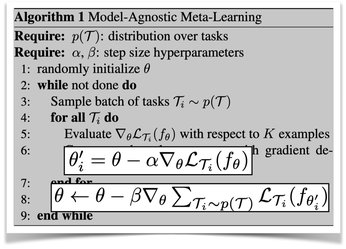
Nominal Semantics of First-Class Automatic Differentiation

Jack Czenszak and Alexander K. Lew

Yale

Motivation

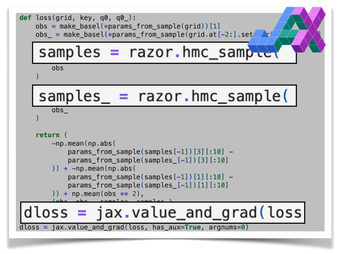
- First-class automatic differentiation (AD) is a desirable language feature used throughout the machine learning literature (see right);
- Many modern machine learning frameworks (JAX, PyTorch, and JuliaDiff [4]) implement first-class AD using the *tagged forward-mode algorithm* to avoid *perturbation confusion* (see below) [6];
- However, despite effort, this algorithm has *not* been formally proven correct, making these heavily-used systems untrustworthy.



Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
randomly initialize θ
while not done **do**
  Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
  for all $\mathcal{T}_i$ **do**
    Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
    $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
  end for
  $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
end while

Finn et al. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. 2017.



```
def loss(grid, key, q0, q0_):
    obs = make_basel(*params_from_sample(grid))[1]
    obs_ = make_basel(*params_from_sample(grid.at[-2:].set
    samples = razor.hmc_sample(
        obs
    )
    samples_ = razor.hmc_sample(
        obs_
    )
    return (
        -np.mean(np.abs(
            params_from_sample(samples[-1])[3][:10] -
            params_from_sample(samples_[-1])[3][:10]
        )) + -np.mean(np.abs(
            params_from_sample(samples[-1])[1][:10] -
            params_from_sample(samples_[-1])[1][:10]
        )) + np.mean(obs == 2),
dloss = jax.value_and_grad(loss
dloss = jax.value_and_grad(loss, has_aux=True, argnums=0)
```

Chandra et al. Designing Perceptual Puzzles by Differentiating Probabilistic Programs. 2022.

First-Class AD with Standard Dual Numbers (Bad)

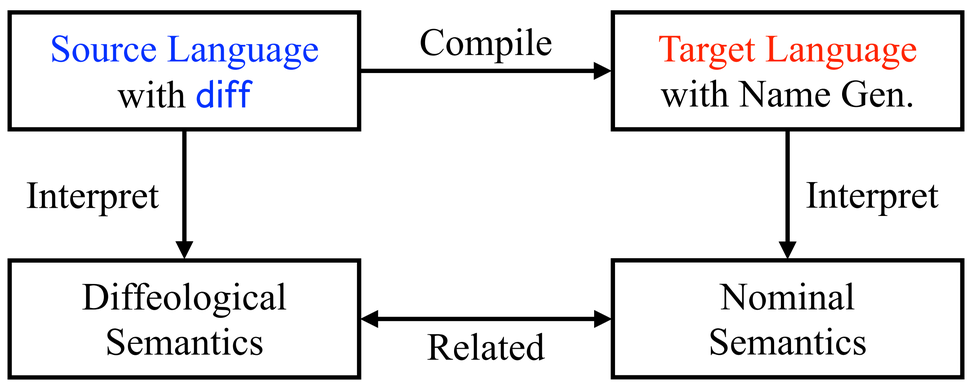
$$
\begin{aligned}
&\mathsf{diff}\ (\lambda x\,.\,x \cdot (\mathsf{diff}\ (\lambda y\,.\,x+y)\ 1))\ 1 \\
&= \mathsf{coeff}\ ((\lambda x\,.\,x \cdot (\mathsf{diff}\ (\lambda y\,.\,x+y)\ 1))\ (1+\varepsilon)) \\
&= \mathsf{coeff}\ ((1+\varepsilon) \cdot (\mathsf{diff}\ (\lambda y\,.\,(1+\varepsilon)+y)\ 1)) \\
&= \mathsf{coeff}\ ((1+\varepsilon) \cdot (\mathsf{coeff}\ ((\lambda y\,.\,(1+\varepsilon)+y)\ (1+\varepsilon)))) \\
&= \mathsf{coeff}\ ((1+\varepsilon) \cdot (\mathsf{coeff}\ ((1+\varepsilon)+(1+\varepsilon)))) \\
&= \mathsf{coeff}\ ((1+\varepsilon) \cdot \underline{(\mathsf{coeff}\ (2+2\varepsilon))})\ \text{Combined!} \\
&= \mathsf{coeff}\ ((1+\varepsilon) \cdot 2) = \mathsf{coeff}\ (2+2\varepsilon) = 2\ \text{❌}
\end{aligned}
$$

