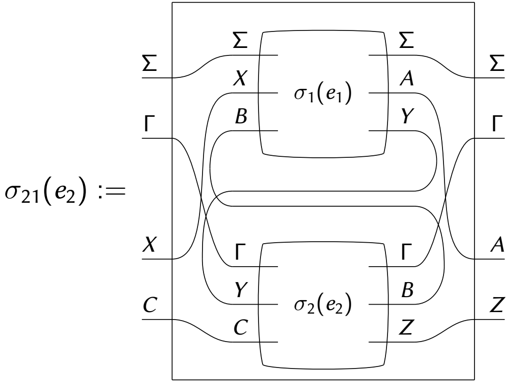


Parameterized dynamical semantics: composition law

Given some $e_2 : \mathbf{Sys}(Cy^Z)$, we can use the given semantics function σ_2 to obtain a system $\sigma_2(e_2)$, and compose this with e_2 to obtain a system of the type By^Y, which we call e_1:



With this $e_1 : \mathbf{Sys}(By^Y)$, we can use σ_1 to obtain a system $\sigma_1(e_1)$. Compose this with $\sigma_2(e_2)$ to get the composite system we seek:



Two active inference doctrines

Finally, with all that set-up, we are able to characterize formally various free-energy implementations as compositional active inference doctrines.

Specifically, we have (at least) two theorems characterizing such doctrines:

- the **'Laplace' doctrine**, from by the 'classical' free energy literature, with a structure much like biological neural circuits
- another from the machine learning literature: the **variational autoencoder doctrine**

And the compositional structure means that it should be possible to write a rigorous 'compiler' for such models, generalizing (for example) the DCM software.

But I'm not going to spell out the details today, as they are quite familiar!

Instead, I would like to point out some inelegancies and works-in-progress.

Some oddities

Some aspects of the formalism I presented are not quite satisfactory:

- There is a distinction between 'internal' and 'external' interfaces which is collapsed onto one type of box: from an external perspective, you don't observe my internal interfaces!
- As a system moves or evolves, its wiring might change: both internally (*e.g.* synaptic plasticity or pruning) and externally (*e.g.* by entering a conference call). But the wiring diagrams I presented are all static! (Shouldn't action affect the world??)
- As a system moves around its environment, the expected type of its observations might change (*e.g.* depending on task contingencies), and indeed there might be uncertainty about these types!
- Where is the system's internal representation of the external wiring structure?
  How can we see it in the formalism?

There seems to be quite an elegant answer, related both to nested systems (boxes may be filled with boxes) and structure-learning.

Overview

Dependent types: the logic of hierarchical interaction

All of those oddities can be resolved using **dependent types**:
typically when we have a set or a space, it just 'is'.

Dependent types allow objects to vary according to our position in some 'base space'.

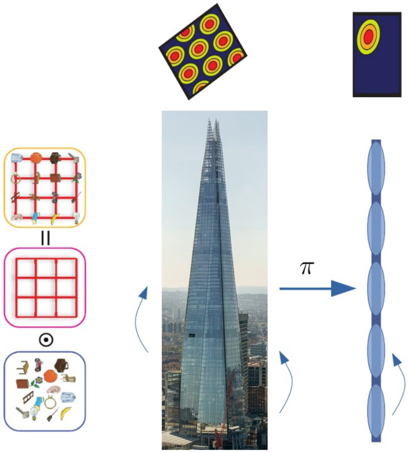
Some examples:

- A creature's 'internal interface' (my morphology) depends on its species.
- My 'wiring' might change depending on my current task or state. The type of expected perceptions or appropriate actions may vary similarly.
- A useful weather forecast might depend on whether I am on land or at sea.
- A parameterized game depends on its parameters.
- Bayesian inversion depends on the prior.

All of these are dependent types, but classical generative models cannot easily account for this.

Mathematically: dependent types are bundles

- A **bundle** is a collection of spaces $E(b)$ parameterized by some other space B.
- We can think of this as a **dependent sum**, writing $\mathbf{E} := \sum_{b:B} E(b)$.
- The elements of the total space E are 'dependent' pairs (b, e), with $b : B$ and $e : E(b)$.
- If $E(b) = E$ (*i.e.* not dependent), then we have a **trivial bundle**: $\mathbf{E} = \sum_b E(b) = \sum_b E = B \times E$.
- We identify the bundle with the projection $\pi : \mathbf{E} \to B$ onto the base space: $\pi(b, e) = b$.
- Many cognitive situations have this structure (consider the cognitive map of the Shard), but *it is not captured by classical generative models*!



Some current directions

- We can generalize the Stat indexed category defining Bayesian lenses in a way that interacts nicely with a topos and gives us convex sums of types and terms.
- Bayesian updating induces a kind of base changed in this generalized Stat category.
- We should think of the wiring as being encoded in the structure of the base space (*i.e.*, the parameterization).
- We can consider *action* and *intervention* as a change in this wiring: another kind of base change. (This has a nice type-theoretic interpretation!)

Thank you!