

ω PAP Spaces: Reasoning Denotationally About Higher-Order, Recursive Probabilistic and Differentiable Programs

Mathieu Huot*, Alexander K. Lew*, Vikash K. Mansinghka, and Sam Staton

Abstract—We introduce a new setting, the category of ω PAP spaces, for reasoning denotationally about expressive differentiable and probabilistic programming languages. Our semantics is *general enough* to assign meanings to most practical probabilistic and differentiable programs, including those that use general recursion, higher-order functions, discontinuous primitives, and both discrete and continuous sampling. But crucially, it is also *specific enough* to exclude many pathological denotations, enabling us to establish new results about both deterministic differentiable programs and probabilistic programs. In the deterministic setting, we prove very general correctness theorems for automatic differentiation and its use within gradient descent. In the probabilistic setting, we establish the almost-everywhere differentiability of probabilistic programs’ trace density functions, and the existence of convenient base measures for density computation in Monte Carlo inference. In some cases these results were previously known, but required detailed proofs with an operational flavor; by contrast, all our proofs work directly with programs’ denotations.

I. INTRODUCTION

This paper introduces a new setting, the category of ω PAP spaces, for reasoning denotationally about expressive probabilistic and differentiable programs, and demonstrates its utility in several applications. The ω PAP spaces are built using the same categorical machinery [1] that underlies the ω -quasi-Borel spaces [2] and the ω -diffeological-spaces [3, 4], two other recent proposals for understanding higher-order, recursive probabilistic and differentiable programs. The key difference is that instead of taking the *measurable* maps (as in ω Qbs) or the *smooth* maps (as in ω Diff) as the primitives in our development, we instead use the functions that are *piecewise analytic under analytic partition*, or PAP [5].

Whereas the smooth functions *exclude* many primitives used in practice (e.g., $< : \text{real} \rightarrow \text{real} \rightarrow \mathbb{B}$), and the measurable functions *admit* many pathological examples from analysis (the Cantor function, the Weierstrass function, space-filling curves), the PAP functions manage to exhibit very few pathologies while still including nearly all primitives exposed by today’s differentiable and probabilistic programming languages. As a result, our semantics can interpret most differentiable and probabilistic programs that arise in practice, *and* can be used to provide short denotational proofs of many interesting properties. As evidence for this claim, we use our semantics to establish new results (and give new, simplified proofs of old results) about the correctness of automatic differentiation, the

convergence of automatic-differentiation gradient descent, and the supports and densities of recursive probabilistic programs.

A. A “Just-Specific-Enough” Semantic Model

In denotational accounts of deterministic and probabilistic programming languages, deterministic programs typically denote *functions* and probabilistic programs typically denote *measure kernels*. But what kinds of functions and kernels?

The more specific our answer to that question, the more we can hope to prove—purely denotationally—about our languages. For example, consider the question of *commutativity*: is it the case that, whenever x does not occur free in s ,

$$\llbracket \text{let } x = t \text{ in let } y = s \text{ in } u \rrbracket = \llbracket \text{let } y = s \text{ in let } x = t \text{ in } u \rrbracket?$$

In the probabilistic setting, this amounts to asking whether iterated integrals can always be reordered. If our semantics interprets programs as *arbitrary* measure kernels, then there is no obvious answer, because Fubini’s theorem, which justifies the interchange of iterated integrals, does not apply unconditionally. But by interpreting an expressive probabilistic language using only the *s-finite kernels*, Staton [6] was able to prove commutativity denotationally, using the specialization of Fubini’s theorem to the s-finite case.

Unfortunately, in the quest for semantic models with convenient properties, we may end up excluding programs that programmers might write in practice. For example, in [3], all programs are interpreted as *smooth* functions, enabling an elegant semantic account of why automatic differentiation algorithms work (even in the presence of higher-order functions). But to achieve this nice theory, all non-smooth *primitives* (including, e.g., the function $\lambda x.\lambda y.x < y$) were excluded from the language under consideration. As a result, the theory has little to say about the behavior of automatic differentiation on non-smooth programs.

In this work, our aim is to find a “just specific enough” semantic model of expressive probabilistic and differentiable programming languages: specific enough to establish interesting properties via denotational reasoning, but general enough to include nearly all programs of practical interest. Like the recently introduced *quasi-Borel predomains* [2], our model covers most probabilistic and differentiable programs that can be expressed in today’s popular languages: it supports general recursion, higher-order functions, discrete and continuous sampling, and a broad class of primitive functions that includes nearly all the mathematical operations exposed by

*Equal contribution

```

cantor : R -> R
cantor x =
  if x <= 1/3 then
    cantor (3 * x)
  else if x < 2/3 then
    x * x
  else
    cantor (3 * x - 2)

```

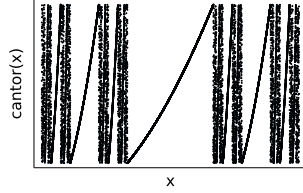


Fig. 1. A program that diverges on the $\frac{1}{3}$ -Cantor set, but halts elsewhere in $(0, 1)$. The characteristic function of the $\frac{1}{3}$ -Cantor set is not PAP, so a naively defined interpretation of “partial PAP functions into \mathbb{R} ” as PAP functions into $\mathbb{R} + 1$ would not suffice to interpret a language with general recursion. Our definition of partial ω PAP maps in Section II resolves the issue.

comprehensive libraries for scientific computing and machine learning (such as `numpy` and `scipy`). But crucially, it also *excludes* many pathological functions and kernels that cannot be directly implemented in practice, such as the characteristic function of the Cantor set (whose construction involves an infinite limit that programs cannot compute in finite time). As a result, it is often possible to establish that a desirable property holds of *all* ω PAP maps between two spaces, and to then conclude that it holds of all programs of the appropriate type, without reasoning inductively about their construction or their operational semantics.

The starting point in our search for a semantic model is Lee et al. [5]’s notion of *piecewise analyticity under analytic partition* (or PAP). The PAP functions are a particularly well-behaved subset of the functions between Euclidean spaces, and [5] makes a compelling argument that they are “specific enough” in the sense we describe above. For example, even though they are not necessarily differentiable, PAP functions admit a generalized notion of derivative, as well as a generalized chain rule that can be used to give a denotational justification of automatic differentiation. These nice properties do not hold for slight generalizations of PAP, e.g. the class of almost-everywhere differentiable functions.

Nevertheless, a reasonable worry is that PAP functions may be *too* narrow a semantic domain: do all programs arising in practice really denote PAP functions, or does the definition assume properties that not all programs enjoy? Lee et al. [5] establish that in a first-order language with PAP primitives and conditionals, all programs denote PAP functions. But real-world programs use many additional features, such as higher-order functions, general recursion, and probabilistic effects. In the presence of these features, do terms of first-order type still denote PAP functions? Consider the program in Fig. 1, for example: it uses recursion to define a partial function that diverges on the $\frac{1}{3}$ -Cantor set, and halts on its complement. But we know that the (total) *indicator function* for the $\frac{1}{3}$ -Cantor set is not PAP. This may seem like reason to doubt that [5]’s proposal can be cleanly extended to the general-recursive setting.

Fortunately, however, it can: in Section II, we extend the definition of PAP to cover partial and higher-order functions. The resulting class of functions, the ω PAP maps, suffice to

interpret all recursive, higher-order programs in an expressive call-by-value PCF, with a type of real numbers and PAP primitives (Theorem II.2). The ω PAP category also supports a strong monad of measures (Definition IV.5), enabling an interpretation of continuous sampling and soft conditioning, features common in modern probabilistic languages.

To be clear, existing semantic frameworks can interpret similarly expressive languages (e.g., [2]). Our value proposition is that the ω PAP domain *usefully excludes* certain pathological denotations that other settings allow. After defining our semantics in Section II, we devote the remainder of the paper to presenting evidence for this claim: new results (and new proofs of old results) that follow cleanly from denotational reasoning about ω PAP spaces and maps.

B. Differentiable Programming and Automatic Differentiation

Our first application of our semantics is to the study of *automatic differentiation* (AD), a family of techniques for mechanically computing derivatives of user-defined functions. It has found widespread use in machine learning and related disciplines, where practitioners are often interested in using gradient-based optimization algorithms, such as gradient descent, to fit model parameters to data. Languages with support for automatic differentiation are often called “differentiable programming languages,” even if it may be the case that some programs do not denote everywhere-differentiable functions.

When applied to straight-line programs built from differentiable primitives (without `if` statements, loops, and other control flow), AD has a straightforward justification using the chain rule. But in the presence of more expressive programming constructs, AD is harder to reason about: some programs encode partial or non-differentiable functions, and even when programs *are* differentiable, AD can fail to compute their true derivatives at some inputs.

Our ω PAP semantics gives a clean account of the class of partial functions that recursive, higher-order programs built from “AD-friendly” primitives can express. In Section III, we use this characterization to establish guarantees about AD’s behavior when applied to such programs:

- We define a generalized notion of derivative, based on [5]’s *intensional derivatives*, and establish that every first-order ω PAP map is intensionally differentiable.
- We adapt Huot et al. [3]’s denotational correctness proof of AD to prove that AD correctly computes intensional derivatives of all programs with first-order type, even if they use recursion and discontinuous primitives (Theorem III.3). This result is stronger than Mazza and Pagani [7]’s, which follows as a straightforward corollary. But the main benefit over their development is the simplicity of our proof, which mirrors Huot et al. [3]’s quite closely despite the greatly increased complexity of our language.
- Using this characterization of what AD computes, we further prove the novel result that intensional gradient descent (which optimizes a *differentiable* function but using gradients computed by AD) converges with probability 1, if initialized with a randomized initial location *and* a

randomized learning rate (Theorem III.7). This implies that gradient descent can be safely applied to recursive programs with conditionals so long as they denote differentiable functions, even if AD (which can disagree with the true derivative at some inputs) is used to compute gradients. Interestingly, if either the initial location *or* the learning rate is *not* randomized, a.s.-convergence is not guaranteed, and we present counterexamples of PyTorch programs that denote differentiable functions but cause PyTorch’s gradient descent implementation to diverge (Propositions III.5 & III.6).

Taken together, these results give a characterization of the behavior of AD in real systems like PyTorch and Tensorflow, even when they are applied to non-differentiable or recursive programs.

C. Probabilistic Programming

Probabilistic programming languages extend deterministic languages with support for *random sampling* and *soft conditioning*, making them an expressive representation both for probability distributions and for general (unnormalized) measures. Many probabilistic programs are intended to model some aspect of the world, and many questions of interest can be posed as questions about the expectations of certain functions under (normalized versions of) these models. Probabilistic programming languages often come with tools for automatically running *inference algorithms* that estimate such posterior expectations. But the machinery for inference may make assumptions about the programs the user has written—for example, that the program has a density with respect to a well-behaved reference measure, or that its density is differentiable. Our next application of our semantics is to the problem of verifying that such properties hold for any program in an expressive probabilistic language.

In Section IV, we extend the core language from Section II to include the **sample** and **score** commands. This requires changing our monad of effects, which in Section II covered only divergence, to also interpret randomness and conditioning. We consider two different strong monads on ωPAP : one interprets probabilistic programs as s-finite measures (following [2]), and the other as weighted sampling procedures (following [8, 9]). In each case, we can prove interesting properties of all probabilistic programs by working just with the class of ωPAP maps they denote:

- Interpreting programs as weighted samplers, we prove that almost-surely terminating probabilistic programs have almost-everywhere differentiable weight functions (Theorem IV.1). This has been previously shown by Mak et al. [9], but our proof is substantially simpler, with no need for [9]’s stochastic symbolic execution technique. Our theorem is also slightly stronger, covering some programs that do not almost surely halt, and ruling out some pathological a.e.-differentiable weight functions.
- Interpreting probabilistic programs as ωPAP measures, we prove that all programs of type \mathbb{R}^n denote distributions supported on a countable union of smooth

submanifolds of \mathbb{R}^n , and thus admit convenient densities with respect to a particular class of reference measures (Theorem IV.3).

Besides being interesting in their own right, these results have consequences for the practical design and implementation of probabilistic programming systems.

The first result is relevant to the application of gradient-based inference algorithms, which often require (at least) almost-everywhere differentiability to be sound.

The second result is relevant for the automatic computation of *densities* or *Radon-Nikodym derivatives* of probabilistic programs, key ingredients in higher-level algorithms such as importance sampling and MCMC. To see why, suppose a user has constructed two closed probabilistic programs of type \mathbb{R}^n , p and q , and wishes to use q as an importance sampling proposal for p . If $\llbracket p \rrbracket$ is absolutely continuous with respect to $\llbracket q \rrbracket$ ($\llbracket p \rrbracket \ll \llbracket q \rrbracket$), such an importance sampler does exist, but implementing it requires computing importance weights: at a point $x \sim \llbracket q \rrbracket$, we must compute $\frac{d\llbracket p \rrbracket}{d\llbracket q \rrbracket}(x)$. If $\llbracket p \rrbracket$ and $\llbracket q \rrbracket$ were both absolutely continuous with respect to the Lebesgue measure on \mathbb{R}^n , we could compute probability density functions ρ_p and ρ_q for $\llbracket p \rrbracket$ and $\llbracket q \rrbracket$ respectively, and then compute the importance weight as the ratio of these densities. But not all programs denote measures that *have* densities with respect to the Lebesgue measure.¹ Our result gives us an alternative to the Lebesgue measure: we can compute the density of $\llbracket p \rrbracket$ (and $\llbracket q \rrbracket$) with respect to the *Hausdorff measures* over the manifolds on which they are supported. (Indeed, computing densities with respect to Hausdorff measures has previously been proposed by Radul and Alexeev [10]; our result is that, unlike the Lebesgue base measure, this choice does not *exclude* any possible programs a user might write, because *every* program is absolutely continuous with respect to such a base measure.)

D. Summary

- We present the category of ωPAP spaces (Section II), a new setting suitable for a denotational treatment of higher-order, recursive differentiable or probabilistic programming languages. It supports sound and adequate semantics of a call-by-value PCF with a type of real numbers and a very expressive class of primitives.
- To demonstrate that our semantics enables denotational reasoning about a broad class of interesting program properties, we present new results (and new, much-simplified proofs of old results) about differentiable (Section III) and probabilistic (Section IV) programs.

II. ωPAP SEMANTICS

This section develops the ωPAP spaces, and shows how to use them to interpret a higher-order, recursive language

¹Consider, e.g., the program that samples $u \sim \text{Unif}(0, 1)$, and returns $(u, u) \in \mathbb{R}^2$. Because it is supported only on a 1-dimensional line segment within the plane, it has no density function with respect to the Lebesgue measure in the plane. That is, there is no nonnegative function ρ such that the probability of an event $E \subseteq \mathbb{R}^2$ is equal to $\iint_{\mathbb{R}^2} 1_E(x, y) \cdot \rho(x, y) dx dy$.

with conditionals and discontinuous primitives. Our starting point is [5]’s definition of *piecewise analyticity under analytic partition* (PAP), a property of some functions defined on Euclidean spaces. The ω PAP construction significantly extends Lee et al. [5]’s definition to cover *partial* and *higher-order* functions between Euclidean or other spaces. The ω PAP setting is general enough to interpret almost all primitives encountered in practice, but restricted enough to allow precise denotational reasoning.

A. ω PAP Spaces

We first recall a standard definition from analysis:

Definition II.1 (analytic function). *If the Taylor series of a smooth function $f : U \rightarrow V$ (for $U \subseteq \mathbb{R}^n, V \subseteq \mathbb{R}^m$) converges pointwise to f in a neighborhood around x , we say f is analytic at x . An analytic function is a function that is analytic at every point in its domain.*

When a subset of \mathbb{R}^n can be carved out by finitely many analytic inequalities, we call it an *analytic set*, following Lee et al. [5]. We call countable unions of such sets *c-analytic*.

Definition II.2 (analytic set [5]). *We call a set $A \subseteq \mathbb{R}^n$ an analytic set if there exists a finite collection $\{g_i\}_{i \in I}$ of analytic functions into \mathbb{R} , with open domain $U \subseteq \mathbb{R}^n$, such that*

$$A = \{x \in U \mid \forall i \in I. g_i(x) \leq 0\}.$$

Definition II.3 (c-analytic set). *A set $A \subseteq \mathbb{R}^n$ is c-analytic if it is a countable union of analytic subsets of \mathbb{R}^n .*

Lee et al. [5]’s key definition was the class of *PAP functions*, which are piecewise analytic:

Definition II.4 (PAP function [5]). *For $U \subseteq \mathbb{R}^n$ c-analytic and V any subset of \mathbb{R}^m , we say $f : U \rightarrow V$ is piecewise analytic under analytic partition (PAP) if there exists a countable family $\{(A_i, f_i)\}_{i \in I}$ such that:*

- 1) *the sets A_i are analytic and form a partition of U ;*
- 2) *each $f_i : U_i \rightarrow \mathbb{R}^m$ is an analytic function defined on an open domain $U_i \supseteq A_i$;*
- 3) *when $x \in A_i$, $f_i(x) = f(x)$.*

In our development, PAP functions will play, at first order, the role that smooth functions play in the construction of diffeological spaces [11], and that measurable functions play in the construction of quasi-Borel spaces [12]. The change to PAP is essential: many primitives exposed by languages like TensorFlow and PyTorch fail to be smooth at some inputs, but virtually none fail to be PAP. And while PAP functions are all measurable, they helpfully exclude many pathological measurable functions, making it possible to prove stronger results through denotational reasoning.

The development that follows generalizes PAP functions, so that we can talk about higher-order and partial PAP functions. To do so, it uses recently developed categorical techniques [1, 2, 4] (see Appendix B for details).

Definition II.5 (ω PAP space). *An ω PAP space is a triple $(|X|, \mathcal{P}_X, \leq_X)$, where $(|X|, \leq)$ is an ω cpo and \mathcal{P}_X is a family*

of sets of functions, called plots in X . For each c-analytic set A , we have $\mathcal{P}_X^A \subseteq |X|^A$. Plots have to satisfy the following closure conditions:

- *Every map from the one point of \mathbb{R}^0 to $|X|$ is a plot in X .*
- *If $\phi \in |X|^A$ is a plot in X and $f : A' \rightarrow A$ is a PAP function between c-analytic sets A' and A , then $\phi \circ f$ is a plot in X .*
- *Suppose the c-analytic sets $A_j \subseteq A$ form a countable partition of the c-analytic set A , with inclusions $i_j : A_j \rightarrow A$. If $\phi \circ i_j$ is a plot in X for every j , then ϕ is a plot in X .*
- *Whenever $(\alpha_i)_i$ is an ω -chain in \mathcal{P}_X^A under the pointwise order, $(\bigvee_i \alpha_i)(x) := \bigvee_i (\alpha_i(x))$ defines a plot $\bigvee_i \alpha_i \in \mathcal{P}_X^A$.*

Definition II.6 (ω PAP map). *An ω PAP map $f : X \rightarrow Y$ is a Scott-continuous function between the underlying ω cpos $(|X|, \leq_X)$ and $(|Y|, \leq_Y)$, such that for every plot $\phi \in \mathcal{P}_X^A$, $f \circ \phi \in \mathcal{P}_Y^A$.*

We now look at several examples of ω PAP-spaces. Our first example establishes that PAP functions between Euclidean spaces are a special case of ω PAP maps.

Example II.1. *Any c-analytic set A (including, e.g., \mathbb{R}^n itself) can be given the structure of an ω PAP-space $(A, \mathcal{P}_A, =)$ where for every c-analytic set C , the plots \mathcal{P}_A^C are given by*

$$\mathcal{P}_A^C := \{f : C \rightarrow A \mid f \text{ is a PAP function}\}.$$

Next, we construct products, coproducts, and exponential ω PAP spaces, which will be useful for interpreting the tuples, sums, and function types of our language.

Example II.2 (Products). *Given ω PAP spaces X and Y , define $X \times Y$ to be their product as ω cpos, with plots $\langle f, g \rangle \in \mathcal{P}_{X \times Y}^A$ whenever $f \in \mathcal{P}_X^A$ and $g \in \mathcal{P}_Y^A$.*

More generally, given ω PAP spaces X_i for $i \in I$, we can define $\prod_{i \in I} X_i$ by their product as ω cpos, with plots $f \in \mathcal{P}_{\prod_{i \in I} X_i}^A$ whenever each projection $\pi_i \circ f \in \mathcal{P}_{X_i}^A$.

Example II.3 (Coproducts). *Let I be a countable index set, and X_i a ω PAP space for each $i \in I$. Then $\bigsqcup_i X_i$ is again an ω PAP space, with carrier set $\bigsqcup_i |X_i|$, and the partial order inherited from each X_i (elements of different spaces X_i and X_j are not comparable). A function $f : A \rightarrow \bigsqcup_i |X_i|$ is a plot if there exists a countable partition $\{A_j\}_j$ of A into c-analytic sets, such that for each j , there is a plot $f_j : A_j \rightarrow X_{k_j}$, where $k_j \in I$, such that on A_j , $f(x) = \mathbf{in}_{k_j} f_j(x)$, i.e., if the preimage of each X_i under f is a c-analytic set for each i .*

Example II.4 (Exponentials). *Let X and Y be ω PAP spaces. Then $X \Rightarrow Y$ is again a ω PAP space, with carrier set $\omega\text{PAP}(X, Y)$, the set of ω PAP-morphisms from X to Y . $f \leq_{X \Rightarrow Y} g$ iff for all x , $f(x) \leq_Y g(x)$. A function $f : A \rightarrow |X \Rightarrow Y|$ is a plot if $\mathbf{uncurry} f : A \times X \rightarrow Y$ is an ω PAP morphism.*

B. Core Calculus

We present our core calculus in Figure 2. It is a simple call-by-value variant of PCF (e.g. as in Abramsky and McCusker [13]), with a ground type \mathbb{B} for Booleans, and a ground type **real** for real numbers. It is parametrized by a set of primitives f representing PAP functions. They typically include analytic functions such as $+$, $*$, \exp , \tan and comparison operators such as $>$, $=$. The language also includes constants c for each real c and **true**, **false** for Booleans. The typing rules and operational semantics are standard, and given in Appendices A-A and A-B.

