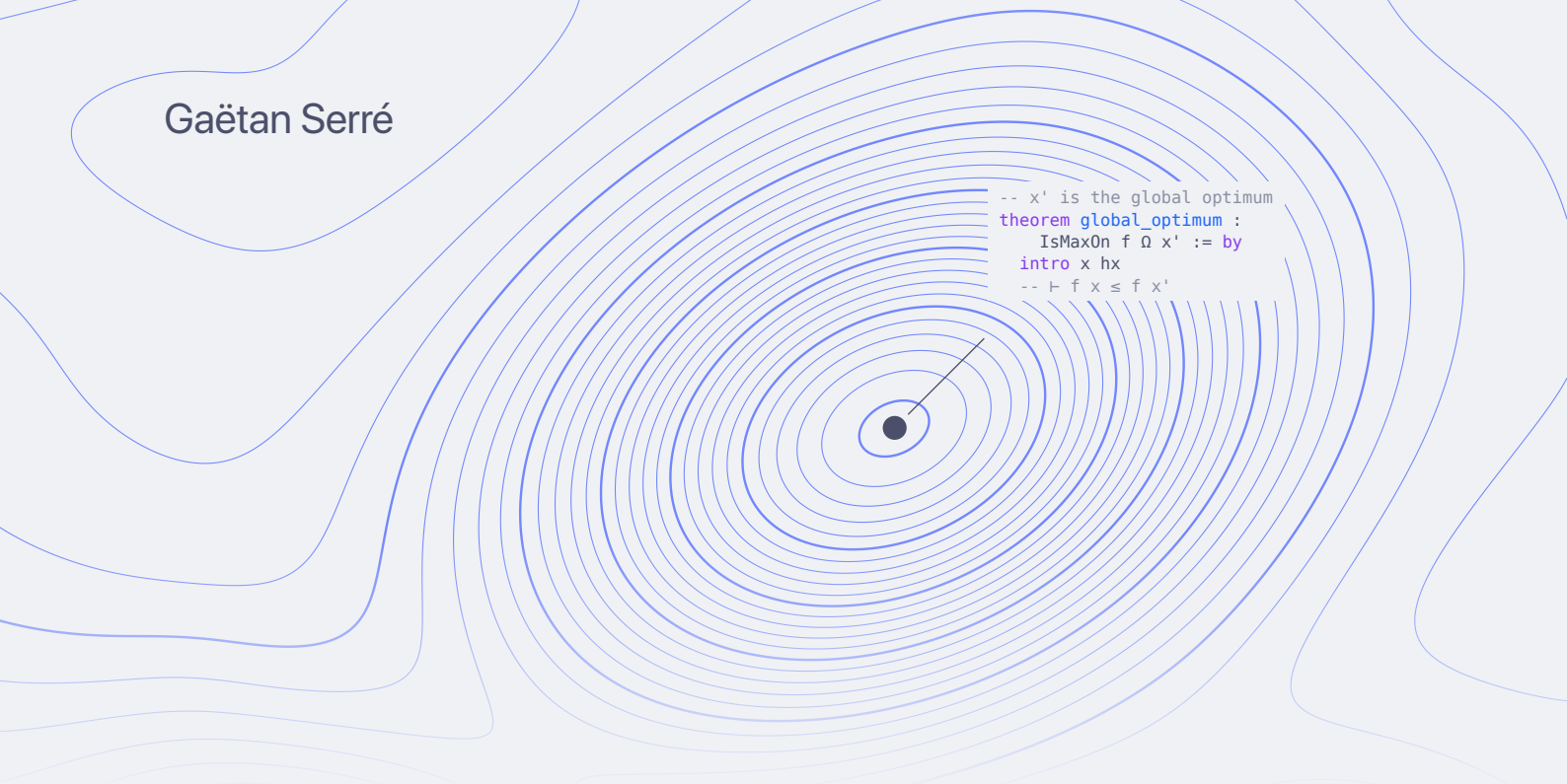
PHD THESIS

Methodological Advances and Formalization in Global Optimization

Gaëtan Serré



Thesis prepared in the research unit Centre Borelli (Université Paris-Saclay, CNRS, ENS Paris-Saclay), under the supervision of **Nicolas Vayatis**, Professor, and the co-supervision of **Argyris Kalogeratos**, Researcher.

Defended at Paris-Saclay on November 16, 2026.

Composition du jury

Membres du jury avec voix délibérative

Alain Trouvé
Président
Professeur des universités,
École Normale Supérieure Paris-Saclay

Gérard Biau
Rapporteur & Examinateur
Professeur,
Sorbonne Université

Massimiliano Pontil
Rapporteur & Examinateur
Professeur,
Istituto Italiano di Tecnologia, University College London

Gabriel Peyré
Examinateur
Directeur de recherche,
École normale supérieure

Sylvie Boldo
Examinatrice
Directrice de recherche,
Inria Saclay

Anna Korba
Examinatrice
Professeure,
ENSAE

Remerciements

Cette thèse n'aurait pas pu être réalisée sans le soutien de nombreuses personnes. Je tiens à exprimer ma profonde gratitude à tous ceux qui m'ont accompagné tout au long de ce parcours.

Aux rapporteurs. Je tiens à remercier mes rapporteurs, Gérard Biau et Massimiliano Pontil, pour le temps et l'attention qu'ils ont consacrés à l'évaluation de mon travail de thèse. Leurs commentaires élogieux et leur intérêt pour mes recherches m'ont profondément honoré et m'ont donné une confiance nécessaire pour traverser la période assez stressante de la préparation de la soutenance.

Au jury. Je remercie également les autres membres du jury : Sylvie Boldo, Rémy Degenne, Anna Korba, Gabriel Peyré et Alain Trouvé, pour avoir accepté de participer à l'évaluation de mon travail et pour leurs remarques constructives et enrichissantes. Je tiens à remercier particulièrement Rémy Degenne pour son implication dans mon aventure LEAN, ce qui m'a permis de découvrir un domaine de recherche passionnant, et je suis très heureux et impatient de continuer à travailler avec lui dans le cadre de mon post-doctorat.

À mes directeurs. Je remercie énormément mes directeurs de thèse, Nicolas Vayatis et Argyris Kalogeratos. Merci, Nicolas, de m'avoir permis de réaliser ce projet, et merci pour la confiance que tu m'as accordée en me laissant une grande liberté dans mes choix de recherche. Merci beaucoup pour les meetings qu'on a eus : tes conseils, peu importe le sujet, sont toujours d'une extrême pertinence et efficacité. Merci, Argyris, pour ton implication et ta disponibilité, et d'avoir su me guider dans mes choix, malgré la diversité des sujets que j'ai abordés. Merci aussi pour toutes les discussions informelles que nous avons eues, elles m'ont beaucoup aidé à mieux comprendre le monde de la recherche.

Au Centre Borelli. Merci beaucoup à tous les membres du Centre Borelli de m'avoir permis de travailler dans un environnement aussi stimulant et agréable. Merci à tous les organisateurs d'événements sociaux, qui resteront gravés dans ma mémoire (particulièrement les karaokés de Laurent Oudre). Merci aux personnels administratifs pour leur efficacité, leur gentillesse et leur humour. Un merci tout particulier à Gwladys Stouvenel, qui, depuis mon année au MVA, a su égayer mes journées par sa bonne humeur. Un grand merci également à toi, Samuel, pour ton entrain et toute l'expérience que tu m'as transmise, en particulier en 2^{e} année de thèse. C'était un plaisir de partager un bureau avec toi et de discuter (et rigoler) de tout et de rien.

À mes amis. Chloé, Paul et Émilien, merci pour votre présence tout au long de ces années. Je suis très admiratif de votre évolution et j'espère qu'on gardera nos liens d'amitié encore longtemps.

- Émilien, ta convivialité, ta générosité, ton humour, ta passion et ton hospitalité font de toi une personne très importante dans ma vie. Merci de m'avoir fait apprécier Massy-Palaiseau à travers toutes les soirées raclette au fromage suisse

et des gins de qualité. Je serai éternellement reconnaissant pour toute l'aide que tu m'as apportée, en particulier pour les problèmes de Linux et de compilation, sans quoi je n'aurais jamais pu valider mon magistère ! Je suis très admiratif de ce que tu es devenu et fier de l'évolution de notre amitié. Merci pour tous les moments passés ensemble et merci par avance pour tous ceux à venir.

- Paul, merci pour ces huit années d'amitié. Je ne saurais énumérer tous les souvenirs que nous avons partagés, des bancs des premiers amphis du 333 jusqu'aux OUPS (quel parcours !) en passant par les soirées pizza maison, qui, petit à petit, sont devenues un rituel ouvert à d'autres. Merci pour les games de LoL (même si je sais que ça t'a bouffé une partie de l'âme pendant un temps). Ta passion, ta bonne humeur à toute épreuve, ta culture, ta sociabilité et ta curiosité ont profondément influencé la version adulte de moi-même (en particulier à travers les discussions sur tous les sujets possibles que l'on a eues au fil des années). Tu es une vraie source d'inspiration et je suis très heureux de t'avoir comme ami. Ton évolution est impressionnante et j'ai très hâte de voir quelle personne tu deviendras.
- Chloé, le mélange de ta douceur, ta bienveillance, ton calme, ton ouverture d'esprit et ton humour fait de toi une personne exceptionnelle. Ta volonté de passer des moments de qualité avec tes amis est admirable et a donné lieu à de nombreux souvenirs, tels que les séjours en Auvergne, en Normandie, ou les soirées jeux d'enquête (ou celles du jeu « Dig Out », qui étaient exceptionnelles). Je pense particulièrement aux événements annuels célébrant (ou déplorant, en fonction de l'avancement de la thèse) la recherche publique, moments si particuliers qu'il est difficile d'en parler à d'autres personnes sans paraître bizarre. Merci pour tes recommandations cinématographiques et musicales, et de prendre en compte les miennes (les gens sont souvent plus enclins à uniquement recommander !). Je suis fier de ton évolution et de pouvoir te compter parmi mes amis. Un grand merci également pour les illustrations des parties de ma thèse, qui ajoutent une touche personnelle et artistique à mon travail.

Merci à Gwendal, Perceval, Gaspard et Éloi, valeureux collègues du bureau des délices[1]. De stagiaires à docteurs (ou futurs docteurs à l'heure où j'écris ces lignes), nous avons partagé de nombreux moments : de frustration, de stress intense, de travail acharné, mais aussi (parfois) de joie et de fous rires. Surtout, nous avons créé un grand nombre de références (je pense notamment à « `tii aah` » et aux « twenty-four H », merci, Perceval) qui sont maintenant des liens indestructibles entre nous. Merci pour votre amitié et votre soutien. La thèse est une aventure qui peut s'avérer solitaire, mais grâce à vous, j'ai toujours été entouré de personnes avec qui partager les bons et les mauvais moments. J'espère que nous continuerons à nous voir après la thèse, et que nous pourrons créer de nouveaux souvenirs ensemble. Merci tout particulièrement à toi, Perceval, pour ton aide pendant le MVA : clairement, sans toi, je n'en serais pas là aujourd'hui.

[1] Salle 3S28

Gwendal a droit à son paragraphe personnel en récompense de son effort de venir jusqu'aux contrées de Gif-sur-Yvette quasiment tous les jours pendant plus de trois ans (les autres, je vous aime quand même).

- Gwendal, tu es la personne avec qui j'ai passé le plus de temps pendant ces trois années. On a vécu plein de moments de vie, des plus simples, comme se demander à chaque fois ce qu'on va manger à midi, aux plus marquants, comme des voyages à l'autre bout du monde. Celui de l'été 2026 est d'ailleurs un de mes meilleurs souvenirs. Au-delà de tout ce qu'on a vu (et surtout mangé), j'ai l'impression que cela nous a permis de passer un cap dans notre amitié, la rendant insensible au temps et à la distance. Merci pour ta gentillesse, ton humour et tes références, de m'avoir fait découvrir Gravity Falls (moments inoubliables de la 2^{e} année de thèse) et, plus généralement, de m'avoir laissé entrer dans ta vie. Le grand nombre de tes passions et ta culture, si diversifiées, font de toi quelqu'un qu'on a envie d'avoir près de soi. Tu as laissé une empreinte indélébile dans mon âme. Merci pour tout.

Merci à Arthur, Willy, Gian, Julien, Caroline, Julie, Philomène, Sana, Juliette, Paul, Antoine et Thaddeus pour leur présence tout au long de ces années et pour tous les différents moments que nous avons partagés. Merci à toutes les personnes que je n'ai pas citées mais qui ont contribué à rendre ces années de thèse plus agréables et enrichissantes.

À ma famille. Je remercie ma sœur Julie et mon frère Romain pour leur soutien, leur curiosité et leur compréhension de ma situation. J'ai tellement de chance d'avoir grandi à vos côtés ! Le temps est passé (les souvenirs sur Just Dance 4 sont loin maintenant), mais je suis très heureux de la relation qu'on a aujourd'hui, malgré la distance. Je vous aime fort.

Enfin, merci à mes parents pour leur soutien indéfectible depuis que je suis parti de la maison, il y a plus de huit ans maintenant. Merci pour votre confiance, votre aide, votre compréhension et votre amour. Je vous suis très reconnaissant, car je n'aurais jamais pu arriver à ce stade sans vous. Merci, Papa, de m'avoir transmis ton goût pour les sciences, la rigueur et ta curiosité. Ce n'est peut-être pas conscient, mais je suis persuadé que ton parcours m'a beaucoup influencé dans mon choix de carrière, et je suis fier d'être passé par la même école que toi (malgré un changement de nom). Merci, Maman, de m'avoir transmis ta sensibilité et ta sociabilité, qui m'ont permis de m'ouvrir aux autres et de créer des liens d'amitié solides. Si j'ai pu être aussi bien entouré pendant toutes ces années, c'est parce que j'ai reçu une bonne éducation ! Merci à vous deux de m'avoir transmis toutes ces valeurs : une partie de vous vit en moi. Merci pour tout, je vous aime énormément.

Introduction

The problem of identifying a global optimum of an unknown real-valued function defined over an arbitrary finite-dimensional Euclidean space arises naturally across a broad spectrum of scientific disciplines and occupies a central position in numerous practical applications. This class of problems is variously referred to in the literature as *global optimization*, *black-box optimization*, or *non-convex optimization*. The diversity of these designations is not merely terminological: it reflects a fundamental structural characteristic of the problem, namely that the objective function is entirely unknown to the practitioner and, as a consequence, may exhibit non-convexity, so that a critical point does not necessarily correspond to a global optimum. Since the global optimization framework provides a principled methodology for identifying the best input parameters of an arbitrary system with respect to a given performance criterion, it finds application in a remarkably wide range of fields, including machine learning, epidemiological simulation via agent-based models, physical simulations, optimal control, and many other domains (Awasthi et al., 2024; El Yadari et al., 2024; Franceschi et al., 2024; Houssein et al., 2024; Lee et al., 2017; Pintér, 1991).

Despite its ubiquity and practical relevance, global optimization remains a fundamentally challenging problem, primarily due to the absence of any exploitable structural information about the objective function. The only information accessible to the practitioner consists in pointwise evaluations of the function, a severe restriction that renders classical deterministic optimization methods either ineffective or computationally intractable, and thereby motivates the development of algorithms that rely exclusively on such evaluations to guide the search for a global optimum. An additional layer of difficulty arises from the fact that evaluating the objective function may itself be computationally expensive — as is the case, for instance, when the function represents the output of a large-scale simulation or the result of a costly forward pass through a large neural network. This practical constraint necessitates the design of frugal algorithms capable of producing high-quality solutions within a strictly limited evaluation budget. Global optimization methods are thus particularly well-suited to settings in which the objective function either admits no closed-form expression — for example, because it is the output of a complex black-box simulation — or in which no structural information is available that could be leveraged by classical methods, such as convexity or smoothness. An illustrative example of such a function, exhibiting no a priori exploitable structure, is depicted in Figure 1.

The *search space* — that is, the domain over which the optimization is carried out — largely determines the nature of the global optimization problem under consideration. When this domain is finite or countably infinite, the problem falls within the scope of *combinatorial* (or *discrete*) *global optimization*. Conversely, when the search space is uncountable, the problem is referred to as *continuous global optimization*. Yet this topological distinction, while fundamental, does not fully capture the conceptual gulf between the two paradigms. The deeper difference

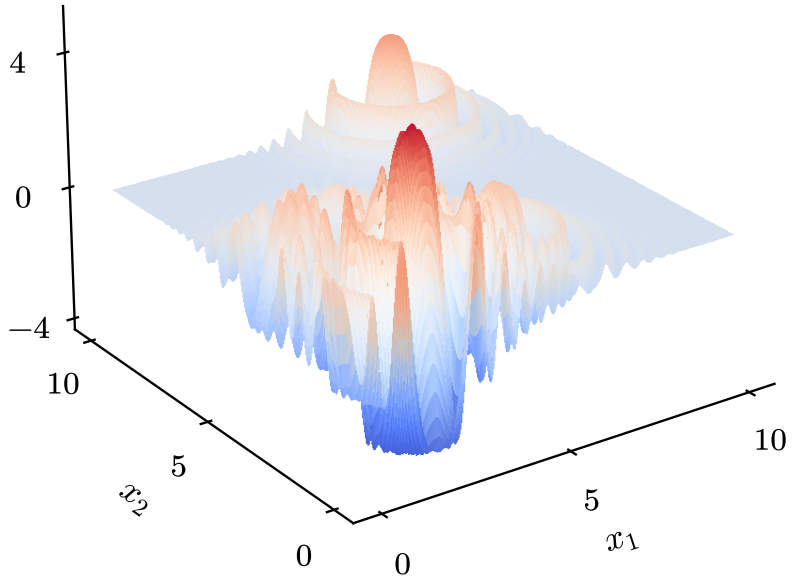


Figure 1. *The two dimensional Langermann function.*

lies in the notion of regularity that can be assumed — or exploited — about the objective function.

In continuous global optimization, the search space is endowed with a metric structure that is meaningful with respect to the objective: points that are close in the domain are expected, under standard regularity assumptions such as continuity, to yield similar function values. This regularity is not merely a convenient analytical assumption; it is the very foundation upon which surrogate-based methods or gradient approximations are built. Evaluations are not exchangeable — their locations carry information, and a model fitted to a set of observations would yield an entirely different prediction were those observations reassigned to arbitrary points in the domain.

In discrete global optimization, by contrast, no such structural assumption is warranted a priori. Two feasible solutions that are adjacent in some combinatorial sense need not produce similar objective values, and any notion of proximity is itself problem-dependent, typically hand-crafted rather than intrinsic to the domain. When no regularity can be assumed or inferred, the set of known evaluations is, in a precise sense, permutation-invariant: reassigning observed function values to visited solutions arbitrarily would leave the algorithm's behaviour unchanged. This invariance is a restatement of the No Free Lunch principle (Wolpert & Macready, 2002): averaged over all possible objective functions, no search strategy can outperform blind enumeration. In practice, however, discrete optimization problems of interest are rarely structureless. Algorithms such as simulated annealing rely on a problem-specific neighbourhood structure that induces a form of local regularity — an assumption that, when valid, allows them to outperform random search. The contrast with continuous global optimization is thus not merely one of

domain cardinality, but of the nature and origin of the regularity being exploited: intrinsic and metric in the continuous case, combinatorial and problem-specific in the discrete one.

The present thesis is devoted exclusively to the continuous setting, and we adopt the term *global optimization* as a shorthand for *continuous global optimization* throughout. We further assume that the search space is compact, an assumption that is both natural in many applied contexts and mathematically convenient, as it guarantees — by virtue of the extreme value theorem — that any continuous function defined over this domain attains at least one global optimum.

Formally, the global optimization problem may be stated as follows: given a compact domain $\Omega \subseteq \mathbb{R}^d$ and a continuous function $V : \Omega \to \mathbb{R}$, the goal consists in finding a point $x^\star$ such that

$$x^\star \in \operatorname*{argopt}_{x \in \Omega} V(x),$$

where "argopt" denotes either "argmax" or "argmin" depending on the nature of the optimization problem considered. To address this problem, standard sequential optimization procedures begin by evaluating the objective function at an initial point $X_0 \in \Omega$, typically drawn at random. Then, at each iteration n, they draw a new query point $X_{n+1} \in \Omega$ from a distribution that is a measurable function of the history $(X_0, ..., X_n, V(X_0), ..., V(X_n))$. This function does not depend on V, which is only accessed through its evaluations.

The apparent simplicity of this formulation belies the intrinsic difficulty of the problem, which has stimulated a sustained research effort over several decades (Kirkpatrick et al., 1983; Price, 1983) and continues to give rise to numerous contributions to this day, notably in the context of particle-based methods (Bungert et al., 2025; Fornasier et al., 2024; Gerber et al., 2025; Pinnau et al., 2017; Riedl, 2024). While no structural assumption on the objective function is available a priori, mild regularity hypotheses such as Lipschitz continuity can nonetheless be introduced to guide the design of principled optimization strategies.

A fundamental and elegant connection exists between global optimization and probabilistic sampling, mediated by the Gibbs measure. Given the objective function $V : \Omega \to \mathbb{R}$ and an inverse temperature parameter $\beta > 0$, the associated Gibbs measure — also known as the Boltzmann distribution — μ_β on Ω is defined by its density

$$\mathrm{d}\mu_\beta(x) = \frac{\exp(-\beta V(x))}{Z_\beta}\,\mathrm{d}x, \quad Z_\beta = \int_\Omega \exp(-\beta V(x))\,\mathrm{d}x,$$

where Z_β is the partition function serving as a normalizing constant. This distribution assigns higher probability mass to regions where V takes smaller values, with the degree of concentration governed by the temperature parameter $1/\beta$. A classical result establishes that, in the zero-temperature limit $\beta \to +\infty$, the Gibbs measure concentrates exclusively around the global minima of V: more precisely, μ_β converges towards a distribution supported on the set of global optima (Hwang, 1980). This observation reveals that global optimization can be reformulated as a sampling

problem: finding a global optimum of V is equivalent to drawing a sample from μ_β at sufficiently low temperature. This duality is not merely of theoretical interest — it underlies many practical algorithms, from simulated annealing (Kirkpatrick et al., 1983), which simulates a Markov chain whose stationary distribution is μ_β with a gradually decreasing temperature, to more recent particle-based methods and McKean-Vlasov dynamics that seek to approximate μ_β through interacting particle systems. This perspective thus provides a unifying probabilistic lens through which a broad class of global optimization algorithms can be understood and analyzed.

Notwithstanding the extensive body of theoretical analyses available in the literature, global optimization algorithms have largely been studied in isolation, despite the fact that they sometimes share deep structural similarities. The first part of this thesis is devoted to addressing the following question:

What common structural principles underlie existing global optimization algorithms, and to what extent can these principles be systematically leveraged to design new algorithms and modular components applicable across a broad family of methods?

We address this question through methodological advances within two prominent families of global optimization algorithms: *decision-based* and *McKean-Vlasov-based* approaches. For each family, we provide a unified treatment of existing algorithms, exposing their shared structural foundations, and we propose both new algorithms and reusable modular components that can be deployed across the family. These contributions are accompanied by a software library containing implementations of these algorithms and modules, which exploits the family structure to expose a unified interface and to facilitate the development of new algorithms by composition.

A natural continuation of this family-based structuration of global optimization algorithms is to identify the simplest possible abstraction that faithfully captures the common principles underlying both existing and future algorithms. The motivations for seeking such an abstraction are twofold. On the one hand, it yields a unified mathematical view of global optimization algorithms, providing a rigorous formulation of an object that solves global optimization problems. This, in turn, enables the statement of results of a meta-algorithmic nature — that is, results that require reasoning simultaneously over the entire class of global optimization algorithms — in a mathematically well-founded manner. On the other hand, provided that the abstraction is both sufficiently simple and sufficiently expressive, it can serve as a foundation for the mechanized formalization of global optimization algorithms in proof assistants.

Proof assistants are software tools that enable the formalization of mathematical statements and their proofs in a fully rigorous, machine-checkable manner. They are based on extensions of the Calculus of Constructions (Coquand & Huet, 1988), in particular the Calculus of Inductive Constructions (CIC) (Paulin-Mohring, 2015), in which propositions are interpreted as types and proofs as terms. A proof assistant can be viewed as an interface through which users express formal statements and derivations, which are then checked by a small trusted core component known as the kernel. The kernel implements a minimal set of typing and reduction rules derived

from the underlying type theory, and is responsible for ensuring the correctness of all accepted proofs. Figure 2 illustrates the general architecture of a proof assistant. Since the validity of any formal derivation ultimately reduces to kernel-level type-checking — a process entirely independent of human judgment — the level of confidence afforded by a proof assistant substantially exceeds that of conventional peer review.

The benefits of formalizing global optimization algorithms within such a framework are manifold. First and foremost, it enables the formal verification of the correctness of algorithms and their convergence properties, which is of particular importance in safety-critical applications where erroneous algorithmic guarantees could have severe consequences. Even within academic research, the proofs accompanying published results in applied mathematics are frequently relegated to supplementary materials that are not subject to the same scrutiny as the main body of the paper, a practice that can give rise to undetected errors with potentially far-reaching consequences when the algorithms in question are deployed in critical settings. Beyond verification, the formalization of global optimization algorithms in proof assistants also facilitates the development of new, theoretically grounded algorithms: mathematical statements and definitions formalized in a proof assistant constitute first-class objects that can be reused, combined, and extended in a modular yet rigorous fashion, closely mirroring the way new algorithms in the field are constructed by recombining and building upon existing algorithmic components from the literature.

Finally, the recent and rapid progress of Large Language Models (LLMs) and their demonstrated effectiveness in addressing mathematical reasoning tasks (Alon et al., 2026; OpenAI, 2026; Sothanaphan, 2026) has brought proof assistants to the forefront as a tool for evaluating the correctness of LLM-generated mathematical content. The standard pipeline involves a first LLM producing an informal proof, followed by a second LLM specialized in translating informal mathematical content into its formal counterpart, which then attempts to produce a formal proof that can be mechanically verified. As proof assistants provide a high level of assurance, a proof validated by such a system can be considered a correct proof of the corresponding formal statement. This pipeline, however, introduces a critical bottleneck: while the correctness of the formal proof itself is delegated to the proof assistant, the faithfulness of the translation from informal to formal — that is, whether the formalized statement accurately captures the intended mathematical content — cannot be automatically guaranteed. Verifying this faithfulness requires expert human oversight, which constitutes a time-consuming and costly process, all the more so as the throughput of modern LLMs substantially exceeds the productivity of human domain experts. A natural strategy to alleviate this bottleneck is to equip LLMs with carefully curated, human-verified libraries of formalized definitions and results, which can serve as anchors for the translation process and reduce the surface area of the translation that requires manual validation.

This consideration constitutes the primary motivation for the abstraction of global optimization algorithms developed in the second part of this thesis, which aims to address the following question:

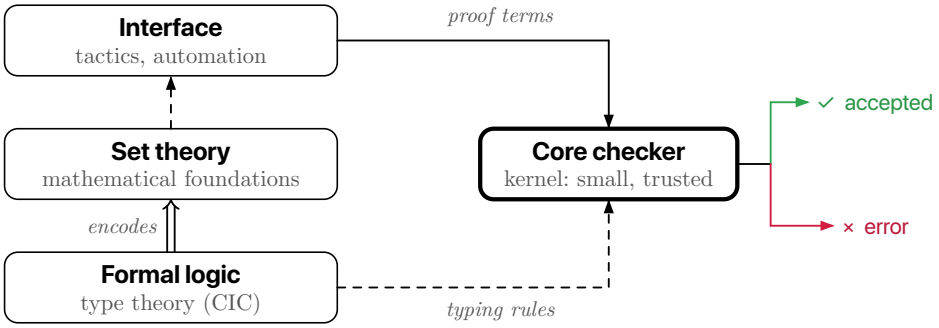


Figure 2. *Architecture of a proof assistant. The core checker (or kernel) is a small, trusted component that verifies the correctness of proofs based on the rules of the underlying type theory. The interface provides a user-friendly environment for writing and checking proofs, often with support for tactics and automation.*

What is the simplest yet sufficiently expressive abstraction that faithfully encapsulates any existing or conceivable global optimization algorithm, and how can such an abstraction serve as a foundation for building a verified library of formalized algorithms in a proof assistant?

Some existing optimization methods

Throughout this manuscript, several global optimization methods are introduced, some of which are examined in detail in dedicated chapters. This section presents a range of relevant approaches within the scope of this thesis, including methods that will be revisited and studied more thoroughly in subsequent sections.

Pure Random Search. The simplest conceivable strategy for global optimization is *Pure Random Search* (PRS), which amounts to drawing candidate points uniformly at random over the search space and retaining the best observed evaluation. Despite its conceptual simplicity, PRS is notoriously inefficient in high-dimensional settings, as it makes no use of previously collected information to guide the sampling process. Its value lies primarily in its role as a reference baseline, and it is employed as such in several experiments throughout this thesis.

Simulated Annealing. Introduced by (Kirkpatrick et al., 1983), *Simulated Annealing* ranks among the most widely adopted methods in global optimization. The algorithm operates by iteratively proposing candidate solutions whose acceptance is governed by a probability depending jointly on the change in objective value and a time-varying temperature parameter. As this temperature is progressively reduced, the method transitions from broad exploratory behavior toward local exploitation, in the spirit of the physical annealing process it emulates. Its theoretical properties have been studied in depth — see, for instance, Trouvé (1996) — and its original combinatorial formulation has been extended to continuous domains. Of particular conceptual interest is the connection between Simulated Annealing and Markov Chain Monte Carlo methods: the algorithm can be interpreted as sampling from a Gibbs distribution, thereby being one of the earliest methods witnessing the deep interplay between global optimization and probabilistic sampling.

Partitioning strategies. DIRECT (Jones, 2001), Deterministic Optimistic Optimization (DOO) (Munos, 2011), and Simultaneous Optimistic Optimization (SOO) (Munos, 2011) constitute a family of methods that adaptively partition the search space into successively finer regions, evaluating the objective at representative points within each cell. These approaches perform particularly well in low-dimensional settings, where the hierarchical decomposition of the domain enables efficient identification of promising regions. Several extensions of SOO have since been proposed, including a stochastic variant (Valko et al., 2013) and a multi-objective generalization (Al-Dujaili & Suresh, 2018).

Multistart strategies. Methods such as Multistart Gradient Descent (MSGD), Multi-start Langevin (LANGEVIN), and Multi-Level Single-Linkage (MLSL) (Rinnooy Kan et al., 1985) address global optimization by launching a local procedure from multiple, independently chosen initial points. Beyond their natural amenability to parallelization, these methods can incorporate mechanisms to avoid redundant exploration of already-visited regions of the search space (Rinnooy Kan et al., 1985).

Hit-and-run strategies. Proposed independently by (Boneh & Golan, 1979) and (Smith, 1984), hit-and-run algorithms proceed by sampling a direction uniformly from the unit sphere at each iteration and performing a line search along that direction from the current point, accepting the move only when it yields an improvement in the objective value.

Evolutionary strategies. This broad family encompasses the CMA-ES algorithm (Hansen & Ostermeier, 2001), discussed in detail in Section A.1.1, the CRS algorithm (Price, 1983), and genetic algorithms (Vie et al., 2020; Zhan et al., 2022). Rooted in the principles of natural selection, these methods maintain a population of candidate solutions that evolves through operations of selection, recombination, and mutation. Their ability to sustain diverse exploratory behavior across the search space makes them particularly suited to high-dimensional and multimodal problems. Many of these algorithms can be encompassed under the broader framework of Estimation of Distribution Algorithms (Etxeberria, 1999; Larranaga & Bielza, 2025; Lozano, 2006; Mühlenbein & Mahnig, 1999; Mühlenbein & Paass, 1996; Pelikan et al., 1999), which are examined at greater length in Chapter A.1.

Expected improvements strategies. Bayesian Optimization (BAYESOPT) (Brochu et al., 2010; Jones et al., 1998; Močkus, 1974), treated in detail in Section A.1.2, is a sequential model-based approach in which a probabilistic surrogate of the objective function is iteratively updated and used to select evaluation points by maximizing an acquisition function — typically the expected improvement over the current best known value. This methodology is especially effective for expensive black-box objectives, though it does not scale gracefully to high-dimensional settings or large evaluation budgets, owing to the computational cost of fitting and querying the surrogate model. BAYESOPT can itself be viewed as a form of Estimation of Distribution Algorithm, in the sense that it maintains and updates a probabilistic model over the search space to guide the optimization process.

Decision-based strategies. Decision-based approaches constitute one of the central contributions examined in this thesis. Their defining principle is to determine

whether to evaluate the objective at a given point solely on the basis of information accumulated from prior evaluations, without constructing an explicit surrogate model. This principle generalizes the strict improvement condition underlying *Pure Adaptive Search* (PAS) (Zabinsky, 2003; Zabinsky & Smith, 1992) into a fuzzy decision rule, allowing candidate points to be accepted even without improving the current best value, in the spirit of algorithms such as Simulated Annealing (Kirkpatrick et al., 1983). It can also be seen as a discrete, sequential counterpart to *adaptive importance sampling* (Cappé et al., 2008; Rubinstein, 1999): rather than reweighting an explicit proposal density, a decision rule directly filters candidate points drawn from a fixed reference distribution, implicitly biasing the search toward promising regions of the search space without ever representing or updating such a density. The term *decision-based* is chosen to emphasize this filtering mechanism: at every iteration, the algorithm faces a binary decision — to evaluate or not to evaluate a given candidate — and it is precisely the rule governing this decision, rather than any model of the objective itself, that drives the search. This terminology also distinguishes these approaches from *model-based* or *surrogate-based* strategies, such as Bayesian optimization, where the decision of where to evaluate next is mediated by an explicit probabilistic model of the objective; here, the decision rule operates directly on the history of evaluations, making the decision process itself the primary object of design. This paradigm underlies several algorithms discussed in detail in Chapter A.2, including LIPO and ADALIPO (Malherbe & Vayatis, 2017a), LIPO+ and ADALIPO+ (Serré et al., 2024), RANKOPT and ADARANKOPT (Malherbe et al., 2016; Malherbe et al., 2026), and ECP (Fourati et al., 2025).

Particles-based strategies. This class of methods maintains a population of interacting particles whose collective dynamics drive the exploration of the search space. The paradigm was popularized by Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), which gave its name to the broader family. A substantial subfamily is bio-inspired, including heuristic methods such as the Grey Wolf Optimizer (Mirjalili et al., 2014) and the Whale Optimization Algorithm (Mirjalili & Lewis, 2016), which, while empirically effective in certain settings, generally lack rigorous theoretical underpinnings. A contrasting and theoretically grounded subfamily grounds the particle dynamics in systems of — possibly stochastic– differential equations of McKean–Vlasov type, which enables the analysis of the evolution of the particle distribution over time. This second subfamily is among the principal subjects of investigation in this thesis and is examined in Chapter A.3. It includes Consensus-Based Optimization (CBO) (Pinnau et al., 2017), a rapidly developing area of current research (Bungert et al., 2025; Fornasier et al., 2024; Gerber et al., 2025; Riedl, 2024) discussed in Section A.3.1, as well as Stein Boltzmann Sampling (SBS) (Serré et al., 2025), an original contribution of this thesis presented in Section A.3.2. Note that LANGEVIN and MSGD, introduced above in the context of multistart methods, can likewise be interpreted as degenerate McKean–Vlasov dynamics in the absence of particle interaction; they serve as baselines in several experiments.

Global optimization is a vast and rapidly evolving field, and many relevant algorithms necessarily fall outside the scope of this manuscript. Among those not

discussed here is GLOPTIKERNEL (Rudi et al., 2025), which reformulates global optimization as a convex program with infinitely many constraints and solves it via a principled subsampling of those constraints. While numerous other algorithms could plausibly be situated within the families considered in this thesis, the methods surveyed here are deemed sufficiently representative to illuminate the core principles underlying each of these families. A complementary perspective on the historical development of the field is provided by the timeline of major innovations in global optimization presented in Figure 3, which, while not exhaustive, offers a structured overview of the landscape.

Contributions

The contributions of this thesis are twofold. The first part provides methodological advances within two families of global optimization algorithms: decision-based approaches and McKean-Vlasov-based approaches. For each family, we develop a unified theoretical view of existing algorithms, introduce new algorithms and modular components applicable across the family, and release an open-source library (GLOBE) that exploits this family structure to offer a unified interface and streamline the development of novel algorithms. The second part introduces a general mathematical abstraction for global optimization algorithms, capable of encapsulating any existing or future algorithm of this class. This abstraction enables rigorous reasoning about meta-algorithmic statements quantifying over *all* global optimization algorithms, and lays the groundwork for the mechanized formalization of such algorithms in proof assistants — a contribution of particular relevance in the context of the recent and rapid progress of Large Language Models (LLMs) in mathematical problem solving.

Part A

Decision-based approaches

In Chapter A.2, we develop a unified framework for decision-based optimization algorithms, a family of methods parameterized by a sequence of decision rules $D_n : \Omega^{n+1} \times \mathbb{R}^{n+1} \times \Omega \to [0, 1]$. At each iteration n, the decision rule D_n maps the full history of function evaluations to a probability distribution over the search space Ω, which is then used to select the next query point. We demonstrate that this formulation encompasses a broad range of existing algorithms, including LIPO and ADALIPO (Malherbe & Vayatis, 2017a), as well as RANKOPT and ADARANKOPT (Malherbe et al., 2016; Malherbe et al., 2026).

Within this framework, we propose extensions to the LIPO and ADALIPO algorithms — methods designed for the maximization of Lipschitz functions — that equip them with an adaptive stopping criterion based on the ratio between the measure of the search space induced by the decision rule and its Lebesgue measure. This criterion terminates the search when the ratio falls below a prescribed threshold, yielding faster execution times without sacrificing approximation quality. In addition, our extension introduces an improved exploration-exploitation trade-off, allowing the decision rule to adopt a more explorative behavior in early iterations

and gradually shift toward exploitation as the optimization progresses. Although developed in the context of LIPO and ADALIPO, these modifications are in fact applicable to any decision-based algorithm, which illustrates the practical benefit of maintaining a unified view of the family.

We then provide a thorough treatment of the RANKOPT and ADARANKOPT algorithms. Unlike smoothness-based methods, these algorithms do not require Lipschitz regularity; instead, they exploit the induced ranking structure of the objective function, that is, the nested inclusion of its sublevel sets. Although these algorithms were originally introduced in Malherbe et al. (2016), their full algorithmic description and efficient implementation remained incomplete in the literature. Our contributions are fourfold: we provide a corrected statement and proof of a key result motivating the construction of the algorithms; we give a self-contained description of the algorithms, highlighting the auxiliary techniques required for an efficient implementation; we develop a C++/Python implementation; and we reproduce the original numerical experiments using this implementation.

Finally, we present BOBYQA (Powell, 2009) — a trust-region method — as a modular component compatible with any decision-based algorithm. This component addresses a structural limitation shared by the entire decision-based family: these algorithms tend to undersample regions of low probability mass, making it difficult to locate global optima situated at the extremum of a sharp peak. The trust-region component overcomes this limitation by fitting a local surrogate model of the objective in a neighborhood of the current best solution, thereby enabling accurate approximation of the function near its extremum.

McKean-Vlasov-based approaches

In Chapter A.3, we develop a unified framework for McKean-Vlasov-based optimization algorithms, a family of methods in which a system of interacting particles evolves continuously according to a system of (stochastic) differential equations. We present the Consensus-Based Optimization algorithm (Pinnau et al., 2017), a paradigmatic representative of this family that has attracted substantial attention in recent years (Bungert et al., 2025; Fornasier et al., 2024; Gerber et al., 2025; Riedl, 2024). We also introduce the Stein Boltzmann Sampling algorithm (Serré et al., 2025), a contribution of this thesis. It uses Stein Variational Gradient Descent (SVGD) (Liu & Wang, 2016) to sample from the Gibbs distribution associated with the objective function. We give its asymptotic convergence guarantees, which hold for a Stein-universal kernel that vanishes on the boundary of the search space (see Section A.3.3). We also present two algorithmic variants of SBS, based on particle filtering and warm-starting, and an empirical validation through numerical experiments.

In Section A.3.3, we study SVGD on a compact domain. Its theory is developed on $\mathbb{R}^d$ (Duncan et al., 2023a; Korba et al., 2020; Liu, 2017; Lu et al., 2019), where nothing keeps the particles in the search space. We prove that, with a kernel that vanishes on the boundary of the domain, the continuous-time dynamics of SVGD exists and keeps the particles in the domain, without any regularity of the boundary. Along this dynamics, the Kullback–Leibler divergence to the target decreases, at a rate

equal to the kernelized Stein discrepancy. When the kernel is Stein-universal, the distribution of the particles converges weakly to the target. We apply these results to SBS. They are formalized in L∃∀N.

We then describe modular components applicable across McKean-Vlasov-based algorithms, designed to address another structural limitation of this family: the dynamics of the particles' distribution become deterministic in the mean-field limit (Tugaut, 2014), which impedes exploration of the search space and may cause the algorithms to become trapped in local optima. To mitigate this issue, we introduce a common noise mechanism that injects a stochastic perturbation simultaneously into all particles, in a manner that does not vanish in the mean-field limit. We propose two noise schemes: the first is an adaptation of the approach developed in Germain & Monmarché (2025), and the second is introduced here for the first time. This work is the result of a collaboration with Pierre Germain and Samuel Gruffaz; the contributions of this thesis are primarily focused on the implementation and numerical validation of the common noise mechanism, while its theoretical analysis was carried out by the aforementioned collaborators.

GLobal Optimization Benchmark

In Chapter A.4, we present the GLOBE library, an open-source collection[2] of implementations global optimization algorithms in C++, including all algorithms and modular components introduced in this thesis, together with a Python interface for broader accessibility. The library is architected to mirror the theoretical family structure described above: the decision-based and McKean-Vlasov families are realized as abstract base classes, from which concrete algorithms inherit, enabling all modular components to be reused across the family with minimal overhead. In particular, implementing a new decision-based algorithm requires only the specification of the decision rule, as all remaining infrastructure is provided by the abstract class. Similarly, a new McKean-Vlasov-based algorithm requires only the definition of the particle dynamics, while the discretization scheme, the common noise mechanism, and all other components are inherited from the base class. The library further provides a suite of benchmark functions and a benchmarking framework for systematic performance comparison, and is designed to serve both research and practical purposes.

Part B

A unifying framework for global optimization

In Chapter B.1, we introduce a general mathematical framework for global optimization algorithms. The central object of this framework is the notion of a *global optimization algorithm*, formalized as a sequence of Markov kernels $\kappa_n : \Omega^{n+1} \times \mathbb{R}^{n+1} \rightsquigarrow \Omega$ that encode the conditional distribution of the n-th query point given the history of all previous evaluations. By invoking the Ionescu-Tulcea extension theorem (Tulcea, 1949), this sequence of kernels canonically induces a probability measure over the space of infinite sample paths — the *trajectory measure* of the algorithm — as well as a family of finite-dimensional marginal distributions

[2] github.com/gaetanserre/GLOBe

representing the joint law of any finite prefix of the sequence. We establish several natural properties of these finite marginals and demonstrate the expressiveness of the framework by showing explicitly that it encompasses decision-based algorithms, McKean-Vlasov-based algorithms, and the CMA-ES algorithm (Hansen & Ostermeier, 2001). As a further illustration of its utility, we use this abstraction to provide a rigorous proof of a meta-algorithmic result stating that a global optimization algorithm is consistent over all Lipschitz functions if and only if it samples the entire search space (Malherbe & Vayatis, 2017a, Proposition 3).

The LEAN proof assistant

In Chapter B.2, we present the LEAN proof assistant, a software system for the mechanized verification of mathematical statements and their proofs. We describe the main features of LEAN and its mathematical ecosystem, with particular emphasis on the formalization of probability theory in the `Mathlib` library — a comprehensive, community-driven library of formalized mathematics in LEAN. Within this context, we present several contributions of this thesis to the formalization of Markov kernels, including a tool that automatically translates equalities between Markov kernels into equalities between morphisms in an appropriate category, thereby enabling the systematic reuse of category-theoretic machinery in probabilistic reasoning. These developments are intended to serve, directly or indirectly, as building blocks for the ongoing formalization of stochastic algorithms in LEAN, and of global optimization algorithms in particular.

Formalization of global optimization algorithms

In Chapter B.3, we present a LEAN formalization of the abstract framework for global optimization algorithms developed in Chapter B.1, together with proofs of the properties of the induced probability measures and a mechanized verification of the result of Malherbe & Vayatis (2017a, Proposition 3). We further provide LEAN formalizations of the PRS, LIPO, RANKOPT, and CMA-ES algorithms as instances of the abstract framework, following the constructions described in Chapter B.1, and thereby establish the first LEAN library of formalized global optimization algorithms. These formalization efforts substantiate the expressiveness and robustness of the proposed framework, and constitute a meaningful step toward the long-term goal of scaling the mechanized verification of optimization and stochastic algorithms in proof assistants — a goal that has become increasingly relevant in light of the growing effectiveness of Large Language Models in mathematical reasoning.

Associated publications

Ultimately, the work presented in the manuscript has led to various publications and presentations at international conferences.

- Serré G., Beja-Battais P., Chirrane S., Kalogeratos A. and Vayatis N. (2024). **LIPO+: Frugal global optimization for lipschitz functions.** *In Hellenic Conference on Artificial Intelligence.*

- Serré G., Kalogeratos A. and Vayatis N. (2025). **Stein Boltzmann Sampling: A Variational Approach for Global Optimization.** *In International Conference on Artificial Intelligence and Statistics.*
- Serré G., Kalogeratos A. and Vayatis N. (2026). **GLOBe: A Modular Global Optimization library.** *In Hellenic Conference on Artificial Intelligence.*
- Serré G., Kalogeratos A. and Vayatis N. (2026). **A Unifying Framework for Global Optimization: From Theory to Formalization.** *Submitted to Mathematical Programming.*
- Serré G., Germain P., Gruffaz S. and Kalogeratos A. (2026). **Enhancing Exploration in Global Optimization by Noise Injection in the Probability Measures Space.** *To be submitted to Transactions on Machine Learning Research.*
- Malherbe C, Serré G., Kalogeratos A. and Vayatis N. (2026). **A Ranking Approach to Global Optimization.** *To be submitted to Journal of Machine Learning Research.*



Figure 3. *Timeline of main innovations in global optimization (adapted from Rios & Sahinidis (2013)). Colors indicate the family of methods each innovation belongs to.*

Notations

This section introduces the notation used throughout this manuscript. Notations are organized by thematic groups for convenience.

Optimization and measurable spaces.

V is the function to optimize.

Ω is the compact space over which the optimization is performed.

$X^\star$ is the set of global optimizers of V.

$\mathbb{B}$ is the set of boolean values: $\{\top, \bot\}$.

The finite set $\{1, ..., N\}$ is denoted by $[N]$.

For any sequence $X_0, ..., X_n \in \Omega^{n+1}$, we write $\mathsf{C}_n \overset{\text{def}}{\equiv} (X_0, ..., X_n)$ and $f(\mathsf{C}_n) \overset{\text{def}}{\equiv} (f(X_0), ..., f(X_n))$.

Linear algebra.

$\mathbb{S}^d_{++}$ denotes the set of symmetric positive definite matrices of size d.

$\Omega_k[X]$ denotes the set of polynomial functions of degree k over a given subset of $\mathbb{R}^d$.

Measure theory.

$\mathcal{P}(E)$ denotes the set of probability measures on a given measurable space E. When $\Omega \subseteq \mathbb{R}^d$, $\mathcal{P}(\Omega)$ is identified with the set of probability measures ρ on $\mathbb{R}^d$ such that $\rho\big(\mathbb{R}^d \setminus \Omega\big) = 0$, i.e. that are *carried by* Ω.

λ denotes the Lebesgue measure on $\mathbb{R}^d$.

For a measure ν and a measurable function q that is nonnegative ν-almost everywhere, $q \cdot \nu$ denotes the measure of density q with respect to ν.

$\mathfrak{M}(X)$ denotes the σ-algebra of measures on X, i.e., the coarsest σ-algebra on the set of measures on X such that, for any measurable set B of X, the map $\mu \mapsto \mu(B)$ is measurable.

$\mathfrak{P}(X)$ denotes the σ-algebra of probability measures on X.

$\mathbf{1}$ denotes the one-point space with σ-algebra $\{\varnothing, \{1\}\}$.

$\mu|_A$ denotes the restriction of a measure μ to a set A.

$\mu\|_A$ denotes the renormalized restriction of a measure μ to a set A, defined as $\mu\|_A = \frac{\mu|_A}{\mu(A)}$.

The pushforward of a measure under a measurable function $f : (\mathcal{X}, \mathcal{B}) \to (\mathcal{Y}, \mathcal{A})$ is denoted by $\#$, defined as $f\#\nu(A) = \nu\big(f^{-1}(A)\big)$ for any $A \in \mathcal{A}$ and any measure ν on $(\mathcal{X}, \mathcal{B})$.

For a measure $\rho \in \mathcal{P}\big(\mathbb{R}^d\big)$, $m_p(\rho)$ denotes its p-th order moment, i.e., $m_p(\rho) = \int_{\mathbb{R}^d} |x|^p \, \mathrm{d}\rho(x)$.

$\mathcal{P}_p\big(\mathbb{R}^d\big)$ denotes the subspace of $\mathcal{P}\big(\mathbb{R}^d\big)$ consisting of probability measures with finite p-th moment.

Functional analysis and transition kernels.

For $A \subseteq \mathbb{R}^d$, A° and ∂A denote the interior and the boundary of A.

For real normed spaces E and F, the operator norm of a continuous linear map $L : E \to F$ is $\|L\| = \sup_{\|u\| \leq 1} \|L(u)\|$, and E^* denotes the space of continuous linear forms on E, with this norm. For a map g between real normed spaces, differentiable at x, $Dg(x)$ denotes its derivative at x. When $g : \mathbb{R}^d \to \mathbb{R}$, $\nabla g(x) \in \mathbb{R}^d$ is the vector such that $Dg(x)(u) = \nabla g(x)^\top u$ for every $u \in \mathbb{R}^d$. For a vector field v on $\mathbb{R}^d$, $\nabla \cdot v = \operatorname{tr} Dv$ denotes its divergence.

$B(X)$ denotes the set of bounded strongly-measurable functions on X.

Given two measurable spaces X and Y, $X \rightsquigarrow Y$ denotes the type of transition kernels from X to Y.

$\otimes_k$ denotes the composition between kernels.

$\overset{\leftarrow}{\circ}$ denotes the reverse composition between functions.

Optimization and kernel notation.

"argopt" denotes either "argmax" or "argmin" depending on the optimization problem considered.

$k_\sigma^{(d)}(x, y) \overset{\text{def}}{=} \exp\left(-\frac{\|x-y\|_2^2}{\sigma}\right)$ denotes the Gaussian kernel for any $(x, y) \in \left(\mathbb{R}^d\right)^2$ and $\sigma > 0$.

Contents

Part A – Methodological advances in global optimization 19
1 Background: Estimation of distribution 21
1.1 CMA-ES 22
1.2 Bayesian optimization 26
2 Decision-based approaches 31
2.1 Existing approaches 34
2.1.1 LIPO 34
2.1.2 Adaptative LIPO 34
2.1.3 Limitations of LIPO 36
2.2 LIPO+ and AdaLIPO+ 36
2.2.1 Stopping criterion 36
2.2.2 Decaying exploration rate 37
2.2.3 Numerical experiments 38
2.2.4 Limitations 39
2.3 RankOpt 41
2.3.1 The ranking structure of a real-valued function 41
2.3.2 The RankOpt algorithm 43
2.3.3 Adaptive RankOpt 44
2.3.4 Implementation and computational aspects 45
2.3.5 Numerical experiments 46
2.4 Trust region module 57
3 McKean-Vlasov-based approaches 61
3.1 Consensus-based optimization 64
3.2 Stein Boltzmann Sampling 65
3.2.1 Boltzmann distribution 66
3.2.2 Stein Variational Gradient Descent 68
3.2.3 Guarantees of SBS 69
3.2.4 SBS variants 69
3.2.5 Choice of hyperparameters 71
3.2.6 Experimental evaluation 72
3.2.7 Discussion 74
3.3 SVGD on a compact domain 76
3.3.1 Main results 77
3.3.2 Setting 78
3.3.3 Stein's method 79
3.3.4 The kernelized Stein discrepancy 84
3.3.5 The steepest descent of the KL divergence 90
3.3.6 SVGD in continuous time 92
3.3.7 The time derivative of the KL divergence 101
3.3.8 Convergence 105
3.3.9 Application to Stein Boltzmann Sampling 112
3.4 Common noise plugin module 114

3.4.1 Noise through moment perturbation . . . 115
3.4.2 Noise using tangent space . . . 117
3.4.3 Numerical experiments . . . 122
4 GLobal Optimization Benchmark (GLOBe) . . . 129
4.1 The GLOBe library . . . 131
4.1.1 Modular architecture . . . 132
4.1.2 Base optimizer interface . . . 133
4.1.3 Decision-based optimizers . . . 133
4.1.4 Particle-based optimizers . . . 134
4.1.5 Modular plugins . . . 134
4.2 Benchmarking features . . . 135
4.2.1 Benchmarking functions . . . 135
4.2.2 Benchmarking engine . . . 137

Appendix

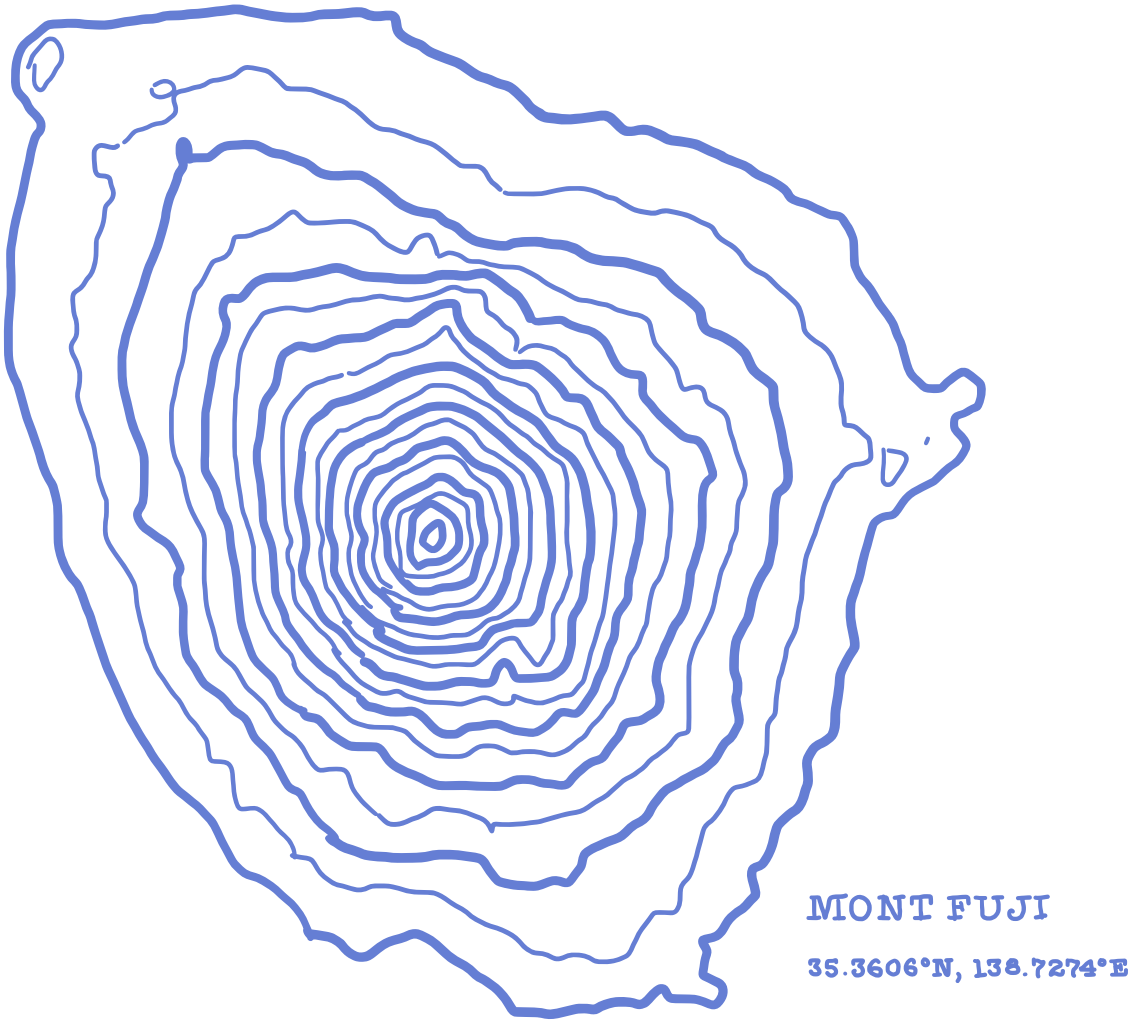
Part B – A unified and formalized framework for global optimization . . . 155
1 A unifying framework for global optimization . . . 157
1.1 Instantiation of the framework . . . 161
1.2 Properties of the measures . . . 163
1.2.1 Equality of restricted measures . . . 163
1.2.2 Monotonicity by truncation . . . 164
1.3 Sketch proof of Proposition B.1.2 . . . 166
2 Introduction to formal methods . . . 171
2.1 LEAN theorem prover . . . 173
2.2 Probability theory in LEAN . . . 174
2.2.1 Markov kernels in LEAN . . . 177
3 Formalization of global optimization algorithms . . . 189
3.1 Formalization of decision-based algorithms . . . 193
3.2 Formalization of CMA-ES . . . 195
3.3 On the formalization of McKean-Vlasov-based approaches . . . 196
3.4 Discussion and perspectives . . . 197

Appendix

Bibliography . . . 216

Part A

Methodological advances in global optimization



Contents

1 Background: Estimation of distribution 21
1.1 CMA-ES 22
1.2 Bayesian optimization 26
2 Decision-based approaches 31
2.1 Existing approaches 34
2.2 LIPO+ and AdaLIPO+ 36
2.3 RankOpt 41
2.4 Trust region module 57
3 McKean-Vlasov-based approaches 61
3.1 Consensus-based optimization 64
3.2 Stein Boltzmann Sampling 65
3.3 SVGD on a compact domain 76
3.4 Common noise plugin module 114
4 GLobal Optimization Benchmark (GLOBe) 129
4.1 The GLOBe library 131
4.2 Benchmarking features 135

Appendix

1 — Background: Estimation of distribution

Abstract. We briefly present the family of Estimation of Distribution Algorithms (EDAs), a general class of optimization methods that learn from promising solutions and use this knowledge to guide the search. Rather than constituting a primary object of study in this thesis, these methods are introduced here because they will serve as reference algorithms in our numerical benchmarks. We discuss the principles underlying this family and examine two prominent algorithms: the Covariance Matrix Adaptation Evolution Strategy, based on an adaptive Gaussian distribution, and Bayesian Optimization, which leverages surrogate models when function evaluations are expensive.

Evolutionary Algorithms (EAs) represent a natural class of methods for black-box optimization, relying solely on function evaluations to guide populations of candidate solutions toward promising regions. Classical EAs, such as genetic algorithms, achieve this through variation operators—crossover and mutation—whose design encodes implicit assumptions about the problem structure that may prove inappropriate or suboptimal. In contrast, Estimation of Distribution Algorithms (EDAs) (Etxeberria, 1999; Larranaga & Bielza, 2025; Lozano, 2006; Mühlenbein & Mahnig, 1999; Mühlenbein & Paass, 1996; Pelikan et al., 1999) reformulate this process: rather than hand-crafting operators to recombine solutions, EDAs make explicit the underlying probabilistic model.

EDAs follow a canonical procedure: at each generation, a statistical model is estimated from selected individuals, and new candidates are sampled from this model. This shift reframes the central design question from "which operators?" to "which probabilistic model?", leading to a substantial body of research.

The expressiveness of the model determines which variable interactions can be exploited. Early EDAs employed simple models: the Univariate Marginal Distribution Algorithm (Mühlenbein, 1997) assumes independence and estimates products of univariate marginals, whereas the Bivariate Marginal Distribution Algorithm and Bayesian Optimization Algorithm (Pelikan et al., 1999; Pelikan & Mühlen-

bein, 1999) (distinct from Bayesian Optimization, see Section A.1.2) progressively capture higher-order dependencies via graphical models.

For continuous search spaces, the multivariate Gaussian distribution emerges as a natural choice, giving rise to Gaussian EDAs. By fitting a full-covariance Gaussian to selected solutions, these methods capture linear dependencies and align the sampling distribution with the objective's geometry. This principle underpins the *Covariance Matrix Adaptation Evolution Strategy* (CMA-ES) (Hansen & Ostermeier, 2001), detailed in Section A.1.1. CMA-ES may be viewed as a principled Gaussian EDA enriched with carefully designed adaptation rules for mean, covariance, and step-size, ensuring robust performance across diverse problems.

A complementary research direction combines EDAs with Bayesian inference and surrogate modeling. Rather than sampling from a parametric distribution, these methods construct a surrogate—typically a Gaussian process—to model the objective function, and leverage an acquisition function balancing exploration and exploitation to select evaluation points. This framework, known as Bayesian Optimization (Brochu et al., 2010; Jones et al., 1998; Močkus, 1974), proves particularly valuable when function evaluations are costly, as the surrogate provides a computationally cheap approximation. Its connection to the broader EDA paradigm stems from the shared principle of leveraging a learned probabilistic model to guide search.

We examine these two paradigms in subsequent sections: Section A.1.1 details CMA-ES and its theoretical foundations as a continuous Gaussian EDA, while Section A.1.2 presents Bayesian Optimization and its methodology for expensive black-box optimization.

1.1 CMA-ES

CMA-ES, the *Covariance Matrix Adaptation Evolution Strategy* (Hansen & Ostermeier, 1996; 2001), exemplifies a sophisticated continuous Gaussian EDA and stands as a canonical algorithm in black-box optimization. It is a zeroth-order, sequential stochastic algorithm that iteratively samples candidates from a multivariate Gaussian distribution, updated through the information learned from prior evaluations. The choice of Gaussian distribution is principled: given its mean and covariance matrix, it maximizes entropy, providing the distribution of greatest uncertainty, a desirable property for unguided exploration. CMA-ES achieves its objective by adaptively modifying the mean and covariance matrix to converge toward the global optimum using solely the evaluation history. Given its complexity and numerous variants (Hansen et al., 2003; Hansen, 2006; 2023), the exposition focuses on foundational principles and key mechanisms; comprehensive details are available in Hansen (2023); Lozano (2006).

Throughout this section, μ and λ denote the number of selected and sampled individuals per generation, respectively. The term *generation* replaces iteration to recall the evolutionary inspiration of the method.

Mean update. At each generation g, CMA-ES samples λ points from a multivariate Gaussian distribution

$$\left(X_i^{(g)}\right)_{1\leq i\leq\lambda} \sim m^{(g)} + \sigma^{(g)}\mathcal{N}\big(0, C^{(g)}\big),$$

where $m^{(g)}$ is the mean vector, $\sigma^{(g)}$ is the "overall" standard deviation and $C^{(g)}$ is the covariance matrix. It is equivalent to say that $x_i^{(g)} \sim \mathcal{N}\Big(m^{(g)}, \big(\sigma^{(g)}\big)^2 C^{(g)}\Big)$.

To update the mean, CMA-ES begins by sorting the points according to their evaluation, then selects the μ best ones

$$f\left(x_{1:\lambda}^{(g)}\right) \leq ... \leq f\left(x_{\mu:\lambda}^{(g)}\right) \leq f\left(x_{\mu+1:\lambda}^{(g)}\right) \leq ... \leq f\left(x_{\lambda:\lambda}^{(g)}\right), \tag{1}$$

where the notation "$i:\lambda$" denotes the index of the i-th best point. The update rule of the mean at generation $g+1$ is a weighted average of those points

$$m^{(g+1)} = m^{(g)} + c_m \sum_{i=1}^{\mu} w_i \left(x_{i:\lambda}^{(g)} - m^{(g)}\right), \tag{2}$$

where

$$\sum_{i=1}^{\mu} w_i = 1, \quad w_1 \geq ... \geq w_\mu \geq 0, \tag{3}$$

and c_m is a learning rate, usually set to 1. In that case, Eq. 2 simply becomes

$$m^{(g+1)} = \sum_{i=1}^{\mu} w_i x_{i:\lambda}^{(g)}.$$

The choice of the weights is crucial in CMA-ES as they represent the trade-off between exploration and exploitation. Indeed, if the weights are to sparse, the algorithm will consider points that are far from a potential global minimum as much as points that are close to it. On the other hand, if the weights are too concentrated, the algorithm will not explore enough the space and might be stuck in a local minimum. To address this issue, the authors defined the quantity

$$\mu_{\text{eff}} = \left(\frac{\|w\|_1}{\|w\|_2}\right)^2 = \frac{\left(\sum_{i=1}^{\mu}|w_i|\right)^2}{\sum_{i=1}^{\mu} w_i^2} = \frac{1}{\sum_{i=1}^{\mu} w_i^2}.$$

From Eq. 3, one can easily derive $1 \leq \mu_{\text{eff}} \leq \mu$, the latter happening when all the weights are equal. The μ_{eff} quantity represents the loss of variance due to the selection of the best points. According to the authors, $\mu_{\text{eff}} \approx \lambda/4$ indicates a reasonable setting of w_i. A simple and decent setting is to set $w_i \propto \mu - i + 1$ and $\mu \approx \lambda/2$, which gives $\mu_{\text{eff}} \approx 3\lambda/8$.

Covariance matrix update. To update the covariance matrix, one must estimate it using the points $(x_i)_{1\leq i\leq\lambda}$. For simplicity, assume $\sigma = 1$. If $\sigma \neq 1$, the covariance matrix can be rescaled by $1/\sigma^2$.

With sufficiently many samples, one can estimate the empirical covariance matrix

$$C_{\text{emp}}^{(g+1)} = \frac{1}{\lambda - 1} \sum_{i=1}^{\lambda} \left(x_i^{(g+1)} - \frac{1}{\lambda} \sum_{i=1}^{\lambda} x_i^{(g+1)} \right) \left(x_i^{(g+1)} - \frac{1}{\lambda} \sum_{i=1}^{\lambda} x_i^{(g+1)} \right)^\top.$$

An alternative estimator replaces the empirical mean with the true mean $m^{(g)}$ from the distribution

$$C_{\lambda}^{(g+1)} = \frac{1}{\lambda} \sum_{i=1}^{\lambda} \left(x_i^{(g+1)} - m^{(g)} \right) \left(x_i^{(g+1)} - m^{(g)} \right)^\top. \tag{4}$$

The matrix $C_{\lambda}^{(g+1)}$ is an unbiased maximum likelihood estimator of $C^{(g)}$. The key distinction lies in the reference mean: $C_{\text{emp}}^{(g+1)}$ uses the empirical mean of the actual sample, while $C_{\lambda}^{(g+1)}$ uses the true distribution mean. CMA-ES employs the latter approach.

However, to obtain a covariance matrix better adapted to the best solutions, CMA-ES employs weighted averaging with the selection mechanism from Eq. 2

$$C_{\mu}^{(g+1)} = \sum_{i=1}^{\mu} w_i \left(x_{i:\lambda}^{(g+1)} - m^{(g)} \right) \left(x_{i:\lambda}^{(g+1)} - m^{(g)} \right)^\top. \tag{5}$$

This estimator prioritizes the best solutions, enabling faster convergence. However, it demands sufficiently many samples: μ_{eff} must be large enough to yield reliable estimates. For the Sphere function, μ_{eff} must be roughly $10d$ for reliable estimation. Since $\mu_{\text{eff}} \approx \lambda/4$, this requires $\lambda \approx 40d$. In practice, such sample counts are often prohibitive, necessitating more efficient estimation methods. CMA-ES addresses this via the *rank-μ* method, which leverages information across generations.

Rank-μ method The rank-μ method incorporates information from previous generations into the current estimate

$$C_{\text{rank-}\mu}^{(g+1)} = \frac{1}{g+1} \sum_{i=0}^{g} \frac{1}{{\sigma^{(i)}}^2} C_{\mu}^{(i+1)}, \tag{6}$$

where $\sigma^{(i)}$ is the step-size at generation i. This assigns equal weight to all generations. A natural refinement weights recent generations more heavily via exponential smoothing

$$C_{\text{rank-}\mu}^{(g+1)} = (1 - c_\mu) C_{\text{rank-}\mu}^{(g)} + c_\mu \frac{1}{{\sigma^{(g)}}^2} C_{\mu}^{(g+1)}, \tag{7}$$

where $c_\mu \leq 1$ is the learning rate. A principled choice is $c_\mu \approx \min\left(1, \frac{\mu_{\text{eff}}}{d^2}\right)$. Eq. 7 can be written as

$$C_{\text{rank-}\mu}^{(g+1)} = (1 - c_\mu) C_{\text{rank-}\mu}^{(g)} + c_\mu \sum_{i=1}^{\mu} w_i Y_{i:\lambda}^{(g+1)} \cdot Y_{i:\lambda}^{(g+1)\top}, \tag{8}$$

where $Y_{i:\lambda}^{(g+1)} = \frac{X_{i:\lambda}^{(g+1)} - m^{(g)}}{\sigma^{(g)}}$. The name *rank-$\mu$* reflects that the rank of the weighted sum of outer products is $\min(\mu, d)$. More generally, CMA-ES extends Eq. 8 to employ λ weights without constraints on their sum or sign

$$C_{\text{rank-}\mu}^{(g+1)} = \left(1 - \sum_{i=1}^{\lambda} w_i c_\mu\right) C_{\text{rank-}\mu}^{(g)} + c_\mu \sum_{i=1}^{\lambda} w_i Y_{i:\lambda}^{(g+1)} Y_{i:\lambda}^{(g+1)\top}. \tag{9}$$

To further concentrate the estimation of the covariance matrix on the best solutions, CMA-ES employs the *rank-1* method.

Rank-1 method The *rank-1* method uses only the best solution to update the covariance matrix. The update rule is just an adaption of Eq. 8 where only $Y_{1:\lambda}^{(g+1)}$ is used

$$C_{\text{rank-1}}^{(g+1)} = \left(1 - \sum_{i=1}^{\lambda} w_i c_1\right) C_{\text{rank-1}}^{(g)} + c_1 Y_{1:\lambda}^{(g+1)} Y_{1:\lambda}^{(g+1)\top},$$

where, according to the authors, $c_1 \approx 2/n^2$ is a good choice. However, this method discards directional information, since

$$Y_{1:\lambda}^{(g+1)} Y_{1:\lambda}^{(g+1)\top} = -Y_{1:\lambda}^{(g+1)} \left(-Y_{1:\lambda}^{(g+1)\top}\right).$$

To recover this directional information, CMA-ES introduces an *evolution path*

$$p_c^{(g+1)} = (1 - c_c) p_c^{(g)} + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}} \frac{\sum_{i=1}^{\mu} w_i \left(x_i^{(g+1)} - m^{(g)}\right)}{\sigma^{(g)}}, \tag{10}$$

where $c_c \leq 1$ and $p_c^{(0)} = 0$. The term $\sqrt{c_c(2 - c_c)\mu_{\text{eff}}}$ is a normalization constant in order to $p_c^{(g)}$ to follow the same distribution as $Y_{1:\lambda}^{(g)}$.

The covariance matrix update in CMA-ES combines the rank-μ and rank-1 components via Eq. 9 and Eq. 10

$$C^{(g+1)} = \underbrace{\left(1 - c_1 - c_\mu \sum_{i=1}^{\lambda} w_i\right)}_{\approx 1} C^{(g)} + c_1 \underbrace{p_c^{(g+1)} p_c^{(g+1)\top}}_{\text{rank-1 update}} + c_\mu \underbrace{\sum_{i=1}^{\lambda} w_i Y_{i:\lambda}^{(g+1)} Y_{i:\lambda}^{(g+1)\top}}_{\text{rank-}\mu \text{ update}} \tag{11}$$

Step-size update. The final parameter is the step-size $\sigma^{(g)}$. CMA-ES adapts it via the evolution of the step directions, represented by $Y_{i:\lambda}^{(g)}$. Three distinct behaviors arise:

- the steps consistently point in the same direction, suggesting the optimum lies ahead; increasing the step-size accelerates convergence;
- the steps cancel out, indicating that the optimum lies in the interior of the region spanned by the steps; decreasing the step-size enables finer exploration;
- the steps follow nearly orthogonal directions, indicating that the algorithm traces the contour lines of the objective function; the step-size remains constant.

