# M2 Internship Proposal: Behavioural Distances for Higher-Order Languages with Continuous Probabilities.

Raphaëlle Crubillé
Laboratoire d'Informatique et Systèmes (Marseille)

Probabilistic extensions of lambda-calculus have been thoroughly studied, both from an operational and from a denotational point of view [4, 9, 16, 14, 10, ...]. The most basic way to equip a programming language with probabilistic primitives is to extend it with an operator *choice*(), which generates Boolean at random: this idea leads to the *discrete randomised programming* paradigm, where a program can perform a 0/1 choice at any point during its execution. For some applications—for instance when we want our programs to *model* e.g. a physical phenomenon—we need to add primitives that are able to sample not only from *discrete* probability distributions, but also from *continuous* ones—for instance the uniform distribution on $[0, 1]$, or the Gaussian distribution on $\mathbb{R}$: that's the *continuous randomised programming* paradigm. In this internship, our topic of study will be lambda-calculus extended with continuous probabilistic sampling, as considered for instance in [1, 7].

A central problem in the study of higher-order programming languages has been the question of *program equivalence*: when can we say that two programs, syntactically different, are nonetheless equivalent ? As an illustration, if we consider program transformations made for optimisation purposes, we would like to be able to say and prove that the output program is in some sense equivalent to the input program. A broadly accepted definition is *contextual equivalence*, that was first introduced by Morris [15]: two programs are equivalent when they behave exactly the same no matter in which *context* we put them, and this definition is made self-contained by deciding that contexts can be modelled as programs written in the same language. Formally, it means:

$$M \sim N \qquad \text{when} \qquad \forall C : \text{ context }, \quad \text{Obs}(C[M]) = \text{Obs}(C[N]),$$

where Obs is a notion of *observation* associated to the language—for instance in a language with Booleans as a ground type, we can say that all contexts have Boolean type, and Obs(C[M]) becomes the probability that $C[M]$ returns 0. In a probabilistic setting, however, we would like to be able to say that a program transformations returns a program that is not always equivalent to the input program, but that behaves the same with at least some given probability: this idea led to the concept of *programs distances*, for discrete randomised programming languages [13, 5, 6, 12, 8, ...]. Similarly, quantitative ideas appear in *computational security* [3, 2]: two programs are *computationally indistinguishable* when no admissible adversary is able to distinguish them with more than *negligible probability*. Programs distances have also been investigated in the setting of *differential privacy* [11].

By contrast with the case of equivalences, however, there is no universally accepted notion of *contextual distance*. A natural generalisation of contextual equivalence could be:

$$d(M,N) = \sup_{C \text{ context}} |\mathrm{Obs}(C[M]) - \mathrm{Obs}(C[N])|$$

The definition above, however, does not behave as expected as soon as contexts have *copying abilities*: in particular, if we start with a $\lambda$-calculus $\Lambda$ with copying, and with a type system ensuring termination, a *trivialisation phenomenon* occurs [6]: the distance between two programs in $\Lambda_{choice}$ is always either 0 or 1. For this reason, most notions of programs distance—with few exceptions—are developed for languages with some kind of linear or graded type systems, that ensures that the copying abilities of contexts are limited.

Until now, programs distances have been developed only for *discrete* randomised programming languages; the goal of this internship is to explore what happens in the continuous case, i.e. for $\lambda$-calculi with *continuous* probabilistic sampling, and real numbers as base type. The first challenge is definitional: giving a meaning to $|\mathrm{Obs}(C[M]) - \mathrm{Obs}(C[N])|$ for a context $C[\,] : \mathbb{R}$ implies to choose a distance on the set of probability distributions over $\mathbb{R}$. Several different mathematical notions are available here: total variation distances, Kantorowitch distance..., with no guarantee on how well they behave with respect to computation. In a second stage, we would like to look at whether the trivialisation result mentioned earlier still holds in this setting; for this problem, the deep connection between the trivialisation proof and the law of large numbers could be a starting point.

# References


[1] G. Barthe, R. Crubillé, U. Dal Lago, and F. Gavazzo. On feller continuity and full abstraction. *Proc. ACM Program. Lang.*, 6(ICFP):826–854, 2022.

[2] A. Broadbent and M. Karvonen. Categorical composable cryptography. In *International Conference on Foundations of Software Science and Computation Structures*, pages 161–183. Springer International Publishing Cham, 2022.

[3] A. Cappai and U. Dal Lago. On equivalences, metrics, and polynomial time. In *Proc. of FCT*, pages 311–323, 2015.

[4] R. Crubillé and U. Dal Lago. On probabilistic applicative bisimulation and call-by-value $\lambda$-calculi. In *Proc. of ESOP*, pages 209–228, 2014.

[5] R. Crubillé and U. Dal Lago. Metric reasoning about $\lambda$-terms: The affine case. In *Proc. of LICS*, pages 633–644, 2015.

[6] R. Crubillé and U. Dal Lago. Metric reasoning about $\lambda$-terms: The general case. In *European Symposium on Programming*, pages 341–367. Springer, 2017.

[7] U. Dal Lago and F. Gavazzo. On bisimilarity in lambda calculi with continuous probabilistic choice. In B. König, editor, *Proceedings of the Thirty-Fifth Conference on the Mathematical Foundations of Programming Semantics, MFPS 2019, London, UK, June 4-7, 2019*, volume 347 of *Electronic Notes in Theoretical Computer Science*, pages 121–141. Elsevier, 2019.

[8] U. Dal Lago, N. Hoshino, and P. Pistone. On the lattice of program metrics. In M. Gaboardi and F. van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction, FSCD 2023, July 3-6, 2023, Rome, Italy*, volume 260 of *LIPIcs*, pages 20:1–20:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[9] U. Dal Lago, D. Sangiorgi, and M. Alberti. On coinductive equivalences for higher-order probabilistic functional programs. In *Proc. of POPL*, pages 297–308, 2014.

[10] V. Danos and T. Ehrhard. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Inf. Comput.*, 209(6):966–991, 2011.

[11] A. A. de Amorim, M. Gaboardi, J. Hsu, and S.-y. Katsumata. Probabilistic relational reasoning via metrics. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–19. IEEE, 2019.

[12] T. Ehrhard. Differentials and distances in probabilistic coherence spaces. *Log. Methods Comput. Sci.*, 18(3), 2022.

[13] D. Gebler, K. G. Larsen, and S. Tini. Compositional metric reasoning with probabilistic process calculi. In *Proc. of FoSSaCS*, pages 230–245, 2015.

[14] C. Jones and G. D. Plotkin. A probabilistic powerdomain of evaluations. In *Prof. of LICS*, pages 186–195, 1989.

[15] J. H. Morris Jr. *Lambda-calculus models of programming languages.* PhD thesis, Massachusetts Institute of Technology, 1969.

[16] N. Ramsey and A. Pfeffer. Stochastic lambda calculus and monads of probability distributions. In *Prof. of POPL*, pages 154–165, 2002.