$\tau ::=$	types
real	real numbers
\mathbb{B}	booleans
$\tau_1 \times \tau_2$	product
$\tau_1 \rightarrow \tau_2$	function
$t ::=$	terms
x	variable
$c \mid f(t_1, \dots, t_n)$	constants/operations
$\langle t_1, t_2 \rangle$	pair constructor
case t_1 of $\langle x_1, x_2 \rangle \rightarrow t_2$	pair elimination
if t_1 then t_2 else t_3	conditionals
$\lambda x. t \mid t s$	function abstraction/app.
$\mu f. \lambda x. t$	recursive function

Fig. 2. Types and terms of our language

C. Denotational semantics

We give a denotational semantics to our language using ω PAP spaces and maps. A program $\Gamma \vdash t : \tau$ is interpreted as an ω PAP map $\llbracket t \rrbracket : \llbracket \Gamma \rrbracket \rightarrow \mathbf{L}\llbracket \tau \rrbracket$, for a suitable partiality monad \mathbf{L} that we now define.

Given an ω PAP space Y , we define the ω PAP space $\mathbf{L}Y$ as follows.

- The underlying set is $|\mathbf{L}Y| := |Y| \sqcup \{\perp\}$.
- The order structure is given by $\forall x \in |Y|, \perp \leq_{\mathbf{L}Y} x$ and $\forall x, x' \in |Y|, x \leq_{\mathbf{L}Y} x'$ iff $x \leq_Y x'$.
- Plots are given by

$$\mathcal{P}_{\mathbf{L}Y}^A := \{g : |A| \rightarrow |Y| \sqcup \{\perp\} \mid \exists B \subseteq A \text{ c-analytic}; \\ g^{-1}(|Y|) = \text{Im}(|B|) \text{ and } g|_{\text{Im}(|B|)} \in \mathcal{P}_Y^B\}$$

The intuition for $\mathcal{P}_{\mathbf{L}Y}^A$ is that the *partial* ω PAP maps from Euclidean space into Y are defined on c-analytic sets of inputs, but the region on which they are *not* defined need not satisfy any special property. This is how we account for the example from Fig. 1; although it is undefined on the $\frac{1}{3}$ -Cantor set (which is not c-analytic), its domain (the complement of the $\frac{1}{3}$ -Cantor set) is c-analytic.

We also have (natural) ω PAP-morphisms, that we can write using lambda-calculus notation as

- $\text{return}_X : X \rightarrow \mathbf{L}X$ given by $\lambda x. \text{inl } x$
- $\gg_{X,Y} : \mathbf{L}X \times (X \Rightarrow \mathbf{L}Y) \rightarrow \mathbf{L}Y$ given by $\lambda(x, g). \text{match } x \text{ with } \text{inr } \perp \mapsto \text{inr } \perp, \text{inl } x \mapsto g(x)$

where **inl**, **inr** are respectively left and right injections (as set functions, they are not ω PAP-morphisms).

\mathbf{L} extends to ω PAP maps $f : X \rightarrow Y$ by setting $\mathbf{L}f : \mathbf{L}X \rightarrow \mathbf{L}Y$ to equal **inl** $\circ f$ on all inputs **inl** x and setting $\mathbf{L}f(\text{inr } \perp) = \text{inr } \perp$. This definition is inspired by similar constructs present in [2, 4]. Overall, $(\mathbf{L}, \text{return}, \gg)$ forms a (commutative) monad.

Proposition II.1. $(\mathbf{L}, \text{return}, \gg)$ is a (strong, commutative) monad on the category of ω PAP-spaces and morphisms.

We can now interpret types and terms of our language. The semantics of types and terms is given in Figure 3. A context $\Gamma = (x_1 : \tau_1, \dots, x_n : \tau_n)$ is interpreted as the product space $\llbracket \Gamma \rrbracket \stackrel{\text{def}}{=} \prod_{i=1}^n \llbracket \tau_i \rrbracket$. Substitution is given by the usual Kleisli composition of effectful programs, as is standard for effectful functional languages [14].

$\llbracket \text{real} \rrbracket$	$\stackrel{\text{def}}{=} (\mathbb{R}, \mathcal{P}_{\mathbb{R}}, =)$
$\llbracket \mathbb{B} \rrbracket$	$\stackrel{\text{def}}{=} (1 + 1, \mathcal{P}_{1+1}, =)$
$\llbracket \langle \tau_1, \tau_2 \rangle \rrbracket$	$\stackrel{\text{def}}{=} \llbracket \tau_1 \rrbracket \times \llbracket \tau_2 \rrbracket$
$\llbracket \tau_1 \rightarrow \tau_2 \rrbracket$	$\stackrel{\text{def}}{=} \llbracket \tau_1 \rrbracket \Rightarrow \mathbf{L}\llbracket \tau_2 \rrbracket$
$\llbracket x \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \text{return } \rho(x)$
$\llbracket c \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \text{return } c$
$\llbracket f(t_1 \dots t_n) \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \llbracket t_1 \rrbracket(\rho) \gg (\lambda x_1. \llbracket t_2 \rrbracket(\rho) \gg \dots \gg (\lambda x_n. f(x_1, \dots, x_n)))$
$\llbracket \langle t_1, t_2 \rangle \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \llbracket t_1 \rrbracket(\rho) \gg (\lambda x. \llbracket t_2 \rrbracket(\rho) \gg (\lambda y. \text{return } (x, y)))$
$\llbracket \lambda x : \tau. t \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \text{return } \lambda v. \llbracket t \rrbracket(\rho[x \mapsto v])$
$\llbracket \text{case } t_1 \text{ of } \langle x_1, x_2 \rangle \rightarrow t_2 \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \llbracket t_1 \rrbracket(\rho) \gg (\lambda v. \llbracket t_2 \rrbracket(\rho[x \mapsto v]))$
$\llbracket t_1 t_2 \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \llbracket t_1 \rrbracket(\rho) \gg (\lambda f. \llbracket t_2 \rrbracket(\rho) \gg f)$
$\llbracket \text{if } t_1 \text{ then } t_2 \text{ else } t_3 \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \llbracket t_1 \rrbracket(\rho) \gg \lambda b. \text{if } b \text{ then } \llbracket t_2 \rrbracket(\rho) \text{ else } \llbracket t_3 \rrbracket(\rho)$
$\llbracket \mu f. \lambda x. t \rrbracket(\rho)$	$\stackrel{\text{def}}{=} \text{return } \bigvee_{i \in \mathbb{N}} f_i$
where $f_0 = \lambda x. \text{inr } \perp, f_{i+1} = \lambda v. \llbracket t \rrbracket(\rho[f \mapsto f_i, x \mapsto v])$	

Fig. 3. Denotational semantics of types and terms in ω PAP. Blue is used for syntax and bold for mathematical functions.

Our denotational semantics in ω PAP is sound and adequate w.r.t. our (standard) operational semantics (Appendix A-B).

Theorem II.2. *The denotational semantics of our language is sound and adequate.*

Proof sketch. We show that ω PAP spaces and morphisms are a category of ω -concrete sheaves on a concrete site, as defined in Matache et al. [1], and use their development. More details are given in Appendix B. \square

III. REASONING ABOUT DIFFERENTIABLE PROGRAMS

Our first application of our semantics is to the problem of characterizing the behavior of *automatic differentiation* on the terms of our language. Automatic differentiation is a program transformation intended to convert programs representing real-valued functions into programs representing their derivatives. When all programs denote smooth functions, this informal specification can easily be made precise; for example, Figure 4 illustrates the correctness property that Huot et al. [3] prove for forward-mode AD in a higher-order language with smooth primitives. Their proof is elegant and compositional, relying on logical relations defined over the *denotations* of open terms.

In our setting, not all programs denote smooth functions, and even when they do, standard AD algorithms can return incorrect results, in ways that depend on exactly how the smooth function in question was expressed as a program. For example, Mazza and Pagani [7] define a program `SillyId` = $\lambda x : \mathbb{R}. \text{if } x = 0 \text{ then } 0 \text{ else } x$, that denotes the identity function (with derivative $\lambda x.1$), but for which AD incorrectly computes a derivative of 0 when $x = 0$. Because two extensionally equal programs can have *different* derivatives as computed by AD, it may seem unclear how any proof strategy based on Figure 4, relating the syntactic operation of AD to the semantic operation of derivative-taking, could apply. Indeed, in showing their correctness result for AD, Mazza and Pagani [7] had to develop new operational techniques for reasoning about derivatives of *traces* of recursive, branching programs.

Our ω PAP semantics lets us take a different route, illustrated in Figure 5. Like Huot et al. [3], we work directly with the denotations of terms, which in our setting, are ω PAP maps. But we do not attempt to assign a unique *derivative* to each ω PAP function; instead, we define a *relation* on ω PAP maps that characterizes when one is an *intensional derivative* [5] of another. Our correctness proof, which follows exactly the structure of Huot et al. [3]’s, then establishes that AD produces a program denoting *an* (not *the*) intensional derivative of the original program’s denotation (Theorem III.3).

All ω PAP maps have intensional derivatives, so there is no need to restrict the correctness result to only the differentiable functions. But when an ω PAP map *is* differentiable in the standard sense, its intensional derivatives agree almost everywhere with its true derivative. Thus, reasoning denotationally, we are able to recover Mazza and Pagani [7]’s result: AD is almost everywhere correct. Interestingly, this almost-everywhere correctness result is not sufficient to prove the convergence of randomly initialized gradient descent, even for programs with differentiable denotations (Proposition III.5).

A. Correctness of Automatic Differentiation

Terms in our language denote ω PAP maps $X \rightarrow \mathbf{LY}$, and so to speak of *correct AD* in our language, we need some notion of derivative that applies to such maps. We begin by recalling Lee et al. [5]’s notion of *intensional derivative*, then lift it to our setting:

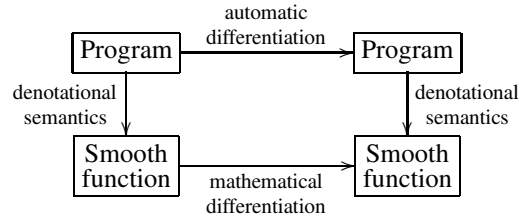


Fig. 4. Common approach to the correctness of AD, e.g. [3].

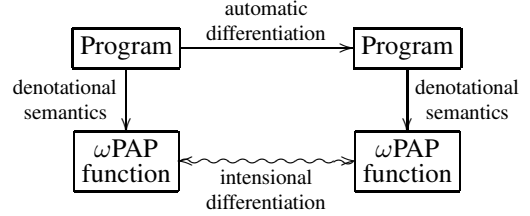


Fig. 5. Our approach to the correctness of AD.

Definition III.1 (intensional derivative [5]). *Let $f : U \rightarrow V$ be an ω PAP map, with c -analytic domain $U \subseteq \mathbb{R}$ and $V \subseteq \mathbb{R}$ (seen as ω PAP spaces). Then an ω PAP map $g : U \rightarrow \mathbb{R}$ is an intensional derivative of f if there exists a countable family $\{(A_i, f_i)\}_{i \in I}$ such that:*

- the sets A_i are analytic and form a partition of U ;
- the functions $f_i : U_i \rightarrow \mathbb{R}$ are analytic with open domain $U_i \supseteq A_i$; and
- for $x \in A_i$, $f(x) = f_i(x)$ and $g(x) = \frac{d}{dx} f_i(x)$.

Definition III.2 (lifted intensional derivative). *Let $f : U \rightarrow \mathbf{LV}$ be an ω PAP map, with c -analytic domain $U \subseteq \mathbb{R}$ and $V \subseteq \mathbb{R}$ (seen as ω PAP spaces). Then an ω PAP map $g : U \rightarrow \mathbf{LR}$ is a lifted intensional derivative of f if it is defined on exactly the inputs that f is, and restricted to this common domain, g is an intensional derivative of f .*

(These definitions can be straightforwardly extended to the multivariate case, where $U \subseteq \mathbb{R}^n$ and $V \subseteq \mathbb{R}^m$, and our full correctness theorem does cover such functions. But here we demonstrate the reasoning principles in the simpler univariate setting. Of course, a program of type `real` \rightarrow `real` may still be defined in terms of multivariate primitives.)

We now establish that AD, applied to terms with a free parameter $\theta : \text{real} \vdash t : \text{real}$, computes these lifted intensional derivatives. It does so in two steps:

- First, we apply a macro \mathcal{D} (defined in Figure 6) to t , yielding a term $\theta : \text{real} \times \text{real} \vdash \mathcal{D}(t) : \text{real} \times \text{real}$. This new term operates not on a single real value, but on a *dual number*, which intuitively stores both a value and its (intensional) derivative with respect to an external parameter.
- Then, $\theta : \text{real} \vdash \text{let } \theta = \langle \theta, 1 \rangle \text{ in case } \mathcal{D}(t) \text{ of } \langle x, \frac{dx}{d\theta} \rangle \rightarrow \frac{dx}{d\theta}$ is output as the lifted intensional derivative of $\llbracket t \rrbracket$.

$\mathcal{D}(\text{real})$	$\stackrel{\text{def}}{=} \text{real} \times \text{real}$	$\mathcal{D}(\tau_1 \times \tau_2)$	$\stackrel{\text{def}}{=} \mathcal{D}(\tau_1) \times \mathcal{D}(\tau_2)$
$\mathcal{D}(\mathbb{B})$	$\stackrel{\text{def}}{=} \mathbb{B}$	$\mathcal{D}(\tau_1 \rightarrow \tau_2)$	$\stackrel{\text{def}}{=} \mathcal{D}(\tau_1) \rightarrow \mathcal{D}(\tau_2)$
$\mathcal{D}(x)$	$\stackrel{\text{def}}{=} x$		
$\mathcal{D}(\underline{c} : \text{real})$	$\stackrel{\text{def}}{=} \langle \underline{c}, 0 \rangle$		
$\mathcal{D}(\underline{c} : \mathbb{B})$	$\stackrel{\text{def}}{=} \underline{c}$		
$\mathcal{D}(f(t_1, \dots, t_n))$	$\stackrel{\text{def}}{=} f_{\mathcal{D}}(\mathcal{D}(t_1), \dots, \mathcal{D}(t_n))$		
$\mathcal{D}(\lambda x. t)$	$\stackrel{\text{def}}{=} \lambda x. \mathcal{D}(t)$		
$\mathcal{D}(t \ s)$	$\stackrel{\text{def}}{=} \mathcal{D}(t) \ \mathcal{D}(s)$		
$\mathcal{D}(\langle t_1, t_2 \rangle)$	$\stackrel{\text{def}}{=} \langle \mathcal{D}(t_1), \mathcal{D}(t_2) \rangle$		
$\mathcal{D}(\text{case } t \text{ of } \langle x_1, x_2 \rangle \rightarrow s)$	$\stackrel{\text{def}}{=} \text{case } \mathcal{D}(t)$		
	$\text{of } \langle x_1, x_2 \rangle \rightarrow \mathcal{D}(s)$		
$\mathcal{D}(\mu f. t)$	$\stackrel{\text{def}}{=} \mu f. \mathcal{D}(t)$		
$\mathcal{D}(\text{if } t_1 \text{ then } t_2 \text{ else } t_3)$	$\stackrel{\text{def}}{=} \text{if } \mathcal{D}(t_1) \text{ then } \mathcal{D}(t_2)$		
	$\text{else } \mathcal{D}(t_3)$		

Fig. 6. AD macro for our higher-order recursive language

The key hurdle in showing the correctness of this AD procedure is establishing that the macro \mathcal{D} is correctly propagating intensional dual numbers. To show this compositionally, and in a way that exploits our denotational framework, we first define for each type τ a relation $V(\tau)$ between pairs of ω PAP plots in $\llbracket \tau \rrbracket$ and in $\llbracket \mathcal{D}(\tau) \rrbracket$. It captures more precisely the properties that the dual numbers flowing through our program should have.

Definition III.3 (correct dual-number intensional derivative at $A \subseteq \mathbb{R}$). *Let τ be a type and let $f : A \rightarrow \llbracket \tau \rrbracket$, for some c-analytic subset $A \subseteq \mathbb{R}$. Then we say $f' : A \rightarrow \llbracket \mathcal{D}(\tau) \rrbracket$ is a correct dual-number intensional derivative of f if $(f, f') \in V_{\tau}(A)$, where $V_{\tau}(A)$ is defined inductively:*

- $V_{\text{real}}(A) = \{(f, f') \mid f'(x) = (f(x), g(x)) \text{ for some intensional derivative } g \text{ of } f\}$.
- $V_{\mathbb{B}}(A) = \{(f, f') \mid f = f'\}$.
- $V_{\tau_1 \times \tau_2}(A) = \{(f, f') \mid (\pi_i \circ f, \pi_i \circ f') \in V_{\tau_i}(A) \text{ for } i = 1, 2\}$.
- $V_{\tau_1 \rightarrow \tau_2}(A) = \{(f, f') \mid \forall (g, g') \in V_{\tau_1}(A), (\lambda x. f(x)(g(x)), \lambda x. f'(x)(g'(x))) \in V_{\tau_2}(A)\}$.

Readers may recognize V as defining a (semantic) logical relation; its definition at product and function types is completely standard, so only at ground types did we make any real choices. For partial functions, we need to extend the definition:

Definition III.4 (correct lifted dual-number intensional derivative at $A \subseteq \mathbb{R}$). *Let τ be a type and let $f : A \rightarrow \mathbb{L}\llbracket \tau \rrbracket$, for some c-analytic subset $A \subseteq \mathbb{R}$. Then we say $f' : A \rightarrow \mathbb{L}\llbracket \mathcal{D}(\tau) \rrbracket$ is a correct lifted dual-number intensional derivative of f if f and f' are defined (i.e., not \perp) on the same domain and, restricted to this common domain, f' is a correct dual-number intensional derivative of f .*

We are now in a position to state more precisely what it would mean for the \mathcal{D} macro to correctly propagate dual numbers. In particular, the macro transforms terms $\Gamma \vdash t : \tau$ into their *correct dual-number translations* $\mathcal{D}(\Gamma) \vdash \mathcal{D}(t) : \mathcal{D}(\tau)$, the specification for which is given below.

Definition III.5 (correct dual-number translation). *Let τ_1 and τ_2 be types, and let $f : \llbracket \tau_1 \rrbracket \rightarrow \mathbb{L}\llbracket \tau_2 \rrbracket$. Then $f_{\mathcal{D}} : \llbracket \mathcal{D}(\tau_1) \rrbracket \rightarrow \mathbb{L}\llbracket \mathcal{D}(\tau_2) \rrbracket$ is a correct dual-number translation of f if, for all c-analytic A , and all pairs $(g, g') \in V_{\tau_1}(A)$, $f_{\mathcal{D}} \circ g'$ is a correct lifted dual-number intensional derivative of $f \circ g$.*

We assume that each primitive $f : \tau_1 \rightarrow \tau_2$ in our language is equipped with a *built-in* correct dual-number translation $f_{\mathcal{D}}$, which the definition of \mathcal{D} in Figure 6 uses. Because the primitives are translated correctly, so are whole programs:

Lemma III.1 (Correctness of \mathcal{D} macro (limited)). *For any term $\Gamma \vdash t : \tau$, $\llbracket \mathcal{D}(t) \rrbracket : \llbracket \mathcal{D}(\Gamma) \rrbracket \rightarrow \llbracket \mathcal{D}(\tau) \rrbracket$ is a correct dual-number translation of $\llbracket t \rrbracket$.*

Proof Sketch. This is the Fundamental Lemma of our logical relations proof, and can be obtained using general machinery [15], after verifying several properties of our particular setting:

- $V_{\text{real}}(A)$ is closed under *piecewise gluing*: if $\{A_i\}_{i \in I}$ is a countable partition of A , and $(f|_{A_i}, f'|_{A_i}) \in V_{\text{real}}(A_i)$ for each $i \in I$, then $(f, f') \in V_{\text{real}}(A)$. This follows from our definition of intensional derivative.
- $V_{\text{real}}(A)$ is closed under *restriction*: if $(f, f') \in V_{\text{real}}(A)$, and $A' \subseteq A$ is c-analytic, then $(f|_{A'}, f'|_{A'}) \in V_{\text{real}}(A')$. This also follows easily from our definition.
- The lifted dual-number intensional derivatives are closed under *least upper bounds*: if $\{(f_n, f'_n)\}_{n \in \mathbb{N}}$ is a sequence with each $f'_n : A \rightarrow \mathbb{L}\llbracket \mathbb{R} \times \mathbb{R} \rrbracket$ a correct lifted dual-number intensional derivative of $f : A \rightarrow \mathbb{L}\mathbb{R}$, and with $f_n \leq f_{n+1}$ and $f'_n \leq f'_{n+1}$ for each $n \in \mathbb{N}$, then $\bigvee_{n \in \mathbb{N}} f'_n$ is a correct lifted dual-number intensional derivative of $\bigvee_{n \in \mathbb{N}} f_n$. This is more involved, but its proof mirrors the one that establishes $\mathbb{L}\mathbb{R}$ as an ω PAP space in the first place: we explicitly construct representations of the lubs in question as piecewise functions, satisfying Definition III.2.