These cases are illustrated in Figure A.1.1. To distinguish these cases, CMA-ES monitors the length of the conjugate evolution path

$$p_\sigma^{(g+1)} = (1 - c_\sigma) p_\sigma^{(g)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\text{eff}}} C^{-\frac{1}{2}} \sum_{i=1}^{\mu} w_i Y_{i:\lambda}^{(g+1)},$$

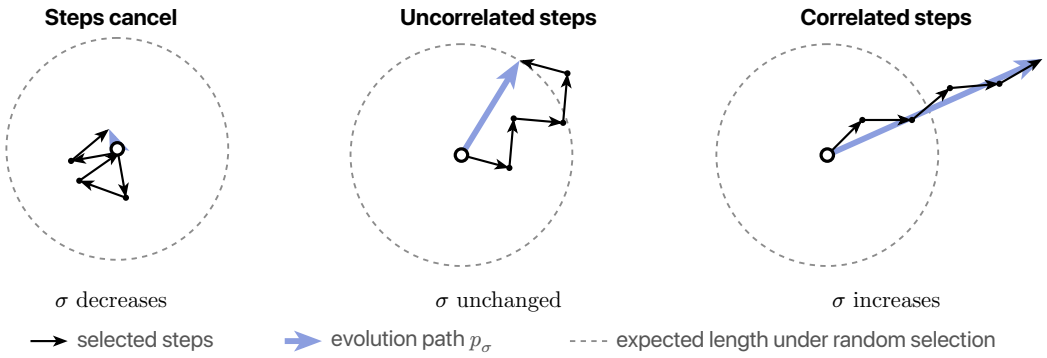


Figure A.1.1. *Step-size adaptation from the length of the evolution path, compared to its expected length under random selection (dashed circle).* Left*: steps cancel each other, the step-size must be decreased.* Middle*: steps follow almost orthogonal directions, the step-size must be kept constant.* Right*: steps are correlated and go in the same direction, the step-size must be increased. Adapted from Hansen (2023).*

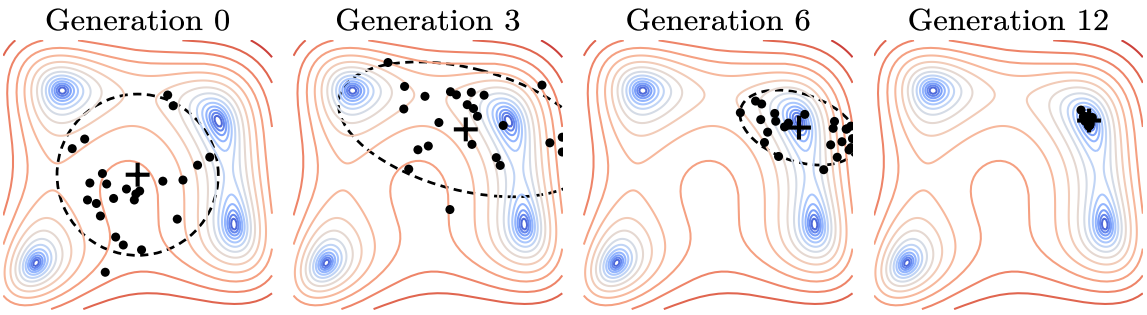


Figure A.1.2. *Evolution of CMA-ES samples on the Himmelblau function (level lines). Each panel shows the samples of one generation (dots), the mean of the search distribution (cross) and its 2σ ellipse (dashed). The distribution of samples shifts progressively toward an optimum across generations, demonstrating how the algorithm's iterative adaptation concentrates the search around an optimal solution.*

One can verify that if $p_\sigma^{(0)} \sim \mathcal{N}(0, I)$, then $p_\sigma^{(g)} \sim \mathcal{N}(0, I)$ for all g. Thus, comparing its norm with its expected value guides the step-size adaptation: if $\left\|p_\sigma^{(g)}\right\|_2$ is below the expected value, σ should decrease; otherwise it should increase. The update rule is

$$\sigma^{g+1} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\left\|p_\sigma^{g+1}\right\|_2}{\mathbb{E}_{X\sim\mathcal{N}(0,I)}[\|X\|_2]} - 1\right)\right),$$

where $d_\sigma \approx 1$. This concludes the exposition of CMA-ES's core principles. Figure A.1.2 illustrates how the samples progressively concentrate around an optimum on the Himmelblau function.

1.2 Bayesian optimization

Bayesian Optimization BAYESOPT is a sequential, model-based strategy for the global optimization of black-box functions that are expensive to evaluate (Brochu et al., 2010; Jones et al., 1998; Močkus, 1974). It is particularly suited to settings where the objective function $V : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$ carries a significant computational

or monetary cost and if the dimension d is not too high (typically $d \leq 20$). It operates by construction a surrogate model of the objective function supposedly cheaper to evaluate to guide the search for the global optimum. The surrogate model provides a probabilistic estimate of the function's behavior across the search space, enabling informed decision-making about where to sample next. This is achieved through an acquisition function that balances exploration (sampling in regions of high uncertainty) and exploitation (sampling in regions predicted to be of interest). By iteratively updating the surrogate model with new observations, Bayesian Optimization efficiently navigates the search space, making it a powerful tool for optimizing expensive black-box functions.

Surrogate Model. The surrogate model in Bayesian Optimization is a Gaussian process (GP), which places a prior distribution directly over functions. A GP is fully specified by a mean function $m(x)$ and a covariance kernel $k(x, x')$, where the kernel encodes prior assumptions about the smoothness and structure of V. Formally, for any finite set of observations $\mathbf{C}_n$, the GP prior states that the vector of function values is jointly Gaussian

$$\begin{pmatrix} V(X_0) \\ \vdots \\ V(X_n) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} m(X_0) \\ \vdots \\ m(X_n) \end{pmatrix}, K \right),$$

where $[K]_{ij} = k(X_i, X_j)$. This is commonly abbreviated as $V \sim \mathcal{GP}(m, k)$, with the understanding that this notation refers to the above finite-dimensional marginals for any choice of input points. In practice, the mean function is set to zero, and the kernel to the squared-exponential (RBF)

$$k_\theta(x, x') = \sigma_V^2 \exp\left(-\frac{\|x - x'\|_2^2}{2l^2} \right),$$

with hyperparameters $\theta = (\sigma_V^2, l)$ representing the signal variance and length-scale, respectively. Fitting the GP amounts to selecting θ given the observed data. This is done by maximizing the log-marginal likelihood given by

$$\log \mathbb{P}(V(\mathbf{C}_n) \mid \mathbf{C}_n, \theta) = -\frac{1}{2}\left(\underbrace{(\mathbf{C}_n)^\top K_\theta^{-1} V(\mathbf{C}_n)}_{\text{data fit}} + \underbrace{|K_\theta|}_{\text{complexity}} + \underbrace{(n+1)\log 2\pi}_{\text{normalization}} \right).$$

This maximization is typically performed via gradient-based optimization such as L-BFGS or derivative-free optimization methods such as DIRECT (Jones, 2001) or BOBYQA (Powell, 2009) (see Martinez-Cantin (2014)).

Once $\hat{\theta}$ is fitted, the GP posterior at a new point x is Gaussian with mean and variance given by

$$\mu(x) = k_x^\top K_{\hat{\theta}}^{-1} V(\mathbf{C}_n), \;\; \sigma^2(x) = k(x, x) - k_x^\top K_{\hat{\theta}}^{-1} k_x,$$

where $k_x = [k_{\hat{\theta}}(X_0, x), ..., k_{\hat{\theta}}(X_n, x)]^\top$. The posterior mean serves as a predictor of the function value at x, while the posterior variance quantifies the uncertainty of this prediction, guiding the acquisition function in selecting promising candidates

for evaluation. Crucially, the GP interpolates the observed data, ensuring that $\mu(X_i) = V(X_i)$ and $\sigma^2(X_i) = 0$ for all $0 \leq i \leq n$.

Acquisition function. At each iteration, BAYESOPT selects the next point to evaluate by maximizing an *acquisition function* α. which encodes a trade-off between exploration (querying regions of high uncertainty) and exploitation (querying regions of promising predicted value). Common acquisition functions include the Probability of Improvement (PI) and Expected Improvement (EI).

- **PI**: $\alpha_{\mathrm{PI}}(x) = \mathbb{P}_{V(x)\sim\mathcal{GP}}\left(V(x) > \max_{0\leq i\leq n} V(X_i)\right)$. The drawback of PI is that it does not take into account the magnitude of the improvement, which can lead to suboptimal performance.
- **EI**: $\alpha_{\mathrm{EI}}(x) = \mathbb{E}_{V(x)\sim\mathcal{GP}}\left[\max\left(V(x) - \max_{0\leq i\leq n} V(X_i), 0\right)\right]$. This term is a refinement of **PI** that takes into account the magnitude of the improvement, leading to better performance in practice.

Both acquisition functions admit closed-form expressions given the GP posterior, and both vanish at observed points since $\sigma(X_i) = 0$, which naturally prevents re-evaluating the same location. A hyperparameter $\xi \geq 0$ can be introduced in both to impose a minimum improvement threshold, trading off exploitation for exploration; it is omitted here for brevity. Maximizing the acquisition function is typically performed the same way as the fitting of the GP hyperparameters.

Bayesian Optimization loop. The algorithm proceeds iteratively:

i. **Initialization:** Sample an initial set of points X_0 from a space-filling design (e.g. Latin Hypercube Sampling) and evaluate V at these points.
ii. **Fitting the surrogate model:** Fit the GP to the observed data to obtain the posterior distribution.
iii. **Acquisition optimization:** Maximize the acquisition function to select the next point X_{n+1} to evaluate.
iv. **Evaluation:** Evaluate $V(X_{n+1})$, update the dataset with this new observation and return to step **ii.**

An illustration of the BAYESOPT behavior is given in Figure A.1.3.

Complexity. A practical limitation of BAYESOPT is its scalability. Fitting the GP and computing the posterior both require inverting the Gram matrix K, which costs $\mathcal{O}(n^3)$ in time. Furthermore, the number of observations needed to adequately cover Ω grows exponentially with d, a manifestation of the curse of dimensionality. In practice, BAYESOPT is therefore most effective in low-dimensional settings and with budgets of at most a few hundred evaluations, though several approaches have been proposed to mitigate these limitations (Kandasamy et al., 2015; Wang et al., 2016).

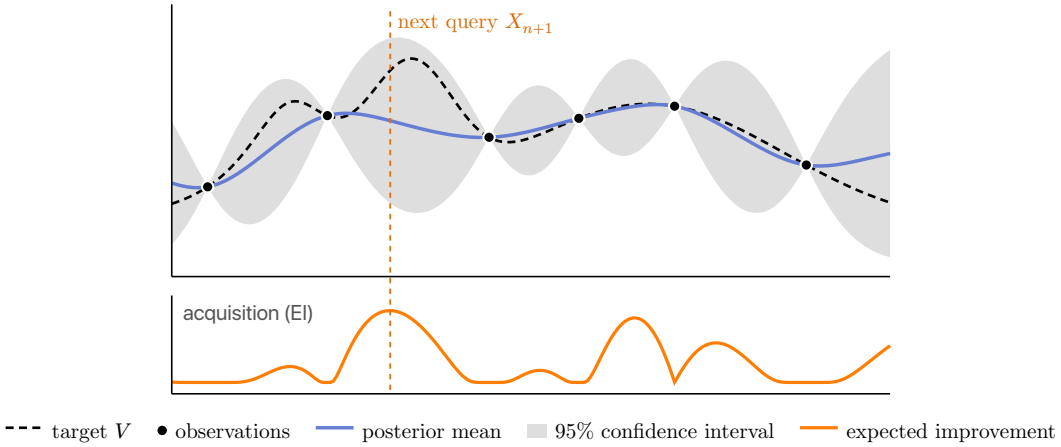


Figure A.1.3. *Illustration of* BAYESOPT *using the Expected Improvement acquisition function.* Top*: the target function (dashed black curve), the GP posterior mean (blue curve) and the* 95% *confidence interval of the GP posterior (shaded area), given the observations.* Bottom*: the acquisition function (orange curve), whose maximizer is the next evaluation point.*

2 — Decision-based approaches

Abstract. We present the family of *decision-based* algorithms for global optimization, a broad class of methods that rely on decision rules to guide the search for the global optimum. These algorithms sample candidate points from a fixed distribution and accept them based on a decision rule that evaluates their potential as global optimizers. By framing various optimization strategies within this unified framework, we can analyze their properties and develop general improvements applicable across the family, such as using trust-region methods to enhance performance. We examine existing decision-based algorithms for Lipschitz optimization and present novel contributions developed in this thesis.

Several global optimization algorithms can be seen as particular cases of a more general family of algorithms that we call *decision-based* approaches (Fourati et al., 2025; Fourati et al., 2026; Malherbe et al., 2016; Malherbe & Vayatis, 2017a; Serré et al., 2024). These algorithms rely on a sequence of *decision rules* to guide the search for the global optimum. Decision rules are $[0, 1]$-valued functions that take as input the history of the optimization process (i.e. the points that have been sampled and their corresponding function values) and a candidate point, and output a value in $[0, 1]$ that indicates the potential of the candidate point to be a global optimizer. This is related to the notion of *probabilistic predicate* (Morgan et al., 1996) and *fuzzy predicate* (Hájek, 2001).

The term *decision-based* is chosen to emphasize the role of this predicate: at every iteration, the algorithm faces a binary decision — to evaluate or not to evaluate a given candidate — and it is precisely the rule governing this decision, rather than any model of the objective itself, that drives the search. This terminology distinguishes these approaches from *model-based* or *surrogate-based* strategies, such as Bayesian optimization, where the decision of where to evaluate next is mediated by an explicit probabilistic model of the objective; here, the decision rule operates directly on the history of evaluations, making the decision process itself the primary object of design.

At each iteration, decision-based algorithms sample candidate points from a fixed probability distribution (e.g. the uniform distribution) and accepts this candidate with probability given by a decision rule. If the candidate point is accepted, the function is evaluated at this point and the value is stored to compute the decision rule at the next iteration. The decision rules are designed to capture the potential of a candidate point to be a global optimizer based on the history of the optimization process. By doing so, these algorithms can focus their search on promising regions of the search space. The only difference between decision-based algorithms is the decision rules they use.

Definition A.2.1 – *Decision rule*

Let $n \in \mathbb{N}$. A *decision rule* is a function $D_n : \Omega^{n+1} \times \mathbb{R}^{n+1} \times \Omega \to [0, 1]$.

Remark

We say that a decision rule D is *valid* if, for any $x^\star \in X^\star$ and any sequence of points C_n, it holds $D(\mathsf{C}_n, V(\mathsf{C}_n), x^\star) > 0$. In other words, a valid decision rule never rejects a global optimizer.

An equivalent way formulate decision-based algorithms is to make use of *stochastic predicates*, i.e, Markov kernels to $\mathbb{B}$. At each iteration, the algorithm sample a boolean value from the kernel given the history of the optimization process and the candidate point. Any measure returned by a *decision kernel* is of the form $\alpha\delta_\top + (1-\alpha)\delta_\bot$, $\alpha \in [0, 1]$. A valid decision kernel would be a kernel that returns $\delta_\top$ on any global optimizer.

This framework is closely related to the notion of *Pure Adaptive Search* (PAS) (Zabinsky, 2003; Zabinsky & Smith, 1992), of which it can be seen as a generalization. In PAS, the decision rule is a strict improvement condition: a candidate point is accepted if and only if it improves upon the best value observed so far, corresponding to the indicator $\mathbb{1}[V(X) < \min_{0 \le i \le n} V(X_i)]$ — assuming a minimization problem. Decision-based algorithms relax this hard predicate into a *fuzzy* one: rather than enforcing a binary improvement condition, decision rules can encode softer notions of promise that allow candidate points to be accepted even without improving the current best value. This relaxation naturally promotes better exploration of the search space, in the spirit of algorithms such as Simulated Annealing (Kirkpatrick et al., 1983), and enables decision rules to be tailored to the geometry of the problem at hand. Note that PAS is not a decision-based algorithm, as its decision rule depends on the value of the objective function at the candidate point, which is not known before evaluation.

This perspective also resonates with the literature on *adaptive importance sampling* (Cappé et al., 2008; Rubinstein, 1999). There, one seeks to estimate an integral or expectation by drawing samples from a proposal distribution that overweights regions of interest, the discrepancy with the target distribution being corrected through importance weights; adaptive variants, such as the cross-entropy method (Rubinstein, 1999), iteratively refine this proposal using previously drawn samples so that it progressively concentrates around promising regions, typically level sets

of the objective or rare-event regions. Decision-based approaches can be seen as a discrete, sequential counterpart of this idea: instead of reweighting a continuous proposal density, a decision rule directly filters candidate points, drawn from a fixed reference distribution over the search space, on the basis of information accumulated from past evaluations. This amounts to constructing an implicit and evolving notion of "where to sample next", without ever having to represent or update an explicit proposal density, as is required in importance sampling.

Decision-based algorithms can be seen as instances of the following abstract algorithm.

Algorithm A.2.1: DECISION-BASED ALGORITHM

Input:
$V : \Omega \to \mathbb{R}$; $n \in \mathbb{N}$; $\mu \in \mathcal{P}(\Omega)$; a sequence of decision rules $(D_n)_{n\in\mathbb{N}}$
Output:
An approximation of $x^\star$

Initialisation:
$X_0 \sim \mu$
for $i = 0$ **to** n **do**

  do

    $X \sim \mu$
    $b \sim \mathcal{B}(D_i(\mathsf{C}_i, V(\mathsf{C}_i), X))$ *Bernoulli r.v. parametrized by the decision rule*

  while $\neg b$ **end** *Candidate rejection loop*
  $X_{i+1} \leftarrow X$
  $V(X_{i+1}) \leftarrow V(X)$

end
return $\underset{0\le i\le n}{\text{argopt}}\ V(X_i)$

Interpreting decision-based approaches as instances of our abstract algorithm simplifies their study to analyzing decision rule sequences. Beyond this simplification, the framework itself offers a new avenue for research: identifying improvements at the algorithmic level that benefit all decision-based methods.

In the following, we present existing decision-based algorithms and introduce new ones developed in this thesis, show their formulation within this framework, and propose a general module for enhancing performance.

Future work includes the development of more sophisticated modules within the decision-based framework, in particular through hybridization with other optimization paradigms, as well as a deeper investigation of the theoretical properties of decision-based algorithms when augmented with such modules.

2.1 Existing approaches

The following sections present existing decision-based algorithms for global optimization of Lipschitz functions.

Definition A.2.2 – *Lipschitz function*

Let $\kappa \in \mathbb{R}_+^*$. A function $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ is said to be κ-Lipschitz if, for any $x, y \in \Omega$,

$$|f(x) - f(y)| \leq \kappa \, \|x - y\|_2.$$

2.1.1 LIPO

LIPO is a sequential and stochastic algorithm introduced in Malherbe & Vayatis (2017a) designed to maximize a κ-Lipschitz function $V : \Omega \to \mathbb{R}$ where κ is known. LIPO is a *decision-based* algorithm where the probability measure μ is the uniform distribution on Ω and the decision rules D_n^{LIPO} are defined as follows

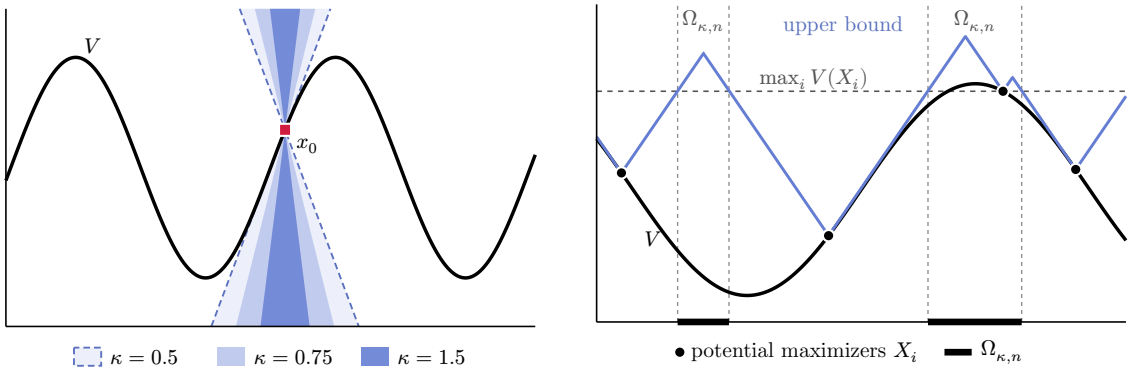
$$D_n^{\text{LIPO}}(\mathbf{C}_n, V(\mathbf{C}_n), x) = \begin{cases} 1 \text{ if } \max\limits_{0 \leq i \leq n} V(X_i) \leq \min\limits_{0 \leq i \leq n} V(X_i) + \kappa \, \|x - X_i\|_2 \\ 0 \text{ otherwise} \end{cases}. \quad (12)$$

Note that this decision rule is actually a predicate, as it only takes values in $\{0, 1\}$. Therefore, LIPO accepts a candidate point if and only if the inequality in Eq. 12 is satisfied.

By making use of the κ-Lipschitz property of V, LIPO constructs an upper bound of the function at x, which is given by the right-hand side of the inequality in Eq. 12, and checks the potential of the the candidate point before evaluating the function at it. The next candidate is considered as a potential maximizer if the upper bound at X_{n+1} is greater than the function value at the best maximizer found so far. If the candidate is accepted, LIPO evaluates V at x, and the value is stored to compute Eq. 12 at the next iteration. The fact that V is κ-Lipschitz ensures that no global maximizer is rejected thus that D_n^{LIPO} is a valid decision rule. The algorithm is consistent because the region of potential maximizers tightens around the global maximizers as the number of evaluations increases. LIPO, as well as ADALIPO that we will see in the next subsection, are consistent over Lipschitz functions, with a convergence rate of at least $\mathcal{O}\big(N^{-1/d}\big)$ (with high probability), where N is the number of actual function evaluations performed. An example of the upper bound is given in Figure A.2.4b. In a nutshell, LIPO maximizes V while minimizing the number of function evaluations by evaluating V only at candidates that are potential maximizers.

2.1.2 Adaptative LIPO

While LIPO is simple and efficient, the Lipschitz constant of V needs to be known. This information is not available in general. As a response to that issue, the ADALIPO variant (Malherbe & Vayatis, 2017a) alternates stochastically between two states: a



(a) A Lipschitz function (b) LIPO's upper bound

Figure A.2.4. ***a)*** *An example of a Lipschitz function ($V \overset{\text{def}}{=} x \mapsto 0.5 \sin x$). The hourglass-shaped double cones are generated using a slope of $\pm\kappa$. As this function is 0.5-Lipschitz, the graph of the function is always outside of cones generated by a $\kappa \geq 0.5$. The bigger κ, the thinner are the cones.* ***b)*** *Example of an upper bound constructed by LIPO, given the potential maximizers. $\Omega_{\kappa,n}$ (thick segments) is the regions of the domain where the potential maximizers can be found at time n, i.e., where the upper bound exceeds the best value found so far (dashed line), and therefore the regions in which V is worth being probed. The upper bound (blue) is a piecewise linear function with slope coefficient in $\{-\kappa, \kappa\}$ that passes through each potential maximizer.*

state of exploration, where all candidates get accepted, and a state of exploitation, where an estimation of the Lipschitz constant is used to reject candidates with the LIPO upper bound of Eq. 12. The Lipschitz constant at time n is estimated using the following formula

$$\hat{\kappa}_n \overset{\text{def}}{=} \inf\left\{ \kappa_{i\in\mathbb{Z}} \;\middle|\; \max_{k\neq j} \frac{|V(X_k) - V(X_j)|}{\|X_k - X_j\|_2} \leq \kappa_i \right\},$$

where $(\kappa_i)_{i\in\mathbb{Z}}$ can be any real-valued sequence. The authors propose to use the sequence $\kappa_i = (1+\alpha)^i$. Thus, one can compute the closed-form expression of $\hat{\kappa}_n$

$$\hat{\kappa}_n = (1+\alpha)^{i_n},$$
$$\text{where } i_n = \left\lceil \ln\left(\max_{k\neq j} \frac{|V(X_k) - V(X_j)|}{\|X_k - X_j\|_2} \right) \frac{1}{\ln(1+\alpha)} \right\rceil.$$

This computation ensures that κ is not overestimated. Indeed, for any sequence $(\kappa_i)_{i\in\mathbb{Z}}$, if there is i such that $\kappa_i \leq \kappa \leq \kappa_{i+1}$, then, at any time n, it holds $\hat{\kappa}_n \leq \kappa_{i+1}$ (see Malherbe & Vayatis (2017a)). At each iteration, ADALIPO enters the exploration state with probability $p \in (0,1)$, which is a fixed hyperparameter of the algorithm. By accepting any uniformly sampled candidates, the exploration state prevents the estimation of κ from being locally biased within the subspace of potential maximizers. This behavior allows the method to be consistent over Lipschitz functions.

The decision rules of ADALIPO are given by

$$D_n^{\text{ADALIPO}}(\mathbf{C}_n, V(\mathbf{C}_n), x) = \max(D_n^{\text{LIPO}}(\mathbf{C}_n, V(\mathbf{C}_n), x), p),$$

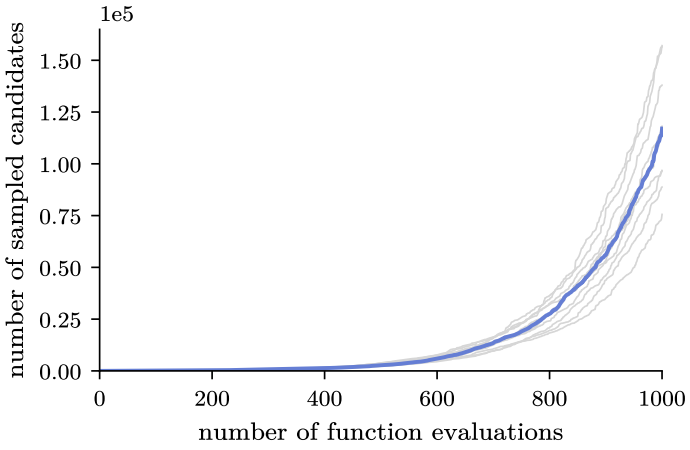


Figure A.2.5. *Number of sampled candidates required by LIPO vs number of potential maximizers (i.e. evaluations of V on the x-axis) for the Rastrigin function: median over* 10 *runs (blue) and individual runs (gray). The number of function evaluations corresponds to the number of candidates that satisfy the inequality in Eq. 12. It is evident that more and more candidates need to be sampled for finding the n-th potential maximizer as time passes.*

where D_n^{LIPO} is the decision rule of LIPO with κ replaced by $\hat{\kappa}_n$. Indeed, if a candidate point is not a potential maximizer according to the LIPO decision rule, it can still be accepted with probability p to ensure exploration. In addition, the decision rule of ADALIPO is valid as, for any global maximizer $x^\star$ and any sequence of points C_n, it holds $D_n^{\mathsf{LIPO}}(\mathsf{C}_n, V(\mathsf{C}_n), x^\star) = 1$ and therefore $D_n^{\mathsf{ADALIPO}}(\mathsf{C}_n, V(\mathsf{C}_n), x^\star) = 1$.

2.1.3 Limitations of LIPO

The main drawback of the LIPO's approach is that the number of uniform samples required to find a potential maximizer grows exponentially with time. Indeed, when the region of potential maximizers $\Omega_{\kappa,n}$ (see Figure A.2.4b) gets to be small, the algorithm needs a long time before it finds a candidate in that region. Figure A.2.5 demonstrates this phenomenon.

2.2 LIPO+ and AdaLIPO+

To address the issue presented in Section A.2.1.3, we present LIPO+ and ADALIPO+, improved versions of the aforementioned algorithms, aiming at a better empirical performance (Serré et al., 2024). The first point of improvement is a stopping criterion, motivated by the discussion around Figure A.2.5, that allows the algorithm to stop when the number of samples required to find a potential maximizer is growing exponentially. The second point of improvement concerns ADALIPO, where we introduce a decaying probability of entering the exploration state. This allows a faster convergence while restricting the approximation of the Lipschitz constant κ to the region of interest.

2.2.1 Stopping criterion

As illustrated in Figure A.2.5, the number of samples required to find a potential maximizer seems to grow exponentially whenever the region of potential maximizers

is small: the uniform random sampling has a low probability of finding a candidate in that region. This effect is unavoidable and will eventually happen over time, but the number of previous potential maximizers needed for reaching this state is unknown and depends on both the function and its domain. As it is hard to select a fixed number of evaluations as a stopping criterion, we propose to stop the algorithm whenever the slope of the function represented in Figure A.2.5 exceeds a given threshold Δ. The slope is computed over a window of size w. A bigger Δ or w allows the algorithm to run longer, reaching to more precise final approximation. We denote by LIPO+ the version of LIPO with this stopping criterion. The decision rules of LIPO+ are the same as LIPO, but the algorithm is stopped whenever the slope of the curve in Figure A.2.5 over w exceeds Δ.

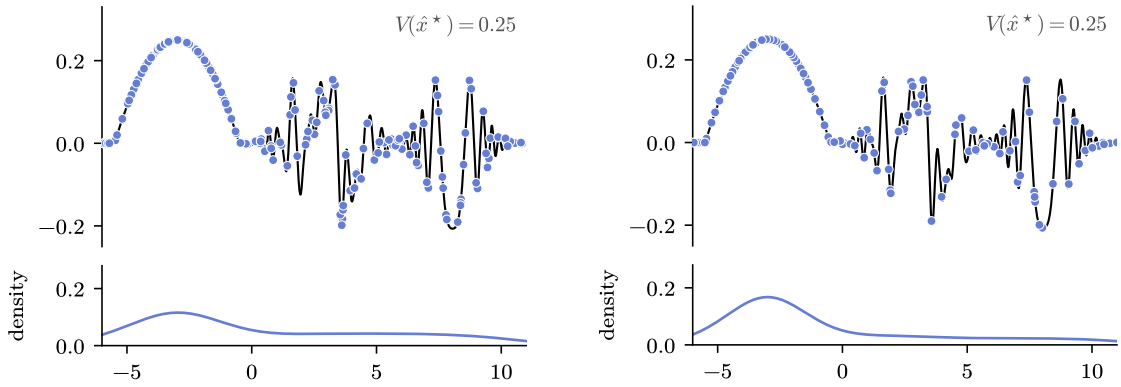
It is worth noting that this stopping criterion is not specific to LIPO and can be applied to **any decision-based algorithm**. To incorporate this stopping criterion into the abstract algorithm presented in Algorithm A.2.1, one can simply add the possibility to stop the algorithm whenever some condition depending on the history of the optimization process is satisfied. In the case of LIPO+, this condition is the one described above.

2.2.2 Decaying exploration rate

As explained is Section A.2.1.2, the transition between the exploration and the exploitation state is controlled by a Bernoulli random variable of fixed parameter p: $Y \sim \mathcal{B}(p)$. By intuition and experience, we can assume that exploring a lot at the beginning of the process is a good way to approximate correctly the Lipschitz constant κ first, and favor more and more the exploitation state as the number of evaluations increases. We propose to take $p(n) = \min\left(1, \frac{1}{\ln(n+1)}\right)$, with the convention that $p(0) = 1$. This new exploration-exploitation trade-off changes the way the Lipschitz constant is estimated and might break some of the theoretical guarantees of ADALIPO. Indeed, as the iterations increase, the probability of entering the exploration state decreases, and the estimation of κ will be restricted to the region of potential maximizers, and hence ADALIPO+ might highly underestimate the global κ. However, as the relative complement of this region w.r.t. to Ω is ignored by the algorithm, the estimation is likely to be optimal over the current region of potential maximizers. Informally, this means that if a function is globally κ_1-Lipschitz but locally κ_2-Lipschitz around the global maximizer, with $\kappa_2 < \kappa_1$, then ADALIPO+ will be able to leverage the local Lipschitz constant κ_2 to construct a tighter upper bound and therefore to converge faster than ADALIPO. An illustration of this intuition is given in Figure A.2.6. We denote by ADALIPO+ the version of ADALIPO with this decaying exploration rate and the stopping criterion described in the previous section. The decision rules of ADALIPO+ are given by

$$D_n^{\text{ADALIPO+}}(\mathbf{C}_n, V(\mathbf{C}_n), x) = \max(D_n^{\text{LIPO}}(\mathbf{C}_n, V(\mathbf{C}_n), x), p(n)),$$

where D_n^{LIPO} is the decision rule of LIPO with κ replaced by $\hat{\kappa}_n$.



(a) ADALIPO: $\hat{\kappa} = 1.72$ (b) ADALIPO+: $\hat{\kappa} = 1.5$

Figure A.2.6. *Example of the difference of the Lipschitz constant estimation between ADALIPO and ADALIPO+. One can see that, while ADALIPO+ provides an approximation of the same quality as ADALIPO, it has a lower Lipschitz constant estimation. It is due to the fact that the samples used for the estimation are more concentrated in the parabola-shaped region around the global maximizer, which has a local Lipschitz constant smaller than the global one. The evaluated points are shown in blue, with their density at the bottom. The $\hat{\kappa}$ values are averaged over 100 runs of each algorithm on the same function while the figures are generated from a single run.*

2.2.3 Numerical experiments

We compare the vanilla LIPO algorithms to ours, LIPO+ and ADALIPO+. We set $\Delta = 600$ and $w = 5$. We use standard benchmark functions used in global optimization literature (Surjanovic & Bingham, 2024). Some have few local minima (e.g. the Sphere function) while others have many (e.g. the Rastrigin function). See Figure A.2.26 for an visalization of the landscapes of these functions. We set the dimension to $d = 2$ for all functions. We run each algorithm 100 times on each function and report the results in Table A.2.1. The budget (maximum number of evaluations) depends on the function as some are easier to optimize than others. LIPO and ADALIPO exhaust the entire budget, while LIPO+ and ADALIPO+ may terminate earlier if the stopping criterion is met. One can see that, with significantly less evaluations, LIPO+ and ADALIPO+ compete with the original version.

In addition, we consider the ADALIPO+|ns variant of ADALIPO+ without using the stopping criterion defined in Section A.2.2.1. Same as the algorithms of the LIPO family, ADALIPO+|ns stops when the budget is exhausted. To compare the performance of ADALIPO+|ns to the original algorithms, we give them an infinite budget and stop them whenever the following condition is met: $g(\theta) \leq \max_{1 \leq i < t} V(X_i)$, where $\theta \in [0, 1]$ a chosen threshold and

$$g(\theta) = \max_{x \in \Omega} V(x) - \left(\max_{x \in \Omega} V(x) - \int_\Omega \frac{V(x)}{\lambda(\Omega)} \, \mathrm{d}x \right) (1 - \theta), \tag{13}$$

where λ is the standard Lebesgue measure. This condition allows us to stop the algorithm whenever we consider it has reached close enough to the true maximum. The distance required is controlled by θ: the closer to 1, the smaller the distance. We set $\theta = 0.99$ for all the functions. The results are recorded in Table A.2.2. As one

Table A.2.1. *Empirical results comparing the original algorithms and the proposed improved versions. "# evals" is the number of function evaluations (mean and standard deviation), and $d_{\max}$ is the distance from the real maximum. One can see that, with significantly less evaluations, LIPO+ and ADALIPO+ compete with the original version in terms of distance to the real maximum.*

	Hölder		Rastrigin		Sphere	
	# evals	$d_{\max}$	# evals	$d_{\max}$	# evals	$d_{\max}$
LIPO	2000	**0.0018**	1000	**0.0512**	25	0.0306
	(104)		(34)			
LIPO+	1505	**0.0018**	869	0.1282	25	0.0320
ADALIPO	2000	0.003	1000	0.4106	25	0.0227
ADALIPO+	**719**	0.023	**753**	0.0569	**20**	**0.0063**
	(457)		(133)		(5)	

Table A.2.2. *Empirical performance of our ADALIPO+|ns variant. The table shows the total number of function evaluations (mean and standard deviation) required to meet the condition stated in Eq. 13. ADALIPO+|ns outperforms LIPO and ADALIPO in almost all problems.*

	Himmelblau	Hölder	Rastrigin	Rosenbrock	Sphere	Square
LIPO	100	508	670	**11**	46	**43**
	(86)	(217)	(183)	(10)	(10)	(22)
ADALIPO	97	319	913	12	28	62
	(77)	(201)	(297)	(11)	(8)	(47)
ADALIPO+|ns	**65**	**228**	**616**	**11**	**22**	51
	(46)	(136)	(187)	(10)	(6)	(36)

can see, ADALIPO+|ns significantly outperforms the original on this benchmark. It even succeeds to beat LIPO on almost every problem, while knowing less information on the function. It corroborates the fact our decaying exploration rate is a good strategy to improve the empirical performance of ADALIPO.

2.2.4 Limitations

A limitation of LIPO+ and ADALIPO+ is that the original results of consistency provided in Malherbe & Vayatis (2017a) concerning the vanilla versions are not directly applicable. We do not provide any theoretical guarantees on the convergence of our methods. Since the consistency of the original algorithms is one of their key features of the LIPO family, this is a major drawback for the proposed improvements. Another limitation related to the curse of dimensionality is inherited from the vanilla approach. As the dimension increases, the volume of the region of potential maximizers increases exponentially and thus, the probability of rejecting a point sampled uniformly in the domain decreases accordingly. We provide the following upper bound for LIPO, which also holds for the other algorithms presented in this section as they are all based on the same principle of rejecting candidates outside of the region of potential maximizers.

Theorem A.2.1 – *LIPO's rejection probability*

For any κ-Lipschitz function f, let $(X_i)_{i\in[n]}$ be the previous potential maximizers of LIPO at time n. For any $x \in \Omega$, let $R(x,n)$ be the event of rejecting x at time n, i.e.

$$R(x, n+1) = \min_{i\in[n]} f(X_i) + \kappa \left\|x - X_i\right\|_2 < \max_{i\in[n]} f(X_i).$$

Then, the probability of rejecting a point sampled uniformly in the domain is upper bounded as follows

$$\mathbb{P}(R(x, n+1)) \leq \frac{n\pi^{d/2}\Delta^d}{\kappa^d \Gamma(d/2+1)\lambda(\Omega)},$$

where $\mathbb{P} = \frac{\lambda}{\lambda(\Omega)}$, $\Delta \stackrel{\text{def}}{=} \max_{x\in\Omega} f(x) - \min_{x\in\Omega} f(x)$ and Γ is the standard Gamma function.

Proof. At time $n+1$, a candidate point x is rejected iff it belongs to a specific ball within Ω

$$\begin{aligned}
&\min_{i\in[n]} V(X_i) + \kappa \left\|x - X_i\right\|_2 < \max_{i\in[n]} V(X_i) \\
\iff &\kappa \left\|x - X_j\right\|_2 < \max_{i\in[n]} V(X_i) - V\big(X_j\big), \text{ where } j \in \operatorname*{argmin}_{i\in[n]} V(X_i) + \kappa \left\|x - X_i\right\|_2 \\
\iff &x \in B\left(X_j, \frac{\max_{i\in[n]} V(X_i) - V\big(X_j\big)}{\kappa}\right) \bigcap \Omega \\
&\subseteq B\left(X_j, \frac{\max_{i\in[n]} V(X_i) - V\big(X_j\big)}{\kappa}\right).
\end{aligned}$$

Therefore, we have the following set inclusion

$$\begin{aligned}
\{x \mid R(x, n+1)\} &\subseteq \bigcup_{j\in[n]} B\left(X_j, \frac{\max_{i\in[n]} V(X_i) - V\big(X_j\big)}{\kappa}\right) \\
&\subseteq \bigcup_{j\in[n]} B\Big(X_j, \frac{\Delta}{\kappa}\Big),
\end{aligned}$$

which finally leads to the expected result

$$\begin{aligned}
\mathbb{P}(\{x \mid R(x, n+1)\}) &\leq \mathbb{P}\left(\bigcup_{j\in[n]} B\Big(X_j, \frac{\Delta}{\kappa}\Big)\right) \\
&\leq \frac{n\pi^{d/2}\Delta^d}{\kappa^d \Gamma(d/2+1)\lambda(\Omega)}.
\end{aligned}$$

□

2.3 RankOpt

In the following sections, we present two decision-based algorithms introduced in Malherbe et al. (2016). One of the goals of this thesis was to extend the ideas from that paper, in particular by providing a complete and efficient implementation of these algorithms. Our contributions are fourfold: we provide a corrected and generalized statement and proof of Malherbe et al. (2016, Proposition 2), a key result motivating the construction of the algorithms; we give a complete description of the algorithms, highlighting the external techniques required for an efficient implementation; we provide a C++/Python implementation (see Chapter A.4); and we reproduce the numerical experiments of Malherbe et al. (2016) using our implementation. This work has led to the preprint Malherbe et al. (2026), to be submitted to a peer-reviewed journal. For clarity and completeness, we present both the algorithms and the numerical experiments in this section, while deferring the theoretical analysis to Malherbe et al. (2026).

In the original paper, Malherbe et al. (2016) draw on concepts from ranking theory — specifically the estimation of level sets (Clémençon et al., 2008) — to develop global optimization algorithms that do not rely on any smoothness assumption on the objective function. The underlying intuition is as follows: even when the objective function exhibits arbitrarily large variations, much of the information needed to locate its optimum may be encoded in its induced ranking rule, that is, the way in which its level sets are nested within one another. Building on this observation, the authors introduce a decision-based optimization framework in which the complexity of the function is captured by the pairwise ranking it defines, and establish theoretical guarantees in terms of statistical consistency and convergence to the optimum. In this thesis, we propose a detailed implementation of these algorithms and demonstrate that they achieve strong performance on classical global optimization benchmarks.

2.3.1 The ranking structure of a real-valued function

In this section, we introduce the ranking structure as a characterization of the complexity for a general real-valued function to be optimized. First, we observe that real-valued functions induce an order relation over the input space Ω, and the underlying ordering induces a ranking rule which records pairwise comparisons between evaluation points.

Definition A.2.3 – *Induced ranking rule*

The ranking rule $r_f : \Omega^2 \to \{-1, 0, 1\}$ induced by a function $f : \Omega \to \mathbb{R}$ is defined for all pairs $(x, x') \in \Omega^2$ as

$$r_f(x, x') = \operatorname{sgn}(f(x) - f(x')) = \begin{cases} +1 & \text{if } f(x) > f(x') \\ 0 & \text{if } f(x) = f(x'). \\ -1 & \text{if } f(x) < f(x') \end{cases}$$

In addition, a ranking structure is a collection of ranking rules.

The key argument of this work is that the optimization of any weakly regular real-valued function only depends on the nested structure of its level sets. Hence there is an equivalence class of real-valued functions that share the same induced ranking rule as shown by the following theorem.

Theorem A.2.2 – *Ranking rule equivalence class*

Let Ω by an arbitrary space and $h : \Omega \to \mathbb{R}$ an arbitrary function. Then, a function $f : \Omega \to \mathbb{R}$ shares the same induced ranking rule with h (i.e. $\forall (x, x') \in \Omega^2, r_f(x, x') = r_h(x, x')$) if and only if there exists a function $\psi : \mathbb{R} \to \mathbb{R}$, strictly increasing on $\mathrm{Im}(f)$ such that $h = \psi \circ f$.

A contribution of this thesis has been to provide a corrected and generalized statement and proof of this theorem, as well as a LEAN formalization of this key result[3]. For more details on LEAN and formal methods in general, see Chapter B.2.

Proof.

($\Longleftarrow$) Assume that there exists a function $\psi : \mathbb{R} \to \mathbb{R}$, strictly increasing on the image of f such that $h = \psi \circ f$. Since ψ is strictly increasing on $\mathrm{Im}(f)$, the result is straightforward as ψ preserves the order of the values of f

$$\forall (x, x') \in \Omega^2, f(x) < f(x') \Rightarrow \psi(f(x)) < \psi(f(x')) \Rightarrow h(x) < h(x').$$

($\Longrightarrow$) Assume now that $r_f = r_h$. Let f^{-1} be the generalized inverse of f ($f^{-1}(y) = \{x \mid f(x) = y\}$) and let ψ be defined as

$$\psi(y) = h \circ f^{-1}(y) = \begin{cases} h(x) & \text{if } \ y \in \mathrm{Im}(f) \text{ , where } x \in f^{-1}(y) \\ \text{any } z \in \mathbb{R} \text{ otherwise.} \end{cases}.$$

We start to show that ψ is strictly increasing on $\mathrm{Im}(f)$. Given $x_1 < x_2$ in $\mathrm{Im}(f)$, we need to show that $h(z_1) < h(z_2)$ where $z_1 \in f^{-1}(x_1)$ and $z_2 \in f^{-1}(x_2)$. As $r_f = r_h$, it is enough to show that $f(z_1) < f(z_2)$, which is true by definition of f^{-1}.

Finally, for any $x \in \Omega$, we have that $f(x) \in \mathrm{Im}(f)$ and therefore $\psi(f(x)) = h \circ f^{-1} \circ f(x) = h(x)$. Therefore, $h = \psi \circ f$.

□

This theorem states that even if the unknown function V admits noncontinuous or large variations, up to a transformation ψ, there might exist a simpler function $h = \psi \circ f$ that shares the same induced ranking rule. Figure A.2.7 gives an example of three functions that share the same ranking rule while they display different regularity properties.

The approach developed in this work consists of seeing the ranking structure as a characterization of the complexity of the target function V through the complexity of its induced ranking rule.

[3] github.com/gaetanserre/LeanRankOpt

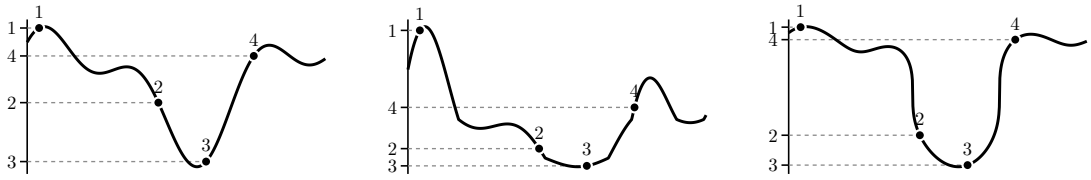


Figure A.2.7. *Three functions sharing the same ranking rule: although their regularity differs, any set of points is ranked identically.*

We first introduce a very large class of ranking rules.

Definition A.2.4 – *Continuous ranking structure*

We say that a real-valued function $f : \Omega \to \mathbb{R}$ has a continuous ranking rule if $r_f \in \mathcal{R}_\infty$ where $\mathcal{R}_\infty := \{r_h \mid h \in \mathcal{C}^0(\Omega, \mathbb{R})\}$ denotes the set of continuous ranking rules (e.g. the set of ranking rules induced by continuous functions).

In the continuation of this definition, we further introduce the polynomial ranking structure.

Definition A.2.5 – *Polynomial ranking structure*

The set of polynomial ranking rules of degree $k \geq 1$ is defined as

$$\mathcal{R}_{\Omega_k[X]} \stackrel{\text{def}}{=} \{r_h : (x, x') \mapsto \operatorname{sgn}(h(x) - h(x')) \mid h \in \Omega_k[X]\}.$$

It is noteworthy that even a polynomial function of degree $k > 1$ may admit a lower degree polynomial ranking rule. For example, consider the polynomial function $f(x) = (x^2 - 3x + 1)^9$. Since $f(x) = \psi(x^2 - 3x)$ where $\psi : x \mapsto (x+1)^9$ is a strictly increasing function, the ranking rule induced by f is a polynomial ranking rule of (at most) degree 2.

In Malherbe et al. (2026), more ranking structures are introduced, such as the convex ranking structure or the sinusoidal polynomial ranking ranking structure. The interested reader is referred to the original paper for more details.

2.3.2 The RankOpt algorithm

In order to properly set up the algorithm, we first introduce some key concepts that will be at the core of its strategy and used throughout the paper. We start with the definition of the empirical ranking loss.

Definition A.2.6 – *Empirical ranking loss*

The empirical ranking loss of a ranking rule $r : \Omega \times \Omega \to \{-1, 0, 1\}$ over a sample $(\mathbf{C}_n, V(\mathbf{C}_n)$ of $n \geq 1$ function evaluations is defined with respect to the induced rule $r_V(X_i, X_j) = \operatorname{sgn}\big(V(X_i) - V(X_j)\big)$, for all $(i, j) \in [n]^2$, as

$$L_n(r) \stackrel{\text{def}}{=} \frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} \mathbb{1}\big(r(X_i, X_j) \neq r_V(X_i, X_j)\big),$$

Based on this definition, one might then recover among a ranking structure $\mathcal{R}$ the subset of ranking rules r which are consistent with the ranking rule r_V induced by the objective function over a sample of function evaluations.

Definition A.2.7 – *Active subset of consistent ranking rules*

The active subset of a ranking structure $\mathcal{R}$ containing the ranking rules that are consistent with r_V over a sample $(\mathbf{C}_n, V(\mathbf{C}_n)$ of $n \geq 1$ function evaluations, is defined by

$$\mathcal{R}_n := \{r \in \mathcal{R} : L_n(r) = 0\}.$$

The RANKOPT algorithm can be stated as a decision-based optimization algorithm that iteratively samples candidate points and accepts or rejects them based on a set of ranking rules, consistent with the ranking rule induced by the objective function V over a sample of function evaluations. Given a ranking structure $\mathcal{R} \subseteq \mathcal{R}_\infty$ such that $r_V \in \mathcal{R}$, the decision rules D_n^{RANKOPT} are defined as follows

$$D_n^{\text{RANKOPT}}(\mathbf{C}_n, V(\mathbf{C}_n), x) = \begin{cases} 1 \text{ if } \exists r \in \mathcal{R}_n, r\left(x, X_{\hat{\imath}_n}\right) \geq 0 \\ 0 \text{ otherwise} \end{cases}, \tag{14}$$

where $\hat{\imath}_n \in \text{argmax}_{0 \leq i \leq n} V(X_i)$. The decision rules involve the active subset $\mathcal{R}_n$, which contains the ranking rules that are consistent with the ranking rule induced by V over the points sampled so far. More precisely the decision rules operate as follows: if there exists no ranking rule $r \in \mathcal{R}_n$ satisfying $r\left(X_{n+1}, X_{\hat{\imath}_n}\right) \geq 0$, then we know from the definition of $\mathcal{R}_n$ that $r_V\left(X_{n+1}, X_{\hat{\imath}_n}\right) = -1$, which necessarily means that $V(X_{n+1}) < V\left(X_{\hat{\imath}_n}\right)$. Thus, the algorithm evaluates the function at a point only if it would certainly return an evaluation at least equal to the highest evaluation $V\left(X_{\hat{\imath}_n}\right)$ observed so far.

Under some mild assumptions on the objective function V and Ω, one can show that the RANKOPT algorithm asymptotically converges to the global optimum of V with probability one. Then, under additional regularity assumptions on the level sets of V, one can derive finite-time upper and lower bounds on the convergence rate of the algorithm. The interested reader is referred to Malherbe et al. (2026) for more details.

2.3.3 Adaptive RankOpt

The ADARANKOPT algorithm extends RANKOPT to the setting where no prior information is available on the ranking rule induced by the objective function f_{opt}, by incorporating a model selection step. Its development closely parallels that of the adaptive extension of the LIPO algorithm presented in Section A.2.1.2. The core idea is to consider a nested sequence of ranking structures $\mathcal{R}_1 \subseteq \mathcal{R}_2 \subseteq \cdots \subseteq \mathcal{R}_\infty$, starting from the simplest one and increasing complexity only when the current structure proves inconsistent with the ranking rule induced by f_{opt} on the observed function evaluations. More precisely, at each iteration n, a ranking structure $\mathcal{R}_{\text{ada}}$ is selected by minimizing the empirical ranking loss $L_n(r)$

$$\mathcal{R}_{\text{ADA}} \overset{\text{def}}{=} \mathcal{R}_{\hat{k}}, \text{ where } \hat{k} \overset{\text{def}}{=} \min\left\{k \in \mathbb{N}^* \;\middle|\; \min_{r \in \mathcal{R}_k} L_n(r) = 0\right\}. \tag{15}$$

It also uses the same exploration/exploitation trade-off as ADALIPO, which is controlled by a parameter $p \in (0,1)$. The decision rules $D_n^{\text{ADARANKOPT}}$ are defined as follows

$$D_n^{\text{ADARANKOPT}}(\mathbf{C}_n, V(\mathbf{C}_n), x) = \max(D_n^{\text{RANKOPT}}(\mathbf{C}_n, V(\mathbf{C}_n), x), p),$$

where the ranking structure $\mathcal{R}_n$ in D_n^{RANKOPT} is replaced by the selected ranking structure $\mathcal{R}_{\text{ADA}}$.

Under the same assumptions as RANKOPT, one can show that the ADARANKOPT algorithm asymptotically converges to the global optimum of V with probability one, no matter the nested sequence of ranking structures considered and the value of the exploration parameter p. Then, under additional assumptions on the level sets of V and on the sequence of ranking structures, one can derive a finite-time upper bound on the convergence rate of the algorithm. The interested reader is referred to Malherbe et al. (2026) for more details.

2.3.4 Implementation and computational aspects

Implementing the ADARANKOPT algorithm raises a practical challenge: one needs an efficient way to compute the empirical ranking loss $L_n(r)$ for any ranking rule r belonging to a given ranking structure $\mathcal{R}_k$. A natural approach is to rely on the nested sequence of polynomial ranking structures (see Definition A.2.5), where $\mathcal{R}_k = \mathcal{R}_{\Omega_k[X]}$ denotes the set of polynomial ranking rules of degree k. A complete implementation of the ADARANKOPT algorithm can then be derived from Corollary 32 of Malherbe et al. (2026).

Corollary A.2.1 – *Malherbe et al. (2026, Corollary 32)*

Let $(\mathbf{C}_n, V(\mathbf{C}_n))$ be any sample of $n+1$ function evaluations with distinct values. Then, there exists a polynomial ranking rule of degree $k \geq 1$ that perfectly ranks the sample if and only if the polyhedral set $\mathcal{K}_n^{\Phi_k}$ is empty; $\mathcal{K}_n^{\Phi_k}$ is defined by

$$\mathcal{K}_n^{\Phi_k} \overset{\text{def}}{=} \left\{\lambda \in \mathbb{R}^n \;\middle|\; M_n^{\Phi_k}\lambda^\top = \mathbf{0} \wedge \langle \mathbf{1}, \lambda\rangle = 1 \wedge \forall i, \lambda_i \geq 0\right\},$$

where $\Phi_k : \mathbb{R}^d \to \mathbb{R}^{\binom{k+d}{d}-1}$ is the function that maps $\mathbb{R}^d$ into the corresponding polynomial feature space of degree k, and $M_n^{\Phi_k} = [C_1, \cdots, C_n]$ is the $(\dim(\Phi_k), n)$ -matrix with its i-th column C_i equal to $\left(\Phi_k\left(X_{(i)}\right) - \Phi_k\left(X_{(i-1)}\right)\right)^\top$ where $X_{(i)}$ is the i-th point in the sample sorted according to its evaluation value, i.e. $V\left(X_{(0)}\right) < \ldots < V\left(X_{(n)}\right)$.

The implementation of ADARANKOPT using the nested sequence of polynomial ranking structures is straightforward. At each iteration n, the algorithm computes $\hat{k}_n$, the least degree such that $\mathcal{K}_n^{\Phi_k}$ is empty given the current sample of function evaluations. Then, for any candidate point x, it computes the polyhedral set $\mathcal{K}_n^{\Phi_{\hat{k}_n}}$ using the sample $(\mathbf{C}_n, V(\mathbf{C}_n)) \cup \left(x, \max_{0 \leq i \leq n} V(X_i) + c\right)$, where $c > 0$ is a small

constant. If $\mathcal{K}_n^{\Phi_{k_n}}$ is empty, then the candidate point x is accepted; otherwise, it is rejected with probability $1-p$.

2.3.5 Numerical experiments

In this section, we compare the empirical performance of ADARANKOPT with polynomial ranking structures to existing state-of-the-art global optimization methods on real and synthetic problems.

Algorithms. The ADARANKOPT algorithm was compared with seven different types of algorithms, developed from various approaches of global optimization:

- ADALIPO+ see Section A.2.2. To improve the performance, we combined ADALIPO+ with a trust region method, leading to ADALIPO+TR (see Section A.2.4).
- ECP (Fourati et al., 2025) is a method based on ADALIPO and ADALIPO+. The difference resides in the way of estimating the Lipschitz constant. The authors chose to enlarge the region of acceptance of the candidates (which is equivalent to enlarging the estimated Lipschitz constant) whenever the algorithm struggles to find a new promising candidate. To improve the performance, we combined ECP with a trust region method, leading to ECP+TR (see Section A.2.4).
- CMA-ES see Section A.1.1
- CRS is a variant of the Controlled Random Search of which includes local mutations. It starts with a random population and randomly evolves these points by an heuristic rule.
- DIRECT (Jones, 2001) is a Lipschitz optimization algorithm where the Lipschitz constant is unknown. It uses a deterministic splitting technique of the search space and it is therefore the only purely deterministic algorithm of the benchmark.
- MLSL (Rinnooy Kan et al., 1985) is a multistart algorithm. It performs a series of local optimizations starting from points randomly chosen by a clustering heuristic that helps to avoid repeated searches of the same local optima.
- PRS (Pure Random Search) is the standard random space-filling method which consists in evaluation the function over a sequence of points uniformly and independently sampled over the input space.

For a fair comparison, the tuning parameters of the algorithms where all set to default and the ADARANKOPT algorithm was used in all the experiments with a parameter p fixed to 1/10. To check the emptyness of the polyhedral set $\mathcal{K}_n^{\Phi_k}$, we used the GLPK library[4]. The core implementation of the ADARANKOPT algorithm was done in C++ with a Python interface, allowing for easy integration with existing Python-based optimization frameworks. The code is included in the GLOBE library (see Chapter A.4). To ensure the termination of the algorithm, we set a maximum number of sampling trials (points sampled from $\mathcal{U}(\Omega)$ and tested for the emptiness

[4] www.gnu.org/software/glpk/glpk.html

of $\mathcal{K}_n^{\Phi_{k_n}}$) and a maximum polynomial degree. Whenever the maximum number of sampling trials is reached, the algorithm stops and returns the best point found so far, while if the maximum polynomial degree is reached, the algorithm stops looking for a polynomial model that fits the previous sampled points. Finally, a trust region module was added to the ADARANKOPT algorithm, leading to ADARANKOPT+TR (see Section A.2.4).

This implementation of ADARANKOPT+TR have four new hyperparameters to tune:

i. The maximum number of sampling trials before stopping the algorithm;

ii. the maximum polynomial degree;

iii. the trust region radius;

iv. the maximum number of function evaluation given to the trust region algorithm to compute the quadratic approximation.

The impact of hyperparameters **i.** and **ii.** is rather straightforward. However, hyperparameters **iii.** and **iv.** are more subtle. Figure A.2.8 provides an illustratation of their impact on synthetic benchmark functions. One can see that, on a noisy function like the Holder Table, a small initial trust region radius and few function evaluations given to the trust region algorithm lead to better performance as the geometry of the function prevents a good quadratic approximation of the function in a large region and thus requires more domain exploration. In this other hand, on a smooth function like the Rosenbrock function, a large initial trust region radius and many function evaluations given to the trust region algorithm lead to faster convergence. However, small values of these hyperparameters still give to decent performance on smooth functions. That is why we set the default values of these hyperparameters to 0.1 and 15 and one should only consider tuning them whenever sufficient information on the function's geometry is available.

Test problems.
The benchmark includes two series of nonconvex optimization problems which involve real data sets and naturally arise in the tuning of machine learning algorithms and two series of artificial problems that are commonly met in standard global optimization benchmarks.

Ridge regression regularization. First, we studied the task of estimating the regularization $\lambda^\star$ and the bandwidth $\sigma^\star$ of a gaussian kernel ridge regression which minimize the empirical mean squared error of the predictions over a 10-fold cross validation with real data sets. The optimization was performed over $(\ln(\lambda), \ln(\sigma)) \in [-2, 4] \times [-5, 5]$ and two data sets taken from the UCI Machine Learning Repository (Asuncion et al., 2007) were employed: *Yacht Hydrodynamics* and *Abalone*. For each data set $(x_1, y_1), ..., (x_n, y_n)$, we considered the following optimization problem

$$(\ln(\lambda^\star), \ln(\sigma^\star)) \in \underset{(\ln(\lambda), \ln(\sigma)) \in \Omega}{\operatorname{argmin}} \frac{1}{10} \sum_{k=1}^{10} \frac{1}{|I_k|} \sum_{i \in I_k} \left(\hat{g}^k_{\lambda,\sigma}(x_i) - y_i \right)^2$$

where

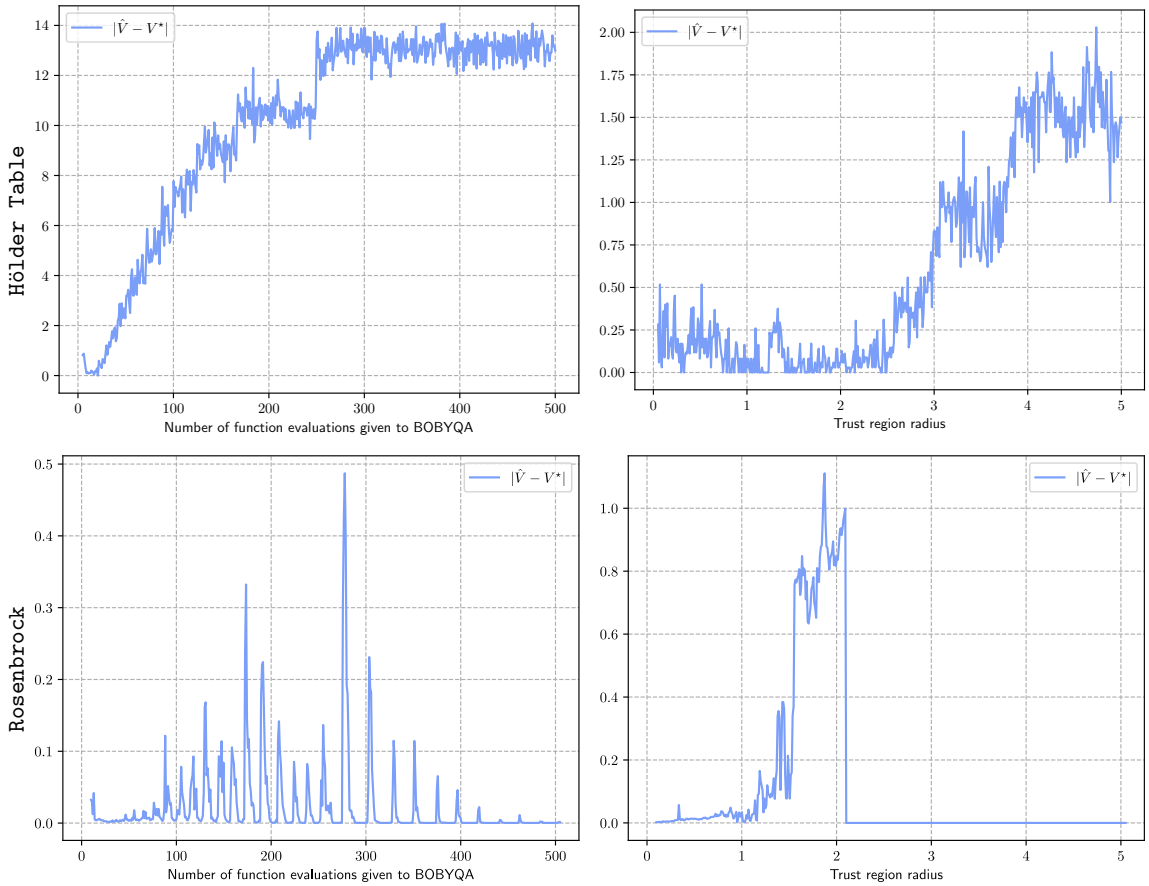


Figure A.2.8. *Impact of the hyperparameters of the trust region algorithm on the Holder Table and Rosenbrock functions.*

$$\hat{g}^k_{\lambda,\sigma} \in \underset{g \in \mathcal{H}_\sigma}{\operatorname{argmin}} \sum_{i \notin I_k} (g(x_i) - y_i)^2 + \lambda \|g\|^2_{\mathcal{H}_\sigma}$$

over $\Omega = [-2, 4] \times [-5, 5]$ and where $I_1, ..., I_{10}$ are 10 disjoint subsets of integers of similar sizes forming a partition of $\{1, ..., n\}$, $\mathcal{H}_\sigma$ is the gaussian RKHS of bandwidth σ and $\|\cdot\|_{\mathcal{H}_\sigma}$ denotes the corresponding norm.

Neural nets hyperparameter calibration. Then, we studied the task of estimating the learning rate $\eta^\star$ and the momentum $\mu^\star$ of an accelerated stochastic gradient descent which minimize the empirical classification error of the predictions of a neural net with fixed architecture tuned with the stochastic gradient descent over a 5-fold cross validation involving real data sets. The optimization was performed over $(\ln(\eta), \ln(1-\mu)) \in \left[-5, \frac{1}{2}\right] \times [-5, 0]$ and two data sets taken from the UCI Machine Learning Repository (Asuncion et al., 2007) were employed: *Wine* and *Iris*. For each data set $(x_1, y_1), ..., (x_n, y_n)$, with $x_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$, we considered the following optimization problem

$$(\ln(\eta^\star), \ln(1-\mu^\star)) \in \underset{(\ln(\eta), \ln(1-\mu)) \in \Omega}{\operatorname{argmin}} \frac{1}{5} \sum_{k=1}^{5} \frac{1}{|I_k|} \sum_{i \in I_k} \mathbb{1}\left(g^k_{\eta,\mu}(x_i) \neq y_i\right)$$

where

$$\hat{g}^k_{\eta,\mu}(x) = \mathbb{1}\left(\sigma_3\left(W_3^k \sigma_2\left(W_2^k \sigma 1\left(W_1^k x\right)\right)\right) \geq \frac{1}{2}\right)$$

over $\Omega = \left[-5, \frac{1}{2}\right] \times [-5, 0]$ and where $I_1, ..., I_5$ are 5 disjoint subsets of integers of similar sizes forming a partition of $\{1, ..., n\}$, the matrix $W_1^k \in \mathbb{R}^{d \times 12}$, $W_2^k \in \mathbb{R}^{12 \times 8}$ and $W_3^k \in \mathbb{R}^{8 \times 1}$ are learned with an accelerated stochastic gradient descent minimizing the cross-entropy binary loss tuned with the parameter (η, μ) after 80 epochs with a batch size of 64 over the data set $\{(x_i, y_i)\}_{i \notin I_k}$ and where σ_1, σ_2 and σ_3 are respectively the ReLu, ReLu and sigmoid activation functions.

Synthetic functions I. We then compared the algorithms on a series of five two-dimensional problems taken from Surjanovic & Bingham (2024): *Branin-Hoo, Himmelblau, McCormick, Levy N.13* and *Styblinski*. The dimensionality of these problems allows an easy visualization of the test functions and it can be seen that this series covers a wide variety of situations, including multimodal and non-linear functions as well as ill-conditioned and well-shaped functions.

Synthetic functions II. The last benchmark we used to assess the performance of the algorithms consists of a set of five synthetic functions with a dimensionality varying from two to seven. It contains the *Holder Table, Rosenbrock, Sphere, Linear slope* and *Deb N.1* functions take from Surjanovic & Bingham (2024). Remark that, due to the dimensionality of the input spaces, very little is known on the structure of these functions.

Protocol and performances metrics. As the computing complexity of the objective functions of the benchmark differs significantly, we set a adaptative budget of function evaluations. For synthetic problems, the budget is set to $n = 1000$. For ridge regression regularization and neural nets hyperparameter calibration, the budget is set to $n = 400$ as the evaluation of these functions requires to train a model over a cross validation. For each target parameter $t = 90\%, 95\%$ and 99%, we have collected the stopping times corresponding to the number of evaluations required by each method to reach the specified target

$$\tau(A, V, k, t) \stackrel{\text{def}}{=} \min\left\{ i \in [n] \;\middle|\; f\left(X_i^{(k)}\right) \geq V_{\text{target}}(t) \right\}$$

where $\min\{\varnothing\} = n$ by convention, $\left\{ f\left(X_i^{(k)}\right) \right\}_{i=1}^{n}$ denotes a sequence of n evaluations made by the algorithm A over the function V on the k-th run with $k \leq K$ and with a target value set to

$$V_{\text{target}}(t) \stackrel{\text{def}}{=} \max_{x \in \Omega} V(x) - \left(\max_{x \in \Omega} V(x) - \int_{x \in \Omega} V(x)\, \mathrm{d}x / \lambda(\Omega) \right) \times (1 - t).$$

This corresponds to Eq. 13, the stopping condition presented in Section A.2.2.3. The normalization of the target with regards to the average value of the function over the domain prevents the performance measures from being dependent of any constant term in the objective function. In practice, the average value was estimated from a Monte Carlo sampling of m evaluations, where $m = 10^6$ for synthetic problems and $m = 1000$ for real task problems, and the maximum of the function was estimated, for the real task problems, by taking the best value observed over all the sets of experiments. Based on these stopping times, we measured performance through a collection of indicators:

i. Proportion of runs for which the algorithm A reached the target t over the objective function V with less than the total budget of n evaluations

$$\hat{p}_K \overset{\text{def}}{=} \frac{1}{K}\sum_{k=1}^{K} \mathbb{1}(\tau(A, V, k, t) < n).$$

ii. Average and standard deviation of the number of evaluations required by the algorithm A to reach the target t over the objective function V within the total budget

$$\overline{\tau}_K(A, V, t) \overset{\text{def}}{=} \frac{1}{|I|}\sum_{k\in I} \tau(A, V, k, t) \quad \text{and} \quad \hat{\sigma}_t(A, V) \overset{\text{def}}{=} \left(\frac{1}{|I|}\sum_{k\in I} (\tau_k - \overline{\tau}_K)^2\right)^{\frac{1}{2}}$$

where $I \overset{\text{def}}{=} \{k = 1, ..., K \mid \tau(A, V, k, t) < n\}$ denotes the set of the indexes of the runs that reached the target t within the total budget.

iii. Aggregation of the proportion of runs for which an algorithm A reached the target t in terms of number $i \geq k$ of function evaluations, where $k = 100$ for neural networks hyperparameter calibration and $k = 150$ for the other problems, over a series $V_{1:N} = \{V_1, ..., V_N\}$ of N optimization problems, where $N = 5$ for synthetic problems and $N = 2$ for real task problems

$$\forall i \geq n, \hat{P}(A, V_{1:N}, t, i) = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\big(\tau\big(A, V_j, k, t\big) \geq i\big).$$

iv. Aggregation of the proportion of runs for which an algorithm A_1 has executed less (or more) evaluations to reach the target t than an algorithm A_2 over a series $V_{1:N} = \{V_1, ..., V_N\}$ of $N = 9$ optimization problems

$$W(A_1, A_2, V_{1:5}, t) := \frac{1}{N}\sum_{j=1}^{N}\frac{1}{K}\sum_{k=1}^{K} \mathbb{1}\big(1.1 \times \tau\big(A_1, V_j, t\big) < \tau\big(A_2, V_j, k, t\big)\big) - \mathbb{1}\big(1.1 \times \tau\big(A_2, V_j, k, \big) < \tau\big(A_1, V_j, k, t\big)\big)$$

Note that this indicator considers that two algorithms performed equally when they displayed the same results within a 10% threshold. Remark also that if $W = 1$, then A_1 always performed better than A_2 and if $W = -1$, then A_1 always performed worse than A_2.

Indicators **i.** and **ii.** record the performance of an algorithm over a single objective function while indicators **iii.** and **iv.** aggregate its performance over each series of five optimization problems of the benchmark (i.e. ridge regression regularization, neural nets hyperparameter tuning, Synthetic functions II and II). These indicators capture the most important properties of global optimization algorithms, such as accuracy, stability and speed of convergence.

