Dioptics: a common generalization of gradient-based learners and open games

David A. Dalrymple
@davidad

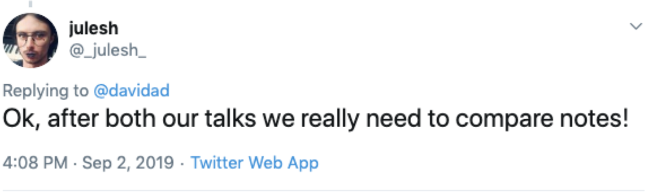
Protocol Labs

SYCO 5
Birmingham, UK
2019-09-05

About This Talk

- Clarifying connections between (a lot of) prior work
- Besides abstractions, main novelty: generalizing backpropagation and gradient descent to Lie groups and framed Riemannian manifolds
- Work in progress; dubious provenance

Haven’t I seen this talk already?



There is a *lot* of overlap with Jules’ talk earlier. A couple differences:

- I only deal with trivializable bundles, $TX \cong X \times X'$
- I’m aiming to cover more than just backpropagation

Notations

- Composition: $(f \mathbin{;} g)(x) \equiv g(f(x)) \equiv x \mathbin{;} f \mathbin{;} g$
- homs: $\mathcal{C}(A, B)$ means $\mathrm{hom}_{\mathcal{C}}(A, B)$. $[A, B]$ denotes the internal hom from A to B. $A \multimap B$ denotes the space of (literally) linear maps from A to B.
- Definitions:

$$\underbrace{\mathbf{eval}}_{\text{name}} \underbrace{{}_{X,Y}}_{\text{variables}} : \underbrace{((X \multimap Y) \otimes X) \to Y}_{\text{type}} := \underbrace{\langle f, x \rangle}_{\text{bindings}} \mapsto \underbrace{f(x)}_{\text{expression}}$$

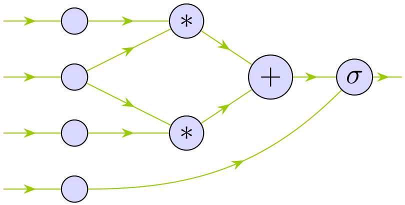
means the same as

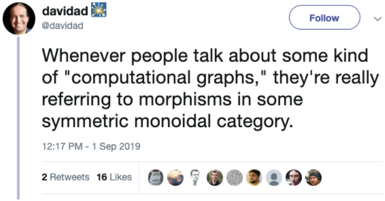
$$\mathbf{eval}_{X,Y} : ((X \multimap Y) \otimes X) \to Y$$
$$\mathbf{eval}_{X,Y} \langle f, x \rangle = f(x)$$

Section 1

Gradient-Based Learners

Machine Learning in 60 seconds

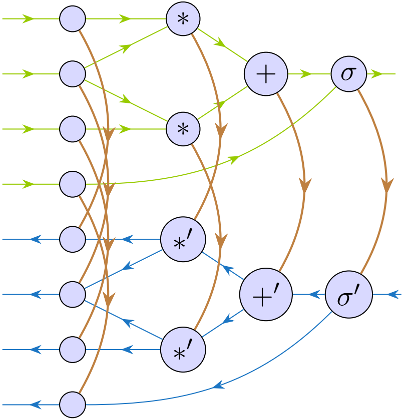




- A (supervised) *machine learning problem* is a function approximation problem.
- A pretty practical class of functions to approximate things with is neural nets.
- Deep learning *is*, in part, about composing layers. The deepness is (sequential) composition depth.
- Modern deep learning (e.g. TensorFlow, PyTorch) uses **computational graphs**.

← How much of modern deep learning can be understood from this perspective?

Backpropagation



- Forward pass computes $x \mapsto y$
- Backward pass computes $\frac{d-}{dx} \leftarrow \frac{d-}{dy}$
- Technically, the name "backpropagation" implies codomain $\mathbb{R}$. Else, reverse-mode automatic differentiation.

Two ideas about how "backpropagation is a functor":

"Simple Essence of Automatic Differentiation"

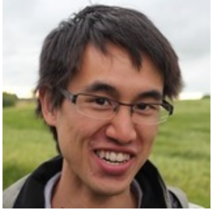
arXiv:1804.00746 [cs.PL]

Conal Elliott

"Backprop as Functor" (presented at SYCO 1!)

arXiv:1711.10455 [math.CT]

Brendan Fong, David Spivak, Rémy Tuyéras



How do these relate?

What's a Derivative?

2 What's a Derivative?

Since automatic differentiation (AD) has to do with computing derivatives, let's begin by considering what derivatives are. If your introductory calculus class was like mine, you learned that the derivative $f'\ x$ of a function $f :: \mathbb{R} \to \mathbb{R}$ at a point x (in the domain of f) is a *number*, defined as follows:

$$f'\ x = \lim_{\varepsilon \to 0} \frac{f\ (x+\varepsilon) - f\ x}{\varepsilon} \tag{1}$$

That is, $f'\ x$ tells us how fast f is scaling input changes at x.

How well does this definition hold up beyond functions of type $\mathbb{R} \to \mathbb{R}$? It will do fine with complex numbers ($\mathbb{C} \to \mathbb{C}$), where division is also defined. Extending to $\mathbb{R} \to \mathbb{R}^n$ also works if we interpret the ratio as dividing a vector (in $\mathbb{R}^n$) by a scalar in the usual way. When we extend to $\mathbb{R}^m \to \mathbb{R}^n$ (or even $\mathbb{R}^m \to \mathbb{R}$), however, this definition no longer makes sense, as it would rely on dividing *by* a vector $\varepsilon :: \mathbb{R}^m$.

This difficulty of differentiation with non-scalar domains is usually addressed with the notion of "partial derivatives" with respect to the m scalar components of the domain $\mathbb{R}^m$, often written "$\partial f / \partial x_j$" for $j \in \{1, \ldots, m\}$. When the codomain $\mathbb{R}^n$ is also non-scalar (i.e., $n > 1$), we have a *matrix* $\mathbf{J}$ (the *Jacobian*), with $\mathbf{J}_{ij} = \partial f_i / \partial x_j$ for $i \in \{1, \ldots, n\}$, where each f_i projects out the i^{th} scalar value from the result of f.

