Classical Linear Logic in Perfect Banach Lattices

Pedro H. Azevedo de Amorim
pedro.azevedo.de.amorim@cs.ox.ac.uk
University of Oxford
Oxford, UK

Leon Witzman
witz0001@e.ntu.edu.sg
Nanyang Technological University
Singapore, Singapore

Dexter Kozen
kozen@cs.conrnell.edu
Cornell University
Ithaca, USA

ABSTRACT

In recent years, researchers have proposed various models of linear logic with strong connections to measure theory, with *probabilistic coherence spaces* (**PCoh**) being one of the most prominent. One of the main limitations of the **PCoh** model is that it cannot interpret continuous measures. To overcome this obstacle, Ehrhard has extended **PCoh** to a category of positive cones and linear Scott-continuous functions and shown that it is a model of intuitionistic linear logic. In this work we show that the category $\mathbf{PBanLat_1}$ of perfect Banach lattices and positive linear functions of norm at most 1 can serve the same purpose, with some added benefits. We show that $\mathbf{PBanLat_1}$ is a model of classical linear logic and that **PCoh** embeds fully and faithfully in $\mathbf{PBanLat_1}$ while preserving the monoidal structure. Finally, we show how $\mathbf{PBanLat_1}$ can be used to give semantics to a higher-order probabilistic programming language with recursion.

1 INTRODUCTION

Probabilistic programming has enjoyed a recent resurgence of interest due to applications in computer vision, machine learning, and the statistical analysis of large datasets, where programs make heavy use of probability. In order to reason about these applications effectively, one must have a sound probabilistic interpretation of programs.

As these use cases become more widespread, compiler engineers and language designers are motivated to implement frameworks that are tailor-made for probabilistic programming. In such frameworks, it is important to equip users with an expressive set of programming primitives that can be soundly implemented, ideally with a specific semantics in mind.

To accomplish this task, it is first necessary to provide a denotational semantics that can interpret these expressive programming languages. This is not an easy task to achieve. A feature which is notoriously difficult to accommodate in a probabilistic setting is higher-order functions. In deterministic semantics, this is usually done with cartesian closed categories. Unfortunately, the standard measure-theoretic apparatus of probability and measure theory is notably not cartesian closed, even when restricted to discrete spaces [2]. This limitation has motivated the construction of *quasi-Borel spaces* [23], a cartesian closed conservative extension of measurable spaces.

Recent work has shown that linear logic has deep connections to the semantics of probabilistic programming languages [8, 9, 11–15, 26, 39]. By using monoidal closed categories instead of cartesian closed categories, linear logic provides an alternative categorical framework for higher-order functions. This was foreshadowed in early work on probabilistic semantics [27] in which bounded linear operators on Banach lattices were used to interpret a first-order imperative probabilistic programming language. This can be seen as evidence that a linear approach might be a natural alternative to cartesian closed categories.

Since then, many probabilistically-flavored models of linear logic have appeared. For instance, the connection between the early work of [27] and linear logic has been recently made precise [8], where the category of regular ordered Banach spaces and regular maps (**RoBan**) was used to extend the semantics of [27] with higher-order functions. It was shown in [8] that **RoBan** is a model of intuitionistic linear logic.

An appealing aspect of the **RoBan** model is that ordered Banach spaces are mathematically well-understood objects with a well-developed classical theory, thus providing a plethora of useful theorems to reason about programs. This is illustrated in [8] by using results from ergodic theory to prove the correctness of a Gibbs sampling algorithm implemented in a higher-order language. However, the programming model supported by the semantics is somewhat brittle, in that the soundness of the system depends on a tricky interaction between three different type grammars with several syntactic restrictions.

A different approach was taken in [9], in which a category **PCoh** was defined and shown to be a model of classical linear logic. The model was used to interpret a version of PCF extended with discrete probabilities [15]. Although this category handles discrete probabilities very nicely, it cannot interpret continuous distributions such as the normal distribution over $\mathbb{R}$, a severe limitation for real-world applications. To remedy this, a category of positive cones with measurability paths and linear Scott-continuous functions $\mathbf{CLin}_m$ has recently been introduced and shown to be a conservative extension of the intuitionistic fragment of **PCoh** [13].

From a programming point of view, the language of [13] is an extension of the simply typed λ-calculus with recursion, making it a simple and expressive programming model. However, the definition of positive cone with measurability paths deviates from standard objects from the probability literature and thus would require a large amount of mathematical effort to rephrase useful theorems that could be used to reason about programs.

Although these previous approaches are valuable contributions to our understanding of higher-order probabilistic programming through linear logic, missing up to now is a comprehensive model that embodies all desirable aspects:

- extends **PCoh** to admit continuous measures;
- is a model of classical (not just intuitionistic) linear logic, thus allowing it to handle other computational interpretations of linear logic such as session types;
- has a simple and expressive programming model that can handle higher-order computation with recursion;

- is based on well-understood classical structures from measure theory and functional analysis.

In this paper we propose such a model. Our model extends **PCoh** with continuous probabilities and satisfies all of the properties above. Our model is based on complete normed vector lattices, called *Banach lattices.* To accommodate the second point, we work with spaces with an involutive linear negation, the so-called *perfect spaces.*

Compared to previous models, our model has simpler tensor product and exponential constructions, which we believe lead to a more perspicuous and theoretically satisfying generalization of **PCoh**. For example, we invite a comparison with $\mathbf{CLin}_m$, where the constructions rely on categorical machinery which, though elegant, are indirect.

Most importantly, Banach lattices can be seen as an abstraction of ordinary measure spaces and are well-studied in functional analysis, with many results from measure theory holding for certain classes of Banach lattices. There is a vast literature on the subject; see [17] for a thorough introduction.

In order to justify the viability of our model, we show that it can be used to interpret a recently introduced higher-order probabilistic calculus [3], and we extend the core calculus of that paper with recursion, the nonlinear modality ! and a new typing rule based on enriched category theory.

Summary of contributions.

- In §3, we define the category $\mathbf{PBanLat_1}$ of perfect Banach lattices and order-continuous positive linear operators with norm at most 1 and show that it is a model of classical linear logic.
- In §4, we show that there is a full and faithful monoidal closed functor $\mathbf{PCoh} \to \mathbf{PBanLat_1}$. This is a more adequate extension than the model $\mathbf{CLin}_m$ proposed in [13], since it also accommodates the classical aspects of the linear structure of **PCoh**.
- In §5, we show that $\mathbf{PBanLat_1}$ is isomorphic to a category of lattices of positive complete cones.
- In §6, we describe an extension to the calculus recently defined in [3] and use $\mathbf{PBanLat_1}$ to interpret it.

Our work contributes both to the study of quantitative models of linear logic as well as to a deeper understanding of higher-order probability theory, shedding light on the importance of linear logic as a vehicle to interpret higher-order programs without cartesian closure. Furthermore, in the tradition of [8, 27], by working with vector spaces that are deeply connected to measure theory, powerful classical results can be brought to bear on the verification of probabilistic programs.

2 RIESZ SPACES

Our model depends on technical definitions and constructions from the vector lattice literature. This section contains a brief self-contained presentation of the subject. We point the interested reader to the introductory texts [1, 42] for good presentations of much of the material presented in this section.

Although we are primarily interested in Banach lattices—normed vector lattices with a completeness property—we start by defining the objects in the general unnormed case.

Definition 2.1. Let $\mathbb{R}_+ = \{a \in \mathbb{R} \mid a \geq 0\}$. A *Riesz space* is a partially-ordered vector space $(V, \leq)$ over $\mathbb{R}$ such that

- if $x \leq y$, then $x + w \leq y + w$;
- if $x \leq y$, then $\alpha x \leq \alpha y$ for $\alpha \in \mathbb{R}_+$; and
- it is an upper semilattice with respect to $\leq$ with join operation $\vee$.

It follows that the space is also a lattice with meet operation $x \wedge y = -(-x \vee -y)$.

Many standard vector spaces are Riesz spaces.

Example 2.2. The following are Riesz spaces:

- $\mathbb{R}^n$ with the pointwise ordering;
- the set of bounded sequences of real numbers with pointwise ordering;
- the set of signed measures on a measurable space;
- the set of bounded measurable functions on a measurable space.

Unlike the real numbers, there are elements that are neither negative nor positive, but a notable characteristic of Riesz spaces is that every element decomposes uniquely into its positive and negative parts.

Definition 2.3. For v an element of a Riesz space, define $v^+ = v \vee 0$, $v^- = (-v) \vee 0$ and $|v| = v \vee -v = v^+ + v^-$.

Then v^+ and v^- are the unique positive elements such that $v = v^+ - v^-$ and $v^+ \wedge v^- = 0$. Thus Riesz spaces are completely characterized by their positive elements. This often simplifies constructions, as one can often prove a property for the positive elements, then extend to the entire space using this decomposition.

Given a Riesz space V, let V^+ denote the set of positive elements of V. Using the decomposition property mentioned above, it follows that $V = V^+ - V^+$, where $-$ applied to sets denotes elementwise subtraction.

2.1 Order convergence

Every topology gives rise to a notion of convergence. For normed spaces, one usually studies convergence in the norm topology. However, ordered spaces also carry an *order topology.*

Definition 2.4. Let D be a directed set and V a Riesz space. A *net* $\{v_\alpha\}_{\alpha \in D}$ is a function $D \to V$. We say that the net is increasing (respectively, decreasing) and write $\{v_\alpha\}\uparrow$ (respectively, $\{v_\alpha\}\downarrow$) if $\alpha \leq_D \beta$ implies $v_\alpha \leq_V v_\beta$ (respectively, $v_\alpha \geq_V v_\beta$).

Definition 2.5. Given a decreasing net $\{x_\alpha\}$, we write $\{x_\alpha\} \downarrow 0$ if $\inf\{x_\alpha\} = 0$.

Definition 2.6 (Order convergence)**.** We say that a net $\{x_\alpha\}$ *converges in order* to x and write $x_\alpha \to x$ if there is a decreasing net $\{y_\alpha\} \downarrow 0$ such that for all α, $|x_\alpha - x| \leq y_\alpha$.

In general, this notion of convergence is neither weaker nor stronger than convergence in norm. However, when a net converges in both order and norm, it converges to the same value in both.

2.2 Riesz subspaces, solids, ideals and bands

In the theory of Riesz spaces, there are classes of subspaces that have many interesting properties that will be used in our constructions.

Definition 2.7. A subset S of a Riesz space is

- *solid* if $x \in S$ and $|y| \leq |x|$ implies $y \in S$,
- an *ideal* if it is a solid linear subspace,
- a *band* if it is an ideal and closed under existing suprema.

Definition 2.8. We say that a Riesz space V is *Archimedean* if for every $v \in V^+$, $\{v/n\}_{n \in \mathbb{N}} \downarrow 0$. Furthermore, if every bounded subset of V admits a supremum, then we say that V is *Dedekind complete*.

Proposition 2.9. *Every band in a Dedekind complete Riesz space is Dedekind complete.*

Definition 2.10. A Riesz subspace $A \subseteq V$ is said to be *order dense* if for every element $0 < v \in V$ there is an element $a \in A$ such that $0 < a \leq v$.

Theorem 2.11 ([1], Th. 1.34)**.** *A Riesz subspace A is order dense in an Archimedean Riesz space V iff for every $v \in V^+$,*

$$\{a \in A \mid 0 \leq a \leq v\} \uparrow v.$$

2.3 Order-continuous functions

As usual when studying vector spaces with extra structure, we care only about linear maps that interact nicely with the extra structure. In our case, the linear functions will have to respect the partial order.

We call a linear function $f : V \to W$ *positive* if it maps positive elements of V to positive elements of W; that is, it restricts to a function $V^+ \to W^+$. A linear function is *regular* if it can be written as the difference of two positive functions.

Definition 2.12. A linear function $T : V \to W$ is *order-continuous* if it is continuous in the order topology. Equivalently, T is *order-continuous* if $Tv_\alpha \to Tv$ whenever $\{v_\alpha\}$ is an increasing net with supremum v.

We can also characterize the positive order-continuous functions as those that preserve existing suprema and infima.

Order continuity interacts well with order density. Indeed, it is possible to show using Theorem 2.11 the following lemma

Lemma 2.13. *If V is an Archimedian Riesz space and $f, g : V \to W$ are two linear order-continuous functions that agree on an order-dense subset of V, then $f = g$.*

This lemma will come in handy when constructing our model. Furthermore, the space of order-continuous linear functions on certain Riesz spaces are well-behaved subsets of the regular linear functions.

Theorem 2.14 ([1], Th 1.57)**.** *If W is Dedekind complete, then the set of order-continuous linear functions $V \to W$ is a band in the space of regular functions, thus forms a Dedekind-complete Riesz space.*

Proof. The Riesz space structure is given by Th 1.18 in [1]. □

Definition 2.15. A Riesz space is *separated* if for every distinct pair $v_1, v_2 \in V$, there exists an order-continuous linear functional $f : V \to \mathbb{R}$ such that $f(v_1) \neq f(v_2)$.

2.4 Normed Riesz spaces

Now we will introduce normed Riesz spaces. In the context of probabilistic semantics, the norm plays an important role, as it can be used to distinguish between arbitrary measures and (sub)-probability distributions, the measures with norm at most 1.

Definition 2.16. Let V be a real vector space. A *norm* is a function $\|\cdot\| : V \to \mathbb{R}^+$ such that:

- $\|v\| = 0$ iff $v = 0$
- $\|\alpha v\| = |\alpha| \, \|v\|$
- $\|v + u\| \leq \|v\| + \|u\|$.

For Riesz spaces, we require the norm to satisfy the additional property

$$|v| \leq |u| \text{ implies } \|v\| \leq \|u\|.$$

If the Riesz space is also complete with respect to the norm, we call it a *Banach lattice*. In vector space models of linear logic, the norm is typically used to distinguish between the product & and the coproduct ⊕, as they both have the same underlying set, but distinct norms. However, in the context of program semantics, the norm also has the extra role of allowing the interpretation of recursive programs.

Example 2.17. The set $\mathcal{M}(\mathbb{R})$ of signed measures over the Borel σ-algebra on $\mathbb{R}$ is a Riesz space (cf. §2.6). We can equip it with the *total variation* norm

$$\|\mu\| = \mu^+(\mathbb{R}) + \mu^-(\mathbb{R}).$$

Theorem 2.14 shows that by assuming the right amount of structure on the Riesz space, the set of order-continuous linear functions between Riesz spaces also has a lattice structure. It is not immediately clear whether this result generalizes to the normed case. Luckily, Dedekind completeness is once again enough.

Example 2.18. Let V and W be normed Riesz spaces with W Dedekind complete. The set of order-continuous linear functions $V \to W$ can be equipped with the *regular norm*

$$\|T\|_r = \sup_{\|x\|_V \leq 1} \||T|(x)\|_W$$

where $|T|$ is given by Theorem 2.14 and Definition 2.3.

Definition 2.19. Let V be a normed Riesz space. The *closed unit ball* of V is the set $\mathcal{B}(V) = \{v \in V \mid \|v\| \leq 1\}$.

2.4.1 Banach lattices. Banach lattices are normed Riesz spaces that are also Banach spaces. In the usual categorical study of Banach spaces, the relevant morphisms are the norm-continuous linear functions.

Definition 2.20. A linear function f between normed Riesz spaces V and W is said to be *norm-continuous* (or *norm-bounded*) if $\sup_{v \in \mathcal{B}(V)} \|f(v)\|$ is finite.

Since we are interested in spaces with two distinct structures, a partial order and a norm, it is not immediately clear which class of morphisms one should care about. In general, the space of all norm-continuous linear functions between Banach lattices is not a Banach lattice, making them unable to give semantics to linear implication.