Results and comments. Results are collected in a series of figures and tables. The proportion of runs reached the specified targets in terms of function evaluations (Indicator **iii.**) are displayed in Figure A.2.9. The number of runs for which an algorithm has executed less evaluation than another algorithm (Indicator **iv.**) are displayed in Figure A.2.10. Finally, the number of runs that reached the target

within the total budget, as well as the average number and standard deviation of iterations required to reach the target (indicators I and II) are collected in Tables A.2.3 and A.2.4. Our main observations are the following:

- As one should expect, there is no algorithm that performs uniformly better than other over each problem of the benchmark. Note that this makes the comparison difficult as one can always find a problem for which an algorithm is better than another.
- On the contrary, the PRS algorithm tends to be outperformed by any method over most problems of the benchmark (e.g. see the proportion of runs that reached a near-optimal solution in terms of function evaluations in Figure A.2.9). This simple observation suggests that using a global optimization algorithm lead generally to better results than the naive random search over a large class of problems.
- In terms of the proportion of runs that reached a near-optimal solution within the given budget, it should be noticed that most methods are able to reach the 95% target over most problems of the benchmark (see Tables A.2.3 and A.2.4) while there is no method that is able to solve more than 80% of the neural nets tuning problems with a target set to 99%. It is a difficult problem with few structure where the optimum if located in a very small portion of the space.
- In terms of speed of convergence, the DIRECT and MLSL algorithms need in average less function evaluations to reach a near-optimal solution when they converge than the CMA-ES and CRS algorithms (e.g. see Figures A.2.9 and A.2.10). Moreover, as seen on Figure A.2.9, the ADARANKOPT+TR algorithm generally ranks between these two groups.
- The ADARANKOPT+TR method displays—as one should expect—very competitive results on test problems with estimated ranking rules of moderate complexity with regards to the sequence of ranking structures set as input (e.g. see *Himmelblau* or *Styblinski*). Moreover, experiments *Linear Slope* and *Rosenbrock* also confirm that the algorithm can be robust against the dimensionality of the input space in the case of test functions with estimated ranking rule of low complexity.
- In contrast, the method stalls on test problems which do not admit an estimated ranking rule of moderate complexity (e.g. see *Deb N.1* and *Holder Table*). Indeed, the algorithm can not estimate efficiently the ranking rules induced by some classes of functions with a single sequence of ranking structures set as input. Considering at the same time multiple sequences of ranking structures might be a promising approach to address this issue, allowing the algorithm to adapt to wider varieties of shapes.
- Finally, it can be observed that in the case of test functions with strong global structure but many local optima, the algorithm reaches the 95% target with few function evaluations but fails at getting to the 99% target (e.g. see *Styblinski*). Indeed, the algorithm starts moving toward the global optima by learning the global structure of the function but then considers ranking rules of a level of

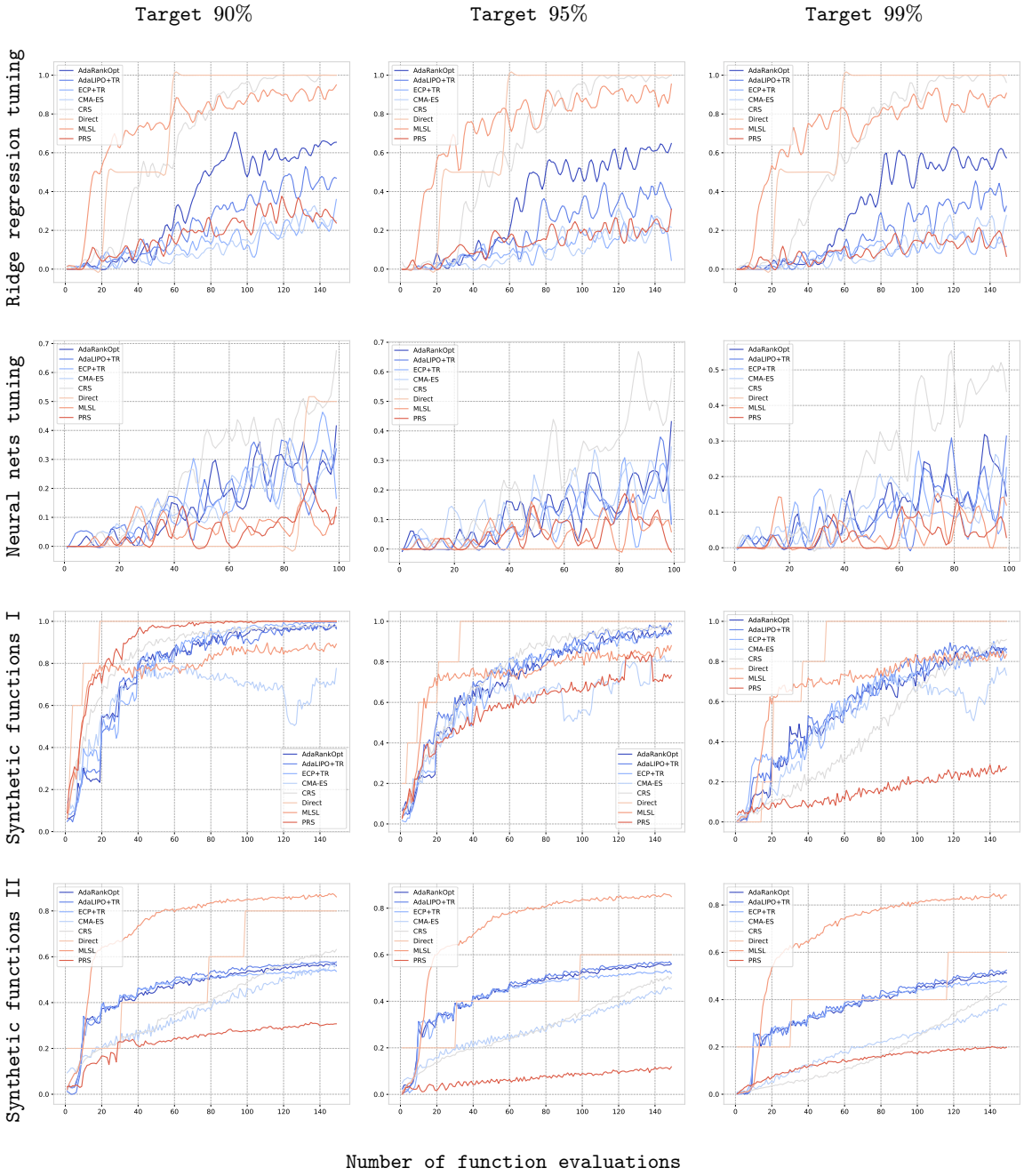


Figure A.2.9. *Aggregation of the proportion of runs that reached the targets* 90%, 95% *and* 99% *in terms of function evaluations over each series of optimization problems.*

complexity higher than required when many local variations are met. Adding a noise parameter would allow the algorithm to be more robust against this type of local perturbations. We refer the interested reader to Malherbe et al. (2026) for more details on this issue.

These empirical results aim at **(i)** providing numerical evidence that the main algorithm of the paper is competitive with the state-of-the-art methods and effective on a wide range of optimization problems and **(ii)** identifying some limits of the ranking-based approach we developed that could be solved with further extensions. However, a complete and detailed empirical analysis of the merits and limitations of the algorithm with these extensions is beyond the scope of this paper and will be carried out in future work.

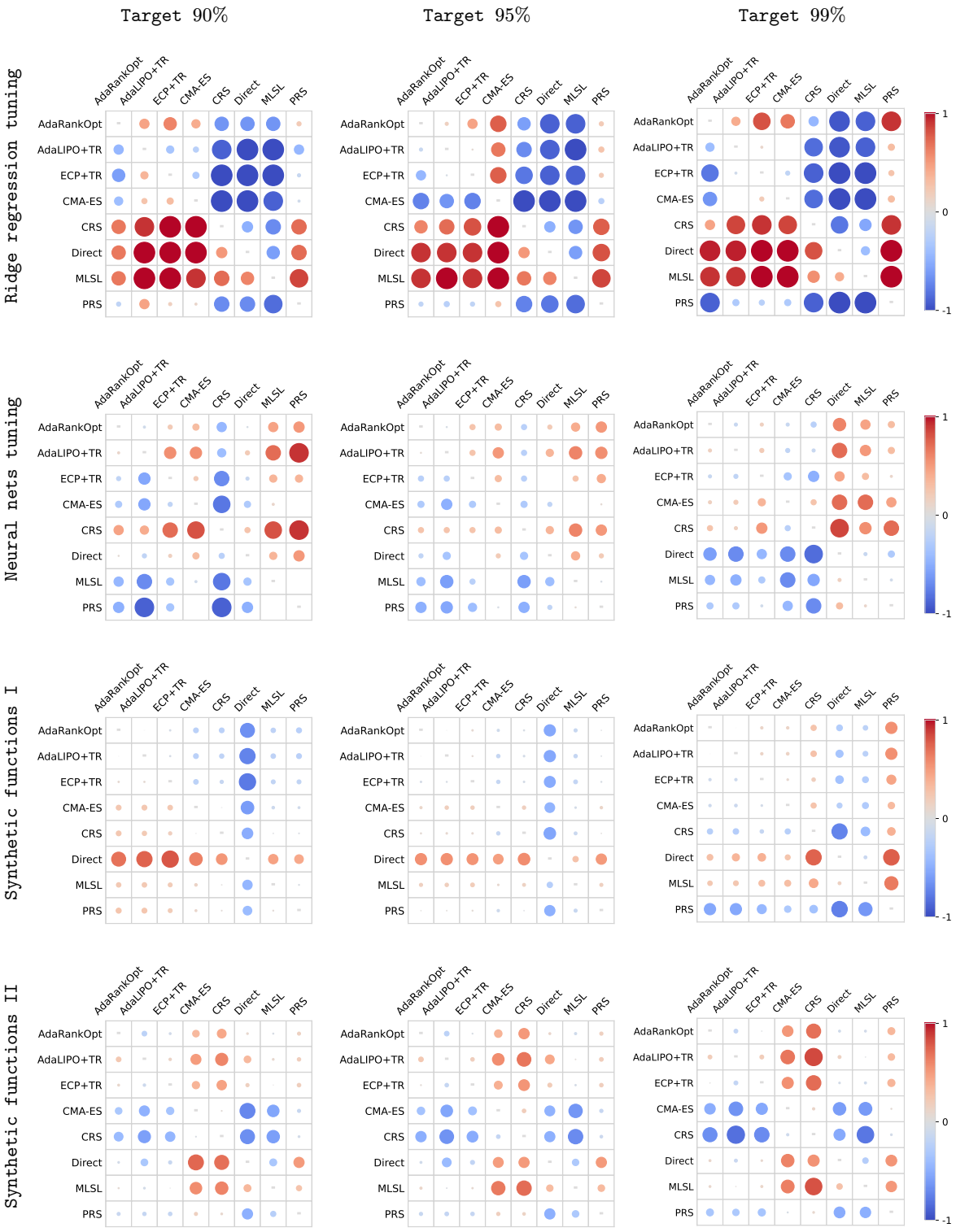


Figure A.2.10. *Aggregation of the proportion of runs for which an algorithm has executed less (or more) evaluations to reach a specified target than another algorithm over each series of optimization problems. The matrix display the indicators* $W(A_1, A_2, V_{1:5}, t)$ *where* A_1 *is located on the raw and* A_2 *on the column.*

Table A.2.3. *Results of the numerical experiments on the ridge regression and neural net tuning problems. The table displays the proportion of runs that reached the specified targets and the average and standard deviation of evaluations required to reach the target. In bold, the best result obtained in terms of average of function evaluations.*

BENCHMARKS	ADARANKOPT+TR	ECP+TR	ADALIPO+TR	CMA-ES	CRS	DIRECT	MLSL	PRS
RIDGE REGRESSION REGULARIZATION, TARGET 90%								
YACHT	70% 150 (139)	30% 46.0 (98.1)	10% 18.0 (54.0)	10% 24.6 (73.8)	100% 69.1 (18.1)	100% 59.0 (00.0)	**100% 51.5** (39.2)	0% 00.0 (00.0)
ABALON.	90% 71.0 (42.8)	80% 99.0 (67.4)	60% 98.0 (128)	50% 123 (148)	100% 30.4 (19.7)	**100% 23.0** (00.0)	90% 11.4 (04.2)	80% 73.2 (80.0)
TOTAL	80% 110.5 (91.2)	55% 72.5 (82.8)	35% 58.0 (91.2)	30% 74.1 (111)	100% 49.8 (18.9)	**100% 41.0** (00.0)	95% 31.4 (21.7)	40% 36.6 (40.0)
RIDGE REGRESSION REGULARIZATION, TARGET 95%								
YACHT	70% 103 (98.8)	20% 26.0 (52.0)	10% 36.0 (108)	80% 117 (93.7)	100% 76.6 (24.3)	100% 59.0 (00.0)	**100% 47.0** (43.0)	20% 65.5 (132)
ABALON.	100% 77.0 (33.8)	70% 63.0 (96.2)	70% 113 (113)	100% 68.4 (21.5)	100% 42.5 (09.4)	100% 23.0 (00.0)	**100% 18.6** (08.5)	70% 92.1 (89.6)
TOTAL	85% 90.0 (66.3)	44% 44.5 (74.1)	40% 74.5 (110)	90% 92.7 (57.6)	100% 59.5 (16.9)	100% 41.0 (00.0)	**100% 32.8** (25.7)	44% 78.8 (110)
RIDGE REGRESSION REGULARIZATION, TARGET 99%								
YACHT	40% 68.0 (114)	40% 34.0 (54.6)	10% 11.0 (33.0)	30% 103 (157)	100% 69.2 (22.2)	**100% 59.0** (00.0)	80% 47.4 (62.0)	0% 00.0 (00.0)
ABALON.	100% 76.0 (34.7)	80% 93.0 (70.0)	50% 76.0 (106)	40% 79.8 (116)	100% 45.9 (28.5)	100% 23.0 (00.0)	**100% 17.3** (09.3)	100% 132 (57.7)
TOTAL	70% 72.0 (74.7)	60% 63.5 (62.3)	30% 43.5 (69.5)	35% 91.5 (137)	100% 57.5 (25.4)	**100% 41.0** (00.0)	90% 32.4 (35.7)	50% 66.2 (28.8)
NEURAL NETS HYPERPARAMETER TUNING, TARGET 90%								
IRIS	70% 69.0 (60.6)	100% 151 (73.8)	70% 149 (136)	20% 28.2 (80.7)	100% 115 (58.5)	**100% 88.0** (00.0)	30% 37.7 (74.1)	50% 124 (137)
WINE	**40% 73.0** (108)	80% 185 (127)	30% 73.0 (123)	20% 16.2 (32.4)	80% 43.4 (25.1)	0% 00.0 (00.0)	0% 00.0 (00.0)	0% 00.0 (00.0)
TOTAL	**55% 71.0** (84.7)	90% 168.0 (100)	50% 111.0 (130)	20% 22.2 (56.6)	90% 79.3 (41.8)	50% 44.0 (00.0)	15% 18.9 (37.0)	25% 62.0 (68.9)
NEURAL NETS HYPERPARAMETER TUNING, TARGET 95%								
IRIS	40% 59.0 (104)	80% 119 (103)	40% 70.0 (96.3)	20% 09.6 (19.4)	80% 93.2 (68.6)	**100% 129** (00.0)	60% 130 (129)	40% 48.1 (73.0)
WINE	**90% 146** (120)	80% 152 (111)	80% 167 (132)	20% 03.6 (09.0)	60% 71.1 (107)	0% 00.0 (00.0)	0% 00.0 (00.0)	0% 00.0 (00.0)
TOTAL	**65% 102** (112)	80% 135.5 (107)	60% 118.5 (114)	20% 6.6 (14.2)	70% 82.2 (87.9)	50% 64.5 (00.0)	30% 65.4 (64.7)	20% 24.1 (36.5)
NEURAL NETS HYPERPARAMETER TUNING, TARGET 99%								
IRIS	60% 45.0 (52.8)	60% 95.0 (86.6)	30% 54.0 (110)	80% 64.2 (70.2)	**100% 156** (64.9)	0% 00.0 (00.0)	30% 14.8 (29.4)	40% 49.0 (77.4)
WINE	**60% 75.0** (79.4)	80% 84.0 (86.3)	60% 122 (128)	60% 38.4 (44.4)	70% 62.2 (78.0)	0% 00.0 (00.0)	0% 00.0 (00.0)	20% 37.7 (81.1)
TOTAL	**60% 60.0** (66.1)	70% 89.5 (86.5)	44% 88.0 (119)	70% 51.3 (57.3)	85% 109.2 (71.4)	0% 0.0 (00.0)	15% 7.4 (14.7)	30% 43.4 (79.2)

Table A.2.4. *Results of the numerical experiments synthetic problems. The table displays the proportion of runs that reached the specified targets and the average and standard deviation of evaluations required to reach the target. In bold, the best result obtained in terms of average of function evaluations.*

BENCHMARKS	ADARANKOPT+TR	ECP+TR	ADALIPO+TR	CMA-ES	CRS	DIRECT	MLSL	PRS
				SYNTHETIC FUNCTIONS I, TARGET 90%				
BRANIN	100% 21.4 (11.4)	100% 21.3 (11.9)	100% 20.3 (12.2)	100% 07.4 (03.1)	100% 10.5 (08.7)	**100% 05.0** (00.0)	100% 09.4 (26.1)	100% 10.1 (10.3)
HIMMEL.	100% 11.7 (05.2)	100% 11.5 (05.0)	100% 11.3 (04.1)	100% 07.1 (02.9)	100% 17.2 (15.1)	**100% 04.0** (00.0)	99% 10.2 (12.4)	100% 16.1 (14.4)
LEVY	100% 61.6 (35.9)	100% 49.4 (35.4)	100% 37.8 (45.9)	73% 24.2 (19.2)	100% 16.8 (13.8)	**100% 03.0** (00.0)	99% 51.1 (118)	100% 16.7 (16.1)
MCCORM.	100% 35.0 (18.5)	99% 32.9 (17.3)	100% 34.4 (19.3)	59% 22.8 (22.4)	100% 14.2 (10.7)	**100% 11.0** (00.0)	99% 35.4 (79.3)	100% 14.9 (14.3)
STYBL.	100% 38.4 (45.4)	100% 43.7 (51.0)	100% 48.1 (52.2)	**100% 10.5** (06.7)	90% 43.7 (36.1)	100% 20.0 (00.0)	99% 124 (101)	100% 73.0 (69.1)
TOTAL	100% 33.6 (23.3)	99% 31.8 (24.1)	100% 30.4 (26.8)	86% 14.4 (10.9)	98% 20.5 (16.9)	**100% 08.6** (00.0)	99% 46.1 (67.6)	100% 26.1 (24.9)
				SYNTHETIC FUNCTIONS I, TARGET 95%				
BRANIN	100% 22.8 (14.4)	100% 17.1 (14.1)	100% 16.5 (12.5)	100% 40.5 (12.9)	99% 20.2 (17.7)	**100% 05.0** (00.0)	98% 14.6 (39.1)	100% 20.4 (20.9)
HIMMEL.	100% 14.1 (10.2)	100% 13.4 (09.0)	100% 13.7 (09.3)	100% 22.4 (13.7)	100% 32.5 (26.6)	100% 22.0 (00.0)	**100% 12.5** (07.3)	100% 34.7 (32.8)
LEVY	99% 81.6 (59.3)	99% 88.5 (42.2)	100% 117 (121)	24% 07.9 (15.5)	100% 35.2 (24.4)	**100% 03.0** (00.0)	98% 99.9 (171)	100% 44.0 (44.3)
MCCORM.	100% 38.6 (25.9)	99% 34.5 (21.4)	100% 40.3 (24.5)	42% 16.1 (22.1)	97% 30.1 (39.7)	**100% 11.0** (00.0)	98% 42.9 (87.5)	100% 32.2 (30.0)
STYBL.	100% 52.4 (58.0)	100% 52.9 (60.1)	100% 73.3 (72.2)	99% 25.2 (17.8)	89% 58.2 (44.9)	**100% 34.0** (00.0)	100% 122 (102)	99% 154 (147)
TOTAL	99% 41.9 (33.6)	99% 41.3 (29.4)	100% 52.3 (48.0)	73% 22.4 (16.4)	97% 35.3 (30.7)	**100% 15.0** (00.0)	98% 58.5 (81.5)	99% 57.1 (55.1)
				SYNTHETIC FUNCTIONS I, TARGET 99%				
BRANIN	100% 35.0 (27.5)	100% 34.1 (28.6)	100% 35.3 (32.8)	100% 43.7 (20.8)	97% 66.2 (85.0)	**100% 16.0** (00.0)	97% 32.0 (76.7)	100% 93.2 (95.1)
HIMMEL.	100% 49.8 (38.5)	100% 53.2 (41.6)	100% 50.5 (36.7)	**100% 46.2** (25.8)	100% 88.3 (47.9)	100% 51.0 (00.0)	99% 17.3 (20.1)	99% 174 (163)
LEVY	99% 133 (105)	99% 101 (54.0)	95% 271 (233)	21% 18.3 (38.1)	100% 84.9 (39.9)	**100% 22.0** (00.0)	99% 110 (175)	97% 241 (222)
MCCORM.	100% 31.0 (35.9)	99% 31.0 (32.7)	100% 37.7 (46.3)	**100% 16.7** (11.2)	97% 67.2 (48.6)	100% 22.0 (00.0)	97% 54.6 (108)	99% 151 (148)
STYBL.	99% 125 (90.9)	99% 133 (128)	99% 119 (149)	**100% 19.6** (11.5)	89% 97.5 (55.6)	100% 38.0 (00.0)	100% 123 (106)	72% 290 (293)
TOTAL	99% 74.9 (59.6)	99% 70.7 (57.0)	99% 102.9 (99.9)	84% 28.9 (21.5)	96% 80.8 (55.4)	**100% 29.8** (00.0)	98% 67.6 (97.4)	93% 190.2 (184)

SYNTHETIC FUNCTIONS II, TARGET 90%

DEB	9% 52.8 (185)	6% 37.4 (157)	5% 29.6 (138)	23% 156 (307)	5% 27.4 (132)	**100% 175** (00.0)	90% 77.1 (130)	0% 00.0 (00.0)
HOLDER	95% 85.9 (68.9)	91% 65.9 (50.3)	99% 152 (146)	29% 16.9 (32.8)	67% 36.4 (44.1)	**100% 32.0** (00.0)	88% 138 (194)	99% 181 (174)
LINEAR	6% 14.1 (55.3)	14% 44.6 (125)	0% 00.6 (19.6)	100% 32.0 (10.6)	100% 159 (80.5)	100% 92.0 (00.0)	**100% 18.7** (18.6)	0% 00.2 (06.9)
ROSENB.	100% 15.4 (11.4)	100% 15.6 (11.9)	100% 14.6 (09.3)	100% 14.9 (09.9)	100% 11.1 (10.1)	**100% 03.0** (00.0)	99% 14.8 (46.0)	100% 07.2 (06.5)
SPHERE	**100% 10.0** (00.0)	100% 10.0 (00.0)	100% 10.0 (00.0)	99% 137 (133)	100% 95.7 (30.5)	100% 80.0 (00.0)	99% 15.0 (24.1)	2% 11.0 (84.4)
TOTAL	62% 35.6 (64.3)	62% 34.7 (68.9)	61% 41.6 (62.8)	70% 71.6 (98.8)	74% 66.0 (59.5)	**100% 76.4** (00.0)	95% 52.9 (82.9)	40% 40.1 (54.4)

SYNTHETIC FUNCTIONS II, TARGET 95%

DEB	1% 11.3 (91.1)	1% 07.1 (70.4)	1% 04.9 (52.8)	17% 127 (291)	1% 04.8 (54.8)	**100% 286** (00.0)	87% 90.9 (148)	0% 00.0 (00.0)
HOLDER	95% 92.4 (82.5)	93% 64.8 (54.1)	99% 175 (179)	34% 17.6 (29.7)	56% 169 (317)	**100% 32.0** (00.0)	88% 137 (188)	93% 276 (241)
LINEAR	0% 01.1 (15.4)	2% 08.6 (57.6)	0% 01.3 (28.8)	100% 143 (55.3)	99% 199 (130)	100% 100 (00.0)	**100% 21.2** (30.7)	0% 00.0 (00.0)
ROSENB.	100% 27.3 (21.4)	100% 24.2 (21.9)	100% 23.1 (18.0)	100% 18.4 (11.5)	100% 20.7 (17.5)	**100% 03.0** (00.0)	99% 22.7 (82.2)	100% 11.0 (11.5)
SPHERE	**100% 10.0** (00.3)	100% 10.0 (00.3)	100% 10.0 (00.3)	98% 191 (159)	100% 105 (32.5)	100% 212 (00.0)	100% 16.2 (29.4)	0% 02.7 (35.3)
TOTAL	59% 28.4 (42.2)	59% 23.0 (40.9)	60% 42.9 (55.9)	70% 99.6 (109)	71% 99.9 (110)	**100% 126** (00.0)	94% 57.7 (95.7)	38% 57.9 (57.7)

SYNTHETIC FUNCTIONS II, TARGET 99%

DEB	0% 01.5 (32.9)	0% 00.7 (20.9)	0% 00.0 (00.0)	11% 89.5 (255)	0% 00.0 (00.0)	**100% 379** (00.0)	84% 98.4 (144)	0% 00.0 (00.0)
HOLDER	94% 133 (113)	92% 89.2 (77.1)	94% 268 (225)	8% 08.0 (30.6)	51% 163 (298)	**100% 32.0** (00.0)	88% 148 (198)	40% 182 (288)
LINEAR	0% 00.3 (08.8)	0% 00.0 (00.0)	0% 00.0 (00.0)	100% 224 (71.7)	98% 251 (171)	**100% 118** (00.0)	99% 21.3 (21.4)	0% 00.0 (00.0)
ROSENB.	100% 67.3 (55.4)	100% 66.1 (70.3)	100% 48.8 (49.4)	100% 25.8 (19.3)	100% 71.3 (43.9)	**100% 03.0** (00.0)	98% 25.1 (56.4)	100% 60.3 (66.2)
SPHERE	**100% 10.0** (00.0)	100% 10.0 (00.0)	100% 10.0 (00.0)	94% 206 (157)	100% 122 (39.7)	100% 300 (00.0)	100% 16.3 (08.5)	0% 00.0 (00.0)
TOTAL	58% 42.4 (42.0)	58% 33.2 (33.7)	58% 65.5 (55.1)	62% 110.8 (106)	70% 121.7 (110)	**100% 166** (00.0)	94% 62.0 (85.9)	28% 48.6 (70.9)

2.4 Trust region module

Trust region methods are a class of iterative optimization algorithms that construct a local model m_n of the objective function V around the current iterate X_n. At each iteration, X_{n+1} is obtained by minimizing m_n within a neighborhood of X_n – the *trust region* — whose size is adjusted adaptively: expanded when the model agrees well with the true objective, contracted otherwise.

A key benefit of the decision-based formulation is that it naturally accommodates trust region methods as a local search module. In this section, we present the main components of trust region algorithms, with a focus on the BOBYQA algorithm (Powell, 2009), and describe how they can be embedded within any decision-based algorithm to improve its local convergence behavior.

Trust region framework. At each iteration n, a trust region method solves the following subproblem

$$s_n = \underset{s \in \Omega}{\operatorname{argmin}}\ m_n(X_n + s) \text{ subject to } \|s\|_2 \leq \Delta_n,$$

where m_n is the local surrogate of V (typically a quadratic model) and Δ_n is the *trust region radius*. After computing the candidate step s_n, the algorithm evaluates the *ratio of actual to predicted reduction*

$$\rho_n = \frac{V(X_n) - V(X_n + s_n)}{m_n(X_n) - m_n(X_n + s_n)}.$$

If ρ_n is sufficiently large, $X_{n+1} = X_n + s_n$ and the trust region is expanded; if ρ_n is small, the candidate step is rejected and the trust region is contracted.

BOBYQA. BOBYQA is a derivative-free trust region algorithm introduced in Powell (2009) specifically designed for bound-constrained optimization problems. Using an an *interpolation set* of m points $\mathcal{Y}_n = \{y_1, ..., y_m\}$, it fits a quadratic model of the objective function

$$m_n(X_n + s) = V(X_n) + g_n^\top s + \frac{1}{2} s^\top H_n s,$$

to the interpolation data by requiring $m_n(y_i) = f(y_i)$ for all $y_i \in \mathcal{Y}_n$. At each iteration, it solves the trust region subproblem using an iterative procedure based an the Cauchy point method. The ratio ρ_n is then computed and the step is accepted or rejected accordingly, as well as the trust region radius is updated. BOBYQA maintains two trust region radii throughout the optimization process: the current radius Δ_n and a lower bound on Δ_n, δ. The invariant $\delta \leq \Delta_n$ is enforced at all time. Whenever Δ would fall below, the algorithm decreases δ and sets $\Delta_n \leftarrow \delta$. The algorithm begins with a user-supplied initial value $\delta \stackrel{\text{def}}{=} \delta_{\text{begin}}$ and terminates when δ drops below a user-specified threshold δ_{end}. This two-level scheme ensures that BOBYQA explores the domain at coarse scales before refining the search, providing a form of multi-resolution behavior without explicit multi-start logic.

Trust region as a module. Trust region methods such as BOBYQA can be embedded as a local search module within any decision-based algorithm. As described in Algorithm A.2.1, decision-based algorithms iteratively sample candidate points according to a probability measure (often the uniform distribution) and accept or reject them through a sequence of decision rules. A fundamental limitation of this scheme is that the search space is explored purely through random sampling, which is poorly suited to exploiting basins of attraction: the probability of drawing a point near the bottom of a basin is low, and decreases rapidly with dimension. Incorporating a trust region module addresses this limitation by injecting local, model-based search whenever the global sampler identifies a promising region. The procedure is straightforward: at each iteration, a candidate point x is sampled and evaluated by the decision rule as usual. If the candidate is accepted, a trust region method is initialized at x and run for a fixed number of iterations or until convergence; the best point found is then returned as the next iterate. If the candidate is rejected, the trust region step is skipped entirely.

This design has two key properties. First, the theoretical guarantees of the base algorithm might be preserved. Since the trust region module is only invoked after a successful acceptance, the best point found by the trust region method has a function value at least as good as the accepted candidate. However, the change in the distribution of accepted points may affect the properties of the base algorithm. Second, the additional computational cost is confined to promising regions of the search space, where it is most likely to yield improvement. An additional benefit is that the trust region radius can be reused as an exclusion criterion: a candidate that falls within the trust region of a previously accepted point is unlikely to lead to further improvement, and can be rejected without evaluation, thus avoiding redundant local searches. This modular approach is described in Algorithm A.2.2, and is concretely implemented in the GLOBE library presented in Chapter A.4. An illustrative example of the performance improvement provided by this module is given in Table A.2.5.

Table A.2.5. *Performance improvement provided by the trust region module. The* 2 *dimensional functions are evaluated* 1000 *times and the results are averaged over* 10 *runs. The table shows the best function value found (mean and standard deviation) for* ADALIPO+ *and* ADARANKOPT *with and without the trust region module. The addition of the trust region module significantly improves the performance of both algorithms.*

	Rastrigin	Ackley	Square
ADALIPO+	0.238	0.405	0.030
	(0.444)	(0.201)	(0.035)
ADALIPO+TR	0.102	0.087	$1e^{-8}$
	(0.298)	(0.061)	($4e^{-8}$)
ADARANKOPT	6.062	12.626	0.002
	(2.109)	(3.373)	(0.002)
ADARANKOPT+TR	$3e^{-4}$	0.038	$2e^{-8}$
	($5e^{-4}$)	(0.043)	($4e^{-8}$)

Algorithm A.2.2: DECISION-BASED ALGORITHM WITH TRUST REGION MODULE

Input:
$V : \Omega \to \mathbb{R}$; $n \in \mathbb{N}$; $\mu \in \mathcal{P}(\Omega)$; a sequence of decision rules $(D_n)_{n\in\mathbb{N}}$; a trust region method TR

Output:
An approximation of $x^\star$

Initialisation:
$X_0 \sim \mu$
for $i = 0$ **to** n **do**

  do

    $X \sim \mu$
    if X is in the trust region of a previously accepted point **then**

      continue

    end, $b \sim \mathcal{B}(D_i(\mathsf{C}_i, V(\mathsf{C}_i), X))$ *Bernoulli r.v. parametrized by the decision rule*

  while $\neg b$ **end** *Candidate rejection loop*
  $V(X_{i+1}), X_{i+1} \leftarrow \mathsf{TR}(X)$ *Run the trust region algorithm initialized at X*

end
return $\underset{0\le i\le n}{\text{argopt}}\, V(X_i)$

3 — McKean-Vlasov-based approaches

Abstract. We present the family of *McKean-Vlasov-based* approaches, a class of global optimization algorithms that leverage the mathematical framework of McKean-Vlasov stochastic differential equations to model the dynamics of interacting particles. These algorithms include well-known methods such as *Consensus-Based Optimization* (CBO) or Langevin-based algorithms. We introduce this framework using CBO as a canonical example, and then present *Stein Boltzmann sampling* (SBS), a novel algorithm introduced in this thesis, which extends the *Stein Variational Gradient Descent* (SVGD) method to the global optimization setting by incorporating a Boltzmann distribution as a target measure. We then study SVGD on a compact domain. We prove that its continuous-time dynamics exists and keeps the particles in the domain, and that the Kullback–Leibler divergence to the target decreases along it. When the kernel is Stein-universal, this dynamics converges to the target. Finally, we develop noise injection modules, a joint work with Pierre Germain and Samuel Gruffaz, that can be applied to any McKean-Vlasov-based algorithm to address the issue of vanishing stochasticity in the mean-field dynamics and to improve the exploration capabilities of these algorithms.

A significant amount of global optimization algorithms can be seen as *particle-based* algorithms, which rely on a population of (interacting) particles to explore and exploit the search space. Well known examples of particle-based algorithms include the family of particle swarm optimization (PSO) algorithms (Fang et al., 2023; Houssein et al., 2021; Kennedy & Eberhart, 1995) and the wide family of bio-inspired algorithms (e.g. Mirjalili et al. (2014); Wang et al. (2024)). The strucutre of the dynamics of these algorithms are informally very similar but mathematically quite different, which makes it hard to reason about them in a unified way. It is due to the fact that these algorithms are often made from intuitive heuristics and lack a rigorous mathematical formulation. However, a subclass of *particle-based* algorithms are designed with a rigorous mathematical formulation such as

Consensus-based optimization (CBO) algorithms (Carrillo et al., 2021; Pinnau et al., 2017), Langevin-based algorithms (Gao et al., 2020; Prigogine & Balescu, 1957; Roberts & Tweedie, 1996; Welling & Teh, 2011), and *Stein Boltzmann sampling* (SBS) (Serré et al., 2025), a contribution of this thesis. The dynamics of the particles in these algorithms can be described by McKean-Vlasov stochastic differential equations, which are a class of stochastic differential equations where the evolution depends on the distribution of the solution itself. This specificity allows to model the interaction between particles in a rigorous way.

Definition A.3.8 – *McKean-Vlasov process*

Let $b : \mathbb{R}^d \times \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}^d$ and $\sigma : \mathbb{R}^d \times \mathcal{P}(\mathbb{R}^d) \to \mathcal{M}_d(\mathbb{R})$ be measurable functions where $\mathcal{M}_d(\mathbb{R})$ is the space of $d \times d$ real matrices. A process $(X_t)_{t\in\mathbb{R}_+}$ is said to be a *McKean-Vlasov process* if it solves the following system

$$\begin{cases} \mathrm{d}X_t &= b(X_t, \rho_t)\,\mathrm{d}t + \sigma(X_t, \rho_t)\,\mathrm{d}B_t \\ \mathcal{L}(X_0) &= \rho_0 \end{cases}, \tag{16}$$

where $\rho_t \overset{\text{def}}{=} \mathcal{L}(X_t)$ describes the law of the process at time t and B_t is a standard Brownian motion.

The law of a McKean-Vlasov process is the solution to the following non-linear Fokker-Planck equation

$$\partial_t \rho_t = -\nabla \cdot (b(\cdot, \rho_t)\rho_t) + \frac{1}{2}\sum_{i,j=1}^{d} \partial^2_{x_i x_j}\big(a_{ij}(\cdot, \rho_t)\rho_t\big), \tag{17}$$

where $a = \sigma\sigma^\top$ is the diffusion matrix (Carmona et al., 2018).

In fact, the dynamics of the particles in these algorithms approximate McKean-Vlasov dynamics through multiple systems of interacting particles. Indeed, the particles (noted $(X^i)_{i\in[N]}$) are solutions to

$$\begin{cases} \mathrm{d}X^i_t = b\big(X^i_t, \hat{\rho}_{X_t}\big)\,\mathrm{d}t + \sigma\big(X^i_t, \hat{\rho}_{X_t}\big)\,\mathrm{d}B_t \\ (X^i_0)_{i\in[N]} \text{ are i.i.d. with law } \rho_0 \end{cases},$$

where $\hat{\rho}_{X_t}$ is the empirical distribution of the particles at time t, i.e.,

$$\hat{\rho}_{X_t} \overset{\text{def}}{=} \frac{1}{N}\sum_{i\in[N]} \delta_{X^i_t}.$$

Under Lipschitz conditions on the coefficients b and σ, the empirical distribution $\hat{\rho}_{X_t}$ converges as $N \to \infty$ to the measure ρ_t that satisfy the corresponding McKean-Vlasov system in Definition A.3.8. This phenomenon, known as *propagation of chaos* (Sznitman, 2006), ensures that the particles in these algorithms become asymptotically independent and identically distributed according to ρ_t.

The mean-field limit reduces the stochastic particle system to a deterministic evolution on probability space governed by Eq. 17, which allows to analyze the

asymptotic behavior of the algorithm and to derive convergence guarantees. All algorithms previously mentioned share the same dynamics structure. We call *McKean-Vlasov-based* approaches any algorithm whose dynamics can be described by systems of interacting particles that approximate McKean-Vlasov dynamics.

A simple example of a McKean-Vlasov-based algorithm would be the LANGEVIN algorithm i.e., $N+1$ particles following the Langevin dynamics

$$\mathrm{d}X_t^i = -\nabla V(X_t^i)\,\mathrm{d}t + \sqrt{2\kappa}\,\mathrm{d}B_t^i, \tag{18}$$

where $\kappa > 0$ is a temperature parameter. In this algorithm, the drift function is $b(x,\mu) = -\nabla V(x)$ and the diffusion function is $\sigma(x,\mu) = \sqrt{2\kappa} I_d$. The particles in this algorithm do not interact with each other as the dynamics of each particle is independent of the distribution of the particles.

McKean-Vlasov-based algorithms can be seen as instances of the following abstract algorithm.

Algorithm A.3.3: MCKEAN-VLASOV-BASED ALGORITHM

Input:
$V : \Omega \to \mathbb{R}$; $N, n \in \mathbb{N}$; $\rho \in \mathcal{P}(\Omega)$; a *drift* function $b : \mathbb{R}^d \times \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}^d$; a *diffusion* function $\sigma : \mathbb{R}^d \times \mathcal{P}(\mathbb{R}^d) \to \mathcal{M}_d(\mathbb{R})$; a sequence of step sizes $(\varepsilon_n)_{n\in\mathbb{N}}$

Output:
An approximation of $x^\star$

Initialisation:
$(X_1^i)_{i\in[N]} \sim \rho^{\otimes N}$ *N i.i.d. samples from ρ*
for $t = 1$ **to** n **do**

$\xi_t^i \sim \mathcal{N}(0_d, I_d)^{\otimes N}$
$X_{t+1}^i \leftarrow X_t^i + \varepsilon_{t+1} \cdot b\big(X_t^i, \hat{\rho}_{X_t}\big) + \sqrt{\varepsilon_{t+1}} \cdot \sigma\big(X_t^i, \hat{\rho}_{X_t}\big)\xi_t^i$ *Euler-Maruyama scheme*

end
return $\underset{i\in[N],\ t\in[n]}{\operatorname{argopt}} V(X_t^i)$

As in Chapter A.2, we can leverage this shared structure to reason about these algorithms in a unified way and to develop general modules that can be applied to any McKean-Vlasov-based algorithm.

In the following, we present some of the algorithms that can be seen as McKean-Vlasov-based approaches and introduce SBS, a new algorithm developed in this thesis. We then study the theory of SVGD on a compact domain, which gives guarantees for SBS. We also present a general module for improving the performance of these algorithms by adding common noise to the dynamics.

Future work will focus on the design of common-noise modules, including improved schemes aimed at further enhancing performance even in smooth landscapes, as well as a theoretical analysis of particle dynamics in the presence of common noise. Although the latter constitutes a challenging problem, it is of particular importance, as it may provide qualitative evidence that common noise effectively mitigates the

lack-of-exploration issue in McKean–Vlasov–based algorithms (Tugaut, 2014), a known limitation of these methods. Such results would support the view that the proposed approach is not only empirically effective, but also theoretically grounded.

3.1 Consensus-based optimization

Consensus-based optimization (CBO) is a derivative-free McKean-Vlasov-based algorithm first introduced in Pinnau et al. (2017) and further developed in Carrillo et al. (2018); Carrillo et al. (2021) designed to solve global minimization problems. The core idea is to model a population of particles whose dynamics are driven by two competing mechanisms: drift toward a weighted *consensus point* (a proxy for the global minimizer) and stochastic exploration through diffusion noise. Over time, the particles are designed to concentrate around the global minimum of the objective function.

Formulation. Consider a system of N interacting particles $(X_t^i)_{i\in[N]}$ whose dynamics are governed by the following stochastic differential equations

$$\begin{aligned} \mathrm{d}X_t^i = &-\lambda\Big(X_t^i - c_V\Big[\hat{\rho}_{X_t}\Big]\Big)H^\varepsilon\Big(V(X_t^i) - V\Big(c_V\Big[\hat{\rho}_{X_t}\Big]\Big)\Big)\,\mathrm{d}t \\ &+\sqrt{2}\sigma D\Big(X_i^t - c_V\Big[\hat{\rho}_{X_t}\Big]\Big)\,\mathrm{d}B_t^i, \end{aligned}$$

where

- $\lambda > 0$ is the drift rate, controlling the strength of the attraction toward the consensus point;
- $\sigma > 0$ is the diffusion coefficient, controlling the intensity of the stochastic exploration;
- H^ε is a smooth regularization of the Heaviside function, defined as $H^\varepsilon(x) = 1/2\ \mathrm{erf}(x/\varepsilon) + 1/2$, which ensures that particles are only attracted to the consensus point if their function value is worse than that of the consensus point;
- $D\Big(X_i^t - c_V\Big[\hat{\rho}_{X_t}\Big]\Big)$ is either $\Big\|X_i^t - c_V\Big[\hat{\rho}_{X_t}\Big]\Big\|_2^2 I_d$ or $\mathrm{diag}\Big(X_i^t - c_V\Big[\hat{\rho}_{X_t}\Big]\Big)$, depending on whether the diffusion is isotropic or anisotropic, respectively. In the isotropic case, the diffusion is stronger for particles that are farther from the consensus point, while in the anisotropic case, the diffusion is stronger along the directions where the particle is farther from the consensus point. In both cases, the diffusion vanishes as particles approach the consensus point, allowing for fine convergence to local minima. Anisotropic diffusion is often preferred in practice due to its better performance in high-dimensional settings (Carrillo et al., 2021).
- B_t^i are independent standard Brownian motions in $\mathbb{R}^d$;
- $c_V[\rho]$ is the consensus point, defined as a weighted average of the particle positions

$$c_V[\rho] \overset{\text{def}}{=} \frac{\int_\Omega x \omega_V^\alpha(x)\,\mathrm{d}\rho}{\int_\Omega \omega_V^\alpha(x)\,\mathrm{d}\rho},$$

where $\omega_V^\alpha \overset{\text{def}}{=} \exp(-\alpha V(x))$ and $\alpha > 0$. When $\rho = \hat{\rho}_{X_t}$, it reduces to

$$c_V\left[\hat{\rho}_{X_t}\right] = \frac{\sum_{i\in[N]} X_t^i \omega_V^\alpha(X_t^i)}{\sum_{i\in[N]} \omega_V^\alpha(X_t^i)}.$$

The CBO algorithm is thus indeed a McKean-Vlasov type algorithm, with the following drift and diffusion functions

$$b(x,\rho) = -\lambda(x - c_V[\rho])H^\varepsilon(V(x) - V(c_V[\rho])) \text{ and}$$
$$\sigma(x,\rho) = \sqrt{2}\sigma D(x - c_V[\rho]).$$

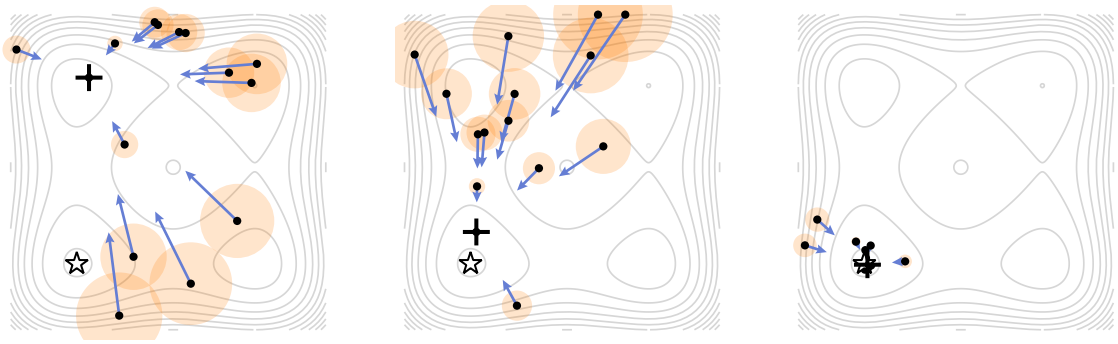
The central object is the consensus point, which concentrates to argmin $V(X_i)$ as $\alpha \to \infty$. The argmin is not directly usable as it cannot be computed using only the distribution of the particles, which would break the McKean-Vlasov structure of the algorithm. An illustration of the dynamics of the CBO is given in Figure A.3.11.

Convergence guarantees. Using isotropic noise and under suitable assumptions on V, it can be shown that the individual particles in CBO follow the gradient flow associated with the function $x \mapsto \|x - x^\star\|_2^2$. This gradient flow is independent of the geometry of V, meaning the CBO performs a canonical convexification of a large class of non-convex optimization problems. The details of this analysis are given in Fornasier et al. (2024). The anisotropic case is analyzed in Fornasier et al. (2022).

Consensus-based optimization has demonstrated strong empirical performance across a wide range of applications, with particular success in high-dimensional machine learning tasks (Fornasier et al., 2021; Fornasier et al., 2022). The method has since attracted considerable attention, spawning an active line of research with numerous variants, extensions, and analysis (Bianchi et al., 2025; Borghi et al., 2023; Bungert et al., 2025; Chao et al., 2026; Gerber et al., 2025). This popularity, combined with the McKean-Vlasov structure that underlies the algorithm, was one of the principal motivations for introducing the general class of McKean-Vlasov-based algorithms studied in this thesis, and for developing the common-noise modules designed to improve their performance — see Section A.3.4.

3.2 Stein Boltzmann Sampling

We present *Stein Boltzmann Sampling* (SBS), an adaptation of *Stein Variational Gradient Descent* (SVGD) (Liu & Wang, 2016), a popular algorithm of Bayesian inference, to global optimization (Serré et al., 2025). SBS uses SVGD to sample from the Boltzmann distribution (BD) of the objective function on Ω, which concentrates around its global minimizers. The theoretical analysis of SVGD on a compact domain is the subject of Section A.3.3.



(a) $t = 0$ (b) $t = 1.1$ (c) $t = 4$

Figure A.3.11. *Illustration of the isotropic CBO dynamics (with $H^{\varepsilon} = 1$) at successive times, with $N = 15$ particles (black dots) on the two-dimensional Styblinski-Tang function (gray level lines), whose global minimizer is the star. The blue arrows are the drift of the particles toward the consensus point $c_V\left[\hat{\rho}_{X_t}\right]$ (black cross), and the radius of the orange disks is proportional to the standard deviation of their diffusion. Particles farther from the consensus point experience stronger drift and diffusion.* ***a)*** *The consensus point lies in a local basin.* ***b)*** *A particle has found the global basin, and the consensus point jumps there.* ***c)*** *The particles concentrate around the global minimizer.*

We first recall the BD and SVGD. Then, we summarize the guarantees that Section A.3.3 gives for SBS, and we present its variants, the choice of its hyperparameters and its experimental evaluation.

3.2.1 Boltzmann distribution

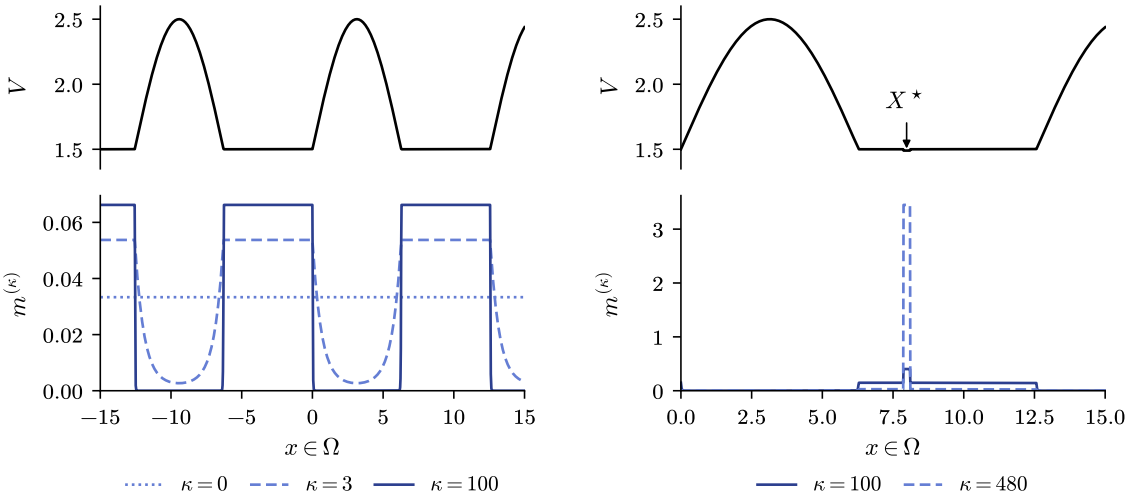
The BD is a well-known distribution in statistical physics (Boltzmann, 1970). Also named the Gibbs measure, it is used to model the distribution of the energy of a system in thermal equilibrium. It is also deeply entangled with global optimization as it can be used to construct a probability distribution that concentrates around the minima of a function. This link has been made explicit in the litterature on *Simulated Annealing* (Azencott, 1989; Kirkpatrick et al., 1983).

Definition A.3.9 – *Continuous Boltzmann distribution*

Given a function $f \in C^0(\Omega, \mathbb{R})$, the Boltzmann distribution over Ω is induced by the probability density function $m^{(\kappa)}_{f,\Omega} : \Omega \to \mathbb{R}_{\geq 0}$ defined by

$$m^{(\kappa)}_{V,\Omega}(x) = m^{(\kappa)}(x) = \frac{e^{-\kappa f(x)}}{\int_{\Omega} e^{-\kappa f(t)}\, \mathrm{d}t}, \forall \kappa \in \mathbb{R}_{\geq 0}.$$

The parameter κ is called the *inverse temperature*. The higher κ is, the more concentrated the mass is around the minima of f. When κ tends to infinity, the BD tends to a distribution supported over the minima of f. If $\lambda(X^\star) > 0$, the BD tends to a uniform distribution over $X^\star$ (see Figure A.3.12a). If $\lambda(X^\star) = 0$, it tends to a distribution over $X^\star$, where the concentration of the mass depends on the local geometry of the minimizing manifold (Hwang, 1980).



(a) (b)

Figure A.3.12. ***a)*** *The density of the Boltzmann distribution $m^{(\kappa)}$ (Definition A.3.9) (blue lines, bottom) becomes uniform over the set of minimizers $X^\star$ of the given function V to optimize (**black** line, top), as its parameter κ tends to infinity.* ***b)*** *In this example, the volume of the set $X^\star$ is much smaller than the volume of local minimizers in the flat region. The value of the function at the local minimizers is also closer to the value of the global ones. Setting κ to 100 does not suffice to concentrate the majority of the mass of $m^{(\kappa)}$ around the global minimizers.*

Properties A.3.1

Let $m^{(\kappa)}$ be defined as in Definition A.3.9. Then, the following properties hold:

- If $\lambda(X^\star) = 0$, then $\forall x \in \Omega$,

$$\lim_{\kappa \to \infty} m^{(\kappa)}(x) = \begin{cases} \infty \text{ if } x \in X^\star \\ 0 \text{ otherwise} \end{cases}.$$

- If $0 < \lambda(X^\star)$, then $\forall x \in \Omega$,

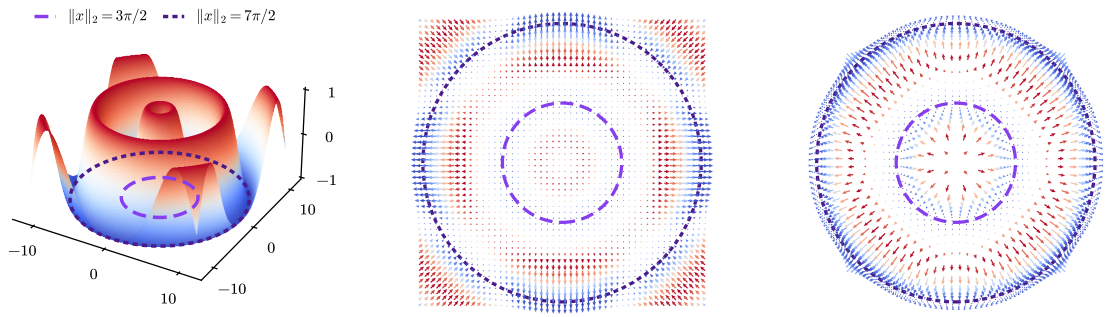
$$\lim_{\kappa \to \infty} m^{(\kappa)}(x) = \begin{cases} \lambda^{-1}(X^\star) \text{ if } x \in X^\star \\ 0 \text{ otherwise} \end{cases}.$$

- $\forall f C^0(\Omega, \mathbb{R})$,

$$\lim_{\kappa \to \infty} \int_\Omega f(x) m^{(\kappa)}(x)\, \mathrm{d}x = f^\star.$$

See Hwang (1980) for the proofs of these properties.

A visual representation of the BD is given in Figure A.3.12a. One can see that, as κ increases, $m^{(\kappa)}$ becomes more and more concentrated around the minima of V. In addition, Figure A.3.12b illustrates the difficulty of choosing a good value for κ. This will be made more clear in the next following when we present SBS. To sample from the BD, we need to compute the integral $\int_\Omega e^{-\kappa V(t)}\, \mathrm{d}t$, which however, is likely to be intractable for a general V.



(a) $x \mapsto \sin \|x\|_2$ (b) $t = 0$ (c) $t = 700$

Figure A.3.13. *Illustration of the vector field* $\left(\phi^*_{\hat{\rho}_{X_t}}\right)_\Phi$ *of the Stein witness (see Theorem A.3.4) in a discrete-time setting where* π *is the BD.* ***a)*** *The function to optimize,* $x \mapsto \sin \|x\|_2$*, whose global minimizers are the dashed circles.* ***b)*** *The initial particles start getting attracted toward the two ring-shaped manifolds.* ***c)*** *After some SVGD iterations, there are stronger forces in the vector field and the particles get concentrated around those minimizing regions.*

3.2.2 Stein Variational Gradient Descent

Sampling from an intractable distribution is a common task in Bayesian inference, where the normalizing constant of the posterior distribution cannot be computed. SVGD (Liu & Wang, 2016) iteratively transforms an arbitrary measure ρ into a target distribution π, using only the score of π. It is based on Stein's method (Stein, 1972), and its theory has been developed in several works (Duncan et al., 2023a; Korba et al., 2020; Korba et al., 2021; Liu, 2017; Lu et al., 2019). In the case of SBS, π is the BD of V (Definition A.3.9) for some $\kappa > 0$, whose score is $s = -\kappa \nabla V$. Its normalizing constant is not needed. We present SVGD briefly. Its definitions and results are stated precisely in Section A.3.3.

Kernel and Stein witness. Let k be a kernel on $\mathbb{R}^d$, given by a feature map $\Phi : \mathbb{R}^d \to H_0$, i.e. $k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{H_0}$. Let $\mathcal{H} = H_0^d$. An element $f \in \mathcal{H}$ defines the vector field $f_\Phi(y) = \left(\langle f_i, \Phi(y) \rangle_{H_0} \right)_{1 \leq i \leq d}$. The Stein operator of π maps a vector field v to $\mathcal{A}_s v = s^\top v + \nabla \cdot v$. By Theorem A.3.4, there exists a unique $\phi^*_\rho \in \mathcal{H}$, the *Stein witness* of ρ, such that

$$\forall f \in \mathcal{H}, \quad \mathbb{E}_{x \sim \rho}[\mathcal{A}_s f_\Phi(x)] = \left\langle f, \phi^*_\rho \right\rangle_{\mathcal{H}}.$$

Up to normalization, it is the direction of $\mathcal{H}$ along which ρ violates Stein's identity the most. It is also the direction of steepest descent of the Kullback–Leibler divergence to π (Theorem A.3.5). Thus, SVGD updates ρ in this direction, with a small step size $\varepsilon > 0$, i.e.

$$\rho_{n+1} = \left(I_d + \varepsilon \left(\phi^*_{\rho_n} \right)_\Phi \right) \# \rho_n. \tag{19}$$

SBS is a McKean-Vlasov-based algorithm. In practice, ρ_0 is the empirical measure of N particles. Let $\hat{\rho}_n = \frac{1}{N} \sum_{j=1}^N \delta_{X_n^j}$. Then, Eq. 19 moves each particle as follows

$$\begin{cases} X_{n+1}^i = X_n^i + \varepsilon\left(\phi^*_{\hat{\rho}_n}\right)_\Phi(X_n^i) \\ \left(X_0^i\right)_{i\in[N]} \sim \rho_0^{\otimes N} \end{cases} \tag{20}$$

where

$$\left(\phi^*_{\hat{\rho}_n}\right)_\Phi(x) = \frac{1}{N}\sum_{j=1}^{N}\Big(k(X_n^j, x)s(X_n^j) + \nabla_{X_n^j} k(X_n^j, x)\Big).$$

This formulation makes it clear that SBS is a *McKean-Vlasov-based* algorithm, with a zero diffusion and the drift $b(x, \rho) = \left(\phi^*_\rho\right)_\Phi(x)$. The forces driving the particles are determined by a mixture of individual and collective information (see Section A.3.2.7). A usual choice for k is the Gaussian kernel k_σ. An illustration of this equation is given in Figure A.3.13.

3.2.3 Guarantees of SBS

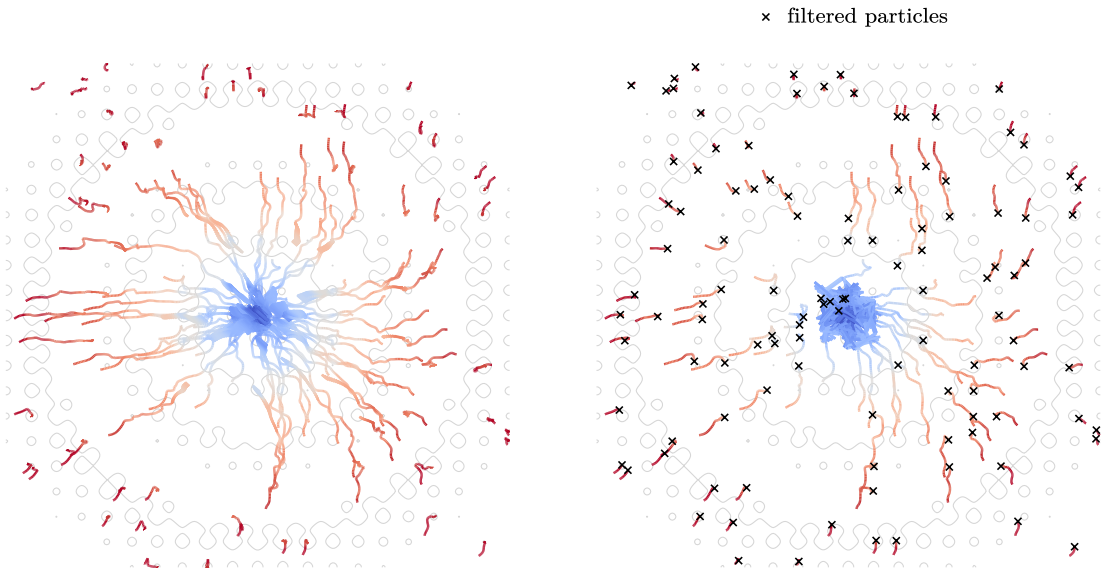
The theoretical analysis of SBS studies the distribution ρ_t of the particles in continuous time. It describes the particles when their number tends to infinity and the step size ε tends to 0 (Liu, 2017; Lu et al., 2019). The particles then follow the McKean-Vlasov equation Eq. 16, with the drift b above and a zero diffusion. In Section A.3.3, we study this dynamics on a compact domain, and we apply the results to SBS (Corollary A.3.4). Let the kernel of SBS satisfy Assumption A.3.2. In particular, its feature map vanishes on $\partial\Omega$. Then, the continuous-time dynamics of SBS exists, keeps the particles in Ω, and decreases the Kullback–Leibler divergence to the BD. If, moreover, the kernel is Stein-universal (Definition A.3.12), ρ_t converges weakly to the BD. Thus, for a large κ, the distribution of the particles approaches a distribution concentrated around the global minimizers of V.

The Gaussian kernel k_σ, which we use in the experiments below, does not satisfy Assumption A.3.2, as it does not vanish on $\partial\Omega$. The construction of a Stein-universal kernel that satisfies Assumption A.3.2, and new experiments with this kernel, are left for future work. Moreover, these guarantees concern the continuous-time dynamics, and not the iteration with finitely many particles.

3.2.4 SBS variants

In addition to the main SBS method, we introduce two variants that can be more efficient in practice. The first one, called SBS-PF, uses a particle filtering approach that removes the less promising particles (without replacing them). The second one, called SBS-HYBRID, is a hybrid method that uses SBS as a continuation for other global optimization methods, or –seen the other way around– those methods are used to initialize SBS. SBS-PF uses less budget than SBS, and SBS-HYBRID uses some of the budget to run one of the pre-existing methods to initialize SBS with better starting points; the aim is to approximate the global minimum better than SBS with the same budget.

SBS with particle filtering (SBS-PF). We use a simple particle filtering idea: to remove particles (i.e. candidate minimizers of V) that are less promising or stuck in bad



(a) SBS (b) SBS-PF

Figure A.3.14. *Illustration of the flow of the trajectories of particles over the iterations. The color gradient represents the* 2*d Ackley function value, from blue (low) to red (high). The trajectories draw the discretized flow of the distribution of particles.* ***a)*** *SBS: the particles are initialized uniformly at random over the domain, and then get updated by making a small step in the direction induced by SVGD forces.* ***b)*** *SBS-PF variant with particle filtering: the particles are initialized and updated as before, but the less promising ones get rapidly removed and are not replaced. This is visible as there are less persisting trajectories in areas where the function has high values. This strategy results in a significant reduction of the budget while having comparable performance.*

local minima. We choose to remove particles that do not move and correspond to significantly higher function values than the others, hence particles that are very likely stuck in bad local minima. This is done by removing particles using their function values and the distance between their previous and actual positions. More precisely, if these two quantities for a particle are respectively higher than the q -th and lower than the p-th percentiles of the function values and "previous-to-actual" distances of the particles, then that particle is removed. The difference between SBS and this variant is visualized in Figure A.3.14. One can see that, in SBS-PF, the least promising candidates are rapidly removed without being replaced, so that the remaining particles are more likely to converge to the global minimum. This strategy results in a significant reduction of the budget used, while having comparable optimization results to SBS. Note that Corollary A.3.4 does not apply to SBS-PF, which removes particles. Thus, SBS-PF has no theoretical guarantee. However, the empirical results show that SBS-PF is efficient in practice, and it is a good alternative to SBS when the budget is limited.

SBS-hybrid. This variant is based on the idea of using SBS or SBS-PF as a continuation for particles- or distribution-based methods, such as WOA (Mirjalili & Lewis, 2016) or CMA-ES (Hansen & Ostermeier, 2001). Indeed, the design of SBS allows to initialize the particles with the result of one such method, and then resume the optimization process with an SBS variant. More specifically, we introduce SBS-HYBRID that runs few iterations of both CMA-ES and WOA to choose the most

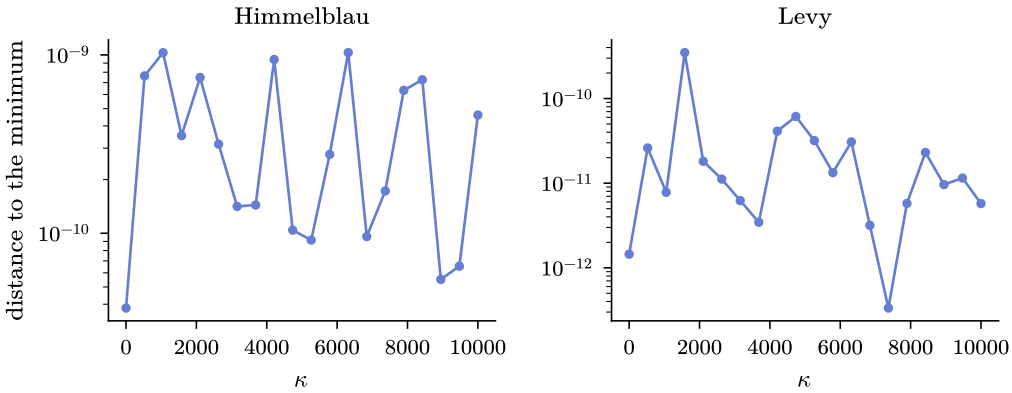
promising result, and then continues the optimization with SBS. Both WOA and CMA-ES are efficient methods, thus, a small number of iterations allows to find a good starting distribution for SBS. Moreover, both methods are not well-suited for a large budget, but for different reasons: CMA-ES uses early stopping rules (e.g. the condition number of the covariance matrix), and WOA takes more time to run than SBS for the same budget. SBS-HYBRID can be seen as a combination of the asymptotically consistent SBS method, on top of very efficient but non-consistent methods. Among SBS-HYBRID's strengths, we can see that: i) it empirically provides high-quality results, and ii) it keeps the guarantees of Corollary A.3.4, provided that the initial distribution given by WOA and CMA-ES has a finite Kullback–Leibler divergence to the BD.

3.2.5 Choice of hyperparameters

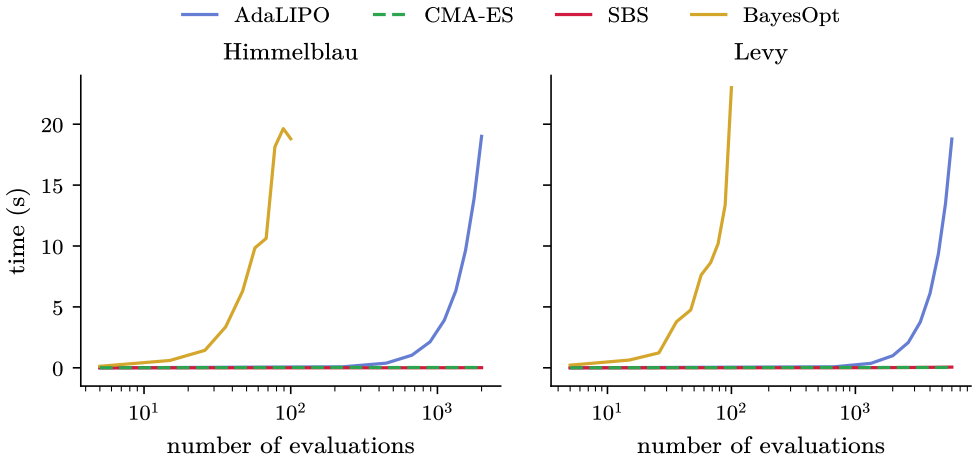
In this section, we discuss the choice of the hyperparameters of SBS and its variants. We focus on the choice of κ and σ, which carry complex information about the behavior of the method.

Choice of temperature. As detailed earlier, κ controls the shape of the BD from which SVGD samples. The bigger κ is, the more the mass of the distribution gets concentrated around the global minimizers of the function. Intuitively, the optimal κ such that a satisfying amount of the mass is around the global minimizers depends on the geometry of the function around local minima (the asymptotic behavior of the BD depends on the local geometry, see Hwang (1980)). Nevertheless, one can see in Figure A.3.15a that, in practice, the choice of κ does not significantly affect the performance of SBS. The reason is that, if the modes of the BD that contain most of the density mass are the ones around the global minimizers, then SVGD would succeed in moving some particles in the areas of those modes, provided there are enough particles. The three following parameters control how the mass is repartitioned: the value of κ, the ratio between the volume of the region of local minimizers and the volume of the region of global minimizers, and the value of the function at the local minimizers. These three parameters are interdependent: the value of one can compensate for the value of the others. It is rather unlikely to encounter a function where these three parameters do not compensate each other. It would require the function to have an arbitrary small volume ratio or an arbitrary small distance between the local and global minima (see Figure A.3.12b). Thus, a very large κ, such as 10^3, compensates almost all potential issues related to the geometry of the function and ensures a good performance on average.

Choice of bandwith. All SBS variants use the RBF kernel with a bandwidth σ, the choice of which is crucial for their performance. As detailed in Section A.3.2.7, the size of σ controls the forces developed between particles. When a lot of particles are close together, they repel each other. This behavior enforces the exploration of the function domain, but at the same time it prevents SBS from converging at narrow regions where global minimizers could be located. A natural choice is $\sigma = \frac{1}{N^2}$, where N is the number of particles, which ensures that, when the particles are few, σ gets large enough and SBS explores the domain. On the other hand, with a lot a particles the exploration is ensured by the initial uniform distribution ρ_0



(a) Log-distance between the computed solution and the global minimum vs. the value of κ for SBS.



(b) Execution time vs. the number of evaluations for BAYESOPT, ADALIPO, CMA-ES, and SBS.

Figure A.3.15. *Insights for the compared algorithms:* ***a)*** *shows the low impact of κ on the performance of SBS.* ***b)*** *shows the time to run for BAYESOPT and ADALIPO grows exponentially and is significantly higher than for CMA-ES or SBS. In each case (a) and (b), the* ***left*** *plot is for the Himmelblau, and the* ***right*** *is for the Levy function.*

and the small induced σ allows the particles to converge to the global minimum, even in narrow regions. For the SBS-PF variant, σ changes during the optimization process, as particles are being filtered out. For the SBS-HYBRID variant, as the initial particles are supposed to be already well-positioned and possibly close to the global minimum, σ is set to a very small value, e.g. $\sigma \overset{\text{def}}{=} 10^{-10}$.

3.2.6 Experimental evaluation

Among the SBS variants, we provide one named SBS-PF-HYBRID that employs SBS-PF as the final step of SBS-HYBRID. In our numerical comparison, we compare the SBS variants with the following state-of-the-art global optimization methods: CMA-ES (Hansen & Ostermeier, 2001); WOA (Mirjalili & Lewis, 2016), a particle-swarm method; ADALIPO (Malherbe & Vayatis, 2017a); BAYESOPT (Brochu et al., 2010; Martinez-Cantin, 2014); and CBO (Pinnau et al., 2017).

We use classical two-dimensional benchmark functions for global optimization. Some are noisy and multimodal (Ackley, Drop wave, Egg Holder, Holder Table,

Michalewicz, Rastrigin, Levy), others are smooth (Branin, Goldstein Price, Himmelblau, Rosenbrock, Camel, Sphere) (see more information in Surjanovic & Bingham (2024)). For the results of Table A.3.6, we ran each method 10 times on each function.

In the literature, the budget is defined as the number of function evaluations, however, the actual computational time can vary significantly between the methods and needs to be taken into account. Thus, the budget is set in order for the methods to stop in a reasonable time when run on a personal computer[5]. As one can see in Figure A.3.15b, the running time of ADALIPO and BAYESOPT is significantly higher than for the other methods. For this reason, their budget is set lower than for the other methods: 2K for ADALIPO, 100 for BAYESOPT, and 800K for the rest.

We introduce the following *empirical competitive ratio*

$$\mathrm{ECR}(m) = \frac{1}{|\mathscr{B}|} \sum_{V \in Y} \min\left(100, \frac{dV_{\mathrm{m}}}{dV^*}\right), \tag{21}$$

where $\mathscr{B}$ is the set of benchmark functions, dV_{m} is the distance between the global minimum of V and the approximation found by the method m, and dV^* is the smallest distance among all methods. "ECR" provides information about the average precision compared to the best method (lower is better; the best is 1).

In the results of Table A.3.6, one can see that SBS outperforms almost all state-of-the-art methods and scores the fourth rank on average. SBS-PF achieves comparable results on average with significantly less function evaluations ($\sim 97\%$ budget reduction). Moreover, SBS-HYBRID and SBS-PF-HYBRID outperform all the other methods on average. They combine the efficiency of either CMA-ES and WOA with the suitability of SBS for large budgets, while the addition of particle filtering reduces the budget by $\sim 67\%$. In parallel, SBS, SBS-PF-HYBRID, and SBS-HYBRID score respectively the 3^{rd}, 2^{nd}, and 1^{st} rank on the competitive ratio measure, showing that their approximations are precise on average compared to the other methods.

In Table A.3.7, we provide the results of the same experiment on 50 dimensional benchmark functions, by restricting to only those methods that can run in low computational time. The budget is set to 8M function evaluations. One can observe that SBS and its variants outperform all competitors, and SBS-PF achieves the best results with a budget reduction of $\sim 97\%$ compared to SBS. However, the budget reduction of SBS-PF-HYBRID is less significant ($\sim 16\%$) due to the fact that the high dimensionality of the functions and the initial distribution makes the less promising particles harder to distinguish. Overall, all SBS variants seem to be robust to function shapes, contrary to CMA-ES for instance, which is very precise on valley-shaped functions but struggles on the multimodal functions.

[5]the experiments were performed on an Apple M2 chip with 8 cores and 16GB of RAM.