So far, we've seen that the derivative of a function could be a single number (for $\mathbb{R} \to \mathbb{R}$), or a vector (for $\mathbb{R} \to \mathbb{R}^n$), or a matrix (for $\mathbb{R}^m \to \mathbb{R}^n$). Moreover, each of these situations has an accompanying chain rule, which says how to differentiate the composition of two functions. Where the scalar chain rule involves multiplying two scalar derivatives, the vector chain rule involves "multiplying" two *matrices* $\mathbf{A}$ and $\mathbf{B}$ (the Jacobians), defined as follows:

$$(\mathbf{A} \cdot \mathbf{B})_{ij} = \sum_{k=1}^{m} \mathbf{A}_{ik} \cdot \mathbf{B}_{kj}$$

Since one can think of scalars as a special case of vectors, and scalar multiplication as a special case of matrix multiplication, perhaps we've reached the needed generality. When we turn our attention to higher derivatives (which are derivatives of derivatives), however, the situation gets more complicated, and we need yet higher-dimensional representations, with correspondingly more complex chain rules.

Fortunately, there is a single, elegant generalization of differentiation with a correspondingly simple chain rule. First, reword Definition 1 above as follows:

$$\lim_{\varepsilon \to 0} \frac{f\ (x+\varepsilon) - f\ x}{\varepsilon} - f'\ x = 0$$

Equivalently,

$$\lim_{\varepsilon \to 0} \frac{f\ (x+\varepsilon) - (f\ x + \varepsilon \cdot f'\ x)}{\varepsilon} = 0$$

Notice that $f'\ x$ is used to linearly transform ε. Next, generalize this condition to say that $f'\ x$ is a *linear map* such that

$$\lim_{\varepsilon \to 0} \frac{\| f\ (x+\varepsilon) - (f\ x + f'\ x\ \varepsilon) \|}{\| \varepsilon \|} = 0.$$

In other words, $f'\ x$ is a *local linear approximation* of f at x. When an $f'\ x$ satisfying this condition exists, it is indeed unique [Spivak, 1965, chapter 2].

The derivative of a function $f :: a \to b$ at some value in a is thus not a number, vector, matrix, or higher-dimensional variant, but rather a *linear map* (also called "linear transformation") from a to b, which we will write as "$a \multimap b$". The numbers, vectors, matrices, etc mentioned above are all different *representations* of linear maps; and the various forms of "multiplication" appearing in their associated chain rules are all implementations of linear map *composition* for those representations. Here, a and b must be vector spaces that share a common underlying field. Written as a Haskell-style type signature (but omitting vector space constraints),

$$\mathcal{D} :: (a \to b) \to (a \to (a \multimap b))$$

From the type of $\mathcal{D}$, it follows that differentiating twice has the following type:[1]

$$\mathcal{D}^2 = \mathcal{D} \circ \mathcal{D} :: (a \to b) \to (a \to (a \multimap a \multimap b))$$

The type $a \multimap a \multimap b$ is a linear map that yields a linear map, which is the curried form of a *bilinear* map. Likewise, differentiating k times yields a k-linear map curried $k-1$ times. For instance, the *Hessian* matrix H corresponds to the second derivative of a function $f :: \mathbb{R}^m \to \mathbb{R}$, having m rows and m columns (and satisfying the symmetry condition $H_{i,j} \equiv H_{j,i}$).

[1] As with "$\to$", we will take "$\multimap$" to associate rightward, so $u \multimap v \multimap w$ is equivalent to $u \multimap (v \multimap w)$.

3 Rules for Differentiation

3.1 Sequential Composition

With the shift to linear maps, there is one general chain rule, having a lovely form, namely that the derivative of a composition is a *composition* of the derivatives [Spivak, 1965, Theorem 2-2]:

Theorem 1 (compose/"chain" rule)

$$\mathcal{D}\ (g \circ f)\ a = \mathcal{D}\ g\ (f\ a) \circ \mathcal{D}\ f\ a$$

If $f :: a \to b$ and $g :: b \to c$, then $\mathcal{D}\ f\ a :: a \multimap b$, and $\mathcal{D}\ g\ (f\ a) :: b \multimap c$, so both sides of this equation have type $a \multimap c$.[2]

Strictly speaking, Theorem 1 is not a compositional recipe for differentiating sequential compositions, i.e., it is *not* the case $\mathcal{D}\ (g \circ f)$ can be constructed solely from $\mathcal{D}\ g$ and $\mathcal{D}\ f$. Instead, it also needs f itself. Fortunately, there is a simple way to restore compositionality. Instead of constructing just the derivative of a function f, suppose we *augment* f with its derivative:

$$\mathcal{D}_0^+ :: (a \to b) \to ((a \to b) \times (a \to (a \multimap b))) \quad \text{-- first try}$$
$$\mathcal{D}_0^+\ f = (f, \mathcal{D}\ f)$$

As desired, this altered specification is compositional:

$$\begin{aligned} & \mathcal{D}_0^+\ (g \circ f) \\ = {} & (g \circ f, \mathcal{D}\ (g \circ f)) && \text{-- definition of } \mathcal{D}_0^+ \\ = {} & (g \circ f, \lambda a \to \mathcal{D}\ g\ (f\ a) \circ \mathcal{D}\ f\ a) && \text{-- Theorem 1} \end{aligned}$$

Note that $\mathcal{D}_0^+\ (g \circ f)$ is assembled entirely from components of $\mathcal{D}_0^+\ g$ and $\mathcal{D}_0^+\ f$, which is to say from g, $\mathcal{D}\ g$, f, and $\mathcal{D}\ f$. Writing out $g \circ f$ as $\lambda a \to g\ (f\ a)$ underscores that the two parts of $\mathcal{D}_0^+\ (g \circ f)\ a$ both involve $f\ a$. Computing these parts independently thus requires redundant work. Moreover, the chain rule itself requires applying a function and its derivative (namely f and $\mathcal{D}\ f$) to the same a. Since the chain rule gets applied recursively to nested compositions, this redundant work multiplies greatly, resulting in an impractically expensive algorithm.

Fortunately, this efficiency problem is easily fixed. Instead of pairing f and $\mathcal{D}\ f$, *combine* them:[3]

$$\mathcal{D}^+ :: (a \to b) \to (a \to b \times (a \multimap b)) \quad \text{-- better!}$$
$$\mathcal{D}^+\ f\ a = (f\ a, \mathcal{D}\ f\ a)$$

Combining f and $\mathcal{D}\ f$ into a single function in this way enables us to eliminate the redundant computation of $f\ a$ in $\mathcal{D}^+\ (g \circ f)\ a$, as follows:

Corollary 1.1 (Proved in Appendix C.1) *$\mathcal{D}^+$ is (efficiently) compositional with respect to $(\circ)$. Specifically,*

- Elliott constructs a "derivative" functor D^+

For $X, Y : \mathbf{Euc}$, $f : X \to Y$, let

$$Df : (\overbrace{X}^{x} \to \overbrace{(X \multimap Y)}^{f'(x) := g}) := x \mapsto \text{the unique linear } g \text{ s.t.}$$

$$\lim_{\varepsilon \to 0} \frac{\left\| f(x+\varepsilon) - \left[f(x) + f'(x)(\varepsilon) \right] \right\|}{\|\varepsilon\|} = 0$$

Chain rule:

$$D(f \mathbin{;} g)(x) = Df(x) \mathbin{;} Dg(f(x))$$

Problem – not functorial: depends on un-D'd f.