Normed Riesz spaces are also problematic, as not every order-continuous function is norm-continuous, making it unclear how one would equip the space of order-continuous functions with a norm. However, if the codomain is a Banach lattice, then every order-continuous linear function is also norm-continuous [1]. This suggests that one should work with Banach lattices but only use order-continuous linear functions.

Definition 2.21. The category $\mathbf{BanLat_1}$ has separated Banach lattices as objects and order-continuous positive linear functions of norm at most one as morphisms.

A subtlety when working with a norm and a partial order is that there are two distinct notions of convergence in play that on the surface appear only tenuously related. However, a useful property has been identified in the literature that brings some harmony between the two.

Definition 2.22. A normed Riesz space is said to satisfy the *(sequential) weak Fatou property* if every norm-bounded monotone (sequence) net has a supremum.

In the context of program semantics, the sequential version of this property has been used before to interpret recursive programs [9, 14].

Lemma 2.23. *Let $f : V \to V$ be a positive order-continuous function (not necessarily linear) such that $f(\mathcal{B}(V)) \subseteq \mathcal{B}(V)$. If V satisfies the weak Fatou property, then f admits a fixpoint.*

Proof. It can be directly shown that the limit of the ω-chain $\{f^n(0)\}_{n\in\mathbb{N}}$ is a fixpoint of f. Note that when f is linear, the theorem is trivially true, since $f(0) = 0$. □

Lemma 2.24 ([17] Lem 354B(d))**.** *Every band in a Banach lattice is a Banach lattice.*

Theorem 2.25. *If V and W are Banach lattices, then the set of order-continuous linear functions between V and W is a Banach lattice.*

Proof. The proof is a direct consequence of Banach lattices being Dedekind complete ([17] Prop 354E(e)) and the space of order-continuous being a band in the space of regular linear functions. □

2.5 Dualities

The category $\mathbf{BanLat_1}$ seems to be a good candidate in which to interpret intuitionistic linear logic. However, since the linear negation connective $(-)^\perp$ is usually interpreted as the linear dual $V \multimap \mathbb{R}$ in models of linear logic based on vector spaces over $\mathbb{R}$, $\mathbf{BanLat_1}$ would not be able to model *classical* linear logic, since there are examples of Banach lattices that are not isomorphic to their bidual, e.g. finitely supported real sequences.

A recurring challenge in models of linear logic is to make an involutive linear negation—typical of finite-dimensional spaces—coexist with $!V$, which requires infinite-dimensional spaces. Since we are interested in defining a model of classical linear logic, we should only work with Riesz spaces that are isomorphic to their bidual.

Definition 2.26. Let V^σ denote the space of order-continuous functionals $V \multimap \mathbb{R}$. A Riesz space V is said to be *perfect* if the map $\sigma_V = \lambda x f . f(x) : V \multimap V^{\sigma\sigma}$ is an isomorphism.

We will write σ for σ_V when V is clear from context.

Definition 2.27. The category $\mathbf{PBanLat_1}$ has perfect Banach lattices as objects and positive order-continuous linear functions of norm at most one as morphisms.

Although the definition of perfect spaces is simple, it is difficult to manipulate in practice. The following theorems provide some alternative characterisations, both in the normed and unnormed cases:

Theorem 2.28 ([29], v. XIII, Th. 41.4)**.** *Let V be a separated normed Riesz space. Then V is perfect and Banach iff V has the weak Fatou property.*

Theorem 2.29 ([1], Th 1.71)**.** *A Riesz space V is perfect iff*

- *it is separated;*
- *whenever $0 \leq x_\alpha\uparrow$ and $\sup_\alpha\{f(x_\alpha)\} < \infty$ for all positive $f \in V^\sigma$, there exists $x \in V$ such that $0 \leq x_\alpha \uparrow x$.*

Corollary 2.30. *Bands of perfect Riesz spaces are also perfect.*

Lemma 2.31. *Every perfect Riesz space is Dedekind complete.*

Proof. The proof follows from the second condition of Theorem 2.29. □

Lemma 2.32. *Every Riesz space of the form V^σ is perfect.*

Proof. To show the first point of Theorem 2.29, assume that $f_1 \neq f_2 \in V^\sigma$. Then there is $v \in V$ such that $f_1(v) \neq f_2(v)$. Using the fact that $\lambda f . f(v)$ is an element of $V^{\sigma\sigma}$, we can conclude that V^σ is separated. For the second point, let us assume that $0 \leq \{f_\alpha\}\uparrow$ and that for all $F \in V^{\sigma\sigma}$, if $F \geq 0$, then $\sup_\alpha F(f_\alpha) < \infty$. From this hypothesis, it follows that for all $v \in V$, if $v \geq 0$, then $\sup_\alpha f_\alpha(v) = \sup_\alpha \sigma(x)(f_\alpha) < \infty$. This means that the function $f(x) = \sup_\alpha f_\alpha(x)$ is well-defined, linear, and order-continuous. By Lemma 1.18 in [1], V^σ is Dedekind complete and f bounds f_α. □

An interesting fact that is not obvious from the definitions is that the bidual of Riesz spaces can be seen as a sort of completion procedure.

Lemma 2.33 ([1], Th. 1.70)**.** *Let V be an Archimedean Riesz space. The set $\sigma(V)$ is an order-dense Riesz subspace of $V^{\sigma\sigma}$.*

It is possible to categorify the theorem above by showing that $(-)^{\sigma\sigma}$ is a functor from a category of Riesz spaces and order-continuous linear functions to a category of perfect Riesz spaces with the same morphisms. This functor is defined analogously to the continuation monad from the theory of programming languages. Furthermore, there is an obvious forgetful functor $U : U : \mathbf{PBanLat_1} \to \mathbf{BanLat_1}$.

Theorem 2.34. *The functor $(-)^{\sigma\sigma} : \mathbf{BanLat_1} \to \mathbf{PBanLat_1}$ is left adjoint to the forgetful functor U.*

Proof. We observe that if $f : V \multimap W$, then $\sigma^{-1} \circ f^{\sigma\sigma} : V^{\sigma\sigma} \multimap W$. In the other direction, if we have a function $f : V^{\sigma\sigma} \multimap W$, we can consider its restriction $f \restriction V : V \multimap W$. To show that these operations are inverses, we use Theorem 2.11 and Theorem 2.33, which allow us to show that if two order-continuous functions agree on $\sigma(V)$, then they agree everywhere. □

Lemma 2.35. *If V is a separated Riesz space, then the function $\sigma : V \multimap V^{\sigma\sigma}$ is injective.*

The theorem and lemma above are useful because they imply that if V is a separated Riesz space and W is a perfect Riesz space, then every order-continuous linear function $f : V \multimap W$ extends uniquely to a function $V^{\sigma\sigma} \multimap W$.

Remark 2.36. It might possible to define a semantics for an expressive probabilistic programming language using only σ-perfect Riesz spaces, those that are isomorphic to their sequential bidual. If that is the case, it would be necessary to have a theorem analogous to Theorem 2.29.

The separability condition would need to be kept to guarantee injectivity, but the other conditions require further investigation, which we leave to future work.

2.6 Signed measures as Riesz spaces

Measures are usually defined as countably additive, nonnegative real-valued functions on a σ-algebra. *Signed measures* provide a slight generalization by dropping the requirement of nonnegativity.

Definition 2.37. Let (X, Σ) be a measurable space. A *signed measure* is a function $\mu : \Sigma \to \mathbb{R}$ such that $\mu(\emptyset) = 0$ and $\mu(\bigcup_{i\in\mathbb{N}} A_i) = \sum_{i\in\mathbb{N}} \mu(A_i)$ for disjoint sets $(A_i)_{i\in\mathbb{N}}$.

An important difference between ordinary measures and signed measures is that signed measures come equipped with a natural vector space structure. Indeed, it can be shown that signed measures are perfect Riesz spaces.

Lemma 2.38. *Let (X, Σ) be a measurable space. The space $\mathcal{M}(X, \Sigma)$ of signed measures is a normed Riesz space.*

Proof. The vector space structure is defined pointwise with lattice structure defined by $\mu \vee \nu = (\mu - \nu)^+ + \nu$ using the Hahn-Jordan decomposition and the norm is the total-variation norm. □

When a measure μ is positive, its total variation norm is its total mass $\mu(X)$.

Theorem 2.39. *Let (X, Σ) be a measurable space. The space $\mathcal{M}(X, \Sigma)$ of signed measures with the total variation norm is a perfect Banach lattice.*

Proof. The proof follows by applying Theorem 2.28, the lemma above and observing that since the order of measures is given pointwise, you can define their suprema pointwise as well. □

3 MODELS OF LINEAR LOGIC

The categorical semantics of linear logic is very well understood; see Mellies [31] for an overview. In this section, we show that $\mathbf{PBanLat_1}$ is a model of classical linear logic.

3.1 Symmetric Monoidal Closed Structure

In order for $\mathbf{PBanLat_1}$ to interpret the multiplicative fragment of linear logic, i.e. give semantics to a linear λ-calculus with tensors, it must be a symmetric monoidal closed category. Concretely, it needs a *monoidal product* $\otimes$ such that for every object A, the functor $A \otimes -$ has a right adjoint $A \multimap -$, known as *linear implication*.

For models based on vector spaces, the monoidal product is typically given by the *tensor product*. For such models, linear implication has a natural interpretation in terms of linear functions. Furthermore, since our spaces are perfect, we have an involutive *linear negation* $A^\perp$ defined as the space $A \multimap \mathbb{R}$, and, in models of classical linear logic, the equation $A \otimes B = (A \multimap B^\perp)^\perp$ holds. Thus the tensor product $\otimes$ can be defined in terms of linear implication $\multimap$ and negation $^\perp$ in such models.

Note that this circumvents one of the main complications with the model of [13], where the existence of a suitable monoidal product is established non-constructively using a categorical density argument.

3.1.1 Internal Homs. Since the category $\mathbf{PBanLat_1}$ has order-continuous linear functions with norm at most 1 as morphisms, it makes sense to define the internal hom object $V \multimap W$ as the space of order-continuous linear functions between perfect Banach lattices V and W. This definition is justified by the following theorem.

Lemma 3.1. *If V and W are perfect Riesz spaces, then the set of order continuous linear functions $V \multimap W$ is a perfect Riesz space.*

Proof. By Theorem 2.14, $V \multimap W$ is a Riesz space. Applying Theorem 2.29, we can also show that it is perfect. To show separability, let $f_1, f_2 : V \multimap W$ be distinct functions. Then there is a point $v \in V$ such that $f_1(v) \neq f_2(v)$. Since W is perfect, it is separated, therefore there exists $g : W \multimap \mathbb{R}$ such that $g(f_1(v)) \neq g(f_2(v))$. Then the order-continuous function $\lambda f\,.\, g(f(v))$ separates the points f_1 and f_2, therefore $V \multimap W$ is separated.

Now let $0 \leq \{f_\alpha\}\uparrow$ be an increasing net such that $\sup_\alpha F(f_\alpha) < \infty$ for all positive $F : (V \multimap W) \multimap \mathbb{R}$. We can define an f such that $f_\alpha \uparrow f$ pointwise. Let $v \in V^+$ and let $F : W \multimap \mathbb{R}$ be a positive functional. Consider the functional $\lambda f\,.\, F(f(v)) : (V \multimap W) \multimap \mathbb{R}$. By hypothesis, $\sup_\alpha (F(f_\alpha(v))) < \infty$, and since W is perfect and $\{f_\alpha(v)\}$ is a positive net in W, there exists $f(v) \in W$ such that $f_\alpha(v) \uparrow f(v)$. This defines f on elements of V^+, and for arbitrary $v \in V$ we take $f(v) = f(v^+) - f(v^-)$. Then $\sup_\alpha f_\alpha = f$. □

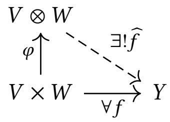
From Lemma 2.25 and the theorem above, it follows that if V and W are perfect Banach lattices, then so is $V \multimap W$. By using standard techniques from the literature on vector models of linear logic, we have

Theorem 3.2. *The operation* $\multimap$: $\mathbf{PBanLat_1}^{op} \times \mathbf{PBanLat_1} \to \mathbf{PBanLat_1}$ *is functorial.*

3.1.2 Monoidal structure. As mentioned above, the monoidal structure on vector space models of linear logic is usually defined as a tensor product, and monoidal closure is obtained from the universal property of tensor products. The usual recipe for defining tensor products is to use a free construction modulo the tensor product equations. When working with infinite-dimensional spaces, a completion procedure may be required as well.

Indeed, this is the approach taken in [16], in which a tensor product is defined for perfect Riesz spaces via a more traditional construction using the completion of the algebraic tensor product. It is also shown in [16] that $V \otimes W \cong (V \multimap W^\perp)^\perp$, meaning that their construction is isomorphic to ours.

In contrast, our construction starts with the definition $V \otimes W \triangleq (V \multimap W^\sigma)^\sigma$, as required by the laws of linear logic. We then show that it satisfies the expected universal property of tensor products:



(3.1)

where φ and f are bilinear functions. We show this using the fact that the internal hom can be used to classify bilinear functions using $V \multimap (W \multimap Y)$, then showing that this space is isomorphic to $V \otimes W \multimap Y$.

Lemma 3.3. *$V \otimes W \multimap Y \cong V \multimap W \multimap Y$.*

Proof. Recall that if V and W are perfect Riesz spaces, then $V \multimap W \cong W^\sigma \multimap V^\sigma$. Then

$$\begin{aligned} V \otimes W \multimap Y &= (V \multimap W^\sigma)^\sigma \multimap Y \\ &\cong Y^\sigma \multimap (V \multimap W^\sigma) \cong V \multimap Y^\sigma \multimap W^\sigma \\ &\cong V \multimap W \multimap Y. \end{aligned}$$ □

Theorem 3.4. *$V \otimes W$, defined as $(V \multimap W^\sigma)^\sigma$, satisfies the universal property of tensor products* (3.1).

Proof. Observe that the set of (norm bounded) bilinear order-continuous functions $V \times W \to Y$ is (isometrically, in the normed case) isomorphic to $V \multimap W \multimap Y$. Restating the diagram (3.1) in this language, we must show that $V \otimes W \multimap Y \cong V \multimap W \multimap Y$. This is exactly Lemma 3.3. □

Using the universal property (3.1) and the (easy to prove) facts that $V \otimes (W \otimes Y) \cong (V \otimes W) \otimes Y$ and $V \otimes W \cong W \otimes V$, we can conclude:

Theorem 3.5. **PBanLat$_1$** *is a symmetric monoidal closed category.*

It is difficult in general to give an intuitive characterization of the elements of a tensor product. This is also the case with our construction. Nevertheless, in the context of measures, we can give some intuition for the elements of $\mathcal{M}(A) \otimes \mathcal{M}(B)$. Let μ_A and μ_B be probability distributions on measurable spaces A and B, respectively. The product distribution $\mu_A \otimes \mu_B$ is the joint probability distribution on $A \times B$ with marginals μ_A and μ_B obtained by sampling μ_A and μ_B independently. This is an element of $\mathcal{M}(A) \otimes \mathcal{M}(B)$, but there are also other joint distributions in $\mathcal{M}(A) \otimes \mathcal{M}(B)$ that do not represent independent samples. For example, let $A = B = \{0, 1\}$ and consider the joint distribution $\frac{1}{2}(\delta_0 \otimes \delta_0 + \delta_1 \otimes \delta_1)$. Sampling this distribution returns $(0, 0)$ or $(1, 1)$, each with probability $1/2$, so the two components are clearly not independent.

In general, not every joint distribution is an element of the tensor product, as explained in [8]. From a programming point of view, the universal property of tensor products says that the behavior of a program taking inputs of type $\mathcal{M}(A) \otimes \mathcal{M}(B)$ is fully characterized by its behavior on inputs that are independent distributions over A and B.