First-Class AD with Tagged Forward-Mode (Correct)

$$
\begin{aligned}
&\mathsf{diff}\ (\lambda x\,.\,x \cdot (\mathsf{diff}\ (\lambda y\,.\,x+y)\ 1))\ 1 \\
&= \mathsf{coeff}_1\ ((\lambda x\,.\,x \cdot (\mathsf{diff}\ (\lambda y\,.\,x+y)\ 1))\ (1+\varepsilon_1)) \\
&= \mathsf{coeff}_1\ ((1+\varepsilon_1) \cdot (\mathsf{diff}\ (\lambda y\,.\,(1+\varepsilon_1)+y)\ 1)) \\
&= \mathsf{coeff}_1\ ((1+\varepsilon_1) \cdot (\mathsf{coeff}_2\ ((\lambda y\,.\,(1+\varepsilon_1)+y)\ (1+\varepsilon_2)))) \\
&= \mathsf{coeff}_1\ ((1+\varepsilon_1) \cdot (\mathsf{coeff}_2\ ((1+\varepsilon_1)+(1+\varepsilon_2)))) \\
&= \mathsf{coeff}_1\ ((1+\varepsilon_1) \cdot \underline{(\mathsf{coeff}_2\ (2+\varepsilon_1+\varepsilon_2))})\ \text{Distinct} \\
&= \mathsf{coeff}_1\ ((1+\varepsilon_1) \cdot 1) = \mathsf{coeff}_1\ (1+\varepsilon_1) = 1\ \text{✅}
\end{aligned}
$$

Formalizing the Tagged Forward-Mode Algorithm

- Compile a high-level *source language* with first-class diff to a *target language* (with an *obvious* implementation strategy) that can express the tagged forward-mode algorithm using name generation;
- Source language is given semantics $[\![-]\!]_1$ in the category of diffeological spaces **Diff** [1];
- Target language is given semantics $[\![-]\!]_2$ in the Schanuel topos $\mathbf{Sch} = [\mathbf{I}, \mathbf{Set}]_{\text{pb}}$, a well-studied semantic model for name generation [7];
- Compiler is proven correct using categorical logical relations.



Source and Target Languages

Source Language Syntax

$$
\begin{aligned}
\text{Type} \ni A, B &::= \mathsf{real} \mid \mathsf{unit} \mid A \times B \mid A \Rightarrow B \\
\text{Expr} \ni M, N &::= x \mid \mathsf{r} \in \mathbb{R} \mid \mathsf{op} \mid \mathsf{diff} \mid \langle\rangle \mid \langle M, N \rangle \\
&\quad \mid \mathsf{fst}\ M \mid \mathsf{snd}\ M \mid \lambda x : A\,.\,M \mid MN
\end{aligned}
$$

Target Language Syntax

$$
\begin{aligned}
\text{Type} \ni A, B &::= \mathsf{dual} \mid \mathsf{name} \mid \mathsf{unit} \mid A \times B \mid A \Rightarrow B \mid \mathcal{T} A \\
\text{Expr} \ni M, N &::= x \mid \mathsf{r} \in \mathbb{R} \mid \mathsf{op} \mid \mathsf{coeff} \mid \mathsf{epsil} \mid \mathsf{new} \mid \langle\rangle \\
&\quad \mid \langle M, N \rangle \mid \mathsf{fst}\ M \mid \mathsf{snd}\ M \mid \lambda x : A\,.\,M \\
&\quad \mid MN \mid x \Leftarrow M;\ N \mid \mathsf{ret}\ M
\end{aligned}
$$