□

The overall correctness theorem follows easily:

Theorem III.2 (Correctness of AD Algorithm (limited)). *For any term $\theta : \text{real} \vdash t : \text{real}$, the ω PAP map $\lambda \theta. \llbracket \text{case } \mathcal{D}(t) \text{ of } \langle x, \frac{dx}{d\theta} \rangle \rightarrow \frac{dx}{d\theta} \rrbracket((\theta, 1))$ is a lifted intensional derivative of $\llbracket t \rrbracket$.*

Proof. We apply Lemma III.1, setting (g, g') from Definition III.5 to $(\lambda \theta. \theta, \lambda \theta. (\theta, 1))$. This implies that $(\llbracket t \rrbracket, \lambda \theta. \llbracket \mathcal{D}(t) \rrbracket(\theta, 1))$ are defined on the same domain, and on that domain A , their restrictions are in $V_{\text{real}}(A)$. This implies that on that domain, $\lambda \theta. \pi_2(\llbracket \mathcal{D}(t) \rrbracket((\theta, 1)))$ is an intensional derivative of $\llbracket t \rrbracket$, from which the result follows by unfolding the definition of $\llbracket \cdot \rrbracket$ on **case** expressions. □

This result, which we achieved via denotational reasoning, immediately implies the result of Mazza and Pagani [7]:

Corollary III.2.1. *If $\theta : \text{real} \vdash t : \text{real}$ denotes a total differentiable function, then the AD algorithm fails at a measure-zero set of inputs.*

Proof. Because $\llbracket t \rrbracket$ is total, it is PAP and AD computes an intensional derivative of $\llbracket t \rrbracket$. Intensional derivatives are almost everywhere equal to true derivatives [5]. \square

Using more complex logical predicates, we can prove correctness for multivariate functions (see Appendix C). In the multivariate setting, forward-mode AD typically computes *Jacobian-vector products*; for us, it computes lifted *intensional* Jacobian-vector products.

Definition III.6 (lifted intensional Jacobian). *Let $f : U \rightarrow \mathbf{LV}$ be an ω PAP map, with $U \subseteq \mathbb{R}^n$ c -analytic and $V \subseteq \mathbb{R}^m$. An ω PAP map $g : U \rightarrow \mathbf{LR}^{m \times n}$ is an intensional Jacobian of f if it is defined (i.e., not \perp) exactly when f is, there exists a countable analytic partition $\{A_i\}_{i \in I}$ of its domain, and there exist analytic functions $f_i : U_i \rightarrow \mathbb{R}^m$ with open domains $U_i \supseteq A_i$ such that when $x \in A_i$, $f(x) = f_i(x)$ and $g(x) = Jf_i(x)$, where J is the Jacobian operator.*

Theorem III.3 (Correctness of AD (full)). *For any term $x_1 : \text{real}, \dots, x_n : \text{real} \vdash t : \text{real} \times \dots \times \text{real}$, and any vector $\mathbf{v} \in \mathbb{R}^n$, the ω PAP map $\lambda\theta.\text{let } \mathbf{x} = \llbracket \mathcal{D}(t) \rrbracket((\theta_1, v_1), \dots, (\theta_n, v_n))$ in $(\pi_2 x_1, \dots, \pi_2 x_m)$ is equal to $g(\theta) \cdot \mathbf{v}$ for some intensional Jacobian g of $\llbracket t \rrbracket$.*

B. Optimization of Differentiable Programs with AD-computed Derivatives

Our characterization of AD’s behavior, proven in the previous section, can be used to establish the correctness of algorithms that use the results of AD. In optimization, for example, a very common technique for minimizing a differentiable function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is gradient descent. Starting at an initial point $x^0 \in \mathbb{R}^d$, for a set step size $\epsilon > 0$, we iterate the following update:

$$x^{t+1} := x^t - \epsilon \nabla f(x^t). \quad (1)$$

If the step size ϵ is too large, we cannot hope to converge on an answer. To ensure that a small-enough step size can be found, we need to know that f ’s gradient changes slowly:

Definition III.7. *A differentiable function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is L -smooth if for all $x, y \in \mathbb{R}^d$*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|.$$

When f is L -smooth, it is well known that ϵ can be chosen small enough for gradient descent to converge (albeit not necessarily to a global minimum):

Theorem III.4 (e.g. [16]). *Let f be L -smooth, bounded below, and $0 < \epsilon < \frac{2}{L}$. Then, for any initial $x^0 \in \mathbb{R}^d$, following the method from Equation 1, the gradient of f tends to 0:*

$$\lim_{t \rightarrow \infty} \nabla f(x^t) = 0$$

and the function f monotonically decreases on values of the sequence $\{x^t\}_t$, that is for all $t \in \mathbb{N}$, $f(x^{t+1}) \leq f(x^t)$.

However, this theorem assumes that the true gradient of f is used to make each update. When gradients are obtained by AD, this may not be the case, even if f is differentiable: we proved only that AD computes *intensional derivatives*, which may disagree with true derivatives on a measure-zero set of inputs.

Intuitively, it may seem as though we can avoid the measure-zero set of inputs on which AD may be incorrect by *randomly* selecting an initial point x^0 . Indeed, similar arguments have been made informally in the literature [7]. But this turns out not to be the case:

Proposition III.5. *Let $\epsilon > 0$. There exists a program P such that $\llbracket P \rrbracket : \mathbb{R} \rightarrow \mathbb{R}$ satisfies the condition of Theorem III.4, and yet, for all $x^0 \in \mathbb{R}$, the gradient descent method diverges to $-\infty$, when the gradients are computed with AD.*

Proof sketch. Let P be the closed program defined by the following code, where lr is our chosen learning rate ϵ .

```
# Recursive helper
def g(x, n):
    if x > 0:
        return ((x - n)*(x - n))/(lr * 2)
    if x == 0:
        return x / lr + n*n / (2*lr)
    else:
        return g(x+1, n+1)

def P(x):
    return g(x, 0)
```

One can show that $\llbracket P \rrbracket = x \mapsto \frac{x^2}{2\epsilon}$ and satisfies the conditions of Theorem III.4, yet gradient descent will diverge to $-\infty$ for all initial points x^0 . The full explanation is given in Appendix D-A. \square

The proof applies to an idealized setting where gradient descent is run with real numbers, rather than floating-point numbers, but in our experiments, PyTorch did diverge on this program. Similar counterexamples can be constructed for some variants of the gradient descent algorithm, e.g. when the learning rate is changing according to a schedule.

Likewise, it is easy to derail the algorithm when the learning rate ϵ is random but the initial point x^0 is fixed.

Proposition III.6. *Let x^0 be a fixed initial point. Then there exists a program P such that $\llbracket P \rrbracket : \mathbb{R} \rightarrow \mathbb{R}$ satisfies the conditions of Theorem III.4, and $\llbracket P \rrbracket'(x^0) \neq 0$, and yet, for all $\epsilon > 0$, the gradient descent method yields $x^t = x^0$ for all t , when the gradients are computed with AD.*

Proof. Simply chose any x^0 for which the intensional derivative is 0, but for which the true derivative is non-zero. \square

The counterexamples constructed for the proofs above are tied to a specific learning rate or to a specific initialization. It is therefore reasonable to think that by randomizing *both* quantities, one might be able to almost surely avoid such counter-examples. Thankfully, this is the case.

$$\frac{\Gamma \vdash e : \mathbb{R}}{\Gamma \vdash \text{score } e : 1} \quad \frac{}{\Gamma \vdash \text{sample} : \mathbb{R}}$$

Fig. 7. Type system of the probabilistic part of the language

Theorem III.7 (Convergence of GD with intensional derivatives). *Let f be PAP, bounded below, L -smooth. Let (ϵ, x_0) be drawn randomly from a continuous probability distribution supported on $(0, \frac{2}{L}) \times \mathbb{R}^d$. Then when gradient descent computes gradients using AD, the true gradient tends to 0 with probability 1:*

$$\lim_{t \rightarrow \infty} \nabla f(x^t) = 0.$$

Furthermore, with probability 1, the function $f(x)$ monotonically decreases: $f(x^{t+1}) \leq f(x^t)$.

The proof is given in Appendix D-B.

IV. REASONING ABOUT PROBABILISTIC PROGRAMS

PAP functions were originally developed in Lee et al. [5] to reason about automatic differentiation, so perhaps it is not surprising that their generalization to ω PAP was useful, in the previous section, for characterizing AD’s behavior. In this section, we show that the ω PAP semantics *also* makes it nice to reason about *probabilistic* programs: by excluding pathological deterministic primitives, we also prevent many pathologies from arising in the probabilistic setting, enabling clean denotational arguments for several nice properties.

To reason about probabilistic programs, we must extend our core calculus (Figure 2) with the standard effectful operations exposed by probabilistic programming languages (PPLs): **sample** samples a real from the uniform distribution on $(0, 1)$, and **score** r reweights program traces (to implement soft constraints). Their typing rules are given in Figure 7.

The language needs no new types, but the semantics change: instead of interpreting our language with the monad \mathbf{L} , which models divergence, we must use a new monad that also tracks probabilistic computation. In Section IV-A, we introduce a monad of *weighted samplers* [8], which can informally be seen as deterministic functions from a source of randomness to output values. We use this monad to reason about the deterministic *weight functions* that some probabilistic programming systems use during inference, presenting a new (much simplified) proof of Mak et al. [9]’s result that almost-surely-terminating probabilistic programs have almost-everywhere differentiable weight functions. In Section IV-B, we introduce a commutative monad of measures, which can be seen as a smaller version of Vákár et al. [2]’s monad of measures, excluding those measures which can’t be defined using ω PAP deterministic primitives. We use this smaller monad to prove that all definable measures in our language are supported on countable unions of smooth manifolds. This implies that they have densities with respect to a particular well-behaved class of base measures, and we close by briefly discussing the implications for PPL designers.

A. Almost-Everywhere Differentiability of Weight Functions

First, we define a monad \mathbf{S} of *weighted samplers*, that interprets programs as partial ω PAP maps from a space Ω of random seeds to a space of weighted values. Intuitively, the measure represented (or “targeted”) by a sampler of weighted pairs (x, w) is the one that assigns measure $\mathbb{E}[w \cdot 1_A(x)]$ to each measurable set A . But this semantics does *not* validate many important program equivalences, like commutativity, intuitively because it exposes “implementation details” of weighted samplers, that can differ even for samplers targeting the same measure. This is intentional: our first goal is to prove a property of these “implementation details,” one that inference algorithms automated by PPLs might rely on.

First, we fix our source of randomness Ω to be the lists of reals, $\sqcup_{i \in \mathbb{N}} \mathbb{R}^i$ (which is an ω PAP space, see Example II.3). We then define our monad:

Definition IV.1 (monad of weighted samplers). *Let \mathbf{S} be the monad $(\mathbf{S}, \text{return}^{\mathbf{S}}, \gg^{\mathbf{S}})$, defined as follows:*

- $\mathbf{S}X := \Omega \Rightarrow \mathbf{L}((X + 1) \times [0, \infty) \times \Omega)$
- $\text{return}^{\mathbf{S}}_X := \lambda x. \lambda r. \text{inl}(\text{inl } x, 1, r)$
- $\gg^{\mathbf{S}}_{X,Y} := \lambda m. \lambda k. \lambda r. m(r) \gg^{\mathbf{L}} (\lambda (x?, w, r'). \text{match } x? \text{ with } \{\text{inr } _ \rightarrow \text{inl}(x?, 0.0, r') \mid \text{inl } x \rightarrow k(x)(r') \gg^{\mathbf{L}} \lambda (y?, v, r''). \text{inl}(y?, w \cdot v, r'')\})$.

$$\begin{aligned} \llbracket \text{sample} \rrbracket(\rho) &\stackrel{\text{def}}{=} \lambda r. \text{match } r \text{ with } \{ \\ &\quad () \rightarrow \text{inl}(\text{inr } \star, 0, r) \\ &\quad \mid (r_1, r_{2:n}) \rightarrow \text{inl}(\text{inl } r_1, 1, r_{2:n}) \} \\ \llbracket \text{score } t \rrbracket(\rho) &\stackrel{\text{def}}{=} \llbracket t \rrbracket(\rho) \gg (\lambda w. \lambda r. \text{inl}(\text{inl } \star, 0 \vee w, r)) \\ \llbracket \mu f. \lambda x. t \rrbracket(\rho) &\stackrel{\text{def}}{=} \text{return } \bigvee_{i \in \mathbb{N}} f_i \\ \text{where } f_0 &= \lambda x. \lambda r. \text{inr } \perp, f_{i+1} = \lambda v. \llbracket t \rrbracket(\rho[f \mapsto f_i, x \mapsto v]) \end{aligned}$$

Fig. 8. Updated semantics for new monad \mathbf{S} of effects

This monad interprets a probabilistic program as a partial function mapping lists of real-valued random samples, called traces, to: (1) the output values they possibly induce in X (or $\text{inr } \star$ if the input trace does not contain enough samples to complete execution of the program), (2) a *weight* in $[0, \infty)$, and (3) a remainder of the trace, containing any samples not yet used. The **return** of the monad consumes no randomness, and successfully returns its argument with a weight of 1. The \gg of the monad runs its continuation on the random numbers remaining in the trace after its first argument has executed, and multiplies the weights from both parts of the computation. Figure 8 gives the semantics of **sample** and **score**, as well as updated semantics for $\mu f. t$ (the only change is that the bottom element, from which least fixed points are calculated, is now the bottom element of $\mathbf{S}X$ rather than $\mathbf{L}X$). The **sample** command fails to produce a value when given an empty trace, but otherwise consumes the first value in the trace and returns it. The **score** command consumes no randomness, and uses its argument (if non-negative) as the weight.

One way to use a weighted sampler is to run it, continually providing longer and longer lists of uniform random numbers until a value from X (and an associated weight) is generated. But many probabilistic programming systems also expose more sophisticated inference algorithms. *Gradient-based* methods, such as Hamiltonian Monte Carlo or Langevin ascent, attempt to find executions with high weights by performing hill-climbing optimization or MCMC over the space of traces, guided by a program's *weight function*.

Definition IV.2 (weight function). *For any $p \in |\mathbf{S} X|$, define its weight function $w_p : \sqcup_{i \in \mathbb{N}} [0, 1]^i \rightarrow [0, \infty)$, as follows:*

- When $p(r) = \mathbf{inr} \perp$, $w_p(r) = 0$.
- When $p(r) = \mathbf{inl} (x, v, r')$ for r' non-empty, $w_p(r) = 0$.
- When $p(r) = \mathbf{inl} (x, v, ())$, $w_p(r) = v$.

The weight function can be viewed as a density with respect to a particular base measure on traces; scaling the base measure by this density yields an unnormalized measure, and the normalized version of this measure, which is a probability distribution placing high mass on traces that lead to high weights, is called the *posterior* over traces. Inference algorithms like HMC are designed to generate samples approximately distributed according to this posterior.

Using our definition of the weight function, we can recover Mak et al. [9]'s result about the almost-everywhere differentiability of weight functions, a convenient property when reasoning about gradient-based inference:

Theorem IV.1. *Probabilistic programs that almost surely halt have almost-everywhere differentiable weight functions.*

Proof. Let $t : X$ be a program. Almost-sure termination implies that for almost all inputs $r \in \sqcup_{i \in \mathbb{N}} [0, 1]^i$, $\llbracket t \rrbracket(r) \neq \mathbf{inr} \perp$. Restricted to $\llbracket t \rrbracket$'s domain (which is almost all of $\sqcup_{i \in \mathbb{N}} [0, 1]^i$), its weight function $w_{\llbracket t \rrbracket}$ is ω PAP, and thus almost-everywhere differentiable. Almost all of almost all of $\sqcup_{i \in \mathbb{N}} [0, 1]^i$ is almost all of $\sqcup_{i \in \mathbb{N}} [0, 1]^i$, concluding the proof. \square

We note that the proof only uses the fact that for almost all $r \in \sqcup_{i \in \mathbb{N}} [0, 1]^i$, $\llbracket t \rrbracket(r) \neq \mathbf{inr} \perp$ —which is strictly weaker than almost-sure termination. Almost-sure termination can fail even though a program halts on any finite trace of random numbers. Some probabilistic context-free grammars, for example, do not always surely halt, but as they execute, they consume an unbounded amount of randomness, so that when provided a finite trace, they will eventually halt with an error. Our proof shows that such programs, although they do not almost surely halt, do have almost-everywhere differentiable weight functions. As such, our result is slightly stronger than that of Mak et al. [9], and followed immediately from the interpretation of probabilistic programs in the ω PAP category.

B. Existence of Convenient Base Measures for Monte Carlo

We now wish to prove a result not about the intensional properties of probabilistic programs (like their weight functions), but the extensional properties of the measures that programs denote. To do so, we construct a final monad in

which to interpret our language: a commutative monad of measures, similar to that of Vákár et al. [2].

Definition IV.3 (ω PAP space of measure weights). *Define \mathbb{W} to be the ω PAP space with:*

- $|\mathbb{W}| = \mathbb{R}_{\geq 0} \cup \{\infty\}$, the extended non-negative reals;
- $w \leq_{\mathbb{W}} v$ when $v = \infty$ or when $w < v < \infty$.
- $\phi \in \mathcal{P}_{\mathbb{W}}^A$ whenever $\phi : A \rightarrow \mathbb{W}$ is measurable.

Note that although many maps *into* \mathbb{W} exist, very few maps exist *out of* \mathbb{W} , because its ω PAP diffeology is so permissive: any measurable function is a plot.

Following Ścibior et al. [8], measures can be seen as equivalence classes of samplers that behave the same way as *integrators*. Given a weighted sampler $p \in |\mathbf{S} X|$, we can define the *integrator* it represents:

Definition IV.4 (integrator associated to a sampler). *We define ω PAP maps $\mathbf{Int}_X : \mathbf{S} X \rightarrow (X \Rightarrow \mathbb{W}) \Rightarrow \mathbb{W}$ sending samplers to their associated integrators: $\mathbf{Int}_X p = \lambda f. \int_{\text{Dom}(\alpha_p)} \Lambda_{\Omega}(dr) f(\alpha_p(r))$, where*

- Λ_{Ω} is a Lebesgue base measure over $|\Omega|$, assigning to each measurable subset $A \subseteq \Omega$ the value $\sum_{i \in \mathbb{N}} \Lambda^i(\{\mathbf{x} \mid (i, \mathbf{x}) \in A\})$ (where Λ^i is the Lebesgue measure on \mathbb{R}^i).
- $\alpha_p : \Omega \rightarrow \mathbf{L} X$ is an ω PAP map that on input r , returns $\mathbf{inr} \perp$ if any of the following hold: r is empty; $r_1 < 0$; $r_{2:|r|} \notin [0, 1]^{|r|-1}$; $p(r_{2:|r|})$ returns $\mathbf{inr} \perp$; or $p(r_{2:|r|})$ returns $\mathbf{inl} (x, v, r')$ and either r' is non-empty, $x = \mathbf{inr} \star$, or $v < r_1$. Otherwise, $p(r_{2:|r|})$ must return $(\mathbf{inl} x, _, _)$, and α_p also returns $\mathbf{inl} x$.
- $\text{Dom}(\alpha_p)$ is the domain of the partial map α_p .

Then, following Vákár et al. [2], we take the measures to be those integrators that lie in the image of \mathbf{Int} :

Definition IV.5 (monad of measures). *The monad \mathbf{M} of measures is defined by:*

- $\mathbf{M} X$ is the image of $\mathbf{S} X$ under \mathbf{Int}_X . It is a subobject of $(X \Rightarrow W) \Rightarrow W$ and inherits its order.
- $\mathbf{return}_X^{\mathbf{M}} := \lambda x. \lambda f. f(x)$.
- $\ggg_{X,Y}^{\mathbf{M}} := \lambda m. \lambda k. \lambda f. m(\lambda x. k(x)(f))$.

$$\begin{aligned} \llbracket \mathbf{sample} \rrbracket(\rho) &\stackrel{\text{def}}{=} \lambda f. \int_{[0,1]} f(x) dx \\ \llbracket \mathbf{score} \, t \rrbracket(\rho) &\stackrel{\text{def}}{=} \lambda f. \llbracket t \rrbracket(\rho)(\lambda w. (0 \vee w) \cdot f(\star)) \\ \llbracket \mu f. \lambda x. t \rrbracket(\rho) &\stackrel{\text{def}}{=} \mathbf{return} \bigvee_{i \in \mathbb{N}} f_i \\ \text{where } f_0 &= \lambda x. \lambda f. 0, f_{i+1} = \lambda v. \llbracket t \rrbracket(\rho[f \mapsto f_i, x \mapsto v]) \end{aligned}$$

Fig. 9. Updated semantics for new monad \mathbf{M} of effects

This monad is a submonad of the continuation monad [2], and so the \mathbf{return} and \ggg are inherited. The **sample** command is interpreted as the uniform measure on $[0, 1]$, and the **score** t command as a measure on a one-point space, with a certain total mass given by t . Our semantics under \mathbf{M} (Fig. 9) is related to the one from the previous section (Fig. 8) in that for any closed term $\vdash t : X$, $\mathbf{Int}_X(\llbracket t \rrbracket_{\mathbf{S}}) = \llbracket t \rrbracket_{\mathbf{M}}$. Using

this, we can establish the following lemma about the definable measures:

Lemma IV.2. *Let $\vdash t : \mathbf{real} \times \cdots \times \mathbf{real}$. Then, there exists a partial ω PAP function $f_t : \Omega \rightarrow \mathbb{R}^n$ such that $\llbracket t \rrbracket = f_{t*} \Lambda_\Omega$.*

That is, any probabilistic program returning reals must arise as a well-behaved transformation of “input randomness” represented by Ω . This is not particularly surprising, of course, because a probabilistic program is *defined* by specifying an (ω PAP) transformation of input randomness, but it highlights the way we will use our ω PAP semantics to establish general properties of the measures denoted by probabilistic programs.