3.2 $*$-autonomous categories

Classical linear logic differs from its intuitionistic variant by requiring that linear negation be involutive, that is, $A^{\perp\perp} = A$ for every formula A. Categorically, this is modeled by *$*$-autonomous categories*, symmetric monoidal closed categories $\mathbf{C}$ with a functor $(-)^* : \mathbf{C}^{\mathrm{op}} \to \mathbf{C}$ such that every object A is naturally isomorphic to A^{**} and for every three objects A, B, C, there is a natural bijection $\mathrm{Hom}(A \otimes B, C^*) \cong \mathrm{Hom}(A, (B \otimes C)^*)$. Equivalently, a $*$-autonomous category is a symmetric monoidal closed category $\mathbf{C}$ equipped with a *dualizing object* $\perp$ such that for every object A, the unit $\partial_A : A \to (A \multimap \perp) \multimap \perp$ is an isomorphism.

In our case, the dualizing object is $\mathbb{R}$, the unit is the linear function $\sigma_V : V \to V^{\sigma\sigma}$, and the isomorphism holds by assumption.

Theorem 3.6. **PBanLat$_1$** *is a $*$-autonomous category.*

3.3 Cartesian and co-Cartesian structure

Cartesian and co-Cartesian structure are useful in the formation of product and sum types. In models of linear logic, these are represented by linear conjunction & and disjunction $\oplus$, respectively. In **PBanLat$_1$**, both operations have $V \times W$ as their underlying set with lattice operations defined componentwise. In the normed case, we can distinguish them by choosing different norms.

Definition 3.7. Let V and W be normed Riesz spaces. We define

- the product $V \& W = (V \times W, \|-\|_{\mathrm{sum}})$, where $\|(v, w)\|_{\mathrm{sum}} = \|v\| + \|w\|$.
- the coproduct $V \oplus W = (V \times W, \|-\|_{\max})$, where $\|(v, w)\|_{\max} = \max(\|v\|, \|w\|)$.

Since convergence for both is defined componentwise, by using Theorem 2.28 we can show that if V and W are perfect and Banach, then $V \& W$ and $V \oplus W$ are as well. The unit $\top$ for the product and 0 for the coproduct are both the trivial Riesz space $\{0\}$.

Theorem 3.8. **PBanLat$_1$** *is (co-)Cartesian.*

3.4 Exponentials

The trickiest part in defining models of linear logic is interpreting the exponential modality !, which can be used as a type constructor in a linear programming language that allows variables to be reused and discarded. In particular, this modality recovers the expressive power of the simply-typed λ-calculus and, in the context of probabilistic models of linear logic, it allows distributions to be resampled. Before we present its formal construction, we will give some intuition behind it.

In many concrete models, the construction of $!V$ captures the idea that its elements are those of the form $V^{\otimes n}$ for $n \geq 0$. This intuition also holds for vector space models, with the additional requirement that it must also accommodate a vector space structure.

Categorically, such a modality is captured by a monoidal comonad ! equipped with operations that model the structural rules of *contraction* and *weakening*.

Definition 3.9. An *exponential* in a categorical model of linear logic is a comonad $(!, \varepsilon, \rho)$, where $\varepsilon_A : !A \multimap A$ and $\rho_A : !A \multimap !!A$, such that every object $!A$ carries a commutative comonoid structure

$$(!A, \underline{d}_A : !A \to !A \otimes !A, \underline{e}_A : !A \to 1),$$

or, alternatively, it satisfies the Seely isomorphisms $!(A \& B) \cong !A \otimes !B$ and $!\top \cong 1$.

Our construction is inspired by [26]. Let V be a perfect Banach lattice and consider bounded functions $f : \mathcal{B}(V)^+ \to \mathbb{R}$ that can be written as an infinite sum of n-linear positive order-continuous functions, also known as power series:

$$f(x) = \sum_{n=0}^{\infty} f_n(x, \dots, x). \tag{3.2}$$

Since the functions f_n are positive and so are their arguments, f converges absolutely on all of $\mathcal{B}(V)^+$. It is also clear that f is monotone and order-continuous, as each of the f_n are so. By taking the formal difference of positive power series, we obtain a vector space. Furthermore, this vector space can be equipped with a normed Riesz space structure which we call $S(V;\mathbb{R})$, where the order structure is given pointwise.

Lemma 3.10. *If V is a perfect Banach lattice and $v \in \mathcal{B}(V)^+$, the Dirac distribution $\delta_v \in S(V;\mathbb{R})^\sigma$, where $\delta_v(F) = F(v)$.*

Proof. The linearity of δ_v is immediate and in order to prove its order continuity, let $\{(f_{0,\alpha}, f_{1,\alpha}, \dots)\}_\alpha \uparrow (f_0, f_1, \cdots)$ be an ascending net in $S(V;\mathbb{R})$. Without loss of generality, we may assume that this net is contained in the positive unit ball of $S(V;\mathbb{R})$. For every natural number n, $\sup_\alpha(\sum_0^n f_{i,\alpha}(v)) = \sum_0^n(\sup_\alpha(f_{i,\alpha}(v)))$, by order-continuity of addition. From this we can conclude $\sup_\alpha(\delta_v(f_\alpha)) \uparrow \delta_v(f)$. □

We will define $!V$ to be the "smallest" perfect Banach lattice which is contained in $S(V;\mathbb{R})^\sigma$ and contains the Dirac distributions. Such a space exists since, $S(V;\mathbb{R})^\sigma$ is perfect and Banach, and these properties are close under arbitrary intersection. In the rest of the section we will make the construction precise.

Free Riesz spaces. There has been much work done on free constructions for partially ordered vector spaces. We are interested in the construction of [41] showing that every partially ordered vector space can be extended to a Riesz space.

Theorem 3.11 ([41], Th 3.5). *Let A be a partially ordered vector space. There is a Riesz space $\tilde{A}$ and an order-continuous bipositive[1] inclusion function $\iota : A \longrightarrow \tilde{A}$ such that the image of ι is order-dense in $\tilde{A}$.*

Furthermore, there is an analogue of this construction for normed Riesz spaces.

Lemma 3.12. *Let A be a partially ordered normed vector space, V an Archimedean normed Riesz space, and W a perfect Banach lattice. If $A \subseteq V$ is order-dense, then every positive order-continuous linear function $f : A \longrightarrow W$ with norm at most 1 extends uniquely to an order-continous linear function on V.*

Proof. By order density, one can define $\tilde{f}(v) = \sup\{f(a) \mid a \leq v\}$. It follows from Theorem 2.28 that the supremum is well-defined. The proposed extension is linear because both addition and scalar multiplication are order-continuous. Furthermore, the extension is order-continuous:

$$\sup_\alpha \tilde{f}(v_\alpha) = \sup_\alpha \sup_\beta f(v_{\alpha,\beta}) = \sup_\beta f(\sup_\alpha v_\alpha) = \tilde{f}(\sup_\alpha v_\alpha) \qquad \square$$

[1] that is, preserves and reflects the poset structure, which implies injectivity, since $f(v_1) \leq f(v_2)$ and $f(v_2) \leq f(v_1)$ implies $v_1 = v_2$.

Lemma 3.13 ([5], Lem 5.3). *If $v_1, \dots, v_n \in \mathcal{B}(V)^+$ are distinct vectors, then the distributions $\delta_{v_1}, \dots, \delta_{v_n}$ are linearly independent.*

Let $\Delta(V)$ be the normed Riesz space generated by the partially ordered vector space spanned by $\{\delta_v\}_{v\in\mathcal{B}(V)^+}$ according to Theorem 3.11.

Lemma 3.14. *The space $\Delta(V)$ embeds as a Riesz subspace of $S(V;\mathbb{R})^\sigma$. Thus it is Archimedean.*

Proof. This is a direct consequence of Corollary 3.10 from [41] and the fact that $S(V;\mathbb{R})^\sigma$ is Archimedean. □

Lemma 3.15. *Let V be a perfect Banach lattice and $v \in \mathcal{B}(V)^+$.*

- *$v \leq v'$ if, and only if, $\delta_v \leq \delta_{v'}$*
- *$\{v_\alpha\}_\alpha \uparrow v$ if, and only if $\{\delta_{v_\alpha}\}_\alpha \uparrow \delta_v$*

Proof. We begin with the forwards direction. Let $\{v_\alpha\}_\alpha \subset \mathcal{B}(V)^+$ be an ascending chain of positive elements, which by the weak Fatou property converges to an element of $\mathcal{B}(V)^+$. We want to show that $\delta(\sup_\alpha v_\alpha) = \sup_\alpha \delta(v_\alpha)$. We have the following equations

$$\delta(\sup_\alpha v_\alpha)(f) = f(\sup_\alpha v_\alpha) = \sup_\alpha f(v_\alpha) = \sup_\alpha \delta(v_\alpha)(f).$$

The second equation holds because f is order-continuous.

In order to prove monotonicity, we use the fact that every positive power series is the sum of positive n-linear functions, therefore

$$v \leq v' \Rightarrow \delta(v)(\sum_i f_i) = \sum_i f_i(v, \dots v) \leq \sum_i f_i(v', \dots v') = \delta(v')(\sum_i f_i).$$

The other direction follows from observing that since, by assumption, the objects of $\mathbf{PBanLat_1}$ are perfect, we can recover a lot of information about V by restricting the domain of δ_v to the linear functionals V^σ. Therefore, for every positive $f : V^\sigma$, $f(v) \leq f(v')$, which implies $\sigma(v) \leq \sigma(v')$, where $\sigma : V \longrightarrow V^{\sigma\sigma}$ is the isomorphism. Since σ preserves the lattice structure of V, $v \leq v'$. From a similar argument, assuming that $\{v_\alpha\}_\alpha \uparrow v$ it follows that $\{\sigma(v_\alpha)\}_\alpha \uparrow \sigma(v)$, which concludes the proof by the second point of Theorem 2.29. □

Corollary 3.16. *The function $\delta : \mathcal{B}(V)^+ \longrightarrow \mathcal{B}(!V)^+$ is Scott continuous.*

Closure under suprema. Now we have a normed Riesz space which is a subset of $S(V;\mathbb{R})^\sigma$, but it is not necessarily perfect or Banach. In order get both of these properties, we will use Theorem 2.28. We restrict $\Delta(V)$ to its positive unit ball and, by transfinite induction, we add to it all of the suprema of directed sets, and generate the whole space by first generating the positive cone from the positive unit ball and from it, and then taking differences of elements. We call this space $!V$.

Definition 3.17. Let Y be a CPO and $X \subseteq Y$ a poset with the same partial order relation as Y. Let Ord be the class of ordinals and define by transfinite induction the family of partial orders $T(\alpha)$

parametrized by ordinals.

$$\begin{aligned} &T : Ord \to \mathbf{Set} \\ &T(0) = X \\ &T(\alpha+1) = \{\sup\{x_\beta\}_\beta : Y \mid \{x_\beta\} \subseteq T(\alpha)\} \\ &T(\bigvee_{\beta<\alpha} \beta) = \bigcup_{\beta<\alpha} T(\beta). \end{aligned}$$

There is a smallest ordinal α_0 such that $T(\alpha_0+1) = T(\alpha_0)$. We denote $T(\alpha_0)$ as X_{CPO}.

Lemma 3.18. *The partial order X_{CPO} is a CPO.*

Proof. Let $\{x_\alpha\}$ be a directed set. By assumption, it has a supremum in Y and it is an element of $T(\alpha_0+1)$, which concludes the proof, since $T(\alpha_0) = T(\alpha_0+1)$. □

In our case, we choose Y to be the positive unit ball of $S(V;\mathbb{R})^\sigma$ and X to be the positive unit ball of $\Delta(V)$. It follows by transfinite induction and order-continuity of the Riesz space structure, that for every ordinal α, $T(\alpha)$ is the positive unit ball of a normed Riesz space embedded in $S(V;\mathbb{R})^\sigma$. We define $!V$ to be the perfect Banach lattice generated by the positive unit ball $\Delta(V)_{CPO}$.

Lemma 3.19. *Let V be a normed Riesz space and V', W perfect Banach lattices, such that $V \subseteq V'$. Every monotonic order-continuous function $f : \mathcal{B}(V)^+ \to \mathcal{B}(W)^+$ can be extended uniquely to a monotonic order continuous function $f^* : (\mathcal{B}V)^+_{CPO} \to \mathcal{B}(W)^+$.*

Furthermore, if f preserves sub-convex combinations[2], then f^* can be further extended to a positive linear order-continuous function $\tilde{f} : (V_{CPO} - V_{CPO}) \to W$, where $(V_{CPO} - V_{CPO})$ is the perfect Banach lattice generated by the positive unit ball V_{CPO}.

Proof. By transfinite induction on the structure of V_{CPO}, the extension f^* on a point v which is the supremum of a directed set $\{v_\alpha\}_\alpha$, can be defined as $\sup\{f^*(v_\alpha)\}_\alpha$. By monotonicity of f and the weak Fatou property of W, such supremum is well-defined. However, a priori, a distinct directed set $\{v'_{\alpha'}\}_{\alpha'}$ with supremum v might give a different $\sup\{f(v'_{\alpha'})\}_{\alpha'}$.

We show that this is not the case by considering the directed set $\{v_\alpha \wedge v'_{\alpha'}\}_{\alpha,\alpha'}$. By order continuity of $\wedge$, its supremum is v, and since $v_\alpha \wedge v'_{\alpha'} \le v_\alpha$, $\sup\{f(v_\alpha \wedge v'_{\alpha'})\}_{\alpha,\alpha'} \le \sup\{f(v_\alpha)\}_\alpha$. To show the other direction, note that since the supremum of $v_\alpha \wedge v'_{\alpha'}$ is v, for every α_0, there are α, α' such that $v_{\alpha_0} \le v_\alpha \wedge v'_{\alpha'}$, by monotonicity, this allows us to show $\sup\{f(v_\alpha)\}_\alpha \le \sup\{f(v_\alpha \wedge v'_{\alpha'})\}_{\alpha,\alpha'}$, therefore

$$\sup\{f(v_\alpha)\}_\alpha = \sup\{f(v_\alpha \wedge v'_{\alpha'})\}_{\alpha,\alpha'} = \sup\{f(v'_{\alpha'})\}_{\alpha'}$$

Both uniqueness and order continuity of f^* follow by construction and from order continuity of f.

If f also preserves sub-convex combinations, this also means that for $0 \le \alpha \le 1$, $f(\alpha x) = \alpha f(x)$. With this in mind, we define $\tilde{f}(x) = \|x\|\tilde{f}(\frac{x}{\|x\|})$, when $x > 0$. By linearity, this definition can be directly extended to non-positive elements as well. Therefore, assuming without loss of generality that $x, y > 0$,

$$\begin{aligned} &\tilde{f}(x+y) = \|x+y\| f\left(\frac{x+y}{\|x+y\|}\right) = \\ &\|x+y\|\left(f\left(\frac{x}{\|x+y\|}\right) + f\left(\frac{y}{\|x+y\|}\right)\right) \\ &\|x+y\|\left(f\left(\frac{\|x\|x}{\|x\|\|x+y\|}\right) + f\left(\frac{\|y\|y}{\|y\|\|x+y\|}\right)\right) = \\ &\|x\| f\left(\frac{x}{\|x\|}\right) + \|y\| f\left(\frac{y}{\|y\|}\right) = \tilde{f}(x) + \tilde{f}(y). \end{aligned}$$

The same proof strategy can be used for proving $\tilde{f}(\alpha x) = \alpha\tilde{f}(x)$ for any $\alpha \in \mathbb{R}$. □

Note that the proof above is less direct than that of Lemma 3.12 because that lemma has the simplifying hypothesis that the subspace is order dense, which is not the case here.

Lemma 3.20. *The space $!V$ is a perfect Banach lattice.*

Proof. By applying Theorem 2.28, it suffices to show that its positive unit ball is a CPO and that it is a normed Riesz space.