Source Language Semantics in **Diff**

$$
[\![\mathsf{real}]\!]_1 = \mathbb{R} \qquad [\![A \Rightarrow B]\!]_1 = \mathbf{Diff}([\![A]\!]_1, [\![B]\!]_1)
$$
$$
[\![\mathsf{unit}]\!]_1 = \{\star\} \qquad [\![\mathsf{diff}]\!]_1(\gamma) = d
$$

- $d : \mathbf{Diff}(\mathbb{R}, \mathbb{R}) \to \mathbf{Diff}(\mathbb{R}, \mathbb{R})$ is first-order differentiation.

Target Language Semantics in **Sch**

$$
[\![\mathsf{dual}]\!]_2 = \mathbb{D} \qquad [\![\mathsf{coeff}\ M\ N]\!]_2 = \mathbf{coeff} \circ \langle [\![M]\!]_2, [\![N]\!]_2 \rangle
$$
$$
[\![\mathsf{name}]\!]_2 = \mathbb{A} \qquad [\![\mathsf{epsil}\ M]\!]_2 = \mathbf{epsil} \circ [\![M]\!]_2
$$
$$
[\![\mathcal{T} A]\!]_2 = \mathcal{T} [\![A]\!]_2 \qquad [\![\mathsf{new}]\!]_2 = \mathbf{new} \circ !_{[\![\Gamma]\!]_2}
$$

- $\mathcal{T} F = \text{colim}_{X \in \mathbf{I}} F(- + X)$ is the *name generation monad* [7];
- $\mathbb{A}(X) = X$ is the *names object* [7];
- $\mathbf{new}_X(\star) = [1, \mathbf{inr}_{X,1}(\star)]$ generates a *fresh name* [7];
- *Dual numbers* $\mathbb{D}$, $\mathbf{coeff} : \mathbb{A} \times \mathbb{D} \to \mathbb{D}$, and $\mathbf{epsil} : \mathbb{A} \to \mathbb{D}$ are:

$$
\mathbb{D}(X) = \left\{ \sum_{S \subseteq X} p_S \prod_{x \in S} x \in \mathbb{R}[X] \mid p_S \in \mathbb{R} \right\}
$$
$$
\mathbf{coeff}_X(x, p) = p_1 \text{ where } p = p_1 x + p_2 \qquad \mathbf{epsil}_X(x) = x
$$

Acknowledgements. We would like to thank the anonymous LAFI 2026 reviewers for their comments and Pedro H. Azevedo de Amorim, Mathieu Huot, John M. Li, Cameron Moy, and Sam Staton for helpful discussions. **References.** [1] Huot et al. Correctness of automatic differentiation via diffeologies and categorical gluing. 2020. [2] Katsumata. A semantic formulation of ⊤⊤-lifting and logical predicates for computational metalanguage. 2005. [3] Krawiec et al. Provably correct, asymptotically efficient, higher-order reverse-mode automatic differentiation. 2022. [4] Manzyuk et al. Perturbation confusion in forward automatic differentiation of higher-order functions. 2019. [5] Mitchell and Scedrov. Notes on sconing and relators. 1992. [6] Siskind and Pearlmutter. Perturbation confusion and referential transparency: Correct functional implementation of forward-mode AD. 2005. [7] Stark. Names and Higher-Order Functions. 1994.