In particular, we will characterize the definable measures as *absolutely continuous* with respect to a particular class of base measures. We first recall some relevant definitions from differential geometry, starting with the *smooth manifolds*, a well-studied generalization of Euclidean spaces; common examples include lines, spheres, and tori.

Definition IV.6. *A smooth manifold M is a second-countable Hausdorff topological space together with a smooth atlas: an open cover \mathcal{U} together with homeomorphisms $(\phi_U : U \rightarrow \mathbb{R}^n)_{U \in \mathcal{U}}$ called charts such that $\phi_V \circ \phi_U^{-1}$ is smooth on its domain of definition for all $U, V \in \mathcal{U}$. A function $f : M \rightarrow N$ between manifolds is smooth if $\phi_V \circ f \circ \phi_U^{-1}$ is smooth for all charts ϕ_U, ϕ_V of M and N . When there’s a global bound K on the local dimensions n of a smooth manifold, it’s well known by a theorem of Whitney that the manifold can be embedded as a submanifold of an Euclidean space, that is its topology is given by restricting the standard one from the surrounding Euclidean space.*

There is a natural measure on smooth manifolds that generalizes the Lebesgue measure on Euclidean spaces, and matches our intuitive notions of area and volume on curved spaces.

Definition IV.7. *The k -Hausdorff measure on a metric space (X, d) is defined as*

$$H^k(S) := \liminf_{\delta \rightarrow 0} \left\{ \sum_i \text{diam}(U_i)^k \mid S \subseteq \bigcup_i U_i, \text{diam}(U_i) < \delta \right\}$$

where $\text{diam}(U) := \sup \{d(x, y) \mid x, y \in U\}$ and $\text{diam}(\emptyset) := 0$. The Hausdorff dimension is then defined as $\dim_{\text{Haus}}(S) := \inf \{k \mid H^k(S) = 0\}$.

The Hausdorff measure computes the sizes of sets S in a dimension-dependent way:

- The 0-Hausdorff measure counts the points in S .
- The 1-Hausdorff measure sums the lengths of curves within a set. It will be infinite on surfaces, and 0 on sets of isolated points.
- The 2-dimensional Hausdorff measure sums the areas of surfaces, and will be infinite on volumes and 0 on curves. More generally, the k -Hausdorff measure will quantify k -dimensional volumes.

For smooth manifolds, the Hausdorff dimension matches the intuitive notion of dimension, e.g. a sphere in \mathbb{R}^3 has Hausdorff dimension 2, a path on such a sphere will have

Hausdorff dimension 1, and an open of \mathbb{R}^n has dimension n . We now define our class of *s-Hausdorff* measures, which measure the sizes of sets by measuring the d -dimensional volumes of their intersections with d -dimensional manifolds, for every d :

Definition IV.8 (s-Hausdorff measure on \mathbb{R}^n). *An s-Hausdorff measure on \mathbb{R}^n is a measure μ that decomposes as*

$$\mu(A) = \sum_{d=0}^n H^d(A \cap M^d)$$

where each M^d is a countable union $\bigcup_i M_i^d$ of d -dimensional smooth manifolds M_i^d .

Our main result in this section is a construction that assigns an s-Hausdorff measure $B(t)$ to every closed probabilistic program $\vdash t : \mathbf{real} \times \cdots \times \mathbf{real}$, such that $\llbracket t \rrbracket$ has a density w.r.t. $B(t)$. We write $\mu \ll \nu$ to mean that μ is absolutely continuous w.r.t. ν .

Theorem IV.3. *Let $\vdash t : \mathbf{real} \times \cdots \times \mathbf{real}$ be a closed probabilistic program. There exists an s-Hausdorff measure $B(t)$ on \mathbb{R}^n such that*

- $\llbracket t \rrbracket$ has a density (possibly infinite) ρ_t with respect to $B(t)$;
- if μ is another s-Hausdorff measure w.r.t. which $\llbracket t \rrbracket$ has a density, then that density is $\llbracket t \rrbracket$ -almost-everywhere equal to ρ_t .
- if t_2 has the same type and $\llbracket t_1 \rrbracket \ll \llbracket t_2 \rrbracket$ then $B(t_1) \ll B(t_2)$; and
- there is a set $A \subseteq \mathbb{R}^n$ such that $\mathbf{1}_A$ is a morphism and is a density of $B(t_1)$ with respect to $B(t_2)$.

In particular, this means that a sensible design decision for a PPL is to always compute densities of probabilistic programs p with respect to an s-Hausdorff base measure $B(p)$.

In order to compute Radon-Nikodym derivatives between multiple programs, the theorem implies we can separately compute their densities and then take the ratio, so long as we also include a “support check” (of the sort described by [10]).

Finally, the following result shows that the class of s-Hausdorff measures seems appropriate to serve as base measures, as for each s-Hausdorff measure, there is a closed probabilistic program whose posterior distribution has a strictly positive density w.r.t that measure. This means that we cannot ‘carve out’ any mass from the s-Hausdorff measure, and that the underlying supporting manifolds faithfully represent the possible supports of posterior distributions of probabilistic programs.

Theorem IV.4. *Assume that the language has all partial PAP functions as primitives. Then, for every $n \in \mathbb{N}$ and every s-Hausdorff measure μ , there exists a program $\vdash t : \mathbb{R}^n$ such that $\llbracket t \rrbracket$ and μ are mutually absolutely continuous.*

The proof is given at the end of Appendix E-B.

V. RELATED WORK AND DISCUSSION

Summary. We introduced the category of ω PAP spaces and used it as a denotational model for expressive higher-order recursive effectful languages. We first looked at a deterministic language with conditionals and partial PAP functions, which we argued covers almost all differentiable programs that can be implemented in practice. We showed that AD computes correct intentional derivatives in such an expressive setting, extending Lee et al. [5]’s result, and recovering the fact that AD computes derivatives which are correct almost-everywhere. Next, we showed that gradient descent on programs implementing differentiable functions can be soundly used with intentional derivatives, as long as both the learning rate and the initialization are randomized. We then looked at applications in probabilistic programming. After introducing a strong monad capturing traces of probabilistic programs, we gave a denotational proof that the trace density function of every probabilistic program is almost everywhere differentiable. Finally, we defined a commutative monad of measures on ω PAP, and proved that all programs denote measures with densities with respect to some *s-Hausdorff measures*. As such, we argued that the s-Hausdorff measures form a set of convenient base measures for density computations in Monte Carlo inference, and showed that every closed probabilistic program has a density w.r.t. some s-Hausdorff measure.

Together, these results demonstrate the value of denotational reasoning with a “just-specific-enough” model like ω PAP. Results previously established in the literature by careful operational reasoning, such as Theorem IV.1, follow almost immediately after setting up the definitions the right way. And we have also been able to show new theorems, such as Theorem IV.3, by applying methods from analysis to characterize the restricted class of denotations in our semantic domain.

Semantics of higher-order differentiable programming.

We continue a recent line of work on giving denotational semantics for higher-order differentiable programming in functional languages. Our semantics and logical relations proof builds on insights proposed in the smooth setting in Huot et al. [3], and takes inspiration from Vákár [4] for the treatment of recursion. In this work, we considered a simple forward-mode AD translation, but a whole body of work focused their attention on provably correct and efficient reverse-mode AD algorithms on higher-order languages [7, 17, 18, 19, 20]. We believe our correctness result could be adapted to a reverse-mode AD algorithm, perhaps following the neat ideas developed in Krawiec et al. [20], Radul et al. [21], or Smeding and Vákár [22], but we leave this for future work. There are also more synthetic approaches to studying differentiation in a more purely categorical tradition, sometimes called synthetic differential geometry [23]. Some of these approaches are particularly appealing from a theoretical point of view [24, 25], but their precise relations with AD and its implementations remains to be further studied.

Semantics of differentiable programming with non-smooth functions. Our work directly builds on and extends Lee et al. [5]’s setting to a higher-order recursive language. As is shown in Lee et al. [26], when moving beyond the differentiable setting, there are many seemingly reasonable classes of functions that behave pathologically in subtle ways, usually preventing compositional reasoning. ω PAP spaces combine the advantage of restricting to PAP functions at first-order with expressive power provided by abstract categorical constructions, which conservatively extend the first-order setting. Bolte and Pauwels [27] investigated a similar problem as Lee et al. [5] on a more restricted first-order language, but proved a convergence of gradient descent result in their setting. It would be interesting to see if some of the ideas developed in Chizat and Bach [28] could be adapted to prove some convergence of stochastic gradient descent when AD is used on programs denoting PAP functions, thus going beyond the setting of neural networks.

Semantics of probabilistic programming. Our commutative monad of measures takes clear inspiration from Staton [6] and Vákár et al. [2], adding an extra step in the recent search of semantic models for higher-order probabilistic programming [12, 29]. In particular, our work refines the model of Vákár et al. [2] by restricting to PAP functions, instead of merely measurable ones, but keeping its essential good properties for interpreting probabilistic programs. By doing so, our work is closer in spirit to Freer and Roy [30]’s study of computable properties for higher-order probabilistic languages. Our monad for tracking traces of probabilistic programs is similar to what is presented in Lew et al. [31], giving a denotational version of similar work that focus on operational semantics, such as Mak et al. [9]. Our result about densities and s-Hausdorff measures has some of its foundations based on the careful study of s-finite measures and kernels from Vákár and Ong [32]. It would be interesting to see if the work of Lew et al. [33] could be extended to a non-differentiable setting with PAP primitives, proving that an AD algorithm on a probabilistic language yields unbiased gradient estimates, perhaps using the estimator derived in Lee et al. [34] for non-smooth functions.

Disintegration and base measure. Our result on base measures for probabilistic programs is related to the literature on symbolic disintegration [35, 36, 37], which has also had to wrestle with the problem of finding base measures with respect to which expressive programs have densities. Our approach builds on the idea presented by Radul and Alexeev [10]: we extend the Hausdorff base measures to s-Hausdorff base measures, and prove that they suffice for ensuring all programs have densities. We leave open the question of characterizing exactly the class of densities on s-Hausdorff measures that arise from probabilistic programs, and the investigation of the closure of these measures under least upper bounds.

REFERENCES

- [1] C. Matache, S. Moss, and S. Staton, “Concrete categories and higher-order recursion (with applications including probability, differentiability, and full abstraction),” *arXiv preprint arXiv:2205.15917*, 2022.
- [2] M. Vákár, O. Kammar, and S. Staton, “A domain theory for statistical probabilistic programming,” *Proceedings of the ACM on Programming Languages*, vol. 3, no. POPL, p. 36, 2019.
- [3] M. Huot, S. Staton, and M. Vákár, “Correctness of automatic differentiation via diffeologies and categorical gluing,” in *FoSSaCS*, 2020, pp. 319–338.
- [4] M. Vákár, “Denotational correctness of forward-mode automatic differentiation for iteration and recursion,” *arXiv preprint arXiv:2007.05282*, 2020.
- [5] W. Lee, H. Yu, X. Rival, and H. Yang, “On correctness of automatic differentiation for non-differentiable functions,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 6719–6730, 2020.
- [6] S. Staton, “Commutative semantics for probabilistic programming,” in *European Symposium on Programming*. Springer, 2017, pp. 855–879.
- [7] D. Mazza and M. Pagani, “Automatic differentiation in pcf,” *Proceedings of the ACM on Programming Languages*, vol. 5, no. POPL, pp. 1–27, 2021.
- [8] A. Ścibior, O. Kammar, M. Vákár, S. Staton, H. Yang, Y. Cai, K. Ostermann, S. K. Moss, C. Heunen, and Z. Ghahramani, “Denotational validation of higher-order bayesian inference,” *Proceedings of the ACM on Programming Languages*, vol. 2, no. POPL, p. 60, 2017.
- [9] C. Mak, C.-H. L. Ong, H. Paquet, and D. Wagner, “Densities of almost surely terminating probabilistic programs are differentiable almost everywhere,” *Programming Languages and Systems*, vol. 12648, p. 432, 2021.
- [10] A. Radul and B. Alexeev, “The base measure problem and its solution,” in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 3583–3591.
- [11] J. Baez and A. Hoffnung, “Convenient categories of smooth spaces,” *Transactions of the American Mathematical Society*, vol. 363, no. 11, pp. 5789–5825, 2011.
- [12] C. Heunen, O. Kammar, S. Staton, and H. Yang, “A convenient category for higher-order probability theory,” in *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, 2017, pp. 1–12.
- [13] S. Abramsky and G. McCusker, “Call-by-value games,” in *Computer Science Logic: 11th International Workshop, CSL’97 Annual Conference of the EACSL Aarhus, Denmark, August 23–29, 1997 Selected Papers 11*. Springer, 1998, pp. 1–17.
- [14] E. Moggi, “Notions of computation and monads,” *Information and computation*, vol. 93, no. 1, pp. 55–92, 1991.
- [15] S.-y. Katsumata, “Relating computational effects by TT-lifting,” *Information and Computation*, vol. 222, pp. 228–246, 2013.
- [16] B. T. Polyak, “Introduction to optimization. optimization software,” *Inc., Publications Division, New York*, vol. 1, p. 32, 1987.
- [17] A. Brunel, D. Mazza, and M. Pagani, “Backpropagation in the simply typed lambda-calculus with linear negation,” *Proceedings of the ACM on Programming Languages*, vol. 4, no. POPL, pp. 1–27, 2019.
- [18] M. Vákár and T. Smeding, “Chad: Combinatory homomorphic automatic differentiation,” *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 44, no. 3, pp. 1–49, 2022.
- [19] M. Huot and A. Shaikhha, “Denotationally correct, purely functional, efficient reverse-mode automatic differentiation,” 2022. [Online]. Available: <https://arxiv.org/abs/2212.09801>
- [20] F. Krawiec, S. P. Jones, N. Krishnaswami, T. Ellis, R. A. Eisenberg, and A. W. Fitzgibbon, “Provably correct, asymptotically efficient, higher-order reverse-mode automatic differentiation,” *Proc. ACM Program. Lang.*, vol. 6, no. POPL, pp. 1–30, 2022.
- [21] A. Radul, A. Paszke, R. Frostig, M. Johnson, and D. Maclaurin, “You only linearize once: Tangents transpose to gradients,” *arXiv preprint arXiv:2204.10923*, 2022.
- [22] T. J. Smeding and M. I. Vákár, “Efficient dual-numbers reverse ad via well-known program transformations,” *Proceedings of the ACM on Programming Languages*, vol. 7, no. POPL, pp. 1573–1600, 2023.
- [23] J. R. B. Cockett and G. S. Crutwell, “Differential structure, tangent structure, and sdg,” *Applied Categorical Structures*, vol. 22, no. 2, pp. 331–417, 2014.
- [24] R. Cockett, G. Crutwell, J. Gallagher, J.-S. P. Lemay, B. MacAdam, G. Plotkin, and D. Pronk, “Reverse derivative categories,” *arXiv preprint arXiv:1910.07065*, 2019.
- [25] R. Blute, T. Ehrhard, and C. Tasson, “A convenient differential category,” *arXiv preprint arXiv:1006.3140*, 2010.
- [26] W. Lee, X. Rival, and H. Yang, “Smoothness analysis for probabilistic programs with application to optimised variational inference,” *Proceedings of the ACM on Programming Languages*, vol. 7, no. POPL, pp. 335–366, 2023.
- [27] J. Bolte and E. Pauwels, “A mathematical model for automatic differentiation in machine learning,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 10809–10819, 2020.
- [28] L. Chizat and F. Bach, “On the global convergence of gradient descent for over-parameterized models using optimal transport,” *Advances in neural information processing systems*, vol. 31, 2018.
- [29] T. Ehrhard, M. Pagani, and C. Tasson, “Measurable cones and stable, measurable functions: a model for probabilistic higher-order programming,” *Proceedings of the ACM on Programming Languages*, vol. 2, no. POPL, pp. 1–28, 2017.
- [30] C. E. Freer and D. M. Roy, “Computable de finetti

measures,” *Annals of Pure and Applied Logic*, vol. 163, no. 5, pp. 530–546, 2012.

arXiv preprint arXiv:1512.07276, 2015.

- [31] A. K. Lew, M. F. Cusumano-Towner, B. Sherman, M. Carbin, and V. K. Mansinghka, “Trace types and denotational semantics for sound programmable inference in probabilistic languages,” *Proceedings of the ACM on Programming Languages*, vol. 4, no. POPL, pp. 1–32, 2019.
- [32] M. Vákár and L. Ong, “On s-finite measures and kernels,” *arXiv preprint arXiv:1810.01837*, 2018.
- [33] A. K. Lew, M. Huot, S. Staton, and V. K. Mansinghka, “Adev: Sound automatic differentiation of expected values of probabilistic programs,” *Proceedings of the ACM on Programming Languages*, vol. 7, no. POPL, pp. 121–153, 2023.
- [34] W. Lee, H. Yu, and H. Yang, “Reparameterization gradient for non-differentiable models,” in *Advances in Neural Information Processing Systems*, 2018, pp. 5553–5563.
- [35] C.-c. Shan and N. Ramsey, “Exact bayesian inference by symbolic disintegration,” in *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages*, 2017, pp. 130–144.
- [36] K. Cho and B. Jacobs, “Disintegration and bayesian inversion via string diagrams,” *Mathematical Structures in Computer Science*, vol. 29, no. 7, pp. 938–971, 2019.
- [37] P. Narayanan and C.-c. Shan, “Symbolic disintegration with a variety of base measures,” *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 42, no. 2, pp. 1–60, 2020.
- [38] B. Jacobs, *Categorical logic and type theory*. Elsevier, 1999.
- [39] J. C. Mitchell and A. Scedrov, “Notes on sconing and relators,” in *International Workshop on Computer Science Logic*. Springer, 1992, pp. 352–378.
- [40] S.-y. Katsumata, “A semantic formulation of TT-lifting and logical predicates for computational metalanguage,” in *International Workshop on Computer Science Logic*. Springer, 2005, pp. 87–102.
- [41] S. MacLane and I. Moerdijk, *Sheaves in geometry and logic: A first introduction to topos theory*. Springer Science & Business Media, 2012.
- [42] A. Carboni and P. Johnstone, “Connected limits, familial representability and artin glueing,” *Mathematical Structures in Computer Science*, vol. 5, no. 4, pp. 441–459, 1995.
- [43] P. T. Johnstone, S. Lack, and P. Sobociński, “Quasitoposes, quasiadhesive categories and artin glueing,” in *International Conference on Algebra and Coalgebra in Computer Science*. Springer, 2007, pp. 312–326.
- [44] P. T. Johnstone, *Sketches of an elephant: A topos theory compendium*. Oxford University Press, 2002, vol. 2.
- [45] O. Kammar and D. McDermott, “Factorisation systems for logical relations and monadic lifting in type-and-effect system semantics,” *Electronic notes in theoretical computer science*, vol. 341, pp. 239–260, 2018.
- [46] B. Mityagin, “The zero set of a real analytic function,”

APPENDIX A LANGUAGE

A. Type System

The type-system for the deterministic language (Section III) is given in Figure 10.

$$\begin{array}{c}
\frac{}{\Gamma \vdash x : \tau} (x : \tau \in \Gamma) \quad \frac{}{\Gamma \vdash c : \tau} (c \in \mathcal{C}_\tau) \\
\frac{\Gamma, x : \tau_1 \vdash t : \tau_2}{\Gamma \vdash \lambda x : \tau_1. t : \tau_1 \rightarrow \tau_2} \\
\frac{\Gamma \vdash t_1 : \tau_1 \rightarrow \tau_2 \quad \Gamma \vdash t_2 : \tau_1}{\Gamma \vdash t_1 t_2 : \tau_2} \\
\frac{\Gamma \vdash t_1 : \mathbb{B} \quad \Gamma \vdash t_2 : \tau \quad \Gamma \vdash t_3 : \tau}{\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : \tau} \\
\frac{\Gamma, f : \tau_1 \rightarrow \tau_2 \vdash t : \tau_1 \rightarrow \tau_2}{\Gamma \vdash \mu f : \tau_1 \rightarrow \tau_2. t : \tau_1 \rightarrow \tau_2}
\end{array}$$

Fig. 10. Type system of the deterministic part of the language. \mathcal{C}_τ denotes the set of primitives of type τ .

B. Operational semantics

Values are given by the following grammar

$$v ::= x \mid c \mid f \mid \lambda x. t \mid \mu f. t \mid \langle v_1, v_2 \rangle$$

A big step-semantics is given in Figure 11.

$$\begin{array}{c}
\frac{}{c \Downarrow c} \quad \frac{t_2 \Downarrow \lambda x. t \quad t_1 \Downarrow v_1 \quad t[v_1/x] \Downarrow v_2}{t_2 t_1 \Downarrow v_2} \\
\frac{\forall i, t_i \Downarrow v_i \quad t \Downarrow f}{t(t_1, \dots, t_n) \Downarrow f(v_1, \dots, v_n)} \\
\frac{t_1 \Downarrow \text{true} \quad t_2 \Downarrow v}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \Downarrow v} \\
\frac{t_1 \Downarrow \text{false} \quad t_3 \Downarrow v}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \Downarrow v} \\
\frac{t \Downarrow \langle v_1, v_2 \rangle \quad t_2[v_1/x_1, v_2/x_2] \Downarrow v}{\text{case } t \text{ of } \langle x_1, x_2 \rangle \rightarrow t_2 \Downarrow v} \\
\frac{t[\mu f. t/f] \Downarrow v}{\mu f. t \Downarrow v}
\end{array}$$

Fig. 11. Big step operational semantics of the deterministic part of the language

APPENDIX B SOUND AND ADEQUATE DENOTATION

1) *Concrete sites and sheaves:* Our proof of soundness and adequacy uses reusable machinery developed by Matache et al. [1]. Their framework requires establishing various properties; for convenience, in this section, we reproduce the relevant definitions, and recall the theorem that we will use.