Completeness follows from Lemma 3.18. In order to show that $!V$ is a normed Riesz space, we equip it with the same structure as the host space $S(V;\mathbb{R})^\sigma$. Order continuity of the Riesz space structure is the property that guarantees that $!V$ is closed under addition, multiplication by scalar, and the lattice operations. □

Lemma 3.21. *Let V be a normed Riesz space and W a perfect Banach lattice. If $f : \Delta(V) \to W$ is an order continuous positive linear function with norm at most 1, then it extends uniquely to the entire space $!V$. Furthemore, if two functions $f_1, f_2 : !V \to W$ agree on every point δ_v, they are equal.*

Proof. This follows by application of Lemmas 3.12 and 3.19. □

The lemma above tells us that in order to define a morphism over $!V$ it suffices to define it over $\Delta(V)$, which in turn means that it suffices to define it over the Dirac distributions δ_v. We can now proceed to show that ! is indeed a linear logic exponential.

Theorem 3.22. *The mapping $V \mapsto !V$ is functorial.*

Proof. The action on morphisms $f : V \to W$ is the function $!V \multimap !W$ generated by $\delta_v \mapsto \delta_{f(v)}$, which is well defined by the hypothesis that f is order-continuous and with norm at most 1 and Lemma 3.15. Concretely, assume $\{\delta_{v_\alpha}\}_\alpha \uparrow \delta_v$ which implies that $v_\alpha \uparrow v$. By assumption, $f(v_\alpha) \uparrow f(v)$ and by Scott-continuity of δ, $\delta_{f(v_\alpha)} \uparrow \delta_{f(v)}$.

The functor laws follow as a consequence of Lemma 3.21:

$$\begin{aligned} &!(\mathrm{id}_V)(\delta_v) = \delta_v = \mathrm{id}_{!V}(\delta_v) \\ &!(f \circ g)(\delta_v) = \delta_{f(g(v))} = (!f \circ !g)(\delta_v). \end{aligned}$$ □

Now we can define a comonad structure over !, where the counit $\varepsilon_V : !V \multimap V$ is the function generated by $\delta_v \mapsto v$ and the comultiplication $\rho_V : !V \multimap !!V$ is generated by $\delta_v \mapsto \delta_{\delta_v}$. Their order-continuity and norm being at most 1 follow from arguments similar to the functorial action.

Theorem 3.23. *The triple $(!, \varepsilon_V, \rho_V)$ is a comonad in* $\mathbf{PBanLat_1}$.

[2] $f(\alpha x + \beta y) = \alpha f(x) + \beta f(y), 0 \le \alpha + \beta \le 1$

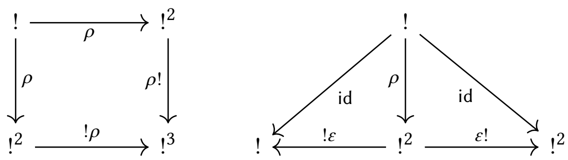


Figure 1: Comonad laws

Proof. it follows from Lemma 3.21 that we need only show commutativity of the diagrams in Figure 1 for the points δ_v:

$$\rho_{!V}(\rho_V(\delta_v)) = \rho_{!V}(\delta_{\delta_v}) = \delta_{\delta_{\delta_v}} = (!\rho_V)(\delta_{\delta_v}) = (!\rho_V)(\rho_V(\delta_v))$$

$$(!\varepsilon_{!V})(\rho_V(\delta_v)) = (!\varepsilon_{!V})(\delta_{\delta_v}) = \delta_v$$

$$(\varepsilon_{!^2V})(\rho_V(\delta_v)) = (\varepsilon_{!^2V})(\delta_{\delta_v}) = \delta_v.$$ □

Finally, we must show that the Seely isomorphisms $!(V \,\&\, W) \cong {!V} \otimes {!W}$ and $!\top \cong 1$ hold. The former is given by the pair of functions generated by $\delta_{(v,w)} \mapsto \delta_v \otimes \delta_w$ and $\delta_v \otimes \delta_w \mapsto \delta_{(v,w)}$, and the latter is given by the functions generated by $\delta_0 \mapsto 1$ and $1 \mapsto \delta_0$. Both of these isomorphisms can be checked by a direct calculation.

Theorem 3.24. *The category* **PBanLat₁** *is a model of classical linear logic.*

Remark 3.25. Our construction deviates slightly from previous work by constructing the comonad directly, instead of defining a monoidal adjunction between a symmetric monoidal closed category and a cartesian one. We opted for the direct construction because it does not need the definition of the cartesian category or a characterization of the morphisms $!V \multimap W$, which would have required a careful analysis of order-convergence of monotone functions $\mathcal{B}(V)^+ \to W$, as was done in [14]. We have conducted such an analysis for a distinct *order continuous* exponential, where the non-linear morphisms are the order continuous functions between positive unit balls. The details are in Appendix B.

3.5 Fixed Points

In order to give semantics to recursive probabilistic programs, we need to be able to compute fixed points of morphisms $!V \multimap V$ with norm ≤ 1. Since our objects are perfect Banach spaces, they satisfy the weak Fatou property, which in turn can be used to define fixed points of Scott continuous endofunctions. After all, every positive Scott-continuous function $f : \mathcal{B}(V) \to \mathcal{B}(V)$ induces an ascending chain $\{f^n(0)\}_{n\in\mathbb{N}}$ which is norm bounded by assumption and therefore admits a supremum.

Definition 3.26. A function $f : \mathcal{B}(V)^+ \to \mathcal{B}(W)^+$ is called *analytic* if there is a positive function $g : !V \multimap W$ with norm ≤ 1 such that $f(v) = g(\delta_v)$.

Due to Lemma 3.21, every function $g : !V \multimap W$ with norm ≤ 1 may only be associated to one analytic function. The definition above suggests that if the function $\delta : \mathcal{B}(V) \to \mathcal{B}(!V)$ is Scott continuous then every analytic function admits a fixed point. We know that this is the case by Corollary 3.16.

Therefore, every positive analytic function g is also order-continuous and monotonic, since they can be seen as the composition of δ with g, meaning that we can compute its fixed point by using Lemma 2.23. Next, consider the program $\lambda F\, f\,.\, f\ (F\ f) : !(!(!V \multimap !V) \multimap V)) \multimap !(!V \multimap V) \multimap V$. It is possible to show that it is positive and that it has norm 1, therefore by precomposing it with δ it admits a fixed point, which we call Fix. By construction, we have the equation $(\mathsf{Fix}\, F)(f) = f((\mathsf{Fix}\, F)(f))$, making $(\mathsf{Fix}\, F)(f)$ the fixed point of f. Furthermore, by construction, $\mathsf{Fix}\, F$ has type $!(!V \multimap V) \multimap V$, meaning that it can be interpreted in **PBanLat₁**.

4 PROBABILISTIC COHERENCE SPACES AND BANACH LATTICES

Probabilistic coherence spaces (PCS) [9] are a model of linear logic with a vector space flavor. It has been shown by Ehrhard [13] that its intuitionistic fragment can be fully and faithfully embedded in a category of positive cones. In this section, we show that Banach lattices can handle not only the intuitionistic fragment of PCS, but the classical one as well. We make use of the vector space construction presented in the original paper [9].

Definition 4.1. A Probabilistic Coherence Space (PCS) is a pair $(X, \mathcal{P}(X))$, where X is a countable set and $\mathcal{P}(X) \subseteq X \to \mathbb{R}^+$ called the *web* such that:

- $\forall a \in X\ \exists \varepsilon_a > 0\ \varepsilon_a \cdot \delta_a \in \mathcal{P}(X)$, where $\delta_a(a') = 1$ iff $a = a'$ and 0 otherwise;
- $\forall a \in X\ \exists \lambda_a\ \forall x \in \mathcal{P}(X)\ x_a \leq \lambda_a$;
- $\mathcal{P}(X)^{\perp\perp} = \mathcal{P}(X)$, where $\mathcal{P}(X)^{\perp} = \{x \in X \to \mathbb{R}^+ \mid \forall v \in \mathcal{P}(X)\ \sum_{a\in X} x_a v_a \leq 1\}$.

Definition 4.2. Let $(X, \mathcal{P}(X))$ be a PCS. Its linear negation is the PCS $(X, \mathcal{P}(X)^{\perp})$.

Definition 4.3. Let $(X, \mathcal{P}(X))$ and $(Y, \mathcal{P}(Y))$ be PCSs. The PCS $X \multimap Y$ is the pair $(X \times Y, \mathcal{P}(X \multimap Y))$, where $P(X \multimap Y)) = \{M : X \times Y \to \mathbb{R}^+ \mid \forall v \in \mathcal{P}(X)\ M \cdot v \in \mathcal{P}(Y)\}$.

The intuition behind Definition 4.1 is that the web of every PCS corresponds to the positive unit ball of a partially-ordered vector space. This idea is used by Ehrhard and Danos [9] to define a functor that maps every PCS to a Banach space. It is possible to show that this vector space can be equipped with a Riesz space structure, where the order is defined pointwise.

Definition 4.4. Given a PCS $(|X|, \mathcal{P}X)$, we define $BX = \{u \in \mathbb{R}^{|X|} \mid |u| \in \mathcal{P}X\}$ and $eX = \bigcup_{\lambda>0} \lambda BX$. The pair $(eX, u \mapsto \sup_{u'\in\mathcal{P}X^{\perp}} \langle |u|, u'\rangle)$ is the normed Riesz space associated with the PCS $(|X|, \mathcal{P}X)$.

It is shown in [9] that eX is a Banach space. Furthermore, the lattice structure can be defined pointwise, making eX a Banach lattice. Later in this section we will show that e can be made into a functor.

4.1 PCoh and duality

Ehrhard and Danos have shown that the partial order plays an important role in understanding how to generalize **PCoh** [9]. In their attempt to obtain an intrinsic representation for probabilistic coherence spaces, the authors use exclusively the norm and cannot prove the equation $e(X^\perp) = e(X)^\perp$, where $e(X)^\perp$ is the norm dual. That said, we can show that this functor preserves order-duality. Note that the proof uses the fact that 2^X is a directed set.

Theorem 4.5. *For every probabilistic coherence space X, there is a natural isomorphism* $e(X^\perp) \cong e(X)^\sigma$.

Proof. If $u \in e(X^\perp)$, consider the element $f_u = \lambda x\,.\,\langle u^+, x\rangle - \langle u^-, x\rangle$. It is possible to show that the function $\lambda x\,.\,\langle u, x\rangle$ is positive and Scott-continuous, therefore order-continuous for every $u \in \mathcal{P}(X)$. Using this result, it is not hard to show that $f_u \in e(X)^\sigma$.

Conversely, consider an element $f \in e(X)^\sigma$. Without loss of generality, we can assume that f is positive. We want to associate to f an element in $e(X^\perp)$. As is shown by [9], we can alternatively characterize the space $e(X)$ as

$$\{u \in \mathbb{R}^{|X|} \mid \exists \lambda > 0\ \forall u' \in \mathcal{P}(X^\perp)\ \langle |u|, u'\rangle \leq \lambda\}.$$

Consider the function $f_\delta = \lambda x\,.\,f(\delta_x)$. Let us show that $f_\delta \in e(X^\perp)$. To do this, we show that for every $u \in \mathcal{P}(X)$, $\langle |f'|, u\rangle$ is uniformly bounded. Let $(u_\alpha)_{\alpha \in \mathbb{P}_{\text{fin}}(X)}$ be the ascending net $u_{\alpha,a} = u_a$ if $a \in \alpha$ and 0 otherwise. By expanding the definition, we get the equality

$$\langle |f_\delta|, u_\alpha\rangle = \sum_{a\in|X|} |f(\delta_a)| u_{\alpha,a} =$$
$$\sum_{a\in|X|} |f(\delta_a u_{\alpha,a})| = \sum_{a\in|X|} f(\delta_a u_{\alpha,a}).$$

We get the last equality from f being a positive function. Since every u_α has finite support, the expression above is well defined.

$$\sum_{a\in|X|} f(\delta_a u_{\alpha,a}) = f\left(\sum_{a\in|X|} \delta_a u_{\alpha,a}\right) = f(u_\alpha)$$

Since f is order-continuous and monotone and $\{u_\alpha\}$ is an increasing net, we can conclude that $\langle |f_\delta|, u\rangle \leq f(u)$, therefore for every $u \in \mathcal{P}(X)$, $\langle |f_\delta|, u\rangle \leq \|f\|$ and $f_\delta \in e(X^\perp)$. If f is not positive, we decompose it as the difference of two positive maps $f = f^+ - f^-$ and define $f_\delta = f^+_\delta - f^-_\delta$.

A direct calculation shows that this is indeed an isomorphism. □

Corollary 4.6. *For every PCS* $(X, \mathcal{P}(X))$ *the vector space eX is a perfect Banach lattice.*

Since convergence for PCS is defined componentwise, it is possible to use a similar proof technique to show

Theorem 4.7. *The operation e is monoidal closed and functorial.*

Proof. The functoriality of e has been proven in Section 5.1 of [9]. □

Another important theorem which is direct to show is.

Theorem 4.8. *The functor* e : **PCoh** $\to$ **PBanLat$_1$** *is full and faithful.*

We still do not know whether this functor also preserves the exponential. Recent work suggests that this might be the case [7].

5 CATEGORIES OF CONES AND PBanLat$_1$

Even though **PBanLat$_1$** is a mathematically natural model of linear logic, it relies on tools from functional analysis not usually familiar to computer scientists. On the other hand, in recent years, cones have found numerous applications in semantics of programming languages and logics. In this section we show that **PBanLat$_1$** is isomorphic to a category cones, meaning that computer scientists can translate their intuitions about cones to this novel setting without having to learn functional analysis.

As it was frequently mentioned throughout this paper, every Banach lattice gives rise to a positive cone. Furthermore, since every **PBanLat$_1$** morphism $f : V \to W$ is positive and has norm at most 1, it restricts to a linear function $\mathcal{B}(V)^+ \to \mathcal{B}(W)^+$. With this observations we state a few definitions from [7, 13], which assume that the cones are separated.

Definition 5.1. A cone C is a $\mathbb{R}^+$-semimodule with a norm $\|\cdot\| : C \to \mathbb{R}^+$.

Every cone can be equipped with the partial order $x \leq y$ if and only if there is a z such that $x + z = y$, meaning that it is possible to define a partial subtraction operation whenever $x \leq y$, calling $y - x$ the element such that $x + (y - x) = y$.

A function $f : C_1 \to C_2$ between cones is linear if it commutes with addition and scalar multiplication, it is monotonic if it preserves the order relation, and it is Scott-continuous if for every directed set x_α with supremum x, $\sup_\alpha f(x_\alpha) = f(x)$. As is the case with partially-ordered vector spaces, there are different classes of cones where the order and the norm have particular properties:

Definition 5.2. A cone C is said to be:

- Sequentially complete if every norm-bounded sequence has a least upper bound.
- Directed complete if every norm-bounded directed set has a least upper bound.
- A lattice cone if the poset structure is a lattice.

Using this notation, it seems appropriate to imagine that there should be a functor **PBanLat$_1$** $\to$ **CLat**, where **CLat** is the category of directed complete cone lattices. It is unclear, however, if there is a mapping on morphisms. Luckily, the lemma below guarantees that the mapping is well-defined.

Lemma 5.3. *Let V and W be two perfect Banach lattices and* $f : V \to W$ *a linear, positive function of norm at most* 1*. The function f is order-continuous if and only if* $\sup_{x\in A} f(x) = f(v)$ *whenever* $A \subseteq V^+$ *is a non-empty upwards-directed set with supremum v.*

Proof. This result is a direct consequence of the weak Fatou property. □

Since the mapping on morphisms is basically the identity, the functorial laws hold, which allows us to conclude that there is a functor **PBanLat$_1$** $\to$ **CLat**.