Compiler Correctness

Source-to-Target Compiler

- Well-typed source programs $\Gamma \vdash M : A$ are compiled to well-typed target programs $\mathscr{C}(\Gamma) \vdash \mathscr{C}(M) : \mathcal{T}\mathscr{C}(A)$ according to:

$$
\begin{aligned}
\mathscr{C}(\mathsf{real}) &= \mathsf{dual} & \mathscr{C}(\langle M, N \rangle) &= x \Leftarrow \mathscr{C}(M); \\
\mathscr{C}(A \times B) &= \mathscr{C}(A) \times \mathscr{C}(B) & &\quad\ y \Leftarrow \mathscr{C}(N); \\
\mathscr{C}(A \Rightarrow B) &= \mathscr{C}(A) \Rightarrow \mathcal{T}\mathscr{C}(B) & &\quad\ \mathsf{ret}\ \langle x, y \rangle \\
\mathscr{C}(\mathsf{r}) &= \mathsf{ret}\ \mathsf{r} & \mathscr{C}(MN) &= f \Leftarrow \mathscr{C}(M); \\
\mathscr{C}(\lambda x\,.\,M) &= \mathsf{ret}\ \lambda x\,.\,\mathscr{C}(M) & &\quad\ x \Leftarrow \mathscr{C}(N); \\
\mathscr{C}(\mathsf{fst}\ M) &= x \Leftarrow \mathscr{C}(M);\ \mathsf{ret}\ x & &\quad\ fx \\
\mathscr{C}(\mathsf{diff}) &= \mathsf{ret}\ \lambda f : \mathsf{dual} \Rightarrow \mathcal{T}\mathsf{dual}\,.\,\mathsf{ret}\ \lambda x : \mathsf{dual}\,. \\
& & & n \Leftarrow \mathsf{new};\ d \Leftarrow f(x + \mathsf{epsil}\ n);\ \mathsf{ret}\ \mathsf{coeff}\ n\ d
\end{aligned}
$$

Correctness Proof by Logical Relations

Theorem (Correctness). Suppose $\cdot \vdash M : \mathsf{real}$ has $[\![M]\!]_1 = [\![\mathsf{r}]\!]_1$ for some $r \in \mathbb{R}$, then $[\![\mathscr{C}(M)]\!]_2 = [\![\mathsf{ret}\ \mathsf{r}]\!]_2$.

Lemma (Fundamental Property). Consider $\Gamma \vdash M : A$. For any pair $(\gamma_1, \gamma_2) \in \mathscr{V}(\Gamma)$, we get $([\![M]\!]_1 \circ \gamma_1, [\![\mathscr{C}(M)]\!]_2 \circ \gamma_2) \in \dot{\mathcal{T}}\mathscr{V}(A)$.

- Proven by categorical *logical relations* [5] indexed by Kripke worlds (similar to [3]) of finite sets of tags:

$$
\mathscr{V}(\mathsf{real})(X) = \left\{ (f, g) \in K(\mathbb{R}, \mathbb{D}) \mid g(i, y) = \sum_{S \subseteq X} \frac{\partial |S|}{\partial S} f(y_{i(x)})_{x \in X} \prod_{x \in S} i(x) \right\}
$$
$$
\mathscr{E}(\mathsf{real})(X) = \left\{ (f, g) \in K(\mathbb{R}, \mathcal{T}\mathbb{D}) \mid g = \eta_{\mathbb{D}} \circ h, (f, h) \in \mathscr{V}(\mathsf{real})(X) \right\}
$$

  with $K(X, F) = \mathbf{Diff}(\mathbb{R}^{\mathbb{A}(-)}, X) \times (\mathbb{P} \Rightarrow F)$ and $\mathbb{P}(X) = \mathbb{R}^{\mathbb{A}(X)}$ s.t. $\mathbb{P}(f)$ pads with zeroes.

- Construct a *gluing category* **Gl** and *fibration for logical relations* [2] $p : \mathbf{Gl} \to \mathbf{Diff} \times \mathbf{Sch}$ by pulling back the evident sub-object fibration along $K : \mathbf{Diff} \times \mathbf{Sch} \to [\mathbf{I}, \mathbf{Set}]$, and lift $\mathbf{id}_{\mathbf{Diff}} \times \mathcal{T}$ using ⊤⊤-lifting [3] with param. $(\langle [\![-]\!]_1, [\![\mathscr{C}(-)]\!]_2 \rangle, \mathscr{E})$ to get $\dot{\mathcal{T}}$ on **Gl**;
- Proof is then by induction, with diff being the main difficulty.