Table A.3.6. *Comparison between all SBS variants and state-of-the-art methods on 2D benchmark functions. Reported values are average distance to the global minimum and standard deviation (lower is better).* SBS-HYBRID *and* SBS *rank 1st and 4th respectively, while* SBS-PF-HYBRID *and* SBS-PF *achieve competitive results with significantly fewer evaluations (67% and 97% budget reduction).*

BENCHMARKS	STATE-OF-THE-ART						PROPOSED METHOD AND VARIANTS			
	LANGEVIN	BAYESOPT	CBO	ADALIPO	CMA-ES	WOA	SBS-PF	SBS	SBS-PF-HYBRID	SBS-HYBRID
ACKLEY	6.8 (1.7)	0.1 (0.1)	0.0 (0.0)	1.3 (0.7)	19.8 (0.0)	**$8e^{-8}$** ($5e^{-8}$)	0.0 ($9e^{-4}$)	$8e^{-4}$ ($3e^{-4}$)	$1e^{-5}$ ($1e^{-5}$)	$7e^{-6}$ ($3e^{-6}$)
BRANIN	$8e^{-5}$ ($1e^{-4}$)	$2e^{-4}$ ($1e^{-4}$)	0.4 (0.3)	0.0 (0.0)	$3e^{-7}$ (0)	$1e^{-6}$ ($1e^{-6}$)	$5e^{-7}$ ($2e^{-7}$)	$3e^{-7}$ ($3e^{-11}$)	**$3e^{-7}$** (0)	$3e^{-7}$ ($5e^{-16}$)
DROP WAVE	0.9 ($1e^{-16}$)	0.2 (0.1)	0.1 (0.1)	0.1 (0.0)	0.3 (0.3)	**$1e^{-15}$** ($1e^{-15}$)	0.0 (0.0)	0.0 (0.0)	0.1 (0.0)	0.0 (0.0)
EGG HOLDER	2008.8 ($2e^{-13}$)	90.3 (60.6)	833.5 ($1e^{-10}$)	40.0 (18.7)	393.6 ($5e^{-14}$)	**$3e^{-5}$** ($5e^{-10}$)	18.0 (19.0)	8.0 (10.2)	7.8 (9.6)	21.5 (32.0)
GOLDSTEIN	$3.9e^{5}$ ($3e^{5}$)	8.3 (6.6)	$4e^{-4}$ ($2e^{-4}$)	0.4 (0.3)	8.1 (24.3)	$1e^{-6}$ ($7e^{-7}$)	$6e^{-7}$ ($5e^{-7}$)	$1e^{-9}$ ($2e^{-9}$)	**$6e^{-14}$** ($6e^{-15}$)	$6e^{-13}$ ($1e^{-12}$)
HIMMELBLAU	47.5 (31.0)	$7e^{-4}$ ($9e^{-4}$)	0.1 (0.1)	0.0 (0.0)	$9e^{-16}$ ($1e^{-15}$)	$1e^{-6}$ ($1e^{-6}$)	$1e^{-7}$ ($3e^{-7}$)	$5e^{-11}$ ($4e^{-11}$)	$4e^{-19}$ ($7e^{-19}$)	**$1e^{-20}$** ($1e^{-20}$)
HOLDER TABLE	3.4 (0.5)	0.4 (0.9)	0.0 ($7e^{-4}$)	0.0 (0.0)	5.0 (5.0)	**$2e^{-6}$** ($2e^{-6}$)	$2e^{-6}$ ($1e^{-7}$)	$2e^{-6}$ ($2e^{-9}$)	$2e^{-6}$ ($1e^{-10}$)	$2e^{-6}$ ($1e^{-10}$)
MICHALEWICZ	9.6 (0.2)	7.9 ($1e^{-5}$)	8.6 (0.0)	7.9 (0.0)	8.0 (0.3)	7.9 ($1e^{-6}$)	7.9 ($1e^{-6}$)	7.9 ($1e^{-10}$)	7.9 ($8e^{-14}$)	**7.9** ($4e^{-15}$)
RASTRIGIN	30.5 (3.7)	2.2 (1.2)	$9e^{-4}$ ($7e^{-4}$)	0.2 (0.2)	5.4 (5.8)	**$6e^{-15}$** ($7e^{-15}$)	$5e^{-6}$ ($3e^{-6}$)	$5e^{-9}$ ($1e^{-8}$)	0.5 (0.7)	0.3 (0.6)
ROSENBROCK	6852.4 (4943.7)	0.2 (0.2)	0.0 (0.0)	0.1 (0.0)	$4e^{-16}$ ($7e^{-16}$)	$2e^{-7}$ ($1e^{-7}$)	$6e^{-5}$ ($9e^{-5}$)	$2e^{-6}$ ($4e^{-6}$)	$5e^{-17}$ ($8e^{-17}$)	**$1e^{-17}$** ($1e^{-17}$)
CAMEL	397.9 (9.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	$2e^{-5}$ ($1e^{-14}$)	$2e^{-5}$ ($2e^{-8}$)	**$2e^{-5}$** ($1e^{-7}$)	$2e^{-5}$ ($1e^{-11}$)	$2e^{-5}$ (0)	$2e^{-5}$ ($1e^{-16}$)
LEVY	83.5 (12.9)	0.1 (0.1)	0.0 (0.0)	0.0 (0.0)	1.0 (2.1)	$8e^{-9}$ ($7e^{-9}$)	$9e^{-8}$ ($8e^{-8}$)	$1e^{-12}$ ($2e^{-12}$)	$3e^{-19}$ ($8e^{-19}$)	**$9e^{-20}$** ($2e^{-19}$)
SPHERE	$9e^{-5}$ ($6e^{-5}$)	$5e^{-4}$ ($5e^{-4}$)	0.0 (0.0)	0.0 ($9e^{-4}$)	$5e^{-16}$ ($1e^{-15}$)	$1e^{-16}$ ($8e^{-17}$)	$5e^{-8}$ ($7e^{-8}$)	$6e^{-12}$ ($7e^{-12}$)	$1e^{-19}$ ($2e^{-19}$)	**$1e^{-21}$** ($3e^{-21}$)
ECR	62.2	46.8	46.7	46.7	24.0	22.4	46.7	20.8	16.2	**15.4**
AVG. RANK	9.38	7.85	7.46	7.15	6.69	3.00	4.15	3.38	3.15	**2.77**
FINAL RANK	10	9	8	7	6	2	5	4	3	**1**

3.2.7 Discussion

We discuss several important aspects of SBS, including its theoretical connections to existing algorithms, the role of kernel hyperparameters, and current limitations that motivate future improvements.

Link with Simulated Annealing. This link is not difficult to see: indeed, both algorithms are asymptotic methods that sample from the BD. However, the way they sample from that distribution is different. Simulated Annealing is a Markov Chain Monte-Carlo method (Azencott, 1989; Kirkpatrick et al., 1983), while SBS is a deterministic variational approach. The minimum temperature parameter of Simulated Annealing is the inverse of the κ parameter of SBS. Thus, any scheduler for Simulated Annealing's temperature can also be used in SBS. However, there is an extra degree of exploration/exploitation in SBS, which is brought by the kernel bandwidth that is employed by the SVGD sampling.

Locality of the kernel. In classical SVGD implementations, the RBF kernel is used:

Table A.3.7. *Comparison between all SBS variants with several state-of-the-art methods on 50 dimensional benchmark functions. For each function, we report the average distance to the global minimum (lower is better). The precision is truncated. SBS-HYBRID runs $1K$ iterations of CMA-ES and WOA. As one can see, SBS-PF ranks 1^{st}, as SBS, while having a significant budget reduction ($\sim 97\%$). Moreover, SBS-PF-HYBRID outperforms SBS-HYBRID while having a budget reduction of $\sim 16\%$. The high dimensionality of the functions makes the particle filtering of SBS-PF-HYBRID less efficient.*

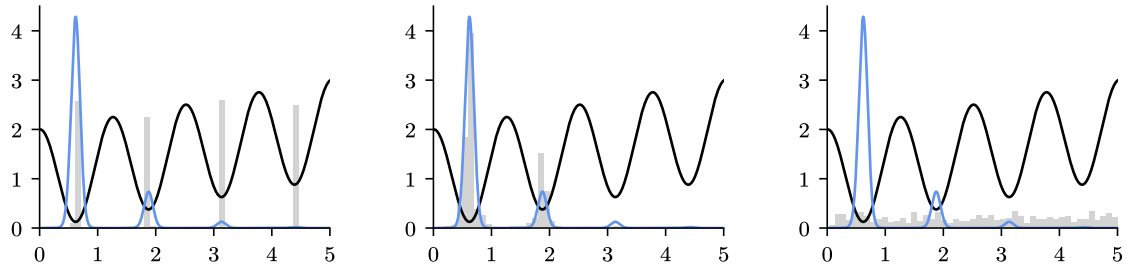
BENCHMARKS	STATE-OF-THE-ART				PROPOSED METHOD AND VARIANTS			
	LANGEVIN	CBO	WOA	CMA-ES	SBS-HYBRID	SBS-PF-HYBRID	SBS	SBS-PF
ACKLEY	21.5	21.1	19.8	19.6	19.5	19.6	**19.0**	19.0
	(0.0)	(0.1)	(0.1)	(0.0)	(0.1)	(0.1)	(0.1)	(0.1)
MICHALEWICZ	8.6	**0.8**	4.4	25.8	24.2	24.1	3.8	2.4
	(0.6)	(0.5)	(0.2)	(2.4)	(1.8)	(2.5)	(0.4)	(0.8)
RASTRIGIN	884.4	841.1	593.8	**107.4**	114.5	115.0	280.5	280.5
	(202.5)	(302.9)	(22.2)	(20.9)	(24.9)	(10.9)	(17.2)	(17.2)
ROSENBROCK	$3.6e^5$	$1.7e^5$	$2e^4$	**0.4**	28.9	28.7	39.5	25.3
	($2.2e^4$)	(0)	(3609.2)	(1.2)	(1.3)	(2.0)	(1.4)	(4.0)
LEVY	2931.7	349.8	224.5	75.3	81.1	70.4	**55.9**	57.8
	(88.4)	(38.3)	(13.4)	(27.0)	(19.6)	(23.2)	(2.6)	(4.4)
SPHERE	0.0	5000.0	646.3	$1e^{-14}$	$\mathbf{7e^{-20}}$	$2e^{-19}$	$2e^{-10}$	$1e^{-4}$
	($3e^{-4}$)	(0)	(31.5)	($2e^{-15}$)	($1e^{-20}$)	($2e^{-19}$)	($1e^{-11}$)	($5e^{-6}$)
ECR	43.9	36.0	35.3	**1.5**	13.4	13.2	18.1	28.2
AVG. RANK	7.17	6.17	5.83	3.67	3.67	3.50	**3.00**	**3.00**
FINAL RANK	7	6	5	4	3	2	**1**	**1**

$$k(x, x') = \exp\left(-\frac{\|x - x'\|_2^2}{2\sigma^2}\right),$$

as it is in the Stein class of any smooth density supported on $\mathbb{R}^d$. The bandwidth σ controls the locality of the attraction and repulsion forces applied on the particles, expressed as

$$\text{attr}(x) = \mathbb{E}_{x' \sim \hat{\rho}_{X_i}}[\nabla \log \pi(x') k(x, x')],$$
$$\text{rep}(x) = \mathbb{E}_{x' \sim \hat{\rho}_{X_i}}[\nabla_{x'} k(x, x')].$$

The first term attracts remote particles to a close cluster of particles, and the second term repels particles that are too close to each other. Hence, they are respectively exploitation and exploration forces. Indeed, the attraction allows particles to "fall" in local minima, wherein a lot of particles are already trapped. The repulsion prevents particles from getting stuck together at a narrow region of the search space, and forces them to explore the space. σ controls the range of these forces. A small σ value leads to a weak repulsion and thus more exploitation. An arbitrary small σ leads to a uniform distribution over the local minima. In the contrary, a large σ leads to more exploration, as the particles will repel themselves even from a very far distance. An arbitrary large σ leads to a uniform discretization of the



(a) $\kappa = 7$, $\sigma = 10^{-15}$ (b) $\kappa = 7$, $\sigma = 0.5$ (c) $\kappa = 7$, $\sigma = 1000$

Figure A.3.16. *Illustration of the exploration/exploitation trade-off in SBS with different values of σ. In **black**, the function $x \mapsto \cos(5x) + x/5 + 1$; in **grey**, the distribution of the particles; in **blue**, the BD $m^{(\kappa)}$. When σ is too small, the particles are uniformly distributed over the local minima. When σ is too large, they are uniformly distributed over the whole domain Ω.*

space. These behaviors are illustrated in Figure A.3.16. In the case of SBS, the value of σ is a user parameter.

Limitations and points of improvement. Due to the gradient approximation by finite difference that is employed by SBS, our methods require a large budget. That is a common issue in gradient-based optimization algorithms. However, contrary to more frugal approaches (e.g. BAYESOPT), SBS and its variants have a way smaller execution time. Some works have been proposed to remove the gradient necessity in SVGD such as Han & Liu (2018) and Braun et al. (2024). The first one uses a proxy distribution instead of the target, with the assumption that the gradient of the log of the proxy distribution is easier to compute. In our context, the design of such a proxy distribution is not straightforward. The second work uses evolution strategy updates instead of the gradient-based updates. This approach is more promising for our context and should be investigated in future works.

Another weakness of SBS is the difficult choice of the kernel. As explained earlier, the kernel controls the particles movements and the performance of one specific kernel choice strongly depends on the geometry of the objective function. This choice is crucial and the user should tune this hyperparameter carefully. A possible way to mitigate this problem could be to find an adaptive kernel that uses only evaluations of the objective function for choosing the best way for the particles to interact. This is a quite complex subject that is out of the scope of the current study, and should be investigated in a more general context.

3.3 SVGD on a compact domain

SVGD (Liu & Wang, 2016) transports a set of particles toward a target distribution π. At each iteration, the particles move along the direction of steepest descent of the Kullback–Leibler divergence to π. This direction is taken in the unit ball of a space of vector fields defined by a kernel. When the number of particles tends to infinity, the distribution ρ_t of the particles follows a McKean-Vlasov equation, as in Definition A.3.8 with a zero diffusion (Liu, 2017; Lu et al., 2019). The analysis

of this equation is done on $\mathbb{R}^d$, for smooth target distributions (e.g. Duncan et al. (2023a); Korba et al. (2020); Lu et al. (2019)).

In our global optimization setting, the search space Ω is compact. SBS (Section A.3.2) applies SVGD to the Boltzmann distribution of the objective function on Ω. Two difficulties arise. First, nothing keeps the particles in Ω, and the theory on $\mathbb{R}^d$ does not apply. Second, the boundary of Ω may have corners or cusps, where it has no outward normal. The integration by parts on Ω, on which the classical proofs rely, is then not available. In Serré et al. (2025), the existence of the continuous-time dynamics and the fact that it stays in Ω were assumed.

In this section, we show that a suitable kernel overcomes these two difficulties, without any regularity of $\partial\Omega$. We first state our three main results. Their precise statements come later in the section.

3.3.1 Main results

We work under two assumptions. The target distribution π has a positive and continuous density p on Ω, and a continuous score s, which plays the role of $\nabla \log p$ (Assumption A.3.1). The kernel is given by a feature map $\Phi : \mathbb{R}^d \to H_0$, i.e. $k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{H_0}$, which is continuously differentiable, Lipschitz, bounded, and zero on $\partial\Omega$ (Assumption A.3.2). For instance, a weighted kernel $w(x)w(y)k_0(x, y)$, where w vanishes on $\partial\Omega$, satisfies the last condition.

For $\rho \in \mathcal{P}(\Omega)$, let ϕ^*_ρ be the Stein witness of ρ (Theorem A.3.4). Its vector field $\left(\phi^*_\rho\right)_\Phi$ is the direction of steepest descent of the Kullback–Leibler divergence to π (Theorem A.3.5). SVGD moves the particles along it. In continuous time, let $T_t(x)$ be the position at time t of the particle that starts from x. Then, $T_0 = I_d$ and

$$\frac{\mathrm{d}}{\mathrm{d}t} T_t(x) = \left(\phi^*_{\rho_t}\right)_\Phi (T_t(x)), \quad \text{where} \quad \rho_t = (T_t)\#\rho_0.$$

We call it the continuous-time SVGD equation (see Eq. 24). We prove the following three results.

Existence. For every $\rho_0 \in \mathcal{P}(\Omega)$, the continuous-time SVGD equation has a solution $(T_t)_{t\geq 0}$, defined for all times, and the particles never leave Ω (Theorem A.3.6 and Lemma A.3.9).

Decrease of the Kullback–Leibler divergence. If $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$, then, along every solution,

$$\forall t \geq 0, \quad \frac{\mathrm{d}}{\mathrm{d}t} \mathrm{KL}(\rho_t \parallel \pi) = -\mathfrak{K}(\rho_t \mid \pi) \leq 0,$$

where $\mathfrak{K}$ is the kernelized Stein discrepancy (Theorem A.3.8).

Convergence. If, moreover, the kernel is Stein-universal (Definition A.3.12), then ρ_t converges weakly to π as t tends to infinity (Theorem A.3.9).

[6] github.com/gaetanserre/CompactSVGD

No regularity of $\partial\Omega$ is assumed, and the measures need no density. All the results of this section are formalized in L∃∀N. The formalization[6] and a website that presents it[7] are available online.

The proofs of these results rely on many intermediate results, which we develop in the rest of this section. In Section A.3.3.2, we state the assumptions on the domain and on the target distribution. In Section A.3.3.3, we recall Stein's method, and we prove Stein's identity for the vector fields that vanish on $\partial\Omega$. In Section A.3.3.4, we define the kernelized Stein discrepancy and the Stein witness. In Section A.3.3.5, we show that the Stein witness is the direction of steepest descent of the Kullback–Leibler divergence. We prove the existence of the continuous-time dynamics in Section A.3.3.6, the decrease of the Kullback–Leibler divergence in Section A.3.3.7, and the convergence in Section A.3.3.8. Finally, we apply these results to SBS in Section A.3.3.9.

3.3.2 Setting

In this section, we introduce the domain on which the measures live and the target distribution. We state the assumptions under which we work.

Notations Let $d \geq 1$ be the dimension. The space $\mathbb{R}^d$ has the Euclidean norm $\|\cdot\|$ and its Borel σ-algebra $\mathcal{B}(\mathbb{R}^d)$. Let $\Omega \subseteq \mathbb{R}^d$ be a compact set. Recall that $\mathcal{P}(\Omega)$ is the set of probability measures ρ on $\mathbb{R}^d$ carried by Ω, i.e. such that $\rho(\mathbb{R}^d \setminus \Omega) = 0$. Note that a measure of $\mathcal{P}(\Omega)$ needs no density with respect to λ. For $\rho \in \mathcal{P}(\Omega)$ and a function g defined on Ω, we write $\mathbb{E}_{x\sim\rho}[g(x)] = \int_\Omega g \,\mathrm{d}\rho$, when this integral exists. For $y \in \mathbb{R}^d$, let δ_y be the Dirac mass at y.

SVGD iteratively transforms an arbitrary measure ρ into a target distribution π. We assume that π has a density p with respect to $\lambda|_\Omega$, and a score s that plays the role of $\nabla \log p$.

Assumption A.3.1 – *Target distribution*

Let $p : \mathbb{R}^d \to \mathbb{R}$ and $s : \mathbb{R}^d \to (\mathbb{R}^d)^*$. The target distribution is $\pi = p \cdot \lambda|_\Omega$, i.e.

$$\forall A \in \mathcal{B}(\mathbb{R}^d), \quad \pi(A) = \int_{A\cap\Omega} p \,\mathrm{d}\lambda.$$

The following conditions hold.

(i) π is a probability measure.
(ii) p is measurable and continuous on Ω, and $p(x) > 0$ for every $x \in \Omega$.
(iii) For every $x \in \Omega^\circ$, $\log p$ is differentiable at x and $D(\log p)(x) = s(x)$.
(iv) s is continuous on Ω.

By (iii), $s = \nabla \log p$ on Ω°, once $(\mathbb{R}^d)^*$ is identified with $\mathbb{R}^d$. Thus, s is the score of π on the interior of Ω, and it is only required to be continuous up to the boundary. By (i), Ω is nonempty. As Ω is compact, (ii) gives that p reaches a positive minimum on Ω. Thus, $\log p$ is continuous and bounded on Ω. Likewise, by (iv), $M_s = \max_{x\in\Omega} \|s(x)\|$ is finite. No regularity of the boundary $\partial\Omega$ is assumed.

[7] gaetanserre.fr/CompactSVGD

Example A.3.1

Let $V : \mathbb{R}^d \to \mathbb{R}$ be continuously differentiable, let $\kappa > 0$, and assume that $\lambda(\Omega) > 0$. The Boltzmann distribution of V on Ω has the density

$$p(x) = \frac{e^{-\kappa V(x)}}{Z}, \quad \text{where} \quad Z = \int_\Omega e^{-\kappa V} \, \mathrm{d}\lambda.$$

It satisfies Assumption A.3.1 with $s = -\kappa DV$. Indeed, $e^{-\kappa V}$ is positive and continuous on the compact set Ω, which has a positive measure. Thus, $0 < Z < \infty$ and $\pi(\mathbb{R}^d) = 1$. Moreover, $\log p = -\kappa V - \log Z$ is differentiable on $\mathbb{R}^d$, and its derivative $-\kappa DV$ is continuous. A characteristic property of the Boltzmann distribution is that, as κ tends to infinity, it tends to a distribution supported on the set of the minimizers of V. It is the target distribution of SBS (see Definition A.3.9 and Section A.3.3.9).

3.3.3 Stein's method

SVGD is based on Stein's method, introduced by Stein (1972). In this section, we first recall the classical notions of this method. Then, we show that Stein's identity holds for the target distribution, without any assumption on $\partial\Omega$, for the vector fields that vanish on $\partial\Omega$. To do so, we compute the first variation of the Kullback–Leibler divergence, i.e. its derivative along a small perturbation of the identity (Theorem A.3.3). This result is also the basis of SVGD (Section A.3.3.5), and its proof is the template of Section A.3.3.7. Finally, we define the Stein discrepancy.

3.3.3.1 The Stein operator and Stein's identity

Stein's method defines the Stein operator associated with a distribution.

Definition A.3.10 – *Stein operator*

Let E be a finite-dimensional real normed space and let $s : E \to E^*$. The Stein operator of score s maps a vector field $v : E \to E$ to the function $\mathcal{A}_s v$, defined by

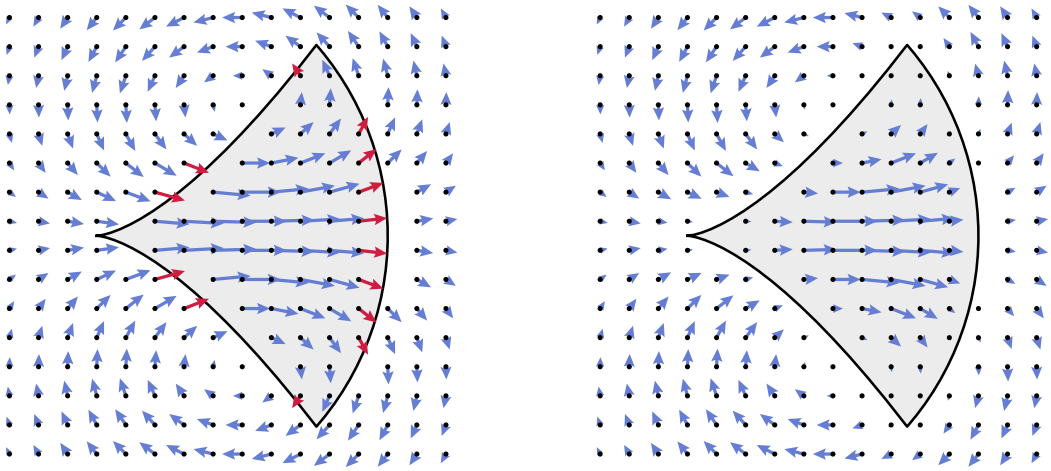
$$\mathcal{A}_s v(x) = s(x)(v(x)) + \nabla \cdot v(x)$$

at every $x \in E$ where v is differentiable.

When $E = \mathbb{R}^d$ and $s = \nabla \log p$ for a positive differentiable density p, we have that $\mathcal{A}_s v = \nabla \log p^\top v + \nabla \cdot v$. This is the Stein operator associated with p, as in Liu et al. (2016) and Liu & Wang (2016). It depends on p only through its score s.

Stein's identity states that the expectation of $\mathcal{A}_s v$ under the distribution of density p is zero (Liu et al., 2016). It holds for the vector fields v of the Stein class of p, which we now describe. As $Dp = ps$, we have that $\nabla \cdot (pv) = p\mathcal{A}_s v$. Thus, Stein's identity holds when the integral of $\nabla \cdot (pv)$ is zero. The Stein class of p is the set of the regular vector fields v that satisfy this condition. On Ω, when $\partial\Omega$ and pv are regular, the divergence theorem gives

$$\int_\Omega \nabla \cdot (pv) \, \mathrm{d}\lambda = \int_{\partial\Omega} pv \cdot \vec{n} \, \mathrm{d}S,$$



(a) $v = u$ (b) $v = \min(1, \operatorname{dist}(\cdot, \partial\Omega)/\delta)u$

Figure A.3.17. *Transport of a grid of particles by $I_d + \varepsilon v$, inside and outside $\Omega = \{(x, y) \in \mathbb{R}^2 \mid y^2 \le x^3, x^2 + y^2 \le 1\}$, where $d = 2$ and $\varepsilon = 0.06$. The set Ω is in gray. Its boundary $\partial\Omega$ has a cusp at 0 and two corners, where it has no outward normal. The field $u = (\partial_y\chi, -\partial_x\chi)$ has two counter-rotating vortices, with $\chi(z) = e^{-\|z-p_1\|^2/\sigma^2} - e^{-\|z-p_2\|^2/\sigma^2}$, $p_1 = (0.5, 0.3)$, $p_2 = (0.5, -0.3)$ and $\sigma = 0.6$. Each arrow goes from a particle x to $x + \varepsilon v(x)$. **a)** The field $v = u$ does not vanish on $\partial\Omega$. The red particles cross $\partial\Omega$. **b)** The field $v = \min(1, \operatorname{dist}(\cdot, \partial\Omega)/\delta)u$, with $\delta = 0.2$, is equal to u at distance at least δ from $\partial\Omega$, and vanishes on $\partial\Omega$. As $\varepsilon K_v < 1$, each arrow is shorter than the distance from its particle to $\partial\Omega$, so no particle crosses $\partial\Omega$. This illustrates Lemma A.3.1.*

where $\vec{n}$ is the outward unit normal and S is the surface measure of $\partial\Omega$. Thus, this condition requires that pv has no flux through $\partial\Omega$.

We now turn to the target distribution $\pi = p \cdot \lambda|_\Omega$ of Assumption A.3.1. Let $x \in \Omega^\circ$. As Ω° is open and $p > 0$ on it, $p = \exp(\log p)$ on a neighborhood of x. By the chain rule and (iii) of Assumption A.3.1, p is differentiable at x and $Dp(x) = p(x)s(x)$. The boundary $\partial\Omega$ may have a positive Lebesgue measure, and Assumption A.3.1 says nothing about the derivative of p on it. The divergence theorem also requires a regular boundary. The classical proof of Stein's identity for π therefore needs a regular $\partial\Omega$. However, for the vector fields that vanish on $\partial\Omega$, this identity holds without any assumption on $\partial\Omega$, as we now show. The proof relies on the first variation of the Kullback–Leibler divergence.

3.3.3.2 First variation of the Kullback–Leibler divergence

Let ρ and ρ' be probability measures on $\mathbb{R}^d$. The Kullback–Leibler divergence of ρ with respect to ρ' is

$$\mathrm{KL}(\rho \parallel \rho') = \int \log \frac{\mathrm{d}\rho}{\mathrm{d}\rho'} \, \mathrm{d}\rho$$

if $\rho \ll \rho'$ and $\log\left(\frac{\mathrm{d}\rho}{\mathrm{d}\rho'}\right)$ is ρ-integrable, and $\mathrm{KL}(\rho \parallel \rho') = \infty$ otherwise. It is nonnegative, and $\mathrm{KL}(\rho \parallel \rho) = 0$. Under Assumption A.3.1, if $\mathrm{KL}(\rho \parallel \pi) < \infty$, then $\rho \ll \pi$. As $\pi\left(\mathbb{R}^d \setminus \Omega\right) = 0$, we have that $\rho \in \mathcal{P}(\Omega)$.

A small perturbation of the identity by a vector field that vanishes on $\partial\Omega$ maps Ω onto itself (see Figure A.3.17). Let I_d be the identity map of $\mathbb{R}^d$.

Lemma A.3.1 – *Perturbations of the identity*
Let $v : \mathbb{R}^d \to \mathbb{R}^d$ be K_v-Lipschitz, with $v(y) = 0$ for every $y \in \partial\Omega$. Let $\varepsilon \in \mathbb{R}$ such that $|\varepsilon| K_v < 1$, and let $\Psi = I_d + \varepsilon v$. Then, Ψ is a homeomorphism of $\mathbb{R}^d$, and

$$\Psi(\Omega) = \Omega \quad \text{and} \quad \Psi(\Omega^\circ) = \Omega^\circ.$$

Proof. For all $x, y \in \mathbb{R}^d$, we have that

$$\|\Psi(x) - \Psi(y)\| \geq \|x - y\| - |\varepsilon| \|v(x) - v(y)\| \geq (1 - |\varepsilon| K_v) \|x - y\|.$$

Thus, Ψ is injective, and its inverse is $(1 - |\varepsilon| K_v)^{-1}$-Lipschitz on $\Psi\big(\mathbb{R}^d\big)$. Let $y \in \mathbb{R}^d$. The map $x \mapsto y - \varepsilon v(x)$ is a contraction of $\mathbb{R}^d$. By the Banach fixed point theorem, it has a fixed point x, i.e. $\Psi(x) = y$. Thus, Ψ is a homeomorphism of $\mathbb{R}^d$.

The same holds for $\Psi_t = I_d + t\varepsilon v$, for every $t \in [0, 1]$. Let $x \in \mathbb{R}^d \setminus \partial\Omega$ and $t \in [0, 1]$. Let $z \in \partial\Omega$. As $v(z) = 0$, we have that $\Psi_t(z) = z$. As $x \neq z$ and Ψ_t is injective, $\Psi_t(x) \neq \Psi_t(z) = z$. Thus, $\Psi_t(x) \notin \partial\Omega$. As t is arbitrary, the continuous path $\gamma : t \mapsto \Psi_t(x) = x + t\varepsilon v(x)$ lies in $\mathbb{R}^d \setminus \partial\Omega$. As Ω is closed, this set is the disjoint union of the open sets Ω° and $\mathbb{R}^d \setminus \Omega$. Thus, $\gamma^{-1}(\Omega^\circ)$ and $\gamma^{-1}\big(\mathbb{R}^d \setminus \Omega\big)$ are disjoint open subsets of $[0, 1]$, whose union is $[0, 1]$. As $[0, 1]$ is connected, one of them is empty, i.e. γ stays in Ω° or stays in $\mathbb{R}^d \setminus \Omega$. In particular, $\Psi(x) = \gamma(1)$ lies in Ω° if and only if $x = \gamma(0)$ does. Hence, Ψ maps each of the three disjoint sets Ω°, $\partial\Omega$ and $\mathbb{R}^d \setminus \Omega$ into itself. As these sets cover $\mathbb{R}^d$ and Ψ is a bijection, these inclusions are equalities. Thus, $\Psi(\Omega^\circ) = \Omega^\circ$ and $\Psi(\Omega) = \Psi(\Omega^\circ \cup \partial\Omega) = \Omega$, which concludes the proof.

□

The next lemma expresses the Kullback–Leibler divergence of a transported measure with respect to π. For a map Ψ differentiable at x, let $\det D\Psi(x)$ be the Jacobian determinant of Ψ at x.

Lemma A.3.2 – *Transport of the Kullback–Leibler divergence*
Suppose that Assumption A.3.1 holds. Let Ψ be a homeomorphism of $\mathbb{R}^d$, differentiable at every point of Ω, such that $\Psi(\Omega) = \Omega$ and $\det D\Psi(x) > 0$ for every $x \in \Omega$. Let ρ be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho \parallel \pi) < \infty$, and such that $\log\det D\Psi$ is ρ-integrable. Then, $\mathrm{KL}(\Psi\#\rho \parallel \pi) < \infty$ and

$$\mathrm{KL}(\Psi\#\rho \parallel \pi) = \mathrm{KL}(\rho \parallel \pi) - \int (\log p(\Psi(x)) - \log p(x) + \log\det D\Psi(x)) \,\mathrm{d}\rho(x).$$

Proof. The Kullback–Leibler divergence is invariant under the bijection Ψ. Indeed, let ρ' be a probability measure on $\mathbb{R}^d$. As Ψ and Ψ^{-1} are measurable, $\rho \ll \rho'$ if and only if $\Psi\#\rho \ll \Psi\#\rho'$, and then $\frac{\mathrm{d}\Psi\#\rho}{\mathrm{d}\Psi\#\rho'} = \frac{\mathrm{d}\rho}{\mathrm{d}\rho'} \circ \Psi^{-1}$. Thus, $\mathrm{KL}(\Psi\#\rho \parallel \Psi\#\rho') = \mathrm{KL}(\rho \parallel \rho')$. With $\rho' = \Psi^{-1}\#\pi$, we obtain

$$\mathrm{KL}(\Psi\#\rho \parallel \pi) = \mathrm{KL}(\rho \parallel \Psi^{-1}\#\pi).$$

Let $q(x) = p(\Psi(x)) \det D\Psi(x)/p(x)$ for $x \in \Omega$, and $q(x) = 1$ otherwise. Let $A \in \mathcal{B}\big(\mathbb{R}^d\big)$. As Ψ is a bijection with $\Psi(\Omega) = \Omega$, we have that $\Psi(A) \cap \Omega = \Psi(A \cap \Omega)$. The map Ψ is injective and differentiable at every point of the measurable set $A \cap$

Ω. This is all that the change of variables formula of (Fremlin, 2003, Section 263) requires. As $\det D\Psi > 0$ on Ω, it gives

$$\Psi^{-1}\#\pi(A) = \pi(\Psi(A)) = \int_{\Psi(A\cap\Omega)} p \,\mathrm{d}\lambda = \int_{A\cap\Omega} p(\Psi(x)) \det D\Psi(x) \,\mathrm{d}\lambda(x) = (q\cdot\pi)(A).$$

Thus, $\Psi^{-1}\#\pi = q\cdot\pi$, which is a probability measure. As $\Psi(\Omega) = \Omega$ and $p > 0$ on Ω, $q > 0$ on Ω. Thus, $\pi \ll q\cdot\pi$ and $\frac{\mathrm{d}\pi}{\mathrm{d}(q\cdot\pi)} = 1/q$, π-almost everywhere. As $\rho \ll \pi$, we have that $\rho \ll q\cdot\pi$ and

$$\log\frac{\mathrm{d}\rho}{\mathrm{d}(q\cdot\pi)} = \log\frac{\mathrm{d}\rho}{\mathrm{d}\pi} - \log q, \quad \rho\text{-almost everywhere.}$$

As $\rho \in \mathcal{P}(\Omega)$, we have that $\log q(x) = \log p(\Psi(x)) - \log p(x) + \log\det D\Psi(x)$ for ρ-almost every x. The functions $\log p\circ\Psi$ and $\log p$ are bounded on Ω, as p reaches a positive minimum and a maximum on the compact set Ω, and $\Psi(\Omega) = \Omega$. Thus, $\log q$ is ρ-integrable. As $\log\left(\frac{\mathrm{d}\rho}{\mathrm{d}\pi}\right)$ is ρ-integrable, so is $\log\left(\frac{\mathrm{d}\rho}{\mathrm{d}(q\cdot\pi)}\right)$. Therefore,

$$\mathrm{KL}(\Psi\#\rho \parallel \pi) = \mathrm{KL}(\rho \parallel q\cdot\pi) = \mathrm{KL}(\rho \parallel \pi) - \int \log q \,\mathrm{d}\rho < \infty,$$

which concludes the proof.

□

We can now differentiate the Kullback–Leibler divergence along a perturbation of the identity. Its derivative at the identity is called the *first variation* of the Kullback–Leibler divergence.

Theorem A.3.3 – *First variation of the Kullback–Leibler divergence*

Suppose that Assumption A.3.1 holds. Let $v : \mathbb{R}^d \to \mathbb{R}^d$ be continuously differentiable and K_v-Lipschitz, with $v(y) = 0$ for every $y \in \partial\Omega$. Let ρ be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho \parallel \pi) < \infty$. Then, $\mathcal{A}_s v$ is ρ-integrable, and there exists $\varepsilon_0 > 0$ such that

$$\forall\varepsilon \in (-\varepsilon_0, \varepsilon_0), \quad \mathrm{KL}((I_d + \varepsilon v)\#\rho \parallel \pi) < \infty.$$

Moreover, the function $h : \varepsilon \mapsto \mathrm{KL}((I_d + \varepsilon v)\#\rho \parallel \pi)$ is differentiable at 0, and

$$h'(0) = -\mathbb{E}_{x\sim\rho}[\mathcal{A}_s v(x)].$$

Proof. As $\mathrm{KL}(\rho \parallel \pi) < \infty$, $\rho \in \mathcal{P}(\Omega)$. The function $\mathcal{A}_s v : x \mapsto s(x)(v(x)) + \nabla\cdot v(x)$ is continuous on the compact set Ω, which carries ρ. Thus, it is ρ-integrable.

Transport As Dv is continuous and Ω is compact, $Dv(\Omega)$ is compact. The map $(\tau, A) \mapsto \det(I_d + \tau A)$ is continuous and equal to 1 at $\tau = 0$. Thus, there exists $\varepsilon_0 \in (0, 1]$ such that $\varepsilon_0 K_v < 1$ and

$$\forall\tau \in [-\varepsilon_0, \varepsilon_0], \forall A \in Dv(\Omega), \quad \det(I_d + \tau A) \geq \frac{1}{2}.$$

Let $\varepsilon \in (-\varepsilon_0, \varepsilon_0)$ and $\Psi_\varepsilon = I_d + \varepsilon v$. By Lemma A.3.1, Ψ_ε is a homeomorphism of $\mathbb{R}^d$ such that $\Psi_\varepsilon(\Omega) = \Omega$ and $\Psi_\varepsilon(\Omega^\circ) = \Omega^\circ$. It is differentiable, with $D\Psi_\varepsilon(x) = I_d +$

$\varepsilon Dv(x)$, so that $\det D\Psi_\varepsilon \geq 1/2$ on Ω. The function $\log\det D\Psi_\varepsilon$ is continuous on the compact set Ω, which carries ρ. Thus, it is ρ-integrable, and Lemma A.3.2 gives $\mathrm{KL}((\Psi_\varepsilon)\#\rho \parallel \pi) < \infty$ and

$$h(\varepsilon) = \mathrm{KL}(\rho \parallel \pi) - \int \ell(\varepsilon, x)\,\mathrm{d}\rho(x),$$

where $\ell(\varepsilon, x) = \log p(x + \varepsilon v(x)) - \log p(x) + \log\det(I_d + \varepsilon Dv(x))$.

Differentiation Let $x \in \Omega$, which holds for ρ-almost every x, and let $\varepsilon \in (-\varepsilon_0, \varepsilon_0)$. If $x \in \Omega^\circ$, then $x + \varepsilon v(x) \in \Omega^\circ$ by Lemma A.3.1. As $\log p$ is differentiable at $x + \varepsilon v(x)$, with derivative $s(x + \varepsilon v(x))$, the chain rule gives

$$\partial_\varepsilon \log p(x + \varepsilon v(x)) = s(x + \varepsilon v(x))(v(x)).$$

Otherwise, $x \in \partial\Omega$, so that $v(x) = 0$ and both sides vanish. Let $D\det(Y)(Z)$ be the derivative of the determinant at the matrix Y in the direction Z. As $\det(I_d + \varepsilon Dv(x)) \geq 1/2 > 0$, we obtain

$$\partial_\varepsilon \ell(\varepsilon, x) = s(x + \varepsilon v(x))(v(x)) + \frac{D\det(I_d + \varepsilon Dv(x))(Dv(x))}{\det(I_d + \varepsilon Dv(x))}.$$

Let M_v be the maximum of $\|v\|$ on Ω, and let M be the maximum of $|D\det(I_d + \tau A)(A)|$ for $(\tau, A) \in [-1, 1] \times Dv(\Omega)$. They are finite by continuity on compact sets. As $x + \varepsilon v(x) \in \Omega$, we have that $|\partial_\varepsilon \ell(\varepsilon, x)| \leq M_s M_v + 2M$. This constant is ρ-integrable, as ρ is a probability measure. Moreover, $\ell(0, \cdot) = 0$, and every $\ell(\varepsilon, \cdot)$ is continuous on Ω, which carries ρ. By the Leibniz integral rule (Folland, 1999, Theorem 2.27), $\varepsilon \mapsto \int \ell(\varepsilon, x)\,\mathrm{d}\rho(x)$ is differentiable at 0, with derivative $\int \partial_\varepsilon \ell(0, x)\,\mathrm{d}\rho(x)$. Finally, Jacobi's formula at the identity, $D\det(I_d)(A) = \operatorname{tr} A$, gives

$$\partial_\varepsilon \ell(0, x) = s(x)(v(x)) + \operatorname{tr} Dv(x) = \mathcal{A}_s v(x),$$

which concludes the proof.

□

Stein's identity for the target distribution follows, as the Kullback–Leibler divergence is minimal at π.

Corollary A.3.2 – *Stein's identity for the target distribution*

Suppose that Assumption A.3.1 holds. Let $v : \mathbb{R}^d \to \mathbb{R}^d$ be continuously differentiable and Lipschitz, with $v(y) = 0$ for every $y \in \partial\Omega$. Then, $\mathcal{A}_s v$ is π-integrable and

$$\mathbb{E}_{x\sim\pi}[\mathcal{A}_s v(x)] = 0.$$

Proof. As $\mathrm{KL}(\pi \parallel \pi) = 0 < \infty$, Theorem A.3.3 applies to $\rho = \pi$. Thus, $\mathcal{A}_s v$ is π-integrable, and $h : \varepsilon \mapsto \mathrm{KL}((I_d + \varepsilon v)\#\pi \parallel \pi)$ is real-valued on $(-\varepsilon_0, \varepsilon_0)$ and differentiable at 0, with $h'(0) = -\mathbb{E}_{x\sim\pi}[\mathcal{A}_s v(x)]$. As the Kullback–Leibler divergence is nonnegative, $h \geq 0 = h(0)$. Thus, h has a minimum at 0, and $h'(0) = 0$ by Fermat's theorem on stationary points, which concludes the proof.

□

Thus, every continuously differentiable and Lipschitz vector field that vanishes on $\partial\Omega$ satisfies Stein's identity for π, without the divergence theorem and without any regularity of $\partial\Omega$. The boundary is handled by the vector field itself, which vanishes on $\partial\Omega$.

3.3.3.3 The Stein discrepancy

Let $\mathcal{F}$ be a set of continuously differentiable and Lipschitz vector fields that vanish on $\partial\Omega$. By Corollary A.3.2, $\mathbb{E}_{x\sim\pi}[\mathcal{A}_s v(x)] = 0$ for every $v \in \mathcal{F}$. Now, one can consider this expectation for another measure $\rho \in \mathcal{P}(\Omega)$. If $\rho \neq \pi$, it is no longer zero in general. In fact, its magnitude relates to how different ρ and π are. It is used to define a discrepancy measure, known as the Stein discrepancy (Gorham & Mackey, 2015). The latter considers the maximum violation of Stein's identity over $\mathcal{F}$, i.e.

$$\mathbb{S}(\rho, \pi) = \sup_{v\in\mathcal{F}} \mathbb{E}_{x\sim\rho}[\mathcal{A}_s v(x)].$$

Note that $\mathbb{S}(\rho, \pi)$ is not symmetric. The choice of $\mathcal{F}$ determines the discriminative power and the tractability of the Stein discrepancy. In Section A.3.3.4, $\mathcal{F}$ is the unit ball of a space of vector fields defined by a kernel, as in Liu et al. (2016). This choice gives a closed-form expression of the square of the supremum, known as the *kernelized Stein discrepancy*.

3.3.4 The kernelized Stein discrepancy

In this section, we define the kernel and the space of vector fields of SVGD, in a setting that contains that of Liu & Wang (2016). Then, we define the kernelized Stein discrepancy of Liu et al. (2016) in this setting, and we deduce its closed-form expression through the Stein witness.

3.3.4.1 The kernel

In Liu & Wang (2016), the vector fields of SVGD lie in a product of reproducing kernel Hilbert spaces (RKHS). Here, the kernel is only given by a feature map Φ, with values in a real Hilbert space H_0. The space H_0 need not be an RKHS. This setting only uses the chain rule to differentiate the vector fields of $\mathcal{H}$ (see below), whereas the RKHS setting requires a reproducing property for the derivatives.

Assumption A.3.2 – *Kernel*

Let H_0 be a real Hilbert space, let $\Phi : \mathbb{R}^d \to H_0$ and let $K, C \geq 0$. The kernel is

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{H_0}, \quad x, y \in \mathbb{R}^d.$$

The following conditions hold.

(i) Φ is continuously differentiable on $\mathbb{R}^d$.
(ii) Φ is K-Lipschitz, i.e. $\|\Phi(x) - \Phi(y)\|_{H_0} \leq K\|x - y\|$ for all $x, y \in \mathbb{R}^d$.
(iii) $\|\Phi(y)\|_{H_0} \leq C$ for every $y \in \mathbb{R}^d$.
(iv) $\Phi(y) = 0$ for every $y \in \partial\Omega$.

Let $\mathcal{H} = H_0^d$ be the space of the d-tuples $f = (f_1, ..., f_d)$ of elements of H_0, with the inner product

$$\langle f, g \rangle_{\mathcal{H}} = \sum_{i=1}^{d} \langle f_i, g_i \rangle_{H_0}.$$

It is a Hilbert space. The elements of H_0 are abstract vectors, and need not be functions on $\mathbb{R}^d$. Thus, an element $f \in \mathcal{H}$ is not a vector field by itself. It defines one through the feature map, as each component f_i is paired with the feature $\Phi(y)$ of the point y. The vector field $f_\Phi : \mathbb{R}^d \to \mathbb{R}^d$ of f is defined by

$$f_\Phi(y) = \big(\langle f_1, \Phi(y) \rangle_{H_0}, ..., \langle f_d, \Phi(y) \rangle_{H_0} \big).$$

In other words, each coordinate of f_Φ is a linear model in the features, whose coefficients are given by f_i. The subscript Φ distinguishes the element f of $\mathcal{H}$ from its vector field f_Φ on $\mathbb{R}^d$.

Example

Let $d = 1$, $H_0 = \mathbb{R}^3$ and $\Phi(y) = (1, y, y^2)$. An element $f = (a, b, c)$ of $\mathcal{H} = \mathbb{R}^3$ is a triple of coefficients. Its vector field is the polynomial $f_\Phi(y) = a + by + cy^2$, and the kernel is $k(x, y) = 1 + xy + x^2y^2$. This feature map only illustrates the notation, as it does not satisfy Assumption A.3.2.

This setting contains the product RKHS of Liu & Wang (2016), as a special case. Let k_0 be a kernel on $\mathbb{R}^d$, let H_0 be its RKHS (Aronszajn, 1950), and let $\Phi(y) = k_0(\cdot, y)$. By the reproducing property, $\langle \Phi(x), \Phi(y) \rangle_{H_0} = k_0(x, y)$ and $f_\Phi(y) = (f_1(y), ..., f_d(y))$ for every $f \in \mathcal{H}$. Thus, $f_\Phi = f$, i.e. the subscript Φ only evaluates the functions f_i at y. In this case, Assumption A.3.2 is a condition on the map $y \mapsto k_0(\cdot, y)$. Conversely, a feature map defines the kernel k, but H_0 need not be the RKHS of k. Thus, $\mathcal{H}$ is not a product RKHS in general. No strict positive definiteness of k is required either, such as the integral strict positive definiteness used by Liu et al. (2016).

In general, we have that $f_\Phi = P_f \circ \Phi$, where $P_f : u \mapsto \big(\langle f_i, u \rangle_{H_0} \big)_{1 \leq i \leq d}$ is a linear map from H_0 to $\mathbb{R}^d$. By the Cauchy–Schwarz inequality in H_0, we have that

$$\forall u \in H_0, \quad \big\| P_f(u) \big\|^2 = \sum_{i=1}^{d} \langle f_i, u \rangle_{H_0}^2 \leq \sum_{i=1}^{d} \|f_i\|_{H_0}^2 \|u\|_{H_0}^2 = \|f\|_{\mathcal{H}}^2 \|u\|_{H_0}^2.$$

Taking square roots, $\big\| P_f(u) \big\| \leq \|f\|_{\mathcal{H}} \|u\|_{H_0}$ for every $u \in H_0$. Thus, the linear map P_f is continuous, and its operator norm is at most $\|f\|_{\mathcal{H}}$. By (i) of Assumption A.3.2 and the chain rule, f_Φ is continuously differentiable, with $Df_\Phi(y) = P_f \circ D\Phi(y)$. As P_f is linear, we have that

$$\forall x, y \in \mathbb{R}^d, \quad \|f_\Phi(x) - f_\Phi(y)\| = \big\| P_f(\Phi(x) - \Phi(y)) \big\| \leq \|f\|_{\mathcal{H}} \|\Phi(x) - \Phi(y)\|_{H_0}.$$

Thus, by (ii) of Assumption A.3.2, f_Φ is $\|f\|_{\mathcal{H}} K$-Lipschitz. Likewise, $\|f_\Phi(y)\| \leq \|f\|_{\mathcal{H}} \|\Phi(y)\|_{H_0}$, so that f_Φ is bounded by $\|f\|_{\mathcal{H}} C$ by (iii), and zero on $\partial\Omega$ by (iv).

Conditions (i) and (ii) of Assumption A.3.2 make every vector field f_Φ continuously differentiable and Lipschitz. Thus, Theorem A.3.3 and Corollary A.3.2 apply to f_Φ. In Section A.3.3.6, these conditions also give global solutions to the ordinary

differential equations driven by these fields, by the Cauchy–Lipschitz theorem. Condition (iii) bounds f_Φ uniformly on $\mathbb{R}^d$. Condition (iv) makes every f_Φ vanish on $\partial\Omega$. By Lemma A.3.1, $I_d + \varepsilon f_\Phi$ then maps Ω onto Ω when $|\varepsilon|\|f\|_{\mathcal{H}} K < 1$ (see Figure A.3.17). This requires no regularity of $\partial\Omega$.

Translation-invariant kernels, such as the Gaussian kernel, do not satisfy condition (iv) (see Figure A.3.18). Indeed, for such a kernel, $k(y, y) = \|\Phi(y)\|^2_{H_0}$ is a constant, which is positive unless $k = 0$. Weighted kernels $k(x, y) = w(x)w(y)k_0(x, y)$, where $w : \mathbb{R}^d \to \mathbb{R}$ vanishes on $\partial\Omega$, satisfy it. Their feature map is $\Phi = w\Phi_0$, where Φ_0 is a feature map of k_0.

In the rest of this section, Assumption A.3.1 and Assumption A.3.2 hold, unless stated otherwise. We call them the *standing assumptions*.

3.3.4.2 Definition of the kernelized Stein discrepancy

The kernelized Stein discrepancy is the Stein discrepancy of Section A.3.3.3.3, with the vector fields f_Φ of the closed unit ball of $\mathcal{H}$.

Definition A.3.11 – *Kernelized Stein discrepancy*

Suppose that the standing assumptions hold. Let $\rho \in \mathcal{P}(\Omega)$. The kernelized Stein discrepancy (KSD) of ρ with respect to π is

$$\mathfrak{K}(\rho \mid \pi) = \sup_{f \in \mathcal{H},\, \|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)]^2.$$

Let $f \in \mathcal{H}$. As s is continuous on Ω and f_Φ is continuously differentiable, $\mathcal{A}_s f_\Phi : x \mapsto s(x)(f_\Phi(x)) + \nabla \cdot f_\Phi(x)$ is continuous on Ω. Recall that $M_s = \max_{x\in\Omega} \|s(x)\|$ (see Section A.3.3.2). As f_Φ is $\|f\|_{\mathcal{H}}K$-Lipschitz, $\|Df_\Phi(x)\| \leq \|f\|_{\mathcal{H}} K$. The trace of a linear map L of $\mathbb{R}^d$ satisfies $|\operatorname{tr} L| \leq d\|L\|$. Thus,

$$\forall x \in \Omega, \quad |\mathcal{A}_s f_\Phi(x)| \leq \|s(x)\|\|f_\Phi(x)\| + d\|Df_\Phi(x)\| \leq \|f\|_{\mathcal{H}}(M_s C + dK).$$

As ρ is carried by Ω, every expectation in Definition A.3.11 is finite, and $0 \leq \mathfrak{K}(\rho \mid \pi) \leq (M_s C + dK)^2$. The KSD depends on π only through its score s. Let $\mathcal{F}$ be the set of the vector fields f_Φ with $\|f\|_{\mathcal{H}} \leq 1$. They are continuously differentiable and Lipschitz, and they vanish on $\partial\Omega$. As the closed unit ball of $\mathcal{H}$ is symmetric and $f \mapsto \mathcal{A}_s f_\Phi$ is linear, we have that $\mathfrak{K}(\rho \mid \pi) = \mathbb{S}(\rho, \pi)^2$. In particular, $\mathfrak{K}(\pi \mid \pi) = 0$ by Corollary A.3.2.

The construction of $\mathcal{H}$ is motivated by the fact that the supremum in Definition A.3.11 has a closed-form expression. This supremum is reached at an explicit element of $\mathcal{H}$, which we compute in Section A.3.3.4.3.

3.3.4.3 The Stein witness

To find this direction, we write the Stein operator of f_Φ as an inner product in $\mathcal{H}$. Let $(e_1, ..., e_d)$ be the canonical basis of $\mathbb{R}^d$. For $x \in \mathbb{R}^d$, let $\xi_x \in \mathcal{H}$ be defined by

$$(\xi_x)_i = s(x)(e_i)\Phi(x) + D\Phi(x)(e_i), \quad 1 \leq i \leq d.$$

We call ξ_x the *Stein feature* of x. The next lemma links ξ_x to the Stein operator of f_Φ. It is the key step of the closed-form expression of the KSD.

Lemma A.3.3 – *Stein feature*

Suppose that the standing assumptions hold. Then,

$$\forall x \in \mathbb{R}^d, \forall f \in \mathcal{H}, \quad \mathcal{A}_s f_\Phi(x) = \langle f, \xi_x \rangle_{\mathcal{H}}.$$

Proof. Let $x \in \mathbb{R}^d$ and $f \in \mathcal{H}$. By (i) of Assumption A.3.2, Φ is differentiable at x. As P_f is linear and continuous, the chain rule gives that $f_\Phi = P_f \circ \Phi$ is differentiable at x, with $Df_\Phi(x) = P_f \circ D\Phi(x)$. The divergence is the trace of the derivative. As $e_i^\top P_f(u) = \langle f_i, u \rangle_{H_0}$ for every $u \in H_0$, we have that

$$\nabla \cdot f_\Phi(x) = \sum_{i=1}^d e_i^\top Df_\Phi(x)(e_i) = \sum_{i=1}^d e_i^\top P_f(D\Phi(x)(e_i)) = \sum_{i=1}^d \langle f_i, D\Phi(x)(e_i) \rangle_{H_0}.$$

As $f_\Phi(x) = \sum_{i=1}^d \langle f_i, \Phi(x) \rangle_{H_0} e_i$ and $s(x)$ is linear, we have that

$$\begin{aligned}
\mathcal{A}_s f_\Phi(x) &= s(x)(f_\Phi(x)) + \nabla \cdot f_\Phi(x) \\
&= \sum_{i=1}^d \Big(\langle f_i, \Phi(x) \rangle_{H_0} s(x)(e_i) + \langle f_i, D\Phi(x)(e_i) \rangle_{H_0} \Big) \\
&= \sum_{i=1}^d \langle f_i, s(x)(e_i)\Phi(x) + D\Phi(x)(e_i) \rangle_{H_0} \\
&= \langle f, \xi_x \rangle_{\mathcal{H}},
\end{aligned}$$

which concludes the proof.

□

Remark

In the RKHS case, i.e. when $\Phi(x) = k_0(\cdot, x)$, we have that $f_\Phi = f$ for every $f \in \mathcal{H}$, and the Stein feature is $\xi_x = \nabla \log p(x) k_0(\cdot, x) + \nabla_x k_0(\cdot, x)$ for $x \in \Omega^\circ$, as in Liu et al. (2016) and Liu & Wang (2016). Thus, Lemma A.3.3 reads

$$\forall x \in \mathbb{R}^d, \forall f \in \mathcal{H}, \quad \mathcal{A}_s f(x) = \langle f, \xi_x \rangle_{\mathcal{H}}.$$

The next lemma gives the regularity of ξ on Ω.

Lemma A.3.4 – *Regularity of* ξ

Suppose that the standing assumptions hold. Then, the following holds.

(i) The map $x \mapsto \xi_x$ is uniformly continuous from Ω to $\mathcal{H}$.

(ii) $B = \max_{x \in \Omega} \|\xi_x\|_{\mathcal{H}}$ is finite.

(iii) For every $\rho \in \mathcal{P}(\Omega)$, the map $x \mapsto \xi_x$ is ρ-integrable.

Proof. By (iv) of Assumption A.3.1, s is continuous on Ω, and so is $x \mapsto s(x)(e_i)$. By (i) of Assumption A.3.2, Φ and $x \mapsto D\Phi(x)$ are continuous, and so is $x \mapsto D\Phi(x)(e_i)$. Thus, every component $x \mapsto (\xi_x)_i$ is continuous on Ω. As $\|f\|_{\mathcal{H}}^2 = \sum_{i=1}^d \|f_i\|_{H_0}^2$, the map $x \mapsto \xi_x$ is continuous from Ω to $\mathcal{H}$. As Ω is compact, the Heine–Cantor theorem gives (i). As Ω is nonempty and compact, the continuous function $x \mapsto \|\xi_x\|_{\mathcal{H}}$ reaches its maximum on Ω, which gives (ii). Finally, $x \mapsto \xi_x$

is ρ-integrable, as it is continuous and bounded by B on Ω, which carries the probability measure ρ. Here, the integral of a function with values in $\mathcal{H}$ is the Bochner integral, which extends the expectation to functions with values in a complete normed space. As ξ is continuous on the compact set Ω, its image is compact, hence separable, so that ξ is measurable in the sense of this integral.

□

We can now deduce this direction, and the closed-form expression of the KSD.

Theorem A.3.4 – *Stein witness*

Suppose that the standing assumptions hold. Let $\rho \in \mathcal{P}(\Omega)$. Then, there exists a unique $\phi_\rho^* \in \mathcal{H}$ such that

$$\forall f \in \mathcal{H}, \quad \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = \langle f, \phi_\rho^* \rangle_\mathcal{H}.$$

It is the Bochner integral $\phi_\rho^* = \mathbb{E}_{x\sim\rho}[\xi_x]$. Moreover, the following holds.

(i) On the closed unit ball of $\mathcal{H}$, the function $f \mapsto \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)]$ reaches its maximum at $f^* = \phi_\rho^* / \|\phi_\rho^*\|_\mathcal{H}$ if $\phi_\rho^* \neq 0$, and at $f^* = 0$ otherwise.

(ii) $\mathfrak{K}(\rho \mid \pi) = \|\phi_\rho^*\|_\mathcal{H}^2 = \mathbb{E}_{x\sim\rho}\left[\mathcal{A}_s (\phi_\rho^*)_\Phi(x)\right]$.

The proof strategy is to remark that, for any $f \in \mathcal{H}$, $\mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = \langle f, \mathbb{E}_{x\sim\rho}[\xi_x] \rangle_\mathcal{H}$. Then, the result follows from the Cauchy–Schwarz inequality.

Proof. By (iii) of Lemma A.3.4, $x \mapsto \xi_x$ is ρ-integrable. As $\mathcal{H}$ is a Hilbert space, it is complete, so that the Bochner integral $\phi_\rho^* = \mathbb{E}_{x\sim\rho}[\xi_x]$ is well defined. Let $f \in \mathcal{H}$. The continuous linear form $\langle f, \cdot \rangle_\mathcal{H}$ commutes with the Bochner integral. By Lemma A.3.3, we have that

$$\langle f, \phi_\rho^* \rangle_\mathcal{H} = \mathbb{E}_{x\sim\rho}[\langle f, \xi_x \rangle_\mathcal{H}] = \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)].$$

This gives the existence of ϕ_ρ^*. We now show that it is unique. Let $f' \in \mathcal{H}$ such that

$$\forall f \in \mathcal{H}, \quad \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = \langle f, f' \rangle_\mathcal{H}.$$

Let $f \in \mathcal{H}$. By the assumption on f' and the identity above, we have that

$$\langle f, f' \rangle_\mathcal{H} = \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = \langle f, \phi_\rho^* \rangle_\mathcal{H}.$$

As the inner product is linear in its second argument, $\langle f, f' - \phi_\rho^* \rangle_\mathcal{H} = \langle f, f' \rangle_\mathcal{H} - \langle f, \phi_\rho^* \rangle_\mathcal{H} = 0$. This holds for every $f \in \mathcal{H}$, in particular for $f = f' - \phi_\rho^*$. Thus,

$$\|f' - \phi_\rho^*\|_\mathcal{H}^2 = \langle f' - \phi_\rho^*, f' - \phi_\rho^* \rangle_\mathcal{H} = 0.$$

As a norm vanishes only at 0, we have that $f' - \phi_\rho^* = 0$, i.e. $f' = \phi_\rho^*$.

Let $f \in \mathcal{H}$ such that $\|f\|_\mathcal{H} \leq 1$. By the Cauchy–Schwarz inequality, we have that

$$\langle f, \phi_\rho^* \rangle_\mathcal{H} \leq |\langle f, \phi_\rho^* \rangle_\mathcal{H}| \leq \|f\|_\mathcal{H} \|\phi_\rho^*\|_\mathcal{H} \leq \|\phi_\rho^*\|_\mathcal{H}.$$

Moreover, $\|f^*\|_\mathcal{H} \leq 1$ and $\langle f^*, \phi_\rho^* \rangle_\mathcal{H} = \|\phi_\rho^*\|_\mathcal{H}$, in both cases. This gives (i). We also have that

$$\mathfrak{K}(\rho \mid \pi) = \sup_{f \in \mathcal{H},\, \|f\|_{\mathcal{H}} \leq 1} \langle f, \phi_\rho^* \rangle_{\mathcal{H}}^2.$$

Every term is at most $\left\|\phi_\rho^*\right\|_{\mathcal{H}}^2$, and the term of f^* is equal to it, so that $\mathfrak{K}(\rho \mid \pi) = \left\|\phi_\rho^*\right\|_{\mathcal{H}}^2$. Finally, the identity above with $f = \phi_\rho^*$ gives

$$\left\|\phi_\rho^*\right\|_{\mathcal{H}}^2 = \langle \phi_\rho^*, \phi_\rho^* \rangle_{\mathcal{H}} = \mathbb{E}_{x \sim \rho}\left[\mathcal{A}_s\left(\phi_\rho^*\right)_\Phi(x)\right].$$

This gives (ii), which concludes the proof.

□

We call ϕ_ρ^* the *Stein witness* of ρ, by analogy with the witness function of the maximum mean discrepancy (Gretton et al., 2012). The latter is the function at which the supremum that defines this discrepancy is reached. Up to normalization, it is the direction of $\mathcal{H}$ along which ρ violates Stein's identity the most. In Section A.3.3.5, we show that it is also the direction of steepest descent of the Kullback–Leibler divergence. As $\|\xi_x\|_{\mathcal{H}} \leq B$ for ρ-almost every x, we have that $\left\|\phi_\rho^*\right\|_{\mathcal{H}} \leq B$. By (iii) of Assumption A.3.2 and the bound on P_f, we have that

$$\forall y \in \mathbb{R}^d, \quad \left\|\left(\phi_\rho^*\right)_\Phi(y)\right\| \leq \left\|\phi_\rho^*\right\|_{\mathcal{H}} \|\Phi(y)\|_{H_0} \leq BC.$$

The vector field of ϕ_ρ^* is thus bounded by a constant that does not depend on ρ. Figure A.3.18 shows this vector field in dimension 1.

As $\mathfrak{K}(\pi \mid \pi) = 0$, (ii) of Theorem A.3.4 gives $\phi_\pi^* = 0$, i.e. the Stein witness of π vanishes.

By (ii) of Theorem A.3.4, the KSD is expressed with the feature map Φ, whose values may lie in an infinite-dimensional space. It can also be computed from the kernel k and the score s only. Let $x, y \in \mathbb{R}^d$ and $1 \leq i \leq d$. As $k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{H_0}$ and Φ is continuously differentiable, we have that

$$\begin{aligned}
\partial_{x_i} k(x, y) &= \langle D\Phi(x)(e_i), \Phi(y) \rangle_{H_0}, \\
\partial_{y_i} k(x, y) &= \langle \Phi(x), D\Phi(y)(e_i) \rangle_{H_0}, \\
\partial_{x_i} \partial_{y_i} k(x, y) &= \langle D\Phi(x)(e_i), D\Phi(y)(e_i) \rangle_{H_0}.
\end{aligned}$$

Expanding the inner product of the Stein features, we obtain

$$\langle \xi_x, \xi_y \rangle_{\mathcal{H}} = k(x, y) \langle s(x), s(y) \rangle + \langle s(x), \nabla_y k(x, y) \rangle + \langle s(y), \nabla_x k(x, y) \rangle + \sum_{i=1}^{d} \partial_{x_i} \partial_{y_i} k(x, y).$$

Here, $s(x)$ and $s(y)$ are identified with vectors of $\mathbb{R}^d$, and $\nabla_x k$ and $\nabla_y k$ are the gradients of k in its first and second variables. Let $\rho \in \mathcal{P}(\Omega)$. As $\phi_\rho^* = \mathbb{E}_{x \sim \rho}[\xi_x]$ and the inner product commutes with the Bochner integral, (ii) of Theorem A.3.4 gives

$$\mathfrak{K}(\rho \mid \pi) = \left\|\mathbb{E}_{x \sim \rho}[\xi_x]\right\|_{\mathcal{H}}^2 = \mathbb{E}_{x \sim \rho, y \sim \rho}\left[\langle \xi_x, \xi_y \rangle_{\mathcal{H}}\right],$$

where x and y are independent. This is the expression of the KSD of Liu et al. (2016), used to compute it in practice (see Figure A.3.19).

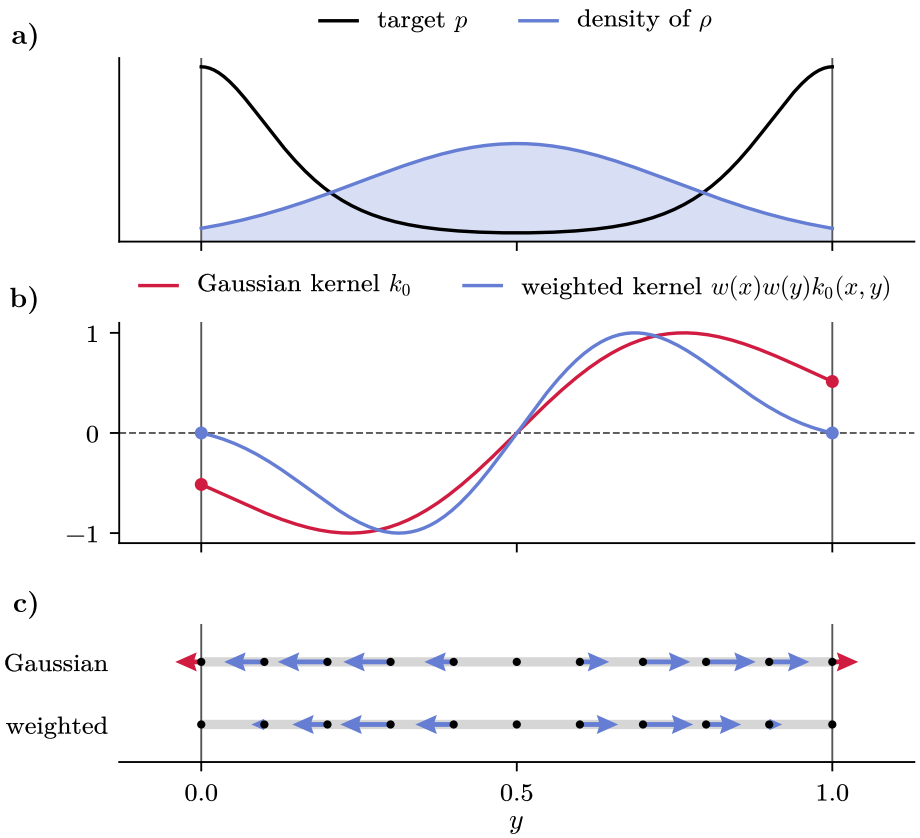


Figure A.3.18. *The vector field of the Stein witness on* $\Omega = [0,1]$. ***a)*** *The target density* $p \propto e^{-3f}$, *with* $f(x) = 16x^2(1-x)^2$, *whose modes are the endpoints of* Ω. *The density of* ρ *is a Gaussian of mean* 0.5 *and standard deviation* 0.25, *restricted to* Ω. ***b)*** *The field* $\left(\phi^*_\rho\right)_\Phi$ *for the Gaussian kernel* $k_0(x,y) = e^{-(x-y)^2/(2 \cdot 0.15^2)}$, *in red, and for the weighted kernel* $w(x)w(y)k_0(x,y)$ *with* $w(x) = x(1-x)$, *in blue. Each field is divided by its maximum absolute value on* Ω. *A positive value moves a particle to the right.* ***c)*** *The particles of a regular grid of* Ω, *and their displacements* $y \mapsto y + \varepsilon\left(\phi^*_\rho\right)_\Phi(y)$ *for a small step* $\varepsilon > 0$, *as in Figure A.3.17. The arrows are red when they leave* Ω. *Both fields move the particles toward the modes of* p. *With the Gaussian kernel, the particles located at* 0 *and* 1 *leave* Ω. *With the weighted kernel, the field vanishes at* 0 *and* 1, *and these particles do not move. Outside* Ω, *the weight* w *can be modified to be bounded and Lipschitz, as required by Assumption A.3.2, without changing the figure.*

3.3.5 The steepest descent of the KL divergence

In this section, we show that the Stein witness of Section A.3.3.4.3 is the direction of steepest descent of the Kullback–Leibler divergence to π, as in Liu & Wang (2016). This is the key of the SVGD algorithm.

Let ρ be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho \parallel \pi) < \infty$. For $f \in \mathcal{H}$ and $\varepsilon \in \mathbb{R}$, the map $I_d + \varepsilon f_\Phi$ moves each point $x \in \mathbb{R}^d$ to $x + \varepsilon f_\Phi(x)$. Let

$$h_f(\varepsilon) = \mathrm{KL}((I_d + \varepsilon f_\Phi)\#\rho \parallel \pi).$$

The derivative $h'_f(0)$ is the first variation of the Kullback–Leibler divergence at ρ along f. The next theorem computes it, and minimizes it over the closed unit ball of $\mathcal{H}$.

Theorem A.3.5 – *KL steepest descent*

Suppose that the standing assumptions hold. Let ρ be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho \parallel \pi) < \infty$. Then, the following holds.

(i) For every $f \in \mathcal{H}$, h_f is real-valued on a neighborhood of 0, differentiable at 0, and $h'_f(0) = -\langle f, \phi^*_\rho \rangle_{\mathcal{H}}$.

(ii) On the closed unit ball of $\mathcal{H}$, the function $f \mapsto h'_f(0)$ reaches its minimum at $f^* = \phi^*_\rho / \left\|\phi^*_\rho\right\|_{\mathcal{H}}$ if $\phi^*_\rho \neq 0$, and at $f^* = 0$ otherwise. If $\phi^*_\rho \neq 0$, f^* is its only minimizer on the closed unit ball.

(iii) $h'_{\phi^*_\rho}(0) = -\mathfrak{K}(\rho \mid \pi)$.

Proof. As $\mathrm{KL}(\rho \parallel \pi) < \infty$, we have that $\rho \in \mathcal{P}(\Omega)$ (see Section A.3.3.3.2). Thus, ϕ^*_ρ is given by Theorem A.3.4. Let $f \in \mathcal{H}$. By Section A.3.3.4.1, f_Φ is continuously differentiable, $\|f\|_{\mathcal{H}} K$-Lipschitz, and zero on $\partial\Omega$. Thus, Theorem A.3.3 applies to $v = f_\Phi$. It gives that h_f is real-valued on a neighborhood of 0, differentiable at 0, and

$$h'_f(0) = -\mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = -\langle f, \phi^*_\rho \rangle_{\mathcal{H}},$$

where the second equality is Theorem A.3.4. This gives (i).