Let $D^+ f : X \to (Y \times (X \multimap Y)) := x \mapsto \langle f(x), Df(x) \rangle$

Proposition (Elliott). D^+ is a symmetric monoidal functor from **Euc** into a category with objects of **Euc** and morphisms of type $X \to_{\mathbf{Euc}} (Y \times (X \multimap Y))$.

What do we really need to assume?

We can work in any category $\mathcal{E}$ which...

- is cartesian closed and locally cartesian closed
- has a product-preserving endofunctor T (given a space $X : \mathcal{E}$, TX is interpreted as its tangent bundle)
- has a "base point" natural transformation $p : \forall X.\, TX \to X$ (that is, $p : T \Rightarrow \mathsf{id}_{\mathcal{E}}$).
- has a semiadditive subcategory $\mathcal{E}\mathbf{Vect}$ of "vector-like spaces" enriched in $\mathcal{E}$
- has a subcategory $\mathcal{E}\mathbf{Triv}$ of "trivializable spaces" s.t. for all $X : \mathcal{E}\mathbf{Triv}$, there is some $X' : \mathcal{E}\mathbf{Vect}$ satisfying the isomorphism (of bundles over X) $TX \cong X \times X'$.
  - Observation: $TX \cong X \times X'$ looks like a constant-complement lens $TX \rightsquigarrow X$
- satisfies one last hard-to-state assumption about "linearity of derivatives"

What's a Derivative, Again?

- If $X, Y : \mathcal{C}\mathbf{Triv}$, then $T(f : X \to Y) : \; TX \to TY \cong X \times X' \to Y \times Y'$
- By naturality of base-point projection $p : T\text{–} \Rightarrow \text{–}$, we have $Tf\langle x, \cdot\rangle = \langle f(x), \cdot\rangle$.
- Therefore $Tf\langle x, x'\rangle = \langle f(x), \pi_2 Tf\langle x, x'\rangle\rangle$.
- So we can define $T^+ f : X \to (Y \times (X' \to Y')) := x \mapsto \langle f(x), \lambda x'.\pi_2 Tf\langle x, x'\rangle\rangle$.
- Our last assumption is that $T^+(f)(x)$ is, in fact, a linear map $X' \multimap Y'$.
- Then $T^+ f : X \to (Y \times (X' \multimap Y'))$, just like Elliott's D^+.

All That and a Pony

Two ways to instantiate those assumptions:

1. $\mathcal{E}$ can be the microlinear spaces of a well-adapted model of synthetic differential geometry, like the Dubuc/Cahiers topos
   - Here, TX is representable as $[D, X]$ where D is the infinitesimal interval
2. $\mathcal{E}$ can be the category of diffeological spaces due to Souriau

In either case, $\mathcal{E}$**Triv** includes all manifolds with trivializable tangent bundles, e.g.

- open subsets of Euclidean spaces
- affine spaces
- Lie groups
- framed manifolds

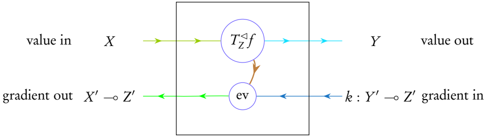
Backpropagation (Reverse-mode automatic differentiation)

$$\text{(forward-mode)} \qquad T^{+}f : X \to \left(Y \times (X' \multimap Y')\right)$$

$$T_Z^{\triangleleft}(f : X \to Y) : X \to \left(Y \times \left[\overbrace{(Y' \multimap Z')}^{k}, (\overbrace{X'}^{d} \multimap Z')\right]\right) :=$$

$$x \mapsto \left\langle f(x), k \mapsto d \mapsto \langle x, d\rangle \mathbin{;} \gamma_X^{-1} \mathbin{;} Tf \mathbin{;} \gamma_Y \mathbin{;} \pi_2 \mathbin{;} k \right\rangle$$

where $\gamma_X : TX \cong X \times X'$, $\gamma_Y : TY \cong Y \times Y'$.



$$T_Z^{\triangleleft} : \mathcal{E}\mathbf{Triv} \to \mathbf{Optic}_{\mathcal{E}} := X \mapsto \left(X, X' \multimap Z'\right)$$

"Categories of Optics"

Definition [Ril18]. In any symmetric monoidal category $\mathcal{C}$,

$$\mathbf{Optic}_{\mathcal{C}}\left((X, X^-), (Y, Y^-)\right) := \int^{M:\mathcal{C}} \mathcal{C}(X, M \otimes Y) \times \mathcal{C}(M \otimes Y^-, X^-)$$

Theorem (Riley). $\mathbf{Optic}_{\mathcal{C}}$ is a symmetric monoidal category with objects of $\mathcal{C} \times \mathcal{C}^{\mathrm{op}}$.

If $\mathcal{C}$ is cartesian, $\mathbf{Optic}_{\mathcal{C}}$ is equivalent to

$$\mathbf{Lens}_{\mathcal{C}}\left((X, X^-), (Y, Y^-)\right) := \underbrace{\mathcal{C}(X, Y)}_{\text{get}} \times \underbrace{\mathcal{C}(X \times Y^-, X^-)}_{\text{put}}$$

If $\mathcal{C}$ is symmetric monoidal closed, $\mathbf{Optic}_{\mathcal{C}}$ is equivalent to

$$\mathbf{CurriedLens}_{\mathcal{C}}\left((X, X^-), (Y, Y^-)\right) := X \to \left(Y \otimes [Y^-, X^-]\right)$$

Theorem (de Paiva). If $\mathcal{C}$ is cartesian closed and locally cartesian closed, $\mathbf{Optic}_{\mathcal{C}}$ is a symmetric monoidal closed category, with internal hom defined as

$$(X, X^-) \rightsquigarrow_{\mathcal{C}} (Y, Y^-) = \left(\left[X, Y \times [Y^-, X^-]_{\mathcal{C}}\right]_{\mathcal{C}}, X \times Y^-\right)$$