Definition B.1 (Reproduced from [1]). A concrete category is a category \mathbf{C} with a terminal object 1 such that the functor $\mathbf{C}(1, -) : \mathbf{C} \rightarrow \mathbf{Set}$ is faithful.

Example B.1. Let **OpenCont** be the category whose objects are open subsets U of \mathbb{R}^n (for all $n \in \mathbb{N}$), and whose morphisms $U \rightarrow V$ are continuous functions. \mathbb{R}^0 is a terminal object, and **OpenCont**($\mathbb{R}^0, -$) simply sends an open set to its underlying set, and sends a continuous function to its underlying function. Therefore, **OpenCont**($\mathbb{R}^0, -$) is faithful and **OpenCont** is concrete.

Definition B.2 (Reproduced from [1]). A concrete site $(\mathbf{C}, \mathcal{J})$ is a small concrete category \mathbf{C} with an initial object 0, together with a coverage \mathcal{J} , which specifies for each object c a set $\mathcal{J}(c)$ of families of maps with codomain c . We call such a family $\{f_i : c_i \rightarrow c\}_{i \in I} \in \mathcal{J}(c)$ a covering family and say it covers c . The coverage must satisfy the following axioms.

- 1) For every map $h : d \rightarrow c$ in \mathbf{C} , if $\{f_i : c_i \rightarrow c\}_{i \in I}$ covers c , then there is a covering family $\{g_j : d_j \rightarrow d\}_{j \in I'}$ of d such that every $h \circ g_j$ factors through some f_i .
- 2) If $\{f_i : c_i \rightarrow c\}_{i \in I}$ covers c , then $\bigcup_{i \in I} \text{Im}(|f_i|) = |c|$.
- 3) the initial object 0 is covered by the empty set.
- 4) The identity is always covering
- 5) If $\{f_i : c_i \rightarrow c\}_{i \in I} \in \mathcal{J}(c)$ and $\{g_{ij} : c_{ij} \rightarrow c_i\}_{j \in J_i} \in \mathcal{J}(c_i)$ for each i , then $\{f_i \circ g_{ij} : c_{ij} \rightarrow c\}_{i \in I, j \in J_i} \in \mathcal{J}(c)$.

Example B.2 (Cartesian spaces and smooth maps). **CartSp** is a concrete site, where $\{f_i : U_i \rightarrow V\}_i$ is a covering family if $\bigcup_i f_i(U_i) = V$, i.e. the images of the f_i cover the codomain V in the usual sense.

Example B.3. Similarly to **CartSp**, **OpenCont** is a concrete site, where $\{f_i : U_i \rightarrow V\}_i$ is a covering family if $\bigcup_i f_i(U_i) = V$, i.e. the images of the f_i cover the codomain V in the usual sense. Its initial object is the empty set.

Example B.4 (Standard Borel spaces). The category **Sbs** has objects the Borel subsets of \mathbb{R} and morphisms the measurable functions between these objects. It is a concrete site where the coverage contains the countable sets of inclusions functions $\{U_i \rightarrow U\}_i$ such that $U = \bigcup_i U_i$ and the U_i are disjoint.

Definition B.3 (Reproduced from [1]). A concrete sheaf X on a concrete site $(\mathbf{C}, \mathcal{J})$ is a set $|X|$, together with, for each object $c \in \mathbf{C}$, a set \mathcal{P}_X^c of functions $|c| \rightarrow |X|$, such that

- each \mathcal{P}_X^c contains all the constant functions;
- for any map $h : d \rightarrow c \in \mathbf{C}$, and any $g \in \mathcal{P}_X^c$, the composite function $g \circ |h| : |d| \rightarrow |X|$ is in \mathcal{P}_X^d ;
- for each function $g : |c| \rightarrow |X|$ and each covering family $\{f_i : c_i \rightarrow c\}_{i \in I}$, if each $g \circ |f_i| \in \mathcal{P}_X^{c_i}$, then $g : |c| \rightarrow |X| \in \mathcal{P}_X^c$.

A morphism $\alpha : X \rightarrow Y$ between concrete sheaves is a function $\alpha : |X| \rightarrow |Y|$ that preserves the structure, namely if $g \in \mathcal{P}_X^c$, then $\alpha \circ g \in \mathcal{P}_Y^c$.

Example B.5. Let **Cont** be the category defined as follows. Its objects are pairs (X, \mathcal{P}_X) where for each open $U \subseteq \mathbb{R}^n$, $\mathcal{P}_X^U \subseteq X^U$ is closed under the following axioms:

- Constant functions are in \mathcal{P}_X^U
- If $f \in \mathcal{P}_X^U$ and $g : V \rightarrow U$ is continuous, then $f \circ g \in \mathcal{P}_X^V$

- If $\{U_i\}_{i \in \mathbb{N}}$ is an open cover of an open U , and for all i , $f_i \in \mathcal{P}_X^{U_i}$, such that for all $i, j \in \mathbb{N}^2$, f_i and f_j agree on $U_i \cap U_j$, then $f : U \rightarrow X$ defined by $x \in U_i \mapsto f_i(x)$, is an element of \mathcal{P}_X^U .

Its morphisms $X \rightarrow Y$ are functions $\alpha : X \rightarrow Y$ such that $\alpha \circ f \in \mathcal{P}_Y^U$ whenever $f \in \mathcal{P}_X^U$. Then, **Cont** is a category of concrete sheaves on the concrete site **OpenCont**.

Example B.6. *Diffeological spaces and Quasi-Borel spaces are examples of categories of concrete sheaves and morphisms between concrete sheaves. Other examples are given in Baez and Hoffnung [11].*

2) ω -Concrete sheaves: To interpret recursion, the concrete-sheaf structure will not be sufficient in general. Instead of evoking what is to my knowledge the yet-to-be-formalised theory of enriched sheaves over the category of ω cpo's, we continue with the setting from Matache et al. [1].

Definition B.4 (Reproduced from [1]). *An ω -concrete sheaf on a site $(\mathbf{C}, \mathcal{J})$ is a concrete sheaf X together with an ordering \leq_X on $|X|$ that equips X with the structure of ω cpo, such that each \mathcal{P}_X^c is closed under pointwise suprema of chains with respect to the pointwise ordering. A morphism $\alpha : X \rightarrow Y$ of ω -concrete sheaves is a continuous function between ω cpo's, $\alpha : |X| \rightarrow |Y|$, that is also a morphism of concrete sheaves.*

ω -concrete sheaves form a category $\omega\text{Conc}(\mathbf{C}, \mathcal{J})$, which is a Cartesian-closed category with binary coproducts [1].

Example B.7. *The category of ω -diffeological spaces [4] and of ω -Quasi-Borel spaces [2] are examples of categories of ω -concrete sheaves.*

3) *Partiality: admissible monos and a partiality monad:* To model recursion, we first need to define a (strong) partiality monad \mathbf{L} on $\omega\text{Conc}(\mathbf{C}, \mathcal{J})$. There are many choices for the lifting of the plots, so we parametrize the definition of partiality monad by a class \mathcal{M} of monomorphisms from the site $(\mathbf{C}, \mathcal{J})$, which we call admissible monos.

Recall that, in any category, monos with the same codomain are preordered. If $m_1 : d_1 \rightarrow c, m_2 : d_2 \rightarrow c$ then $m_1 \leq m_2$ iff there exists $f : d_1 \rightarrow d_2$ such that $m_2 \circ f = m_1$. We write $\text{Sub}(c)$ for the poset quotient of the set of monos with codomain c . For any class \mathcal{M} of monos, we write $\text{Sub}_{\mathcal{M}}(c)$ for the full subposet of $\text{Sub}(c)$ whose elements have representatives in \mathcal{M} .

Definition B.5 (Reproduced from [1]). *A class \mathcal{M} of admissible monos for a concrete site $(\mathbf{C}, \mathcal{J})$ consists of, for each object $c \in \mathbf{C}$, a set of monos $\mathcal{M}(c)$ with codomain c satisfying the following conditions*

- 1) For all $c \in \mathbf{C}, 0 \rightarrow c \in \mathcal{M}(c)$
- 2) \mathcal{M} contains all isomorphisms
- 3) \mathcal{M} is closed under composition
- 4) All pullbacks of \mathcal{M} -maps exist and are again in \mathcal{M} (This makes $\text{Sub}_{\mathcal{M}}$ a functor $\mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$).
- 5) For each c , the function $\text{Sub}_{\mathcal{M}}(c) \rightarrow \mathbf{Set}(|c|, \{0, 1\})$ is componentwise injective and order-reflecting, and the

image of $\text{Sub}_{\mathcal{M}}$ is closed under suprema of ω -chains

6) Given an increasing chain in $\text{Sub}_{\mathcal{M}}(c)$, $(c_n \rightarrow c)_{n \in \mathbb{N}}$, denote its least upper bound by $c_{\infty} \rightarrow c$. Then the closure under precomposition (with any morphism) of the set $\{c_n \rightarrow c_{\infty}\}_{n \in \mathbb{N}}$ contains a covering family of c_{∞} .

Now, given a class \mathcal{M} of admissible monos, we can define a partiality monad $L_{\mathcal{M}}$ on the category of ω -concrete sheaves:

Definition B.6 (Reproduced from [1]). *We define the strong partiality monad $L_{\mathcal{M}}$ associated to the class of admissible monos \mathcal{M} as*

- $|L_{\mathcal{M}}X| = |X| \sqcup \{\perp\}$
- $\forall x \in |X|, \perp \leq_{L_{\mathcal{M}}X} x$
- $\forall x, x' \in |X|, x \leq_{L_{\mathcal{M}}X} x' \text{ iff } x \leq_X x'$
- $\mathcal{P}_{L_{\mathcal{M}}X}^c = \{g : |c| \rightarrow |X| \sqcup \{\perp\} \mid \exists c' \rightarrow c \in \mathcal{M}(c) \text{ s.t. } g^{-1}(|X|) = \text{Im}(|c'|) \text{ and } g|_{\text{Im}(|c'|)} \in \mathcal{P}_X^{c'}\}$

The strong monad structure is exactly the same as the maybe monad on **Set**.

Proposition B.1 ([1]). $L_{\mathcal{M}} : \omega\text{Conc}(\mathbf{C}, \mathcal{J}) \rightarrow \omega\text{Conc}(\mathbf{C}, \mathcal{J})$ is a strong monad.

4) *Soundness and adequacy:* Matache et al. [1] defined a language PCF_v , a call-by-value variant of PCF. It has product, sum types, and a recursion operator similar to ours, and is presented as a fine-grained call-by-value language. This language is essentially equivalent to our language, except that we have extra base types **real** and \mathbb{B} , and more basic primitives involving these types. Our conditional is definable using their sum types. They give their language a standard call-by-value operational semantics, which translates directly to an operational semantics for our language. In summary, even though their theorem is stated for PCF_v , it directly applies to our language, without needing further changes or extra proofs, as long as we also interpret our base types and primitives in the model, as in standard [14].

Theorem B.2 ([1]). *A concrete site with a class of admissible monos, $(\mathbf{C}, \mathcal{J}, \mathcal{M})$, is a sound and adequate model of PCF_v in $\omega\text{Conc}(\mathbf{C}, \mathcal{J})$.*

Having recalled the general setting of ω -concrete sheaves, we now show how ωPAP is an instance of this general construction. To do so, we will show define the category $c\text{PAP}$ and show it is a concrete-site (which is the one for ωPAP) in Section B-6. We will define a set of admissible monos for $c\text{PAP}$ in Section B-7. In order to achieve this, we will need a key technical proposition (Proposition B.3) and a few other technical lemmas, proved in Section B-5.

5) *C-analytic sets:* The definition of c-analytic sets does not require that the analytic sets that make up a c-analytic set be disjoint, but it turns out that it is always possible to partition a c-analytic set into countably many pairwise disjoint analytic sets.

Proposition B.3. *For any c-analytic set A , there exists a countable collection of pairwise disjoint analytic sets $\{B_i\}_{i \in \mathbb{N}}$*

such that $A = \cup_{i \in \mathbb{N}} B_i$.

This is an important property, and most of the rest of this section is devoted to proving the proposition. The proof is due to Alex Lew.

Let $\{A_i\}_{i \in \mathbb{N}}$ be a countable set of analytic subsets of \mathbb{R}^n .

Each analytic set A_i is associated with an open domain U_i and a finite number J_i of analytic functions $\{f_{ij}\}_{j \in [1, \dots, J_i]}$, so that $A_i = \{x \in U_i \mid \forall j. f_{ij} \leq 0\}$. Because any open set is a countable union of open balls in \mathbb{R}^n , we can assume without loss of generality that the U_i are open balls.

Definition B.7. A simple analytic set is a set $A = \{x \in B_\epsilon(x_0) \mid \forall i \leq I. g_i^+(x) > 0 \wedge \forall j \leq J. g_j^-(x) \leq 0\}$, for natural numbers I and J and some $0 < \epsilon \leq \infty$ and $x_0 \in \mathbb{R}^n$.

Lemma B.4. The complement of any simple analytic set is a finite union of pairwise disjoint simple analytic sets.

Proof. A point x can fail to be in a simple analytic set A for $I + J + 1$ mutually exclusive reasons, each of which applies to a simple analytic set of points:

- 1) If $\epsilon < \infty$, it can fail to lie in $B_\epsilon(x_0)$. The set of points outside $B_\epsilon(x_0)$ is simple analytic: let $h_1^-(x) = \epsilon - \|x - x_0\|_2$ and consider $\{x \in \mathbb{R}^n \mid h_1^-(x) \leq 0\}$.
- 2) (For each $1 \leq i \leq I$.) It can lie within $B_\epsilon(x_0)$, and satisfy $g_n^+(x) > 0$ for all $n < i$, but fail to satisfy $g_i^+(x) > 0$. Let $h_n^+(x) = g_n^+(x)$ for $n < i$, and $h_1^-(x) = g_i^+(x)$. Then the set of all such points is $\{x \in B_\epsilon(x_0) \mid \forall n < i. h_n^+(x) > 0 \wedge h_1^-(x) \leq 0\}$.
- 3) (For each $1 \leq j \leq J$.) It can lie within $B_\epsilon(x_0)$, and satisfy $g_i^+(x) > 0$ for all $i \leq I$, and satisfy $g_n^-(x) \leq 0$ for all $n < j$, but fail to satisfy $g_j^-(x) \leq 0$. Let $h_i^+(x) = g_i^+(x)$ for all $i \leq I$, $h_n^-(x) = g_n^-(x)$ for all $n < j$, and $h_{I+1}^+(x) = g_j^-(x)$. Then the set of all points to which *this* reason applies is $\{x \in B_\epsilon(x_0) \mid \forall i \leq I + 1. h_i^+(x) > 0 \wedge \forall n < j. h_n^-(x) \leq 0\}$.

The union of these simple analytic sets is the complement of A . \square

Corollary B.4.1. If A_1, \dots, A_n are simple analytic sets, then $\overline{\cup_{i \leq n} A_i}$ is a finite union of disjoint analytic sets.

Proof. We have $\overline{\cup_{i \leq n} A_i} = \cap \overline{A_i} = \cap (\cup_{m \leq M_i} B_{im})$, where $\{B_{im}\}_{m \leq M_i}$ is a representation of A_i 's complement as a disjoint union of finitely many simple analytic sets. Distributing, this is in turn equal to $\cup_{m_1 \leq M_1} \dots \cup_{m_n \leq M_n} (\cap_{i \leq n} B_{im_i})$. Each element of this large union is analytic, because it is the intersection of n analytic sets. Furthermore, these analytic sets are pairwise disjoint: any pair will disagree on m_i for some i , and so the intersection will include disjoint sets B_{im_i} and $B_{im'_i}$, for $m_i \neq m'_i$. Because these two sets are disjoint, the overall intersections will be disjoint. \square

Corollary B.4.2. If A_1, \dots, A_n are simple analytic sets, then $A_n \setminus \cup_{i < n} A_i$ is a finite union of disjoint analytic sets, each of which is a finite intersection of simple analytic sets.

Proof. $A_n \setminus \cup_{i < n} A_i = A_n \cap \overline{(\cup_{i < n} A_i)}$, and by the previous corollary, the right-hand term can be rewritten as a finite union

of disjoint analytic sets $\cup_{i < m} B_i$. The intersection is then equal to $\cup_{i < m} A_n \cap B_i$. Each element of this finite union is analytic, because analytic sets are closed under intersection. They are also disjoint, since the B_i are disjoint. \square

Lemma B.5. Let $A = \cup_{n \in \mathbb{N}} A_n$ be a countable union of simple analytic sets A_n . Then there are countably many pairwise disjoint analytic sets $\{B_m\}_{m \in \mathbb{N}}$ such that $A = \cup_{m \in \mathbb{N}} B_m$.

Proof. Consider the countably many disjoint sets $A'_k = A_k \setminus (\cup_{i < k} A_i)$. Each A'_k can itself be expressed as a countable union of pairwise disjoint sets, by the previous corollary. \square

Corollary B.5.1. Every c -analytic set A can be written as a countable disjoint union of analytic sets.

Proof. Any open domain U can be expressed as a countable union of open balls. Therefore, in a countable union of arbitrary analytic sets, any analytic set A with a complicated open domain U can be replaced by countably many analytic sets with open-ball domains (and the same defining inequalities as A). We conclude with Lemma B.5, using the fact that a countable family of countable families is a countable family. \square

Next, we show a corollary that will be useful in the next section.

Corollary B.5.2. An inclusion $U \subseteq V$ of c -analytic sets U, V is a PAP function $U \rightarrow V$.

Proof. By Corollary B.5.1, we can write $U = \sqcup_{i \in \mathbb{N}} A_i$ and $V = \sqcup_{j \in \mathbb{N}} B_j$ where the A_i and B_j are analytic sets. Analytic sets are closed under intersection, so $\{A_i \cap B_j\}_{i \in \mathbb{N}, j \in \mathbb{N}}$ is a countable partition of U into analytic sets. Hence, $\{(A_i \cap B_j, id)\}_{i \in \mathbb{N}, j \in \mathbb{N}}$ is a piecewise representation of the inclusion $U \rightarrow V$. \square

Finally, we show a few closure properties of c -analytic sets.

Lemma B.6. If $f : U \rightarrow V$ is analytic and $A \subseteq V$ is analytic, then $f^{-1}(A)$ is analytic.

Proof. By definition, $A = \{x \in (\cap_i X_i^+) \cap (\cap_j X_j^-) \mid \forall i. g_i^+(x) > 0 \wedge \forall j. g_j^-(x) \leq 0\}$ for some analytic functions g_i^+, g_j^- . Thus,

$$\begin{aligned} f^{-1}(A) &= \{x \in f^{-1}\left(\left(\bigcap_i X_i^+\right) \cap \left(\bigcap_j X_j^-\right)\right) \mid \\ &\quad \forall i. g_i^+(f(x)) > 0 \wedge \forall j. g_j^-(f(x)) \leq 0\} \\ &= \{x \in \left(\bigcap_i f^{-1}(X_i^+)\right) \cap \left(\bigcap_j f^{-1}(X_j^-)\right) \mid \\ &\quad \forall i. (g_i^+ \circ f)(x) > 0 \wedge \forall j. (g_j^- \circ f)(x) \leq 0\} \end{aligned}$$

As f is analytic, it is continuous and so the sets $f^{-1}(X_i^+)$ and $f^{-1}(X_j^-)$ are open. And again, as f is analytic, so are the functions $g_i^+ \circ f$ and $g_j^- \circ f$. \square

Corollary B.6.1. If $f : U \rightarrow V$ is PAP and $A \subseteq V$ is c -analytic, then $f^{-1}(A)$ is c -analytic.

Proof. By Corollary B.5.1, we can write $A = \sqcup_i A_i$ for analytic sets A_i . Thus, $f^{-1}(A) = f^{-1}(\sqcup_i A_i) = \sqcup_i f^{-1}(A_i)$ and it is sufficient to show that each $f^{-1}(A_i)$ is c-analytic.

f is PAP so there exists a partition $U = \sqcup_i B_i$ where B_i are analytic sets and analytic functions $f_i : U_i \rightarrow R^n$ such that $f|_{B_i} = f_i$. Therefore, $f^{-1}(A) = \sqcup_i f_i^{-1}(A) \cap B_i$. Each $f_i^{-1}(A)$ is analytic by Lemma B.6, and so each $f_i^{-1}(A) \cap B_i$ is analytic. This means $f^{-1}(A)$ is indeed c-analytic. \square

6) Category cPAP:

Definition B.8 (cPAP). We call cPAP the category whose objects are c-analytic sets and whose morphisms are PAP functions between them. Composition is given by the usual composition of functions.

Lemma B.7. cPAP is a concrete category.

Proof. • **cPAP has a terminal object.** The analytic set \mathbb{R}^0 is terminal.

• **The functor $\text{hom}(1, -) : \text{cPAP} \rightarrow \text{Set}$ is faithful.** The functor is the identity on morphisms. \square

Proposition B.8. cPAP is a concrete site, where the coverings for a c-analytic set U are given by countable c-analytic partitions of U , i.e., $\{(U_i)_i \mid \cup_i U_i = U \wedge \forall i \neq j, U_i \cap U_j = \emptyset\}$.