By (i), minimizing $f \mapsto h'_f(0)$ on the closed unit ball amounts to maximizing $f \mapsto \langle f, \phi^*_\rho \rangle_{\mathcal{H}} = \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)]$. By (i) of Theorem A.3.4, f^* is a maximizer, and $\langle f^*, \phi^*_\rho \rangle_{\mathcal{H}} = \left\|\phi^*_\rho\right\|_{\mathcal{H}}$. Suppose that $\phi^*_\rho \neq 0$. Let $f \in \mathcal{H}$ such that $\|f\|_{\mathcal{H}} \leq 1$ and $\langle f, \phi^*_\rho \rangle_{\mathcal{H}} = \left\|\phi^*_\rho\right\|_{\mathcal{H}}$. Then, $\|f^*\|_{\mathcal{H}} = 1$ and $\langle f, f^* \rangle_{\mathcal{H}} = 1$. Thus,

$$\|f - f^*\|^2_{\mathcal{H}} = \|f\|^2_{\mathcal{H}} - 2\langle f, f^* \rangle_{\mathcal{H}} + \|f^*\|^2_{\mathcal{H}} = \|f\|^2_{\mathcal{H}} - 1 \leq 0,$$

so that $f = f^*$. This gives (ii). Finally, (i) with $f = \phi^*_\rho$ and (ii) of Theorem A.3.4 give

$$h'_{\phi^*_\rho}(0) = -\langle \phi^*_\rho, \phi^*_\rho \rangle_{\mathcal{H}} = -\left\|\phi^*_\rho\right\|^2_{\mathcal{H}} = -\mathfrak{K}(\rho \mid \pi),$$

which concludes the proof.

□

This result is the key of the SVGD algorithm. It means that ϕ^*_ρ is the optimal direction, within $\mathcal{H}$, to update ρ in order to decrease the Kullback–Leibler divergence to π. If $\phi^*_\rho \neq 0$, then $h'_{\phi^*_\rho}(0) = -\mathfrak{K}(\rho \mid \pi) < 0$, so that, for every $\varepsilon > 0$ small enough,

$$\mathrm{KL}\Big(\Big(I_d + \varepsilon\big(\phi^*_\rho\big)_\Phi\Big)\#\rho \parallel \pi\Big) < \mathrm{KL}(\rho \parallel \pi).$$

Consequently, SVGD iteratively updates ρ in the direction ϕ^*_ρ, with a small step size $\varepsilon > 0$, i.e.

$$\rho_{n+1} = \Big(I_d + \varepsilon\big(\phi^*_{\rho_n}\big)_\Phi\Big)\#\rho_n, \quad n \in \mathbb{N}. \tag{22}$$

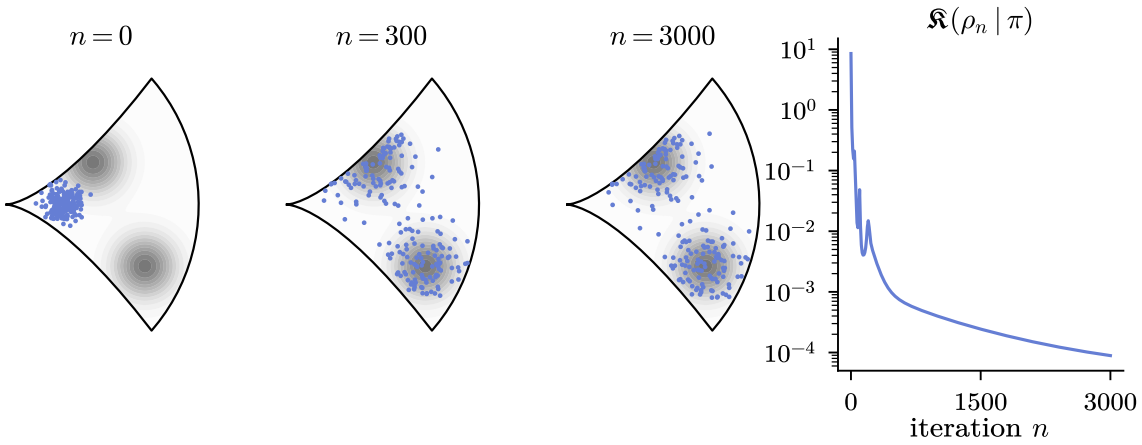


Figure A.3.19. *SVGD with $N = 200$ particles on the domain Ω of Figure A.3.17. The density of the target π is proportional to a mixture of two Gaussians of standard deviation 0.12, restricted to Ω (gray levels). The kernel is $k(z, z') = w(z)w(z')e^{-\|z-z'\|^2/(2 \cdot 0.3^2)}$, with $w(z) = c(z_1^3 - z_2^2)(1 - \|z\|^2)$ and $c \approx 5.38$, so that $\max_{\Omega} w = 1$. The weight w vanishes on $\partial\Omega$. Outside Ω, the weight can be modified to be bounded and Lipschitz, as required by Assumption A.3.2, without changing the figure. The particles start near the cusp and follow Eq. 22 with $\varepsilon = 0.08$.* ***Left.*** *The particles at the iterations $n = 0$, 300 and 3000. They spread over the two modes of π, and none of them leaves Ω.* ***Right.*** *The KSD of the empirical measure ρ_n, computed with its expression in terms of the kernel (see Section A.3.3.4.3), in which the expectation is a double sum over the particles. Empirical measures are not covered by the convergence result of Section A.3.3.8.*

As $\phi_\pi^* = 0$, π is a fixed point of Eq. 22. Let $\rho_0 \in \mathcal{P}(\Omega)$. For every $\rho \in \mathcal{P}(\Omega)$, $\left\|\phi_\rho^*\right\|_{\mathcal{H}} \leq B$, so that $\left(\phi_\rho^*\right)_\Phi$ is BK-Lipschitz. If $\varepsilon BK < 1$, Lemma A.3.1 shows that every map $I_d + \varepsilon\left(\phi_{\rho_n}^*\right)_\Phi$ maps Ω onto Ω. Thus, by induction, $\rho_n \in \mathcal{P}(\Omega)$ for every $n \in \mathbb{N}$.

In practice, ρ_0 is the empirical measure of N particles $x_0^1, \ldots, x_0^N \in \Omega$. Then, every ρ_n is the empirical measure of the particles $x_n^1, \ldots, x_n^N$, and Eq. 22 moves each particle to

$$x_{n+1}^j = x_n^j + \varepsilon\left(\phi_{\rho_n}^*\right)_\Phi(x_n^j).$$

For $\rho = \frac{1}{N}\sum_{j=1}^N \delta_{x^j}$ and $y \in \mathbb{R}^d$, we have that $\phi_\rho^* = \frac{1}{N}\sum_{j=1}^N \xi_{x^j}$. As $\partial_{x_i} k(x, y) = \langle D\Phi(x)(e_i), \Phi(y)\rangle_{H_0}$ (see Section A.3.3.4.3), we obtain

$$\left(\phi_\rho^*\right)_\Phi(y) = \frac{1}{N}\sum_{j=1}^N \left(k(x^j, y)s(x^j) + \nabla_x k(x^j, y)\right).$$

This is the update of Liu & Wang (2016), with the kernel k of the feature map Φ. Figure A.3.19 shows this iteration on the domain of Figure A.3.17.

3.3.6 SVGD in continuous time

In this section, we study SVGD in continuous time. The particles then follow a differential equation, of which the iteration Eq. 22 is a time discretization. We show that this equation has a solution for all times, and that the particles never leave Ω. We do not study the convergence of the iteration to this solution when ε tends to 0.

3.3.6.1 The continuous-time dynamics

We first recall the objects of Section A.3.3.4 that we use. For $\rho \in \mathcal{P}(\Omega)$, the Stein witness is $\phi^*_\rho = \mathbb{E}_{x\sim\rho}[\xi_x]$, and $\left\|\phi^*_\rho\right\|_{\mathcal{H}} \leq B$, where $B = \max_{x\in\Omega} \|\xi_x\|_{\mathcal{H}}$. For $f \in \mathcal{H}$, the vector field f_Φ is $\|f\|_{\mathcal{H}}K$-Lipschitz, bounded by $\|f\|_{\mathcal{H}}C$, and zero on $\partial\Omega$.

Let $\rho_0 \in \mathcal{P}(\Omega)$ be the initial distribution of the particles. For $x \in \mathbb{R}^d$ and $\rho \in \mathcal{P}(\Omega)$, let

$$b(x, \rho) = \left(\phi^*_\rho\right)_\Phi(x),$$

i.e. the vector field of the Stein witness of ρ, evaluated at x. It is the velocity that SVGD gives to a particle located at x, when the particles are distributed according to ρ. Indeed, the iteration Eq. 22 moves this particle to $x + \varepsilon b(x, \rho)$. This is one step of the explicit Euler scheme, with step ε, for the equation in which every particle has the velocity b. We now write this equation. For a random variable X, let $\mathcal{L}(X)$ be its law. Let X_t be the position at time t of a particle whose initial position X_0 is drawn from ρ_0. Then,

$$\begin{cases} \mathrm{d}X_t = b(X_t, \rho_t)\,\mathrm{d}t \\ \mathcal{L}(X_0) = \rho_0 \end{cases} \quad \text{where} \quad \rho_t = \mathcal{L}(X_t). \tag{23}$$

This is a McKean–Vlasov equation, i.e. a differential equation whose coefficients depend on the law of its solution. The function b is its drift, i.e. the velocity of the particles. Here, the drift depends on the distribution ρ_t of all the particles. The equation has no diffusion term, i.e. no noise. It is studied by Liu (2017) and Lu et al. (2019) on the whole space $\mathbb{R}^d$. Here, the particles live in the compact set Ω, and the score is only continuous.

As there is no noise, a particle moves deterministically. For $x \in \mathbb{R}^d$, let $T_t(x)$ be the position at time t of the particle that starts from x. Then, $X_t = T_t(X_0)$, and the distribution of the particles at time t is $\rho_t = (T_t)\#\rho_0$. Thus, Eq. 23 reads

$$\frac{\mathrm{d}}{\mathrm{d}t}T_t(x) = b(T_t(x), \rho_t), \quad \text{where} \quad \rho_t = (T_t)\#\rho_0. \tag{24}$$

The drift $b(\cdot, \rho)$ is only defined for $\rho \in \mathcal{P}(\Omega)$. Thus, we say that a family $(T_t)_{t\geq 0}$ of continuous maps from $\mathbb{R}^d$ to $\mathbb{R}^d$ solves Eq. 24 when $T_0 = I_d$, $\rho_t \in \mathcal{P}(\Omega)$ for every $t \geq 0$, and Eq. 24 holds for all $x \in \mathbb{R}^d$ and $t \geq 0$. At $t = 0$, the derivative is a right derivative.

This equation is not a standard differential equation. In a standard one, the drift is given, and the Cauchy–Lipschitz theorem provides the positions. Here, the drift $b(\cdot, \rho_t)$ is not given. It depends on the distribution ρ_t of the particles, which depends on their positions, which are the unknown. In short, the positions are needed to compute the drift, and the drift is needed to compute the positions.

3.3.6.2 Existence of a solution

Despite this circular dependence, Eq. 24 has a solution, which is defined for all times. This is the main result of this section.

Theorem A.3.6 – *Existence of the continuous-time dynamics*

Suppose that the standing assumptions hold. Let $\rho_0 \in \mathcal{P}(\Omega)$. Then, there exists a family $(T_t)_{t\geq 0}$ of maps from $\mathbb{R}^d$ to $\mathbb{R}^d$ that solves Eq. 24.

Every solution of Eq. 24, and in particular the solutions given by Theorem A.3.6, has three properties, which we prove in Section A.3.3.6.8 (Lemma A.3.9). Let $\rho_t = (T_t)\#\rho_0$. Each property has a simple meaning for the particles. First, every T_t is a homeomorphism of $\mathbb{R}^d$ (Lemma A.3.9 (iii)). Thus, two distinct particles never meet, and the cloud of particles is deformed continuously. Second, $T_t(\Omega) = \Omega$ (Lemma A.3.9 (iii)), i.e. every particle that starts in Ω stays in Ω. The reason is that Φ, and thus every vector field f_Φ, vanishes on $\partial\Omega$. The particles located on $\partial\Omega$ do not move (Lemma A.3.9 (iv)), and, by uniqueness, no other trajectory can reach a point at rest. Note that the definition of a solution only requires that ρ_t be carried by Ω, i.e. that almost every particle stays in Ω. Third, the Stein witness $\phi^*_{\rho_t}$ varies continuously in time (Lemma A.3.9 (i)), and so does the drift $b(\cdot, \rho_t)$. Moreover, every Stein witness is bounded by B, so that $\|b(y, \rho_t)\| \leq BC$ for every $y \in \mathbb{R}^d$, i.e. the speed of the particles is at most BC.

Theorem A.3.6 does not claim that the solution is unique. Once the curve $(\rho_t)_{t\geq 0}$ of a solution is fixed, the drift $b(\cdot, \rho_t)$ is BK-Lipschitz in space and continuous in time, so that each particle has a unique trajectory (Lemma A.3.9). We do not know whether the curve $(\rho_t)_{t\geq 0}$ itself is unique. Our argument would give its uniqueness if the Stein feature were Lipschitz on Ω, which requires more than the continuity of s and $D\Phi$ (see Section A.3.3.6.6).

3.3.6.3 Strategy of the proof

The proof breaks the circular dependence of Eq. 24 into two halves, which are treated one after the other.

The first half goes from the drift to the positions. Suppose that the drift is given at every time. Then, the particles no longer interact, as each of them follows a prescribed velocity field. Its position solves a standard differential equation.

The second half goes from the positions to the drift. Once the positions are known, so is the distribution of the particles at every time. SVGD then prescribes a drift, through the Stein witness of this distribution.

Chaining the two halves, we start from a candidate drift and we obtain a new one, i.e.

$$\text{candidate drift} \longmapsto \text{positions} \longmapsto \text{distribution} \longmapsto \text{drift prescribed by SVGD}.$$

In general, the last drift differs from the candidate. When they coincide, the candidate is consistent with the positions that it produces, and these positions solve Eq. 24. Thus, we look for a candidate that this chain leaves unchanged, i.e. a fixed point.

In the whole proof, a *candidate* is a continuous and bounded curve $m : [0, \infty) \to \mathcal{H}$. At time t, it gives the velocity $m(t)_\Phi(y)$ to a particle located at y. The proof has four steps.

1. For a given candidate m, we compute the positions of the particles, and we study them (Section A.3.3.6.4).
2. We write the chain above as a map $m \mapsto G_{\rho_0}(m)$. We show that a fixed point of G_{ρ_0} gives a solution of Eq. 24 (Section A.3.3.6.5).
3. We show that a fixed point exists when the time is restricted to a bounded interval $[0, a]$ (Section A.3.3.6.6). The argument relies on compactness, which requires a bounded time interval.
4. We reach all times (Section A.3.3.6.7). As the fixed points need not be unique, those of two different intervals need not agree. Thus, we glue fixed points on $[0, 1]$, $[1, 2]$, and so on. Each of them starts from the distribution reached at the end of the previous one.

Finally, Section A.3.3.6.8 shows that every solution of Eq. 24, and not only the one built in this proof, has the properties announced in Section A.3.3.6.2.

3.3.6.4 From the drift to the positions

In this subsection, the drift is given. Let m be a candidate, and let $R \geq 0$ such that $\|m(t)\|_{\mathcal{H}} \leq R$ for every $t \geq 0$. At time t, a particle located at y has the velocity $m(t)_\Phi(y)$. For $x \in \mathbb{R}^d$, let $T_t^m(x)$ be the position at time t of the particle that starts from x. The next lemma states that these positions are well defined, and gives three properties. By (i), the particles never meet and never cross $\partial\Omega$. By (ii), their speed is at most RC. By (iii), two close candidates give close positions.

Lemma A.3.5 – *Positions for a given drift*

Suppose that the standing assumptions hold. Let $R \geq 0$ and let $m : [0, \infty) \to \mathcal{H}$ be continuous, with $\|m(t)\|_{\mathcal{H}} \leq R$ for every $t \geq 0$. Then, for every $x \in \mathbb{R}^d$, the equation

$$\frac{\mathrm{d}}{\mathrm{d}t} T_t^m(x) = m(t)_\Phi(T_t^m(x)), \quad T_0^m(x) = x$$

has a unique solution on $[0, \infty)$. Moreover, the following holds.

(i) For every $t \geq 0$, T_t^m is a homeomorphism of $\mathbb{R}^d$, and $T_t^m(\Omega) = \Omega$.

(ii) For all $x \in \mathbb{R}^d$ and $t, t' \geq 0$, $\|T_t^m(x) - T_{t'}^m(x)\| \leq RC|t - t'|$.

(iii) Let $m' : [0, \infty) \to \mathcal{H}$ be continuous, with $\|m'(t)\|_{\mathcal{H}} \leq R$ for every $t \geq 0$. For all $x \in \mathbb{R}^d$ and $t \geq 0$,

$$\left\|T_t^m(x) - T_t^{m'}(x)\right\| \leq C\, t\, e^{RKt} \sup_{0 \leq r \leq t} \|m(r) - m'(r)\|_{\mathcal{H}}.$$

Proof. **Existence and uniqueness** Let $v(t, y) = m(t)_\Phi(y)$ be the velocity at time t of a particle located at y. By the properties of the vector fields recalled in Section A.3.3.6.1, we have that

$$\forall t, t' \geq 0, \forall y, y' \in \mathbb{R}^d, \quad \|v(t, y) - v(t', y')\| \leq C\|m(t) - m(t')\|_{\mathcal{H}} + RK\|y - y'\|,$$

and $\|v(t, y)\| \leq RC$. The first term comes from $v(t, y) - v(t', y) = (m(t) - m(t'))_\Phi(y)$. Thus, v is continuous, and RK-Lipschitz in space at every time. By the Cauchy–Lipschitz theorem, also known as the Picard–Lindelöf theorem, for all $t_0 \geq 0$ and $y \in \mathbb{R}^d$, the equation $\alpha'(t) = v(t, \alpha(t))$ has a unique solution such

that $\alpha(t_0) = y$. This solution is defined on the whole interval $[0, \infty)$, as the Lipschitz constant of v does not depend on the time. With $t_0 = 0$, this defines T_t^m.

Homeomorphism Let $t \geq 0$. To invert T_t^m, we follow the trajectories backward. For $y \in \mathbb{R}^d$, let $S_t(y) = \alpha(0)$, where α is the solution such that $\alpha(t) = y$. By uniqueness of the solutions, S_t is the inverse of T_t^m. By Grönwall's inequality, two solutions α and β satisfy

$$\forall r, r' \geq 0, \quad \|\alpha(r) - \beta(r)\| \leq e^{RK|r-r'|}\|\alpha(r') - \beta(r')\|.$$

Thus, T_t^m and S_t are continuous, i.e. T_t^m is a homeomorphism.

Invariance of Ω The points of $\partial\Omega$ are at rest. Indeed, let $z \in \partial\Omega$. As $\Phi(z) = 0$ by (iv) of Assumption A.3.2, we have that $v(t, z) = 0$ for every $t \geq 0$. Thus, the constant curve equal to z is a solution. Let α be a solution such that $\alpha(r) \in \Omega$ and $\alpha(r') \notin \Omega$ for some $r, r' \geq 0$. The image by α of the interval between r and r' is connected. It meets Ω and $\mathbb{R}^d \setminus \Omega$, so that it meets $\partial\Omega$ at some point z. By uniqueness of the solutions, α is the constant curve equal to z, which contradicts $\alpha(r) \neq \alpha(r')$. Hence, a solution that meets Ω stays in Ω. In particular, T_t^m and S_t map Ω into Ω. As S_t is the inverse of T_t^m, we obtain $\Omega = T_t^m(S_t(\Omega)) \subseteq T_t^m(\Omega)$. Thus, $T_t^m(\Omega) = \Omega$, which gives (i).

Speed As $\|v(t, y)\| \leq RC$, the derivative of $t \mapsto T_t^m(x)$ is bounded by RC, which gives (ii).

Close candidates give close positions Let $t \geq 0$ and $x \in \mathbb{R}^d$. We measure the distance between the two candidates by

$$\Delta = \sup_{0 \leq r \leq t} \|m(r) - m'(r)\|_{\mathcal{H}},$$

and the gap between the two positions by $\eta(r) = T_r^m(x) - T_r^{m'}(x)$. For $r \in [0, t]$, we have that

$$\begin{aligned}\|\eta'(r)\| &\leq \left\|m(r)_\Phi(T_r^m(x)) - m(r)_\Phi\left(T_r^{m'}(x)\right)\right\| + \left\|\left(m(r) - m'(r)\right)_\Phi\left(T_r^{m'}(x)\right)\right\| \\ &\leq RK\|\eta(r)\| + C\Delta.\end{aligned}$$

The first term compares the same velocity field at two points, and the second one compares two velocity fields at the same point. As $\eta(0) = 0$, Grönwall's inequality gives $\|\eta(t)\| \leq C\,\Delta\,t\,e^{RKt}$, which concludes the proof.

□

3.3.6.5 From the positions to the drift

In this subsection, we close the chain of Section A.3.3.6.3. Let $\rho_0 \in \mathcal{P}(\Omega)$, and let m be a candidate. By (i) of Lemma A.3.5, $(T_t^m)\#\rho_0 \in \mathcal{P}(\Omega)$ for every $t \geq 0$. For this distribution, SVGD prescribes the drift given by the Stein witness $\phi^*_{(T_t^m)\#\rho_0}$. This defines a new curve from $[0, \infty)$ to $\mathcal{H}$, denoted by

$$G_{\rho_0}(m) : t \mapsto \phi^*_{(T_t^m)\#\rho_0}.$$

The candidate m is *consistent* with the positions that it produces when G_{ρ_0} leaves it unchanged, i.e.

$$m = G_{\rho_0}(m), \quad \text{i.e.} \quad \forall t \geq 0, \quad m(t) = \phi^*_{(T_t^m)\#\rho_0}. \tag{25}$$

The curve m appears on both sides of Eq. 25, as the right-hand side depends on m through the positions T_t^m. Thus, Eq. 25 is an equation on m, and not a definition of m.

We now show two facts. First, all the Stein witnesses live in a fixed compact set $\mathcal{W}$. Thus, we can restrict ourselves to the candidates with values in $\mathcal{W}$. Second, a candidate that satisfies Eq. 25 gives a solution of Eq. 24.

Lemma A.3.6 – *Set of the Stein witnesses*

Suppose that the standing assumptions hold. Let $\mathcal{W}$ be the closed convex hull of $\{\xi_x \mid x \in \Omega\}$ in $\mathcal{H}$. Then, $\mathcal{W}$ is compact, and

$$\forall f \in \mathcal{W}, \quad \|f\|_{\mathcal{H}} \leq B, \quad \text{and} \quad \forall \rho \in \mathcal{P}(\Omega), \quad \phi^*_\rho \in \mathcal{W}.$$

Proof. By (i) of Lemma A.3.4, $x \mapsto \xi_x$ is continuous on the compact set Ω. Thus, $\{\xi_x \mid x \in \Omega\}$ is compact. By (ii) of Lemma A.3.4, it is contained in the closed ball of radius B of $\mathcal{H}$. As $\mathcal{H}$ is a Hilbert space, the closed convex hull of a compact set is compact, by Mazur's theorem. Thus, $\mathcal{W}$ is compact. As the closed ball is closed and convex, it contains $\mathcal{W}$.

Let $\rho \in \mathcal{P}(\Omega)$. A Stein witness is a mean of Stein features, and a mean of points of a closed convex set stays in this set. Indeed, by the Hahn–Banach theorem, $\mathcal{W}$ is an intersection of closed half-spaces. Each of them contains ξ_x for ρ-almost every x. It thus contains the mean $\phi^*_\rho = \mathbb{E}_{x\sim\rho}[\xi_x]$, as continuous linear forms commute with the Bochner integral. Hence, $\phi^*_\rho \in \mathcal{W}$.

□

In particular, $G_{\rho_0}(m)$ takes its values in $\mathcal{W}$ for every candidate m. Conversely, a continuous curve $m : [0, \infty) \to \mathcal{W}$ is bounded by B. Thus, it is a candidate, and Lemma A.3.5 applies to it with $R = B$.

Lemma A.3.7 – *A consistent candidate gives a solution*

Suppose that the standing assumptions hold. Let $\rho_0 \in \mathcal{P}(\Omega)$, and let m be a candidate with values in $\mathcal{W}$ such that $m = G_{\rho_0}(m)$. Then, the maps $T_t = T_t^m$ solve Eq. 24.

Proof. Let $T_t = T_t^m$ and $\rho_t = (T_t)\#\rho_0$. As m takes its values in $\mathcal{W}$, it is bounded by B (Lemma A.3.6). By Lemma A.3.5 with $R = B$, every T_t is a homeomorphism of $\mathbb{R}^d$ such that $T_t(\Omega) = \Omega$. Thus, $\rho_t \in \mathcal{P}(\Omega)$. We also have that $T_0 = I_d$. As $m(t) = \phi^*_{\rho_t}$ by Eq. 25, the equation of Lemma A.3.5 reads

$$\frac{\mathrm{d}}{\mathrm{d}t} T_t(x) = m(t)_\Phi(T_t(x)) = \left(\phi^*_{\rho_t}\right)_\Phi(T_t(x)) = b(T_t(x), \rho_t),$$

which concludes the proof.

□

Hence, to prove Theorem A.3.6, it remains to find a candidate m with values in $\mathcal{W}$ such that $m = G_{\rho_0}(m)$.

3.3.6.6 A consistent candidate on a bounded time interval

Let $\rho \in \mathcal{P}(\Omega)$ and $a > 0$. In this subsection, the time is restricted to $[0, a]$. Section A.3.3.6.7 applies its result to several initial distributions. Thus, we write ρ for the initial distribution, instead of ρ_0, and g for a candidate restricted to $[0, a]$. More precisely, let $g : [0, a] \to \mathcal{W}$ be a continuous curve. We extend it to $[0, \infty)$ by the constant $g(a)$. This extension is a candidate bounded by B, and T_t^g denotes the positions that it gives by Lemma A.3.5, with $R = B$. Let

$$Gg(t) = \phi^*_{(T_t^g)\#\rho}, \quad t \in [0, a].$$

This is the map of Section A.3.3.6.5 for the initial distribution ρ, restricted to the times of $[0, a]$. We look for a curve g such that $Gg = g$. We measure the distance between two curves by the norm $\|g\|_\infty = \sup_{t\in[0,a]} \|g(t)\|_{\mathcal{H}}$.

We first control how much the Stein witness changes when the particles are moved. For $r \geq 0$, let

$$\omega(r) = \sup\Big\{ \big\|\xi_x - \xi_y\big\|_{\mathcal{H}} \mid x, y \in \Omega, \|x - y\| \leq r \Big\}.$$

This is the modulus of continuity of the Stein feature on Ω. It measures how much ξ varies between two points of Ω at distance at most r. It is nondecreasing, and $\omega(r)$ tends to 0 as r tends to 0, by (i) of Lemma A.3.4. Let $\psi_1, \psi_2 : \mathbb{R}^d \to \mathbb{R}^d$ be continuous maps that send Ω into Ω. They represent two ways of moving the particles. As ρ is carried by Ω, the change of variables formula gives $\phi^*_{(\psi_1)\#\rho} = \mathbb{E}_{x\sim\rho}\big[\xi_{\psi_1(x)}\big]$, and likewise for ψ_2. As the norm of a mean is at most the mean of the norms, we obtain

$$\big\|\phi^*_{(\psi_1)\#\rho} - \phi^*_{(\psi_2)\#\rho}\big\|_{\mathcal{H}} \leq \mathbb{E}_{x\sim\rho}\Big[\big\|\xi_{\psi_1(x)} - \xi_{\psi_2(x)}\big\|_{\mathcal{H}}\Big] \leq \omega\Big(\sup_{x\in\Omega}\|\psi_1(x) - \psi_2(x)\|\Big) \quad (26)$$

In words, if the two maps send every particle to two points at distance at most r, then the two Stein witnesses differ by at most $\omega(r)$.

This estimate tells how G reacts to a change of the curve. Let $g, g' : [0, a] \to \mathcal{W}$ be continuous curves. By (i) and (iii) of Lemma A.3.5 with $R = B$, the positions that they give at time $t \in [0, a]$ send Ω into Ω, and differ by at most $a\,C\,e^{BKa}\|g - g'\|_\infty$. By Eq. 26, as ω is nondecreasing, we obtain

$$\|Gg - Gg'\|_\infty \leq \omega\big(a\,C\,e^{BKa}\|g - g'\|_\infty\big). \quad (27)$$

A natural idea is to start from any curve g and to apply G repeatedly, as in the proof of the Cauchy–Lipschitz theorem. This works when G is a contraction, i.e. when it brings any two curves closer by a fixed factor. By Eq. 27, this holds when $\omega(r) \leq Lr$ for some $L \geq 0$ and $L\,a\,C\,e^{BKa} < 1$. The first condition means that the Stein feature is Lipschitz on Ω. However, the score s and the derivative $D\Phi$ need not be Lipschitz. We only know that the Stein feature is uniformly continuous, and we cannot show that G is a contraction. We use instead the following theorem,

which requires no contraction. In return, it gives the existence of a fixed point, and not its uniqueness.

Theorem A.3.7 – *Schauder (Schauder, 1930)*

Let Γ be a nonempty, compact and convex subset of a Banach space. Then, every continuous map from Γ to Γ has a fixed point.

To apply it, we need three things.

1. A set Γ of curves g that is convex and compact.
2. The map G sends Γ into Γ.
3. The map G is continuous on Γ, i.e. two close curves g and g' give two close curves Gg and Gg'.

Compactness is the delicate point. The Arzelà–Ascoli theorem gives a criterion. A closed set of continuous curves is compact when two conditions hold. First, the curves take their values in a compact set. Second, they are equicontinuous, i.e. they have a common modulus of continuity. The first condition is given by $\mathcal{W}$. For the second one, we use that the speed of the particles is bounded.

Lemma A.3.8 – *A consistent candidate on a bounded time interval*

Suppose that the standing assumptions hold. Let $\rho \in \mathcal{P}(\Omega)$ and $a > 0$. Then, there exists a continuous curve $g : [0, a] \to \mathcal{W}$ such that

$$\forall t \in [0, a], \quad g(t) = \phi^*_{(T_t^g)\#\rho},$$

where T_t^g is given by Lemma A.3.5 for the extension of g to $[0, \infty)$ by the constant $g(a)$.

Proof. **A convex and compact set of curves** By (ii) of Lemma A.3.5 with $R = B$, the particles move at speed at most BC. Between two times t and t', every particle thus moves by at most $BC|t - t'|$. By Eq. 26, the Stein witness changes by at most $\omega(BC|t - t'|)$. We only keep the curves that vary at this pace. Let Γ be the set of the continuous curves $g : [0, a] \to \mathcal{W}$ such that

$$\forall t, t' \in [0, a], \quad \|g(t) - g(t')\|_{\mathcal{H}} \leq \omega(BC|t - t'|).$$

It is a subset of the Banach space of the continuous curves from $[0, a]$ to $\mathcal{H}$, with the norm $\|\cdot\|_\infty$. The set Γ is nonempty, as it contains the constant curves with values in $\mathcal{W}$. It is convex, as $\mathcal{W}$ is convex and the inequality above is preserved by convex combinations, by the triangle inequality. It is closed, as $\mathcal{W}$ is closed and the inequality above is preserved by uniform limits. Its curves are equicontinuous, as $\omega(r)$ tends to 0 as r tends to 0, and they take their values in the compact set $\mathcal{W}$. Thus, Γ is compact by the Arzelà–Ascoli theorem.

The map G sends Γ into Γ Let $g \in \Gamma$ and $t, t' \in [0, a]$. By (i) of Lemma A.3.5, $(T_t^g)\#\rho \in \mathcal{P}(\Omega)$, so that $Gg(t) \in \mathcal{W}$ by Lemma A.3.6. By Eq. 26, then (ii) of Lemma A.3.5, as ω is nondecreasing, we have that

$$\|Gg(t) - Gg(t')\|_{\mathcal{H}} \leq \omega\left(\sup_{x\in\Omega}\left\|T_t^g(x) - T_{t'}^g(x)\right\|\right) \leq \omega(BC|t-t'|).$$

Thus, $Gg \in \Gamma$.

The map G is continuous Two close curves give close positions, and close positions give close Stein witnesses. This is Eq. 27. As $\omega(r)$ tends to 0 as r tends to 0, G is continuous on Γ.

Conclusion By Theorem A.3.7, G has a fixed point $g \in \Gamma$, which concludes the proof.

□

3.3.6.7 From a bounded time interval to all times

Section A.3.3.6.6 gives a consistent candidate on a bounded time interval, from any initial distribution. The length of this interval is arbitrary, and we take $a = 1$. To reach all times, we proceed interval by interval. We first solve on $[0, 1]$, from the distribution ρ_0. At time 1, the particles have a new distribution ρ_1. We solve again on an interval of length 1, from ρ_1, and so on. Gluing these pieces gives a consistent candidate for all times.

Proof. We prove Theorem A.3.6. We define by induction on $n \in \mathbb{N}$ measures $\rho_n \in \mathcal{P}(\Omega)$ and continuous curves $g_n : [0, 1] \to \mathcal{W}$, starting from the initial distribution ρ_0. Let $n \in \mathbb{N}$ such that $\rho_n \in \mathcal{P}(\Omega)$ is defined. Lemma A.3.8 with $\rho = \rho_n$ and $a = 1$ gives a continuous curve $g_n : [0, 1] \to \mathcal{W}$ such that

$$\forall r \in [0, 1], \quad g_n(r) = \phi^*_{(T_r^{g_n})\#\rho_n}.$$

Let $\rho_{n+1} = \left(T_1^{g_n}\right)\#\rho_n$, which lies in $\mathcal{P}(\Omega)$ by (i) of Lemma A.3.5. Here, ρ_n is the distribution of the particles at time n, and not the n-th iterate of Eq. 22. The curve g_n is the piece of the candidate between the times n and $n + 1$. For $n \in \mathbb{N}$ and $t \in [n, n + 1]$, let $m(t) = g_n(t - n)$. It is well defined, as both pieces are equal to the Stein witness of ρ_{n+1} at time $n + 1$, i.e. $g_n(1) = \phi^*_{\rho_{n+1}} = g_{n+1}(0)$. Thus, m is continuous, with values in $\mathcal{W}$, i.e. it is a candidate with values in $\mathcal{W}$.

Let $t \in [n, n + 1]$ and $x \in \mathbb{R}^d$. Between the times n and $n + 1$, the particles follow the velocity field of g_n. Thus, the curve $r \mapsto T^m_{n+r}(x)$ solves on $[0, 1]$ the equation of Lemma A.3.5 for g_n, from the point $T^m_n(x)$. By uniqueness of the solutions, $T^m_t = T^{g_n}_{t-n} \circ T^m_n$. By induction, $(T^m_n)\#\rho_0 = \rho_n$. Thus,

$$m(t) = g_n(t - n) = \phi^*_{(T^{g_n}_{t-n})\#\rho_n} = \phi^*_{(T^{g_n}_{t-n}\circ T^m_n)\#\rho_0} = \phi^*_{(T^m_t)\#\rho_0}.$$

This is Eq. 25, i.e. $m = G_{\rho_0}(m)$. By Lemma A.3.7, this concludes the proof.

□

3.3.6.8 Properties of the solutions

We now show that every solution of Eq. 24, and not only the solutions given by Theorem A.3.6, has the properties announced in Section A.3.3.6.2. The key point is that, once a solution is fixed, its Stein witnesses form a fixed curve. The trajectories then solve a standard differential equation, and Lemma A.3.5 applies.

Lemma A.3.9 – *Properties of the solutions*

Suppose that the standing assumptions hold. Let $\rho_0 \in \mathcal{P}(\Omega)$, let $(T_t)_{t\geq 0}$ be a solution of Eq. 24, let $\rho_t = (T_t)\#\rho_0$, and let $m(t) = \phi^*_{\rho_t}$. Then, the following holds.

(i) For all $t, t' \geq 0$, $\|m(t) - m(t')\|_{\mathcal{H}} \leq \omega(BC|t - t'|)$, where ω is the modulus of continuity of the Stein feature on Ω, defined in Section A.3.3.6.6. In particular, m is continuous.

(ii) For every $t \geq 0$, $T_t = T_t^m$, where T_t^m is given by Lemma A.3.5 with $R = B$.

(iii) For every $t \geq 0$, T_t is a homeomorphism of $\mathbb{R}^d$, and $T_t(\Omega) = \Omega$.

(iv) For all $z \in \partial\Omega$ and $t \geq 0$, $T_t(z) = z$.

Proof. **Speed** Every Stein witness has norm at most B. As $\|\Phi(y)\|_{H_0} \leq C$, we have that $\|b(y, \rho_t)\| \leq \|m(t)\|_{\mathcal{H}} \|\Phi(y)\|_{H_0} \leq BC$ for all $t \geq 0$ and $y \in \mathbb{R}^d$. By Eq. 24, the particles thus move at speed at most BC, i.e.

$$\forall x \in \mathbb{R}^d, \forall t, t' \geq 0, \quad \|T_t(x) - T_{t'}(x)\| \leq BC|t - t'|.$$

Continuity of the Stein witness Let $t, t' \geq 0$. As $\rho_t, \rho_{t'} \in \mathcal{P}(\Omega)$, the points $T_t(x)$ and $T_{t'}(x)$ lie in Ω for ρ_0-almost every x. By the change of variables formula, and as the norm of a mean is at most the mean of the norms, we have that

$$\|m(t) - m(t')\|_{\mathcal{H}} = \left\|\mathbb{E}_{x\sim\rho_0}\left[\xi_{T_t(x)} - \xi_{T_{t'}(x)}\right]\right\|_{\mathcal{H}} \leq \mathbb{E}_{x\sim\rho_0}\left[\left\|\xi_{T_t(x)} - \xi_{T_{t'}(x)}\right\|_{\mathcal{H}}\right] \leq \omega(BC|t - t'|).$$

The last inequality follows from the speed bound above, the definition of ω, and the fact that ω is nondecreasing. As $\omega(r)$ tends to 0 as r tends to 0, m is continuous, which gives (i).

The trajectories The solution $(T_t)_{t\geq 0}$ is fixed, so that m is a fixed curve in $\mathcal{H}$. By Eq. 24, for every $x \in \mathbb{R}^d$, the curve $t \mapsto T_t(x)$ solves the equation of Lemma A.3.5 for m, from x. By (i), m is continuous, and it is bounded by B. Thus, Lemma A.3.5 applies to m with $R = B$. It gives a unique solution of this equation from each initial position. As $t \mapsto T_t(x)$ and $t \mapsto T_t^m(x)$ are two solutions from x, they are equal, which gives (ii). This only uses the uniqueness of the trajectories for the fixed curve m. It does not use the uniqueness of the solution of Eq. 24, which is not known. By (ii), (iii) follows from (i) of Lemma A.3.5.

The boundary Let $z \in \partial\Omega$. As $\Phi(z) = 0$ by (iv) of Assumption A.3.2, $m(t)_\Phi(z) = 0$ for every $t \geq 0$. Thus, the constant curve equal to z solves the equation of Lemma A.3.5 for m, from z. By uniqueness, $T_t(z) = T_t^m(z) = z$, which concludes the proof.

□

3.3.7 The time derivative of the KL divergence

In Section A.3.3.5, we showed that the derivative of the divergence, in the direction of the vector field of the Stein witness, is $-\mathfrak{K}(\rho \mid \pi)$ (see (iii) of Theorem A.3.5). In continuous time, the particles follow the vector field of the Stein witness of their distribution at every time. Thus, we expect the divergence to decrease at rate $\mathfrak{K}(\rho_t \mid \pi)$ at every time. In this section, we prove it.

3.3.7.1 Decrease of the Kullback–Leibler divergence

Theorem A.3.8 – *Time derivative of the KL divergence*

Suppose that the standing assumptions hold. Let ρ_0 be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$, so that $\rho_0 \in \mathcal{P}(\Omega)$ (see Section A.3.3.3.2). Let $(T_t)_{t\geq 0}$ be a solution of Eq. 24, such as the solutions given by Theorem A.3.6, and let $\rho_t = (T_t)\#\rho_0$. Then, $\mathrm{KL}(\rho_t \parallel \pi) < \infty$ for every $t \geq 0$. Moreover, the function $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$ is differentiable on $[0, \infty)$, with a right derivative at 0, and

$$\forall t \geq 0, \quad \frac{\mathrm{d}}{\mathrm{d}t} \mathrm{KL}(\rho_t \parallel \pi) = -\mathfrak{K}(\rho_t \mid \pi).$$

In particular, $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$ is nonincreasing on $[0, \infty)$.

The Kullback–Leibler divergence thus decreases along the dynamics, at rate $\mathfrak{K}(\rho_t \mid \pi)$. It can only stop decreasing when $\mathfrak{K}(\rho_t \mid \pi) = 0$, i.e. when the Stein witness $\phi^*_{\rho_t}$ vanishes. In Section A.3.3.8, we use this identity to study the convergence of ρ_t to π.

In Liu (2017), this identity is derived from the continuity equation satisfied by the density of ρ_t. Our proof follows the particles instead. It never differentiates the density of ρ_t, and it needs no regularity of $\partial\Omega$.

3.3.7.2 Strategy of the proof

The proof follows that of Theorem A.3.3. There, we differentiated $\varepsilon \mapsto \mathrm{KL}((I_d + \varepsilon v)\#\rho \parallel \pi)$ at $\varepsilon = 0$, for a fixed vector field v. Here, we differentiate $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$ at every time $t \geq 0$. Theorem A.3.3 does not apply directly at time t, as $T_{t+r} \circ T_t^{-1}$ is not of the form $I_d + rv$ for a fixed vector field v. Instead, we write $\mathrm{KL}(\rho_t \parallel \pi)$ as an integral against the fixed measure ρ_0, and we differentiate this integral.

In the rest of this section, ρ_0, $(T_t)_{t\geq 0}$ and $\rho_t = (T_t)\#\rho_0$ are as in Theorem A.3.8. For $t \geq 0$, let $v_t = b(\cdot, \rho_t) = \left(\phi^*_{\rho_t}\right)_\Phi$ be the velocity field of the particles at time t, i.e. the vector field of the Stein witness of ρ_t. Let $J_t(x) = \det DT_t(x)$ be the Jacobian determinant of T_t at x. By Lemma A.3.2 with $\Psi = T_t$, we have that

$$\mathrm{KL}(\rho_t \parallel \pi) = \mathrm{KL}(\rho_0 \parallel \pi) - \int (\log p(T_t(x)) - \log p(x) + \log J_t(x)) \, \mathrm{d}\rho_0(x).$$

Only the integrand depends on t. We differentiate it along the trajectory of x. By the chain rule, the derivative of $\log p(T_t(x))$ is $s(T_t(x))(v_t(T_t(x)))$. By Liouville's formula for the Jacobian determinant of T_t, the derivative of $\log J_t(x)$ is $\nabla \cdot v_t(T_t(x))$. The sum of these two derivatives is the Stein operator of v_t, evaluated at $T_t(x)$. Then, we differentiate under the integral sign, and we change variables with $y = T_t(x)$. This gives

$$\frac{\mathrm{d}}{\mathrm{d}t} \mathrm{KL}(\rho_t \parallel \pi) = -\mathbb{E}_{y\sim\rho_t}[\mathcal{A}_s v_t(y)].$$

The expectation is under ρ_t, and v_t is the vector field of the Stein witness of the same measure ρ_t. Thus, (ii) of Theorem A.3.4 shows that this expectation is $\mathfrak{K}(\rho_t \mid \pi)$.

Compared with Theorem A.3.3, the difficulty is the map T_t. The Jacobian determinant of $I_d + \varepsilon v$ is explicit, whereas that of T_t is not. Section A.3.3.7.3 shows that T_t is differentiable, with $J_t > 0$, and it computes the derivative of $\log J_t$. Section A.3.3.7.4 carries out the computation above. It checks the assumptions of Lemma A.3.2, and it bounds the derivative of the integrand by a constant, which justifies the differentiation under the integral sign.

3.3.7.3 The Jacobian of T_t

We first gather the facts on v_t that we use. Let $m(t) = \phi^*_{\rho_t}$, so that $v_t = m(t)_\Phi$. By (i) of Lemma A.3.9, m is continuous. It is bounded by B, as every Stein witness is (see Section A.3.3.4.3). Recall that $P_f : u \mapsto \left(\langle f_i, u\rangle_{H_0}\right)_{1\le i\le d}$ satisfies $\|P_f\| \le \|f\|_{\mathcal{H}}$, and that $f_\Phi = P_f \circ \Phi$ (see Section A.3.3.4.1). By (i) to (iii) of Assumption A.3.2, v_t is continuously differentiable, with $Dv_t(y) = P_{m(t)} \circ D\Phi(y)$, and

$$\forall t \ge 0, \forall y \in \mathbb{R}^d, \quad \|v_t(y)\| \le BC \quad \text{and} \quad \|Dv_t(y)\| \le BK.$$

As the trace of a linear map L of $\mathbb{R}^d$ satisfies $|\operatorname{tr} L| \le d\|L\|$, we obtain $|\nabla \cdot v_t(y)| \le dBK$ for all $t \ge 0$ and $y \in \mathbb{R}^d$. Finally, by Eq. 24, for every $x \in \mathbb{R}^d$, the curve $t \mapsto T_t(x)$ solves

$$\alpha'(t) = v_t(\alpha(t)), \quad \alpha(0) = x. \tag{28}$$

Once the solution $(T_t)_{t\ge 0}$ is fixed, v_t is a fixed velocity field, so that Eq. 28 is a standard differential equation.

The next lemma states that T_t is differentiable with respect to the initial position. Its Jacobian determinant measures how much the particles spread out. It is positive, and its logarithm grows at most linearly in time.

Lemma A.3.10 – *Jacobian of T_t*

Suppose that the standing assumptions hold, and let $(T_t)_{t\ge 0}$ and v_t be as in Section A.3.3.7.2. Then, for every $t \ge 0$, T_t is continuously differentiable on $\mathbb{R}^d$. Let $J_t(x) = \det DT_t(x)$. Then, for all $x \in \mathbb{R}^d$ and $t \ge 0$, the following holds.

(i) $J_t(x) > 0$.
(ii) $\partial_t \log J_t(x) = \nabla \cdot v_t(T_t(x))$, with a right derivative at $t = 0$.
(iii) $|\log J_t(x)| \le dBK\, t$.

Proof. **Differentiability** For all $t, t' \ge 0$ and $y, y' \in \mathbb{R}^d$, we have that

$$\|Dv_t(y) - Dv_{t'}(y')\| \le \|m(t) - m(t')\|_{\mathcal{H}} \|D\Phi(y)\| + B\|D\Phi(y) - D\Phi(y')\|.$$

As m and $D\Phi$ are continuous, $Dv_t(y)$ is continuous in (t, y). We extend m to negative times by $m(0)$, so that $(t, y) \mapsto v_t(y)$ is defined on the open set $\mathbb{R} \times \mathbb{R}^d$, continuous, with $Dv_t(y)$ continuous in (t, y). As v_t is BK-Lipschitz, Eq. 28 has a unique solution from each x, which is $t \mapsto T_t(x)$. Thus, the differentiable dependence on initial conditions (Hartman, 2002, Chapter V) shows that T_t is continuously differentiable. Let $x \in \mathbb{R}^d$ and $Y_t = DT_t(x)$. Then, Y_t solves the variational equation

$$\partial_t Y_t = Dv_t(T_t(x))Y_t, \quad Y_0 = I_d.$$

In words, a small displacement of the initial position is transported by the derivative of the velocity field along the trajectory.

Liouville's formula Let $A_t = Dv_t(T_t(x))$, and let $\operatorname{adj}(Y)$ be the adjugate matrix of a matrix Y, so that $Y \operatorname{adj}(Y) = \det(Y) I_d$. By Jacobi's formula, we have that

$$\partial_t \det Y_t = \operatorname{tr}(\operatorname{adj}(Y_t)\partial_t Y_t) = \operatorname{tr}(\operatorname{adj}(Y_t)A_t Y_t) = \operatorname{tr}(A_t Y_t \operatorname{adj}(Y_t)) = \operatorname{tr}(A_t) \det Y_t.$$

As $\operatorname{tr}(A_t) = \nabla \cdot v_t(T_t(x))$, the function $t \mapsto J_t(x)$ solves a scalar linear differential equation, with $J_0(x) = \det I_d = 1$. Its solution is given by Liouville's formula, i.e.

$$J_t(x) = \exp\left(\int_0^t \nabla \cdot v_r(T_r(x))\,\mathrm{d}r\right) > 0,$$

which gives (i). The integrand is continuous in r, as $Dv_r(y)$ is continuous in (r, y) and $r \mapsto T_r(x)$ is continuous. Thus, $\log J_t(x)$ is differentiable in t, with derivative $\nabla \cdot v_t(T_t(x))$, which gives (ii).

Bound As $|\nabla \cdot v_r(y)| \leq dBK$ for all $r \geq 0$ and $y \in \mathbb{R}^d$, Liouville's formula gives $|\log J_t(x)| \leq dBK\,t$, which concludes the proof.

□

3.3.7.4 Differentiation of the KL divergence

We can now carry out the computation of Section A.3.3.7.2.

Proof. We prove Theorem A.3.8. Let v_t be as in Section A.3.3.7.2, and let J_t be as in Lemma A.3.10.

Transport Let $t \geq 0$. By (iii) of Lemma A.3.9, T_t is a homeomorphism of $\mathbb{R}^d$ such that $T_t(\Omega) = \Omega$. By Lemma A.3.10, it is differentiable, with $J_t > 0$, and $\log J_t$ is bounded. Thus, $\log J_t$ is ρ_0-integrable. By Lemma A.3.2, $\mathrm{KL}(\rho_t \parallel \pi) < \infty$ and

$$\mathrm{KL}(\rho_t \parallel \pi) = \mathrm{KL}(\rho_0 \parallel \pi) - \int \ell(t, x)\,\mathrm{d}\rho_0(x),$$

where $\ell(t, x) = \log p(T_t(x)) - \log p(x) + \log J_t(x)$.

Differentiation Let $x \in \Omega$, which holds for ρ_0-almost every x. We first differentiate $t \mapsto \log p(T_t(x))$. If $x \in \Omega^\circ$, then $T_t(x) \in \Omega^\circ$ for every $t \geq 0$. Indeed, a homeomorphism of $\mathbb{R}^d$ that maps Ω onto Ω maps Ω° onto Ω°. As $\log p$ is differentiable at $T_t(x)$, with derivative $s(T_t(x))$, the chain rule gives

$$\partial_t \log p(T_t(x)) = s(T_t(x))(v_t(T_t(x))).$$

Otherwise, $x \in \partial\Omega$. Then, x is at rest, i.e. $T_t(x) = x$ by (iv) of Lemma A.3.9, and $v_t(x) = 0$ as $\Phi(x) = 0$. Thus, both sides vanish. With (ii) of Lemma A.3.10, we obtain

$$\partial_t \ell(t, x) = s(T_t(x))(v_t(T_t(x))) + \nabla \cdot v_t(T_t(x)) = \mathcal{A}_s v_t(T_t(x)).$$

As in Theorem A.3.3, we bound this derivative uniformly. Recall that $M_s = \max_{y \in \Omega} \|s(y)\|$ (see Section A.3.3.2). By Section A.3.3.7.3, $\|v_t(y)\| \leq BC$ and $|\nabla \cdot v_t(y)| \leq dBK$ for all $t \geq 0$ and $y \in \mathbb{R}^d$. As $T_t(x) \in \Omega$, $\|s(T_t(x))\| \leq M_s$. Thus,

$$|\partial_t \ell(t,x)| \leq M_s BC + dBK.$$

This constant is ρ_0-integrable. Every $\ell(t,\cdot)$ is also continuous and bounded on Ω, which carries ρ_0. Indeed, $\log p$ is continuous and bounded on Ω, T_t is continuous with $T_t(\Omega) = \Omega$, and $\log J_t$ is continuous, as DT_t is continuous, and bounded by (iii) of Lemma A.3.10. By the Leibniz integral rule (Folland, 1999, Theorem 2.27), $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$ is differentiable at every $t \geq 0$, with

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{KL}(\rho_t \parallel \pi) &= -\int \partial_t \ell(t,x)\,\mathrm{d}\rho_0(x) \\ &= -\mathbb{E}_{x\sim\rho_0}[\mathcal{A}_s v_t(T_t(x))] \\ &= -\mathbb{E}_{y\sim\rho_t}[\mathcal{A}_s v_t(y)].\end{aligned}$$

The last equality is the change of variables $y = T_t(x)$.

The Stein witness As $v_t = \left(\phi^*_{\rho_t}\right)_\Phi$ and $\rho_t \in \mathcal{P}(\Omega)$, (ii) of Theorem A.3.4 gives

$$\mathbb{E}_{y\sim\rho_t}[\mathcal{A}_s v_t(y)] = \mathfrak{K}(\rho_t \mid \pi).$$

Thus, $\frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{KL}(\rho_t \parallel \pi) = -\mathfrak{K}(\rho_t \mid \pi)$.

Monotonicity As $\mathfrak{K}(\rho_t \mid \pi) \geq 0$, the derivative is nonpositive on $[0,\infty)$. By the mean value theorem, $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$ is nonincreasing on $[0,\infty)$, which concludes the proof.

□

3.3.8 Convergence

In this section, we show that the distribution ρ_t of the particles converges to π as t tends to infinity, when the kernel is rich enough. We first define the notion of convergence, and a condition on the kernel, called Stein-universality. Under this condition, the KSD vanishes only at π. Then, we state the main result of this section, and we prove it in three steps.

3.3.8.1 Weak convergence and Stein-universal kernels

Let $C^0(\Omega)$ be the set of continuous functions from Ω to $\mathbb{R}$. Let $\rho \in \mathcal{P}(\Omega)$ and $g \in C^0(\Omega)$. As Ω is compact, g is bounded. As ρ is carried by Ω, the expectation $\mathbb{E}_{x\sim\rho}[g(x)]$ is thus well defined (see Section A.3.3.2). We compare measures through these expectations. A sequence $(\rho_n)_{n\in\mathbb{N}}$ of $\mathcal{P}(\Omega)$ converges weakly to $\rho \in \mathcal{P}(\Omega)$ if

$$\forall g \in C^0(\Omega), \quad \mathbb{E}_{x\sim\rho_n}[g(x)] \xrightarrow[n\to\infty]{} \mathbb{E}_{x\sim\rho}[g(x)].$$

We then write $\rho_n \xrightarrow[n\to\infty]{} \rho$, and likewise for a family $(\rho_t)_{t\geq 0}$ as t tends to infinity.

This is the usual weak convergence of probability measures on the metric space Ω (Billingsley, 1999). Such a measure is only defined on the Borel subsets of Ω, which are the sets $A \in \mathcal{B}(\mathbb{R}^d)$ such that $A \subseteq \Omega$, as Ω is closed. A measure of $\mathcal{P}(\Omega)$ is defined on all of $\mathcal{B}(\mathbb{R}^d)$. We go from one to the other by restriction. Let $\rho \in \mathcal{P}(\Omega)$. Its restriction $\rho|_\Omega$ to Ω is defined by

$$\forall A \in \mathcal{B}(\mathbb{R}^d), \quad A \subseteq \Omega \Longrightarrow \rho|_\Omega(A) = \rho(A).$$

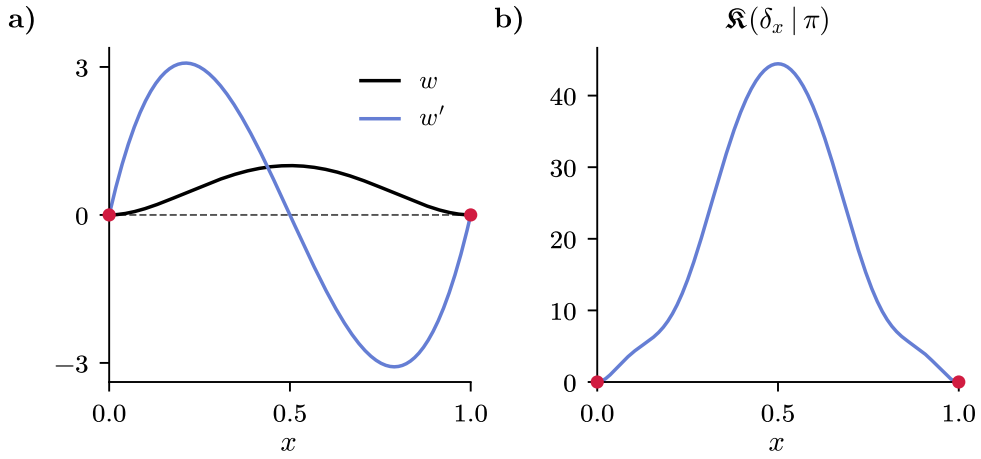


Figure A.3.20. *A kernel whose KSD vanishes at Dirac masses that are not π. Here, $\Omega = [0,1]$, the target π and the Gaussian kernel k_0 are those of Figure A.3.18, and the kernel is $k(x,y) = w(x)w(y)k_0(x,y)$ with $w(x) = 16x^2(1-x)^2$. Its feature map is $\Phi = w\Phi_0$, where Φ_0 is a feature map of k_0. **a)** The weight w, in black, and its derivative w', in blue, vanish on $\partial\Omega = \{0,1\}$ (red dots). Thus, so do $\Phi = w\Phi_0$ and $D\Phi = w'\Phi_0 + wD\Phi_0$. **b)** The KSD of the Dirac mass at x. By Theorem A.3.4, $\mathfrak{K}(\delta_x \mid \pi) = \|\xi_x\|^2_{\mathcal{H}} = (s(x)w(x) + w'(x))^2 + w(x)^2/0.15^2$. It is positive on $(0,1)$, but it vanishes at 0 and 1 (red dots), whereas $\delta_0 \neq \pi$ and $\delta_1 \neq \pi$.*

As $\rho(\mathbb{R}^d \setminus \Omega) = 0$, we have that $\rho|_\Omega(\Omega) = \rho(\mathbb{R}^d) = 1$, i.e. $\rho|_\Omega$ is a probability measure on Ω. No information is lost, as

$$\forall A \in \mathcal{B}(\mathbb{R}^d), \quad \rho(A) = \rho|_\Omega(A \cap \Omega).$$

Conversely, every probability measure ρ' on Ω is the restriction of the measure $A \mapsto \rho'(A \cap \Omega)$ of $\mathcal{P}(\Omega)$. Moreover, $\mathbb{E}_{x\sim\rho}[g(x)] = \int_\Omega g \, \mathrm{d}\rho|_\Omega$ for every $g \in C^0(\Omega)$. The functions of $C^0(\Omega)$ are the bounded continuous functions on Ω. Thus, $\rho_n \rightharpoonup \rho$ if and only if $\rho_n|_\Omega$ converges weakly to $\rho|_\Omega$, as probability measures on the metric space Ω. This allows us to apply two classical results on probability measures on a metric space, with the metric space Ω, in Proposition A.3.1 and in Section A.3.3.8.6.

We now turn to the kernel. Let $\rho \in \mathcal{P}(\Omega)$. By Theorem A.3.4, we have that $\mathfrak{K}(\rho \mid \pi) = \left\|\phi^*_\rho\right\|^2_{\mathcal{H}}$ and $\mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = \langle f, \phi^*_\rho \rangle_{\mathcal{H}}$ for every $f \in \mathcal{H}$. Thus, we have that

$$\mathfrak{K}(\rho \mid \pi) = 0 \iff \phi^*_\rho = 0 \iff \forall f \in \mathcal{H}, \quad \mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = 0, \tag{29}$$

where the converse implication of the second equivalence follows by taking $f = \phi^*_\rho$. In words, the KSD of ρ vanishes if and only if ρ satisfies Stein's identity for every vector field f_Φ, as π does (see Corollary A.3.2). To deduce that $\rho = \pi$, the functions $\mathcal{A}_s f_\Phi$ must be rich enough to determine a measure. This is not always the case. For instance, let $y \in \partial\Omega$ such that $D\Phi(y) = 0$. As $\Phi(y) = 0$ by (iv) of Assumption A.3.2, we have that $\xi_y = 0$. Then, $\phi^*_{\delta_y} = \xi_y = 0$, i.e. $\mathfrak{K}(\delta_y \mid \pi) = 0$, whereas $\delta_y \neq \pi$, as π has a density. Figure A.3.20 illustrates this situation in dimension 1.

Stein-universality excludes this situation (see Proposition A.3.1). Let $f \in \mathcal{H}$. As $\mathcal{A}_s f_\Phi$ is continuous on Ω (see Section A.3.3.4.2), its restriction to Ω lies in $C^0(\Omega)$.

Stein-universality requires that these functions, up to an additive constant, approximate every function of $C^0(\Omega)$, uniformly on Ω.

Definition A.3.12 – *Stein-universal kernel*

Suppose that the standing assumptions hold. The kernel k is *Stein-universal* if the set of functions

$$\{\mathcal{A}_s f_\Phi + c \mid f \in \mathcal{H}, c \in \mathbb{R}\},$$

restricted to Ω, is dense in $C^0(\Omega)$ for the uniform convergence on Ω, i.e.

$$\forall g \in C^0(\Omega), \forall \eta > 0, \exists f \in \mathcal{H}, \exists c \in \mathbb{R}, \quad \sup_{x \in \Omega} |g(x) - \mathcal{A}_s f_\Phi(x) - c| \leq \eta.$$

This is the analog, for the Stein operator, of the universality of a kernel, which requires that its RKHS be dense in $C^0(\Omega)$ (Steinwart, 2001). This condition depends on Φ through the vector fields f_Φ, on π through its score s, and on Ω.

The constants c are needed. Indeed, let $f \in \mathcal{H}$. The vector field f_Φ is continuously differentiable and Lipschitz, and it vanishes on $\partial\Omega$ (see Section A.3.3.4.1). By Corollary A.3.2, we have that $\mathbb{E}_{x\sim\pi}[\mathcal{A}_s f_\Phi(x)] = 0$. Thus, the functions $\mathcal{A}_s f_\Phi$ alone only approximate functions with zero mean under π. In fact, the kernel is Stein-universal if and only if

$$\forall g \in C^0(\Omega), \forall \eta > 0, \exists f \in \mathcal{H}, \quad \sup_{x \in \Omega} \left|g(x) - \mathbb{E}_{y\sim\pi}[g(y)] - \mathcal{A}_s f_\Phi(x)\right| \leq \eta. \quad (30)$$

In words, every function of $C^0(\Omega)$ with zero mean under π is a uniform limit on Ω of functions $\mathcal{A}_s f_\Phi$. Indeed, if Eq. 30 holds, then the inequality of Definition A.3.12 holds with $c = \mathbb{E}_{y\sim\pi}[g(y)]$. Conversely, suppose that the kernel is Stein-universal. Let $g \in C^0(\Omega)$ and $\eta > 0$. By Definition A.3.12 applied with $\eta/2$, there exist $f \in \mathcal{H}$ and $c \in \mathbb{R}$ such that

$$\forall x \in \Omega, \quad |g(x) - \mathcal{A}_s f_\Phi(x) - c| \leq \frac{\eta}{2}.$$

As π is a probability measure carried by Ω and $\mathbb{E}_{y\sim\pi}[\mathcal{A}_s f_\Phi(y)] = 0$, integrating this inequality against π gives

$$\left|\mathbb{E}_{y\sim\pi}[g(y)] - c\right| = \left|\mathbb{E}_{y\sim\pi}[g(y) - \mathcal{A}_s f_\Phi(y) - c]\right| \leq \frac{\eta}{2}.$$

By the triangle inequality, we obtain that, for every $x \in \Omega$,

$$\left|g(x) - \mathbb{E}_{y\sim\pi}[g(y)] - \mathcal{A}_s f_\Phi(x)\right| \leq |g(x) - \mathcal{A}_s f_\Phi(x) - c| + \left|c - \mathbb{E}_{y\sim\pi}[g(y)]\right| \leq \eta,$$

which is Eq. 30.

Proposition A.3.1 – *The KSD vanishes only at π*

Suppose that the standing assumptions hold, and that the kernel is Stein-universal. Then,

$$\forall \rho \in \mathcal{P}(\Omega), \quad \mathfrak{K}(\rho \mid \pi) = 0 \iff \rho = \pi.$$

Proof. If $\rho = \pi$, then $\mathfrak{K}(\rho \mid \pi) = 0$ (see Section A.3.3.4.2). Conversely, let $\rho \in \mathcal{P}(\Omega)$ such that $\mathfrak{K}(\rho \mid \pi) = 0$. By Eq. 29, $\mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)] = 0$ for every $f \in \mathcal{H}$. Let $g \in C^0(\Omega)$ and $\eta > 0$, and let $f \in \mathcal{H}$ be given by Eq. 30. As ρ is a probability measure carried by Ω, we have that

$$\begin{aligned}\left|\mathbb{E}_{x\sim\rho}[g(x)] - \mathbb{E}_{y\sim\pi}[g(y)]\right| &= \left|\mathbb{E}_{x\sim\rho}\left[g(x) - \mathbb{E}_{y\sim\pi}[g(y)] - \mathcal{A}_s f_\Phi(x)\right]\right| \\ &\leq \sup_{x\in\Omega}\left|g(x) - \mathbb{E}_{y\sim\pi}[g(y)] - \mathcal{A}_s f_\Phi(x)\right| \\ &\leq \eta.\end{aligned}$$

As η is arbitrary, $\mathbb{E}_{x\sim\rho}[g(x)] = \mathbb{E}_{y\sim\pi}[g(y)]$ for every $g \in C^0(\Omega)$. Thus, $\rho|_\Omega$ and $\pi|_\Omega$ are two probability measures on the metric space Ω that give the same integral to every bounded continuous function. Such measures are equal (Billingsley, 1999, Theorem 1.2). As ρ and π are determined by their restrictions to Ω, we have that $\rho = \pi$, which concludes the proof.

□

3.3.8.2 Convergence to the target distribution

We can now state the main result of this section. Like Theorem A.3.8, it holds for every solution of Eq. 24.

Theorem A.3.9 – *Convergence to the target distribution*

Suppose that the standing assumptions hold, and that the kernel is Stein-universal. Let ρ_0 be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$. Let $(T_t)_{t\geq 0}$ be a solution of Eq. 24, such as the solutions given by Theorem A.3.6, and let $\rho_t = (T_t)\#\rho_0$. Then,

$$\rho_t \xrightarrow[t\to\infty]{} \pi.$$

For the particles, Theorem A.3.9 reads as follows. Let $g \in C^0(\Omega)$. By Lemma A.3.9 (iii), $T_t(x) \in \Omega$ for every $x \in \Omega$, so that $g(T_t(x))$ is defined for ρ_0-almost every x. The average of g over the positions of the particles at time t is

$$\mathbb{E}_{x\sim\rho_0}[g(T_t(x))] = \mathbb{E}_{y\sim\rho_t}[g(y)] \xrightarrow[t\to\infty]{} \mathbb{E}_{y\sim\pi}[g(y)].$$

Thus, in the long run, the distribution of the particles approaches π.

Theorem A.3.9 has four limitations. First, it gives no rate of convergence. Second, it concerns the continuous-time dynamics Eq. 24. The convergence of the iteration Eq. 22 is not studied (see Section A.3.3.6). Third, the assumption $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$ implies that $\rho_0 \ll \pi$, i.e. that ρ_0 has a density with respect to π. It thus excludes the empirical measure of finitely many particles, used in practice (see Section A.3.3.5). Finally, it assumes that the kernel is Stein-universal. We do not address here the existence or the construction of Stein-universal kernels.

3.3.8.3 Strategy of the proof

By Theorem A.3.8, the Kullback–Leibler divergence decreases along the dynamics, at rate $\mathfrak{K}(\rho_t \mid \pi)$. As it is nonnegative, it has a limit as t tends to infinity. Informally, its derivative $-\mathfrak{K}(\rho_t \mid \pi)$ should then tend to 0, and ρ_t should approach a

measure whose KSD vanishes, which is π by Proposition A.3.1. The proof makes this argument rigorous, in three steps.

1. We show that $\mathfrak{K}(\rho_t \mid \pi)$ tends to 0 (Section A.3.3.8.4). This does not follow from the convergence of the divergence alone, as a function can converge while its derivative keeps oscillating. We show that $t \mapsto \mathfrak{K}(\rho_t \mid \pi)$ is uniformly continuous, which excludes these oscillations.
2. We show that the limit points of ρ_t have a zero KSD (Section A.3.3.8.5). A limit point is a measure $\rho_\infty \in \mathcal{P}(\Omega)$ such that $\rho_{t_n} \rightharpoonup \rho_\infty$ for some sequence $t_n \to \infty$. The weak convergence controls the expectation of $\mathcal{A}_s f_\Phi$ for each fixed f. As the KSD is a supremum of the squares of such expectations, the KSD of the limit is at most the limit of the KSD, which is 0.
3. We show that ρ_t converges weakly to π (Section A.3.3.8.6). As Ω is compact, every sequence $\left(\rho_{t_n}\right)_{n\in\mathbb{N}}$ with $t_n \to \infty$ has a weakly convergent subsequence. By the second step and Proposition A.3.1, its limit is π. Thus, ρ_t converges to π.

Only the last step uses the Stein-universality of the kernel.

3.3.8.4 The KSD tends to zero

Let ρ_0, $(T_t)_{t\geq 0}$ and ρ_t be as in Corollary A.3.3 below. Unlike Theorem A.3.9, this corollary does not assume that the kernel is Stein-universal. By Theorem A.3.8, the derivative of $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$ is $-\mathfrak{K}(\rho_t \mid \pi)$. Integrating it, which is justified once we know that the KSD is continuous in time (Lemma A.3.12), gives

$$\forall a \geq 0, \quad \int_0^a \mathfrak{K}(\rho_t \mid \pi)\,\mathrm{d}t = \mathrm{KL}(\rho_0 \parallel \pi) - \mathrm{KL}(\rho_a \parallel \pi) \leq \mathrm{KL}(\rho_0 \parallel \pi).$$

Thus, the nonnegative function $t \mapsto \mathfrak{K}(\rho_t \mid \pi)$ has a finite integral on $[0, \infty)$. This is not enough to make it tend to 0. For instance, the function

$$\theta(t) = \sum_{n\geq 1} \max(0, 1 - n^2|t - n|)$$

is a sum of spikes centered at the integers. The spike centered at n has height 1 and area $1/n^2$. Thus, θ is nonnegative, and its integral on $[0, \infty)$ is $\sum_{n\geq 1} 1/n^2 < \infty$, whereas $\theta(n) = 1$ for every $n \geq 1$. Let $\ell \in \mathbb{R}$ and $F(t) = \ell + \int_t^\infty \theta(\tau)\,\mathrm{d}\tau$. Then, F decreases and converges to ℓ, whereas its derivative $F' = -\theta$ does not tend to 0 (see Figure A.3.21). The spikes of θ get thinner and thinner, so that θ is continuous but not uniformly continuous. Barbalat's lemma (Khalil, 2002, Lemma 8.2) states that uniform continuity excludes this behavior. We give its short proof.

Lemma A.3.11 – *Barbalat*

Let $u : [0, \infty) \to [0, \infty)$ be uniformly continuous, and let $M \geq 0$ such that $\int_0^a u(t)\,\mathrm{d}t \leq M$ for every $a \geq 0$. Then,

$$u(t) \xrightarrow[t\to\infty]{} 0.$$

Proof. Suppose that $u(t)$ does not tend to 0. Then, there exists $\eta > 0$ such that $u(t) \geq \eta$ for arbitrarily large t. Thus, there exists a sequence $(t_n)_{n\in\mathbb{N}}$ such that

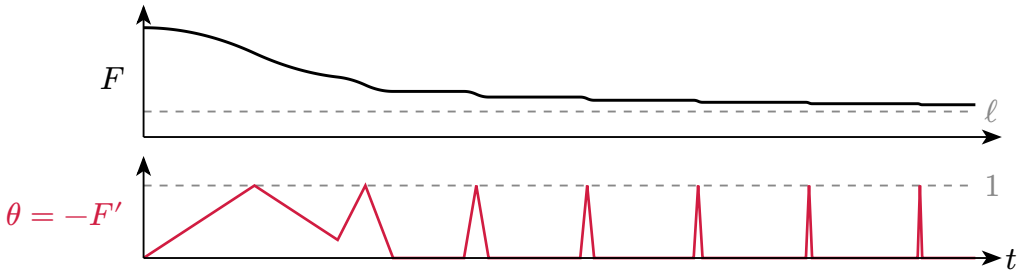


Figure A.3.21. *A function that converges while its derivative does not tend to* 0. ***Top.*** *The function* $F(t) = \ell + \int_t^\infty \theta(\tau)\,\mathrm{d}\tau$ *decreases to its limit* ℓ *(dashed).* ***Bottom.*** *Its derivative* $-F' = \theta$*, with* $\theta(t) = \sum_{n\geq 1} \max(0, 1 - n^2|t-n|)$*, keeps reaching the height* 1 *(dashed). Its spikes have area* $1/n^2$ *and get thinner and thinner, so that* θ *is not uniformly continuous. Here,* F *plays the role of* $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$*, and* θ *the role of* $t \mapsto \mathfrak{K}(\rho_t \mid \pi)$*. This illustrates why Lemma A.3.11 requires uniform continuity.*

$$\forall n \in \mathbb{N}, \quad u(t_n) \geq \eta \quad \text{and} \quad t_{n+1} \geq t_n + 1.$$

As u is uniformly continuous, there exists $\delta \in (0,1)$ such that

$$\forall t, t' \geq 0, \quad |t - t'| \leq \delta \Longrightarrow |u(t) - u(t')| \leq \frac{\eta}{2}.$$

Thus, $u \geq \eta/2$ on every interval $[t_n, t_n + \delta]$. As $\delta < 1$, these intervals are disjoint. As $u \geq 0$, we have that

$$\forall N \in \mathbb{N}, \quad \int_0^{t_N+\delta} u(t)\,\mathrm{d}t \geq \sum_{n=0}^{N} \int_{t_n}^{t_n+\delta} u(t)\,\mathrm{d}t \geq (N+1)\frac{\delta\eta}{2}.$$

The right-hand side tends to infinity with N, which contradicts the bound M. Thus, $u(t)$ tends to 0.