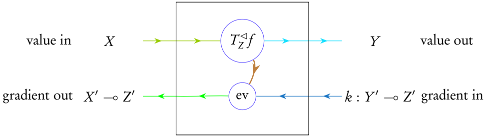
Backpropagation (Reverse-mode automatic differentiation)

(forward-mode) $T^{+} f : X \to \left(Y \times (X' \multimap Y')\right)$

$$T^{\triangleleft}_Z(f : X \to Y) : X \to \left(Y \times \left[\overbrace{(Y' \multimap Z')}^{k}, (\overbrace{X'}^{d} \multimap Z')\right]\right) :=$$

$$x \mapsto \left\langle f(x), k \mapsto d \mapsto \langle x, d \rangle \fatsemi \gamma_X^{-1} \fatsemi Tf \fatsemi \gamma_Y \fatsemi \pi_2 \fatsemi k \right\rangle$$

where $\gamma_X : TX \cong X \times X'$, $\gamma_Y : TY \cong Y \times Y'$.



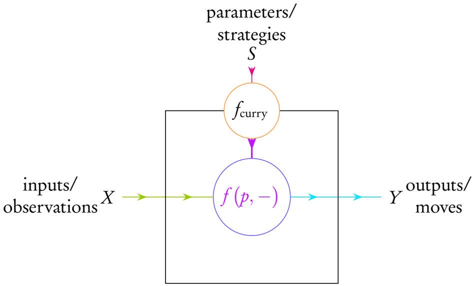
$$T^{\triangleleft}_Z : \mathcal{E}\mathbf{Triv} \to \mathbf{Optic}_{\mathcal{E}} := X \mapsto \left(X, X' \multimap Z'\right)$$

$$\mathbf{Optic}_{\mathcal{E}}\left((X, X^-), (Y, Y^-)\right) \cong X \to \left(Y \times [Y^-, X^-]\right) \qquad X^- := (X' \multimap Z')$$

"Backprop as Functor": **Learn**

Definition [FST17]. Given X, Y : **Set**, a *learner* ℓ from $X \to Y$ is defined by:

$S_\ell : \mathbf{Set}$	the *parameter space*
$I_\ell : S_\ell \times X \to Y$	the *implementation function*
$r_\ell : S_\ell \times X \times Y \to X$	the *request function*
$U_\ell : S_\ell \times X \times Y \to S_\ell$	the *update function*



"Backprop as Functor": **Learn**

Definition [FST17]. Given $X, Y : \mathbf{Set}$, a *learner* ℓ from $X \to Y$ is defined by:

$S_\ell : \mathbf{Set}$	the *parameter space*
$I_\ell : S_\ell \times X \to Y$	the *implementation function*
$r_\ell : S_\ell \times X \times Y \to X$	the *request function*
$U_\ell : S_\ell \times X \times Y \to S_\ell$	the *update function*

Equivalently, a learner $\ell : X \to Y$ is exactly

- a family of lenses, i.e. a set S_ℓ and for each $s : S_\ell$ a lens $\ell_s : (X, X) \rightsquigarrow_{\mathbf{Set}} (Y, Y)$
- $U_\ell : S_\ell \times X \times Y \to S_\ell$

Observation: Also equivalently, a learner $\ell : X \to Y$ is exactly a set S_ℓ and a lens $(S_\ell, S_\ell) \rightsquigarrow_{\mathbf{Set}} \big((X, X) \rightsquigarrow_{\mathbf{Set}} (Y, Y)\big)$, or $\big((S_\ell, S_\ell) \otimes (X, X)\big) \rightsquigarrow_{\mathbf{Set}} (Y, Y)$.

Proposition [FST17]. There is a symmetric monoidal category **Learn** whose objects are sets and whose morphisms are equivalence classes of learners.

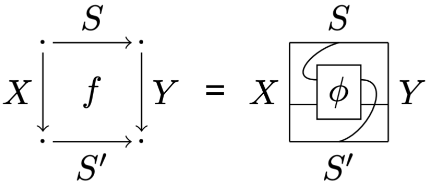
Section 2

Dioptics

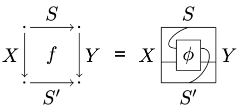
$\mathbf{Dbl}(\mathcal{C})$

Definition (Sprunger, Katsumata). Given a cartesian category $\mathcal{C}$, the double category $\mathbf{Dbl}(\mathcal{C})$ has

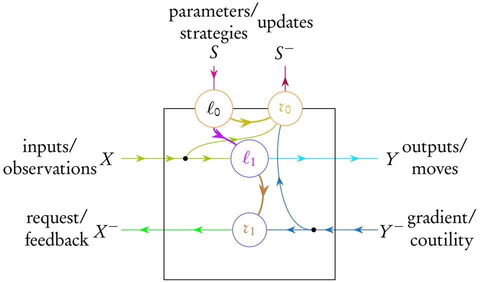
- One 0-cell, written as $\cdot$
- Horizontal and vertical 1-cells both given by objects of $\mathcal{C}$, composed with $\times_{\mathcal{C}}$, with identity given by the terminal object in $\mathcal{C}$.
- A 2-cell with boundary X, Y, S, S' is given by a morphism $\mathcal{C}(S \times X, S' \times Y)$.



$\mathbf{Dbl}(\mathbf{Optic}_{\mathcal{C}})$



2-cells/tiles of $\mathbf{Dbl}(\mathbf{Optic}_{\mathcal{C}})$ are morphisms $\mathbf{Optic}_{\mathcal{C}}\Big((S,S^-)\otimes(X,X^-),(S',S'^-)\otimes(Y,Y^-)\Big)$.
If we look only at tiles with trivial vertical codomain (monoidal unit), we get $\mathbf{Optic}_{\mathcal{C}}\Big((S\times X,S^-\times X^-),(Y,Y^-)\Big)$, exactly the desired structure:



Quotienting by equivalence of parameter space S

If composition of parameterized morphisms involves Cartesian-product-ing their parameter spaces, then associativity of composition does not (directly, strictly) hold.