Next, we would like to map every positive cone to a vector space. Let C be a positive cone and define

$$C - C = \{(c_1, c_2) \mid c_1, c_2 \in C\}/\sim,$$

where $\sim$ is the binary relation $(c_1, c_2) \sim (c_3, c_4)$ iff $c_1 + c_4 = c_2 + c_3$. Intuitively, $C - C$ corresponds to the vector space of formal differences $c_1 - c_2$ of elements in C. The equivalence relation is used to capture the fact that, for instance, $(3, 2)$ and $(4, 3)$ should represent the same real number, since $3 - 2 = 1 = 4 - 3$.

Theorem 5.4. *Let C be a directed complete cone lattice. Then $C - C$ is a perfect Banach lattice.*

Proof. The proof can be found in the Appendix C. □

By linearity, Scott-continuous functions $f : C \to D$ with norm at most 1 extend to order-continuous functions $f : (C-C) \to (D-D)$ with norm at most 1 and we can prove that there is a functor $\mathbf{CLatLin} \to \mathbf{PBanLat_1}$. With this functor and the positive cone restriction functor defined, it is a direct calculation to show:

Theorem 5.5. *The categories $\mathbf{PBanLat_1}$ and $\mathbf{CLatLin}$ are isomorphic.*

6 AN ENRICHED PROBABILISTIC RECURSIVE CALCULUS

Though it is theoretically interesting understanding how $\mathbf{PBanLat_1}$ relates to existing models of linear logic, we are also interested in using it as a semantic basis for a language with probabilistic primitives. Being symmetric monoidal closed, it can give semantics to the linear λ-calculus. This, however, is insufficient from a programming point of view. The linearity restrictions are severely limiting in terms of which programs one can define in this language. A frequently used solution to this lack of expressivity is to use the exponential modality, where the coKleisli category is Cartesian closed, meaning that it can interpret the λ-calculus.

There are two problems with this approach. The first one is that since the λ-calculus has unrestricted β-reduction, it models a call-by-name (CBN) reduction strategy, which is ill-fit for programming with probabilities, since under this semantics, programs are not able to reuse samples. The second problem is more subtle: since probability theory usually proves theorems about random variables or Markov kernels, and not spaces of functionals over power series, it would be hard to apply useful lemmas and theorems from probability theory when reasoning about programs.

The first point has been addressed in previous work [14], where the authors present a modification of their cone-based semantic domain so that it can accommodate a sample reuse operation with the following typing rule:

$$\frac{\Gamma \vdash t : \mathbb{R} \qquad \Gamma, x : \mathbb{R} \vdash u : \mathbb{R}}{\Gamma \vdash \mathsf{let}\, t = x\, \mathsf{in}\, u : \mathbb{R}}$$

Operationally, the program above samples a real number from t, binds the value to x and continues as u, where x will not modify its value throughout the execution of u. In their semantics, $\mathbb{R}$ is interpreted as the positive cone of measures over the real line.

However, this semantics still suffer from the second problem. Markov kernels are not central in their semantics, absolute monotonic functions are. We argue that in a probabilistic programming languages kernels should be at the foreground while the exponential should only be as a syntactic artefact for programming more intricate kernels with the expressive power of the λ-calculus.

In this section we build on recent work [3] that proposes a new syntax for programming with linear operators and Markov kernels. The proposed calculus will deal with both problems at the same time and will maintain a nice separation between value and computation variables at the syntactic level.

The λ_{MK}^{LL} metalanguage. The semantic structure used to interpret the calculus of [3] is given by a triple $(\mathbf{C}, \mathbf{L}, \mathcal{M})$, where $\mathbf{C}$ is roughly a category of Markov kernels[3], $\mathbf{L}$ is a symmetric monoidal closed category and $\mathcal{M} : \mathbf{C} \to \mathbf{L}$ is a lax monoidal functor.

This two-level structure manisfests itself at the syntactic level by having a two-level syntax: the first level is used to program kernels while the second one serves as a kind of metalanguage that has access to higher-order functions, both of which are depicted in Figure 2, where the highlighted parts of the grammar are the extensions that we will explain in the next section. The linear language has linear function types, which allows for higher-order programming and, unlike most languages based on linear logic, it has a modality $\mathcal{M}$, which corresponds to the types that may be sampled from. The variables bound by the linear context are, roughly speaking, computations. In the language for kernels there are no linearity restrictions and, therefore, variables, i.e. samples from distributions, can be freely duplicated and discarded. Under this perspective, the variables in MK programs should be thought of as values. The intuition behind this language is that linearity forbids distributions to be sampled more than once, but once you have the sample in hands, it can be used as many times as you want.

Each layer has its own typing judgement relations $\vdash_{LL}$ and $\vdash_{MK}$, which we go over in more detail in Appendix A. We highlight one of the most interesting rules; it is the rule that allows programs to be transported between layers:

Sample

$$\frac{x_1 : \tau_1 \cdots x_n : \tau_n \vdash_{MK} M : \tau \qquad \Delta; \Gamma_i \vdash_{LL} t_i : \mathcal{M}\tau_i \qquad 0 < i \leq n}{\Delta; \Gamma_1, \cdots, \Gamma_n \vdash_{LL} \mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ M : \mathcal{M}\tau}$$

Operationally, it samples from n LL programs $\{t_i\}_i$, each sample is bound to the corresponding variable in $\{x_i\}_i$ and finally the continuation M is executed.

We want to model λ_{MK}^{LL} with $\mathbf{PBanLat_1}$. For that we still need a CD category and a lax monoidal functor. For the CD category we will use the category of measurable spaces and sub-Markov kernels.

Definition 6.1. The category **sStoch** has measurable spaces as objects and sub-Markov kernels as morphisms, i.e. measurable functions between a measurable space and the space of subprobability distributions over a measurable space.

sStoch is a CD category, which means that it is symmetric monoidal, with the monoidal product being the product measurable space.

Theorem 6.2. *There is a lax monoidal functor $\mathcal{M} : \mathbf{sStoch} \to \mathbf{PBanLat_1}$.*

Proof. There is a standard functor $\mathcal{M}$ that maps measurable sets to the vector space of signed measures and sub-Markov kernels $f : A \to MB$ to the linear function $\mathcal{M}f(\mu) = \int f d\mu$. The proof of linearity is standard, but order-continuity requires a few words. Let $\{\mu_\alpha\} \downarrow 0$ be a descending arrow, $\mathcal{M}f(\mu_\alpha) = \int f d\mu_\alpha \leq \int 1 d\mu_\alpha =$

[3] a CD category, to be more precise

Variables	x, y, z		
Reals	r	$\in$	$\mathbb{R}$
MK Expressions	M	$::=$	$x \mid r \mid \mathsf{uniform} \mid (M_1, M_2) \mid \pi_1\, M \mid \pi_2\, M \mid \mathsf{let}\ x = M\ \mathsf{in}\ N \mid t(M)$
LL Expressions	t, u	$::=$	$x \mid \lambda x\,.\,t \mid t\ u \mid t \otimes u \mid \mathsf{let}\ x \otimes y = t\ \mathsf{in}\ u \mid \mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ M$
$\mid\ !t \mid \mathsf{let}\ !x = t\ \mathsf{in}\ u \mid \mathsf{fix}\ t$			
Types MK	τ	$::=$	$\mathbb{R} \mid \tau \times \tau$
Types LL	$\underline{\tau}$	$::=$	$1 \mid \mathcal{M}\tau \mid \underline{\tau} \multimap \underline{\tau} \mid \underline{\tau} \otimes \underline{\tau} \mid\ !\underline{\tau}$
Linear Contexts	Γ	$::=$	$x_1 : \underline{\tau}_1, \dots, x_n : \underline{\tau}_n$
MK Contexts	Γ	$::=$	$x_1 : \tau_1, \dots, x_n : \tau_n$

Figure 2: Terms and Types of the Enriched λ_{MK}^{LL}

$\mu_\alpha(A)$ which, as μ_α goes to zero, so does $\mu_\alpha(A)$, making $\tilde{f}$ order-continuous. The functorial laws also follows from standard proofs from the literature.

To show that $\mathcal{M}$ is lax monoidal, we need to define a natural transformation $\mu_{X,Y} : \mathcal{M}(X) \otimes \mathcal{M}(Y) \to \mathcal{M}(X \times Y)$ which is easily defined by the universal property of the tensor product and a morphism $\varepsilon : \mathbb{R} \multimap \mathcal{M}(1)$ which maps a real number r to the measure $r\delta_{\{*\}}$, where $*$ is the only member of the singleton set 1. Showing that the necessary diagrams commute follows from the universal property of the tensor product. □

This means that the triple (**sStoch**, $\mathbf{PBanLat_1}$, $\mathcal{M}$) is a λ_{MK}^{LL} model. Though this is already provides a syntax for programming with linear operators and kernels, it is still not sufficiently expressive. While it can write programs that samples from a distribution and uses the sample more than once:

$$\mathsf{sample\ uniform\ as}\ x\ \mathsf{in}\ (x + x)$$

It cannot write other seemingly reasonable programs. Assume that λ_{MK}^{LL} has a primitive $\mathsf{normal} : \mathcal{M}\mathbb{R} \multimap \mathcal{M}\mathbb{R}$ where, given an input r, outputs the normal distribution with variance 1 and mean r, the following program is not well-typed:

$$\mu : \mathcal{M}\mathbb{R} \nvdash_{LL} \mathsf{sample}\ \mu\ \mathsf{as}\ x\ \mathsf{in}\ (\mathsf{normal}\, x) + (\mathsf{normal}\, x) : \mathcal{M}\mathbb{R}$$

even though it never samples more than once from distributions.

Extending λ_{MK}^{LL}. While the description of λ_{MK}^{LL} above gives some context on how to program with it, it is not very well detailed. A more thorough introduction to the language is given in Appendix A. In this section we extend λ_{MK}^{LL} in order to address some of its expressivity limitations by adding recursion, a "let" rule similar to the one defined in [14] and the non-linear modality.

By using the fact that $\mathbf{PBanLat_1}$ is enriched over **CPO** and contains a linear logic exponential, the type system can be readily extended by adding the type constructor ! and the following typing rules:

$$\textsc{!-Var}\ \frac{x : \tau \in \Delta}{\Delta; \cdot \vdash x : \tau} \qquad \textsc{!-Intro}\ \frac{\Delta; \cdot \vdash_{LL} t : \tau}{\Delta; \cdot \vdash_{LL}\ !t : !\tau}$$

$$\textsc{!-Elim}\ \frac{\Delta; \Gamma_1 \vdash_{LL} t : !\overline{\tau_1} \qquad \Delta, x : \overline{\tau_1}; \Gamma_2 \vdash_{LL} u : \overline{\tau_2}}{\Delta; \Gamma_1, \Gamma_2 \vdash_{LL} \mathsf{let}\ !x = t\ \mathsf{in}\ u : \overline{\tau_2}}$$

Note that the typing judgement has two contexts: one for linear variables and one for non-linear variables. The variable rule is interpreted by the counit ε. For the introduction rule, the linear context to be empty and its semantics is given by $!\, [\![t]\!] \circ \rho$ while the elimination rule is the sequential composition of $[\![t]\!]$ and $[\![u]\!]$.

The recursion operation is captured by the following rule:

$$\textsc{LetRec}\ \frac{\Delta, x : !\underline{\tau}; \Gamma \vdash_{LL} t : \tau}{\Delta; \Gamma \vdash_{LL} \mu x.t : \underline{\tau}}$$

Its semantics is defined using the Fix operator defined in Section 3.5. In order to encode a "let" rule similar to the one from [14] we will use ideas from enriched category theory. Ideally, a program of type $\mathcal{M}\tau_1 \multimap \mathcal{M}\tau_2$ should be a kernel and, therefore, it should be allowed to be applied to an MK program of type τ_1. Syntactically, this translates to the following rule:

$$\frac{\Delta; \Gamma_1 \vdash_{LL} t : \mathcal{M}\tau \multimap \mathcal{M}\tau' \qquad \Delta; \Gamma_2; \Xi \vdash_{MK} M : \tau}{\Delta; \Gamma_1, \Gamma_2; \Xi \vdash_{MK} t(M) : \tau'}$$

Note that this rule requires the typing judgement for MK programs to also depend on LL contexts. Though this rule looks promising, semantically, it is not true that that every morphism $\mathcal{M}\tau_1 \to \mathcal{M}\tau_2$ in $\mathbf{PBanLat_1}$ is a kernel. Categorically, the functor $\mathcal{M}$, even though it is faithful, it is not full[4].

We can now motivate the main theorem of this section. The rule above is very appealing from a programming point of view. As such, we will call this extension the "enriched λ_{MK}^{LL}".

Definition 6.3. An enriched λ_{MK}^{LL} model is a triple $(\mathbf{C}, \mathbf{L}, \mathcal{M})$ where $\mathbf{C}$ is a CD category, $\mathbf{L}$ is symmetric monoidal closed, and $\mathcal{M}$ is a full an faithful lax monoidal functor.

The fullness and faithfulness of $\mathcal{M}$ can be leveraged to interpret the new application rule as $\mathcal{M}^{-1}([\![t]\!]_{LL}) \circ [\![M]\!]_{MK}$.

With this definition in mind, is it possible to define a subcategory of $\mathbf{PBanLat_1}$ such that $\mathcal{M}$, when restricted to this subcategory, is full? Of course, since we want this subcategory to still be a model of λ_{MK}^{LL}, we ask it to be a model of intuitionistic linear logic.

Theorem 6.4. *There is a model of intuitionistic linear logic* $\mathbf{L}$ *such that there is a forgetful functor* $U : \mathbf{L} \to \mathbf{PBanLat_1}$ *and the functor*

[4]The space of signed measures can be decomposed as the space of discrete measures and the space of continuous measures. In this case it is possible to define a non-kernel linear operator by pattern-matching on the measure, see [14] for a concrete example.

$\mathcal{M} : \mathbf{sStoch} \to \mathbf{PBanLat_1}$ *lifts to a full and faithful lax monoidal functor* $\widetilde{\mathcal{M}} : \mathbf{sStoch} \to \mathbf{L}$ *with* $\mathcal{M} = U \circ \widetilde{\mathcal{M}}$.

Proof sketch. We proceed this proof by observing that by looking at the programs that are actually definable in λ^{LL}_{MK}, the functor $\mathcal{M}$ is already full. What we must do is to propagate this information to the whole $\mathbf{PBanLat_1}$ and carve out our subcategory.

We achieve this propagation by a categorical logical relations argument, also known as categorical gluing. The full proof can be found in Appendix D. □

Corollary 6.5. *The triple* $(\mathbf{sStoch}, \mathbf{L}, \widetilde{\mathcal{M}})$ *is a model to the enriched* λ^{LL}_{MK}.

Example 6.6. Assume that there is a kernel $\vdash_{LL} f : \mathcal{M}\tau_1 \multimap \mathcal{M}\tau_2$ and you want to compose it with itself n times, for a given natural number. This can be expressed as the following recursive program:

$$\mu F : !(\mathcal{M}(\mathbb{N} \times \tau)) \multimap \mathcal{M}(\tau).\, \lambda v : \mathcal{M}(\mathbb{N} \times \tau).$$
$$\text{sample } v \text{ as } (n, x) \text{ in}$$
$$\text{if } n == 0 \text{ then } x \text{ else } F(n-1, f\,x)$$

When n is 0, it returns the identity function over $\mathcal{M}\tau$, otherwise if $n > 0$, it applies f once more and calls the recursive function with $n - 1$.