□

It remains to show that $t \mapsto \mathfrak{K}(\rho_t \mid \pi)$ is uniformly continuous. This follows from the continuity in time of the Stein witness (Lemma A.3.9 (i)), and from the bound B on the Stein witnesses. Recall that ω is the modulus of continuity of the Stein feature on Ω (see Section A.3.3.6.6).

Lemma A.3.12 – *Uniform continuity of the KSD*

Suppose that the standing assumptions hold. Let $\rho_0 \in \mathcal{P}(\Omega)$, let $(T_t)_{t\geq 0}$ be a solution of Eq. 24, and let $\rho_t = (T_t)\#\rho_0$. Then,

$$\forall t, t' \geq 0, \quad |\mathfrak{K}(\rho_t \mid \pi) - \mathfrak{K}(\rho_{t'} \mid \pi)| \leq 2B\,\omega(BC|t - t'|).$$

In particular, $t \mapsto \mathfrak{K}(\rho_t \mid \pi)$ is uniformly continuous on $[0, \infty)$.

Proof. Let $t, t' \geq 0$. As $\rho_t, \rho_{t'} \in \mathcal{P}(\Omega)$, (ii) of Theorem A.3.4 gives $\mathfrak{K}(\rho_t \mid \pi) = \left\|\phi^*_{\rho_t}\right\|^2_{\mathcal{H}}$, and likewise at t'. Every Stein witness has norm at most B (see Section A.3.3.4.3). By the reverse triangle inequality and Lemma A.3.9 (i), we have that

$$|\mathfrak{K}(\rho_t \mid \pi) - \mathfrak{K}(\rho_{t'} \mid \pi)| = \left(\left\|\phi^*_{\rho_t}\right\|_{\mathcal{H}} + \left\|\phi^*_{\rho_{t'}}\right\|_{\mathcal{H}}\right)\left|\left\|\phi^*_{\rho_t}\right\|_{\mathcal{H}} - \left\|\phi^*_{\rho_{t'}}\right\|_{\mathcal{H}}\right|$$

$$\leq 2B\left\|\phi^*_{\rho_t} - \phi^*_{\rho_{t'}}\right\|_{\mathcal{H}}$$
$$\leq 2B\,\omega(BC|t-t'|).$$

As $\omega(r)$ tends to 0 as r tends to 0, $t \mapsto \mathfrak{K}(\rho_t \mid \pi)$ is uniformly continuous, which concludes the proof.

□

We can now conclude the first step.

Corollary A.3.3 – *Convergence of the KSD*

Suppose that the standing assumptions hold. Let ρ_0 be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$. Let $(T_t)_{t\geq 0}$ be a solution of Eq. 24, and let $\rho_t = (T_t)\#\rho_0$. Then,

$$\mathfrak{K}(\rho_t \mid \pi) \xrightarrow[t\to\infty]{} 0.$$

Proof. As $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$, we have that $\rho_0 \in \mathcal{P}(\Omega)$ (see Section A.3.3.3.2). For $t \geq 0$, let $F(t) = \mathrm{KL}(\rho_t \parallel \pi)$ and $u(t) = \mathfrak{K}(\rho_t \mid \pi)$. By Theorem A.3.8, F is real-valued and differentiable on $[0,\infty)$, with $F' = -u$. By Lemma A.3.12, u is continuous. Thus, the fundamental theorem of calculus gives

$$\forall a \geq 0, \quad \int_0^a u(t)\,\mathrm{d}t = F(0) - F(a) \leq F(0),$$

as $F(a) \geq 0$. As $u \geq 0$ and u is uniformly continuous, Lemma A.3.11 applies with $M = F(0)$, which concludes the proof.

□

As $\mathfrak{K}(\rho_t \mid \pi) = \left\|\phi^*_{\rho_t}\right\|^2_{\mathcal{H}}$, the Stein witness $\phi^*_{\rho_t}$ tends to 0 in $\mathcal{H}$. Thus, the velocity of the particles tends to 0, uniformly in space, as $\|b(y,\rho_t)\| \leq C\left\|\phi^*_{\rho_t}\right\|_{\mathcal{H}}$ for every $y \in \mathbb{R}^d$. The particles slow down. This does not yet show that ρ_t converges, nor that its limit is π.

3.3.8.5 Limit points of the dynamics

Let $(t_n)_{n\in\mathbb{N}}$ be a sequence such that $t_n \to \infty$, and let $\rho_\infty \in \mathcal{P}(\Omega)$ such that $\rho_{t_n} \rightharpoonup \rho_\infty$. By Corollary A.3.3, $\mathfrak{K}\left(\rho_{t_n} \mid \pi\right) \to 0$. We want to deduce that $\mathfrak{K}(\rho_\infty \mid \pi) = 0$. By Definition A.3.11, the KSD of ρ is the supremum of $\mathbb{E}_{x\sim\rho}[\mathcal{A}_s f_\Phi(x)]^2$ over the closed unit ball of $\mathcal{H}$. For each such f, this quantity at ρ_{t_n} converges to its value at ρ_∞ by weak convergence, and it is at most $\mathfrak{K}\left(\rho_{t_n} \mid \pi\right)$.

Lemma A.3.13 – *Limit points have a zero KSD*

Suppose that the standing assumptions hold. Let ρ_0 be a probability measure on $\mathbb{R}^d$ such that $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$. Let $(T_t)_{t\geq 0}$ be a solution of Eq. 24, and let $\rho_t = (T_t)\#\rho_0$. Let $(t_n)_{n\in\mathbb{N}}$ be a sequence such that $t_n \to \infty$, and let $\rho_\infty \in \mathcal{P}(\Omega)$ such that $\rho_{t_n} \xrightarrow[n\to\infty]{} \rho_\infty$. Then, $\mathfrak{K}(\rho_\infty \mid \pi) = 0$.

Proof. Let $f \in \mathcal{H}$ such that $\|f\|_{\mathcal{H}} \leq 1$. The function $\mathcal{A}_s f_\Phi$ is continuous on Ω (see Section A.3.3.4.2). By the weak convergence of ρ_{t_n}, and by Definition A.3.11, we have that

$$\mathbb{E}_{x\sim\rho_\infty}[\mathcal{A}_s f_\Phi(x)]^2 = \lim_{n\to\infty} \mathbb{E}_{x\sim\rho_{t_n}}[\mathcal{A}_s f_\Phi(x)]^2 \leq \lim_{n\to\infty} \mathfrak{K}\left(\rho_{t_n} \mid \pi\right) = 0,$$

where the last limit is given by Corollary A.3.3. Thus, $\mathbb{E}_{x\sim\rho_\infty}[\mathcal{A}_s f_\Phi(x)] = 0$ for every f of the closed unit ball of $\mathcal{H}$. By Definition A.3.11, $\mathfrak{K}(\rho_\infty \mid \pi) = 0$, which concludes the proof.

□

3.3.8.6 Identification of the limit

It remains to show that ρ_t has limit points, and that they are all equal to π. The first point follows from the compactness of Ω. On a compact metric space, every sequence of probability measures has a weakly convergent subsequence. This follows from Prokhorov's theorem (Billingsley, 1999, Theorem 5.1). Its tightness condition requires that, for every $\eta > 0$, a single compact set have mass at least $1 - \eta$ under every measure. On a compact space, it holds with the whole space. By the identification of Section A.3.3.8.1, every sequence of $\mathcal{P}(\Omega)$ thus has a subsequence that converges weakly to a measure of $\mathcal{P}(\Omega)$.

Proof. We prove Theorem A.3.9. Suppose that ρ_t does not converge weakly to π. Then, there exist $g \in C^0(\Omega)$ and $\eta > 0$ such that $\left|\mathbb{E}_{x\sim\rho_t}[g(x)] - \mathbb{E}_{x\sim\pi}[g(x)]\right| \geq \eta$ for arbitrarily large t. Thus, there exists a sequence $(t_n)_{n\in\mathbb{N}}$ such that $t_n \to \infty$ and

$$\forall n \in \mathbb{N}, \quad \left|\mathbb{E}_{x\sim\rho_{t_n}}[g(x)] - \mathbb{E}_{x\sim\pi}[g(x)]\right| \geq \eta. \tag{31}$$

As $\rho_{t_n} \in \mathcal{P}(\Omega)$ for every $n \in \mathbb{N}$, Prokhorov's theorem gives a subsequence $(\tau_k)_{k\in\mathbb{N}}$ of $(t_n)_{n\in\mathbb{N}}$ and a measure $\rho_\infty \in \mathcal{P}(\Omega)$ such that $\rho_{\tau_k} \xrightarrow[k\to\infty]{} \rho_\infty$. As $\tau_k \to \infty$, Lemma A.3.13 gives $\mathfrak{K}(\rho_\infty \mid \pi) = 0$. By Proposition A.3.1, $\rho_\infty = \pi$. Then, $\mathbb{E}_{x\sim\rho_{\tau_k}}[g(x)] \to \mathbb{E}_{x\sim\pi}[g(x)]$, which contradicts Eq. 31. Hence, $\rho_t \xrightarrow[t\to\infty]{} \pi$, which concludes the proof.

□

3.3.9 Application to Stein Boltzmann Sampling

We now apply the results of this section to SBS (Section A.3.2). Let $V : \mathbb{R}^d \to \mathbb{R}$ be continuously differentiable, let $\kappa > 0$, and assume that $\lambda(\Omega) > 0$. The target distribution π is the Boltzmann distribution of V on Ω (Definition A.3.9). By Example A.3.1, it satisfies Assumption A.3.1, with $s = -\kappa DV$.

The Gaussian kernel k_σ, used by SBS, does not satisfy (iv) of Assumption A.3.2 (see Figure A.3.18). The results of this section thus apply to SBS with another kernel. For instance, one can weight k_σ by a function $w : \mathbb{R}^d \to \mathbb{R}$ that vanishes on $\partial\Omega$, i.e.

$$k(x, y) = w(x)w(y)k_\sigma(x, y).$$

Let H_σ be the reproducing kernel Hilbert space of k_σ, and let $\Phi_\sigma(x) = k_\sigma(\cdot, x)$ be its feature map. Then, $\Phi = w\Phi_\sigma$ is a feature map of k, with values in H_σ.

Remark

Let $\sigma > 0$, and let $w : \mathbb{R}^d \to \mathbb{R}$ be continuously differentiable, bounded by C_w and L_w-Lipschitz, with $w(y) = 0$ for every $y \in \partial\Omega$. Then, $\Phi = w\Phi_\sigma$ satisfies Assumption A.3.2, with $K = L_w + C_w\sqrt{2/\sigma}$ and $C = C_w$. Indeed, as k_σ is infinitely differentiable, Φ_σ is continuously differentiable (see Zhou (2008)), and $\|\Phi_\sigma(x)\|^2_{H_\sigma} = k_\sigma(x,x) = 1$ for every $x \in \mathbb{R}^d$. Thus, Φ is continuously differentiable, bounded by C_w, and zero on $\partial\Omega$. Moreover, as $1 - e^{-u} \le u$ for every $u \ge 0$, we have that

$$\|\Phi_\sigma(x) - \Phi_\sigma(y)\|^2_{H_\sigma} = 2\left(1 - e^{-\|x-y\|^2/\sigma}\right) \le \frac{2}{\sigma}\|x - y\|^2$$

for all $x, y \in \mathbb{R}^d$. Thus,

$$\begin{aligned}\|\Phi(x) - \Phi(y)\|_{H_\sigma} &\le |w(x) - w(y)|\|\Phi_\sigma(x)\|_{H_\sigma} + |w(y)|\|\Phi_\sigma(x) - \Phi_\sigma(y)\|_{H_\sigma} \\ &\le \left(L_w + C_w\sqrt{2/\sigma}\right)\|x - y\|.\end{aligned}$$

The results of this section apply to SBS with any kernel that satisfies Assumption A.3.2, as follows.

Corollary A.3.4 – *Continuous-time SBS*

Let $V : \mathbb{R}^d \to \mathbb{R}$ be continuously differentiable, let $\kappa > 0$, and assume that $\lambda(\Omega) > 0$. Let π be the Boltzmann distribution of V on Ω, and let k be a kernel whose feature map Φ satisfies Assumption A.3.2. Let $\rho_0 \in \mathcal{P}(\Omega)$. Then, the following holds.

(i) The continuous-time dynamics Eq. 24 from ρ_0 has a solution $(T_t)_{t\ge0}$, and $T_t(\Omega) = \Omega$ for every $t \ge 0$.

(ii) If $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$, then, for every such solution, the function $t \mapsto \mathrm{KL}(\rho_t \parallel \pi)$, where $\rho_t = (T_t)\#\rho_0$, is nonincreasing, with derivative $-\mathfrak{K}(\rho_t \mid \pi)$.

(iii) If, moreover, k is Stein-universal, then $\rho_t \xrightarrow[t\to\infty]{} \pi$.

In particular, (ii) and (iii) hold for the uniform distribution $\rho_0 = \lambda\|_\Omega$.

Proof. Using Example A.3.1, the standing assumptions hold. Thus, Theorem A.3.6 and Lemma A.3.9 give (i). Then, Theorem A.3.8 gives (ii), and Theorem A.3.9 gives (iii). Finally, let $\rho_0 = \lambda\|_\Omega$. The density of ρ_0 with respect to π is $1/(\lambda(\Omega)p)$ on Ω. As p reaches a positive minimum on Ω, this density is bounded, and so is its logarithm. Thus, $\mathrm{KL}(\rho_0 \parallel \pi) < \infty$, which concludes the proof.

□

Thus, the continuous-time dynamics of SBS exists and keeps the particles in Ω. From any initial distribution with a finite Kullback–Leibler divergence to the Boltzmann distribution, such as the uniform distribution, it decreases this divergence. If a kernel that satisfies Assumption A.3.2 is Stein-universal (Definition A.3.12), this dynamics converges weakly to the Boltzmann distribution. As κ tends to infinity, this distribution concentrates on the minimizers of V (see Section A.3.2). The

Gaussian kernel used in the experiments of Section A.3.2 does not satisfy Assumption A.3.2. The construction of a Stein-universal kernel that satisfies Assumption A.3.2, and new experiments with this kernel, are left for future work. Moreover, these results concern the continuous-time dynamics, and not the iteration with finitely many particles that `SBS` implements.

3.4 Common noise plugin module

In previous sections, we presented several global optimization algorithms that are based on McKean-Vlasov systems of N interacting particles, written as

$$\mathrm{d}X_t^i = b\big(X_t^i, \hat{\rho}_{X_t}\big)\,\mathrm{d}t + \sigma\big(X_t^i, \hat{\rho}_{X_t}\big)\,\mathrm{d}B_t^i \quad \text{(Definition A.3.8)}.$$

In the mean-field limit as $N \to \infty$, the empirical distribution of particles $\hat{\rho}_{X_t}$ converges to a deterministic probability law ρ_t (see Eq. 17), reducing the system to deterministic mean-field dynamics. This deterministic evolution of ρ_t, arising from averaging independent Brownian motions, potentially limits exploration at the population level, causing the dynamics to fail at exploring multiple modes of complex landscapes and to get trapped in suboptimal basins (Tugaut, 2014).

To address this limitation, we inject noise directly into the probability law evolution itself, that we call *ρ-noise* for the sake of conciseness and consistency. This induces correlated motions, enabling collective transitions and facilitating exploration of multiple basins of attraction. This modular approach allows our noise injection strategies to be applied generically to any MKV-based method, treating noise enhancement as a **plugin module** that is independent of the underlying dynamics.

The contribution of this module is twofold:

i. **This is the first work in which *ρ-noise* is used for global optimization**, leading to a plugin approach that extends the practical relevance of existing contributions in probability theory, notably *Stochastic Moment Dynamics* (SMD) (Germain & Monmarché, 2025). SMD relies on conditional MKV theory and introduces finite-dimensional common noise, designed so that key observables (e.g. mean or variance) follow a prescribed stochastic evolution.

ii. **We design an original ρ-noise, which we claim to be canonical when the MKV system is derived from a gradient flow.** We introduce *Geometric Common Noise* (GCN), a novel geometric approach that exploits the tangent-space structure of probability measure spaces to induce Langevin-type dynamics at the level of probability laws. This perspective opens new questions in SPDE theory (Theorem A.3.11). Our analysis further shows that a variant of SMD can be interpreted as a limiting case of GCN (Theorem A.3.12).

These contributions result from a collaboration with Pierre Germain and Samuel Gruffaz. This thesis focuses primarily on the implementation and empirical evalu-

ation of these modules, though we include the theoretical development to provide a rigorous mathematical foundation and defer deepest details and proofs in Appendix A.

3.4.1 Noise through moment perturbation

Mean-field optimization and particle systems. Particle-based optimization methods (Pinnau et al., 2017; Roberts & Tweedie, 1996; Serré et al., 2025) can be viewed as evolutions in the space of probability measures, approximated by interacting particle systems. When particles are driven only by independent noise, stochasticity vanishes in the mean-field limit due to propagation of chaos, often resulting in poor exploration and convergence to local minima (Monmarché & Reygner, 2025; Tugaut, 2014).

Common noise and conditional MKV theory. A natural remedy is to introduce a *common noise* that acts on all particles simultaneously. Unlike independent noise, common noise persists in the mean-field limit and induces 'genuinely' stochastic dynamics for the limiting distribution. This setting, known as *conditional McKean–Vlasov theory* (Carmona & Delarue, 2018), has recently been used to improve exploration in mean-field games (Delarue et al., 2024; Delarue & Vasileiadis, 2025; Maillet, 2023), and design structured perturbations (Delarue & Hammersley, 2025). It is particularly motivated by two failure modes: particle collapse (vanishing variance) and collective trapping in suboptimal basins.

Macroscopic perturbations via stochastic moments. To address these issues, one can perturb directly macroscopic observables (e.g. the empirical mean or variance). This is achieved by augmenting the mean-field dynamics with a suitably designed common noise so that the induced evolution of these moments remains stochastic in the mean-field limit. The resulting ρ-noise, termed *Stochastic Moment Dynamics* (SMD), was introduced in (Germain & Monmarché, 2025) for general observables, together with a theoretical analysis. Here, we provide a self-contained presentation of the method, with a focus on practical implementation.

Macroscopic observable of interest F. Let $N \in \mathbb{N}$, starting from the baseline mean-field particle dynamics Eq. 16, choose a macroscopic observable $Y : \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}^p$ from:

- Mean: $Y_{\text{mean}}(\rho) \overset{\text{def}}{=} \int_{\mathbb{R}^d} x, \mathrm{d}\rho(x)$;
- Second-order moment: $Y_{\text{s.m}}(\rho) \overset{\text{def}}{=} \left(\int_{\mathbb{R}} x^2, \mathrm{d}\rho_{j(x)}\right)_{j\in[d]}$;
- Variance: $Y_{\text{var}}(\rho) \overset{\text{def}}{=} \left(\mathrm{Var}(\rho_j)\right)_{j\in[d]}$;
- Mean + variance: $[F_{\text{mean}}(\rho), F_{\text{var}}(\rho)]$.

SMD General Recipe. The method consists in adding a stochastic perturbation in Eq. 16 according to the following equation

$$\mathrm{d}X_t^i = b\left(X_t^i, \hat{\rho}_{X_t}\right)\mathrm{d}t + \sigma\left(X_t^i, \hat{\rho}_{X_t}\right)\mathrm{d}B_t^i + \beta\left[\tilde{b}\left(X_t^i, \hat{\rho}_{X_t}\right)\mathrm{d}t + \tilde{\sigma}\left(X_t^i, \hat{\rho}_{X_t}\right)\mathrm{d}B_t\right] \quad (32)$$

where $\beta \geq 0$ is an intensity parameter and B_t is a p-dimensional Brownian motion acting on all particles. The resulting dynamics Eq. 32 can thus be viewed as a

combination of two effects: a deterministic mean-field drift–associated to b and σ (driving the system toward minimizers of the objective function)–and a stochastic perturbation (associated to $\tilde{b}$ and $\tilde{\sigma}$) providing macroscopic exploration. The intensity β of this additional exploration perturbation can be tuned in close analogy with the role of temperature in the finite-dimensional Langevin dynamics (see Eq. 18).

Consider independently the additional perturbation driven by $\tilde{b}$ and $\tilde{\sigma}$

$$\mathrm{d}Y_t^i = \tilde{b}\left(Y_t^i, \hat{\rho}_{Y_t}\right)\mathrm{d}t + \tilde{\sigma}\left(Y_t^i, \hat{\rho}_{Y_t}\right)\mathrm{d}B_t. \tag{33}$$

The coefficients $\tilde{b}$ and $\tilde{\sigma}$ are defined so that the chosen macroscopic quantity of $\hat{\rho}_{Y_t}$, independently from the number N of particles in the system, solves the stochastic differential equation given by some coefficients $a \in \mathbb{R}^p$ and $s \in \mathbb{R}^{p\times p}$

$$\mathrm{d}Y\left(\hat{\rho}_{Y_t}\right) = a\left(F\left(\hat{\rho}_{Y_t}\right)\right)\mathrm{d}t + s\left(F\left(\hat{\rho}_{Y_t}\right)\right)\mathrm{d}B_t. \tag{34}$$

The expression of $\tilde{b}$ and $\tilde{\sigma}$ is given by an explicit formula from Germain & Monmarché (2025) and only depends on Y, a and s. Therefore, given a choice of a macroscopic quantity Y, a critical aspect of the method lies in the choice of the coefficients a and s. When Y is positive, a, s are chosen such that Eq. 34 is a Bessel process in such way that $Y\left(\hat{\rho}_{Y_t}\right)$ stay positive, i.e. $a_+^{\delta(z)} = \left(\frac{\delta-\frac{1}{2}}{z_i}\right)_{i\in[d]}$, $s_+^{\delta(z)} = I_d$ where $\delta \geq 2$ is a parameter whose role is explained in Appendix A.1. The formula of $\left(\tilde{b}, \tilde{\sigma}\right)$ are given for any choice of perturbation $Y \in \{F_{\text{mean}}, F_{\text{s.m}}, F_{\text{var}}, [F_{\text{mean}}, F_{\text{var}}]\}$ in the following lemma.

Lemma A.3.14

SMD-Mean $[F = F_{\text{mean}}, (a, s) = (0_d, I_d)]$

$$\tilde{b}(x,\rho) \overset{\text{def}}{=} 0, \tilde{\sigma}(x,\rho) \overset{\text{def}}{=} I_d \tag{35}$$

SMD Second-order moment $\left[F = F_{\text{s.m}}, (a, s) = \left(a_+^\delta, s_+^\delta\right)\right]$

$$\tilde{b}(x,\rho) \overset{\text{def}}{=} \left(\delta - \frac{3}{2}\right)\left[\frac{x}{4m_2(\rho)^2}\right], \tilde{\sigma}(x,\rho) \overset{\text{def}}{=} \text{Diag}\left[\frac{x}{2m_2(\rho)}\right]. \tag{36}$$

SMD Variance $\left[F = F_{\text{var}}, (a, s) = \left(a_+^\delta, s_+^\delta\right)\right]$

$$\tilde{b}(x,\rho) \overset{\text{def}}{=} \left(\delta - \frac{3}{2}\right)\left[\frac{x - m(\rho)}{4\text{Var}(\rho)^2}\right], \tilde{\sigma}(x,\rho) \overset{\text{def}}{=} \text{Diag}\left[\frac{x - m(\rho)}{2\ \text{Var}(\rho)}\right]. \tag{37}$$

SMD Mean + variance $\left[F = F_{\text{mean}}, F_{\text{var}}, (a, s) = \left((0_d, a_+^\delta), (I_d, s_+^\delta)\right)\right]$

$$\tilde{b}(x,\rho) \overset{\text{def}}{=} \left(\delta - \frac{3}{2}\right)\left[\frac{x - m(\rho)}{4\text{Var}(\rho)^2}\right], \tilde{\sigma}(x,\rho) \overset{\text{def}}{=} \left(I_d, \text{Diag}\left[\frac{x - m(\rho)}{2\ \text{Var}(\rho)}\right]\right). \tag{38}$$

The derivation of the coefficients $\tilde{b}$ and $\tilde{\sigma}$ is detailed in Appendix A.3.1, which were given only for dimension $d = 1$ in (Germain & Monmarché, 2025). Note that divisions by 0 may occur; see Appendix A.1 for more details. However, no explosion is observed in any of our experiments (see Figure A.3.26).

Simulation scheme. We use an Euler-Maruyama discretization of Eq. 32 to compute the evolution of the particle system. Let $(\varepsilon_n)_{n\in\mathbb{N}}$ be a sequence of positive (and possibly decreasing) time step. The Euler-Maruyama scheme associated with Eq. 32 is given, for each particle $i \in [N]$, by

$$\begin{aligned} X^i_{n+1} = X^i_n &+ \varepsilon_{n+1} \cdot \left[b\left(X^i_n, \hat{\rho}_{X_n}\right) + \tilde{b}\left(X^i_n, \hat{\rho}_{X_n}\right)\right] \\ &+ \sqrt{\varepsilon_{n+1}} \cdot \left[\sigma\left(X^i_n, \hat{\rho}_{X_n}\right)\xi^i_n + \tilde{\sigma}\left(X^i_n, \hat{\rho}_{X_n}\right)\zeta_n\right], \end{aligned} \tag{39}$$

where $(\xi^i_n)_{i\in[N],n\geq 0}$ and $(\zeta_n)_{n\geq 0}$ are independent standard Gaussian increments of respective dimensions d and p.

The empirical efficiency of these methods is tested on a benchmark of multi-modal functions across multiple MKV dynamics in Section A.3.4.3. Depending on the nature of the objective function to minimize, one should adapt its choice of perturbed macroscopic observable to maximize the efficiency of the method, even though Mean+Var will be recommended as a practical default.

3.4.2 Noise using tangent space

The second approach relies on the geometric structure of the considered probability law space related to a dynamic. We introduce an infinite-dimensional noise that we claim to be canonical by injecting noise in tangent spaces related to the curve $t \mapsto \rho_t$. For the presentation of the approach, we follow the case-study of Stein Boltzmann Sampling (SBS) (see Section A.3.2). The recipe is summarized at the end of the section, the simulation scheme is detailed in Eq. 47.

SBS mean-field dynamics. Given a kernel $k_{\mathsf{SBS}} : (x, y) \in \mathbb{R}^d \times \mathbb{R}^d \mapsto k_{\mathsf{SBS}}(x, y)$, typically a Gaussian kernel $k_{\mathsf{SBS}} = k^{(d)}_\sigma$, the SBS's N-particle dynamic is a MKV dynamics with zero diffusion and drift given by

$$b^{\mathsf{SBS}}(\cdot, \rho) = \mathbb{E}_{x\sim\rho}[-\kappa k_{\mathsf{SBS}}(\cdot, x)\nabla V(x) + \nabla_x k_{\mathsf{SBS}}(\cdot, x)] \quad \text{(Eq. 20)}.$$

Recall that, when $N \to \infty$ and under mild conditions, $\hat{\rho}_{X_t} \to \rho^{\mathsf{SBS}}_t$ following the mean-field dynamics

$$\partial_t \rho^{\mathsf{SBS}}_t = -\nabla \cdot (b^{\mathsf{SBS}}(\cdot, \rho^{\mathsf{SBS}}_t)\rho^{\mathsf{SBS}}_t) \quad \text{(Eq. 17)}, \tag{40}$$

In the following, we detail the structure of Eq. 40 to clarify the ρ-*noise*. We set the temperature parameter κ to 1 for simplicity.

Gradient flow on Wasserstein space. Eq. 40 is in fact a gradient flow related to a specific geometry on the space of probability law (Duncan et al., 2023b). To give an intuition on this interpretation, note that a sum of two probability measures (ρ, ν) is not a probability measure since $(\rho + \nu)(\mathbb{R}^d) = 2$. Instead, due to mass conservation, the following property holds: if a curve $t \in [0, 1] \to \rho_t \in \mathcal{P}_2 = \left\{\rho \in \mathcal{P}(\mathbb{R}^d) : \mathbb{E}_\rho[|X|^2] < \infty\right\}$ is absolutely continuous, then it satisfies the continuity equation in the sense of distribution (Ambrosio et al., 2005, Theorem 8.3.1), i.e., there exists $v_t \in L^2(\rho_t)$ for any $t \in (0, 1)$ such that

$$\partial_t \rho_t + \nabla \cdot (v_t \rho_t) = 0. \tag{41}$$

This means that an infinitesimal displacement around ρ_t has the form $\nabla \cdot (v_t \rho_t)$. Eq. 40 is a particular case of this equation, where $v_t = b^{\mathsf{SBS}}(\cdot, \rho_t)$. In the following, we will refer to v_t as the velocity field of SBS's dynamics.

Note that the discretization of Eq. 41 can be derived using

$$\hat{\rho}_{(n+1)\varepsilon} = (\mathrm{id} + \varepsilon v_{n\varepsilon}) \# \hat{\rho}_{n\varepsilon}, \quad \text{since} \quad \lim_{\varepsilon \to 0} \frac{(\mathrm{id} + v_t) \# \rho_t - \rho_t}{\varepsilon} = \nabla \cdot (v_t \rho_t).$$

For these reasons, the functional space where v_t belongs can be identified as a tangent space using Riemannian geometry vocabulary, and by using the pushforward operation '#' instead of the addition '+'. This statement is made rigorous in Ambrosio et al. (2005, Theorem 8.5.1).

In general, the tangent space is as subspace of

$$L^2(\rho_t) = \left\{ f : \mathbb{R}^d \mapsto \mathbb{R}^d, \text{measurable} : \int |f|^2 d \, \mathrm{d}\rho_t < \infty \right\}.$$

In the case of sbs's dynamics, the functional space is related to a subspace of a product RKHS generated by the kernel $K_{\mathsf{SBS}} = k_{\mathsf{SBS}} I_d$ (see Section A.3.3.4.1 and Duncan et al. (2023b)).

Therefore, to *inject canonical noise* in a trajectory of probability laws (ρ_t), we only have to inject canonical noise in the tangent space where v_t belongs according to its structure,

$$\hat{\rho}_{(n+1)\varepsilon} = \left(\mathrm{id} + \varepsilon v_{n\varepsilon} + \sqrt{\varepsilon} G_n\right) \# \hat{\rho}_{n\varepsilon}, \; G_n \sim \mathcal{G}. \tag{42}$$

The question is then, how to add canonical noise in the tangent space such that its time increments $(G_n)_{n \in \mathbb{N}} \sim \mathcal{G}$ act as an additive noise in equation Eq. 42? Can we characterize their law $\mathcal{G}$, as well as the law $\rho_\cdot^*$ of the resulting continuous time noising process?

Canonical noise in RKHS. Brownian motion, or any Gaussian process, is a random variable in an Hilbert space of functions typed as $\mathbb{R} \to \mathbb{R}$. Its natural generalization to Hilbert spaces of functions $\mathbb{R}^d \to \mathbb{R}^d$ is called a d-dimensional space Gaussian process, or Gaussian random field (Estrade & Fournier, 2025; MacKay & others, 1998). In the following, we focus on RKHS-valued Gaussian random fields.

Definition A.3.13 – *$\mathcal{H}_d$-Gaussian random field*

A random variable $Z : \Omega \to \mathcal{H}_d$ is called a (centered) $\mathcal{H}_d$-Gaussian random field if, for every $f \in \mathcal{H}_d$, the real-valued random variable $\langle G, f \rangle_{\mathcal{H}_d}$ is a centered Gaussian random variable on $\mathbb{R}$. Moreover, if $\mathbb{E}\|G\|^2_{\mathcal{H}_d} < \infty$, then its law is characterized by a covariance kernel $K : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{S}_d^{++} \subset \mathbb{R}^{d \times d}$ and we denote $Z \sim \mathcal{G}(K)$. More precisely, for any $X = (x_i)_{i \in [N]} \in \left(\mathbb{R}^d\right)^N$, $(G(x_i))_{i \in [N]}$ is a $\mathbb{R}^{Nd}$ Gaussian vector, whose covariance matrix $\overline{K}(X) = \left(K_{i,j}(X)\right)_{i,j \in [N]} \in \mathbb{R}^{Nd \times Nd}$ is given by the blocks

$$K_{i,j}(X) = K(x_i, x_j) = \mathbb{E}\left[G(x_i)Z(x_j)^T\right] \in \mathbb{R}^{d\times d}.$$

Heuristically, a centered Gaussian process Z with covariance kernel $\overline{K}$ can be interpreted as a standard Gaussian on the RKHS $\mathcal{H}_d$, formally associated with density $\exp\left(-\frac{1}{2}\|G\|^2_{\mathcal{H}_d}\right)$. This interpretation cannot be made rigorous in $\mathcal{H}_d$, since infinite-dimensional Hilbert spaces do not admit a Lebesgue reference measure. A rigorous meaning is obtained by restricting Z to finite-dimensional subspaces $\mathcal{H}_d^X$ generated by kernel evaluations

$$\mathcal{H}_d^X = \text{span}\left(K(x_i, \cdot)e_j, i, j \in [N] \times [d]\right) \subset \mathcal{H}_d,$$

by considering its orthogonal projection $\pi_X Z$ into $\mathcal{H}_d^X$.

Lemma A.3.15

Let $X = (x_1, ..., x_N)$ with $x_i \neq x_j$ if $i \neq j$, and let $Z \sim \mathcal{G}(K_{\mathsf{SBS}})$. Then the projection $\pi_X Z \in \mathcal{H}_d^X$ admits a density $\mathbb{P}(\pi_X Z \in \mathrm{d}f)$ with respect to the Lebesgue measure induced by the RKHS inner product, given by

$$\mathbb{P}(\pi_X Z \in \mathrm{d}f) \propto \exp\left(-\frac{1}{2}\|f\|^2_{\mathcal{H}_d}\right).$$

Equivalently, $Z(X) = (G(x_1), ..., Z(x_N)) \in \mathbb{R}^{Nd}$ is a centered Gaussian vector with density proportional to $\exp\left(-\frac{1}{2}, g^\top \overline{K}(X)^{-1} g\right)$ with respect to the usual Lebesgue measure on $\mathbb{R}^{Nd}$.

All theoretical details are deferred to Appendix A.3.2.

Lemma A.3.15 justifies that $Z \sim \mathcal{G}(K_{\mathsf{SBS}})$ is Gaussian standard in $\mathcal{H}_d$, and thus is considered canonical. The following corollary justifies that d-dimensional space Gaussian process is an infinite dimensional noise.

Corollary A.3.5

When the kernel $k_{\mathsf{SBS}}(x, y) = k_\sigma^{(d)}(x, y)$ is Gaussian, $Z \overset{\text{def}}{=} (G_i)_{i\in[d]} \sim \mathcal{G}(K_{\mathsf{SBS}})$ can be simulated as follows (Lototsky et al., 2017, Eq 1.1.11)

$$G_i = \sum_{\vec{n}\in\mathbb{N}^d} \lambda_{\vec{n}}\phi_{\vec{n}}(\cdot)\xi^i_{\vec{n}}, \xi^i_{\vec{n}} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1), \vec{n} \in \mathbb{N}^d,$$

where $\left(\Phi_{\vec{n}} = \sqrt{\lambda_{\vec{n}}}\phi_{\vec{n}}\right)_{\vec{n}\in\mathbb{N}^d}$ is an orthonomal basis of the RKHS related to $k_\sigma^{(d)}$ (Berlinet & Thomas-Agnan, 2011; Da Prato & Zabczyk, 2014).

The central observation is that if we aim to inject canonical noise in SBS/SVGD, we should add time-dependent noise in Eq. 41, whose time increments are i.i.d. $\mathcal{H}_d$-Gaussian random field of covariance kernel K_{SBS}, i.e. $\mathcal{G} = \mathcal{G}(\beta^2 K_{\mathsf{SBS}})$ in Eq. 42 where $\beta > 0$ is a temperature parameter.

Continuous Langevin-type dynamics. Our goal is to study the time-continuous dynamics related to Eq. 42 when injecting canonical noise $\mathcal{G} = \mathcal{G}(\beta^2 K_{\mathsf{SBS}})$, to get a Langevin-type dynamic on the space of probability laws, corresponding to a

gradient flow Eq. 41 with a canonical ρ-*noise*. Since the continuity equation is a partial differential equation, its stochastic counterpart naturally takes the form of a stochastic partial differential equation (SPDE). Accordingly, the noise must be modeled as a Gaussian space-time process $(G_t)_{t\geq 0} \sim \mathcal{TZ}(K_{\mathsf{SBS}})$ with increments $Z_{t+\varepsilon} - G_t \sim \mathcal{G}(\varepsilon K_{\mathsf{SBS}}))$, which can be viewed as the infinite-dimensional analogue of Brownian motion. In the SPDE, this leads to the appearance of the infinitesimal noise term $\mathrm{d}G_t$, interpreted as Gaussian white noise, that is, the distributional time derivative of Z_t (Lototsky et al., 2017, Eq. 1.1.5); see Appendix A.2 for an introduction to these objects.

Following Corollary A.3.5, in the case of SBS, the infinitesimal variation of $(G_t)_{t\geq 0} = (G_t^i)_{i\in[d],t\geq 0} \sim \mathcal{TZ}(K_{\mathsf{SBS}})$ can be represented as follows

$$\mathrm{d}G_i(t,\cdot) = \sum_{\vec{n}\in\mathbb{N}^d} \lambda_{\vec{n}}\phi_{\vec{n}}(\cdot)\,\mathrm{d}\xi^i_{\vec{n},t},$$

where $\left(\xi^i_{\vec{n},t}\right)_{t\geq 0}$ are independent Brownian motions (Lototsky et al., 2017, Eq 1.1.10).

To define a valid continuous time dynamic, the idea is first to consider the simplified case where $\rho_0 = \hat{\rho}_0$ is the empirical measure of a particle system. Because the dynamic of the system is fully described by the motion of the particles, the SPDE can be rewritten as an SDE governing the evolution the particles. The initial SPDE is then obtained as a limit of the considered SDEs as $N \to \infty$. The SDE describing the motions of $N+1$ particles is derived from Eq. 43.

Theorem A.3.10

The SDE being the time-continuous/discretized in space version of Eq. 42 with $\mathcal{G} = \mathcal{G}(\beta^2 K_{\mathsf{SBS}})$ and $\beta > 0$ is

$$\mathrm{d}X_t = b^{\mathsf{SBS}}(X_t, \rho_t^N)\,\mathrm{d}t + \beta\overline{K}^{\frac{1}{2}}(X_t)\,\mathrm{d}B_t^{Nd}, \tag{43}$$

with $\rho_t^N = \frac{1}{N}\sum_{i=1}^N \delta_{X_t^i}$ is the empirical measure of the particle system, $\left(B_t^{Nd}\right)_{t\geq 0}$ is a Nd dimensional Brownian motion and $\overline{K}(X) = K_{\mathsf{SBS}}(X) \otimes I_d \in \mathbb{R}^{dN\times dN}$, $K_{\mathsf{SBS}}(X) \overset{\text{def}}{=} \left(k_{\mathsf{SBS}}(X^i, X^j)\right)_{i,j\in[N]}$. Moreover, under mild regularity and growth conditions on V and k, detailed in Assumption A.A.3, the laws of the empirical distributions of the particle systems $\left(\mathcal{L}(\rho^N)\right)_{N\geq 1}$ are tight in the space of probability measures over $C\left([0,T], \mathcal{P}_1(\mathbb{R}^d)\right)$ equipped with the topology of uniform convergence, which implies the existence of converging subsequences.

The proof of this result is given in Appendix A.3.3.

In the following, the term

$$\overline{K}^{\frac{1}{2}}(X_t)\,\mathrm{d}B_t^{Nd} \tag{44}$$

will be referred as *Geometrical Common Noise* (GCN). This diffusion coefficient is the consequence of the addition of the $\sqrt{\varepsilon}G_n$ term in Eq. 42.

Surprisingly, when $\beta = \beta_N \overset{\text{def}}{=} \frac{\sqrt{2}}{\sqrt{N+1}}$ the dynamic Eq. 43 was already studied in the literature under the name of Stochastic Langevin with repulsive force; see Gallego & Insua (2018); Nüsken & Renger (2023) and Duncan et al. (2023b, Section 3). In this setting the noise vanishes at the large N limit and the evolution of ρ_t^N becomes deterministic. Contrarily to GCN, where, under the scaling $\beta_N = \beta > 0$, the noise Eq. 44 doesn't vanish when $N \to \infty$, making GCN a *ρ-noise*. To establish this result, rather than tackling the SPDE directly, we consider a weaker formulation, namely a martingale problem for measure-valued processes (Cuchiero et al., 2024; Dawson & others, 1994; Erny, 2025).

Theorem A.3.11 – *Informal, detailed in Appendix A.3.4*

Assuming ∇V is Lipschitz continuous and mild conditions on (ρ_0^N), detailed in Theorem A.A.14, any limit point $(\rho_t^*)_{t\geq 0}$ of $(\rho^N)_{N\geq 1}$ is solution of a martingale problem for measure valued processes that is heuristically related to the following SPDE,

$$\partial_t \rho_t^* + \nabla \cdot ((b^{\mathsf{SBS}}(\cdot, \rho_t^*) + \beta \, \mathrm{d}G_t))\rho_t^*) + \frac{\beta^2}{2}\Delta_t\left(k_{\mathsf{SBS}}^{\mathrm{diag}}\rho_t^*\right) = 0, \;\; Z \sim \mathcal{TZ}(K_{\mathsf{SBS}}) \quad (45)$$

where $k_{\mathsf{SBS}}^{\mathrm{diag}}(x) \overset{\text{def}}{=} k_{\mathsf{SBS}}(x,x)$. Moreover, in case where $k_{\mathsf{SBS}}(x,x) = 1$ for any $x \in \mathbb{R}^d$, which is verified when $k_{\mathsf{SBS}} = k_\sigma^{(d)}$, ρ_t^* can be approximated using that

$$\rho_{t+\varepsilon}^* \approx (\mathrm{id} + T_t)\#\rho_t^*, \;\; T_t = \varepsilon b^{\mathsf{SBS}}(\cdot, \rho_t^*) + \sqrt{\varepsilon}\beta g_t, \;\; g_t \sim \mathcal{G}(K_{\mathsf{SBS}}). \quad (46)$$

Tightness of the sequence of laws (Theorem A.3.10), together with the identification of any limit point as a solution to the martingale problem (Theorem A.3.11), ensures the existence of at least one solution to the considered martingale problem. Showing uniqueness and then propagation of chaos is still an open difficult question left for future works.

In practice, only the dynamics the particles is simulated using Eq. 43, whose the Euler-Maruyama integration scheme is given with details in Eq. 47. In our experiments the kernel k is Gaussian. The scheme given in Eq. 46 is here only to give more intuition on $\mathrm{d}Z$ in Eq. 45, which becomes βg_t in Eq. 46, accounting the stochastic limit-effect of the infinitesimal d-dimensional space Gaussian process noise that was injected in Eq. 43.

By seeing $t \mapsto \rho_t^{\mathsf{SBS}}$ as a gradient flow on a Wasserstein space (Duncan et al., 2023b), the dynamic $t \mapsto \rho_t^*$ given in Theorem A.3.11 can be seen as a Langevin dynamic on the space of probability law related to the metric given in Duncan et al. (2023b, Definition 7), that is why we consider the noise Eq. 44 as canonical compared to SMD (Section A.3.4.1).

Generalization of the approach. In general, the kernel related to SVGD and the Gaussian process covariance function could be chosen differently, and the noise given by Eq. 44 can be added to any MKV dynamics, the limit-effect will keep the same structure than for SVGD. However, GCN cannot be considered as canonical if there is no link to the gradient flow structure of the original dynamic. That said,

GCN opens the way to new geometrically motivated ρ-*noise* constructions, which may emerge in future work at the intersection of geometry and probability.

Link between GCN and SMD-mean. The following theorem shows that GCN is an interpolation between an infinite-dimensional white noise and SMD-mean, which is a d-dimensional noise, depending on the bandwidth parameter σ when $k = k_\sigma^{(d)}$.

Theorem A.3.12

If the kernel k generating covariance kernel is Gaussian, then, as $\sigma \to \infty$, GCN reduces to SMD-mean, while for $\sigma \to 0$ it reduces to white noise.

The proof of this result is given in Appendix A.3.5.

Simulation scheme. We use an Euler-Maruyama discretization of Eq. 43 to compute the evolution of the particle system. Let $(\varepsilon_n)_{n\in\mathbb{N}}$ be a sequence of positive (and possibly decreasing) time steps. Denoting by k_{noise} the kernel used for Geometric Common Noise (in the previous sections $k_{\text{noise}} = k_{\mathsf{SBS}}$ to have a canonical meaning, but this is not necessary in general), the Euler-Maruyama scheme for the GCN-noised N-particle dynamics is given by

$$X_{n+1}^i = \overline{X}_n^i + \varepsilon_{n+1} \cdot \overline{b}_n + \sqrt{\varepsilon_{n+1}}\overline{\sigma}_n \overline{\xi}_n^{(1)} + \sqrt{\varepsilon_{n+1}} \cdot \overline{K}_{\overline{X}_n}^{1/2} \overline{\xi}_n^{(2)}, \tag{47}$$

where $\overline{X}_n \overset{\text{def}}{=} (X_n^1, ..., X_n^N) \in \mathbb{R}^{Nd}$ is the concatenated state of all particles at time n, and

$$\begin{aligned}
\overline{b}_n &\overset{\text{def}}{=} \left(b\left(X_n^i, \hat{\rho}_{X_n}\right)\right)_{i\in[N]}, \\
\overline{\sigma}_n &\overset{\text{def}}{=} \left(\sigma\left(X_n^i, \hat{\rho}_{X_n}\right)\right)_{i\in[N]}, \\
\hat{\rho}_{X_n} &\overset{\text{def}}{=} \frac{1}{N}\sum_{i=1}^N \delta_{X_n^i}, \\
\overline{K}_{\overline{X}_n} &\overset{\text{def}}{=} K_{\overline{X}_n} \otimes I_d, \\
K_{\overline{X}_n} &\overset{\text{def}}{=} \left(k_{\text{noise}}(X_n^i, X_n^j)\right)_{i,j\in[N]} \in \mathbb{R}^{N\times N},
\end{aligned}$$

and $\overline{\xi}_n^{(1)}, \overline{\xi}_n^{(2)} \sim \mathcal{N}(\mathbf{0}_{Nd}, I_{Nd})$ are independent standard Gaussian increments.

3.4.3 Numerical experiments

We validate our noise injection methods on multimodal benchmark functions (Malherbe & Vayatis, 2017a; Pinnau et al., 2017; Serré et al., 2025), comparing standard MKV dynamics (CBO, SBS, LANGEVIN, MSGD) against their SMD and GCN variants. Implementation uses the GLOBE library, with Euler-Maruyama schemes from Eq. 39 and Eq. 47 (see Chapter A.4). The experiments were conducted on a gradient descentconsumer-grade laptop. Our contribution is to propose noise injection plugins for established MKV methods. Demonstrating that these plugins provide consistent improvements across multiple host algorithms validates this approach. Comparisons against fundamentally different optimizers such as CMA-ES (Hansen & Ostermeier, 2001) or Bayesian optimization fall outside this plugin-based framework, as they address orthogonal algorithmic choices rather than the modular

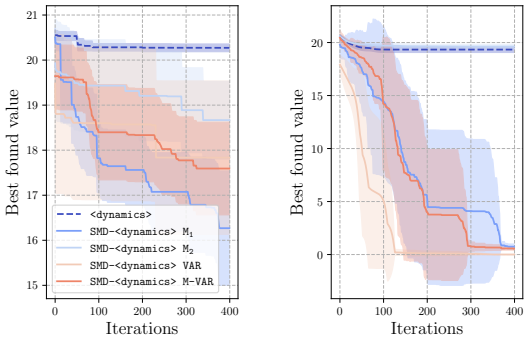
noise injection mechanisms we propose. The synthetic benchmarks we consider are standard in the global optimization literature (Carrillo et al., 2021; Hansen & Ostermeier, 2001; Malherbe et al., 2016; Malherbe & Vayatis, 2017a; Pinnau et al., 2017; Serré et al., 2025). They are characterized by high multimodality, deceptive gradients, narrow basins of attraction, and complex global structure, offering a well-controlled and reproducible evaluation environment across a diverse range of geometries. Their domain-agnostic nature also ensures that our benchmarks remain broadly applicable to the MKV community. Figure A.3.26 illustrates the 2D landscapes of the considered benchmarks, highlighting their diversity and complexity.

Each dynamic is evaluated on 7 multimodal benchmarks in dimension 20 with 150 particles over 300 iterations, averaged over 50 runs. The parameters for the vanilla dynamics are chosen based on standard values in the literature: stepsize $\mathrm{d}t \stackrel{\text{def}}{=} 0.1$; SBS kernel bandwidth $\frac{1}{N^2}$ (Serré et al., 2025); CBO parameters $\lambda \stackrel{\text{def}}{=} 1$, $\varepsilon \stackrel{\text{def}}{=} 10^{-2}$, $\beta \stackrel{\text{def}}{=} 1$, $\sigma \stackrel{\text{def}}{=} 5.1$ (Carrillo et al., 2021); LANGEVIN temperature $\beta \stackrel{\text{def}}{=} 1$; SMD intensity $\beta \stackrel{\text{def}}{=} 1$ and $\delta \stackrel{\text{def}}{=} 2.1$ (see Figure A.3.25 and Appendix A.1); GCN Gaussian kernel bandwidth $\sigma = 1$ (see Figure A.3.23). To assess statistical significance, we perform one or several Mann-Whitney U tests depending on which type of method achieves the best result: if a vanilla dynamic achieves the best result, we compare it against all its SMD and GCN variants; if a SMD or GCN variant achieves the best result, we compare it only against the vanilla dynamic. These tests assess wether atleast one of the SMD or GCN variants significantly outperforms the vanilla dynamic.

For each dynamic, Table A.3.8 reports the best performing method between vanilla, SMD and GCN variants. Best results are in **bold**. Mann-Whitney U tests at level 0.05 are highlighted in gray. We report average rank and empirical competitive ratio (ECR) Eq. 21. Table A.3.8 shows that noise injection consistently enhances baseline performance. SMD variants generally outperform GCN, with significant gains for CBO. MSGD, Langevin and SBS also benefit from noise injection, showing reliable improvements across multiple benchmarks. Figure A.3.22 shows SMD mean perturbation improves CBO and MSGD on Ackley ($d = 4$). Figure A.3.23 demonstrates GCN's robustness to kernel bandwidth σ on Levy ($d = 10$).

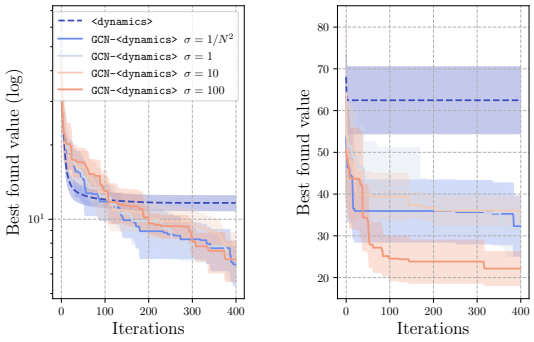
Empirical exploration. A key motivation for common noise plugins is to enhance space exploration and thereby improve convergence. Figure A.3.27 supports this by showing that ρ-noise leads to both better coverage of the search space and sharper concentration around the global minimizer, reflecting a more effective balance between exploration and exploitation.

Problem-dependent effects. More broadly, Table A.3.8 and Figures A.3.22 and A.3.23 reveal that noise injection effectiveness is intimately tied to both the baseline dynamic and problem characteristics. A critical factor is the particle-to-dimension ratio. When the number of particles is sufficiently large relative to the problem dimension, the baseline dynamics converge reliably, and the added noise provides no substantial benefit (see Figure A.3.25). In fact, it may even degrade performance by introducing unnecessary perturbations. Conversely, when particles are scarce relative to dimension, the additional exploration provided by noise injection yields modest but consistent improvements by helping particles escape local minima (see Figure A.3.25). These findings suggest that reproducing our results in higher



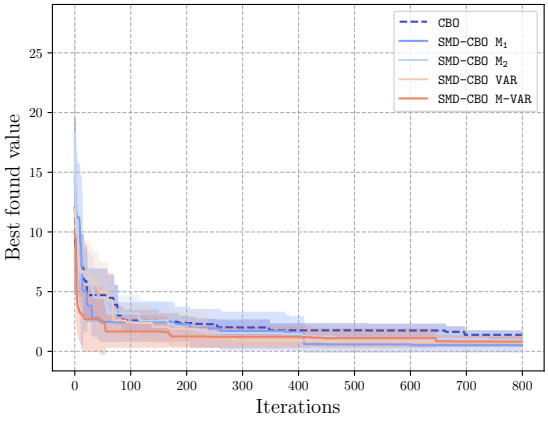
(a) CBO dynamics (b) MSGD dynamics

Figure A.3.22. *Impact of perturbed observables on CBO (left) and multi-start gradient descent dynamics (right) on Ackley function in dimension* 4 *with* 30 *particles. The mean (line) and standard deviation (shaded area) of the best found value are reported over* 5 *runs.*

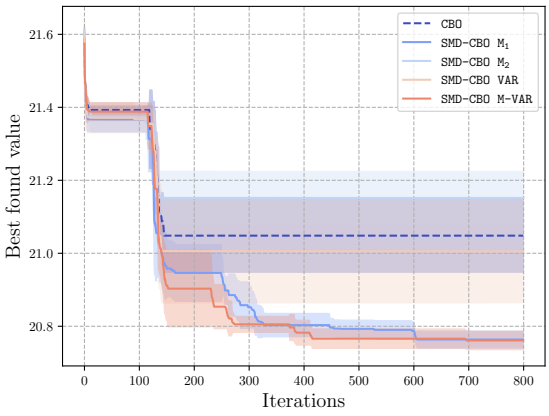


(a) SBS dynamics (b) CBO dynamics

Figure A.3.23. *Impact of* σ *on the RKHS noise for the SBS dynamics (left) and the CBO dynamics (right) on the Levy function in dimension* 10 *with* 10 *particles. The mean (line) and standard deviation (shaded area) of the best found value are reported over* 5 *runs.*



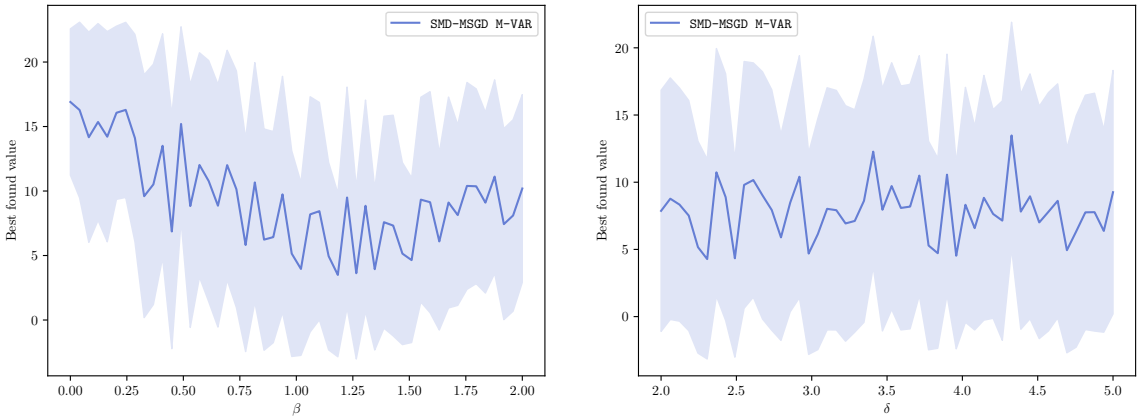
(a) Impact of SMD on CBO dynamics on Rastrigin function in dimension 2 with 10 particles.



(b) Impact of SMD on CBO dynamics on Ackley function in dimension 200 with 10 particles.

Figure A.3.24. *Illustration of the problem-dependent effects of noise injection methods in the multimodal setting. The mean (line) and standard deviation (shaded area) of the best found value are reported over* 5 *runs. In low dimension with sufficient particles (left), the effect of SMD is negligible, while in very high dimension with scarce particles (right), SMD yields modest improvements.*

dimensions requires careful tuning of particle counts to maintain the balance observed in our $d = 20$ experiments. Specifically, one should increase the number of particles proportionally to avoid regime shifts where noise becomes detrimental. Developing adaptive strategies for automatically selecting noise injection methods and hyperparameters based on the problem's intrinsic dimensionality remains an important direction for future work.



(a) The performance improves from $\gamma = 0$ (no noise) to $\gamma = 1.25$, but degrades for larger values, as the noise becomes too large and disrupts the optimization process.

(b) The performance is relatively robust to the choice of δ.

Figure A.3.25. *Study of the impact of the hyperparameters γ and δ of SMD on the performance of MSGD dynamics on the Ackley function in dimension* 4 *with* 50 *particles and* 400 *iterations. The mean (line) and standard deviation (shaded area) of the best found value are reported over* 20 *runs.*

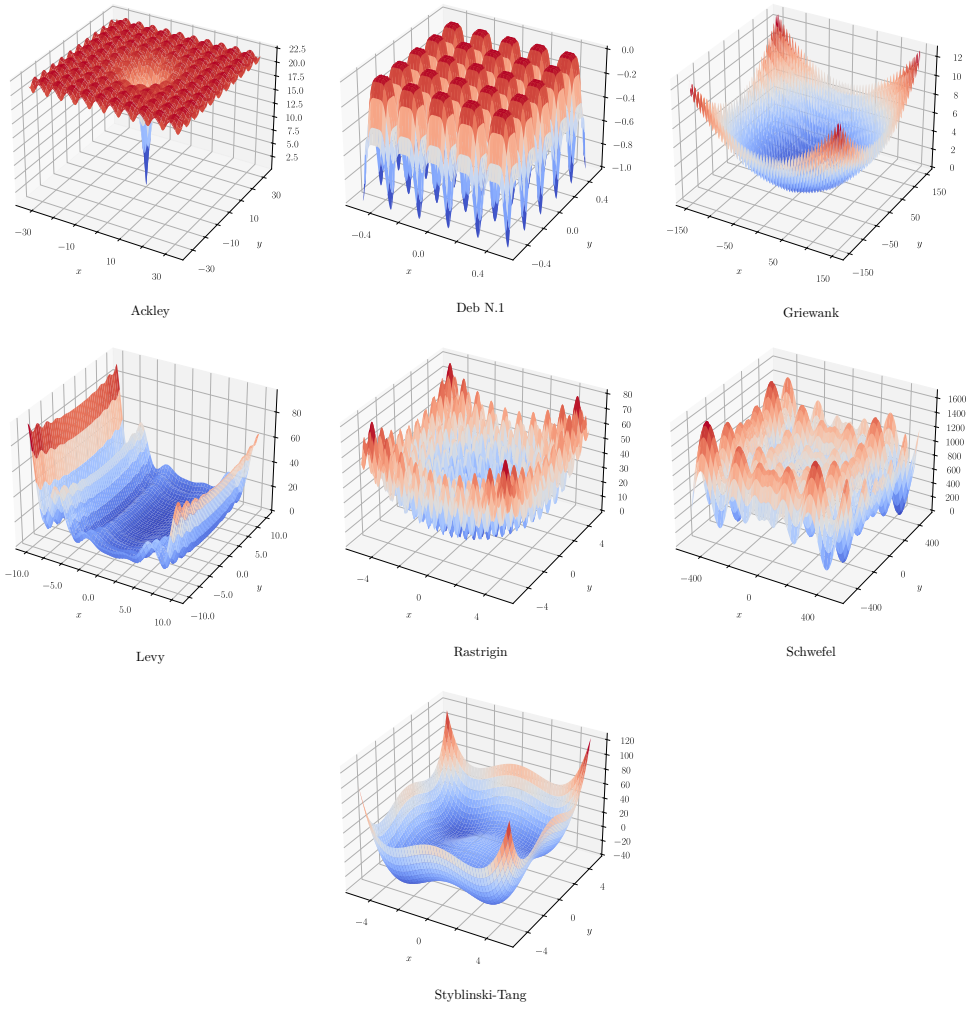


Figure A.3.26. *Visualization of the 2D landscapes of the multimodal functions used in our experiments.*

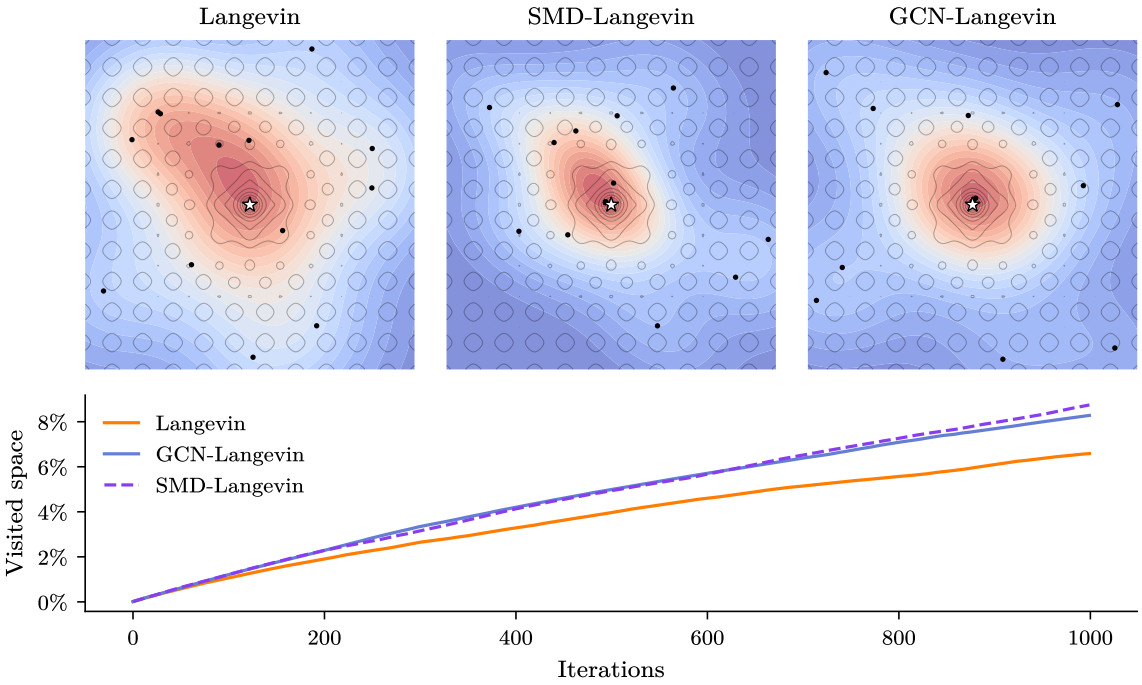


Figure A.3.27. *Coverage of the search space for vanilla Langevin and its GCN and SMD Mean+Var variants on the two-dimensional Ackley function, using* 15 *particles over* 1000 *iterations, from the same initial particles.* ***Top:*** *estimated density of the particles (via Gaussian kernels) at the final iteration, from blue (low) to red (high), over the level lines of the function. The particles are the black dots and the global minimum is the star.* ***Bottom:*** *percentage of the search space covered by the particles over iterations. Coverage is measured by discretizing the search space into an ε-cover and counting the number of cells of side 2ε visited by at least one particle, with $\varepsilon = 0.1$. The GCN and SMD variants achieve both sharper concentration around the global minimum and broader exploration of the search space, suggesting a better ability to escape local minima and ultimately leading to improved optimization performance.*

Table A.3.8. *Comparison between vanilla dynamics and their SMD and GCN variants on* 7 *multimodal benchmark functions in dimension* 20 *using* 150 *particles over* 300 *iterations. For each dynamic, the average best result is reported. The average rank over all benchmarks along with the ECR is reported at the bottom of each table. The maximum p-value of Mann-Whitney U tests is reported in the last column. The best results are in bold and cells where the Mann-Whitney U test is significant at level* 0.05 *are highlighted in gray.*

CBO dynamics

BENCHMARK	CBO	SMD-CBO MEAN	SMD-CBO M^2	SMD-CBO VAR	SMD-CBO MEAN+VAR	GCN-CBO	P-VALUE
ACKLEY	21.097	20.615	21.083	21.088	20.610	**20.591**	< 0.05
DEB N.1	**−1.000**	**−1.000**	**−1.000**	**−1.000**	**−1.000**	**−1.000**	< 0.05
GRIEWANK	20.279	19.562	19.657	19.683	20.073	**19.506**	0.237
LEVY	102.351	86.857	95.816	95.945	**81.912**	85.146	< 0.05
RASTRIGIN	253.358	238.943	250.021	250.463	240.406	**238.768**	< 0.05
SCHWEFEL	5157.189	5127.111	5139.564	5146.530	5199.557	**5111.855**	0.387
STYBLINSKI-TANG	−19.513	**−20.798**	−20.006	−19.881	−20.091	−20.191	< 0.05
AVG RANK	5.14	2.00	3.29	4.14	3.00	**1.29**	
ECR	1.0648	**1.0097**	1.0424	1.0443	1.0132	1.0104	

SBS dynamics

BENCHMARK	SBS	SMD-SBS MEAN	SMD-SBS M^2	SMD-SBS VAR	SMD-SBS MEAN+VAR	GCN-SBS	P-VALUE
ACKLEY	21.126	20.851	21.112	21.102	20.825	**20.810**	< 0.05
DEB N.1	−0.961	−0.835	−1.000	−1.000	**−1.000**	−0.909	< 0.05
GRIEWANK	19.959	19.686	20.646	20.218	19.953	**19.128**	0.173
LEVY	**24.036**	47.680	25.827	25.815	46.601	45.794	0.005
RASTRIGIN	86.442	185.464	83.496	**80.785**	190.917	181.370	0.006
SCHWEFEL	5986.360	5984.992	5989.562	6062.542	**5947.981**	5985.055	0.365
STYBLINSKI-TANG	**−25.668**	−24.339	−23.746	−24.191	−24.578	−25.150	0.008
AVG RANK	3.29	4.00	4.29	3.57	3.00	**2.86**	
ECR	15.1622	15.4879	15.1930	15.1820	**1.3467**	15.4564	

LANGEVIN dynamics

BENCHMARK	LANGEVIN	SMD-LANGEVIN MEAN	SMD-LANGEVIN M^2	SMD-LANGEVIN VAR	SMD-LANGEVIN MEAN+VAR	GCN-LANGEVIN	P-VALUE
ACKLEY	**20.723**	20.767	20.732	20.733	20.757	20.760	1.000
DEB N.1	−0.921	−0.956	−1.000	**−1.000**	**−1.000**	−0.958	< 0.05
GRIEWANK	19.362	19.117	18.711	19.193	18.992	**18.516**	0.097
LEVY	1.329	1.847	1.173	**1.073**	1.904	1.796	< 0.05
RASTRIGIN	184.367	189.482	**183.202**	184.258	193.141	187.751	0.677
SCHWEFEL	2405.830	2436.136	2383.227	**2324.940**	2381.518	2404.611	0.316
STYBLINSKI-TANG	−34.902	−33.186	**−35.050**	−34.914	−32.868	−33.206	0.278
AVG RANK	3.86	5.14	**2.00**	2.14	4.00	3.71	
ECR	15.1944	15.3272	15.1613	**1.0108**	1.2014	15.3117	

MSGD dynamics

BENCHMARK	MSGD	SMD-MSGD MEAN	SMD-MSGD M^2	SMD-MSGD VAR	SMD-MSGD MEAN+VAR	GCN-MSGD	P-VALUE
ACKLEY	**19.977**	20.725	19.985	19.982	20.736	20.736	0.497
DEB N.1	−0.567	−0.834	**−1.000**	**−1.000**	**−1.000**	−0.920	< 0.05
GRIEWANK	19.467	**18.829**	19.557	20.007	19.254	19.358	0.243
LEVY	9.413	1.905	8.873	9.330	1.888	**1.354**	< 0.05
RASTRIGIN	187.689	187.206	193.505	186.581	**183.479**	183.768	0.278
SCHWEFEL	2396.030	2424.021	**2361.875**	2386.382	2380.197	2447.570	0.488
STYBLINSKI-TANG	−36.325	−34.665	**−36.379**	−36.353	−34.457	−34.867	0.042
AVG RANK	4.14	3.86	3.00	3.14	**2.86**	3.57	
ECR	16.0058	15.3008	1.8064	1.8554	**1.1646**	15.2352	

4 —

GLobal Optimization Benchmark (GLOBE)

Abstract. Open-source libraries are have a catalytic role in research pipelines, where new methods must be compared against up-to-date baselines. We present the GLobal Optimization Benchmark (GLOBE) modular Python library (Serré et al., 2026a) that unifies classical and recent continuous global optimization algorithms, including decision-based and mathematically founded particle-based methods, in a single framework. A central contribution of GLOBE is its modular architecture, reusing the family-level structures presented in previous sections to enables plugins to be implemented once and makes them available to all algorithms in the corresponding family. This modular design leverages recent advances in mathematical formalization of global optimization, where structural commonalities across algorithms have been identified and used to develop broadly applicable, formally grounded algorithmic features (see Chapter B.1). The C++ backend relies on Eigen 5 for efficient linear algebra. GLOBE presently includes 14 optimizers, 19 analytical benchmarks along with a random function generator, and an integrated toolkit for direct algorithm comparison.

In this section, We present the GLobal Optimization Benchmark (GLOBE) modular Python library (Serré et al., 2026a) that unifies classical and recent continuous global optimization algorithms, including model-based approaches such as BAYESOPT and CMA-ES, decision-based approaches such as ADALIPO+, and McKean-Vlasov-based approaches such as CBO and SBS. This library provides a common interface for these algorithms, allowing the use of modular plugins such as trust-region modules (see Section A.2.4) and common noise injection (see Section A.3.4).

Existing global optimization libraries and limitations. The landscape of global optimization software is diverse, with many libraries addressing specific problems or focusing on algorithm families. We review the most prominent existing solutions:

- *NLopt (Johnson, 2007)*: Provides a unified interface to classical algorithms (DIRECT, CRS, evolutionary strategies) but lacks structural pattern exploitation and recent advances.
- *Scipy.optimize (Virtanen et al., 2020)*: Broad optimization library where global optimization is secondary to local methods.
- *Bayesian Optimization (Martinez-Cantin, 2014; Nogueira, 2014)*: Specialized for expensive black-box functions, not general-purpose.
- *Optuna (Akiba et al., 2019) and Ray Tune (Liaw et al., 2018)*: Both specialized for hyperparameter tuning in machine learning rather than general continuous global optimization.
- *Nevergrad (Rapin & Teytaud, 2018)*: Well-suited for benchmarking and practical applications but does not explicitly expose structural components.
- *PyGMO (Biscani & Izzo, 2020)*: A comprehensive platform with a wide range of algorithms and parallel optimization features, but focuses on algorithm orchestration rather than modular composition.

While existing libraries serve their respective domains well, each specializes in particular optimization scenarios, and therefore leave a gap in the available open-source solutions:

- **Specialization vs. generality:** Most libraries focus on a specific problem type (e.g. hyperparameter tuning, expensive black-box functions) or a specific algorithm family (e.g. evolutionary algorithms, Bayesian optimization). A unified framework for general continuous global optimization is lacking.
- **Limited code modularity:** Algorithms are typically implemented independently without exploiting common patterns, such as sampling strategies, population evolution, decision making loops. This leads to code duplication and makes it difficult to add new algorithms or features that span multiple families.
- **Missing recent advances:** Many libraries focus on classical algorithms. Recent advances in stochastic differential equation (SDE)-driven methods, particle-based algorithms with common noise (see Chapter A.3), and modern decision-based approaches (see Chapter A.2) are rarely integrated. For research purposes, comparing newly proposed methods against these recent advances is essential to validate their effectiveness, yet existing libraries do not provide easy access to such state-of-the-art methods.