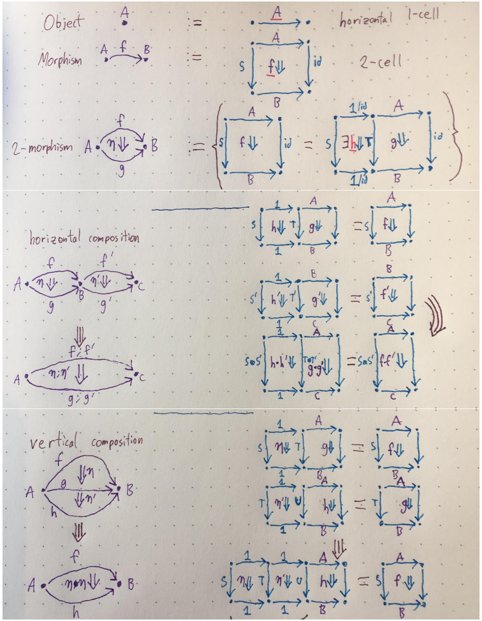
Ways to solve this:

- Make the parameter space into an "opaque" or "existential" type:
  1. Explicit meta-theoretic quotient (as for **Learn**, **Para**, **Game**)
  2. Bind it with a coend (as for **Optic**) — this is what I do for now with $\mathbf{Dioptic}_{F,G}$
- Give up strict associativity; define a bicategory instead. (2-morphisms are reparameterizations.)
- Construct the double category $\mathbf{Dbl}(\mathcal{C})$, using monoidal strictification. Question: how do we recover a symmetric monoidal category?

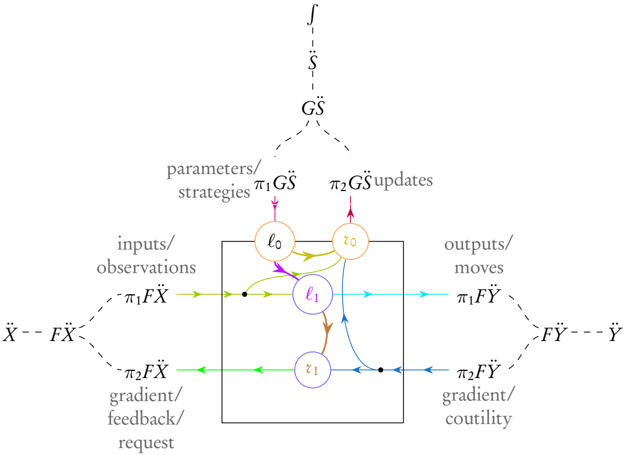
Recovering a symmetric monoidal category from $\mathbf{Dbl}(\mathbf{Optic}_{\mathcal{C}})$

Proposed approach: $\mathbf{Cat}(\mathbf{Cat}) \xrightarrow{?} \mathbf{SymMon2Cat} \xrightarrow{\text{forgetful}} \mathbf{SymMonCat}$

Recovering a symmetric monoidal category from **Dbl**(**Optic**$_{\mathcal{C}}$)



$\mathbf{Dioptic}_{F,G}$



Construction takes as input:

- $\mathcal{C}$ a cartesian closed, locally cartesian closed category
- $\mathcal{S}, \mathcal{T}$ symmetric monoidal categories
- $F : \mathcal{T} \to \mathbf{Optic}_{\mathcal{C}}$, $G : \mathcal{S} \to \mathbf{Optic}_{\mathcal{C}}$ are symmetric oplax monoidal functors
  - Canonical embedding $(\mathcal{C} \times \mathcal{C}^{op}) \hookrightarrow \mathbf{Optic}_{\mathcal{C}}$ can be useful
- **Conjecture**: If F is strong symmetric monoidal, $\mathbf{Dioptic}_{F,G}$ is symmetric monoidal.

$$\mathbf{Dioptic}_{F,G} : \mathcal{T}^{op} \times \mathcal{T} \to \mathbf{Set} := (\ddot{X}, \ddot{Y}) \mapsto \int^{\ddot{S}:\mathcal{S}} \mathbf{Optic}_{\mathcal{C}}(G\ddot{S},\ F\ddot{X} \rightsquigarrow_{\mathcal{C}} F\ddot{Y}) = \int^{\ddot{S}:\mathcal{S}} \mathbf{Optic}_{\mathcal{C}}(G\ddot{S} \times F\ddot{X}, F\ddot{Y})$$

Para as $\mathbf{Dioptic}_{\mathsf{Fwd},\mathsf{Fwd}}$

Let $\mathcal{E}$ be the Dubuc topos or the category of diffeological spaces; then let

$$\mathsf{Fwd} : \mathbf{Euc} \to \mathcal{E} \times \mathcal{E}^{\mathrm{op}} := X \mapsto (X, 1)$$

Then we have

$$\begin{aligned}\mathbf{Dioptic}_{\mathsf{Fwd},\mathsf{Fwd}}(X,Y) &= \int^{S:\mathbf{Euc}} \mathbf{Optic}_{\mathcal{C}}\left((S,1),\ (X,1) \rightsquigarrow_{\mathcal{C}} (Y,1)\right) \\ &\cong \int^{S:\mathbf{Euc}} \mathbf{Lens}_{\mathcal{C}}((S,1) \otimes (X,1), (Y,1)) \\ &\cong \int^{S:\mathbf{Euc}} \mathcal{C}(S \times X, Y)\end{aligned}$$

$\mathbf{GradLearn} := \mathbf{Dioptic}_{T^{\triangleleft}_{\mathbb{R}}, T^{\triangleleft}_{\mathbb{R}}}$

Let $\mathcal{E}$ be the Dubuc topos or the category of diffeological spaces, with $\mathcal{E}\mathbf{Triv}$ the subcategory with trivializable bundles ($TX \cong X \times X'$). Then

$$T^{\triangleleft}_{\mathbb{R}} : \mathcal{E}\mathbf{Triv} \to \mathbf{Optic}_{\mathcal{E}} = X \mapsto \left(X,\, X' \multimap \mathbb{R}\right)$$

We have

$$\begin{aligned}
\mathbf{Dioptic}_{T^{\triangleleft}_{\mathbb{R}}, T^{\triangleleft}_{\mathbb{R}}}(X, Y) &= \int^{S:\mathcal{E}\mathbf{Triv}} \mathbf{Optic}_{\mathcal{E}}\left((S, S' \multimap \mathbb{R}),\ (X, X' \multimap \mathbb{R}) \rightsquigarrow_{\mathcal{E}} (Y, Y' \multimap \mathbb{R})\right) \\
&\cong \int^{S:\mathcal{E}\mathbf{Triv}} \mathbf{Lens}_{\mathcal{E}}\left((S, S'^{*}) \otimes (X, X'^{*}), (Y, Y'^{*})\right) \\
&\cong \int^{S:\mathcal{E}\mathbf{Triv}} \mathbf{Lens}_{\mathcal{E}}\left((S \times X, S'^{*} \times X'^{*}), (Y, Y'^{*})\right) \\
&\cong \int^{S:\mathcal{E}\mathbf{Triv}} \mathcal{E}(S \times X, Y) \times \mathcal{E}\left(S \times X \times Y'^{*}, S'^{*} \times X'^{*}\right)
\end{aligned}$$

$T^{\triangleleft}_{Z}$ *is* strong symmetric monoidal: $(X \times Y)' \multimap Z \cong (X' \multimap Z) \times (Y' \multimap Z)$, due to product-preservation of T and semiadditivity of $\mathcal{E}\mathbf{Vect}$.