Example 6.7. In order to justify the applicability of the enriched λ^{LL}_{MK} calculus as a language for programming computational statistics algorithms, consider the following fragment of the Gibbs sampling mechanism, where it is assumed there are kernels $f_1 : M(X \times Y) \multimap \mathcal{M}(X \times Y\ Z)$, $f_2 : M(X \times Z) \multimap \mathcal{M}(X \times Y\ Z)$, $f_3 : M(Y \times Z) \multimap \mathcal{M}(X \times Y\ Z)$ and a distribution $v : \mathcal{M}(X \times Y \times Z)$:

$$\lambda f_1.\, \lambda f_2.\, \lambda f_3.\, \lambda v.$$
$$\text{sample } v \text{ as } (x, y, z) \text{ in}$$
$$\text{let } (x_1, y_1, z_1) = f_1\,(x, y) \text{ in}$$
$$\text{let } (x_2, y_2, z_2) = f_2\,(x, z_1) \text{ in}$$
$$f_3\,(y_2, z_1)$$

This program does one "loop" of the Gibbs sampling algorithm. Theoretically, this mechanism, in the limit, will converge to the correct distribution. In order to simulate this limit, you can compose the program above with itself an arbitrary number of times, by using Example 6.6.

7 RELATED WORK

There have been a number of semantics of linear logic based on vector space-like objects. Two important families of such semantics are the ones based on probabilistic coherence spaces and the ones based on Banach spaces. As we will explain below, we see our model as a nice synthesis of these two approaches.

Positive Cone Semantics of Linear Logic. [9] shows that probabilistic coherence spaces are a model of classical linear logic and can interpret a call-by-name probabilistic functional programming language. [39] extends their call-by-name λ-calculus with a call-by-value let construct which can be easily interpreted in their original model.

To overcome the limitation that **PCoh** cannot represent continuous distributions, Ehrhard et al. define a cartesian closed category $\mathbf{CStab_m}$ [14], which uses normed $\mathbb{R}^+$-semimodule—which are in correspondence with positive cones of partially ordered vector spaces—to interpret a probabilistic variant of PCF with continuous distributions. In a follow-up paper, Ehrhard [13] has defined a category $\mathbf{CLin_m}$ of sequentially complete positive cones with measurability paths and linear Scott continuous maps in which **PCoh** embeds fully and faithfully.

A similar approach was taken in [37], which defined a category **CCones** of so-called coherent cones and linear contractive functions and showed that it is a model of classical linear logic. These cones come equipped with a different notion of completeness that is stronger than sequential completeness but weaker than ours.

From a mathematical point of view, the objects of both **CCones** and $\mathbf{CStab_m}$ are not as well understood as Banach lattices, making them not ideal semantic frameworks to reason about probabilistic programs. Besides, our model provides a clear mathematical justification for having Fatou-like properties in the semantics: it is forced upon it by Theorem 2.28 instead of being there for denotational reasons, as is the case of $\mathbf{CStab_m}$, or in enabling the exponential construction, as is the case of **CCones**, showing a kind of canonicity of our model.

Vector Space Semantics of Linear Logic. Dahlqvist and Kozen [8] have defined a category of partially ordered Banach spaces **RoBan**, shown that it is a model of intuitionistic linear logic, and used it to interpret a higher-order imperative probabilistic language with while loops and soft-conditioning.

Their model also uses a mathematically well-understood class of vector spaces. That being said, by using a more general class of vector spaces than we do, their model has less structure than ours. A practical consequence of this lack of structure is that in order to guarantee the soundness of their semantics, they define 6 type grammars that are used for different program constructs. As an example, in order to interpret conditionals and while loops the context may only have Dedekind complete types.

Furthermore, their semantics does not have an equivalent to the weak Fatou property and cannot interpret our fixed point operator. In their language, recursion can only be expressed with while-loops.

Another relevant vector space model is the one based on complex coherent Banach spaces [21]. These vector spaces are not partially ordered, making them unsuitable for interpreting recursive programs, they have an involutive linear negation and the exponential was defined in a time when the categorical semantics of the modality was not fully developed and, as a consequence, morphisms $!A \multimap B$ do not compose, making it not a model of linear logic as they are currently defined.

Neither **RoBan** nor $\mathbf{CStab_m}$ are models of classical linear logic.

Languages for Probabilistic Programming. A lot of work has been done in working out what are the right abstractions for programming statistical models. In [28] the authors define a functional programming language with programmable inference, where the programmers may fine-tune certain parameters of general inference algorithms that are exposed to the users as primitives in the language. There are other languages that make inference a primitive in the language [6, 10, 22, 30, 40].

Other languages such as Dice [24], focuses on discrete inference problems in order to handle larger programs. In these kinds of languages, inference is computed exactly. As such, Dice can be classified as a language for symbolic inference, and is one of many that adopt a more symbolic approach to inference [19, 32–34, 36, 38]

Sitting at a higher-level of abstraction, the core calculus presented in [8] has an inference primitive which is interpreted using the adjoint of linear operators. Unfortunately, the authors to do connect it to an actual implementation.

Though the enriched λ_{MK}^{LL} calculus has no primitive for inference, it can implement sampling algorithms such as the Gibbs algorithm, showing that it at least has the potential of implementing modern inference algorithms. Furthermore, it might be possible to use the two-level structure of the calculus as a link between the conceptual operations for inference advocated for by [8] and the low-level implementation details of actual inference algorithms.

8 CONCLUSION

In this paper we have shown that $\mathbf{PBanLat_1}$ is a model of classical linear logic that conservatively extends **PCoh** and can be used to give semantics to a recursive probabilistic calculus. Our model differs from existing extensions of **PCoh** that only extends **PCoh**'s intuitionistic fragment, meaning that they do not have an involutive negation. We believe that our model is a good fit for formal verification purposes because Riesz spaces have decades of research and have been extensively used in the formalization of stochastic processes.

This work creates some exciting directions for future work. Ehrhard has shown that the category of Kothe spaces is a model of differential linear logic [11]. The model looks very similar to **PCoh**, with the exception that they do not require a norm. It is a natural question to ask if by working with perfect unnormed Riesz spaces instead of perfect Banach lattices one gets a model of differential linear logic. Furthermore, this would provide a preliminary answer to the challenge of giving semantics to probabilistic differentiable languages.

ACKNOWLEDGEMENTS

[suppressed for double-blind reviewing]

REFERENCES

[1] Charalambos D Aliprantis and Owen Burkinshaw. 2006. *Positive operators.* Springer.

[2] Robert J Aumann. 1961. Borel structures for function spaces. *Illinois Journal of Mathematics* (1961).

[3] Pedro H. Azevedo de Amorim. 2023. A Higher-Order Language for Markov Kernels and Linear Operators. In *Foundations of Software Science and Computation Structures (FoSSaCS).*

[4] P Nick Benton. 1994. A mixed linear and non-linear logic: Proofs, terms and models. In *International Workshop on Computer Science Logic.*

[5] Richard Blute, Thomas Ehrhard, and Christine Tasson. 2012. A Convenient Differential Category. *Cahiers de topologie et géométrie différentielle catégoriques* 53, 3 (2012), 211–232.

[6] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. Stan: A probabilistic programming language. *Journal of statistical software* (2017).

[7] Raphaëlle Crubillé. 2018. Probabilistic stable functions on discrete cones are power series. In *Logic in Computer Science (LICS).*

[8] Fredrik Dahlqvist and Dexter Kozen. 2019. Semantics of higher-order probabilistic programs with conditioning. In *Principles of Programming Languages (POPL).*

[9] Vincent Danos and Thomas Ehrhard. 2011. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Information and Computation* 209, 6 (2011), 966–991.

[10] Swaraj Dash, Younesse Kaddar, Hugo Paquet, and Sam Staton. 2023. Affine monads and lazy structures for bayesian programming. In *Principles of Programming Languages (POPL).*

[11] Thomas Ehrhard. 2002. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science* 12, 5 (2002), 579–623.

[12] Thomas Ehrhard. 2019. Differentials and distances in probabilistic coherence spaces. *arXiv preprint arXiv:1902.04836* (2019).

[13] Thomas Ehrhard. 2020. On the linear structure of cones. In *Logic in Computer Science (LICS).*

[14] Thomas Ehrhard, Michele Pagani, and Christine Tasson. 2017. Measurable cones and stable, measurable functions: a model for probabilistic higher-order programming. In *Principles of Programming Languages (POPL).*

[15] Thomas Ehrhard, Christine Tasson, and Michele Pagani. 2014. Probabilistic coherence spaces are fully abstract for probabilistic PCF. In *Principles of Programming Languages (POPL).*

[16] DH Fremlin. 1968. Abstract Köthe spaces IV. In *Mathematical Proceedings of the Cambridge Philosophical Society.* Cambridge University Press, 45–52.

[17] David H Fremlin. 2000. *Measure theory.* Torres Fremlin.

[18] Tobias Fritz. 2020. A synthetic approach to Markov kernels, conditional independence and theorems on sufficient statistics. *Advances in Mathematics* 370 (2020), 107239.

[19] Timon Gehr, Sasa Misailovic, and Martin Vechev. 2016. PSI: Exact symbolic inference for probabilistic programs. In *Computer Aided Verification (CAV).*

[20] Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence* (1984).

[21] Jean-Yves Girard. 1999. Coherent Banach spaces: a continuous denotational semantics. *Theoretical Computer Science* 227, 1-2 (1999), 275–297.

[22] Noah D Goodman, Vikash K Mansinghka, Daniel Roy, Keith Bonawitz, and Joshua B Tenenbaum. 2008. Church: a language for generative models. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence.*

[23] Chris Heunen, Ohad Kammar, Sam Staton, and Hongseok Yang. 2017. A convenient category for higher-order probability theory. In *Logic in Computer Science (LICS).*

[24] Steven Holtzen, Guy Van den Broeck, and Todd Millstein. 2020. Scaling exact inference for discrete probabilistic programs. (2020).

[25] Martin Hyland and Andrea Schalk. 2003. Glueing and orthogonality for models of linear logic. *Theoretical computer science* (2003).

[26] Marie Kerjean and Christine Tasson. 2018. Mackey-complete spaces and power series–a topological model of differential linear logic. *Mathematical Structures in Computer Science* 28, 4 (2018), 472–507.

[27] Dexter Kozen. 1979. Semantics of probabilistic programs. In *Symposium on Foundations of Computer Science (SFCS).*

[28] Alexander K Lew, Marco F Cusumano-Towner, Benjamin Sherman, Michael Carbin, and Vikash K Mansinghka. 2019. Trace types and denotational semantics for sound programmable inference in probabilistic languages. In *Principles of Programming Languages (POPL).*

[29] WAJ Luxemberg and AC Zaanen. 1963. Notes on Banach function spaces VI-XIII. *Proceedings of the Koninklijke Nederlandse Akademie van Wetenschappen, Series A* 66 (1963), 251–263.

[30] Vikash K Mansinghka, Ulrich Schaechtle, Shivam Handa, Alexey Radul, Yutian Chen, and Martin Rinard. 2018. Probabilistic programming with programmable inference. In *Programming Language Design and Implementation (PLDI).*

[31] Paul-André Mellies. 2009. Categorical semantics of linear logic. *Panoramas et synthèses* 27 (2009), 15–215.

[32] Praveen Narayanan, Jacques Carette, Wren Romano, Chung-chieh Shan, and Robert Zinkov. 2016. Probabilistic inference by program transformation in Hakaru (system description). In *Functional and Logic Programming (FLOPS).*

[33] Praveen Narayanan and Chung-chieh Shan. 2020. Symbolic disintegration with a variety of base measures. *ACM Transactions on Programming Languages and Systems (TOPLAS)* (2020).

[34] Feras A Saad, Martin C Rinard, and Vikash K Mansinghka. 2021. SPPL: probabilistic programming with fast exact symbolic inference. In *Programming Language Design and Implementation (PLDI).*

[35] H. H. Schaefer. 1970. *Banach Lattices and Positive Operators.* Springer.

[36] Chung-chieh Shan and Norman Ramsey. 2017. Exact Bayesian inference by symbolic disintegration. In *Principles of Programming Languages (POPL).*

[37] Sergey Slavnov. 2021. Linear logic in normed cones: probabilistic coherence spaces and beyond. *Mathematical Structures in Computer Science* 31, 5 (2021), 495–534.

[38] Dario Stein and Sam Staton. 2021. Compositional semantics for probabilistic programs with exact conditioning. In *Logic in Computer Science (LICS).*

[39] Christine Tasson and Thomas Ehrhard. 2019. Probabilistic call by push value. *Logical Methods in Computer Science* 15 (2019).

[40] David Tolpin, Jan-Willem van de Meent, Hongseok Yang, and Frank Wood. 2016. Design and implementation of probabilistic programming language anglican. In *Implementation and Application of Functional programming Languages (IFL).*

[41] Martinus Bernardus Jozephus Gerhardus van Haandel. 1993. *Completions in Riesz space theory.* Ph.D. Dissertation. Katholieke Universiteit Nijmegen.
[42] Adriaan C Zaanen. 2012. *Introduction to operator theory in Riesz spaces.* Springer.

$$\frac{}{\Gamma, x:\tau \vdash_{MK} x:\tau}\textsc{Var} \qquad \frac{r\in\mathbb{R}}{\Gamma\vdash_{MK} r:\mathbb{R}}\textsc{Const} \qquad \frac{}{\Gamma\vdash_{MK} \mathsf{uniform}:\mathbb{R}}\textsc{Uniform} \qquad \frac{\Gamma\vdash_{MK} t:\tau_1 \quad \Gamma, x:\tau_1\vdash_{MK} u:\tau}{\Gamma\vdash_{MK} \mathsf{let}\ x=t\ \mathsf{in}\ u:\tau}\textsc{Let}$$

$$\frac{\Gamma\vdash_{MK} t:\tau_1 \quad \Gamma\vdash_{MK} u:\tau_2}{\Gamma\vdash_{MK} (t,u):\tau_1\times\tau_2}\textsc{Pair} \qquad \frac{\Gamma\vdash_{MK} t:\tau_1\times\tau_2}{\Gamma\vdash_{MK} \pi_1 t:\tau_1}\textsc{Proj1} \qquad \frac{\Gamma\vdash_{MK} t:\tau_1\times\tau_2}{\Gamma\vdash_{MK} \pi_2 t:\tau_2}\textsc{Proj2} \qquad \frac{}{x:\tau\vdash_{LL} x:\tau}\textsc{Axiom}$$

$$\frac{}{\cdot\vdash_{LL} \mathsf{unit}:1}\textsc{Unit} \qquad \frac{\Gamma, x:\tau_1\vdash_{LL} t:\tau_2}{\Gamma\vdash_{LL} \lambda x.t:\tau_1\multimap\tau_2}\textsc{Abstraction} \qquad \frac{\Gamma_1\vdash_{LL} t:\tau_1\multimap\tau_2 \quad \Gamma_2\vdash_{LL} u:\tau_1}{\Gamma_1,\Gamma_2\vdash_{LL} t\ u:\tau_2}\textsc{Application} \qquad \frac{\Gamma_1\vdash_{LL} t:\tau_1 \quad \Gamma_2\vdash_{LL} u:\tau_2}{\Gamma_1,\Gamma_2\vdash_{LL} t\otimes u:\tau_1\otimes\tau_2}\textsc{Tensor}$$

$$\frac{\Gamma_1\vdash_{LL} t:\tau_1\otimes\tau_2 \quad \Gamma_2, x:\tau_1, y:\tau_2\vdash_{LL} u:\tau}{\Gamma_1,\Gamma_2\vdash_{LL} \mathsf{let}\ x\otimes y=t\ \mathsf{in}\ u:\tau}\textsc{LetTensor} \qquad \frac{x_1:\tau_1\cdots x_n:\tau_n\vdash_{MK} M:\tau \quad \Delta;\Gamma_i\vdash_{LL} t_i:\mathcal{M}\tau_i \quad 0\le i<n}{\Delta;\Gamma_1,\cdots,\Gamma_n\vdash_{LL} \mathsf{sample}\ t_i\ \mathsf{as}\ x_i\ \mathsf{in}\ M:\mathcal{M}\tau}\textsc{Sample}$$

Figure 3: Typing rules for λ_{MK}^{LL}

A A METALANGUAGE FOR LINEAR OPERATORS AND MARKOV KERNELS

In this section we further explain the two-level language λ_{MK}^{LL} and its semantics. The language MK corresponds to an effectful language with probabilistic primitives and where free variables are assumed to be values, as opposed to computations. For instance, the program $x:\mathbb{N}, y:\mathbb{N}\vdash_{MK} x+y:\mathbb{N}$ is interpreted as a deterministic program. This language is interpreted in a CD category, which can be seen as an abstraction for programming with commutative effects [18].