Proof. First, note that our definition for the coverings is a well-defined, as inclusions are PAP functions by Corollary B.5.2.

We now show that the given coverings satisfy the 5 axioms of a concrete site.

- 1) Suppose we have a cPAP morphism $g : C \rightarrow D$, and a c-analytic partition $\{D_i\}_{i \in I}$ of D . Then we wish to find a partition of C so that g can be represented as a piecewise gluing of functions with codomains D_i . First, note that since they are objects of cPAP, each D_i can be further partitioned into countably many distinct analytic subsets $\{D_{ij}\}_{j \in J_i}$, $D_{ij} \subseteq D_i$. Furthermore, because g is a morphism, it has a PAP representation $\{(C_k, g_k)\}_{k \in K}$ for analytic subsets $C_k \subseteq C$ and analytic functions g_k . For each i, j, k define $C_{ijk} = \{c \in C_k \mid g_k(c) \in D_{ij}\}$: since g_k is an analytic function and D_{ij} is an analytic set, C_{ijk} is analytic. Define g_{ijk}^* to be the restriction of g_k to C_{ijk} . Then $\{(C_{ijk}, g_{ijk}^*)\}_{i \in I, j \in J_i, k \in K}$ is a piecewise representation of g in which each piece's codomain is D_i for some i .
- 2) Let $\{f_i : c_i \rightarrow c\}_{i \in I}$ be a covering of c . Then $\bigcup_{i \in I} \text{Im}(|f_i|) = |c|$. This follows from the definition of *partition*.
- 3) The initial object 0 is covered by the empty set.
- 4) The identity is always covering.
- 5) Let $\{f_i : c_i \rightarrow c\}_{i \in I}$ be a covering of c and $\{g_{ij} : c_{ij} \rightarrow c_i\}_{j \in J_i}$ be a cover of c_i for each i . Then $\{f_i \circ g_{ij} : c_{ij} \rightarrow c\}_{i \in I, j \in J_i} \in \mathcal{J}(c)$. This also follows from the definition of *partition*, and the fact that a countable union of countable sets is countable. \square

Lemma B.9. cPAP is a subcanonical site.

Proof. Let X be a representable presheaf on cPAP. Then, up to natural isomorphism, X maps a c-analytic set A to $\text{cPAP}(A, B)$ for some fixed c-analytic set B . Consider a covering family $\{(A_i)\}_{i \in I}$ for A , and a compatible collection of plots $\phi_i \in X(A_i)$, which can be identified (via the natural isomorphism) with cPAP morphisms $\phi_i : A_i \rightarrow B$. Then we must show that there is a unique $\phi \in X(A)$ whose restriction to each A_i is ϕ_i . To see this, note that each ϕ_i has a PAP representation $\{(A_{ij}, \phi_{ij})\}_{j \in J_i}$. By the definition of a covering family on cPAP, the A_i are disjoint, and so $\{(A_{ij}, \phi_{ij})\}_{i \in I, j \in J_i}$ is a PAP representation; the ϕ we are looking for is the function it represents. \square

7) **Admissible monos for cPAP:** As an admissible set of monos, we choose those given by the monos defined on an c-analytic subset of their domain of definition. More formally, we define \mathcal{M}_{PAP} for cPAP as follows:

$$\mathcal{M}_{\text{PAP}}(B) = \{m : A \rightarrow B \mid A \cong A', A' \text{ is a c-analytic subset of } B\}$$

Proposition B.10. \mathcal{M}_{PAP} is an admissible class of monos.

Proof. 1) For all $c \in \mathbf{C}$, $0 \rightarrow c \in \mathcal{M}_{\text{PAP}}(c)$: the empty-set is c-analytic.

2) \mathcal{M}_{PAP} contains all isomorphisms: clear.

3) \mathcal{M}_{PAP} is closed under composition: clear.

4) All pullbacks of \mathcal{M}_{PAP} -maps exist and are again in \mathcal{M}_{PAP} : this amounts to showing that the preimage $B := f^{-1}(A)$ for a PAP function f and a c-analytic set A is c-analytic. This is true by Corollary B.6.1.

5) For each c , the function $\text{Sub}_{\mathcal{M}}(c) \rightarrow \text{Set}(|c|, \{0, 1\})$ is componentwise injective and order-reflecting, and the image of $\text{Sub}_{\mathcal{M}}$ is closed under suprema of ω -chains: the function is the identity on sets, and is thus injective and order-reflecting. Given an omega chain $\{A_i \subseteq B\}_{i \in \mathbb{N}}$, we have $i < j \Rightarrow A_i \subseteq A_j$. Let $A := \bigcup_{i \in \mathbb{N}} A_i \subseteq B$. It is a countable union of c-analytic sets and is thus a c-analytic set.

6) Given an increasing chain in $\text{Sub}_{\mathcal{M}}(c)$, $(c_n \rightarrow c)_{n \in \mathbb{N}}$, denote its least upper bound by $c_\infty \rightarrow c$. Then the closure under precomposition (with any morphism) of the set $(c_n \rightarrow c_\infty)_{n \in \mathbb{N}}$ contains a covering family of c_∞ : as shown in the previous point, c_∞ is c-analytic and by Corollary B.5.1, $c_\infty = \sqcup A_i$ where A_i are analytic sets. Thus, $(c_n \cap A_i \rightarrow c_\infty)_{n \in \mathbb{N}, i \in \mathbb{N}}$ is a covering family of c_∞ that is contained in the closure by precomposition of $(c_n \rightarrow c_\infty)_{n \in \mathbb{N}}$. \square

A. Soundness and adequacy

We can now prove Theorem II.2:

Theorem B.11. ωPAP has products, coproducts, exponentials and $\mathbf{L} : \omega\text{PAP} \rightarrow \omega\text{PAP}$ is a strong lifting monad.

Proof. It follows directly from the fact that we have seen that ωPAP is a category of ω -concrete sheaves for the concrete site $c\text{PAP}$ (Proposition B.8), and \mathbf{L} is obtained from an admissible class of monos on $c\text{PAP}$ (Propositions B.10 and B.1), by Proposition 7.5 in Matache et al. [1]. \square

Proof of Theorem II.2. We apply theorem B.2 to the concrete site $(\text{PAP}, \mathcal{J}_{\text{PAP}}, \mathcal{M}_{\text{PAP}})$ with the admissible class of monos \mathcal{M}_{PAP} . \square

APPENDIX C CORRECTNESS OF AD

The main goal of this section is to show Theorem III.2. To do so, we apply the categorical machinery of fibrations for logical relations developed in [15]. They build on the well-known theory of fibrations [38]. As we follow their recipe closely, we recall the minimal amount of fibration theory in Section C-A needed to understand their machinery, which we also reproduce in Section C-A2 for convenience. We finally use the machinery of fibrations for logical relations in Section C-B to prove Theorem III.2, our correctness result on AD.

A. Fibrations for logical relations

Roughly, fibrations offer a useful and unifying point of view for reasoning about generalized predicates on a category. Indeed, the functor $p : \mathbf{Gl} \rightarrow \mathbf{C}$ from a glueing/sconing category [39] to the base category is an example of nice fibration. This justifies to study a categorical version of logical relations using fibrations. We follow closely this point of view developed in [15, 40].

1) *Fibrations:* We recall some basic facts from the theory of fibrations [38].

Definition C.1 (Reproduced from [38]). Let $p : \mathbb{E} \rightarrow \mathbb{B}$ be a functor. A morphism $f : X \rightarrow Y \in \mathbb{E}$ is **Cartesian over** $u : I \rightarrow J \in \mathbb{B}$ if $pf = u$ and every $g : Z \rightarrow Y \in \mathbb{E}$ for which $pg = u \circ w$ for some $w : pZ \rightarrow I$, there is a unique $h : Z \rightarrow X \in \mathbb{E}$ above w with $f \circ h = g$.

Definition C.2 (Reproduced from [38]). A **fibration** is a functor $p : \mathbb{E} \rightarrow \mathbb{B}$ such that for every $Y \in \mathbb{E}$ and $u : I \rightarrow pY \in \mathbb{B}$, there is a Cartesian morphism $f : X \rightarrow Y \in \mathbb{E}$ above u . It is also called **fibred category** or **category (fibred) over** \mathbb{B} .

Example C.1. Let \mathbf{Pred} be the category whose objects are pairs (X, S) of a set X and a subset $S \subseteq X$, and morphisms $(X, S) \rightarrow (Y, T)$ are functions $f : X \rightarrow Y$ such that $f(S) \subseteq T$. Let $p : \mathbf{Pred} \rightarrow \mathbf{Set}$ be the functor that forgets the subsets S . It is a fibration.

Proposition C.1. (Change-of-base, [38]) Let $p : \mathbb{B} \rightarrow \mathbb{B}$ be a fibration and $K : \mathbb{A} \rightarrow \mathbb{B}$ be a functor. Form the pullback in \mathbf{Cat} :

$$\begin{array}{ccc} \mathbb{A} \times_{\mathbb{B}} \mathbb{E} & \longrightarrow & \mathbb{E} \\ K^*(p) \downarrow & \lrcorner & \downarrow p \\ \mathbb{A} & \xrightarrow{K} & \mathbb{B} \end{array}$$

In this situation, the functor $K^*(p)$ is also a fibration.

Example C.2. Consider the functor $K : \mathbf{Set} \times \mathbf{Set} \rightarrow \mathbf{Set}$ that sends a pair of sets to their product. In this case,

$$\begin{array}{ccc} \mathbf{BRel} & \longrightarrow & \mathbf{Pred} \\ K^*(p) \downarrow & \lrcorner & \downarrow p \\ \mathbf{Set} \times \mathbf{Set} & \xrightarrow{K} & \mathbf{Set} \end{array}$$

\mathbf{BRel} is a category of functions that preserve a binary relation. An object in \mathbf{BRel} is a binary relation $S \subseteq X \times Y$, and a morphism $S \subseteq X \times Y \rightarrow R \subseteq X_2 \times Y_2$ is a function $f : X \times Y \rightarrow X_2 \times Y_2$ such that for all $(x, y) \in S$, $f(x, y) \in R$.

Proposition C.2. (Composition, [38]) Let $p : \mathbb{E} \rightarrow \mathbb{B}$ and $r : \mathbb{B} \rightarrow \mathbb{A}$ be fibrations. Then $rp : \mathbb{E} \rightarrow \mathbb{A}$ is also a fibration, in which $f \in \mathbb{E}$ is a Cartesian morphism iff f is a Cartesian morphism for p and $p(f)$ is a Cartesian morphism for r .

For each $I \in \mathbb{A}$ one obtains a fibration $p_I : \mathbb{E}_I = (rp)^{-1}(I) \rightarrow \mathbb{B}_I = r^{-1}(I)$ by restriction.

Definition C.3 (Reproduced from [38]). Given a fibration $p : \mathbb{E} \rightarrow \mathbb{B}$, the fibre category over $I \in \mathbb{B}$ is the subcategory \mathbb{E}_I of \mathbb{E} whose objects are above I and morphisms above id_I .

Definition C.4. $p : \mathbb{E} \rightarrow \mathbb{B}$ is an **opfibration** if $p^{op} : \mathbb{E}^{op} \rightarrow \mathbb{B}^{op}$ is a fibration.

Example C.3. $p : \mathbf{Pred} \rightarrow \mathbf{Set}$ is an opfibration.

2) *Fibrations for logical relations for effectful languages:* We now recall the setting of fibrations for logical relations from [15].

Definition C.5 (Reproduced from [15]). A **partial order bifibration with fibrewise small products** is a faithful functor $p : \mathbb{E} \rightarrow \mathbb{B}$ such that

- p is a fibration
- p is an opfibration
- each fibre category is a partial order
- each fibre category has small products, and the inverse image functors (necessarily) preserve them

Definition C.6. A **fibration for logical relations over a bi-CCC** \mathbb{B} is a partial order bifibration $p : \mathbb{E} \rightarrow \mathbb{B}$ with fibrewise small products such that \mathbb{E} is a bi-CCC and p strictly preserves the bicartesian-closed structure.

Proposition C.3 ([15]). The pullback of a fibration for logical relations along a finite product preserving functor is a fibration for logical relations.

In particular, the usual subscone fibration is recovered as a change of base along the functor $\mathbb{B}(1, -) : \mathbb{B} \rightarrow \mathbf{Set}$.

Definition C.7 (Reproduced from [15]). Let $p : \mathbb{E} \rightarrow \mathbb{B}$ be a functor. Given $X, Y \in \mathbb{E}$, and $f : pX \rightarrow pY$, we write $f : X \dot{\rightarrow} Y$ to denote the following proposition: $\exists f : X \rightarrow Y. p(f) = f$. We say that f has a **lift** (in \mathbb{E}).

The setting of Katsumata [15] works for fairly general languages with effects. One way to describe them is in terms of algebraic operations, or equivalently in terms of generic effects.

Definition C.8 (Reproduced from [15]). Let \mathbf{C} be a category and T a strong monad on it. Given objects $C, D \in \mathbf{C}$, a (D, C) algebraic operation or generic effect for T is a morphism $C \rightarrow TD$.

A lambda-calculus with effect is parametrized by a set of base types and (effectful) primitives, each having an arity and coarity, describing the number and types of arguments and return values. This is encapsulated as a signature:

Definition C.9 (Reproduced from [15]). A λ_c -signature Σ is a tuple $(B, K, O, ar, car : K \cup O \rightarrow GType(B))$ where B is a set of base types, K a set of effect-free constants, O a set of algebraic operations, and $GType(B)$ the set of ground types on B . ar, car are the arity and co-arity functions.

Definition C.10 (Reproduced from [15]). Let Σ be a signature. A $\lambda_c(\Sigma)$ -structure is a tuple $A = (\mathbb{B}, T, A, a)$ where \mathbb{B} is a bi-CCC, T a strong monad on \mathbb{B} , A a functor $B \rightarrow \mathbb{B}$.

In other words, a $\lambda_c(\Sigma)$ -structure gives an interpretation of the base types and primitives of the language. Note that we can always extend A to a structure preserving functor $A[-] : GType(B) \rightarrow \mathbb{B}$.

We now recall the main theorems from [15]:

Definition C.11 (Reproduced from [15]). A property over a $\lambda_c(\Sigma)$ -structure A is a pair (V, C) of functors $V, C : B \rightarrow \mathbb{E}$ such that $p \circ V = A$ and $p \circ C = T \circ A$.

That is, a property is the choice for any base type $b \in B$ of a predicate Vb in the category of predicates \mathbb{E} over values of type b , and of a predicate Cb over computations $T(Ab)$ of type b . The question of interest is then whether all programs preserve the property, which is answered with the following theorem.

Theorem C.4 (Logical relations, [15]). Let Σ be a signature, $A = (\mathbb{B}, T, A, a)$ be a $\lambda_c(\Sigma)$ -structure, $p : \mathbb{E} \rightarrow \mathbb{B}$ be a fibration for logical relations and (V, C) be a property over A . If the property satisfies the following conditions

- for all $b \in B$, **return** has a lift
- for all $k \in K$, $A[k]$ has a lift
- for all $o \in O$, $A[o]$ has a lift

then for all well-typed terms $x_1 : b_1, \dots, x_n : b_n \vdash t : b$, $A[t]$ lifts to the total category.

3) *Fibrations for logical relations for effectful languages with recursion:* So far, we have reviewed the way to use fibrations for logical relations for higher-order effectful languages, but not for languages with recursion. We review this next.

Definition C.12 (Reproduced from [15]). An ω cpo-enriched bi-CCC is a bi-CCC \mathbf{C} such that each homset is an ω cpo, and the composition, tupling $(-, -)$, cotupling $[-, -]$ and

currying $\lambda(-)$ of the bi-CC structure on \mathbf{C} are all monotone and continuous.

Definition C.13 (Reproduced from [15]). A pseudo-lifting monad on an ω cpo-enriched bi-CC category \mathbb{B} is a monad on the underlying non-enriched category such that it has an algebraic operation bt such that its component at I , $bt_I : 1 \rightarrow TI$, is the least morphism.

Definition C.14 (Reproduced from [15]). An ω cpo-enriched $\lambda_c(\Sigma)$ -structure is a tuple (\mathbb{B}, T, A, a) such that \mathbb{B} is an ω cpo-enriched bi-CCC, T is a pseudo-lifting monad over \mathbb{B} and (\mathbb{B}, T, A, a) is a $\lambda_c(\Sigma)$ -structure.

Definition C.15 (Reproduced from [15]). We call $X \in \mathbb{E}$ above $TI \in \mathbb{B}$ admissible if

- $\perp_{pX} : 1 \rightarrow X$
- for all $Y \in \mathbb{E}$ and ω -chain $f_i \in \mathbb{B}(pY, pX)$ such that $f_i : Y \rightarrow X$, we have $\bigsqcup_{i=0}^{\infty} f_i : Y \rightarrow X$.

Theorem C.5 (Logical relations for language with recursion, [15]). Let Σ be a signature, $A = (\mathbb{B}, T, A, a)$ an ω cpo-enriched $\lambda_c(\Sigma)$ -structure, $p : \mathbb{E} \rightarrow \mathbb{B}$ a fibration for logical relations and (V, C) be a property over A . If the property satisfies

- the conditions of Theorem C.4
- for all $b \in \mathbb{B}$, Cb is admissible

Then for all well-typed terms of the language with iteration, $x_1 : b_1, \dots, x_n : b_n \vdash t : b$ we have $A[t] : \prod_i Vb_i \rightarrow Cb$.

B. *Correctness of AD*

We will show correctness of AD (Theorem III.3) via the theory of fibrations for logical relations. The goal is to construct a category of predicates on ω PAP and to encode correctness of AD as a predicate preservation property. To construct such a fibrations for logical relations, we obtain one by pulling back another one along a product preserving functor. There are several possibilities that give us different sort of predicates on ω PAP spaces. One constraint is that we need to be able to interpret our choice of logical predicate for AD as such a predicate, and make sure all of our constants are predicate preserving. This is in particular non-trivial when dealing with constants involving coproducts, such as $< : \mathbf{real} \times \mathbf{real} \rightarrow \mathbb{B}$ as the booleans are interpreted as the coproduct $1 + 1$. Lastly, we need to make sure that our choice of predicate for base types is closed under lubs of ω -chains.

We organize this section as follows. First, we construct a suitable category of predicates on ω PAP \times ω PAP (i.e. a fibration for logical relations). Second, we construct a property (Definition C.11) on this category of predicates. Note that the fact that derivatives are not unique any more is what will make this logical relation more complex. Lastly, we will show the correctness of AD by combining these elements and Theorem C.5.

1) *A fibration for logical relations on ω PAP \times ω PAP:*

Definition C.16. We define the category **Inj**, a subcategory of c PAP, whose objects are c -analytic sets and morphisms are PAP injections.

Lemma C.6. *Inj* is a site, with coverage \mathcal{J} where coverings are given as in cPAP.

Proof. The proof for the case of cPAP carries through to this restricted setting. \square

Definition C.17. We define $\mathbf{Sh}(\mathbf{Inj})$ to be the category of sheaves on the site \mathbf{Inj} .

Lemma C.7. $\mathbf{Sh}(\mathbf{Inj})$ is a Grothendieck topos, and in particular it is Cartesian-closed, complete and co-complete.

Proof. This is standard sheaf theory, see e.g. MacLane and Moerdijk [41]. \square

By a subsheaf P of F , we mean a sheaf F such that for all c -analytic set A , $P(A) \subseteq F(A)$, and $P(f)$ is the restriction of $F(f)$ for all morphisms f .

Definition C.18. We denote by $\mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj}))$ the category whose objects are pairs of sheaves (P, F) of $\mathbf{Sh}(\mathbf{Inj})$, where P is a subsheaf of F , and morphisms $(P, F) \rightarrow (P', F')$ are natural transformations $F \rightarrow F'$ that restrict to the subsheaves.

Lemma C.8. The second projection $(P, F) \mapsto F$ is a fibrations for logical relations $p : \mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj})) \rightarrow \mathbf{Sh}(\mathbf{Inj})$.

Proof. As $\mathbf{Sh}(\mathbf{Inj})$ has pullbacks, it is a fibration. It is clearly faithful. By the theory of glueing ([39, 42, 43, 44]), as $\mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj}))$ is a CCC and has an epi-mono factorisation system, $\mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj}))$ is a CCC, has finite colimits, and p preserves the CCC-structure. In addition, each fibre category is a partial order and has small products. It remains to show that cod is an opfibration and that p preserves coproducts. By Lemma 4.6 in Kammar and McDermott [45], it suffices to show that monos are closed under coproducts, as we already have the other conditions for having a factorization system for logical relations. The arrow category on $\mathbf{Sh}(\mathbf{Inj})$ is cocomplete, as $\mathbf{Sh}(\mathbf{Inj})$ is (it is a topos). As $\mathbf{Sh}(\mathbf{Inj})$ is a topos, it has a epi-mono factorization system, and therefore it has image factorization. Therefore, the category of monos is a reflective subcategory of the arrow category. As such, it is cocomplete, and in particular it has coproducts. So monos are closed under coproducts, and we conclude by Lemma 4.6 in Kammar and McDermott [45], as said above. \square

We now define the functor $F : \omega\text{PAP} \times \omega\text{PAP} \rightarrow \mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj}))$ given by $F(X, Y) = X \times \prod_{i \in \mathbb{N}} Y$, where $X \times \prod_{i \in \mathbb{N}} Y$ forgets that it is an ω -concrete sheaf on the site cPAP, and only remembers that it is in particular a sheaf on the site \mathbf{Inj} . It is a product preserving functor, and therefore, by Proposition C.3, the pullback of $p : \mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj})) \rightarrow \mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj}))$ along F is a fibration for logical relations. We denote this pullback by $\pi : \mathbf{Gl} \rightarrow \omega\text{PAP} \times \omega\text{PAP}$.