Learn as $\mathbf{Dioptic}_{\Delta^{\rightleftarrows}_{\mathbf{Set}},\Delta^{\rightleftarrows}_{\mathbf{Set}}}$

Let

$$\Delta^{\rightleftarrows}_{\mathbf{Set}} : \mathrm{Core}(\mathbf{Set}) \to \mathbf{Optic}_{\mathbf{Set}} := X \mapsto (X, X)$$

Then we have

$$\begin{aligned}\mathbf{Dioptic}_{\Delta^{\rightleftarrows}_{\mathbf{Set}},\Delta^{\rightleftarrows}_{\mathbf{Set}}}(X,Y) &= \int^{S:\mathbf{Set}} \mathbf{Optic}_{\mathbf{Set}}\left((S,S),\ (X,X) \rightsquigarrow_{\mathbf{Set}} (Y,Y)\right) \\ &= \int^{S:\mathbf{Set}} \mathbf{Set}\left(S \times X,\ Y\right) \times \mathbf{Set}\left(S \times X \times Y, S \times X\right) \\ &\cong \mathbf{Learn}(X,Y)\end{aligned}$$

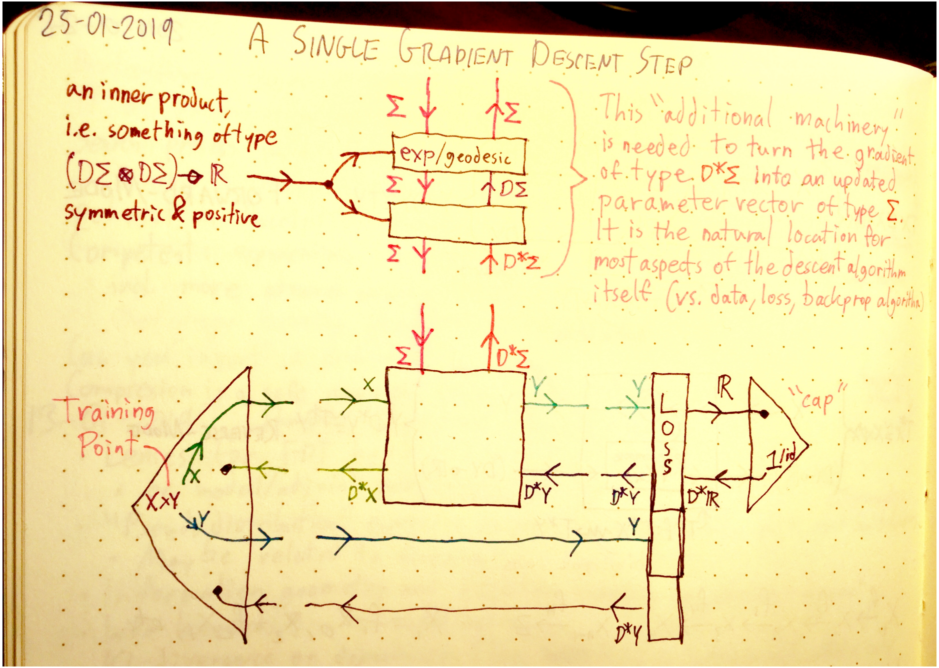
Gradient *descent*

Earlier, I said instead of computing an unknown Y' from an unknown X', we want to compute $X' \multimap \mathbb{R}$ (that is, X'^*) from $Y' \multimap \mathbb{R}$ (that is, Y'^*).
Actually, we want to compute a new value of type S! Fortunately, we have a covector $c : S' \multimap \mathbb{R}$ to work with.
Steps to compute a new value for S, assuming S is equipped with a Riemannian structure (a symmetric, nonnegative, nondegenerate bilinear form $\mathrm{g} : S' \times S' \multimap \mathbb{R}$:

- There exists a unique vector v such that $c = \lambda d.\mathrm{g}(v, d)$.
- Scale the vector by an arbitrary learning rate $\eta : \mathbb{R}$ (and probably -1, if you're minimizing a loss).
  - Handling hyperparameters like η internal to the theory is very WIP, but should work.
- Using the Riemannian structure, compute the unique torsion-free Levi-Civita connection for parallel transport.
- Apply some appropriate theorem for the existence and uniqueness of differential equation solutions to integrate the tangent vector $-\eta v$ along a geodesic $a(t)$ starting from the current parameter state $s_i : S$.
- Let $s_{i+1} := a(1)$.

By vertically composing all that machinery on top of a gradient-based learner of type $(S, S'^*) \rightsquigarrow (X, X'^*) \rightsquigarrow (Y, Y'^*)$, we obtain a dioptic $(S, S) \rightsquigarrow (X, X'^*) \rightsquigarrow (Y, Y'^*)$.

Gradient *descent*



Getting to **Learn**

- We now have $(S, S) \rightsquigarrow (X, X'^*) \rightsquigarrow (Y, Y'^*)$, but in **Learn** we have dioptics of type $(S, S) \rightsquigarrow (X, X) \rightsquigarrow (Y, Y)$.
- Getting to learn requires a bit of a hack—we need to package up the loss function and the gradient of its inverse into every morphism (tile). This introduces a lot of unnecessary operations, and the same is true for [FST17]'s original functor from **Para** $\to$ **Learn**.

    Given a positive number $\eta : \mathbb{R}$ *(the step size) and a differentiable function* $e(x, y) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ *(the loss function) such that* $\frac{\partial e}{\partial x}(z, -) : \mathbb{R} \to \mathbb{R}$ *is invertible* $\forall z : \mathbb{R}$*, we can define a faithful, injective-on-objects, symmetric monoidal functor* $L_{e,\eta} : \mathbf{Para} \to \mathbf{Learn}$ *that sends each parametrised function* $f : S \times X \to Y$ *to the learner* (S, f, U_f, r_f) *defined by*

    $$U_f(s, x, y) := s - \eta \nabla_s \sum_j e\Big(f(s, x)_j, y_j\Big)$$

    $$r_f(s, x, y) := f_x\left(\nabla_x \sum_j e\Big(f(s, x)_j, y_j\Big)\right)$$

    where f_x *is componentwise application of the inverse to* $\frac{\partial e}{\partial x}(x_i, -)$ *for each* i*.*

- The same trick works in a dioptic context, but only for bona fide Euclidean spaces.