Definition A.1. CD categories are symmetric monoidal categories such that every object A has a commutative comonoid structure $\mathsf{copy}_A : A\to A\otimes A$ and $\mathsf{delete}_A : A\to 1$ satisfying certain structural properties.

In the context of probabilistic programming, there are many CD categories to choose from. In particular, for any subprobability monad, its Kleisli category is a CD category. This is the case for the **sStoch** category, since it can be characterized as the category of measurable sets and measurable functions $A\to\mathcal{G}(B)$, where $\mathcal{G}$ is the subprobability monad over **Meas**.

The language LL is basically a linear λ-calculus. By itself, linearity limits the expressivity of the language quite a bit. In the original paper, the author argues that for probabilistic programming, the linear usage of variables is, semantically, too restrictive, since many probabilistic which are linear, in the semantic sense, may use variables for than once [3]. This observation led to the introduction of the $\mathcal{M}$ modality in the LL language which allows MK programs to be called from an LL program. Semantically, this is interpreted as a lax monoidal functor.

Definition A.2. Let **C** and **D** be monoidal categories. A (lax) monoidal functor is a functor $F:\mathbf{C}\to\mathbf{D}$ equipped with a natural transformation $\varepsilon_{A,B}: FA\otimes_{\mathbf{D}} FB\to F(A\otimes_{\mathbf{C}} B)$ and a morphism $I_{\mathbf{D}}\to F(I_{\mathbf{C}})$ making certain coherence diagrams commute.

From a programming point of view, types $\mathcal{M}\tau$ should be thought of as types from which one can sample from. Supposing that the language has a primitive uniform for the uniform distribution over the unit interval the Sample construct can be used to write the program

$$\mathsf{sample\ uniform\ as}\ x\ \mathsf{in}\ (x+x)$$

The program above samples from a uniform distribution and adds the result to itself. This program illustrates why this syntax increases the expressivity of the linear λ-calculus. By allowing the continuation $x+x$ to be an MK program, variables may be freely reused or discards without worrying about syntactic restriction imposed by linearity.

However, once inside the MK language, there is no way of going back to the higher-order language, meaning that the program sample uniform as x in (sample uniform as y in $(x+y)$) is not well-typed. This is mitigated by lax monoidality, which makes it possible to simultaneously sample from distributions: sample (uniform, uniform) as (x,y) in $(x+y)$.

Definition A.3. A model of λ_{MK}^{LL} is a triple $(\mathbf{C},\mathbf{L},\mathcal{M})$, where **C**, a symmetric monoidal closed category **L** and $\mathcal{M}:\mathbf{M}\to\mathbf{C}$ is a lax monoidal functor.

The typing rules are depicted in Figure 3. They are basically the amalgamation of the rules for programming with CD categories, i.e. a first-order expression language with pairs, with symmetric monoidal closed categories, i.e. the linear λ-calculus with tensor types. The main novelty is the introduction of the lax monoidal modality $\mathcal{M}$ and its accompanying typing rule Sample which connects the MK and LL languages.

Much like the typing rules, the categorical semantics of λ_{MK}^{LL} is the combination of the categorical semantics of the internal languages of CD categories and the linear λ-calculus with the exception of the Sample rule that makes use of the functor $\mathcal{M}$. The full semantics is depicted in Figure 4.

$$\textsc{Var}\quad \frac{}{\tau \times \Gamma \xrightarrow{id_\tau \times del_\Gamma} \tau} \qquad \textsc{Let}\quad \frac{\Gamma \xrightarrow{M} \tau_1 \qquad \Gamma \times \tau_1 \xrightarrow{N} \tau_2}{\Gamma \xrightarrow{copy;(id\times M);N} \tau_2} \qquad \times\,\textsc{Intro}\quad \frac{\Gamma \xrightarrow{M} \tau_1 \qquad \Gamma \xrightarrow{N} \tau_2}{\Gamma \xrightarrow{copy;M\times N} \tau_1 \times \tau_2} \qquad \times\,\textsc{Elim}_i\quad \frac{\Gamma \xrightarrow{M} \tau_1 \times \tau_2}{\Gamma \xrightarrow{M;(id_{\tau_i}\times del)} \tau_i}$$

$$\textsc{Var}\quad \frac{}{\underline{\tau} \xrightarrow{id_\tau} \underline{\tau}} \qquad \textsc{Abstraction}\quad \frac{\Gamma \otimes \underline{\tau_1} \xrightarrow{t} \underline{\tau_2}}{\Gamma \xrightarrow{\mathrm{cur}(t)} \underline{\tau_1} \multimap \underline{\tau_2}} \qquad \textsc{Application}\quad \frac{\Gamma_1 \xrightarrow{t} \underline{\tau_1} \multimap \underline{\tau_2} \qquad \Gamma_2 \xrightarrow{u} \underline{\tau_1}}{\Gamma_1 \otimes \Gamma_2 \xrightarrow{(t\otimes u);\mathrm{ev}} \underline{\tau_2}}$$

$$\otimes\,\textsc{Intro}\quad \frac{\Gamma_1 \xrightarrow{t} \underline{\tau_1} \qquad \Gamma_2 \xrightarrow{u} \underline{\tau_2}}{\Gamma_1 \otimes \Gamma_2 \xrightarrow{t\otimes u} \underline{\tau_1} \otimes \underline{\tau_2}} \qquad \otimes\,\textsc{Elim}\quad \frac{\Gamma_1 \xrightarrow{t} \underline{\tau_1} \otimes \underline{\tau_2} \qquad \Gamma_2 \otimes \underline{\tau_1} \otimes \underline{\tau_2} \xrightarrow{u} \underline{\tau}}{\Gamma_1 \otimes \Gamma_2 \xrightarrow{(id\otimes t);u} \underline{\tau}}$$

$$\textsc{Sample}\quad \frac{\tau_1 \times \cdots \times \tau_n \xrightarrow{M} \tau \qquad \Gamma_i \xrightarrow{t_i} \mathcal{M}\tau_i}{\Gamma_1 \otimes \cdots \otimes \Gamma_n \xrightarrow{t_1\otimes\cdots\otimes t_n} \mathcal{M}\tau_1 \otimes \cdots \otimes \mathcal{M}\tau_n \xrightarrow{\mu} \mathcal{M}(\tau_1 \times \cdots \times \tau_n) \xrightarrow{\mathcal{M}M} \mathcal{M}\tau}$$

Figure 4: Categorical Semantics of λ_{MK}^{LL}

B CONTINOUS EXPONENTIAL

Let V be a perfect Banach lattice and consider the positive cone $C(V)^+$ of (norm) continuous bounded functions $f : \mathcal{B}(V)^+ \to \mathbb{R}$. This set can be equipped with the pointwise partial order $f \leq g$ iff $\forall x\, f(x) \leq g(x)$ by using the the sup norm $\|f\| = \sup_{x\in\mathcal{B}(V)^+} |f(x)|$.

Lemma B.1. *The space $C(V)$ is a Riesz space.*

Proof. Let $f, g \in C(V)$ and observe that $f \vee g = \lambda x\,.\, \max(f(x), g(x))$ is bounded and continuous and since we are assuming the pointwise order, it can be shown to be the least upper bound of f and g. □

An *M-space* is a Banach lattice whose norm satisfies $\|x \vee y\| = \|x\| \vee \|y\|$ for positive x, y. If the unit ball of an M-space contains a largest element, it is known as a *unit* of the space.

Theorem B.2. *The space $C(V)$ is an M-space.*

Proof. Let $C_\tau(V)$ be the vector space of bounded (topologically) continuous functions $f : \mathcal{B}(V)^+ \to \mathbb{R}$. In [35, Example 2, p. 103], it is shown that for any topological space X, the vector space of all bounded, real-valued continuous functions on X with the pointwise order is an M-space with unit $\lambda x\,.\,1$. In such spaces, the closed unit ball is the order interval $[-\lambda x\,.\,1, \lambda x\,.\,1]$, so

$$\|f\| \leq 1 \iff f \in [-\lambda x\,.\,1, \lambda x\,.\,1] \iff \sup_{x\in X} |f(x)| \leq 1,$$

therefore $\|f\| = \sup_{x\in X} |f(x)|$. Taking $X = \mathcal{B}(V)^+$, we have that $C_\tau(V)$ is an M-space with norm $\|f\| = \sup_{x\in\mathcal{B}(V)^+} |f(x)|$. □

Corollary B.3. *The space $C(V)$ is a Banach lattice.*

In order to define the exponential $!V$, we are interested in a particular subspace of $C(V)^\sigma$. For every $v \in \mathcal{B}(V)^+$, let $\delta_v \in C(V)^\sigma$ be the *Dirac delta distribution* defined as the functional $\delta_v(f) = f(v)$. We want $!V$ to be a perfect Banach lattice contained in $C(V)^\sigma$ which contains the linear span of the delta distributions as a norm-dense subset. At a high level, our construction comprises two steps. The first step consists of extending the linear span of the Dirac distributions to a Riesz space containing this span as an order-dense subset.

Next, we take the *norm closure* of this Riesz space in $C(V)^\sigma$ and obtain a perfect Banach lattice. By transitivity of order-density, the span of the delta distributions is once again dense in this space.

Free Riesz spaces. Once again we will make use of the free Riesz space construction described in Section 3.4.

Lemma B.4 ([5]). *If $v_1, \ldots, v_n \in \mathcal{B}(V)^+$ are distinct vectors, then the distributions $\delta_{v_1}, \ldots, \delta_{v_n}$ are linearly independent.*

Let $\Delta(V)$ be the normed Riesz space generated by the partially ordered vector space spanned by $\{\delta_v\}_{v\in\mathcal{B}(V)^+}$ according to Theorem 3.11.

L-spaces and order closure. For the next part, we need to introduce some terminology. We say that a Banach lattice V is an *L-space* if for all positive vectors v_1, v_2, $\|v_1 + v_2\| = \|v_1\| + \|v_2\|$. These spaces are also called abstract Lebesgue spaces because they can be seen as a generalization of L_1 spaces. In particular, for every measurable space, the space of signed measures over it is an L-space.

As previously mentioned, a tricky aspect of Banach lattices is that the order and norm topologies do not always give rise to the same continuous functionals. In L-spaces, this distinction disappears. This will be useful in showing that our construction gives both a Riesz and a Banach space.

Lemma B.5. *The space $C(V)^\sigma$ is an L-space.*

Proof. Since $C(V)$ is a Banach lattice, so is $C(V)^\sigma$. Let $F \in C(V)^\sigma$ and positive. Then $\|F\| = F(\lambda x\,.\,1)$, as the function $\lambda x\,.\,1$ is the maximum element of the unit ball in $C(V)$ (see Theorem B.2). Therefore, for positive F_1, F_2, $\|F_1 + F_2\| = F_1(e) + F_2(e) = \|F_1\| + \|F_2\|$. □

We define the *exponential* $!V$ of a space V as the order-closure of $\Delta(V)$ in $C(V)^\sigma$. L-spaces are extremely well-behaved and so are their subspaces. Our proof strategy relies on the fact that L-spaces are always perfect and in showing that $!V$ is an L-space in its own right.

Lemma B.6 (Proposition 356P(a) [17])**.** *Every L-space is perfect.*

Therefore, by showing that $!V$ is an L-space it follows that it is a perfect Banach lattice. A canonical way of obtaining a Banach space out of a normed vector space is by taking its norm-closure. In general, however, since our spaces are also Riesz, it might be the case that the norm-closure is not a Riesz space. Luckily, for L-spaces, order-closed spaces are also norm-closed making them Banach.

Lemma B.7 (Proposition 354O[17])**.** *Every norm-closed Riesz subspace of an L-space is an L-space.*

Lemma B.8. *If V is an L-space and U is order-dense in V then it is also norm-dense.*

Theorem B.9. *The space $!V$ is a perfect Banach lattice with the the span of $\{\delta_v\}_{v \in \mathcal{B}(V)^+}$ as an norm-dense subset.*

Proof. The proof follows by observing that since $!V$ is defined as the norm-closure of a Riesz space, by the corollary above, it is also an L-space, which by Lemma B.6 implies that it is a perfect Banach lattice; the span of the deltas are norm-dense by transitivity of norm-density, since by the lemma above, the deltas are norm-dense in their freely generated Riesz space. □

Lemma B.10. *Let V and W be perfect Banach lattices and $X \subseteq \mathcal{B}(V)^+$ a norm-dense set. Every norm-continuos function $f : X \to \mathcal{B}(W)^+$ extends uniquely to a function $\tilde{f} : \mathcal{B}(V)^+ \to \mathcal{B}(W)^+$.*

Proof. The extension is defined using the fact that $\mathcal{B}(W)^+$ is a complete metric space. □

Linear-Non-Linear adjunction. An important step in our understanding of categorical models of linear logic was taken by Nick Benton [4], where he realized that by taking the coKleisli adjoint factorization of the exponential, you get a very conceptually clean categorical description. We will use his insights in this section to define an alternative exponential to that defined in Section 3.4.

Definition B.11 ([31], Chap. 7, Definition 21)**.** Let $\mathbf{L}$ be symmetric monoidal closed and $\mathbf{M}$ Cartesian. A Linear-Non-Linear adjunction is a strong monoidal adjunction:

$$(\mathbf{M}, \times, 1) \underset{M}{\overset{L}{\rightleftarrows}} (\mathbf{L}, \otimes, I) \quad (L \dashv M)$$

I.e. the adjoint functors preserve the monoidal structures up to isomorphism.

It has been proved that such a categorical setup is enough for the comonad generated by the adjunction to be a linear logic exponential, where the Seely isomorphisms are given by the strong monoidal structure of the adjunction.

Let **OBanLat** be the category with perfect Banach lattices as objects and norm-continuous functions $\mathcal{B}^+(V) \to \mathcal{B}^+(W)$ as morphisms. Since composition is just function composition, this is indeed a category.

Lemma B.12. *The map $\delta : \mathcal{B}^+(V) \to \mathcal{B}^+(!V)$ is norm-continuous.*

Note that, differently from its power series counterpart, the map above is not monotonic.

Example B.13. Consider the endofunction $f(x) = 1 - x$ on the interval $[0, 1]$. The function is a morphism $\mathbb{R} \to \mathbb{R}$ in **OBanLat**. Now, let $0 \leq r \leq s \leq 1$. It is not true that $\delta_r \leq \delta_s$, precisely because $f(s) \leq f(r)$.

Theorem B.14. *There is a monoidal adjunction between* $\mathbf{PBanLat_1}$ *and* **OBanLat**.

Proof. There is an inclusion functor $\mathbf{PBanLat_1} \hookrightarrow \mathbf{OBanLat}$ that maps a space to itself and a linear function to its restriction to the positive unit ball, which is well-defined because morphisms in $\mathbf{PBanLat_1}$ are positive and with norm at most 1.

To go in the other direction, the functor maps spaces V to $!V$ and norm-continous functions f to the function generated by $\delta_v \mapsto \delta_{f(v)}$, which can be extended to a $\mathbf{PBanLat_1}$ morphism because of Lemma B.10.