Remark C.1. Before moving on, let us say a few words on this category of predicates \mathbf{Gl} . One reason why we defined the category of sheaves $\mathbf{Sub}(\mathbf{Sh}(\mathbf{Inj}))$ is that the property we will define for the correctness of AD will not be a concrete subsheaf, and this excludes directly using subobjects on ωPAP

as the fibration for logical relations of interest. Secondly, the base type \mathbb{B} is interpreted as a coproduct, and if we simply choose the category of presheaves on \mathbf{Inj} , primitives like $>$ will not lift, for a similar reason to the one presented in Vákár [4]. We could have chosen sheaves on cPAP instead. This would not change much, but this is a bit overkill for our purpose and simply adds some unnecessary complexity. Thirdly, we do seem to need to have the property indexed by all c -analytic sets A . One reason is that the fact that we have partial functions forces the definition of a lift for \mathbf{LR} , as well as for \mathbb{R} . A second reason is that the non-uniqueness of intentional derivatives creates a technical complication: being a correct intentional derivative at a point is essentially non-informative, so we cannot use the same trick as we did in the smooth case. That is, the problem is that the input space \mathbb{R} is smaller than \mathbb{R}^n , and the quantification only shows that h is ‘correct’ on a set of at most the size of \mathbb{R} inside \mathbb{R}^n , which is Lebesgue-negligible. This problem disappears when we can show that h is correct on each open of an open cover of its domain.

2) A property for the correctness of AD:

Definition C.19. Let $f : A \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$ a PAP function. Given any intentional derivative $g : A \rightarrow \mathbb{R}^n$ of f , we call $\pi_i \circ g : A \rightarrow \mathbb{R}$ an i -th partial intentional derivative of f .

We write hot_i for the vector in \mathbb{R}^n that consist only of zeros, except it has a single 1 at the i -th position. Given any dual-number intentional representation $g : A \times \mathbb{R}^n \rightarrow \mathbb{R}^2$ of f , define the PAP function $g_i : A \rightarrow \mathbb{R}^2$ given by

$$g_i(x) = g(x, \text{hot}_i)$$

g_i is called an i -th partial dual-number intentional representation of f , and its second component is an i -th partial intentional derivative of f . We write $\partial_{\text{DNR}}^i f$ for the set of i -th partial dual-number intentional representations of f .

We extend the definition of $\partial_{\text{DNR}}^i f$ to all natural numbers i by setting $\partial_{\text{DNR}}^i f = \{\lambda x.0\}$ for any $i > n$.

Definition C.20. Let $V(\mathbb{R}) := (\mathbb{R}, \prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R}, V_{\mathbb{R}})$ where for each c -analytic set A , we have

$$V_{\mathbb{R}}(A) = \{(f : A \rightarrow \mathbb{R}, (g_i : A \rightarrow \mathbb{R}^2)_{i \in \mathbb{N}}) \mid f, g \text{ PAP, and } g_i \in \partial_{\text{DNR}}^i f\}$$

Likewise, let $C(\mathbb{R}) := (\mathbf{LR}, \mathbf{L}(\prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R}), C_{\mathbb{R}})$ where for each c -analytic set A , we have

$$C_{\mathbb{R}}(A) = \{(f : A \rightarrow \mathbf{LR}, g : A \rightarrow \mathbf{L}(\prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R})) \mid \text{Dom}(f) = \text{Dom}(g) \text{ and } \forall i. (\widetilde{f}, \widetilde{L\pi_i \circ g}) \in V_{\mathbb{R}}(\text{Dom}(f))\}$$

Let $V(\mathbb{B}) = (\mathbb{B}, \prod_{i \in \mathbb{N}} \mathbb{B}, V_{\mathbb{B}})$ where for each c -analytic set A , we have

$$V_{\mathbb{B}}(A) = \{f : A \rightarrow \mathbb{B}, g : A \rightarrow \prod_{i \in \mathbb{N}} \mathbb{B} \mid \forall i, \pi_i \circ g = f\}$$

Let $C(\mathbb{B}) = (\mathbf{L}\mathbb{B}, \mathbf{L}(\prod_{i \in \mathbb{N}} \mathbb{B}), C_{\mathbb{B}})$ where for each c-analytic set A , we have

$$C_{\mathbb{B}}(A) = \{(f : A \rightarrow \mathbf{L}\mathbb{B}, g : A \rightarrow \mathbf{L}(\prod_{i \in \mathbb{N}} \mathbb{B})) \mid \\ \text{Dom}(f) = \text{Dom}(g) \text{ and} \\ \forall i. (\widetilde{f}, \widetilde{L\pi_i \circ g}) \in V_{\mathbb{B}}(\text{Dom}(f))\}$$

Lemma C.9. *We have defined a property for our language.*

Proof. The only thing to show is that $V_{\mathbb{R}}, C_{\mathbb{R}}, V_{\mathbb{B}}, C_{\mathbb{B}}$ are subsheaves of $\mathbb{R} \times \prod_{i \in \mathbb{N}} \mathbb{R}^2, \mathbf{L}\mathbb{R} \times \mathbf{L} \prod_{i \in \mathbb{N}} \mathbb{R}^2, \mathbb{B} \times \prod_{i \in \mathbb{N}} \mathbb{B}, \mathbf{L}\mathbb{B} \times \mathbf{L} \prod_{i \in \mathbb{N}} \mathbb{B}$ respectively. It is immediate for the C 's once we have shown the result for the V 's. For $V_{\mathbb{R}}$, it amounts to showing two things. First, whether partial intentional derivatives restrict to a subset of the domain of a function, which is evidently true. Second, whether partial intentional derivatives glue if they agree on their intersection, which as for usual derivatives, is true as well. For $V_{\mathbb{B}}$, it is immediate as it is an equalizer of $\mathbb{B} \times \prod_{i \in \mathbb{N}} \mathbb{B}$, which exists in any topos and the canonical map $V_{\mathbb{B}} \rightarrow \mathbb{B} \times \prod_{i \in \mathbb{N}} \mathbb{B}$ is a mono. \square

Let us now look a bit more into what morphisms preserving the property look like.

Definition C.21. Let $V(\mathbb{R}^k) := (\mathbb{R}^k, (\prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R})^k, V_{\mathbb{R}^k})$ where for each c-analytic set A , we have

$$V_{\mathbb{R}^k}(A) = \{(f : A \rightarrow \mathbb{R}^k, g : A \rightarrow (\prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R})^k) \mid \\ \forall 1 \leq j \leq k, (\pi_j f, \pi_j g) \in V_{\mathbb{R}}(A)\}$$

Let $f : \mathbb{R}^n \rightarrow \mathbf{L}\mathbb{R}$ represent a PAP function. We define $V(f)$ to be pairs of functions (f, h) , where $h : (\prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R})^k \rightarrow \mathbf{L}(\prod_{i \in \mathbb{N}} \mathbb{R} \times \mathbb{R})$ is ω PAP, and such that for any c-analytic set A , and any $(g_1, g_2) \in V_{\mathbb{R}^k}(A)$, $(f \circ g_1, h \circ g_2) \in C_{\mathbb{R}}(A)$.

This means that given a PAP functions $g_1, \dots, g_n : A \rightarrow \mathbb{R}$, h will send i -th partial intentional representations of each of the g_i to i -th partial intentional representations of $f \circ \langle g_1, \dots, g_n \rangle$. Note that if $A \subseteq \mathbb{R}^k$, this will be trivially satisfied for any $i > k$.

Let us now see how we will interpret and a lift primitive $t : \mathbf{real}^2 \rightarrow \mathbf{real}$ of our language. This interpretation will only be required for the correctness proof. We set $\llbracket t \rrbracket_{\omega\text{PAP} \times \omega\text{PAP}} = (\llbracket t \rrbracket, \prod_{i \in \mathbb{N}} \llbracket \mathcal{D}(t) \rrbracket)$. In other words, we make countably many copies of the semantics of the AD-translation of t . We can now more generally show that every primitive of the language, whose semantics is interpreted in $\omega\text{PAP} \times \omega\text{PAP}$ following the example above, will lift to **G1**.

Lemma C.10. *Every primitive of the language has a lift in **G1**.*

Finally, to satisfy the conditions of Theorem C.5, we need to show that for each base B , $C(B)$ is admissible.

Lemma C.11. *$C(\mathbb{B})$ and $C(\mathbb{R})$ are admissible.*

3) *Correctness of AD:*

Lemma C.12. *ω PAP is an ω cpo-enriched bi-CCC.*

Proof. Each homset is an ω cpo by forgetting the PAP-structure. For composition and coproducts, it is immediate as lubs are taken point-wise. For products, if $f = \bigvee_i f_i$ and $g = \bigvee_i g_i$, then $(f, g) = (\bigvee_i f_i, \bigvee_i g_i) = \bigvee_i (f_i, g_i)$. Finally, if $f_i : X \times Y \rightarrow Z$, for currying we have $(\bigvee f_i)(x, y) = \bigvee f_i(x, y) = \bigvee \lambda(f_i)(x)(y)$. As lubs are taken pointwise, we have $\bigvee \lambda(f_i)(x)(y) = (\bigvee \lambda(f_i)(x))(y)$. This means that $\lambda(\bigvee f_i) = \bigvee \lambda(f_i)$. The monotonicity part is similar.

In summary, it follows from the fact that the forgetful functor $\omega\text{PAP} \rightarrow \omega\text{CPO}$ preserves and reflects limits, coproducts, exponentials, and the ω cpo-structure. \square

Lemma C.13. *Let $f : A \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$ be PAP. Then $h : A \times \mathbb{R}^n \rightarrow \mathbb{R}^2$ is a dual-number representation of f iff for all natural number i , $h(x, \text{hot}_i)$ is an i -th partial dual number representation of f .*

Theorem C.14. *For any $x_1 : \mathbf{real}, \dots, x_n : \mathbf{real} \vdash t : \mathbf{real}$, $\llbracket t \rrbracket_{\omega\text{PAP} \times \omega\text{PAP}}$ has a lift.*

Proof. We have done all the work in the previous subsections. We simply check the condition to apply Theorem C.5.

First, note that we indeed have a semantics in $\omega\text{PAP} \times \omega\text{PAP}$, given by $\llbracket t \rrbracket_{\omega\text{PAP} \times \omega\text{PAP}} := (\llbracket t \rrbracket, \prod_{i \in \mathbb{N}} \llbracket \mathcal{D}(t) \rrbracket)$, extending what we have seen in Section C-B2. Then,

- We have defined a fibrations for logical-relations $\pi : \mathbf{G1} \rightarrow \omega\text{PAP} \times \omega\text{PAP}$ in Section C-B1.
- We have defined a property (V, C) in **G1** in Section C-B2.
- We check that $x \mapsto \text{return}(x)$ lifts, but this is immediate.
- Each primitive has a lift, by Lemma C.10.
- ωPAP is an ω cpo-enriched bi-CCC by Lemma C.12.
- \mathbf{L} is a pseudo-lifting monad. It means that we have an algebraic element $\perp_A : 1 \rightarrow \mathbf{L}A$ that is interpreted in $\mathbf{L}A$ as the bottom element. This is true by construction.
- $C(\mathbb{R})$ and $C(\mathbb{B})$ are admissible, by Lemma C.11.

\square

Corollary C.14.1 (Correctness of AD (limited)). *For any term $x_1 : \mathbf{real}, \dots, x_n : \mathbf{real} \vdash t : \mathbf{real}$, the PAP function $\llbracket \mathcal{D}(t) \rrbracket : A \times \mathbb{R}^n \rightarrow \mathbb{R} \times \mathbb{R}$ is a dual-number intentional representation of $\llbracket t \rrbracket : A \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$.*

Proof. Let $x_1 : \mathbf{real}, \dots, x_n : \mathbf{real} \vdash t : \mathbf{real}$. By Theorem C.14, $\llbracket t \rrbracket_{\omega\text{PAP} \times \omega\text{PAP}}$ has a lift. Consider the c-analytic set \mathbb{R}^n and the pair

$$(id_{\mathbb{R}^n}, \lambda x. \prod_{i \leq k \leq n} \prod_{i \in \mathbb{N}} (\pi_k(x), [i = k] \pi'_i(x)))$$

where, by convention, $\pi_i(x) = \pi'_i(x) = 0$ whenever $i > n$. We use Iverson brackets, $[i = k]$, which equals to 1 when the

condition is satisfied, and 0 otherwise. This pair is an element of $V_{\mathbb{R}^k}(\mathbb{R}^n)$. This means that

$$\left(\llbracket t \rrbracket, \prod_{i \leq k \leq n} \prod_{i \in \mathbb{N}} [\mathcal{D}(t)] \circ (\lambda x. \prod_{1 \leq k \leq n} (\pi_k(x), [i = k] \pi'_i(x))) \right) \in C_{\mathbb{R}}(\mathbb{R}^n)$$

xw This simplifies a bit more, and by the definition of $C_{\mathbb{R}}(\mathbb{R}^n)$, this means that for all i , $\llbracket \mathcal{D}(t) \rrbracket(x, \text{hot}_i)$ is an i -th intentional partial representation of $\llbracket t \rrbracket$. By Lemma C.13, this means that $\llbracket \mathcal{D}(t) \rrbracket$ is a dual-number intentional representation for $\llbracket t \rrbracket$, as desired. \square

With the same proof, Theorem C.14 gives us directly the stronger result at all ground types:

Corollary C.14.2 (Correctness of AD (full)). *For any term well-typed term $\Gamma \vdash t : \tau$ of ground type τ in a ground context Γ , the PAP function $\llbracket \mathcal{D}(t) \rrbracket : A \times \mathbb{R}^n \rightarrow \mathbb{R} \times \mathbb{R}$ is a dual-number intentional representation of $\llbracket t \rrbracket : A \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$.*

APPENDIX D

FAILURE AND CONVERGENCE OF GRADIENT DESCENT

A. Failure of gradient descent for PAP functions

Proof. Let $\epsilon > 0$. Let P be the closed program defined by the following code, where lr is our chosen learning rate ϵ .

```
# Recursive helper
def g(x, n):
    if x > 0:
        return ((x - n) * (x - n)) / (lr * 2)
    if x == 0:
        return x / lr + n * n / (2 * lr)
    else:
        return g(x+1, n+1)
```

```
def P(x) = g(x, 0)
```

This can easily be written in our language with recursion and conditionals, but it would slightly impact readability.

$\llbracket P \rrbracket$ is PAP, and by inspection, we see that $\llbracket P \rrbracket = x \mapsto \frac{x^2}{2\epsilon}$. This is proved by simple induction on \mathbb{N} in inner the recursive function g defining P . Let's now show that $f := \llbracket P \rrbracket$ satisfies the hypothesis of Theorem III.4 and is strictly convex:

- f is strictly convex: this is well-known for the x^2 function
- f is bounded below: $f \geq 0$
- f is L -smooth for some L :

$$\nabla f(x) - \nabla f(y) = \frac{2x}{2\epsilon} - \frac{2y}{2\epsilon} = \frac{1}{\epsilon}(x - y)$$

And therefore f is $\frac{1}{\epsilon}$ -smooth.

This means that for all $0 < \epsilon' < \frac{2}{\frac{1}{\epsilon}} = 2\epsilon$, gradient descent on f converges to the global minimum 0. Now, let us see that this is not the case when we use the intentional derivatives for f computed by AD, in the case $\epsilon' := \epsilon$.

Applying our standard AD macro to P , we obtain a PAP function P' that returns $\frac{x}{\epsilon}$ when $x \in \mathbb{R}_{>0}$, and $\frac{(x+|x|+1)-(|x|+1)}{\epsilon}$ when $x \in \mathbb{R}_{<0} - \{-n \mid n \in \mathbb{N}\}$, and $\frac{1}{\epsilon}$ for $x \in \{-n \mid n \in \mathbb{N}\}$.

Let $x^0 \in \mathbb{R}$ be any initialization for the gradient descent algorithm. Then, after one step of gradient descent, we obtain $x^1 := x^0 - \epsilon \times P'(x^0)$. If $x^0 > 0$, then $\llbracket P'(x^0) \rrbracket = \frac{x^0}{\epsilon}$ and thus $x^1 = 0$. When $x^0 < 0$ is not an integer, $\llbracket P'(x^0) \rrbracket = \frac{x^0}{\epsilon}$ as thus $x^1 = 0$ as well. Finally, when $x^0 < 0$ is an integer, $\llbracket P'(x^0) \rrbracket = \frac{1}{\epsilon}$ and therefore $x^1 = x^0 - 1$. In any case, x^1 is a non-positive integer, and for x^2 we'll always have $x^2 = x^1 - 1$. Therefore, $x^t \rightarrow -\infty$ as $t \rightarrow \infty$.

A similar program P can be constructed when the learning rate is non-fixed. The recursive program inside P 's definition simply calls $\epsilon(n)$ instead of using ϵ , and the analysis will be very similar, except that x^1 will not necessarily be an integer when x^0 is a non-positive integer, but this happens for almost no x^0 . \square

B. Almost-sure convergence of Gradient Descent

Proof. Given a fixed intentional derivative ∇f of f , let $F_{int}(\epsilon, x) = (\epsilon, x - \epsilon \nabla_x f)$. Let $F_{true}(\epsilon, x) = (\epsilon, x - \epsilon \nabla_x^* f)$ where $\nabla_x^* f$ is the true gradient of the differentiable function f . We restrict the domain of F_{true} and F_{int} to $(0, \frac{2}{L}) \times \mathbb{R}^d$. For $\delta > 0$ and $t \in \mathbb{N}$, we define

$$X_{t,\delta}^* := \{(\epsilon, x^0) \in (0, \frac{2}{L}) \times \mathbb{R}^d \mid |\nabla^* f(\pi_2 F_{true}^t(\epsilon, x^0))| \leq \delta\}$$

where g^t is the t -fold composition of a function g with itself. As f is L -smooth, $\nabla^* f$ is locally L -Lipschitz and thus continuous. Therefore, by the first conclusion of Theorem III.4, for any $\delta > 0$,

$$\bigcup_{t=0}^{\infty} X_{t,\delta}^* = (0, \frac{2}{L}) \times \mathbb{R}^d$$

We define similarly, for any $\delta > 0$,

$$X_{t,\delta} := \{(\epsilon, x^0) \in (0, \frac{2}{L}) \times \mathbb{R}^d \mid |\nabla f(\pi_2 F_{int}^t(\epsilon, x^0))| \leq \delta\}$$

We will show by induction on $t \in \mathbb{N}$ that $X_{t,\delta}$ and $X_{t,\delta}^*$ only differ by a null-set, i.e. that there exists null-sets $N_{t,\delta}$ and $M_{t,\delta}$ such that $X_{t,\delta} \cap N_{t,\delta} = X_{t,\delta}^* \cap M_{t,\delta}$. This implies that

$$\begin{aligned} \bigcup_{t=0}^{\infty} X_{t,\delta} &\supseteq \bigcup_{t=0}^{\infty} X_{t,\delta} \cap N_{t,\delta} \\ &= \bigcup_{t=0}^{\infty} X_{t,\delta}^* \cap M_{t,\delta} \\ &\supseteq \left(\bigcup_{t=0}^{\infty} X_{t,\delta}^* \right) \cap \left((0, \frac{2}{L}) \times \mathbb{R}^d - \bigcup_{t=0}^{\infty} M_{t,\delta} \right) \\ &= (0, \frac{2}{L}) \times \mathbb{R}^d \cap \left((0, \frac{2}{L}) \times \mathbb{R}^d - \bigcup_{t=0}^{\infty} M_{t,\delta} \right) \\ &= (0, \frac{2}{L}) \times \mathbb{R}^d - \bigcup_{t=0}^{\infty} M_{t,\delta} \end{aligned}$$

As a countable union of negligible sets is negligible, this means that $\bigcup_{t=0}^{\infty} X_{t,\delta}$ equals $(0, \frac{2}{L}) \times \mathbb{R}^d$ up to a negligible set, as desired.

By induction on $t \in \mathbb{N}$, we show the following property:

$$\forall A \subseteq (0, \frac{2}{L}) \times \mathbb{R}^d \text{ measurable of measure 0,} \\ \{(\epsilon, x) \mid F_{true}^t(\epsilon, x) \in A\} \text{ is measurable with measure 0}$$

If $t = 0$, F_{true}^t is the identity and it's obvious. Let $t \in \mathbb{N}$ such that the property holds. Let $A \subseteq (0, \frac{2}{L}) \times \mathbb{R}^d$ be of measure 0. To use the induction hypothesis, it suffices to show that $F_{true}^{-1}(A)$ has measure 0. The Jacobian matrix of F is an $(n+1) \times (n+1)$ matrix, given by

$$J(F_{true})_{\epsilon, x} = \begin{pmatrix} 1 & 0_n^T \\ Jf_x^T & I_n - \epsilon Hf_x \end{pmatrix}$$

where I_n is the $n \times n$ identity matrix, 0_n the zero vector in \mathbb{R}^n , Hf_x is the Hessian matrix of f , and $(-)^T$ is the transpose operation.