Section 3

Open Games

"Compositional Game Theory": **Game**

Definition [GHWZ18]. Given $X, X^-, Y, Y^- : \mathbf{Set}$, an *open game* $\mathcal{G}$ from $(X, X^-) \to (Y, Y^-)$ is defined by:

$$S_{\mathcal{G}} : \mathbf{Set} \qquad \text{the } \textit{strategy profile space}$$

$$P_{\mathcal{G}} : S_{\mathcal{G}} \times X \to Y \qquad \text{the } \textit{play function}$$

$$C_{\mathcal{G}} : S_{\mathcal{G}} \times X \times Y^- \to X^- \qquad \text{the } \textit{coplay function}$$

$$E_{\mathcal{G}} : S_{\mathcal{G}} \times X \times (Y \to Y^-) \to S_{\mathcal{G}} \to \mathbf{2} \qquad \text{the } \textit{equilibrium function}$$

We define these auxiliary functors, with codomain $\mathbf{Set} \times \mathbf{Set}^{\mathrm{op}}$:

$$E^+ := S \mapsto (S, \mathbf{2}), \quad C^+ := (X, X^-) \mapsto (X, [X, X^-])$$

$$B^+ := E^+ \mathbin{;} C^+ = S \mapsto (S, [S, \mathbf{2}])$$

The oplaxator of E^+ is defined using conjunction:

$$E^+.\Delta_{S,T} : \overbrace{E^+(S \times T)}^{(S \times T, \mathbf{2})} \to_{\mathbf{Set} \times \mathbf{Set}^{\mathrm{op}}} \overbrace{E^+ S \otimes E^+ T}^{(S \times T, \mathbf{2} \times \mathbf{2})} := \big((s,t) \mapsto (s,t), (a \wedge b) \leftarrow\!\shortmid (a,b)\big)$$

Conjecture. **Game** has a faithful, identity-on-objects functor into $\mathbf{Dioptic}_{C^+, B^+}$.

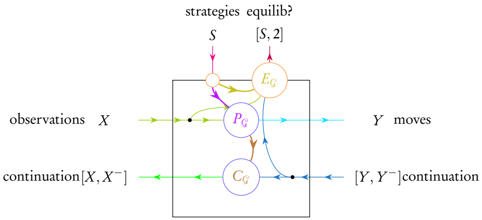
Open Games as Dioptics

$$\begin{aligned}
&\mathbf{Dioptic}_{C^+,B^+}\left((X,X^-),(Y,Y^-)\right)\\
&= \int^{\ddot{S}:\mathbf{Set}} \mathbf{Optic}_{\mathbf{Set}}\left(B^+\ddot{S},\, C^+(X,X^-) \rightsquigarrow_{\mathbf{Set}} C^+(Y,Y^-)\right)\\
&= \int^{S:\mathbf{Set}} \mathbf{Optic}_{\mathbf{Set}}\left((S, S\to \mathbf{2}),\, \left(X, X\to X^-\right) \rightsquigarrow_{\mathbf{Set}} \left(Y, Y\to Y^-\right)\right)\\
&\cong \int^{S:\mathbf{Set}} \mathbf{Optic}_{\mathbf{Set}}\left((S, S\to \mathbf{2}),\, \left(X \to \left(Y \times \left((Y\to Y^-)\to(X\to X^-)\right)\right), X\times(Y\to Y^-)\right)\right)\\
&\cong \int^{S:\mathbf{Set}} \mathbf{Set}\left(S, X \to \left(Y \times \left((Y\to Y^-)\to(X\to X^-)\right)\right)\right) \times \mathbf{Set}\left(S\times X\times(Y\to Y^-), (S\to\mathbf{2})\right)\\
&\cong \int^{S:\mathbf{Set}} \mathbf{Set}\left(S\times X, Y\right) \times \mathbf{Set}\left(S\times X\times(Y\to Y^-), (X\to X^-)\right) \times \mathbf{Set}\left(X\times(Y\to Y^-), (S\times S\to\mathbf{2})\right)\\
&\leftarrow \coprod_{S:\mathbf{Set}} \mathbf{Set}\left(S\times X, Y\right) \times \mathbf{Set}\left(S\times X\times(Y\to Y^-), (X\to X^-)\right) \times \mathbf{Set}\left(X\times(Y\to Y^-), (S\times S\to\mathbf{2})\right)\\
&\overset{\phi}{\leftarrow} \coprod_{S:\mathbf{Set}} \overbrace{\mathbf{Set}\left(S\times X, Y\right)}^{\text{play function } P} \times \overbrace{\mathbf{Set}\left(S\times X\times Y^-, X^-\right)}^{\text{coplay function } C} \times \overbrace{\mathbf{Set}\left(X\times(Y\to Y^-), (S\times S)\to\mathbf{2}\right)}^{\text{best-response function } B}\\
&\hookleftarrow \mathbf{Game}\left((X,X^-),(Y,Y^-)\right)
\end{aligned}$$

where

$$\phi := (S,P,C,B) \mapsto \left(S, P, (s,\not{x},k) \mapsto x \mapsto C\left(s,x,k\left(P(s,x)\right)\right), B\right) \quad \phi^{\leftarrow} := (S,P,K,B) \mapsto \left(S, P, \left(s,x,Y^-\right) \mapsto K\left(s,x,(y\mapsto Y^-)\right)(x), B\right)$$

Dishonest morphisms



- Nothing I've done says the continuation that's output to the left has to be true.
- The sequential composition rule holds up, but a "dishonest" tile can corrupt a whole diagram out of the subcategory that corresponds to **Game**.