Contribution of the GLOBe library. GLOBE directly addresses these challenges through a modular, general-purpose framework for continuous global optimization that: **i.** consolidates classical and state-of-the-art algorithms, enabling researchers to benchmark new methods against up-to-date baselines; **ii.** provides integrated benchmarking tools for direct and reproducible algorithm comparison; and **iii.** factors common patterns at the family level to facilitate the creation of new algorithms and plugins. This modular architecture is grounded in recent advances in the mathematical formalization of global optimization[8] (Powell, 2019; Serré et

[8] See Chapter B.1

Table A.4.9. *Optimization algorithms implemented in GLOBE*

Algorithm	Family	Reference
ADALIPO+	Decision-based	Serré et al. (2024)
ECP	Decision-based	Fourati et al. (2025)
ADARANKOPT	Decision-based	Malherbe et al. (2016)
PSO	Particle-based	Kennedy & Eberhart (1995)
CBO	Particle-based	Pinnau et al. (2017)
LANGEVIN	Particle-based	Prigogine & Balescu (1957)
SBS	Particle-based	Serré et al. (2025)
MSGD	Particle-based	—
DIRECT	Misc.	Jones (2001)
CRS	Misc.	Price (1983)
CMA-ES	Misc.	Hansen & Ostermeier (2001)
BAYESOPT	Misc.	Nogueira (2014)
MLSL	Misc.	Rinnooy Kan et al. (1985)
GRADIENT DESCENT	Misc.	—
PURE RANDOM SEARCH	Misc.	—

al., 2026b; Serré et al., 2026c), which revealed structural commonalities across algorithms and enabled the design of broadly applicable, formally justified features. By reflecting these mathematical structures, GLOBE achieves both extensibility and research-friendliness. The C++ backend relies on the Eigen library to provide efficient vectorized linear algebra operations, ensuring high performance across all algorithms. GLOBE currently includes 15 optimizers, 19 analytical benchmarks with a random function generator, and an integrated toolkit for algorithm comparison.

The documentation of GLOBE is available online[9], the code is open-source under the GPL-3.0 license[10] and the Python package can be installed via `pip`[11]. We welcome contributions from the community to expand the library with new algorithms, benchmarks, and features.

4.1 The GLOBe library

GLOBE currently implements 15 global optimization algorithms, as summarized in Table A.4.9.

[9] gaetanserre.fr/GLOBe

[10] github.com/gaetanserre/GLOBe

[11] pypi.org/project/globe-opti

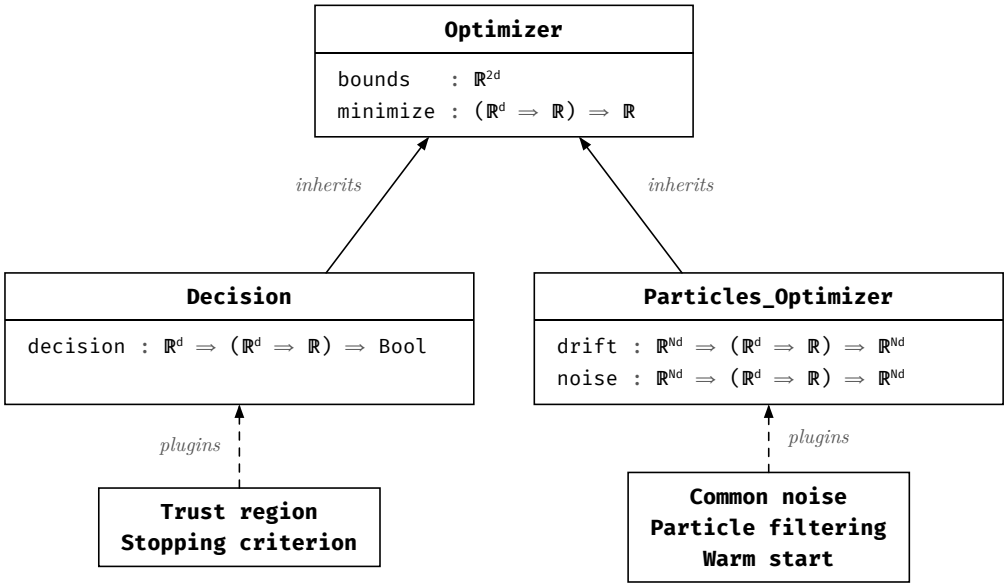


Figure A.4.28. *GLOBE's architecture showing the class hierarchy and modular decomposition. The base* `Optimizer` *interface is extended by one base class per family:* `Decision` *(for decision-based methods) and* `Particles_Optimizer` *(for particle-based methods). Plugins are attached to these base classes, so that a plugin written once is valid for all the algorithms of its family: the trust region (see Section A.2.4) and the stopping criterion of LIPO+ (see Section A.2.2.1) for decision-based methods, the common noise (see Section A.3.4), particle filtering and warm start (see Section A.3.2.4) for particle-based methods.*

4.1.1 Modular architecture

The core design philosophy of GLOBE is to achieve *composition through inheritance and modularity*. Rather than treating all algorithms as independent implementations, GLOBE identifies common patterns across algorithm families and factors them into reusable components. This family-level factorization is especially important for plugins: modules are attached to family base classes, so a single plugin implementation is valid for all algorithms in that family. This approach is illustrated in Figure A.4.28.

This modular architecture goes beyond engineering convenience, as it is deeply rooted in recent advances in the mathematical formalization of global optimization algorithms. Works such as (Powell, 2019; Serré et al., 2026b; Serré et al., 2026c) have revealed fundamental structural patterns common to multiple algorithm families, formally characterizing the mathematical invariants and common building blocks that transcend individual algorithms. Notably, (Serré et al., 2026b) develops frameworks enabling the application of formal methods to global optimization, providing a principled foundation for algorithm analysis and composition. By codifying these formal structures, GLOBE's modular design reflects the mathematical reality of global optimization, enabling a principled approach to algorithm development and composition. The adopted perspective aligns with the broader trend toward formal methods in machine learning and optimization research, where structured mathematical frameworks provide both theoretical grounding and practical extensibility.

```cpp
class Optimizer {
public:
  Optimizer(vec_bounds bounds, string name);
  ~Optimizer();

  virtual result_eigen minimize(
    function<double(dyn_vector)> f
  ) = 0;

  void set_stop_criterion(double criterion);

protected:
  vec_bounds bounds;
  string name;
  mt19937_64 re; // Random engine
  bool has_stop_criterion;
  double stop_criterion;
}
```

Listing A.4.1. *Abstract base* `Optimizer` *class interface showing the minimal API that all algorithms must implement.*

4.1.2 Base optimizer interface

All optimization algorithms in GLOBE inherit from a common abstract base class that defines the minimal interface required, as shown in Listing A.4.1. This interface ensures that all algorithms can be used interchangeably. The base class handles:

- **Bounds management**: The search space definition.
- **Random number generation**: Seeded random number generation engine for reproducibility.
- **Stopping criteria**: Pluggable stopping conditions (target values).
- **Python interoperability**: Provides the C++/Python interface layer. As a result, this interoperability burden is already handled when implementing new algorithms in C++.

4.1.3 Decision-based optimizers

Decision-based algorithms make progress by maintaining a history of function evaluations, and according to this history they decide which region of the search space to sample from next (see Chapter A.2). Examples of this class include ADALIPO+ (Lipschitz-based adaptive partitioning) (Serré et al., 2024), ECP (Every Call is Precious) (Fourati et al., 2025; Fourati et al., 2026), and ADARANKOPT (ranking-based decisions) (Malherbe et al., 2016).

The common pattern across these algorithms is:

- *Initialize*: Sample an initial point $x_0 \in \Omega$ uniformly at random.
- *Iterate*: At iteration t, using the history $\{(x_i, f(x_i))\}_{i=0}^{t-1}$, compute a region $R_t \subseteq \Omega$.
- *Sample*: Draw a new candidate point x_t at random over R_t.
- *Evaluate*: Compute $f(x_t)$ and append it to history.

This pattern is defined by the `Decision` base class that inherits from `Optimizer`. The `Decision` class handles the evaluation loop and history management, while specific algorithms (`ADALIPO+`, `ECP`, etc.) only need to implement their decision-making logic via callback functions. Based on Algorithm A.2.1, this factorization avoids duplicating the iteration framework and sampling process across multiple decision-based methods, and allows family-level plugins (e.g. trust-region refinement) to be implemented once and reused by all decision-based algorithms.

4.1.4 Particle-based optimizers

Particle-based methods maintain a population of $N+1$ particles that evolve at each iteration according to stochastic dynamics. Many such algorithms can be modeled as discretizations of McKean-Vlasov stochastic differential equations

$$\mathrm{d}X_t^{(i)} = b\Big(X_t^{(i)}, \hat{\rho}_{X_t}\Big)\,\mathrm{d}t + \sigma\Big(X_t^{(i)}, \hat{\rho}_{X_t}\Big)\,\mathrm{d}B_t^{(i)},$$

where b is the drift (deterministic evolution), σ is the diffusion (stochastic noise), and $\hat{\rho}_{X_t}$ is the empirical law of $X_t^{(i)}$. See Chapter A.3 for details. Examples of particle-based algorithms in `GLOBE` Consensus-Based Optimization (`CBO`) (Pinnau et al., 2017), Langevin dynamics (Prigogine & Balescu, 1957), Stein Boltzmann Sampling (`SBS`) (Serré et al., 2025), and Multi-Start Gradient Descent (MSGD).

The common pattern is:

- *Initialize*: Draw $N+1$ particles $X_0^{(0)}, ..., X_0^{(N)}$ uniformly from Ω.
- *Iterate*: At each iteration, compute drift and noise terms, then update all particles.
- *Discretize*: Use an Euler-Maruyama scheme to discretize the SDE and update particles.

This pattern is defined by the `Particles_Optimizer` base class that inherits from `Optimizer`. Concrete algorithms (`CBO`, `SBS`, …) only need to implement the drift and noise computation for their specific dynamics. The base class is based on Algorithm A.3.3 and handles the Euler-Maruyama discretization loop, particle updates, boundary handling, and adaptive scheduling; it also provides a natural integration point for family-level modules (e.g. common noise (see Section A.3.4)), so a plugin implementation can be applied consistently across all particle-based algorithms.

4.1.5 Modular plugins

Beyond the base hierarchies, `GLOBE` provides orthogonal modules that enhance algorithms.

Trust region module. Decision-based algorithms can be augmented with local refinement using trust-region methods. Specifically, when a promising region is identified, the algorithm can temporarily switch to a local optimizer (e.g. `BOBYQA` (Powell, 2009)) to refine the solution before resuming global exploration. See Section A.2.4 for details. This is implemented as an optional family-level module on top of the `Decision` base class, making the plugin valid for all decision-based optimizers without modifying their core logic. See Section A.2.4 for details.

Common noise module. Particle-based methods based on McKean-Vlasov dynamics face a fundamental challenge: when particles evolve under independent noise alone, exploration efficiency suffers due to insufficient diversity in particle trajectories (Tugaut, 2014). The recent work in Serré et al. (2026c), addresses this by introducing common noise schemes that are shared across all particles, enhancing global exploration:

- *SMD (Stochastic Moment Dynamics):* Injects noise into aggregate particle statistics, specifically the population mean and variance. This approach perturbs the collective evolution, improving coverage of the search space.
- *GCN (Geometrical Common Noise):* Constructs a canonical Gaussian common noise using the geometric structure of the underlying process.

See Section A.3.4 for details. GLOBE implements these noise schemes as modular components integrated at the `Particles_Optimizer` family level, so the same plugin is valid for any particle-based optimizer. This allows users to enhance exploration capabilities without modifying algorithm-specific core dynamics.

Particle filtering. GLOBE provides a flexible filtering interface at the family level, enabling algorithm-specific strategies for all particle-based optimizers. Currently, it implements quantile-based filtering (Serré et al., 2025) that removes particles with small displacement and poor function values, improving exploration efficiency. See Section A.3.2.4 for details. New strategies can be added without modifying core algorithm dynamics such as Sequential Importance Resampling (Kitagawa, 1993).

4.2 Benchmarking features

GLOBE provides an integrated benchmarking suite combining diverse synthetic test functions, a random function generator, and comprehensive analysis tools, enabling systematic and reproducible algorithm comparison.

4.2.1 Benchmarking functions

GLOBE includes a wide range of synthetic benchmark functions, as well as a random generator of functions, which can be used to evaluate the performance of optimization algorithms. It also implements a toolkit to perform large-scale experiments and analyze results. All benchmarks can run in any arbitrary number of dimensions.

Benchmark interface. All benchmark functions inherit from a common `Benchmark` class that provides a standardized interface. Each benchmark encapsulates the function expression, tracks evaluation counts, provides gradient estimates via finite differences, and supports visualization. This unified API enables consistent comparison across all benchmark types.

Analytical benchmark functions. GLOBE includes a comprehensive suite of 19 analytical benchmark functions organized by modality, see Table A.4.10. Most functions include known global minima, allowing precise assessment of algorithm performance.

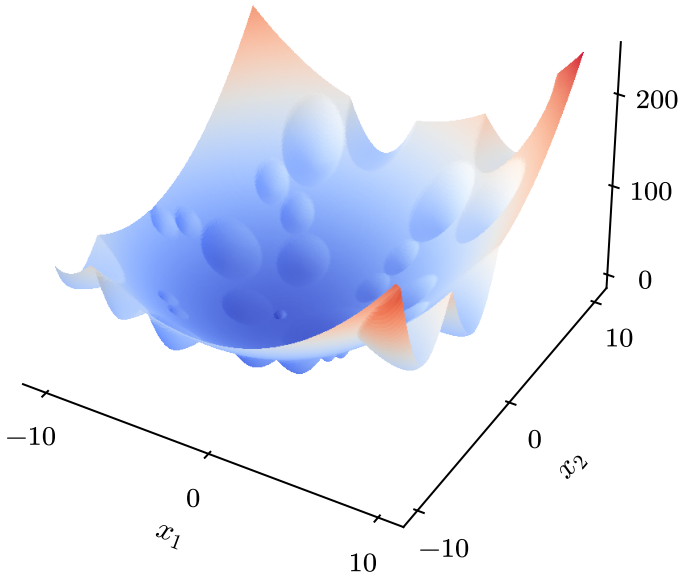


Figure A.4.29. *Example of a random 2D function generated by pyGKLS. The number of local minima and the value of the global minimum are controllable via parameters, allowing for systematic benchmarking across varying levels of difficulty.*

Table A.4.10. *Analytical benchmark functions in GLOBE*

Category	Functions	Characteristics
Unimodal	Sphere, Ellipsoid, Bentcigar, Hyperellipsoid, Zakharov, Trid, Sumpow, Rosenbrock, Dixonprice	Single optimum, smooth landscapes, test convergence speed
Multimodal	Rastrigin, Ackley, Schwefel, Levy, Michalewicz, Langermann, Deb, Griewank, Styblinski-Tang	Numerous local minima, test global exploration

Random function generator. GLOBE integrates pyGKLS[12], a Python wrapper around the GKLS function generator (Gaviano et al., 2003). GKLS generates random smooth multidimensional functions with:

- known global and local minima locations;
- controllable multi-modality degree;
- controllable smoothness (from C^2 to C^0);
- reproducibility via fixed seeds.

Figure A.4.29 gives an example of a function generated by pyGKLS. This generator provides a rigorous alternative to fixed analytical benchmarks, enabling statistical analysis across random function instances.

[12] github.com/gaetanserre/pyGKLS

4.2.2 Benchmarking engine

globe provides an integrated benchmarking framework via its main `GLOBE` class, which orchestrates comprehensive experimental comparisons. The class takes as input three main components: a list of optimizers (with optional hyperparameter configurations), a list of benchmark functions, and a list of metrics. This modular design enables systematic analysis across multiple algorithms, problems, and performance criteria.

The typical workflow is as follows:

i. *Configuration*: Users specify optimizer instances (or names with hyperparameters), benchmark functions, and evaluation metrics.

ii. *Multiple runs*: The framework runs each optimizer-benchmark combination n times to account for stochasticity, and collects the solution values from each run.

iii. *Metric computation*: For each optimizer-benchmark pair, the framework computes all requested metrics across the collected solutions. This enables richer analysis than simple optimality gaps.

iv. *Aggregation and reporting*: Results are aggregated and displayed as comparative tables (console or LaTeX format), allowing side-by-side algorithm comparison.

Extensible metrics. GLOBE provides a flexible framework where each metric is a callable that processes the solutions from multiple runs. Built-in metrics include:

- *Approximation quality*: Mean and standard deviation of solution values across runs, providing robustness assessment.
- *f-target value*: A problem-adaptive threshold computed as

$$f_{\text{target}} = f_{\min} + (f_{\text{mean}} - f_{\min})(1 - p),$$

  where $f_{\min}$ is the global minimum, f_{mean} is the mean value over a large random sample from Ω, and p is a configurable proportion (default $p = 0.99$). This represents the quantile of the random distribution of function values. This corresponds to Eq. 13, the stopping condition presented in Section A.2.2.3.
- *Success rate (proportion)*: Percentage of runs where the algorithm achieves a solution value below the f-target threshold, effectively measuring how often the optimizer succeeds in reaching a specified percentile performance level.

The metrics interface is designed to be extensible: users can implement custom metrics by inheriting from the base `Metric` class and by defining their evaluation logic. This enables domain-specific performance analysis (e.g. computational budget constraints, constraint satisfaction for engineering problems).

Practical workflow. As illustrated in Listing A.4.2, the typical workflow with GLOBE is simple and intuitive: users specify the algorithms to compare (with their optional hyperparameters), the benchmark functions, and the metrics of interest. The framework automatically runs multiple iterations, collects results, and aggregates statistics to facilitate comparative analysis. This modular approach enables researchers to quickly benchmark new algorithms against the state-of-the-art without requiring tedious manual management of experiments.

```python
from globe import GLOBe
from globe.benchmarks import PyGKLS, create_bounds

pygkls = PyGKLS(2, 15, [-100, 100], -100,
  smoothness="ND")
globe = GLOBe(
    ["CBO", ("SBS", {"iter": 10}), "SBS", "AdaLIPO+TR",
      "CMA-ES", "PSO"],
    ["Square", pygkls],
    ["Proportion"],
    bounds=create_bounds(2, -99, 99, 2),
)
globe.run(n_runs=10, verbose=1, latex_table=True)
```

Listing A.4.2. *Example of a benchmarking experiment using GLOBE. The code configures a benchmark with two-dimensional functions: the* `Square` *function and an instance of pyGKLS generated function, and compares the performance of six optimizers (CBO, SBS with* 10 *iterations, SBS with default parameters, ADALIPO with trust region, CMA-ES, and PSO) using the* Proportion *metric. The results are collected over* 10 *runs for each optimizer-benchmark pair and a LaTeX table is generated containing the comparative results.*

Appendix

A —

Theory of common noise plug-ins

Abstract. This section contains theoretical development and proofs associated with the common noise modules presented in Section A.3.4.

A.1 About the explosion of SMD

For a given macroscopic observable F, poorly chosen coefficients a and s may cause the dynamics Eq. 33 and Eq. 32 to blow up in finite time. This issue arises because many macroscopic observables (e.g., variances) take values in a restricted subset of $\mathbb{R}^p$. As long as the stochastic perturbation of $F\left(\hat{\rho}_{Y_t}\right)$ remains within its admissible domain, the dynamics are well defined; however, if positivity constraints are violated, the coefficients become singular and the evolution breaks down.

To enforce positivity when perturbing second-order moments or variances, the coefficients a and s are chosen as d independent δ-Bessel processes (Revuz & Yor, 2013, Chapter 11), which preserve positivity for $\delta \geq 2$. These correspond to the SDE Eq. 34 with $(a, s) = \left(a_+^\delta, s_+^\delta\right)$, understood component-wise. In contrast, since the mean is unconstrained, its perturbation is taken to be standard Brownian motion ($a = 0$, $s = I_d$).

The coefficients defined in Eq. 36, Eq. 37, and Eq. 38, are not globally Lipschitz in ρ, reflecting potential singular behavior as second-order moments or variances approach zero. While this could in principle lead to finite-time explosion, Germain & Monmarché (2025, Section 6) shows that the stochastic perturbation alone is non-explosive. It is therefore conjectured that the full dynamics Eq. 32, combining this perturbation with a broad class of mean-field optimization algorithms, is also non-

explosive under mild conditions. Since no explosion was observed in our numerical experiments, we do not consider regularized dynamics in the remainder of this thesis.

A.2 Random Generalized Functions

Many stochastic models arising in machine learning, physics, and spatial statistics involve random objects that are too irregular to be defined pointwise. A convenient and rigorous framework to handle such objects is that of *random generalized functions* (also called distribution-valued random variables).

Let $\mathcal{D}$ denote a space of test functions (e.g., $C_c^\infty(\mathbb{R}^d)$ or the Schwartz space $\mathcal{S}$). A generalized function is a continuous linear functional $T : \mathcal{D} \to \mathbb{R}$. A *random generalized function* X is a mapping such that, for every $\varphi \in \mathcal{D}$, the quantity $X(\varphi)$ is a real-valued random variable, and the map $\varphi \mapsto X(\varphi)$ is linear and and stable under limits: $\varphi_n \to \varphi$ in $\mathcal{D}$ implies $X(\varphi_n) \xrightarrow{\mathbb{P}} X(\varphi)$.

This formulation avoids pointwise evaluation and instead characterizes randomness through its action on test functions. Classical stochastic processes that fail to exist as functions—most notably Gaussian white noise—are naturally defined in this sense. For example, white noise W is characterized by

$$\mathbb{E}[W(\varphi)W(\psi)] = \langle \varphi, \psi \rangle_{L^2},$$

and therefore defines a centered Gaussian random generalized function.

Random generalized functions play a central role in stochastic partial differential equations, Gaussian processes with singular covariance kernels, and probabilistic models involving infinite-dimensional randomness, see Da Prato & Zabczyk (2014) to go further.

A.3 Proofs

A.3.1 Derivation of the coefficients $\tilde{b}$ and $\tilde{\sigma}$

The derivation of the coefficients $\tilde{b}$ and $\tilde{\sigma}$ follows from Germain & Monmarché (2025, Eq. 8). This formula, stated under the assumption that the observable can be expressed as a moment of the measure

$$F(\mu) = \mu(f) \stackrel{\text{def}}{=} \int_{\mathbb{R}^d} f(x)\, \mathrm{d}\mu(x) \tag{48}$$

with $f : \mathbb{R}^d \to \mathbb{R}^p$, gives the expression of $\tilde{b}$ and $\tilde{\sigma}$ in terms of f, and of the coefficients a and s

$$\begin{cases} \tilde{\sigma}_\mu(x) = \nabla f(x)(\mu(\nabla f^T \nabla f))^{-1} s(\mu(f)) \\ \tilde{b}_\mu(x) = \nabla f(x)(\mu(\nabla f^T \nabla f))^{-1} \left[a(\mu(f)) - \frac{1}{2}\mu(\tilde{\sigma}_\mu \tilde{\sigma}_\mu^T : \nabla^2 f)\right] \end{cases} \tag{49}$$

where $\sigma_\mu \tilde{\sigma}_\mu^T : \nabla^2 f \overset{\text{def}}{=} \left(\sigma_\mu \tilde{\sigma}_\mu^T : \nabla^2 f_1, ..., \sigma_\mu \tilde{\sigma}_\mu^T : \nabla^2 f_p\right)$ is a vector of $\mathbb{R}^p$, and $A : B$ represents the usual Hilbert-Schmidt scalar product between two matrices of $\mathbb{R}^{d\times d}$.

Because F_{mean} and $F_{s.m}$ directly fit this setting with

$$f_{\text{mean}} : x \to x, \;\; f_{s.m} : x \mapsto (x_1^2, .., x_d^2)$$

and respective coefficients $a_{\text{mean}} = 0, s_{\text{mean}} = I_d$ and $a_{s.o}(z) = \left[\left(\frac{\delta - 1/2}{z_i}\right)\right]_{i\in[d]}, s_{s.o} = I_d$

Equations Eq. 35 and Eq. 36 giving the expression of $\tilde{b}$ and $\tilde{\sigma}$ for F_{mean} and $F_{s.m}$ follow from a direct explicit computation of Eq. 49.

Observables F_{var} and $F_{\text{mean+var}}$ do not directly fit this setting so one cannot directly compute equation $\tilde{b}$ and $\tilde{\sigma}$ with Eq. 49. To do so, first assume that the particle system defined by Eq. 34 has its observable $F_{\text{mean+var}}\left(\hat{\rho}_{Y_t}\right)$ satisfying the following SDE

$$\begin{cases} \mathrm{d}F_{\text{mean}}\left(\hat{\rho}_{Y_t}\right) = \varepsilon \, \mathrm{d}B_t^1 \\ \mathrm{d}F_{\text{var}}\left(\hat{\rho}_{Y_t}\right) \;\; = \frac{\delta - 1}{F_{\text{var}}\left(\hat{\rho}_{Y_t}\right)} \mathrm{d}t + \mathrm{d}B_t^2 \end{cases} \tag{50}$$

where B^1 and B^2 are two independent d dimensional Brownian motions and $\varepsilon \in \{0, 1\}$ corresponds respectively to the control of the Variance alone and of the mean and the variance. Again, the previous equation should be understood component by component since $F_{\text{var}}\left(\hat{\rho}_{Y_t}\right) \in \mathbb{R}^d$. Using the fact that $F_{\text{var}}\left(\hat{\rho}_{Y_t}\right) = F_{s.m}\left(\hat{\rho}_{Y_t}\right) - F_{\text{mean}}\left(\hat{\rho}_{Y_t}\right)^2$, and that because of Itô's formula

$$\mathrm{d}F_{\text{mean}}\left(\hat{\rho}_{Y_t}\right)^2 = 2\varepsilon F_{\text{mean}}\left(\hat{\rho}_{Y_t}\right) \mathrm{d}B_t^1 + \varepsilon^2 \, \mathrm{d}t$$

Eq. 50 rewrites as

$$\begin{cases} \mathrm{d}F_{\text{mean}}\hat{\rho}_{Y_t} \;\; = \varepsilon \, \mathrm{d}B_t^1 \\ \mathrm{d}F_{s.m}\left(\hat{\rho}_{Y_t}\right) = 2\varepsilon F_{\text{mean}}\left(\hat{\rho}_{Y_t}\right) \mathrm{d}B_t^1 + \varepsilon^2 \, \mathrm{d}t + \frac{\delta - 1}{F_{s.m}\left(\hat{\rho}_{Y_t}\right) - F_{\text{mean}}\left(\hat{\rho}_{Y_t}\right)^2} \mathrm{d}t + \mathrm{d}B_t^2 \end{cases}$$

which can be written as

$$\mathrm{d}\hat{\rho}_{Y_t}(f) = a\left(\hat{\rho}_{Y_t}(f)\right) \mathrm{d}t + s\left(\hat{\rho}_{Y_t}(f)\right) \mathrm{d}B_t$$

with $f = (f_{\text{mean}}, f_{s.m}) : \mathbb{R}^d \to \mathbb{R}^{2d}$, $B = (B^1, B^2)$ is a $2d$ dimensional Brownian process, and for $(z_1, z_2) \in \mathbb{R}^{2d}$

$$a(z_1, z_2) = \begin{pmatrix} 0 \\ \frac{\delta + 1}{2(z_2 - z_1^2)} + \varepsilon^2 \end{pmatrix}, \;\; s(z_1, z_2) = \begin{pmatrix} \varepsilon & 0 \\ 2\varepsilon z & 1 \end{pmatrix}$$

This formulation now fits the setting of Eq. 49, which can be computed explicitly. this finally gives

$$\tilde{b}(x,\rho) = \left(\delta - \frac{3}{2}\right)\left[\frac{x - m(\rho)}{4\mathrm{Var}(\rho)^2}\right],\ \tilde{\sigma}(x,\rho) = \left(\varepsilon I_d, \mathrm{Diag}\left[\frac{x - m(\rho)}{2\mathrm{Var}(\rho)}\right]\right)$$

Taking $\varepsilon = 1$ gives the desired result for $F_{\text{mean+var}}$. Taking $\varepsilon = 0$ (which corresponds to a non perturbation of the mean of the system), and extracting the variance component gives the result for F_{var}.

A.3.2 Proof of Lemma A.3.15

Proof. Let $X = (x_1, .., x_n) \in \mathbb{R}^{Nd}$ such that $x_i \neq x_j$ if $i \neq j$.

Define the evaluation map

$$\mathrm{Ev}_X : \mathcal{H}_d \to \mathbb{R}^{Nd},\ \mathrm{Ev}_X(f) \stackrel{\text{def}}{=} (f(x_1), ..., f(x_N)),$$

where $f(x_i) \in \mathbb{R}^d$ and the concatenation is in $\mathbb{R}^{Nd}$.

Restriction of Ev_X to $\mathcal{H}_d^X$ is an isomorphism. Consider the restriction $\mathrm{Ev}_X : \mathcal{H}_d^X \to \mathbb{R}^{Nd}$. First, Ev_X is injective on $\mathcal{H}_d^X$. Indeed, let $f \in \mathcal{H}_d^X$ and assume $\mathrm{Ev}_X(f) = 0$. Write $f(\cdot) = \sum_{i=1}^N K(x_i, \cdot)c_i$ for some $c = (c_1, ..., c_N) \in \mathbb{R}^{Nd}$ with $c_i \in \mathbb{R}^d$. Then for each $l \in [N]$

$$0 = f(x_\ell) = \sum_{i=1}^N K(x_i, x_\ell)c_i,$$

which in block form reads $(K_X \otimes I_d)c = \overline{K}, c = 0$. Because K is strictly positive definite and the points (x_i) are distinct, then K_X is invertible, hence $\overline{K}$ is invertible and $c = 0$, so $f = 0$. Thus Ev_X is injective.

Since $\dim(\mathcal{H}_d^X) < Nd$ and the family $\{K(x_i, \cdot)e_j\}$ spans $\mathcal{H}_d^X$, we have $\dim(\mathcal{H}_d^X) = Nd$. Therefore Ev_X is bijective. In particular, for any $g = (g_1, ..., g_N) \in \mathbb{R}^{Nd}$ with $g_i \in \mathbb{R}^d$, there exists a unique $f \in \mathcal{H}_d^X$ such that $f(X) = g$, namely

$$f(\cdot) = \sum_{i=1}^N K(x_i, \cdot)c_i, c = (K_X^{-1} \otimes I_d), g = \overline{K}^{-1}g. \tag{51}$$

Isometry after equipping $\mathbb{R}^{Nd}$ with the kernel-induced norm. Define on $\mathbb{R}^{Nd}$ the inner product

$$\langle g, h\rangle_{\overline{K}^{-1}} \stackrel{\text{def}}{=} g^\top \overline{K}^{-1}h,\ \|g\|^2_{\overline{K}^{-1}} \stackrel{\text{def}}{=} g^\top \overline{K}^{-1}g.$$

Then $\mathrm{Ev}_X : \mathcal{H}_d^X \to \left(\mathbb{R}^{Nd}, \langle\cdot,\cdot\rangle_{\overline{K}^{-1}}\right)$ is an isometry. Indeed, let $f \in \mathcal{H}_d^X$ and write $f = \sum_i K(x_i, \cdot)c_i$. Using the product structure of $\mathcal{H}_d$ and reproducing kernel calculus,

$$\begin{aligned}
\|f\|^2_{\mathcal{H}_d} &= \left\langle \sum_i K(x_i, \cdot)c_i, \sum_\ell K(x_\ell, \cdot)c_\ell \right\rangle_{\mathcal{H}_d} \\
&= \sum_{i,\ell} c_i^\top c_\ell, K(x_i, x_\ell) \\
&= c^\top (K_X \otimes I_d)c
\end{aligned}$$

$$= c^\top \overline{K}, c.$$

On the other hand $g \overset{\text{def}}{=} f(X) = \overline{K}c$, hence $c = \overline{K}^{-1}g$, and therefore

$$\|f\|^2_{\mathcal{H}_d} = \left(\overline{K}^{-1}g\right)^\top \overline{K}, \left(\overline{K}^{-1}g\right) = g^\top \overline{K}^{-1} g = |g|^2_{\overline{K}^{-1}}. \tag{52}$$

This proves that $\|f\|_{\mathcal{H}_d} = \|\mathrm{Ev}_X(f)\|_{\overline{K}^{-1}}$ for all $f \in \mathcal{H}_d^X$.

Reference (Lebesgue) measure on $\mathcal{H}_d^X$. Since $\mathcal{H}_d^X$ is a finite-dimensional Hilbert space, the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_d}$ induces a canonical volume measure, defined as the Lebesgue measure in any orthonormal coordinate system. Concretely, choose an orthonormal basis $(u_1, ..., u_{Nd})$ of $\mathcal{H}_d^X$ and define, for Borel sets $A \subset \mathcal{H}_d^X$,

$$\mu_{\mathcal{H}_d^X}(A) \overset{\text{def}}{=} \lambda_{Nd}(\Phi(A)), \Phi(f) \overset{\text{def}}{=} \left(\langle f, u_1 \rangle_{\mathcal{H}_d}, ..., \langle f, u_{Nd} \rangle_{\mathcal{H}_d}\right),$$

where λ_{Nd} is the usual Lebesgue measure on $\mathbb{R}^{Nd}$. This definition does not depend on the chosen orthonormal basis. Equivalently, using the isometry Eq. 52, $\mu_{\mathcal{H}_d^X}$ is the pullback by Ev_X of the Lebesgue measure on $\mathbb{R}^{Nd}$ associated with the inner product $\langle \cdot, \cdot \rangle_{\overline{K}^{-1}}$, i.e.

$$\mu_{\mathcal{H}_d^X} = (\mathrm{Ev}_X)^{-1} \# \lambda_{\overline{K}^{-1}},$$

where $\lambda_{\overline{K}^{-1}}$ denotes the Lebesgue measure on $\mathbb{R}^{Nd}$ in an orthonormal basis for $\langle \cdot, \cdot \rangle_{\overline{K}^{-1}}$. In standard Euclidean coordinates $g \in \mathbb{R}^{Nd}$, one has

$$\mathrm{d}\lambda_{\overline{K}^{-1}}(g) = \left(\det \overline{K}\right)^{-\frac{1}{2}} \mathrm{d}g,$$

since an orthonormalization amounts to the linear change of variables $z = \overline{K}^{-\frac{1}{2}} g$.

Finite-dimensional Gaussian density. Let $G(X) \sim \mathcal{N}\left(0, \overline{K}\right)$. Its density with respect to the standard Lebesgue measure $\mathrm{d}g$ on $\mathbb{R}^{Nd}$ is

$$p_{G(X)}(g) = \frac{\exp\left(-\frac{1}{2} g^\top \overline{K}^{-1} g\right)}{(2\pi)^{Nd/2} \left(\det \overline{K}\right)^{1/2}}.$$

By Eq. 52, for $f \in \mathcal{H}_d^X$ and $g = \mathrm{Ev}_X(f)$, $g^\top \overline{K}^{-1} g = \|f\|^2_{\mathcal{H}_d}$. Moreover, since Ev_X is an isometry between $\mathcal{H}_d^X$ and $\left(\mathbb{R}^{Nd}, \langle \cdot, \cdot \rangle_{\overline{K}^{-1}}\right)$, the change-of-variables formula yields

$$\mathbb{P}(\pi_X G \in \mathrm{d}f) = p_{G(X)}(\mathrm{Ev}_X(f))\, \mathrm{d}\lambda_{\overline{K}^{-1}}(\mathrm{Ev}_X(f)) \propto \exp\left(-\frac{1}{2}\|f\|^2_{\mathcal{H}_d}\right) \mathrm{d}\mu_{\mathcal{H}_d^X}(f),$$

which is the claimed statement.

□

A.3.3 Proof of Theorem A.3.10

Assumption A.A.3

- **i.** ∇V has at most a linear growth: $\nabla V(x) \leq C(1 + |x|)$
- **ii.** $k, \nabla_x k$ are bounded on $\mathbb{R}^d \times \mathbb{R}^d$, and k is symetric.
- **iii.** $\sup_{N \geq 1} \mathbb{E}[m_4(\rho_0^N)] < \infty$

The growth assumptions imposed on the initialization (ρ_0^N) and on ∇V are fairly standard in the SDE literature. The assumption on the kernel k may appear restrictive, but it remains sufficiently mild to accommodate Gaussian kernels k_σ.

Proof. Start from the following equation

$$\hat{\rho}_{(n+1)\varepsilon} = \big(\mathrm{id} + \varepsilon v_{n\varepsilon} + \sqrt{\varepsilon} G_n\big) \# \hat{\rho}_{n\varepsilon}, \;\; G_n \sim \mathcal{G}(\beta^2 K_{\mathsf{SBS}}) \tag{53}$$

This equation can be interpreted as a noisy version of the discretized in time continuity equation

$$\partial_t \rho_t + \nabla \cdot (\rho_t v_t) = 0$$

As it is classical within PDE theory, evolutions of this type are usually discretized in space thanks to the use of particle systems. First approximate $\hat{\rho}_0$ by the empirical distribution of a particle system $\overline{\rho}_0^\varepsilon \overset{\text{def}}{=} \frac{1}{N}\sum_{i=1}^N \delta_{X_0^{\varepsilon,i}}$. Then, the iteration Eq. 53 initialized with $\overline{\rho}_0$ rewrites as an iteration on the particles

$$X_{n+1}^{\varepsilon,i} = \varepsilon v_{n(X_n^{\varepsilon,i})} + \sqrt{\varepsilon} G_n(X_n^{\varepsilon,i}) \tag{54}$$

The iteration Eq. 53 initialized with $\hat{\rho}_0$ is then approximated with

$$\hat{\rho}_{n\varepsilon} \approx \overline{\rho}_{n\varepsilon} = \frac{1}{N}\sum_{i=1}^N \delta_{X_n^{\varepsilon,i}}$$

We now want to derive a continuous time version of Eq. 54, obtained when $\varepsilon \to 0$. To do so, remark that the vector $G_n(X_n^\varepsilon) \overset{\text{def}}{=} (G_n(X_n^{\varepsilon,1}), ..., G_n(X_n^{\varepsilon,N})) \in \mathbb{R}^{Nd}$ is Gaussian, and its covariance matrix is given by

$$C_{\beta(X_n^\varepsilon)} = \beta^2 \overline{K}(X_n^\varepsilon), \;\; K_{\mathsf{SBS}}(X_n^\varepsilon) = \begin{pmatrix} K_{1,1} & \dots & K_{1,N} \\ \vdots & \ddots & \vdots \\ K_{N,1} & \dots & K_{N,N} \end{pmatrix}, \;\; K_{i,j} = k_{\mathsf{SBS}}(x_i, x_j) I_d.$$

which can be reformulated as

$$\overline{K}(X) = K_{\mathsf{SBS}}(X) \otimes I_d, \;\; K_{\mathsf{SBS}}(X)_{i,j} = k_{\mathsf{SBS}}(X^i, X^j)$$

Then, using the theory of Gaussian vectors, Eq. 54 can be expressed in a matrix form by

$$X_{n+1}^\varepsilon = \varepsilon v_{n(X_n^\varepsilon)} + \sqrt{\varepsilon \beta^2 \overline{K}(X_n^\varepsilon)} \xi_{n+1} \tag{55}$$

where $(\xi_n)_{n\in\mathbb{N}}$ are i.i.d standard Gaussian variables of dimension Nd.

One can identify Eq. 55 as Euler-Maruyama discretization of the following SDE

$$\mathrm{d}X_t = v_{t(X_t)}\,\mathrm{d}t + \beta\sqrt{\overline{K}(X_t)}\,\mathrm{d}B_t \tag{56}$$

The equations Eq. 56, Eq. 54 and Eq. 55 correspond respectively to to the continuous in time/discretized in space; continuous in space/discretized in time and discretized in time/discretized in space version of the underlying continuous in time/continuous in space dynamic Eq. 45.

To show the tightness of the laws of the empirical distribution of the particle systems, let us first recall the following classical tightness criterion for stochastic processes taking values in a polish space.

Theorem A.A.13 – *Kolmogorov tightness criterion*

Let E be a Polish space (metric, separable and complete), and $\left(Y^N\right)_{N\geq 1}$ be a sequence of E-valued stochastic processes defined on $[0,T]$.

Assume that:

i. **(Marginal tightness)** For every $t\in[0,T]$, the family $\left(Y_t^N\right)_{N\geq 1}$ is tight in E;

ii. **(Kolmogorov-type increment bound)** There exist constants $p>0$, $\alpha>0$, and $C>0$ such that

$$\sup_{N\geq 1}\mathbb{E}\left[d(Y_t^N,Y_s^N)^p\right]<C\ |t-s|^{1+\alpha},\ \ \forall s,t\in[0,T].$$

Then the sequence of laws $\left(\mathcal{L}(Y^N)\right)_{N\geq 1}$ is tight in the space of probability measures over $C([0,T],E)$ equipped with the topology of uniform convergence.

The result is standard in the literature and follows from Kolmogorov's continuity theorem applied uniformly in N, combined with a compact containment argument and an Arzelà-Ascoli type criterion in probability.

We can apply this result to our setting. Note that a similar proof can be done with another set of assumptions on V if it is coercive outside of a compact set (allowing in this case the study of functions with a bigger growth at infinity, at the price of a control on the direction of ∇V).

To prove Theorem A.3.10, we are going to apply Kolmogorov's tightness criterion in $E=\mathcal{P}_1\left(\mathbb{R}^d\right)$ with $p=4$ and $\alpha=1$. In order to simplify the notations, in the following C is a constant independent of N,s,t wich may vary from line to line.

The proof relies on the following estimate

$$\sup_{N\geq 1}\sup_{t\in[0,T]}\mathbb{E}[m_4(\rho_t^N)]<\infty$$

Notice that the first condition of Kolmogorov's tightness criterion directly follows from this estimate. To prove this estimate, apply Ito's formula to $x\mapsto\varphi(x)=\|x\|^4$ to compute $m_4(\rho_t^N)=\frac{1}{N}\sum_{i=1}^N\left\|X_t^{i,N}\right\|^4$

$$m_4(\rho_t^N)=m_4(\rho_0^N)+\int_0^t A\varphi(\rho_s^N)\,\mathrm{d}s+N_t$$

where

$$A(\rho_s^N)=\frac{1}{N}\sum_{i=1}^N\nabla\varphi(X_s^i)\cdot b(X_s^i,\rho_s^N)+\frac{1}{N}\sum_{i=1}^N k(X_s^i,X_s^i)\Delta\varphi(X_s^i)$$

and N_t is a local martingale. Note that $A(\rho)$ corresponds to $\mathcal{A}F(\rho)$, with $F(\rho) = \langle \rho, \varphi \rangle$. $\mathcal{A}$ is an operator that corresponds to the generator of the measured-valued process ρ^N. This observation will be central for the proof of Theorem A.A.14.

Our assumptions ensure that $A(\rho_s^N) \leq C(1 + m_4(\rho_s^N))$, with C independent of N. Then, after localizing the local martingale with an appropriate stopping time τ_k, taking the expected value in equation the previous equation gives

$$\mathbb{E}\Big(m_4\big(\rho^N_{t\wedge\tau_k}\big)\Big) \leq C\left(1 + \int_0^t \mathbb{E}\Big[m_4\big(\rho^N_{s\wedge\tau_k}\big)\Big]\,\mathrm{d}s\right)$$

Grownwall and Fatou's lemma conclude the proof.

In order to prove that the second condition of Kolmogorv's tightness criterion is verified, remark that $\mathbb{E}\Big[W_1(\rho_t^N, \rho_s^N)^4\Big] \leq \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^N \big\|X_t^i - X_s^i\big\|^4\Big]$

Moreover, if we define the local martingale $M_t^{i,N} = \beta \int_s^t \sum_{j=1}^N K\big(\overline{X}\big)_{i,j}\,\mathrm{d}W_r^j$, we have that $X_t^i - X_s^i = \int_s^t b(X_r^i)\,\mathrm{d}r, +, M_t^i - M_s^i$

Young's inequality gives

$$\begin{aligned}\mathbb{E}\Big[W_1(\rho_t^N, \rho_s^N)^4\Big] &\leq \frac{C}{N}\sum_{i=1}^N\left(\mathbb{E}\left[\left\|\int_s^t b(X_r^i, \rho_r^N)\,\mathrm{d}r\right\|^4\right] + \mathbb{E}\Big[\big\|M_t^{i,N} - M_s^{i,N}\big\|^4\Big]\right)\\ &\leq \frac{C}{N}\sum_{i=1}^N\left(|t-s|^3\int_s^t \mathbb{E}\Big[\big\|b(X_r^i, \rho_r^N)\big\|^4\Big]\,\mathrm{d}r + \mathbb{E}\Big[\big\|M_t^{i,N} - M_s^{i,N}\big\|^4\Big]\right)\end{aligned}$$

The first integrand is bounded because $\|b(x,\rho)\| \leq C(1 + m_1(\rho))$ and because of the estimate obtained at the first step of the proof. The quadratic variation of the martingale term is given by

$$\big\langle M^{i,N}\big\rangle_t = \beta^2 \int_0^t k(X_r^i, X_r^i) I_d\,\mathrm{d}r$$

Then, because k is bounded, BDG's inequality finally ensures that the second term is bounded by a constant times $|t-s|^2$. Because $|t-s|^4 \leq T^2|t-s|^2$, this finally yields $\mathbb{E}\Big[W_1(\rho_t^N, \rho_s^N)^4\Big] \leq C|t-s|^2$ which concludes the proof.

□

A.3.4 Details of Theorem A.3.11

First, we present rigorously the martingale problem and the related convergence result. Then, we explain how to relate the martingale problem to the SPDE of interest heuristically.

Martingale problem.

Definition A.A.14

We define the set of cylindrical functions on $\mathcal{P}_1(\mathbb{R}^d)$ by $\mathrm{Cyl}(\mathcal{P}_1(\mathbb{R}^d))$ the set of functions that can be expressed as:

$$F(\mu) = f(\langle \mu, \varphi_1 \rangle, ..., \langle \mu, \varphi_n \rangle)$$

with $n \geq 1$, $\varphi_1, .., \varphi_n \in C_c^{\infty(\mathbb{R}^d)}$, $f \in C_b^2(\mathbb{R}^n)$.

Remark that a cylindrical function is continuous for the W_1 topology on $\mathcal{P}_1(\mathbb{R}^d)$. We can now define the martingale problem associated to the SPDE.

Definition A.A.15

Let $\Omega = C([0,T], \mathcal{P}_1(\mathbb{R}^d))$, $\rho : \omega \in \Omega \mapsto \omega$ be the canonical process, and $(\mathcal{F}_t)_{t\in[0,T]}$ be its associated filtration. Given $\mu_0 \in \mathcal{P}(\mathcal{P}_1(\mathbb{R}^d))$, we say that a law $\mathbb{Q}$ on Ω solves the martingale problem associated to μ_0 if

i. $\mathbb{Q} \circ \rho_0^{-1} = \mu_0$

ii. For all cynlindrical function $F(\cdot) = f(\langle \cdot, \varphi_1 \rangle, ..., \langle \cdot, \varphi_n \rangle)$, the process

$$M_t^{F(\rho)} \overset{\text{def}}{=} F(\rho_t) - F(\rho_0) - \int_0^t \mathcal{A}F(\rho_s)\,\mathrm{d}s$$

is an $(\mathcal{F}_t)_{t\in[0,T]}$ martingale under the law $\mathbb{Q}$, where

$$\mathcal{A}F(\rho_s) = \sum_{i=1}^n \partial_i f(z(\rho_s)) \int_{\mathbb{R}^d} b(x, \rho_s) \cdot \nabla\varphi_i(x)\,\mathrm{d}\rho_s(x) + \frac{\beta^2}{2} \sum_{i,j=1}^n \partial_{i,j} f(z(\rho_s)) \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \nabla\varphi_i(x) \cdot \nabla\varphi_j(y) k(x,y)\,\mathrm{d}\rho_s(x)\,\mathrm{d}\rho_s(y)$$

and $z(\rho_s) \overset{\text{def}}{=} (\langle \rho_s, \varphi_1 \rangle, ..., \langle \rho_s, \varphi_n \rangle)$.

The operator $\mathcal{A}$ must be understood the generator of the solution of the SPDE, seen as a Markov process in $\mathcal{P}_1(\mathbb{R}^d)$. This allows us to avoid tackling directly the SPDE, as it is a usual approach in the finite dimensional setting with the study of SDE.

We can now state the limiting result.

Theorem A.A.14 – *Formal version of the first part of Theorem A.3.11*

Suppose that ∇V is Lipschitz, and that the particle systems verify the following initialization condition

$$\lim_{N\to\infty} W_1\left(\mathcal{L}\left(\frac{1}{N}\sum_{i=1}^N \delta_{X_0^{i,N}}\right), \mu_0\right) = 0.$$

Then, any limit point of $\{\mathcal{L}(\rho^N)\}_{N\geq 1}$ solves the previous martingale problem related to $\mathcal{A}$.

Proof. Let $\mathbb{Q}_N \overset{\text{def}}{=} \mathcal{L}(\rho^N)$, and assume that a subsequence of these laws converges to $\mathbb{Q}$. To simplify the notations in the following the subsequence won't be distinguished from the original sequence. First, remark that the limiting assumption of the initial empirical distribution of the particle system ensures that $\mathbb{Q}$ verify the first criteria of the martingale problem.

Let

$$\rho_t^N = \frac{1}{N}\sum_{a=1}^N \delta_{X_t^a},\ F(\rho) = f(\langle\rho,\varphi_1\rangle,...,\langle\rho,\varphi_n\rangle),$$

and define

$$z_i^{N(t)} = \langle\rho_t^N,\varphi_i\rangle = \frac{1}{N}\sum_{a=1}^N \varphi_i(X_t^a),\ z^{N(t)} = \big(z_1^N(t),...,z_n^N(t)\big).$$

We write the particle system as

$$\mathrm{d}X_t = b_{\mathsf{SBS}}(X_t,\rho_t^N),\mathrm{d}t + \beta\Big(K(X_t)^{1/2}\otimes I_d\Big)\,\mathrm{d}B_t,$$

where

$$X_t = (X_t^1,...,X_t^N)\in\mathbb{R}^{dN},\ B_t = (B_t^1,...,B_t^N)\in\mathbb{R}^{dN},\ B_t^a\in\mathbb{R}^d,$$

and

$$K(X_t) = \big(k_{\mathsf{SBS}}\big(X_t^a,X_t^b\big)\big)_{a,b\in[N]}.$$

Equivalently, for each particle $a\in[N]$,

$$\mathrm{d}X_t^a = b_{\mathsf{SBS}}(X_t^a,\rho_t^N),\mathrm{d}t + \beta\sum_{b=1}^N K_{ab}^{1/2}(X_t),\mathrm{d}B_t^b,\ B_t^b\in\mathbb{R}^d.$$

Applying Itô's formula to $z_i^{N(t)}$ gives

$$\begin{aligned}\mathrm{d}z_i^{N(t)} &= \frac{1}{N}\sum_{a=1}^N \nabla\varphi_i(X_t^a)\cdot\mathrm{d}X_t^a + \frac{1}{2}\frac{1}{N}\sum_{a=1}^N \mathrm{trace}\Big(D^2\varphi_i(X_t^a),\mathrm{d}\langle X^a\rangle_t\Big)\\ &= \langle b_{\mathsf{SBS}}(\cdot,\rho_t^N)\cdot\nabla\varphi_i,\rho_t^N\rangle\,\mathrm{d}t + \frac{\beta^2}{2}\big\langle k(\cdot,\cdot)_{\mathrm{diag}}\Delta\varphi_i,\rho_t^N\big\rangle\,\mathrm{d}t + \mathrm{d}M_t^{N,i},\end{aligned}$$

where

$$k(\cdot,\cdot)_{\mathrm{diag}}(x) = k_{\mathsf{SBS}}(x,x),$$

and

$$\mathrm{d}M_t^{N,i} = \frac{\beta}{N}\sum_{a=1}^N\sum_{b=1}^N K_{ab}^{1/2}(X_t),\nabla\varphi_i(X_t^a)\cdot\mathrm{d}B_t^b.$$

Indeed,

$$\mathrm{d}\langle X^a, X^c\rangle_t = \beta^2 \sum_{b=1}^{N} K_{ab}^{1/2}(X_t) K_{cb}^{1/2}(X_t) I_d, \mathrm{d}t = \beta^2 k(X_t^a, X_t^c) I_d, \mathrm{d}t.$$

In particular,

$$\mathrm{d}\langle X^a\rangle_t = \beta^2 k(X_t^a, X_t^a) I_d, \mathrm{d}t.$$

Hence the quadratic covariation of the martingales associated with the linear observables is

$$\mathrm{d}\big\langle M^{N,i}, M^{N,j}\big\rangle_t = \beta^2 \iint \nabla\varphi_i(x) \cdot \nabla\varphi_j(y), k(x,y), \mathrm{d}\rho_t^N(x)\, \mathrm{d}\rho_t^N(y)\, \mathrm{d}t.$$

Applying Itô's formula to $F(\rho_t^N) = f(z^N(t))$ yields

$$\mathrm{d}F(\rho_t^N) = \sum_{i=1}^{n} \partial_i f(z^N(t)), \mathrm{d}z_i^N(t) + \frac{1}{2} \sum_{i,j=1}^{n} \partial_{ij} f(z^N(t)), \mathrm{d}\big\langle z_i^N, z_j^N\big\rangle_t.$$

The martingale part is therefore

$$\mathrm{d}M_t^{N,F} = \sum_{i=1}^{n} \partial_i f(z^N(t))\, \mathrm{d}M_t^{N,i},$$

that is,

$$\mathrm{d}M_t^{N,F} = \frac{\beta}{N} \sum_{a=1}^{N} \sum_{b=1}^{N} K_{ab}^{1/2}(X_t) \left[\sum_{i=1}^{n} \partial_i f(z^N(t)), \nabla\varphi_i(X_t^a) \right] \cdot \mathrm{d}B_t^b.$$

Collecting the finite variation terms gives

$$\mathrm{d}F(\rho_t^N) = \mathcal{A}F(\rho_t^N), \mathrm{d}t + \mathrm{d}M_t^{N,F},$$

with

$$\mathcal{A}F(\rho) = \sum_{i=1}^{n} \partial_i f(z(\rho)) \left[\int b_{\mathsf{SBS}}(x,\rho) \cdot \nabla\varphi_i(x)\, \mathrm{d}\rho(x) + \frac{\beta^2}{2} \int k(x,x) \Delta\varphi_i(x)\, \mathrm{d}\rho(x) \right]$$
$$+ \frac{\beta^2}{2} \sum_{i,j=1}^{n} \partial_{ij} f(z(\rho)) \iint \nabla\varphi_i(x) \cdot \nabla\varphi_j(y), k(x,y), \mathrm{d}\rho(x)\, \mathrm{d}\rho(y).$$

and defining

$$\Phi_{F(x,\rho)} \stackrel{\text{def}}{=} \sum_{i=1}^{n} \partial_i f(z(\rho)), \nabla\varphi_i(x), \;\; z(\rho) = (\langle\rho,\varphi_1\rangle, ..., \langle\rho,\varphi_n\rangle).$$

gives

$$\mathrm{d}M_t^{N,F} = \frac{\beta}{N} \sum_{a=1}^{N} \sum_{b=1}^{N} K_{ab}^{1/2}(X_t), \Phi_F(X_t^a, \rho_t^N) \cdot \mathrm{d}B_t^b$$

with

$$\mathrm{d}\langle M^{N,F}\rangle_t = \beta^2 \iint \Phi_F(x,\rho_t^N)\cdot\Phi_F(y,\rho_t^N), k(x,y), \mathrm{d}\rho_t^N(x)\,\mathrm{d}\rho_t^N(y)\,\mathrm{d}t.$$

In particular,

$$F(\rho_t^N) = F(\rho_0^N) + \int_0^t \mathcal{A}F(\rho_s^N)\,\mathrm{d}s + M_t^{N,F} \tag{57}$$

Where $M_t^{N,F}$ is a real martingale (because its quadratic variation is bounded). Note that the expression of the generator $\mathcal{A}F$ is identified through this procedure. Then, for any $k \in \mathbb{N}^*$ and $0 \le t_1 < .. < t_k < s < t \le T$, and $G : \mathcal{P}_1\big(\mathbb{R}^d\big)^k \to \mathbb{R}$ continuous and bounded

$$\mathbb{E}\Big[G\big(\rho_{t_1}^N, .., \rho_{t_k}^N\big)\big(M^F_{t(\rho^N)} - M^F_{s(\rho^N)}\big)\Big] = 0$$

and then because of Eq. 57

$$\mathbb{E}\left[G\big(\rho_{t_1}^N, .., \rho_{t_k}^N\big)\left(\int_s^t \mathcal{A}F(\rho_u^N)\,\mathrm{d}u\right)\right] = 0$$

which can be rewritten in the canonical space as

$$\mathbb{E}_{\mathbb{Q}_N}\left[G\big(\rho_{t_1}, .., \rho_{t_k}\big)\left(\int_s^t \mathcal{A}F(\rho_u)\,\mathrm{d}u\right)\right] = 0$$

Defining $H : \rho \in \Omega \mapsto G\big(\rho_{t_1}, .., \rho_{t_k}\big)\big(\int_s^t \mathcal{A}F(\rho_u)\,\mathrm{d}u\big)$, one can check that H is continuous in $C\big([0,T], \mathcal{P}_1\big(\mathbb{R}^d\big)\big)$ (because ∇V is Lipschitz), and that

$$|H(\rho)| \le C\left(1 + \int_0^T m_1(\rho_r)\,\mathrm{d}r\right)$$

Then, because of the moment estimate of the previous proof $\sup_{N\ge1}\mathbb{E}_{\mathbb{Q}_N}[|H(\rho)|^2] < \infty$ Then, for all $\varepsilon > 0$, one can find $M \ge 0$ such that $\sup_{N\ge1}\mathbb{E}_{\mathbb{Q}_N}\Big[|H(\rho)|\mathbb{1}_{|H|>M}\Big] \le \varepsilon$

$$\mathbb{E}_{\mathbb{Q}_N}[H(\rho)] = \mathbb{E}_{\mathbb{Q}_N}\Big[H(\rho)\mathbb{1}_{|H|\le M}\Big] + \mathbb{E}_{\mathbb{Q}_N}\Big[H(\rho)\mathbb{1}_{|H|>M}\Big]$$

one can pass to the limit in the previous expression because $H\mathbb{1}_{|H|\le M}$ is continuous and bounded and because $\mathbb{Q}_N \Rightarrow \mathbb{Q}$ to get

$$|\mathbb{E}_{\mathbb{Q}}[H(\rho)]| \le \varepsilon$$

and because this expression is true for all $\varepsilon > 0$, that

$$\mathbb{E}_{\mathbb{Q}}[H(\rho)] = 0.$$

As this equality is true for any sequence of time index $0 \le t_1 < .. < t_k < s < t$, and for any continuous and bounded functions G, one can find that for all $0 \le s < t \le T$, $\mathbb{E}_{\mathbb{Q}}\Big[M_t^{F(\rho)}|\mathcal{F}_s\Big] = M_s^{F(\rho)}$, which concludes the proof.

□

Heuristic derivation of the SPDE in Eq. 45.

Let $(\rho_t)_{t\geq 0}$ be a solution of the martingale problem associated with the generator $\mathcal{A}$. Let $\varphi, \psi \in C_c^{\infty(\mathbb{R}^d)}$. Testing the martingale problem on linear observables

$$F(\rho) = \langle \rho, \varphi \rangle,$$

gives

$$\mathrm{d}\langle \rho_t, \varphi \rangle = \left[\int_{\mathbb{R}^d} b_{\mathsf{SBS}}(x, \rho_t) \cdot \nabla \varphi(x) \, \mathrm{d}\rho_t(x) + \frac{\beta^2}{2} \int_{\mathbb{R}^d} k(x,x) \Delta \varphi(x) \, \mathrm{d}\rho_t(x) \right] \mathrm{d}t + \mathrm{d}M_t(\varphi).$$

In order to justify the stochastic integral representation of the martingale $M(\varphi)$, we first compute the quadratic variation $\langle M(\varphi), M(\psi) \rangle_t$. To do so, now set

$$F(\rho) = \langle \rho, \varphi \rangle \langle \rho, \psi \rangle.$$

Then $F(\rho) = f(z_1, z_2)$ with $z_1 = \langle \rho, \varphi \rangle$, $z_2 = \langle \rho, \psi \rangle$ and $f(z_1, z_2) = z_1 z_2$.

Hence

$$\partial_{12} f = \partial_{21} f = 1, \ \partial_{11} f = \partial_{22} f = 0.$$

Applying the generator gives

$$\mathcal{A}F(\rho) = (\text{first-order terms}) + \beta^2 \iint \nabla \varphi(x) \cdot \nabla \psi(y), k(x,y), \mathrm{d}\rho(x) \, \mathrm{d}\rho(y).$$

On the other hand, by Itô's formula,

$$\mathrm{d}(\langle \rho_t, \varphi \rangle \langle \rho_t, \psi \rangle) = (\text{first-order terms}) \, \mathrm{d}t + \mathrm{d}\langle M(\varphi), M(\psi) \rangle_t + \mathrm{d}M_t^F.$$

Identifying the second-order contributions yields

$$\mathrm{d}\langle M(\varphi), M(\psi) \rangle_t = \beta^2 \iint \nabla \varphi(x) \cdot \nabla \psi(y), k(x,y) \, \mathrm{d}\rho_t(x) \, \mathrm{d}\rho_t(y) \, \mathrm{d}t. \tag{58}$$

This covariance structure is the point at which it is natural to represent the martingale M as a stochastic integral against a Wiener process.

Let $\mathcal{H}_k^d$ be the vector-valued RKHS associated with the kernel kI_d, and let $(W_t)_{t\geq 0}$ be the canonical cylindrical Wiener process on $\mathcal{H}_k^d$, i.e.,

$$\mathbb{E}\Big[\langle h, W_t \rangle_{\mathcal{H}_k^d} \langle g, W_s \rangle_{\mathcal{H}_k^d} \Big] = (t \wedge s) \langle h, g \rangle_{\mathcal{H}_k^d}, \ h, g \in \mathcal{H}_k^d.$$

For $\varphi \in C_c^{\infty(\mathbb{R}^d)}$, define

$$H_{t(\varphi)} \overset{\text{def}}{=} \int_{\mathbb{R}^d} k(x, \cdot) \nabla \varphi(x) \, \mathrm{d}\rho_{t(x)}.$$

Because of the reproducing property,

$$\left\| H_{t(\varphi)} \right\|_{\mathcal{H}_k^d}^2 = \iint_{\mathbb{R}^d \times \mathbb{R}^d} \nabla \varphi(x) \cdot \nabla \varphi(y), k(x,y), \mathrm{d}\rho_t(x), \mathrm{d}\rho_t(y) \leq \|k\|_\infty \|\nabla \varphi\|_\infty^2$$

Then, because the process $(H_t(\varphi))_{t\in[0,T]}$ is predictable and square integrable

$$\mathbb{E}\left[\int_0^T \|H_s(\varphi)\|^2_{\mathcal{H}_k^d}\,\mathrm{d}s\right] < \infty,$$

the following stochastic integral is well defined

$$N_{t(\varphi)} \overset{\text{def}}{=} \beta \int_0^t \left\langle H_{s(\varphi)}, \mathrm{d}W_s \right\rangle_{\mathcal{H}_k^d}.$$

By the Itô isometry, for any φ, ψ,

$$\langle N(\varphi), N(\psi)\rangle_t = \beta^2 \int_0^t \langle H_s(\varphi), H_s(\psi)\rangle_{\mathcal{H}_k^d}\,\mathrm{d}s.$$

Using again the reproducing property we get

$$\langle H_s(\varphi), H_s(\psi)\rangle_{\mathcal{H}_k^d} = \iint \nabla\varphi(x)\cdot\nabla\psi(y), k(x,y), \mathrm{d}\rho_s(x)\,\mathrm{d}\rho_s(y).$$

Then, by identifying this expression with equation Eq. 58, we finally get

$$\langle M(\varphi), M(\psi)\rangle_t = \langle N(\varphi), N(\psi)\rangle_t,\ \forall t \in [0,T],\ \forall \varphi, \psi \in C_c^{\infty(\mathbb{R}^d)}$$

Heuristically, this suggests identifying $M(\varphi)$ with $N(\varphi)$. The key interest in this representation of the martingales $M(\varphi)$ is that the same noise W drives all martingales $N(\varphi)$. This noise will correspond to the noisy part of the final SPDE.

This is the infinite-dimensional analogue of the finite-dimensional representation theorem, which states that any continuous martingale can be represented as a stochastic integral against a single Brownian motion.

The representation theorem for martingales in Hilbert spaces (Da Prato & Zabczyk, 2014, Theorem 8.2) appears as a natural route to make the above heuristic rigorous, since it allows one to represent a Hilbert-valued martingale with a given quadratic variation as a stochastic integral with respect to a cylindrical Wiener process.

In the following, we assume that the previous identification between M and N can be done. The weak evolution then becomes

$$\mathrm{d}\langle\rho_t, \varphi\rangle = \left[\int b_{\mathsf{SBS}}(x,\rho_t)\cdot\nabla\varphi(x)\,\mathrm{d}\rho_t(x) + \frac{\beta^2}{2}\int k(x,x)\Delta\varphi(x)\,\mathrm{d}\rho_t(x)\right]\mathrm{d}t + \mathrm{d}N_t(\varphi).$$

By integration by parts, the deterministic drift terms correspond formally to

$$\int b_{\mathsf{SBS}}(x,\rho_t)\cdot\nabla\varphi(x)\,\mathrm{d}\rho_t(x) = \langle\nabla\cdot(b_{\mathsf{SBS}}(\cdot,\rho_t)\rho_t), \varphi\rangle,$$

and

$$\int k_{\mathsf{SBS}}(x,x)\Delta\varphi(x)\,\mathrm{d}\rho_t(x) = \left\langle\Delta\left(k_{\mathrm{diag}}\rho_t\right), \varphi\right\rangle,$$

where

$$k_{\mathrm{diag}}(x) = k_{\mathsf{SBS}}(x,x).$$

Moreover, because

$$N_{t(\varphi)} = \beta \int_0^t \left\langle H_{s(\varphi)}, \mathrm{d}W_s \right\rangle_{\mathcal{H}_k^d}, \; H_{s(\varphi)} = \int k_{\mathsf{SBS}}(x, \cdot) \nabla \varphi(x) \, \mathrm{d}\rho_s(x).$$

Using the reproducing property of the RKHS,

$$\langle k_{\mathsf{SBS}}(x, \cdot) u, \mathrm{d}W_t \rangle_{\mathcal{H}_k^d} = u \cdot \mathrm{d}W_t(x),$$

we obtain the equivalent formal representation

$$\mathrm{d}N_{t(\varphi)} = \beta \int_{\mathbb{R}^d} \nabla \varphi(x) \cdot \mathrm{d}W_t(x) \, \mathrm{d}\rho_t(x). \tag{59}$$

Thus, the expression with the spatial integral is simply a shorthand for the Hilbert-space stochastic integral: one first integrates in x against ρ_t to build the integrand $H_{t(\varphi)}$, and then integrates in time against the cylindrical Wiener process W.

Equivalently, we define the random distribution-valued process

$$\mathcal{N}_t \overset{\text{def}}{=} -\beta \int_0^t \nabla \cdot (\rho_s \, \mathrm{d}W_s)$$

through its action on test functions

$$\langle \mathcal{N}_t, \varphi \rangle \overset{\text{def}}{=} \beta \int_0^t \int_{\mathbb{R}^d} \nabla \varphi(x) \cdot \mathrm{d}W_s(x) \, \mathrm{d}\rho_s(x) = N_t(\varphi).$$

Finally, collecting the previous equalities yields the following

$$\mathrm{d}\rho_t = \nabla \cdot (b_{\mathsf{SBS}}(\cdot, \rho_s)\rho_s) \, \mathrm{d}s + \frac{\beta^2}{2} \Delta\big(k_{\mathrm{diag}} \rho_s\big) \, \mathrm{d}s - \beta \nabla \cdot (\rho_s, \mathrm{d}W_s).$$

The SPDE should be understood in a weak sense, namely through its action against test functions, and is only a formal interpretation of the martingale problem. The rigorous object is the family of martingales $M_t(\varphi)$ and their quadratic variations. The colored-noise representation is introduced only after identifying the covariance structure of these martingales.

A.3.5 Proof of Theorem A.3.12

Proof.

To show that the GCN converges to white noise or to its SMD mean, it suffices to prove that, for any finite collection $X = (X_1, ..., X_n)$, the associated law converges to the expected limit. To this end, we consider its covariance matrix $\overline{K}_X = K(X) \otimes I_d$.

We have that for any $X \in \mathbb{R}^{Nd}$, $K_X \to \overline{J}$ where $\overline{J} = J \otimes I_d$ where $J = \mathbb{1}_{i,j \in [N]}$ is the matrix with only 1 values, thus $K_X^{\frac{1}{2}} \to \frac{\overline{J}}{\sqrt{N}}$ since $\overline{J}^2 = N\overline{J}$. Therefore in the case $\sigma \to \infty$, we have $k_\sigma(x, y) = 1$, then by Eq. 44, $\overline{G}_t^N = \left(\sum_{i=1}^N \frac{B_t^i}{\sqrt{N}} \right)_{j \in [N]}$ meaning

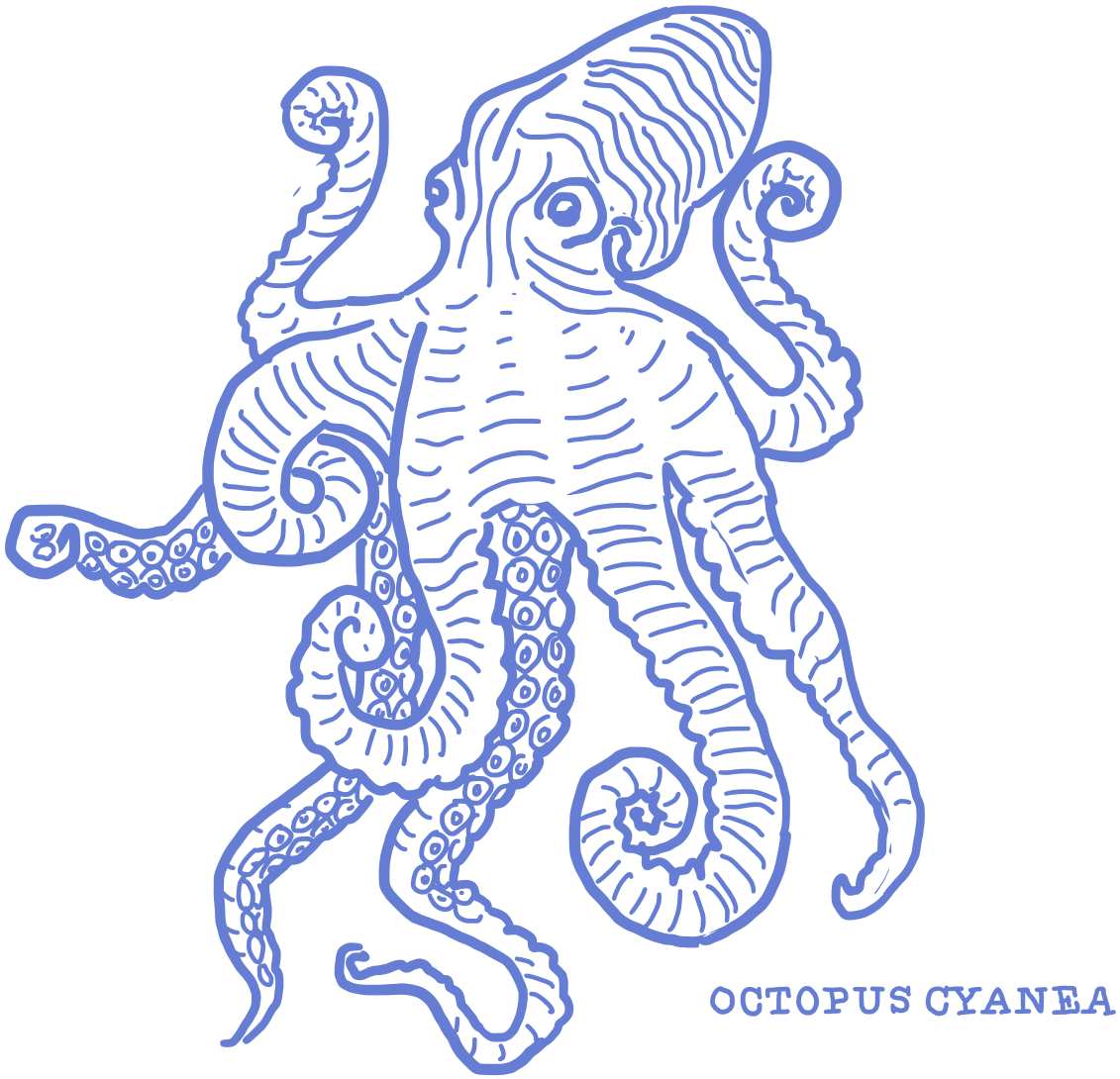
that $G_t\big(X_j^N\big) = Z$ for any $j \in [N]$ with $Z \sim \mathcal{N}(0, I_d)$. In other words, when $\sigma \to \infty$, GCN is SMD-mean.

In the case, $\sigma \to 0$, $k_\sigma(x,y)^{(d)} \to \mathbb{1}_0(x-y)$, meaning that $G \sim \mathcal{G}(k)$ satisfies $\mathbb{E}[G(x)G(y)] = \mathbb{1}_0(x-y)$ which defines a white noise.