The adjunction isomorphisms by mapping a linear morphism $f : !V \to W$ to $f \circ \delta : V \to W$ and an order-continuous morphism $g : V \to W$ to the extension of $g \circ \delta$ via Lemma B.10. Showing that they are mutual inverses is a direct calculation and application of the extension lemma. □

C PROOF OF THEOREM 5.4

Let C be a directed complete lattice cone. In order to define functions over it we use the universal property of quotients: it suffices to define it over every pair (c_1, c_2) while guaranteeing that the function acts the same over every equivalence class.

For instance, the vector space structure can be simply defined componentwise. Let $(c_1, c_2), (c_3, c_4) \in C - C$ then we define

$$(c_1, c_2) + (c_3, c_4) = (c_1 + c_3, c_2 + c_4)$$
$$\alpha(c_1, c_2) = (\alpha c_1, \alpha c_2) \text{ for } \alpha \geq 0$$
$$\alpha(c_1, c_2) = (-\alpha c_2, -\alpha c_1) \text{ otherwise}$$

The lattice operations require a bit more ingenuity, and we first observe the equation $u \vee v = u + (v - u)^+$ which holds in every Riesz space, reducing the lowest upper bound operation to addition and the positive part. By doing some algebraic manipulations we get that if $(c_1, c_2), (c_3, c_4) \in C - C$ then we define $(c_1, c_2) \vee (c_3, c_4) = (c_1, c_2) - ((c_3, c_4) - (c_1, c_2))^+ = (c_1, c_2) + (c_3 + c_2 - (c_1 + c_4) \wedge (c_2 + c_3), 0) = (c_1 + c_3 + c_2 - (c_1 + c_4) \wedge (c_2 + c_3), c_2)$. The lattice equations such as commutativity and idempotency follow by unfolding the definitions and from C being a lattice.

Before defining a norm over $C - C$ we first need the following lemma

Lemma C.1. $(C - C)^+ \cong \{(c, 0) \mid c \in C\} \cong C$.

Proof. The mapping $\{(c, 0) \mid c \in C\} \to (C - C)^+$ is the injection through the equivalence class function and the mapping in the other direction can be constructed by observing that whenever $(c_1, c_2) \geq (0, 0)$ it can be shown that $c_1 \geq c_2$ and, therefore, $(c_1 - c_2, 0) = (c_1, c_2)$ and this decomposition is unique, since $(c, 0) = (d, 0)$ implies, by definition of $\sim$ that $c = d$. The second isomorphism is trivial. □

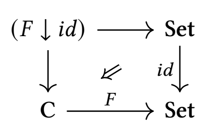
Given a norm over C it is possible to extend it to a norm over $C - C$. This follows from the property of normed Riesz spaces, where $\||v|\| = \|v\|$ which forces us to define $\|(c_1, c_2)\| = \||(c_1, c_2)|\|_C$. Note that since $|(c_1, c_2)|$ is a positive element of $C - C$, by the lemma above it can be mapped back to an element of C which, in turn, has a norm.

Therefore, we have shown that $C - C$ is a normed Riesz space. Since C has the directed completeness property it follows that $C - C$ has the weak Fatou property and, therefore, it is Banach and perfect.

D PROOF OF THEOREM 6.4

Glued models of linear logic.

Definition D.1. Let $\mathbf{C}$ be a category and $F : \mathbf{C} \to \mathbf{Set}$. The comma category $(F \downarrow id)$ is defined as the following lax pullback.

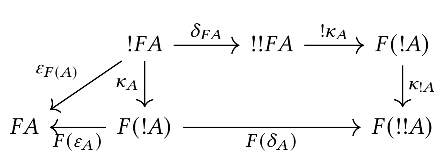


Concretely, its objects are triples $(A : \mathbf{C}, X : \mathbf{Set}, f : X \to F(\mathbf{C}))$ and morphisms are pairs $(f : A \to B, g : X \to Y)$ making the usual diagrams commute. The will use the notation $Gl_F(\mathbf{C})$ to refer to the comma category $(F \downarrow id)$.

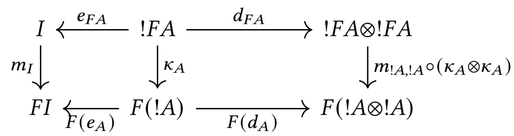
The comma category is one of the building blocks in categorical logical relations. There has been much work done in proving when such a category has similar categorical properties to $\mathbf{C}$ [25].

Definition D.2. Let $\mathbf{C}$ and $\mathbf{D}$ be models of intuitionistic linear logic (ILL). A functor $F : \mathbf{C} \to \mathbf{D}$ will be called distributively linear if it is monoidal (in the usual lax sense) and there is a natural transformation $\kappa : !_{\mathbf{D}} F \to F !_{\mathbf{C}}$ such that:

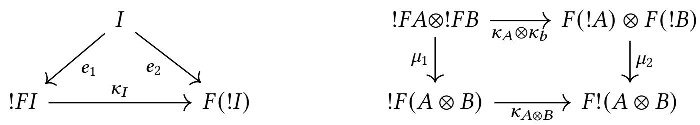
respects the comonad structure:



respects the comonoid structure:



and is monoidal:



In the monoidal diagram above, the monoidal structure (e_1, μ_1) and (e_2, μ_2) are defined using the facts that ! is, by definition a lax monoidal functor, and lax monoidality is stable under composition.

Lemma D.3 ([25]). *Consider* **Set** *as a model of ILL with the trivial identity comonad and Cartesian closed structure, and* **C** *a model of ILL such that it has a linearly distributive functor $F : \mathbf{C} \to \mathbf{Set}$, then the comma category $Gl_F(\mathbf{C})$ is a model fo ILL and the forgetful functor preserves the linear logic structure on the nose.*

We will prove Theorem 6.4 by applying the lemma above, it is only a matter of finding the appropriate functor. When one wants to prove properties about closed programs in a particular type theory, the global sections $\mathbf{C}(I, -)$ is used. Since we are interested in understading the behaviour of open programs where the context only has variables of types $\mathcal{M}\tau_1, \dots, \mathcal{M}\tau_n$, we will consider the functor $\bigtimes_{\tau_1,\dots,\tau_n \in \mathbf{sStoch}} \mathbf{PBanLat_1}(\mathcal{M}\tau_1 \otimes \cdots \otimes \mathcal{M}\tau_n, -)$. Note that when the index of the infinite product is the empty list, it is the global sections functor.

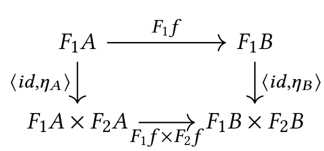
All we have to do now is to prove that it is linearly distributive. We do this by proving the lemma below.

Lemma D.4. *Let $F_1 : \mathbf{C} \to \mathbf{Set}$ be a linearly distributive functor and $F_2 : \mathbf{C} \to \mathbf{Set}$ a functor such that there is a natural transformation $\eta : F_1 \to F_2$. The functor $F = F_1 \times F_2$ is linearly distributive.*

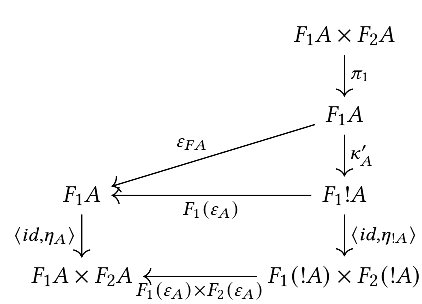
Proof. In order to prove that F is linearly distributive, we must first define its lax monoidal structure and the natural transformation $\kappa : F \to F!$. Let m'_I and $m'_{A,B}$ be the lax monoidal structure of F_1 and κ' its linearly distributive structure. We define:

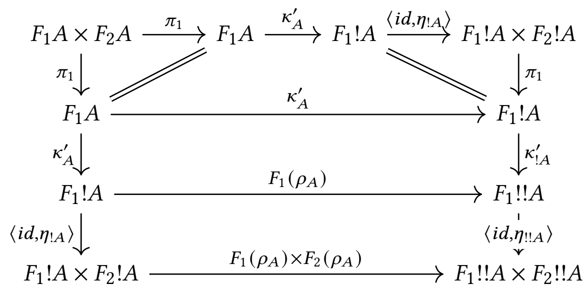
$$m_I = m'_I; \langle id, \eta_I \rangle$$
$$m_{A,B} = (\pi_1 \times \pi_1); m'_{A,B}; \langle id, \eta_{A \otimes B} \rangle$$
$$\kappa_A = \pi_1; \kappa'_A; \langle id, \eta_{!A} \rangle$$

In order to prove that this structure does indeed satisfy the linearly distributivity conditions, note that since we are using the trivial ILL model in **Set**, the comonad diagrams get simpler. Throughout the proof we will repeatedly make use of the following naturality diagram:

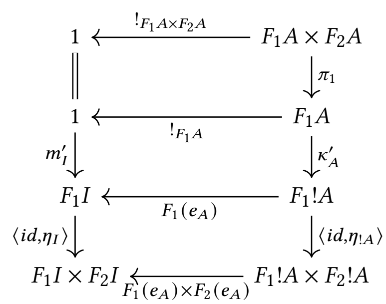


We start from the comonadic diagrams:

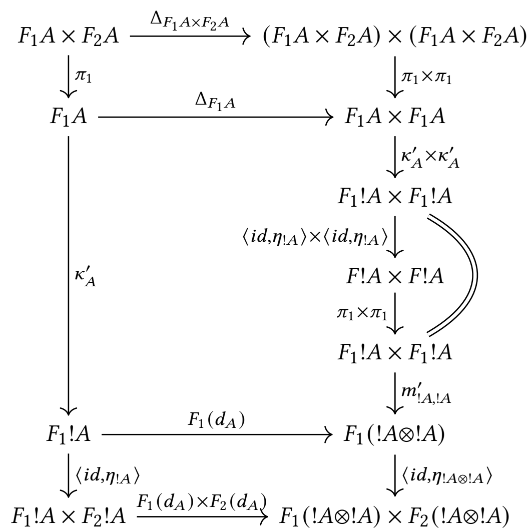




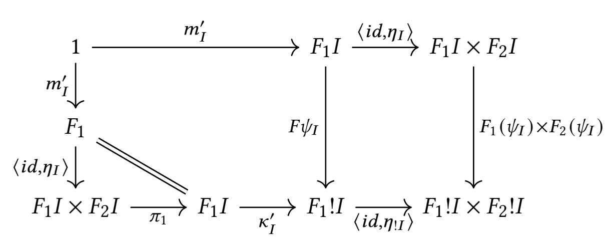
Next, the comonoid diagrams:



Note that the morphism ! above is the unique morphism into the terminal object 1.



Lastly, the monoidal diagrams:



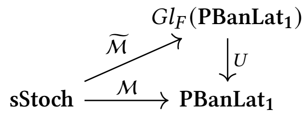
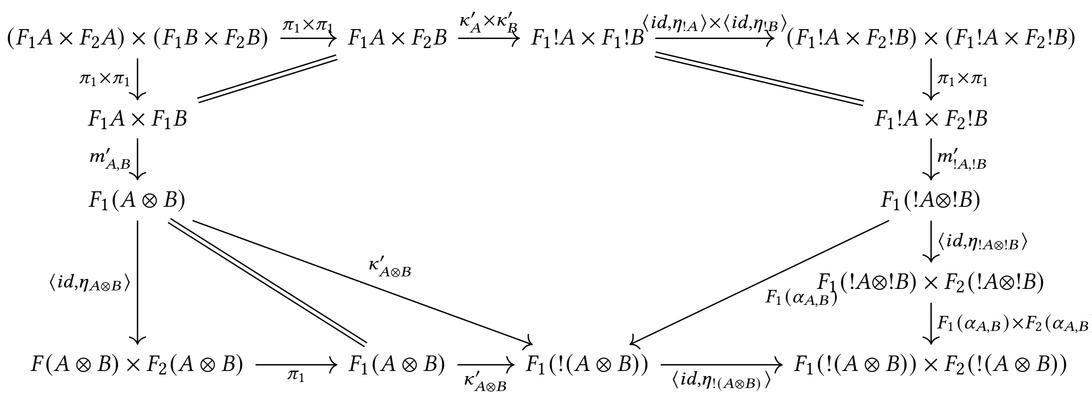


Figure 5: Lifting of the λ_{MK}^{LL} model



□

Corollary D.5. *The functor* $\bigtimes_{\tau_1,\dots,\tau_n \in \mathbf{sStoch}} \mathbf{PBanLat_1}(\mathcal{M}\tau_1 \otimes \cdots \otimes \mathcal{M}\tau_n, -)$ *is linearly distributive.*

Theorem D.6. *The comma category* $Gl_F(\mathbf{PBanLat_1})$ *is an ILL model and the forgetful functor preserves the structure on the nose.*

Proof. This follows directly from the lemmas above. □

This shows that we can use the glued category to interpret our recursive probabilistic calculus, with the exception of the fixed point operation, since in order to it to be a morphism in the glued category, the set X should be equipped with an ωCPO structure. We can achieve this by restricting the glued category to only such X.

Lemma D.7. *The full subcategory of* $Gl_F(\mathbf{PBanLat_1})$ *such that the maps* $(X \to FA)$ *are injections and* ω*CPOs under the inherited order of* F *is an ILL model.*

Proof. It suffices to show that this subcategory is closed under the all of the connectives of ILL, which can de done by unfolding the definitions and noting that, by Theorem 2.28, the positive unit ball of perfect Banach lattices are CPOs. □

Theorem D.8. *The model* $(\mathbf{sStoch}, \mathbf{PBanLat_1}, \mathcal{M})$ *lifts to a model* $(\mathbf{sStoch}, Gl_F(\mathbf{PBanLat_1}), \widetilde{\mathcal{M}})$ *such that the forgetful functor preserves the structure and* $\widetilde{\mathcal{M}}$ *is full and faithful.*

Proof. We start by defining the action of $\widetilde{\mathcal{M}}$ on objects and morphisms,

$$\widetilde{\mathcal{M}}(A) = (\mathcal{M}A, X_A = \{f : \mathcal{M}A_1 \otimes \cdots \otimes \mathcal{M}A_n \to \mathcal{M}A \mid \exists g \in \mathbf{sStoch}(A_1 \times \cdots \times A_n, A), f = \mu^n; \mathcal{M}g\})$$
$$\widetilde{\mathcal{M}}(g) = \mathcal{M}(g)$$

In the defintion above, $\mu^n_{A_1,\dots,A_n} : \mathcal{M}A_1 \otimes \cdots \otimes \mathcal{M}A_n \to \mathcal{M}(A_1 \times \cdots \times A_n)$ is the natural transformation that can be generated using the lax monoidal structure of $\mathcal{M}$. The action on morphisms is well-defined by functoriality of $\mathcal{M}$, since $\mu; \mathcal{M}(g'); \mathcal{M}(g) = \mu; \mathcal{M}(g'; g)$. From functoriality of $\mathcal{M}$, it also follows the functor laws of $\widetilde{\mathcal{M}}$.

The monoidality of $\widetilde{\mathcal{M}}$ follows from proving that the monoidal structure $\mathcal{M}$ lifts to the glued category, which follows from naturality of μ and by definition of the tensor product in the glued category, that ensures that the morphism $\mathcal{M}A_1 \otimes \cdots \otimes \mathcal{M}A_n \to \mathcal{M}B_1 \otimes \mathcal{M}B_2$ can be factored as $g_1 \otimes g_2$. A similar argument can be made for the unit $I \to FI$.

The lifted functor is faithful because so is $\mathcal{M}$, and its fullness can be proved by first considering a morphism $f : \mathcal{M}A \to \mathcal{M}B$ in $Gl_F(\mathbf{C})$ and noting that the identity $\mathcal{M}A \to \mathcal{M}A$ is in X_A, which allows us to conclude, by definition of morphism in the glued category, that $id; f = f$ is in X_B, meaning that it is indeed in the image of $\widetilde{\mathcal{M}}$.

□