The rank of $J(F_{true})_{\epsilon, x}$ is lower bounded by one plus the rank of $I_n - \epsilon Hf_x$. The latter is strictly less than n exactly when $\frac{1}{\epsilon}$ is an eigenvalue of Hf_x . For a fixed x , this can only occur for finitely many ϵ (at most n). Thus, $J(F_{true})_{\epsilon, x}$ has max rank $n+1$ almost everywhere. Denote by R this set. f being PAP, F is PAP as well. Therefore, F is almost everywhere analytic, and denote this set C . Let $N = R \cap C$, and let Z be its complement. Note that Z is negligible. On N , F is $F \in \mathcal{C}^1$ and its Jacobian is invertible. Therefore, by the implicit function theorem, the result is locally true around any point $x \in N$. We can use a countable cover $\{U_i\}_i$ of N , and thus

$$\begin{aligned} F_{true}^{-1}(A) &= F_{true}^{-1}(A) \cap N \cup F^{-1}(A) \cap Z \\ &\subseteq F_{true}^{-1}(A) \cap \left(\bigcup_i U_i \right) \cup Z \\ &= \bigcup_i F_{true}^{-1}(A) \cap U_i \cup Z \end{aligned}$$

$F_{true}^{-1}(A)$ is obviously measurable as A is, and as null-sets are closed under countable unions, this shows that $F_{true}^{-1}(A)$ has measure 0, which concludes the induction.

Now, we show by induction on $t \in \mathbb{N}$ that for almost all $(\epsilon, x^0) \in (0, \frac{2}{L}) \times \mathbb{R}^d$,

$$F_{true}^t(\epsilon, x^0) = F_{int}^t(\epsilon, x^0)$$

Once again, the base case is immediate. Let $t \in \mathbb{N}$ such that the property holds. As ∇f is an intentional derivative of f , for almost all $x \in \mathbb{R}$, we have that $\nabla_x f = \nabla_x^* f$. Therefore, for almost all (ϵ, x) , we have that $F_{true}(\epsilon, x) = F_{int}(\epsilon, x)$. Let A be the set of points on which they differ. By the previous proposition, $F_{true}^{-1}(A)$ has measure 0. Therefore, F_{true}^{t+1} and F_{int}^{t+1} may differ only on the set $F_{true}^{-1}(A)$, which has measure 0. This concludes the induction.

As a corollary of what we have just shown, we have that for almost all (ϵ, x^0) , x^t is the same, whether it was obtained from true or AD-computed gradients. Indeed, x^t is either $\pi_2 F_{true}^t(\epsilon, x^0)$ or $\pi_2 F_{int}^t(\epsilon, x^0)$, and we have shown that $F_{true}^t(\epsilon, x^0)$ and $F_{int}^t(\epsilon, x^0)$ agree almost everywhere. Therefore $f(x^{t+1}) \leq f(x^t)$ remains true

almost-everywhere for all t , as a countable union of negligible sets is negligible. Finally, if f is strongly convex, then $\{x^t\}_t$ will reach the global minimum. As x^t is the same, whether it was obtained from true or AD-computed gradients, for almost all (ϵ, x^0) , this means that gradient-descent with AD-computed gradients will also almost surely converge to the global minimum. \square

APPENDIX E S-HAUSDORFF MEASURES

A. Monad of measure on ω PAP

The goal of this section is to give the key steps in the proof of Theorem IV.2. These are the following. We adapt the proof technique used in Vákár et al. [2]. We define an expectation operator $\text{Int}_X : \mathbf{S}X \rightarrow (X \rightarrow \mathbb{W}) \rightarrow \mathbb{W}$. We show that it is a well-defined ω PAP-morphism, and natural in X . Next, we show that ω PAP admits a proper orthogonal-factorization system, extending the known result on categories of concrete sheaves to this category of ω -concrete sheaves. Using a theorem from Kammar and McDermott [45], we show that the image of Int induces a strong monad \mathbf{M} on ω PAP. This monad is easily shown to be commutative.

B. Pushforward of PAP functions

In this section, we prove the following Theorem:

Theorem E.1. *Let $f : U \rightarrow \mathbb{R}^n$ be the total restriction of a partial ω PAP morphism to its domain $U \subseteq \mathbb{R}^m$. Let λ be the restriction of the Lebesgue measure to U . Then $f_*\lambda$ has a density w.r.t. some s -Hausdorff measure μ on \mathbb{R}^n .*

It is the core technical theorem that is required to prove Theorem IV.3. To prove Theorem E.1, we will first prove a restricted form of it:

Theorem E.2. *Let $f : U \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$ be real analytic on a connected open U . Then there is an integer k such that $f(U)$ is contained in a countable union $M := \bigcup_{i \in \mathbb{N}} M_i$ of smooth submanifolds M_i of \mathbb{R}^m of dimension k , up to a set of k -Hausdorff measure 0. Furthermore, $f_*\lambda$ has a density w.r.t. the k -Hausdorff measure on M .*

Before proving the theorem, we need a few other technical lemmas.

Lemma E.3. *Let $\mu = \sum_{0 \leq k \leq m} \sum_{i \in \mathbb{N}} f_i^k \cdot \mu_i^k \in \mathcal{M}\mathbb{R}^m$ for some non-negative measurable functions $f_i^k : \mathbb{R}^m \rightarrow \mathbb{R}^+ \cup \{\infty\}$, and such that μ_i^k is the k -Hausdorff measure on some k -dimensional submanifold M_i^k of \mathbb{R}^m . Let B be the s -Hausdorff measure given by the (μ_i^k) . Then μ has a density w.r.t. B .*

Proof. Let $X_k = \bigcup_i M_i^k$ and μ_k be the k -Hausdorff measure on X_k . Let $g_k = \sum_{i \in \mathbb{N}} 1_{x \in M_i^k} \cdot f_i^k$. As the f_i^k are non-negative and can take the value ∞ , the limit exists and therefore the g_k are measurable. By construction, we have the equality $\sum_{i \in \mathbb{N}} f_i^k \cdot \mu_i^k = g_k \cdot \mu_k$ for all $0 \leq k \leq m$. This follows from the simpler equality $f_1 \cdot \nu|_A + f_2 \cdot \nu|_B = (f_1 + f_2) \cdot \nu|_{A \cup B}$ which is also valid for countable unions, given that the μ_i^k

are all restrictions of the k -Hausdorff measure. Therefore $\mu = \sum_{1 \leq k \leq m} g_k \cdot \mu_k$.

Let $g = g_0 + 1_{x \notin X_0}(g_1 + 1_{x \notin X_1}(g_2 + 1_{x \notin X_2}(\dots + 1_{x \notin X_{m-1}}g_m)))$. g is clearly measurable and non-negative (with the convention that $0 \cdot \infty = 0$ here). For all $i < k$, X_i is H^k negligible. Thus $g_k \cdot \mu_k = g_k \cdot \mu_k|_{X_k - X_i} = (1_{x \notin X_i}g_k) \cdot \mu_k$. From this fact, we deduce that $\sum_{1 \leq k \leq m} g_k \cdot \mu_k = g \cdot B$, and therefore conclude that $\mu = g \cdot B$, i.e. that μ has a density w.r.t. B . \square

Lemma E.4. *Let $f : U \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$ be real analytic on a connected open U . Let k be the maximal rank of f on U . Then $V := \{x \in U \mid \text{rank}(f(x)) = k\}$ is*

- open,
- dense in U ,
- and $\lambda(U - V) = 0$.

Proof. Let's first show V is open. Given a coordinate system for U and V , for any $x \in V$, denote by $G(x)$ the matrix for $T_x f$ in these bases. By hypothesis, $G(x)$ has at least an invertible $k \times k$ sub-matrix, and this is a characterisation of rank. Then denote by $H : M_{n \times m}(\mathbb{R}) \rightarrow \mathbb{R} : A \mapsto \sum_{B \text{ } k \times k \text{ submatrix of } A} |\det(B)|$. H is continuous as the determinant is, and $V = (HG)^{-1}(\mathbb{R} - \{0\})$ is thus open. Let's show V is dense. Note that this is not true in general for smooth functions. Pick a base for U and V again, and let $G(x)$ the matrix for $T_x f$ in these bases. Thus $x \mapsto G(x)$ is also real analytic.

Let $p \in \bar{V} - V$. Pick a $k \times k$ submatrix B of matrices in $M_{n \times m}(\mathbb{R})$, then let $H_B : M_{n \times m}(\mathbb{R}) \rightarrow \mathbb{R}$ be the determinant of that submatrix. Assume $H_B G$ is 0 on an open neighbourhood W_B of p for all B . Then $W = \cap W_B$ is a finite intersection of opens and thus is open, and $W \cap V \neq \emptyset$ as by assumption p is in the boundary of V . But that's a contradiction as we know that for some B and any x in the intersection, $H_B G(x) \neq 0$. Therefore, there is a B such that $H_B G$ is not 0 at some point x in a neighbourhood W_B on p . As f is analytic, so is df , and so is G . The determinant is a polynomial function of its arguments, it is thus real analytic, and thus so is H_B . As $H_B G$ is analytic and non-identically 0 on W_B , it must be non-zero on a dense subset of W_B . Let's assume that the closure \bar{V} is not all of U . We obtain a contradiction by picking a point close to \bar{V} that would end up in one of the W_B . In more detail, assume by way of contradiction that $U - \bar{V} \neq \emptyset$. Let $p \in U - \bar{V}$. Let $g : U \rightarrow \mathbb{R}^+$ be defined as $g(x) = d(x, \bar{V})$. By assumption, as \bar{V} is closed, we have $g(p) > 0$. Also, note that g is continuous. As k is the maximal rank reached by f , $\bar{V} \neq \emptyset$. Thus, there exists $y \in \bar{V}$ such that $g(y) = 0$. As U is connected (and thus path connected, as an open of a Euclidean space), there is a continuous path $c : [0, 1] \rightarrow U$ such that $c(0) = y$ and $c(1) = p$. The composition $g \circ c$ is continuous and so there a point $x \in \bar{V} - V$ such that $c(t) = x$ and $g(x) = 0$ for some $t_0 < 1$. By compactness of $[0, 1]$, let's choose the biggest such t_0 , which exists as $c(1) = p \notin \bar{V}$. For all $t > t_0$, we therefore have $g(c(t)) > 0$. However, by the result above, there is a dense open subset W of a W_B , a

neighbourhood of x , such that $W \subseteq V$. By density of W , this means that $c(t) = 0$ on $c^{-1}(W_B) \cap [t_0, 1] \neq \emptyset$, a contradiction.

We will now prove that $X := U - V$ has Lebesgue measure 0. For a fixed $k \times k$ matrix B of any $n \times m$ matrix, consider the same analytic function H_B as above. Then $g_B := G; H_B : U \rightarrow \mathbb{R}$ is analytic. Using the theorem from Mityagin [46], it is either the 0 function or the preimage of 0 has Lebesgue measure 0. We can write $X = \bigcap_{B \text{ } k \times k \text{ submatrix}} g_B^{-1}(\{0\})$. As f has rank k somewhere, not every g_B is the 0 analytic function, and thus X is contained in a Lebesgue measure 0 set, and therefore has Lebesgue measure 0. \square

Lemma E.5. *Let $f : A \rightarrow C$ where $A \subseteq \mathbb{R}^n$ is λ -measurable, and let $U \subseteq A$ be λ -measurable. Assume that $\lambda(A - U) = 0$. Then $f_*\lambda = (f|_U)_*\lambda$.*

Proof.

$$\begin{aligned} f_*\lambda(S) &= \lambda(f^{-1}(S)) \\ &= \lambda(f^{-1}(S) \cap A) \\ &= \lambda(f^{-1}(S) \cap (A - U + U)) \\ &= \lambda((f^{-1}(S) \cap (A - U)) \cup (f^{-1}(S) \cap U)) \\ &\leq \lambda(f^{-1}(S) \cap (A - U)) + \lambda(f^{-1}(S) \cap U) \\ &\leq \lambda(A - U) + \lambda|_U(f^{-1}(S)) \\ &= 0 + \lambda|_U(f^{-1}(S)) \\ &= (f|_U)_*\lambda(S) \end{aligned}$$

As the other inequality is obvious, we showed that $f_*\lambda(S) = (f|_U)_*\lambda(S)$. \square

Lemma E.6. *Let $f : U \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^m$ be real analytic on a connected open U . Further, assume f has constant rank. Then $f(U)$ is a countable union of manifolds.*

Proof. As f has constant rank, by the constant rank theorem, f is locally of the form $(x_1, \dots, x_n) \mapsto (x_1, \dots, x_k, 0, \dots, 0)$. More precisely, around any $x \in U$, there is a neighborhood W of x and smooth diffeomorphisms ϕ, ψ such that $g := \phi \circ f|_W \circ \psi(x_1, \dots, x_n) = (x_1, \dots, x_k, 0, \dots, 0)$. As g is a projection, it is an open map. As ϕ, ψ are diffeomorphisms, $f|_W$ is also an open map. This means that $f(W)$ is a k -dimensional submanifold of \mathbb{R}^m (technically it is direct if $k = m$, otherwise we need an extra argument where we post-compose with a projection first, obtain the result above, and then use the submersion theorem to obtain that the 0 level of the projection is a manifold, which is $f(W)$).

Now consider the open cover of U given by $\{W_x \mid x \in U\}$ where W_x is a neighbourhood of x as above. As opens of Euclidean spaces are Lindelöf spaces, we can extract a countable open sub-cover of U . Thus, $F(V)$ is a countable union of k -dimensional manifolds. \square

Proof of Theorem E.2. Let k be the maximal rank of f on U . Let $V := \{x \in U \mid \text{rank}(f(x)) = k\}$. Let's first show that it is sufficient to show the result for $f|_V$.

By Lemma E.4, V is open and dense in U , and $U - V$ has Lebesgue measure 0. By Lemma E.5, $f_*\lambda = (f|_V)_*\lambda$,

so we reduced the second statement to showing it for $(f|_V)$. In addition, by Sard's theorem, $f(U - V)$ has k -Hausdorff measure 0. Therefore, it is sufficient to prove the theorem for $f|_V$, which we call f again.

Now, write $V = \bigsqcup_i V_i$ where the V_i are the connected components of V . On each connected component V_i , $f|_{V_i}$ verifies the hypothesis of Lemma E.6. Thus $f(V_i)$ is a countable union of k -dimensional manifolds for each V_i . As Euclidean spaces are locally connected spaces with a countable basis, the same holds for V . Therefore, V has at most countably many connected components V_i . A countable union of countable sets is countable, and we finished proving the first part of the theorem.

Finally, it remains to show that $f_*\lambda$ has a density w.r.t. the k -Hausdorff measure on $f(V)$. By Lemma E.3, it is sufficient to show it for each $(f|_W)_*\lambda$, $f(W)$, where W is as constructed above. First, let's show that it is also sufficient to restrict to the case where W is contained in a compact set. There exists a countable cover K_i of compact sets of \mathbb{R}^n that cover W (e.g. by taking closed spheres of radius 1 at points with rational coordinates). Each $W_i := K_i \cap W$ is thus compact in W . Assume for now that we have proved that $(f|_{W_i})_*\lambda$ has a density w.r.t. $f(W)$, i.e. that $(f|_{W_i})_*\lambda(A) = \int_A h_i(x) d\mathcal{H}^k(x)$ for some measurable function $h_i : f(W) \rightarrow \mathbb{R}^+ \cup \{\infty\}$. Then,

$$\begin{aligned} f_*\lambda(A) &= \sum_i (f|_{W_i})_*\lambda(A) \\ &= \sum_i \int_A h_i(x) d\mathcal{H}^k(x) \\ &= \int_A \left(\sum_i h_i(x) \right) d\mathcal{H}^k(x) \end{aligned}$$

where the last equality holds by the Fubini-Tonelli theorem as $h_i \geq 0$. $h := \sum_i h_i$ is measurable, and we are done.

So we can now assume that W is contained in a compact set. As f is analytic, it is locally Lipschitz and is thus Lipschitz on W , as W is contained in a compact set. What's more, As f has rank k , its Jacobian matrix J_x at every point x has rank k . A standard result in linear algebra asserts that $J_x J_x^T$ also has rank k and therefore the correction term $J_W f(x)$ is non-zero for all x . Let $g(x) := \frac{1}{J_W f(x)}$ is therefore measurable and non-negative. By the co-area formula, we have

$$\begin{aligned} f_*\lambda(A) &= \lambda(f^{-1}(A)) \\ &= \int_{f^{-1}(A)} 1 \cdot d\lambda \\ &= \int_{f^{-1}(A)} g(x) J_W f(x) \cdot d\lambda \\ &= \int_{f(W)} \left(\int_{f^{-1}(y)} g(x) d\mathcal{H}^{n-k}(x) \right) d\mathcal{H}^k(y) \\ &= \int_{f(W)} h(y) d\mathcal{H}^k(y) \end{aligned}$$

where $h(y) := \int_{f^{-1}(y)} g(x) d\mathcal{H}^{n-k}(x)$ is measurable and non-negative, yet possibly infinite. \square

Lemma E.7. *Let A be an analytic set. There is an open $U \subseteq A$ such that $\lambda(A - U) = 0$.*

Proof. Let $A = \cap_i X_i^+ \cap \cap_j X_j^-$ as in the definition of analytic sets. Write $X_j^- = X_j \sqcup Z_j$ where $Z_j = (g_j^-)^{-1}(\{0\})$. Without loss of generality, assume that none of the g_j^- are the constant 0 function. Then, $U := \cap_i X_i^+ \cap \cap_j X_j$ is clearly open and $U \subseteq A$. Let's now show that $A - U$ has Lebesgue measure 0. It suffices to show that this is the case for each Z_j , as they cover $A - U$. However, by Mityagin [46]'s result, g_j^- is either the 0 function or the preimage of 0 has Lebesgue measure 0. This shows that Z_j have Lebesgue measure 0, as desired. \square

Theorem E.8. *Let $f : A \rightarrow \mathbb{R}^m$ be PAP. Then $f_*\lambda$ has a density w.r.t. an s -Hausdorff measure μ on \mathbb{R}^m .*

Proof. Let $f : A \rightarrow \mathbb{R}^m$ be PAP. A is c-analytic, so we can write it as $A = \bigsqcup_i A_i$ where each A_i is analytic and (f_i, A_i) is a PAP representation for f . By Lemma E.7, there is an open $U_i \subseteq A_i$ such that $\lambda(A_i - U_i) = 0$. By Lemma E.5, it is thus sufficient to consider the case where f_i is defined on an open set U_i . Any open $V \subseteq \mathbb{R}^n$ can always be written as $V = \bigsqcup_i V_i$ for a countable family of connected opens V_i . Applying this to the U_i , we have a countable family of analytic functions $f_{i,j} : U_{i,j} \rightarrow \mathbb{R}^m$ defined on connected opens $U_{i,j}$, where $f_{i,j}$ is the restriction of f_i to $U_{i,j}$. By theorem E.2, for each (i, j) , there exists an integer $0 \leq k_{i,j} \leq m$ such that $(f_{i,j})_*\lambda = \sum_{l \in \mathbb{N}} f_l^{k_{i,j}} m_l^{k_{i,j}}$, where each $f_l^{k_{i,j}} : \mathbb{R}^m \rightarrow \mathbb{R}^+ \cup \{\infty\}$ is measurable, and each $m_l^{k_{i,j}}$ is the $k_{i,j}$ -Hausdorff measure on some $k_{i,j}$ -submanifold $M_l^{i,j}$ of \mathbb{R}^m . Therefore,

$$\begin{aligned} f_*\lambda &= \sum_{i,j} (f_{i,j})_*\lambda \\ &= \sum_{i,j} \sum_{l \in \mathbb{N}} f_l^{k_{i,j}} m_l^{k_{i,j}} \\ &= \sum_{0 \leq k \leq m} \sum_{\{(i,j) \mid k_{i,j}=k\}} \sum_l f_l^k m_l^k \\ &= \sum_{0 \leq k \leq m} \sum_n f_n^k m_n^k \end{aligned}$$

where in the last equality we simply re-index the countable sum. We can now conclude by Lemma E.3 that $f_*\lambda$ has a density w.r.t. some s -Hausdorff measure. \square

Finally, we prove Theorem IV.4.

Proof. By a theorem of Whitney (refined by Morrey and Grauert), all (finite-dimensional, second-countable, Hausdorff, without boundary) smooth manifolds admit a (unique) analytic structure, and two smooth manifolds are diffeomorphic iff they are analytic-equivalent. Our case is even simpler, as we consider embedded submanifolds of \mathbb{R}^n . Let M be a smooth submanifold of \mathbb{R}^n . There exists a tubular envelope $U \subseteq \mathbb{R}^n$ of M , and a smooth projection $p : U \rightarrow \mathbb{R}^n$ such that $M = p(U)$. This means that $p_*\lambda|_U$ and H^k on M are mutually absolutely continuous. By reasoning in local coordinates, using analytic charts, we see that p is locally a linear projection, and is thus analytic, and therefore PAP. Now let μ be an

arbitrary s-Hausdorff measure, and $\{M_i^k\}_{i,k}$ be a support for μ . Let $p_{i,k} : U_{i,k} \rightarrow \mathbb{R}^n$ be analytic projections such that $p(U_{i,k}) = M_i^k$. Every open set $U \subseteq \mathbb{R}^n$ is diffeomorphic to a bounded open V of radius at most 1, and the diffeomorphism and its inverse can be chosen to be analytic functions. Let $\phi_{i,k} : V_{i,k} \rightarrow U_{i,k}$ be such functions. Let $W_{i,k}$ be the same opens as the $V_{i,k}$, but translated to be disjoint, which is always possible as the $V_{i,k}$ have radius at most 1 and there are countably many of them. Let $t_{i,k} : W_{i,k} \rightarrow V_{i,k}$ be the associated translations, which are linear and therefore analytic. Let $W = \bigsqcup_{i,k} W_{i,k}$. Then the function $f : W \rightarrow \mathbb{R}^n$ defined by $x \in W_{i,j} \mapsto t_{i,k}; \phi_{i,k}; p_{i,k}$ is analytic, and $f_*\lambda$ and μ are mutually absolutely continuous. \square