But it's not even monoidal

- Unfortunately $\mathbf{Dioptic}_{C^+,B^+}$ fails to be monoidal, because C^+ is not a strong monoidal functor (merely bilax monoidal and Frobenius monoidal)
- There are natural transformations

$$\mu_{(X,X^-),(Y,Y^-)} : \overbrace{\left(X, [X, X^-]\right)}^{C^+((X,X^-))} \otimes \overbrace{\left(Y, [Y, Y^-]\right)}^{C^+((Y,Y^-))} \longrightarrow \overbrace{\left(X \times Y, [X \times Y, X^- \times Y^-]\right)}^{C^+((X,X^-)\otimes(Y,Y^-))}$$

and

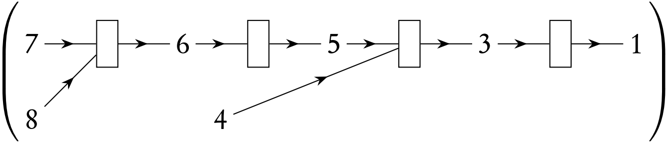
$$\Delta_{(X,X^-),(Y,Y^-)} : \overbrace{\left(X, [X, X^-]\right)}^{C^+((X,X^-))} \otimes \overbrace{\left(Y, [Y, Y^-]\right)}^{C^+((Y,Y^-))} \longleftarrow \overbrace{\left(X \times Y, [X \times Y, X^- \times Y^-]\right)}^{C^+((X,X^-)\otimes(Y,Y^-))}$$

but they are *not* inverses

- The backwards part (put) of μ, of type $X \times Y \times [X \times Y, X^- \times Y^-] \to [X, X^-] \times [Y, Y^-]$ is lossy
- As a result, in $\mathbf{Dioptic}_{C^+,B^+}$, $\mathrm{id}_X \otimes \mathrm{id}_Y \ncong \mathrm{id}_{X \otimes Y}$

Is there a different way?

- Problem: passing $X \to X^-$ to f and $Y \to Y^-$ to g loses information about the joint dependency $X \times Y \to X^- \times Y^-$.
- Perhaps continuations can be upgraded to some kind of "nominal diagrams" that express dependencies on all uncles, and from which joint information can be recovered.



- Decorated cospans?

Future work

- Actually proving stuff
- Working out the quotienting machinery
  - Blue-sky idea: if it works, can it replace coend in the definition of **Optic** itself?

$$\mathbf{Optic}_{\mathcal{C}}\left((X, X^-), (Y, Y^-)\right) := \int^{M:\mathcal{C}} \mathcal{C}(X, M \otimes Y) \times \mathcal{C}(M \otimes Y^-, X^-)$$

- Proving stuff *in Coq*
- Generalizing to nontrivializable bundles (merge with Jules')
- Trying more computable base fields than $\mathbb{R}$

Characterizing truthfulness

- Would be nice to axiomatize which dioptics are in the image of the faithful functor $\mathbf{Game} \hookrightarrow \mathbf{Dioptic}_{C+,B+}$
- Naïvely, might hope that "truthful" $\sim$ "lawful", but there seems to be no applicable definition of "lawful"

Synthesizing functors between categories of dioptics

- The alleged functors

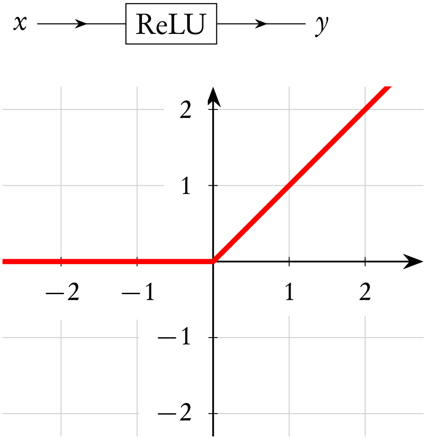
$$T^* : \mathbf{Para} \cong \mathbf{Dioptic}_{\mathsf{Fwd},\mathsf{Fwd}} \to \mathbf{Dioptic}_{T^{\triangleleft}_{\mathbb{R}},T^{\triangleleft}_{\mathbb{R}}} =: \mathbf{GradLearn}$$

$$L^*_{e,\eta} : \mathbf{GradLearn} := \mathbf{Dioptic}_{T^{\triangleleft}_{\mathbb{R}},T^{\triangleleft}_{\mathbb{R}}} \to \mathbf{Dioptic}_{\Delta^{\rightleftarrows}_{\mathsf{Set}},\Delta^{\rightleftarrows}_{\mathsf{Set}}} \cong \mathbf{Learn}$$

$$D^*_{\eta} : \mathbf{GradLearn} := \mathbf{Dioptic}_{T^{\triangleleft}_{\mathbb{R}},T^{\triangleleft}_{\mathbb{R}}} \to \mathbf{Dioptic}_{T^{\triangleleft}_{\mathbb{R}},\Delta^{\rightleftarrows}_{\mathsf{Set}}} =: \mathbf{GradDesc}$$

  all go from one category of dioptics to another.
- Is there a generic "recipe"?

Nonsmooth activation functions



$$\mathrm{ReLU}(x) := \max\{x, 0\}$$

- ReLU ("Rectified Linear Unit") is a pervasive ML primitive

At least 5 ways to handle:

0. Pretend $\mathrm{ReLU}'(0) := 1$
1. Smooth almost everywhere
2. Subdifferentiable
3. Semismooth from the right
4. $\mathrm{ReLU}'(0) := \bot$ 😎

Selected References

Riley, Mitchell. *Categories of Optics*, 7th Sept. 2018, arXiv: 1809.00738v2 [math.CT] (cited on p. 14).

Fong, Brendan, David I. Spivak and Rémy Tuyéras. *Backprop as Functor: A compositional perspective on supervised learning*, 13th Dec. 2017, arXiv: 1711.10455v2 [math.CT] (cited on pp. 16, 17, 30).

Ghani, Neil, Jules Hedges, Viktor Winschel and Philipp Zahn. "Compositional game theory", *Logic in Computer Science*, LICS '18 (Oxford, UK), 9th–12th July 2018, DOI: 10.1145/3209108.3209165 (cited on p. 32); preliminary version on arXiv: 1603.04641v3 [cs.GT] (15th Mar. 2016).

Elliott, Conal. "The simple essence of automatic differentiation", *Proc. ACM on Programming Languages*, ICFP 2018 (St. Louis, MO, USA), vol. 2.70, 24th–26th Sept. 2018, DOI: 10.1145/3236765; extended version on arXiv: 1804.00746v4 [cs.PL]; URL: http://conal.net/papers/essence-of-ad/ (Mar. 2018).

Sprunger, David and Shin-ya Katsumata. *Differentiable Causal Computations via Delayed Trace*, 4th Mar. 2019, arXiv: 1903.01093v1 [cs.LO].

Acknowledgments

- Thanks to Eliana Lorch for key insights
- Thanks to Jules Hedges, David Spivak, and Brendan Fong for support and conversations.

Thank you for your attention!

David A. Dalrymple
@davidad

Protocol Labs

SYCO 5
Birmingham, UK
2019-09-05