□

Part B

A unified and formalized framework for global optimization



Contents

1 A unifying framework for global optimization 157
1.1 Instantiation of the framework 161
1.2 Properties of the measures 163
1.3 Sketch proof of Proposition B.1.2 166
2 Introduction to formal methods 171
2.1 LEAN theorem prover 173
2.2 Probability theory in LEAN 174
3 Formalization of global optimization algorithms 189
3.1 Formalization of decision-based algorithms 193
3.2 Formalization of CMA-ES 195
3.3 On the formalization of McKean-Vlasov-based approaches 196
3.4 Discussion and perspectives 197

Appendix

1 —

A unifying framework for global optimization

Abstract. We introduce an abstract measure-theoretic framework that serves as a tool to rigorously study stochastic iterative global optimization algorithms as a unified class. The framework is formulated in terms of probability kernels, which, via the Ionescu–Tulcea theorem, induce probability measures on the space of sequences of algorithm iterations, endowed with two intuitive properties. This framework answers the need for a general, implementation-independent formalism in the analysis of such algorithms, providing a starting point for formalizing global optimization results in proof-assistants. It is also a natural extension of the family-based organizational principles developed in previous sections, enabling rigorous reasoning about properties and behaviors of algorithms in a general, implementation-independent manner. To illustrate the relevance of our tool, we show that common algorithms fit naturally in the framework, and we also use it to give a rigorous proof of a general consistency theorem for stochastic iterative global optimization algorithms (Malherbe & Vayatis, 2017a, Proposition 3).

Throughout this manuscript, we have examined global optimization algorithms through the lens of *families*: decision-based approaches (see Chapter A.2) and McKean-Vlasov-based approaches (see Chapter A.3). This family-based organization revealed shared structural patterns within each class, enabling modular plugin architectures and unified implementations in the GLOBE library (see Chapter A.4). However, this family-level perspective raises a natural question: *can these organizing principles be extended and formalized into a unified mathematical framework that encompasses all stochastic iterative global optimization algorithms?*

This section provides an affirmative answer by proposing an abstract, measure-theoretic framework that unifies the analysis of decision-based, McKean-Vlasov, and all other stochastic iterative global optimization algorithms within a single kernel-based formalism. Rather than treating individual algorithms in isolation, this framework captures the essential commonalities that define stochastic iterative

global optimization algorithms, enabling rigorous reasoning about their properties and behaviors in a general, implementation-independent manner.

The framework is grounded in probability kernels — mathematical objects describing the probabilistic transitions between states in a stochastic process — which provide a unified language for expressing the internal mechanisms of any iterative algorithm without explicitly modeling it. While kernel-based perspectives on optimization algorithms have appeared in the literature (e.g., Gomez (2019)), the present work goes further by enabling rigorous measure-theoretic constructions on sequence spaces via the Ionescu–Tulcea theorem (Tulcea, 1949), a foundational step that prior approaches had not adequately developed.

Two foundational theorems about these measures are then established, providing key properties for rigorous analysis of global optimization algorithms. These results serve as sanity checks confirming that the framework aligns with the family-based organizational principles developed earlier. The framework is then put to use in a rigorous proof of Malherbe & Vayatis (2017a, Proposition 3), a general equivalence result that holds for *any* stochastic iterative global optimization algorithm — demonstrating how unifying abstractions yield results that transcend individual families.

Definition B.1.16 also gives a theoretical counterpart to the family-based organization of the previous chapters. In GLOBE, each family is a base class from which its algorithms inherit, and a plugin written for a family works for all of them (see Chapter A.4). With a general definition, the same hierarchy holds for the theory. Decision-based and McKean-Vlasov-based approaches are special cases of Definition B.1.16 (see Section B.1.1), and each of their algorithms is a special case of its family. Thus, a result can be proved once, at the most general level where it holds. A result on every algorithm holds for every family, and a result on a family holds for each of its algorithms. For instance, Malherbe & Vayatis (2017a, Proposition 3) holds for any algorithm, hence for every decision-based and McKean-Vlasov-based algorithm. In the same way, a module of a family, such as the trust region of Section A.2.4, could be studied once for all the algorithms of this family.

This hierarchy carries over to the formalization of Definition B.1.16 in Chapter B.3, which relies on the proof assistant LEAN (Moura & Ullrich, 2021) introduced in Chapter B.2. A family can be defined once as a special case of the formalized definition, and each of its algorithms only gives what is specific to it. A theorem proved for any algorithm then holds for each family and each algorithm, without a new proof. This is the case of Malherbe & Vayatis (2017a, Proposition 3), which is formalized once for all algorithms in Chapter B.3. The primary contribution of the framework remains mathematical. It gives a general foundation to reason about stochastic iterative global optimization algorithms, whatever their implementation or family.

The abstract framework. Given a continuous function $V : \Omega \to \beta$, any stochastic iterative global optimization algorithm $\mathscr{A}$ operates the same way. At first, it samples a point $X_0 \in \Omega$. This point is chosen according to the internal structure of the algorithm, potentially at random. Since V has not been evaluated yet, the

distribution of X_0 is the same for any function. Then, the algorithm iteratively samples points X_n in Ω for $n > 0$. Apart from the structure of $\mathscr{A}$, the distribution of X_n only depends on the previous iterations and the evaluations of the function V at these points. Thus, $\mathscr{A}$ can be reduced to two elements in which the internal structure is encapsulated.

Definition B.1.16 – *Stochastic iterative global optimization algorithm*

Given a measurable search space Ω and an evaluation space β (usually $\mathbb{R}$) associated with a measurable function $V : \Omega \to \beta$ to optimize, a stochastic iterative global optimization algorithm $\mathscr{A}$ is defined by:

i. an initial probability measure $\nu \in \mathcal{P}(\Omega)$ that describes the distribution of the first point X_0 sampled by the algorithm, and

ii. a sequence $\left(\kappa_n : \Omega^{n+1} \times \beta^{n+1} \rightsquigarrow \Omega\right)_{n\in\mathbb{N}}$, such that, for any $n \in \mathbb{N}$, κ_n is a probability kernel on the product space $\Omega^{n+1} \times \beta^{n+1}$ that describes the distribution of the next point X_{n+1} given the previous points $\mathbf{C}_n$ and their evaluations $V(\mathbf{C}_n)$. Given V, this corresponds to the conditional probability $\mathbb{P}(X_{n+1} \mid X_0, ..., X_n)$.

While this definition may appear abstract, it captures the essential structure of any stochastic iterative global optimization algorithm. The initial measure ν describes how the algorithm samples its first point, which is independent of the function being optimized. The sequence of kernels κ_n encapsulates the algorithm's decision-making process at each iteration, determining how it samples the next point based on the history of previous points and their evaluations. The sequence κ_n could also be seen as the mathematical representation of the main "`for loop`" that appears in sequential algorithms. Also, this definition is sufficiently general to encompass a wide variety of algorithms, not only those that are explicitly designed as global optimization methods, but also any iterative method that samples points based on previous evaluations, such as bandits, importance sampling, and more.

Link with stochastic processes. The foundational idea upon which this definition builds is that an iterative stochastic global optimization algorithm can be seen as a stochastic process. Indeed, provided the search space Ω is measurable, with $\mathscr{Y}$ being the associated σ-algebra, one could construct the probability measure $\mathbb{P}$ on $\Omega^{\mathbb{N}}$ that describes the law of infinite sequences of points sampled by the algorithm. The iterations of the algorithm is then a collection of random variables, which can be written as

$$\{X_n : \Omega^{\mathbb{N}} \to \Omega \mid n \in \mathbb{N}\},$$

and can be seen as a stochastic process with respect to the probability space $(\Omega^{\mathbb{N}}, \mathscr{Y}^{\otimes\mathbb{N}}, \mathbb{P})$. However, constructing $\mathbb{P}$ (a measure on infinite sequences of Ω) solely from the internal structure of the algorithm (which only provides the distribution of the next point sampled by the algorithm given the previous points and their evaluations) is not straightforward. One way to proceed is to invoke the Ionescu–Tulcea theorem (Tulcea, 1949), which operates on a sequence of probability kernels, exactly the data provided by Definition B.1.16.

Theorem B.1.15 – *Ionescu-Tulcea extension theorem (Tulcea, 1949)*

Let $(\Omega_0, \mathcal{A}_0, P_0)$ be a probability space and $(\Omega_n, \mathcal{A}_n)_{n\in\mathbb{N}}$ a sequence of measurable spaces. For each $n \in \mathbb{N}^*$, let $\kappa_n : \left(\Omega^{n-1}, \mathcal{A}^{n-1}\right) \rightsquigarrow (\Omega_n, \mathcal{A}_n)$ be a probability kernel, where

$$\Omega^n \overset{\text{def}}{=} \prod_{k=0}^{n} \Omega_k \quad \text{and} \quad \mathcal{A}^n \overset{\text{def}}{=} \bigotimes_{k=0}^{n} \mathcal{A}_k.$$

Then, for each $n \in \mathbb{N}$, there exists a probability measure P_n defined on $(\Omega^n, \mathcal{A}^n)$ by the following formula

$$P_n \overset{\text{def}}{=} P_0 \otimes \bigotimes_{k=1}^{n} \kappa_k,$$

and there exists a uniquely defined probability measure $\mathbb{P}$ on $(\Omega^{\mathbb{N}}, \otimes_{k=1}^{\infty} \mathcal{A}_k)$, so that

$$P_n(A) = \mathbb{P}\left(A \times \prod_{k=n+1}^{\infty} \Omega_k\right).$$

Thus, the measure $\mathbb{P}$ has conditional probabilities equal to the stochastic kernels κ.

The Ionescu-Tulcea extension theorem (ITET) and the Kolmogorov extension theorem (KET) (Kolmogorov, 1933) both construct probability measures on infinite product spaces, but differ in three key respects: **i.** the starting objects (transition kernels for ITET, compatible families of finite-dimensional marginals for KET); **ii.** ITET requires no topological assumptions on the underlying space, whereas KET typically assumes a Polish space structure; and **iii.** ITET applies to countable product spaces, while KET accommodates uncountable index sets (even though the Carathéodoy enxtension theorem (Carathéodory, 1918) can be uses alongside ITET, see Kallenberg (2021, Theorem 8.24 & Corollary 8.25)). In our setting, ITET is the appropriate tool: we are given a sequence of transition kernels, we impose no topological structure on the search space Ω (which may, for instance, be discrete), and we seek a measure on the space of infinite sequences of points produced by the algorithm — a countable product space.

In Definition B.1.16 we are given the initial law of X_0 and kernels κ_n defined on $\Omega^{n+1} \times \mathbb{R}^{n+1}$. Theorem B.1.15, however, requires kernels on Ω^{n+1}. To bridge this gap we precompose each κ_n along the evaluation map

$$\begin{aligned} e_V : \Omega^{n+1} &\to \Omega^{n+1} \times \mathbb{R}^{n+1} \\ e_V(\mathsf{C}_n) &= (\mathsf{C}_n, V(\mathsf{C}_n)), \end{aligned}$$

obtaining pullback kernels on Ω^{n+1} which are then used in the Ionescu–Tulcea construction.

Definition B.1.17 – *Pullback kernels on finite sequences*

Given a topological space Ω endowed with a σ-algebra $\mathscr{Y}$, a natural number n, and a measurable function $V : \Omega \to \mathbb{R}$. We define $\kappa_n'^{(V)} : \Omega^{n+1} \rightsquigarrow \Omega$ as the pullback of κ_n along e_V, i.e., for any $x \in \Omega^{n+1}$,

$$\kappa_n'^{(V)}(x, \cdot) \overset{\text{def}}{=} \kappa_n(e_V(x), \cdot).$$

These new kernels enable the construction of probability measures on finite sequences of points sampled by an algorithm driven by a target function V. Indeed, for any $n \in \mathbb{N}$, a probability measure $\mu_n^{(V)}$ (the notation here emphasizes the dependence on V) can be defined on $\left(\Omega^{n+1}, \bigotimes_{k=0}^{n} \mathscr{Y}\right)$ by

$$\mu_n^{(V)} \overset{\text{def}}{=} \nu \otimes \bigotimes_{k=0}^{n-1} \kappa_k'^{(V)}.$$

Moreover, using Theorem B.1.15, we can obtain $\mu_\infty^{(V)}$, which is the probability trajectory measure on infinite sequences of points sampled by the algorithm over the function V such that

$$\forall n \in \mathbb{N}, A \in \bigotimes_{k=0}^{n} \mathscr{Y}, \mu_n^{(V)}(A) = \mu_\infty^{(V)}\left(A \times \prod_{k=n+2}^{\infty} \Omega\right). \tag{60}$$

Therefore, each measure $\mu_n^{(V)}$ is the restriction of $\mu_\infty^{(V)}$ to the first $n+1$ elements of infinite sequences. These measures are fully determined by the initial distribution ν and the kernels $(\kappa_n)_{n\in\mathbb{N}}$ and satisfy two intuitive properties useful for rigorous analysis of global optimization algorithms that we detail in Section B.1.2.

1.1 Instantiation of the framework

To assess the validity of Definition B.1.16, we encompass several well-known algorithms in the literature into this framework.

Decision-based approaches. Recall that Chapter A.2 describes *decision-based approaches* as an initial probability measure μ and a sequence of decision rules D_n. Any decision-based algorithm $\mathscr{A}$ can be easily expressed in the frame of Definition B.1.16

$$\begin{aligned} \nu &\overset{\text{def}}{=} \mu \\ \forall A \in \mathscr{Y}, \kappa_n(e_V(\mathsf{C}_n), A) &\overset{\text{def}}{=} \frac{\int_A D_n(\mathsf{C}_n, V(\mathsf{C}_n), x)\, \mathrm{d}\mu(x)}{\int_\Omega D_n(\mathsf{C}_n, V(\mathsf{C}_n), x)\, \mathrm{d}\mu(x)}. \end{aligned} \tag{61}$$

Note that for κ_n to a Markov kernel, two properties must hold:

i. the integral of $x \mapsto D_n(\mathsf{C}_n, V(\mathsf{C}_n), x)$ over Ω must be strictly positive for the renormalization to be a probability measure, and

ii. the function $\kappa_n(\cdot, A)$ must be measurable.

The first property is not trivial as D_n is expected to concentrate over promising regions of the search space, and thus could have strictly positive values only on zero-measure sets as n grows. The second one depends on the choice of $\mathscr{Y}$, but is verified for all algorithms described in Chapter A.2 where $\Omega \subseteq \mathbb{R}^d$ and $\mathscr{Y}$ is the associated Borel σ-algebra. More broadly, this should hold for any topological search space with its Borel σ-algebra and standard decision rules: only pathological constructions would violate the measurable property of kernels.

CMA-ES. CMA-ES is a population-based method that maintains a Gaussian distribution over $\mathbb{R}^d$, updated at each iteration from the history of evaluated points (see Section A.1.1 for details). To express CMA-ES within the framework of Definition B.1.16, we reformulate the problem at the batch level: instead of sampling individual points in $\mathbb{R}^d$, CMA-ES samples batches of λ points in $\mathbb{R}^{d\times\lambda}$. For a function $V : \mathbb{R}^d \to \mathbb{R}$, we define its batch extension

$$V_\lambda : \mathbb{R}^{d\times\lambda} \to \mathbb{R}^\lambda$$
$$(x_1, ..., x_k) \mapsto (V(x_1), ..., V(x_k)).$$

The initial measure ν is the λ-fold product of multivariate Gaussians with mean $m \in \mathbb{R}^d$ and covariance $C \in \mathbb{S}^d_{++}$ (both algorithm parameters). The transition kernels are

$$\kappa_n(e_V(\mathbf{C}_n)) = \bigotimes_{i=1}^{\lambda} \mathcal{N}(\mu(e_V(\mathbf{C}_n)), \Sigma(e_V(\mathbf{C}_n))),$$

where $\mu(e_V(\mathbf{C}_n)) \in \mathbb{R}^d$ and $\Sigma(e_V(\mathbf{C}_n)) \in \mathbb{S}^d_{++}$ are the mean and covariance of the updated Gaussian distribution at iteration n. These are determined by the algorithm's internal update rules from the full history of previous batches and evaluations, and must not be confused with the empirical mean or covariance of the current batch $\mathbf{C}_n$.

McKean-Vlasov-based approaches. Recall that McKean-Vlasov-based approaches consist in a system of $N+1$ discretized in time McKean-Vlasov stochastic differential equations representing the evolution of $N+1$ particles (see Chapter A.3 for details). To express such algorithms within the framework of Definition B.1.16, we reuse the batch reformulation described for CMA-ES, but with a batch size of $N+1$. The initial measure ν is the $N+1$-fold product of initial distributions of the particles (e.g. uniform over Ω). Concerning the transition kernels, notice that it is entirely determined by the individual transition of each particle. As each particle evolves according to a discretized McKean-Vlasov SDE, the theory of SDEs provides provides the existence of an infinitesimal generator of a special kind of Feller process with the same transition probabilities as the discretized SDE (Chaintron & Diez, 2022; Del Moral, 2004; 2013; Le Gall, 2016). This generator can be used to construct the individual transition kernel of each particle, and the transition kernel of the batch is then the $N+1$-fold product of these individual kernels. Moreover, if the step size of the numerical scheme used to solve the system is fixed (such as in the Unadjusted Langevin algorithm (Durmus & Moulines, 2017)), then the kernels κ_n are the same for any n. For background on Feller processes, their generators, and their connection to SDEs, we refer the reader to Appendix A, where the main

definitions and results are also formalized in LEAN. As this material is not central to the thesis, it is relegated to the appendix.

1.2 Properties of the measures

In this section, we present two intuitive properties of the measures $\left(\mu_n^{(V)}\right)_{n\in\mathbb{N}}$ that arise from our framework. They are expected from the way stochastic iterative global optimization algorithms operate, yet their formal proofs require careful measure-theoretic reasoning. They act as sanity checks for the soundness of our framework, confirming that it aligns with intuitive expectations about algorithm behavior, and they also serve as exemplars of reasoning within it. The properties are needed in the formal proof of Malherbe & Vayatis (2017a, Proposition 3).

1.2.1 Equality of restricted measures

Given two functions $f, g : \Omega \to \mathbb{R}$ that are equal on a set $X \subseteq \Omega$, a stochastic iterative global optimization algorithm $\mathscr{A}$ will behave identically if restricted to X. More precisely, the measures describing the sequences of points sampled by the algorithm for both functions coincide when we only consider sequences that lie entirely within X. This property is stated in the following theorem (see Figure B.1.30 for an illustration).

Theorem B.1.16 – *Equality of restricted measures*

Let $\mathscr{A}$ be a stochastic iterative global optimization algorithm as in Definition B.1.16, $X \subseteq \Omega$ a measurable set, $n \in \mathbb{N}$, and $f, g : \Omega \to \mathbb{R}$ two measurable functions such that $f(x) = g(x)$ for any $x \in X$. The following holds

$$\mu_n^{(f)}|_{X^{n+1}} = \mu_n^{(g)}|_{X^{n+1}}.$$

Proof.

$\boldsymbol{n = 0}$. Trivially, $\mu_0^{(f)} = \nu = \mu_0^{(g)}$.

$\boldsymbol{n \geq 1}$. By the Sierpiński–Dynkin π-λ theorem, it suffices to check equality on measurable rectangles $\prod_{i=0}^{n} B_i$ where (B_i) are measurable subsets of Ω. Define $C_i \overset{\text{def}}{=} B_i \cap X$ for each i. For both measures, we have

$$\begin{aligned}\mu_n^{(f)}\left(\prod_{i=0}^{n} B_i \cap X\right) &= \left(\nu \otimes \bigotimes_{k=0}^{n-1} \kappa_k'^{(f)}\right)\left(\prod_{i=0}^{n} C_i\right) \\ &= \int_{x\in C_0} \bigotimes_{k=0}^{n-1} \kappa_k'^{(f)}\left(x, \left\{(y_i)_{0<i\leq n} \;\middle|\; (x,y) \in \prod_{i=0}^{n} C_i\right\}\right) \mathrm{d}\nu(x),\end{aligned}$$

with the second equality holding because the integral over $x \notin C_0$ vanishes (yielding empty inner sets). The same applies to $\mu_n^{(g)}$.

Let $A_n \overset{\text{def}}{=} \prod_{i=1}^{n} C_i$, so that $\left\{(y_i)_{0<i\leq n} \;\middle|\; (x,y) \in \prod_{i=0}^{n} C_i\right\} = A_n$ for $x \in C_0$. It remains to show

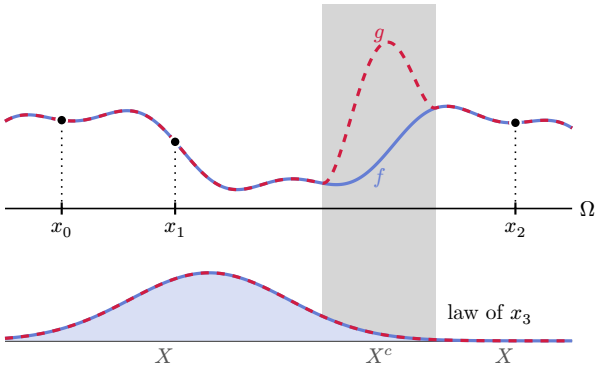


Figure B.1.30. *Illustration of Theorem B.1.16 on one iteration.* ***Top.*** *The functions f (blue) and g (red, dashed) are equal on X, i.e. outside the gray band X^c, where g has a bump. The points x_0, x_1 and x_2 lie in X, and they get the same values on f and on g.* ***Bottom.*** *The law of the next point x_3 only depends on these points and on their values. Thus, it is the same on f and on g (blue area, red dashes on its outline), including its mass in X^c.*

$$\left(\bigotimes_{k=0}^{n-1} \kappa_k'^{(f)}(x,\cdot)\right)\Big|_{A_n} = \left(\bigotimes_{k=0}^{n-1} \kappa_k'^{(g)}(x,\cdot)\right)\Big|_{A_n} \tag{62}$$

for all $x \in C_0$, which implies the desired equality.

We proceed by induction. For $n = 1$: Since $C_0 \subseteq X$ and f, g agree on X, we have $f(x) = g(x)$ for $x \in C_0$, so $\kappa_0'^{(f)}(x,\cdot)|_{A_1} = \kappa_0'^{(g)}(x,\cdot)|_{A_1}$.

For the inductive step: assume the property holds for some $n \in \mathbb{N}$. For any measurable set $B \subseteq \Omega^{n+1}$, we have

$$\begin{aligned}\left(\bigotimes_{k=0}^{n} \kappa_k'^{(f)}(x,B)\right)\Big|_{A_{n+1}} \\ = \int_{y\in A_n} \kappa_n'^{(f)}((x,y),\{z \mid (y,z)\in B\})|_{C_{n+1}}\, \mathrm{d}\left(\bigotimes_{k=0}^{n-1} \kappa_k'^{(g)}(x,y)\right)\Big|_{A_n}\end{aligned}$$

by the induction hypothesis. Since $x \in C_0$ and $y \in A_n$ imply $(x,y) \in X^{n+1}$, we have $f((x,y)) = g((x,y))$, yielding equality of the kernels $\kappa_n'^{(f)}$ and $\kappa_n'^{(g)}$. This concludes the induction.

□

1.2.2 Monotonicity by truncation

A fundamental property ensures coherence when comparing events across different time horizons: if a condition on the first m algorithm iterations logically implies a condition on the first n iterations (with $n \leq m$), then the probability of the former cannot exceed the probability of the latter. This monotonicity in probability, formalized below, reflects the intuitive principle that stricter constraints cannot have larger probability mass.

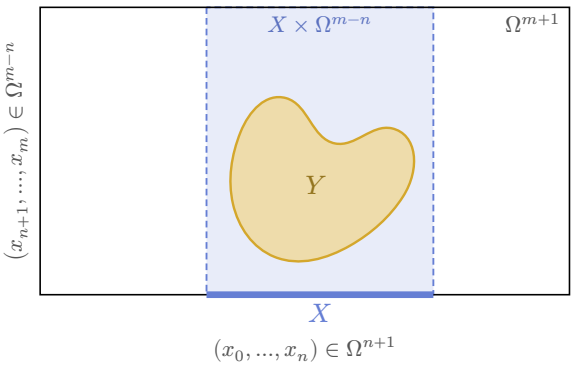


Figure B.1.31. *Sequences of length $m+1$, represented in $\Omega^{m+1} = \Omega^{n+1} \times \Omega^{m-n}$. The horizontal axis holds the first $n+1$ elements of a sequence and the vertical axis its last $m-n$ elements. The sequences whose first $n+1$ elements lie in X form the blue band $X \times \Omega^{m-n}$. Using Eq. 60, its mass is $\mu_m^{(f)}(X \times \Omega^{m-n}) = \mu_n^{(f)}(X)$. Any set Y (in yellow) satisfying the hypothesis of Theorem B.1.17 lies in this band. Thus, $\mu_m^{(f)}(Y) \leq \mu_n^{(f)}(X)$. This illustrates Theorem B.1.17.*

Theorem B.1.17 – *Monotonicity by sequence truncation*

Let $\mathscr{A}$ be a stochastic iterative global optimization algorithm as in Definition B.1.16, $f : \Omega \to \mathbb{R}$ a measurable function, $n, m \in \mathbb{N}$ such that $n \leq m$, a measurable set $X \subseteq \Omega^{n+1}$, and a measurable set $Y \subseteq \Omega^{m+1}$. The following holds

$$Y \subseteq \left\{ (x_i)_{0 \leq i \leq m} \;\middle|\; (x_i)_{0 \leq i \leq n} \in X \right\} \Rightarrow \mu_m^{(f)}(Y) \leq \mu_n^{(f)}(X).$$

Proof. The proof is trivial using Eq. 60. We have that

$$\begin{aligned} \mu_m^{(f)}(Y) &= \mu_\infty^{(f)}\left(Y \times \prod_{k=m+1}^{\infty} \Omega \right) \\ &= \mu_\infty^{(f)}\left(\left\{ (x_i)_{i \in \mathbb{N}} \;\middle|\; (x_i)_{0 \leq i \leq m} \in Y \right\} \right). \end{aligned}$$

Moreover, for any $x \in \Omega^{\mathbb{N}}$ such that $(x_i)_{0 \leq i \leq m} \in Y$, we have by hypothesis that $(x_i)_{0 \leq i \leq n} \in X$. Thus,

$$\left\{ (x_i)_{i \in \mathbb{N}} \;\middle|\; (x_i)_{0 \leq i \leq m} \in Y \right\} \subseteq \left\{ (x_i)_{i \in \mathbb{N}} \;\middle|\; (x_i)_{0 \leq i \leq n} \in X \right\},$$

and therefore

$$\begin{aligned} \mu_\infty^{(f)}\left(\left\{ (x_i)_{i \in \mathbb{N}} \;\middle|\; (x_i)_{0 \leq i \leq m} \in Y \right\} \right) &\leq \mu_\infty^{(f)}\left(\left\{ (x_i)_{i \in \mathbb{N}} \;\middle|\; (x_i)_{0 \leq i \leq n} \in X \right\} \right) \\ &= \mu_n^{(f)}(X). \end{aligned}$$

□

Now that we have established these properties, we can use our framework to prove general results about stochastic iterative global optimization algorithms. For instance, we can give a rigorous proof of Malherbe & Vayatis (2017a, Proposition 3), which states that for an algorithm to be consistent on any Lispchitz function needs to sample the whole search space.

Proposition B.1.2 – *Malherbe & Vayatis (2017a, Proposition 3)*

Let $\mathscr{A}$ be a stochastic iterative global optimization algorithm and Ω a compact measurable metric space. The sequence $(X_0, ..., X_n)$ denotes a sequence of $n+1$ points sampled by $\mathscr{A}$ over a function V. The following statements are equivalent

i. For any Lipschitz continuous function V defined on Ω,

$$\sup_{x\in\Omega} \min_{i=0...n} \text{dist}\,(X_i, x) \xrightarrow{\mathbb{P}} 0.$$

ii. For any Lipschitz continuous function V defined on Ω,

$$\max_{i=0...n} V(X_i) \xrightarrow{\mathbb{P}} \max_{x\in\Omega} V(x).$$

Statement **ii.** is the standard consistency definition for global optimization, while **i.** asserts that any point in the search space can be approximated by algorithm samples. Though conceptually intuitive—Lipschitz regularity allows one to construct degenerate functions concentrated over arbitrarily small regions—proving the equivalence rigorously is non-trivial: the result applies to *any* algorithm, requiring a formal definition like Definition B.1.16. The proof further relies on the measure properties of Section B.1.2. While Malherbe & Vayatis (2017a)'s proof ideas is sound in spirit, it remains incomplete: lacking a rigorous algorithm definition, the measure properties could only be assumed, not formally derived. Our framework closes these gaps, enabling a complete justification of Malherbe & Vayatis (2017a)'s argument grounded in formally verified results.

1.3 Sketch proof of Proposition B.1.2

In this section, we briefly discuss the strategy of the proof of Proposition B.1.2 developed in the preprint (Malherbe & Vayatis, 2017b) of the original paper (Malherbe & Vayatis, 2017a), considering that stochastic and iterative global optimization algorithms are defined as in Definition B.1.16. We do not detail the entire proof, which is already well explained in that paper, but rather highlight the key steps where the framework developed in the present work is essential.

Modus ponens. The right implication of Proposition B.1.2 is quite intuitive. Indeed, if the algorithm samples the search space, then it means that, for any point in this space, the probability for a point sampled by the algorithm to not be arbitrarily close to this point goes to 0. Therefore, for any $\varepsilon > 0$, it suffices to select a $\delta > 0$ such that, for any point x in the search space such that the distance between x and the maximizer of V (noted $x^\star$) is lower than δ, then $V(x^\star) - V(x) < \varepsilon$. This δ can be obtained using the continuity of V. Thus, for any $n \in \mathbb{N}$, we have the following set inclusion

$$\left\{(X_i)_{0\le i\le n} \;\middle|\; V(x^\star) - \max_{i=0...n} V(X_i) > \varepsilon\right\} \subseteq \left\{(X_i)_{0\le i\le n} \;\middle|\; \sup_{x\in\Omega} \min_{i=0...n} \text{dist}\,(X_i, x) > \delta\right\}.$$

This allows us to show that, for any $n \in \mathbb{N}$,

$$\mu_n^{(V)}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; V(x^\star) - \max_{i=0\dots n} V(X_i) > \varepsilon\right\}\right) <$$
$$\mu_n^{(V)}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; \sup_{x\in\Omega}\min_{i=0\dots n} \text{dist }(X_i, x) > \delta\right\}\right),$$

by the monotonicity of measures. Then, we can conclude that the left-hand side also goes to 0 as n goes to infinity using the squeeze theorem, as we know by hypothesis that

$$\mu_n^{(V)}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; \sup_{x\in\Omega}\min_{i=0\dots n} \text{dist }(X_i, x) > \delta\right\}\right) \xrightarrow[n\to\infty]{} 0.$$

Reverse modus ponens. The left implication is more involved and is proven by contradiction, i.e., we prove that the facts that the algorithm is consistent on any Lipschitz function and that there exists a Lipschitz function f and $\varepsilon_1, \varepsilon_2 > 0$ such that

$$\forall N \in \mathbb{N}, \exists n \geq N, \varepsilon_2 < \mu_n^{(f)}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; \sup_{x\in\Omega}\min_{i=0\dots n} \text{dist }(X_i, x) > \varepsilon_1\right\}\right)$$

are incompatible.

First, we use the fact that Ω is a compact space to construct an $(\varepsilon_1/2)$-cover of Ω, i.e., a finite collection of points $\left\{c_1, ..., c_{N_1}\right\}$ in Ω such that, $\bigcup_{i=1\dots N_1} B(c_i, \varepsilon_1/2) = \Omega$.

Next, we use the contradiction hypothesis, Theorem B.1.17, and the monotonicity of measures to show that there exists a $c^\star \in \left\{c_1, ..., c_{N_1}\right\}$ such that, for any $n \in \mathbb{N}$,

$$0 < \frac{\varepsilon_2}{2N_1} < \mu_n^{(f)}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; \forall 0 \leq i \leq n, X_i \notin B(c^\star, \varepsilon_1/2)\right\}\right). \tag{63}$$

Proof. To do so, we again reason by contradiction, assuming that for any c_i in the cover, there exists an $n_i \in \mathbb{N}$ such that

$$\mu_{n_i}^{(f)}\left(\left\{(X_j)_{0\leq j\leq n_i} \;\middle|\; \forall 0 \leq j \leq n_i, X_j \notin B(c_i, \varepsilon_1/2)\right\}\right) < \frac{\varepsilon_2}{2N_1}. \tag{64}$$

Then, we select $n^\star$, the maximum of all these n_i's.

As, for any c_i, there exists a $n_i < n^\star$ such that Eq. 64 and the following set inclusion hold

$$\left\{(X_j)_{0\leq j\leq n^\star} \mid \forall 0 \leq j \leq n^\star, X_j \notin B(c_i, \varepsilon_1/2)\right\} \subseteq$$
$$\left\{(X_j)_{0\leq j\leq n_i} \mid \forall 0 \leq j \leq n_i, X_j \notin B(c_i, \varepsilon_1/2)\right\},$$

we can use Theorem B.1.17 to show that, for any c_i in the cover,

$$\mu_{n^\star}^{(f)}\left(\left\{(X_j)_{0\leq j\leq n^\star} \mid \forall 0 \leq j \leq n^\star, X_j \notin B(c_i, \varepsilon_1/2)\right\}\right) < \frac{\varepsilon_2}{2N_1}. \tag{65}$$

Finally, showing that

$$\left\{(X_i)_{0\leq i\leq n^\star} \;\middle|\; \sup_{x\in\Omega}\min_{i=0\dots n^\star} \text{dist }(X_i,x) > \varepsilon_1\right\} \subseteq \bigcup_{i=1\dots N_1}\left\{(X_j)_{0\leq j\leq n^\star} \mid \forall 0\leq j\leq n^\star, X_j\notin B(c_i,\varepsilon_1/2)\right\}$$

and using Eq. 65 with the σ-additivity of measures, we obtain the contradiction

$$\mu_{n^\star}^{(f)}\left(\left\{(X_i)_{0\leq i\leq n^\star} \;\middle|\; \sup_{x\in\Omega}\min_{i=0\dots n^\star} \text{dist }(X_i,x) > \varepsilon_1\right\}\right) < \varepsilon_2.$$

□

Next, we construct a Lipschitz function $\tilde{f}$ that is indistinguishable from f outside the ball $B(c^\star,\varepsilon_1/2)$ and such that its maximum is strictly greater than the maximum of f over Ω. This function is constructed as follows

$$\tilde{f}(x) = \begin{cases} f(x) + 2\left(1-\frac{\text{dist }(c^\star,x)}{\varepsilon_1/2}\right)\times\left(\max_{x\in\Omega} f(x) - \min_{x\in\Omega} f(x) + 1\right) & \text{if } \; x\in B(c^\star,\varepsilon_1/2) \\ f(x) & \text{otherwise.} \end{cases}$$

Note that our expression for $\tilde{f}$ is slightly simpler than the one in Malherbe & Vayatis (2017a), while serving the same purpose, and it also corrects an issue in the original proof, which fails when f is constant. A visual representation of this function is given in Figure B.1.32.

Finally, let $\delta \overset{\text{def}}{=} \frac{1}{2}\left(\max_{x\in\Omega}\tilde{f}(x) - \max_{x\in\Omega} f(x)\right) > 0$. As $\tilde{f}$ is Lipschitz continuous, the algorithm is consistent on it by hypothesis

$$\mu_n^{(\tilde{f})}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; \max_{x\in\Omega}\tilde{f}(x) - \max_{i=0\dots n}\tilde{f}(X_i) > \delta\right\}\right) \xrightarrow[n\to\infty]{} 0.$$

However, by construction of $\tilde{f}$, for any sequence of points $(X_i)_{0\leq i\leq n}$ such that, for any $0\leq i\leq n$, $X_i\notin B(c^\star,\varepsilon_1/2)$ implies that

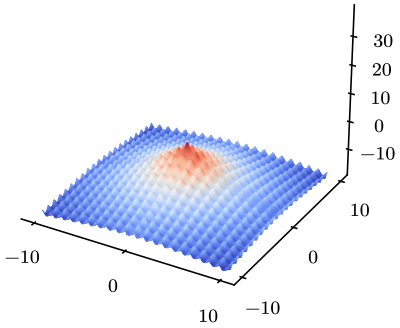
$$\max_{x\in\Omega}\tilde{f}(x) - \max_{i=0\dots n}\tilde{f}(X_i) > \delta.$$

Therefore, using Eq. 63 and Theorem B.1.16, we have that

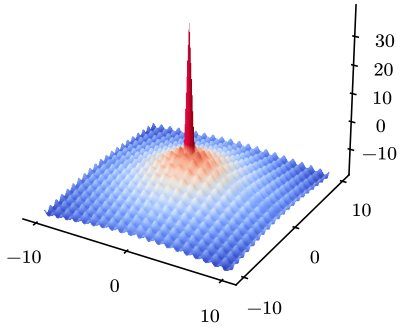
$$0 < \frac{\varepsilon_2}{2N_1} < \mu_n^{(\tilde{f})}\left(\left\{(X_i)_{0\leq i\leq n} \;\middle|\; \max_{x\in\Omega}\tilde{f}(x) - \max_{i=0\dots n}\tilde{f}(X_i) > \delta\right\}\right),$$

which contradicts the fact that the algorithm is consistent on $\tilde{f}$.

The central role Definition B.1.16 is evident in this proof, as Theorem B.1.16 and Theorem B.1.17 are indispensable tools for the rigourous verification of this result.



(a) The reversed Ackley function.



(b) $\tilde{f}$

Figure B.1.32. *Visualization of* $\tilde{f}$ *when* f *is the reversed Ackley function on* $[-10, 10]^2$ *with* $c \stackrel{\text{def}}{=} (0, 0)^\top$ *and* $\varepsilon_1 \stackrel{\text{def}}{=} 1$.

2 — Introduction to formal methods

Abstract. We give a brief introduction to formal methods, with a focus on the LEAN proof assistant, which we use thoroughly in this thesis. We present how probability theory is formalized in LEAN, and we give an overview of some contributions of this thesis in the formalization of Markov kernels and their categorical structure, which are key mathematical objects in our framework for stochastic iterative global optimization algorithms. Most of these contributions has been merged into the `Mathlib` library, a high quality, community-driven, general-purpose library of formalized mathematics in LEAN, demonstrating the relevance of our work for the broader mathematical community and its potential to facilitate future formalization efforts in optimization and related fields.

A key motivation for developing the abstract framework presented in the previous sections is its amenability to formal verification. The framework's mathematical structure — grounded in probability kernels and the Ionescu–Tulcea theorem — translates naturally into proof assistant formalism, enabling mechanized verification of both theoretical results and algorithm correctness.

Proof assistants and formal verification. Proof assistants are software systems designed to formalize and mechanically verify mathematical proofs. They are based on extensions of the Calculus of Constructions (Coquand & Huet, 1988), in particular the Calculus of Inductive Constructions (CIC) (Paulin-Mohring, 2015), in which propositions are interpreted as types and proofs as terms.

A proof assistant can be viewed as an interface through which users express formal statements and derivations, which are then checked by a small trusted core component known as the kernel. The kernel implements a minimal set of typing and reduction rules derived from the underlying type theory, and is responsible for ensuring the correctness of all accepted proofs. Figure B.2.33 illustrates the general architecture of a proof assistant. The kernel is intentionally kept as small and simple as possible, in order to facilitate auditing and increase confidence in its correctness. In contrast, the user interface provides a higher-level and more expressive environ-

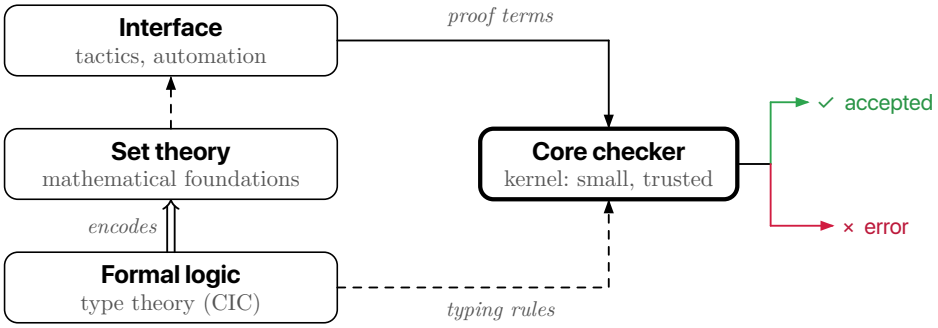


Figure B.2.33. *Architecture of a proof assistant. The core checker (or kernel) is a small, trusted component that verifies the correctness of proofs based on the rules of the underlying type theory. The interface provides a user-friendly environment for writing and checking proofs, often with support for tactics and automation.*

ment, supporting tactics and automation mechanisms that significantly reduce the burden of formal development.

This design provides a strong notion of assurance: any proof accepted by a proof assistant is mechanically certified to be correct, under the assumption of the soundness of its kernel.

Over the past decade, proof assistants have achieved remarkable successes in foundational mathematics, including the formalization of the four-color theorem (Gonthier & others, 2008) and Feit-Thompson theorem (Gonthier et al., 2013) (in `Rocq` (The Coq Development Team, 2024)), the Kepler conjecture (Hales et al., 2017) (in `Isabelle/HOL` (Nipkow et al., 2002)), the sphere eversion (Doorn et al., 2023) and continuum hypothesis independence (Han & Doorn, 2020) (in LEAN (Moura & Ullrich, 2021)), and others (Barco et al., 2026; Bobbin et al., 2024; Wieser & Song, 2022). More recently, LEAN has gained prominence due to its user-friendly interface and the comprehensive `Mathlib` (The mathlib Community, 2020) library, which covers measure theory, topology, and real analysis. Influential mathematicians like Terence Tao or William Timothy Gowers have embraced LEAN for formalizing research (Gowers et al., 2025; Tao, 2022), documenting their experience online[13,14]. Tao has even advocated for a synergistic role between proof assistants and machine learning algorithms[15]: formal verification can ensure correctness of LLM-assisted proofs, while LLM can suggest proof strategies and accelerate formalization.

Why formalization matters for applied fields. Despite these successes in foundational mathematics, formal verification remains largely absent from applied areas such as optimization; a striking gap given that applied fields have the most to gain. In applied mathematics, where algorithmic performance and empirical results dominate, mathematical proofs are often treated as secondary. This creates a dangerous situation: undetected errors in algorithm analyses can invalidate entire research

[13] terrytao.wordpress.com/2023/11/18/formalizing-the-proof-of-pfr-in-lean4-using-blueprint-a-short-tour

[14] terrytao.wordpress.com/2025/05/31/a-lean-companion-to-analysis-i

[15] terrytao.wordpress.com/wp-content/uploads/2024/03/machine-assisted-proof-notices.pdf

programs, yet proofs rarely receive the rigorous peer review standard in pure mathematics. Formal verification offers a principled solution to this gap, providing mechanized validation where traditional review processes fall short. For optimization algorithms in particular, the ability to formally verify correctness guarantees for convergence, complexity bounds, or algorithm invariants could transform how the field builds and validates new methods.

Towards formalization in LEAN. Definition B.1.16 is designed to be formalized. Measures and Markov kernels, on which it relies, are formalized in `Mathlib`, as is the Ionescu-Tulcea theorem (Marion, 2026). Thus, the definition translates directly into LEAN. Its properties, and its instances for specific algorithms, can then be verified mechanically. In Chapter B.3, we give a complete LEAN formalization of Definition B.1.16, together with a formalization of Proposition B.1.2. We also instantiate the definition for LIPO, RANKOPT and CMA-ES, following the constructions of Section B.1.1. As explained in Chapter B.1, this formalization follows a hierarchy. Families of algorithms are defined once, as special cases of the general definition. In LEAN, the general definition is a type, and a theorem stated for every object of this type holds for each family and each algorithm without a new proof.

Before formalizing global optimization algorithms, we introduce the LEAN theorem prover. We then present how measure theory is formalized in LEAN, together with our contributions on Markov kernels and their categorical structure (see Section B.2.2).

2.1 LEAN theorem prover

LEAN (Moura & Ullrich, 2021) is an open-source proof assistant based on dependent type theory. It is a programming language with a strong type system that allows users to define mathematical objects, state theorems, and construct proofs in a formal language that can be checked by a computer. It is based on the *calculus of inductive constructions* (Coquand & Huet, 1988), which consider propositions as types and proofs as terms inhabiting those types. Thus, proving a statement in LEAN reduces to constructing a term of the corresponding type. To help users constructing such terms, LEAN provides a "tactic mode", where users can apply high-level commands (called tactics) to manipulate the current proof state. Tactics often correspond to common proof techniques in mathematics, such as induction, case analysis, or contradiction. To prove complex mathematical statements, users often combine LEAN with the `Mathlib` library (The mathlib Community, 2020), which provides a comprehensive collection of definitions, theorems, and proofs in various areas of mathematics, including real analysis, measure theory, and topology.

To give a sense of how LEAN proofs look like, we provide a simple example below. The following LEAN code states and proves that the function $x : \mathbb{R} \mapsto x + 1$ is strictly monotone.

```
import Mathlib

example : StrictMono (fun (x : ℝ) ↦ x + 1) := by
  intro x y hxy
  /-
    x y : ℝ
    hxy : x < y
    ⊢ x + 1 < y + 1
  -/
  exact (add_lt_add_iff_right 1).mpr hxy
```

Let us briefly explain this code. The first line uses the `import` keyword to import the definitions of real numbers and of strict monotonicity from `Mathlib`. The third line states the example, where `StrictMono` is the proposition that a function is strictly monotone. The fourth line begins the proof, where `intro` is a tactic that introduces two arbitrary real numbers x and y such that $x < y$. The proof state is then updated to show that $x + 1 < y + 1$. The fourth line uses a lemma from `Mathlib` that states that, for any real numbers x, y, a, $x + a < y + a \iff x < y$.

In order to states proposition, one need to define the mathematical objects involved. There is multiple ways to do so in LEAN. For instance, one could use the `def` keyword to define a function, a type or a predicate. This is the case for the `StrictMono` predicate

```
/-- A function `f` is strictly monotone if `a < b` implies
  `f a < f b`. -/
def StrictMono (f : α → β) : Prop :=
  ∀ ⦃a b⦄, a < b → f a < f b
```

or for the definition of the lower Lebesgue integral

```
/-- The **lower Lebesgue integral** of a function `f` with
respect to a measure `μ`. -/
def lintegral (μ : Measure α) (f : α → ℝ≥0∞) : ℝ≥0∞ :=
  ⨆ (g : α →ₛ ℝ≥0∞) (_ : ⇑g ≤ f), g.lintegral μ
```

More sophisticated mathematical structures are formalized through the `structure` and `class` keywords. These constructs are introduced in the following sections where we present the LEAN formalization of measurable spaces, measures, and Markov kernels.

2.2 Probability theory in LEAN

The fundamental objetcs of probability theory — or more generally of measure theory — are measurable spaces, measures, and Markov kernels. We dedicate an entire section to the formalization of the latter, as they are central to the framework presented in Definition B.1.16 and as we present original contributions in this area.

For the formalization of measurable spaces and measures, we refer to the `Mathlib` library (The mathlib Community, 2020), which provides comprehensive definitions and results in these areas.

Measurable space. A measurable space is a set X equipped with a σ-algebra $\mathscr{X}$, which is a collection of subsets of X that includes the empty set, is closed under complements, and is closed under countable unions. In `Mathlib`, a measurable space is defined as a *type class*, which is a way to attach additional objects and properties to a type or an object. The `Mathlib` definition of a measurable space is as follows

LƎ∀N

```
class MeasurableSpace (X : Type*) where
  /-- Predicate saying that a given set is measurable.
  For a set X ⊆ X, `MeasurableSet' X` is the proposition
  "X is measurable" -/
  MeasurableSet' : Set X → Prop
  /-- The empty set is a measurable set. -/
  measurableSet_empty : MeasurableSet' ∅
  /-- The complement of a measurable set is a measurable
  set. -/
  measurableSet_compl : ∀ s,
    MeasurableSet' s → MeasurableSet' sᶜ
  /-- The union of a sequence of measurable sets is
  a measurable set. -/
  measurableSet_iUnion : ∀ (f : ℕ → Set X),
    (∀ i, MeasurableSet' (f i)) →
      MeasurableSet' (⋃ i, f i)
```

This type class attaches to a type `X` — a set in classical mathematics — a σ-algebra by providing the `MeasurableSet'` predicate that determines whether a subset of X is measurable. It also provides the necessary properties to ensure that `MeasurableSet'` defines a σ-algebra.

The type class approach exploits LƎ∀N's *type class inference* system, which automatically resolves instances as needed. This enables flexible use of measure-theoretic objects as users need not explicitly specify the measurable space structure, as long as a `MeasurableSpace` instance is available for their type.

Measure. A measure on a measurable space $(X, \mathscr{X})$ is a function $\mu : \mathscr{X} \to \overline{\mathbb{R}}_+$ that assigns a non-negative extended real number to each measurable set, satisfying the properties of non-negativity, null empty set, and countable additivity. In `Mathlib`, a measure is defined as a *structure*, a type that bundles together data and properties (such as a function and its properties), and can inherit from other structures. A structure is conceptually similar to a class in object-oriented programming. The `Mathlib` definition of a measure is rather involved: it inherits from the `OuterMeasure` structure, which is a function from sets to $\overline{\mathbb{R}}_+$ that sends $\varnothing$ to 0 and is countably subadditive, and it includes the property of countable additivity on measurable sets. The `Mathlib` definition of a measure is as follows

LƎ∀N

```
structure Measure (X : Type*) [MeasurableSpace X]
    extends OuterMeasure X where
```

```
  -- σ-additivity
  m_iUnion {f : ℕ → Set X} : (∀ i, MeasurableSet (f i)) →
      Pairwise (Disjoint on f) →
        toOuterMeasure (⋃ i, f i) =
          ∑' i, toOuterMeasure (f i)
  /-- The outer measure is the canonical extension of its
  restriction to measurable sets -/
  trim_le : toOuterMeasure.trim ≤ toOuterMeasure
```

This structure defines a new type "`Measure X`" for measures on a measurable space X. It can be used to create an object μ that represents a measure

```
variable {X : Type*} [MeasurableSpace X] (μ : Measure X)
```

To specify that μ is a probability measure, we use the `IsProbabilityMeasure` type class, which requires the measure of the entire space (denoted `univ`) to equal 1

```
/-- A measure `μ` is called a probability measure if
`μ univ = 1`. -/
class IsProbabilityMeasure (μ : Measure X) : Prop where
  measure_univ : μ univ = 1
```

To define a probability measure μ on a measurable space X, one can write

```
variable {X : Type*} [MeasurableSpace X] (μ : Measure X)
  [IsProbabilityMeasure μ]
```

A typical consequence of the type class inference system is the automatic resolution of instances when applying lemmas. Consider the following lemma, which states that a probability measure is nonzero

```
lemma ne_zero (μ : Measure X) [IsProbabilityMeasure μ] :
    μ ≠ 0 := ...
```

Applying this lemma to the measure μ defined above yields

```
#check ne_zero μ
```

```
ne_zero μ : μ ≠ 0
```

The proof `ne_zero μ` establishes `μ ≠ 0` through automatic instance resolution: the `IsProbabilityMeasure μ` instance is synthesized without explicit specification.

Conversely, removing the `IsProbabilityMeasure` instance from the context produces

```
failed to synthesize instance of type class
  IsProbabilityMeasure μ
```

This error indicates that the type class inference system cannot locate an instance of `IsProbabilityMeasure μ`, preventing application of the `ne_zero` lemma.

Moreover, since every probability measure is also a finite measure, LEAN's type class inference system automatically resolves this relationship. Consider the following lemma, which states that for any set A, the measure of A under a finite measure is finite

```
lemma measure_lt_top (μ : Measure X) [IsFiniteMeasure μ]
    (A : Set X) : μ A < ∞ := ...
```

LEAN

Applying this lemma to the measure μ defined above and to an arbitray `(A : Set X)` yields

```
#check measure_lt_top μ A
```

LEAN

```
measure_lt_top μ A : μ A < ∞
```

The proof `measure_lt_top μ A` establishes `μ A < ∞` via automatic instance resolution: the `IsFiniteMeasure μ` instance is synthesized from the `IsProbabilityMeasure μ` instance without explicit specification.

Similarly, removing the `IsProbabilityMeasure` instance from the context produces an identical error.

```
failed to synthesize instance of type class
  IsFiniteMeasure μ
```

2.2.1 Markov kernels in LEAN

A transition kernel $\kappa : X \rightsquigarrow Y$ between measurable spaces $(X, \mathcal{X})$ and $(Y, \mathcal{Y})$ is a function assigning to each point $x \in X$ a measure $\kappa(x)$ on $(Y, \mathcal{Y})$ such that $x \mapsto \kappa(x, A)$ is measurable for every measurable set A in $\mathcal{Y}$. When each $\kappa(x)$ is a probability measure, κ is called a Markov kernel. Markov kernels are central to probability theory, serving as fundamental tools for defining conditional distributions, constructing stochastic processes, and in the context of optimization algorithms, describing algorithmic transitions and sample distributions across iterations. For a comprehensive treatment, see Kallenberg (2021).

The formalization of transition kernels in `Mathlib` is detailed in Degenne (2025). It is defined as a structure that bundles together a function from X to the space of measures on Y and the measurability condition.

```
structure Kernel (X Y : Type*) [MeasurableSpace X]
    [MeasurableSpace X] where
  toFun : X → Measure Y
  measurable' : Measurable toFun
```

LEAN

Note that the measurability condition is expressed as a property of the underlying function itself, which states $\forall \mathcal{M} \in \mathfrak{M}(Y), \kappa^{-1}(\mathcal{M}) \in \mathcal{X}$. This is equivalent to the condition stated before.

Three type classes are then defined to capture various properties of kernels:

- `IsMarkovKernel` captures the property that a kernel is a Markov kernel, i.e., that for any $x \in X$, $\kappa(x)$ is a probability measure on Y

```
class IsMarkovKernel (κ : Kernel X Y) where
  isProbabilityMeasure : ∀ x, IsProbabilityMeasure (κ x)
```

- `IsFiniteKernel` captures the property that a kernel is a finite kernel, i.e., that for any $x \in X$, $\kappa(x)$ is a finite measure on Y with a uniform bound

```
class IsFiniteKernel (κ : Kernel X Y) where
  exists_univ_le : ∃ C : ℝ≥0∞, C < ∞ ∧
    ∀ x, κ x univ ≤ C
```

- `IsSFiniteKernel` captures the property that a kernel is an s-finite kernel, i.e., that κ is a countable sum of finite kernels

```
class IsSFiniteKernel (κ : Kernel X Y) where
  tsum_finite : ∃ κs : ℕ → Kernel X Y,
    (∀ n, IsFiniteKernel (κs n)) ∧ κ = Kernel.sum κs
```

Analogously to the case of measures, defining a Markov kernel in LEAN involves creating an object of type `Kernel` and providing an instance of `IsMarkovKernel` for it. For instance, to define a Markov kernel κ from X to Y, one can write

```
variable {X Y : Type*} [MeasurableSpace X]
    [MeasurableSpace Y] (κ : Kernel X Y)
    [IsMarkovKernel κ]
```

`Mathlib` defines several important specialized Markov kernels. The most pertinent to subsequent sections are:

- the identity kernel $\mathrm{id}_X : X \rightsquigarrow X$ maps each point $x \in X$ to the Dirac measure at x;
- the copy kernel $\Delta_X : X \rightsquigarrow X \times X$ maps each point $x \in X$ to the Dirac measure at (x, x);
- the discard kernel $\varepsilon_X : X \rightsquigarrow \mathbf{1}$ maps each point $x \in X$ to the unique probability measure on the one-point space $\mathbf{1}$;
- the swap kernel $\beta_{XY} : X \times Y \rightsquigarrow Y \times X$ maps each point $(x, t) \in X \times Y$ to the Dirac measure at (t, x).

Several operations on kernels are also defined, such as the composition of kernels (noted $\circ_\kappa$) or the parallel composition (noted $\parallel_\kappa$), which, given two s-finite kernels

$\kappa : X \rightsquigarrow Y$ and $\eta : Z \rightsquigarrow W$, produces a kernel $\kappa \parallel_k \eta : X \times Z \rightsquigarrow Y \times W$ defined by

$$(\kappa \parallel_k \eta)(x,t) = \kappa(x) \otimes \eta(t),$$

where "$\otimes$" is the product of measures. The s-finiteness assumption is essential to ensure that the product measure is a measurable function.

According to Degenne (2025), kernels are central objects in `Mathlib`'s probability theory, essential for defining conditional distributions through disintegration, independence of random variables, sub-Gaussian random variables, and product measures on countable and uncountable product spaces via the Ionescu-Tulcea theorem (Marion, 2026). Contributing to this area of `Mathlib` has significant impact on the formalization of probability theory in LEAN, enabling rigorous foundations for stochastic algorithms and other probabilistic computational methods.

In the next sections, we present original contributions in the formalization of results about Markov kernels in `Mathlib`, which are of independent interest beyond the context of optimization algorithms.

Representation of Markov kernels. Lemma 4.22 of Kallenberg (2021) establishes that any Markov kernel on a standard Borel space admits a representation through a measurable function with randomness isolated to the unit interval.

Lemma B.2.16 – *Kallenberg (2021, Lemma 4.22)*

Let $(X, \mathscr{X})$ and $(Y, \mathscr{Y})$ be measurable spaces with $(Y, \mathscr{Y})$ a standard Borel space. For any Markov kernel $\kappa : X \rightsquigarrow Y$, there exists a measurable function $f : X \times [0,1] \to Y$ such that

$$\forall x \in X, \kappa(x) = \mathcal{U}_{\#} f(x, \cdot),$$

where $\mathcal{U}$ is the uniform distribution on $[0,1]$.

This representation is powerful: for any random variables ξ on X and η on Y, there exists κ such that $\mathbb{P}(\eta \in B \mid \xi = x) = \kappa(x, B)$ for all $x \in X, B \in \mathscr{Y}$. By Lemma B.2.16, there exists a measurable function $f : X \times [0,1] \to Y$ such that $f(x, \vartheta) \sim \kappa(x)$ for $\vartheta \sim \mathcal{U}$. This procedure enables the proof of the transfer lemma (Kallenberg, 2021, Theorem 8.17 & Corollary 8.18) or the Skorohod's representation theorem (Kallenberg, 2021, Theorem 5.31 & Corollary 8.19), which are fundamental results in probability theory.

Proof. Any standard Borel space embeds into the real line, which itself embeds into the unit interval; thus assume without loss of generality that $Y = [0,1]$. Define

$$f(x,y) = \sup\{z \in [0,1] \mid \kappa(x, [0,z]) < y\}, \quad x \in X, y \in [0,1].$$

The function f is measurable on $\mathscr{X} \times \mathfrak{B}_{[0,1]}$ because $\{(x,y) \mid \kappa(x,[0,z]) < y\}$ is measurable for each $z \in [0,1]$, and the set $\{(x,y) \mid z < f(x,y)\}$ can be written as

$$\bigcup_{q \in \mathbb{Q} \cap (z,1]} \{(x,y) \mid \kappa(x,[0,q]) < y\}, \forall z \in [0,1].$$

We get for any $x \in X, w \in [0,1]$,

$$\begin{aligned}\mathcal{U}_{\#} f(x,\cdot)[0,w] &= \mathcal{U}\{y \in [0,1] \mid f(s,y) \leq w\} \\ &= \mathcal{U}\{y \mid \sup\{z \mid \kappa(x,[0,z]) < y\} \leq w\} \\ &= \mathcal{U}\{y \mid t \leq \kappa(x,[0,w])\} \\ &= \kappa(x,[0,w]).\end{aligned}$$

□

We first formalized in LEAN the special case where $Y = [0,1]$ as the following `Mathlib`'s lemma

LEAN
```
lemma exists_measurable_map_eq_unitInterval_aux
    (κ : Kernel X I) [IsMarkovKernel κ] :
    ∃ (f : X → I → I), Measurable (uncurry f) ∧
      ∀ x, volume.map (f x) = κ x := by ...
```

Here, `I` denotes $[0,1]$, `volume` is the uniform distribution on `I`, and `uncurry` transforms a curried function `f : X → I → I` into a function `X × I → I`. The general case for arbitrary standard Borel spaces Y is obtained by composing two measurable embeddings: the `exists_measurableEmbedding_real` theorem establishes that there exists a measurable embedding of Y into $\mathbb{R}$, and the sigmoid function provides a measurable embedding of $\mathbb{R}$ into $[0,1]$ (formalized as `measurableEmbedding_sigmoid` in `Mathlib`). The composition of these embeddings yields the desired result for an arbitrary standard Borel space.

LEAN
```
theorem exists_measurable_map_eq_unitInterval
    (κ : Kernel X Y) [IsMarkovKernel κ] :
    ∃ (f : X → I → Y), Measurable (uncurry f) ∧
      ∀ a, volume.map (f a) = κ a := by ...
```

This formalization effort resulted in several merged pull requests to `Mathlib`: `PR#27512`; `PR#27513`; `PR#27515`; `PR#27517`; `PR#28780`; and `PR#30112`. Beyond their independent interest, such results about Markov kernels are essential for formalizing optimization algorithms: enriching the library's foundations facilitates the formalization of higher-level theoretical results. Moreover, this formalization is used in the *Lean Machine Learning* project[16] to construct a specific probability space used to formalize results on bandits algoritms.

Markov kernels as categories. The formalization of category theory in Mathlib is well established, with extensive structures, type classes, and results covering a broad spectrum of the field (Asgeirsson et al., 2024; Carlier, 2026; Carneiro & Riehl, 2025; Riou, 2025). The library also provides a suite of tactics and automated tools specifically designed for reasoning about morphism composition, including `cat_disch`, `slice`, `coherence`, and `monoidal`. Connecting Markov kernels to category theory is an idea that was present from the very beginning of the formalization of transition kernels in Mathlib (Degenne, 2025), but remained largely unexplored. The benefits

[16] `leanmachinelearning.org`

of establishing such a connection are twofold. On one hand, it would allow one to leverage the powerful categorical machinery to reason about Markov kernels, since viewing them as morphisms in a category naturally structures their composition. On the other hand, it would make use of an underappreciated feature of proof assistants: while in classical mathematics a connection between two objects from distant areas can remain purely conceptual, in a proof assistant such a connection becomes fully concrete, in the sense that the formal tools developed for one object can be directly and mechanically applied to the other.

We contribute to this direction by introducing several tactics that allow one to switch between the different representations of Markov kernels: either as functions from one measurable space to the space of measures on another, or as morphisms in a category. The latter representation is particularly well-suited for reasoning about kernel composition, as it unlocks the `monoidal` tactic to prove equalities that would be substantially harder to establish in the functional representation. To our knowledge, this is an original contribution not only in its results, but in its very concept: we are not aware of any existing work in proof assistants that formally bridges two mathematical objects from distant areas, and builds on that bridge to enable automated reasoning — regardless of which objects are involved. This work also makes substantial use of the cumulative nature of Mathlib as a unified mathematical library, where results from any area of mathematics can be combined within a single coherent framework.

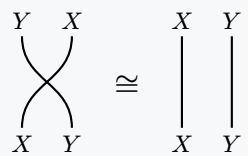
To implement this connection, we first prove that measurable spaces and s-finite kernels form a *copy-discard category*, giving kernels a formal categorical identity as morphisms. We then build a tactic that translates an equality between s-finite kernels into an equality between morphisms in that category, unlocking the full suite of categorical tactics. In particular, `monoidal` can then be applied to prove equalities of composed kernels that would be tedious to establish in the functional representation alone.

Copy-discard category.

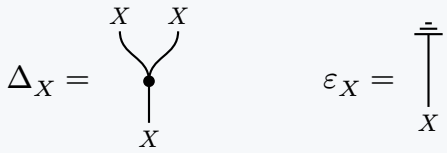
Definition B.2.18 – *Copy-discard category*

An object C is a *copy-discard category* if it has a tensor product "$\otimes$", a monoidal unit "$\mathbf{1}$", and if, for any object $X \in C$

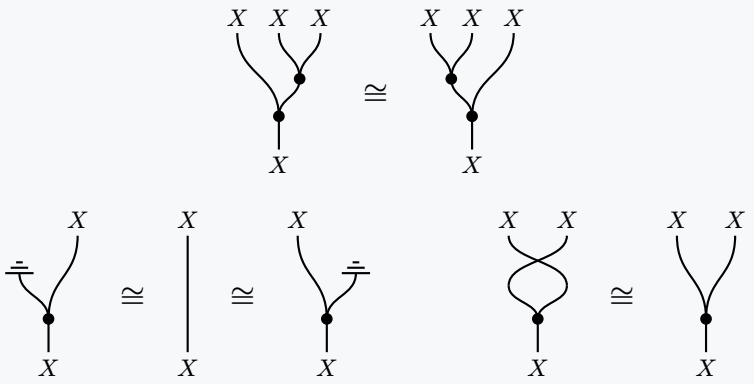
- **i.** $X \otimes \mathbf{1} \cong X$ and $\mathbf{1} \otimes X \cong X$;
- **ii.** for any $Y, Z \in C$, $X \otimes (Y \otimes Z) \cong (X \otimes Y) \otimes Z$;
- **iii.** for any $Y \in C$, there is an *braiding* isomorphism $\beta_{XY} : X \otimes Y \cong Y \otimes X$, depicted in string diagrams as



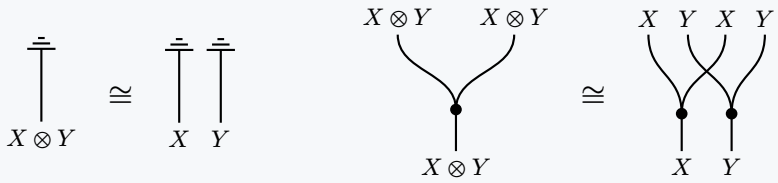
- **iv.** X is equipped with a comultiplication morphism $\Delta_X : X \to X \otimes X$ and a counit morphism $\varepsilon_X : X \to \mathbf{1}$, depicted in string diagrams as



v. it satisfies the *commutative comonoid equations*, depicted in string diagrams as



vi. for any $Y \in C$, it is compatible with the *monoidal structure*, depicted in string diagrams as



The category of measurable spaces and s-finite kernels as morphisms is known to constitute a copy-discard category (Cho & Jacobs, 2019; Fritz, 2020; Moss & Perrone, 2023), which is formally characterized by:

- the cartesian product serving as the tensor product of the monoidal structure;
- reverse composition of kernels as the composition operation between morphisms;
- parallel composition of kernels as the tensor product operation between morphisms;
- the one-point space $\mathbf{1}$ functioning as the monoidal unit;
- the swap kernel β_{XY} functioning as the braidin morphisms;
- the copy kernel Δ_X functioning as the copying morphism;
- the discard kernel ε_X functioning as the discarding morphism.

When applied to the context of transition kernels, the string diagrams defined in Definition B.2.18 admit a concrete interpretation. For example, the first diagram in the second line of property **v.** establishes that, for any stochastic mechanism represented by a kernel, the operation of copying the random output followed by discarding one copy is equivalent to the identity operation. Similarly, the second diagram shows that copying the random output and subsequently swapping the two copies produces the same effect as copying the random output alone.

We establish the formalization of measurable spaces and s-finite kernels as a copy-discard category within `Mathlib`, providing the essential foundation for connecting Markov kernels to categorical structures.

We first introduce the `SFinKer` structure, which encodes the type of measurable spaces within a fixed universe level.

```
structure SFinKer : Type (u + 1) where
  of ::
  /-- The underlying measurable space. -/
  carrier : Type u
  [str : MeasurableSpace carrier]
```

We defer extensive discussion of universe levels, noting only that they establish type hierarchies in LEAN, where `Type u` constitutes an object of `Type (u + 1)`.

We then introduce the morphisms of the category, packaged in a structure that bundles together the kernel and the requisite s-finiteness property.

```
structure Hom (X Y : SFinKer.{u}) where
  /-- The underlying morphism. -/
  hom : Kernel X Y
  /-- The property that the morphism satisfies. -/
  property : IsSFiniteKernel hom
```

To establish that objects and morphisms form a category, we instantiate a type class on the object type `SFinKer`, which specifies the identity morphism, the composition operation on morphisms, and the associativity of composition.

```
instance : Category SFinKer where
  Hom X Y := Hom X Y
  id X := ⟨Kernel.id, inferInstance⟩
  comp κ η := ⟨η.1 ∘ₖ κ.1, inferInstance⟩
  assoc κ η ξ := by simp [Kernel.comp_assoc]
```

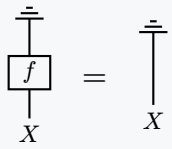
We then establish that this category satisfies the copy-discard category axioms by instantiating the `CopyDiscardCategory` type class, which requires the instantiation of subordinate type classes including `SymmetricCategory` and `MonoidalCategory`. This is accomplished through the specification of the necessary morphisms (including the swap, copy, and discard kernels) and formal verification that they satisfy the required axioms.

```
instance : CopyDiscardCategory SFinKer.{u} where ...
```

In addition, we formalize that the category of measurable spaces and Markov kernels is a Markov category, a special case of copy discard category.

Definition B.2.19 – *Markov category*

An object C is a *Markov category* if it is a copy-discard category and the following string diagram holds



for any morphism $f : X \to Y$.

In the context of Markov kernels, this property states that, for any kernel $\kappa : X \rightsquigarrow Y$, $\varepsilon_X \otimes_k \kappa = \varepsilon_X$. Note that it is only true for Markov kernels, as the measure $\kappa(x)$ is a probability measure:

$$\begin{aligned}
\forall x \in X, A \in \{\varnothing, \{1\}\}, (\varepsilon_X \otimes_k \kappa)(x, A) &= \int_Y \varepsilon_X(y, A)\, \mathrm{d}\kappa(x) \\
&= \int_Y \mathbb{1}_A(1)\, \mathrm{d}\kappa(x) \\
&= \kappa(x, Y) \times \mathbb{1}_A(1) \\
&= \mathbb{1}_A(1) \\
&= \varepsilon_X(x, A).
\end{aligned}$$

As this result holds for Markov kernels — which are special kinds of s-finite kernels — we can define the category of measurable spaces and Markov kernels as a subcategory of `SFinKer`, and we can then prove that this category is a Markov category by instantiating the `MarkovCategory` type class. In the litterature, this category is often referred to as `Stoch`, and it is the canonical example of Markov category.

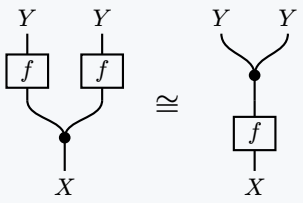
```
1 instance : MarkovCategory Stoch.{u} where
2   discard_natural κ := by ext : 2; simp
```

This formalization effort resulted in a merged pull request to `Mathlib`: PR#36779. It will serve as the foundation of the tactics that allow to switch between the functional and categorical representations of Markov kernels.

Deterministic kernels. Viewing transition kernels as morphisms in a category allows to leverage string diagrams to reason about them. A canonical example is the characterization of deterministic morphisms.

Definition B.2.20 – *Deterministic morphism*

A morphism $f : X \to Y$ in a copy-discard category is *deterministic* if the following string diagram holds



In the context of transition kernels, this property states that a kernel $\kappa : X \rightsquigarrow Y$ is deterministic if and only if copying the random output from κ produces the same distribution as copying the input and then applying κ to both copies. This view lead to a whole new part of transition kernel in `Mathlib`, where we define the type class `IsDeterministic` for kernels, and we prove that, given mild assumptions on $\mathcal{Y}$, a kernel is deterministic if and only if it is of the form $\kappa(x) = \delta_{f(x)}$ for some measurable function $f : X \to Y$. This also lead to several merged pull requests to `Mathlib`: PR#38211; PR#38212; and PR#37851. This characterization is useful in formalizing the deterministic components of sequential algorithms.

Tactics for switching between representations. Having established that measurable spaces and s-finite kernels form a copy-discard category, we exploit this categorical structure to construct tactics that facilitate conversion between the functional and categorical representations of kernels.

We present the `kernel_hom` tactic, which constitutes a formal proof step within the LEAN environment. When the proof goal consists of an equality between s-finite kernels (or their specializations: finite or Markov kernels), the `kernel_hom` tactic systematically translates this equality from the functional representation into an equivalent statement concerning morphisms in the `SFinKer` category. To illustrate this mechanism, consider two s-finite kernels $\kappa : X \rightsquigarrow Y$ and $\eta : W \rightsquigarrow Z$, and the following kernel equality

$$\beta_{YZ} \otimes_k (\kappa \parallel_k \eta) = (\eta \parallel_k \kappa) \otimes_k \beta_{XW}.$$

This result, designated `swap_parallelComp` within `Mathlib`, is formalized in the kernel-function representation through a proof requiring ten lines of code using explicit integral manipulations. Upon application of the `kernel_hom` tactic, this equality is reinterpreted as an equivalence between morphisms in the `SFinKer` category, which can then be proved using the `monoidal` tactic[17].

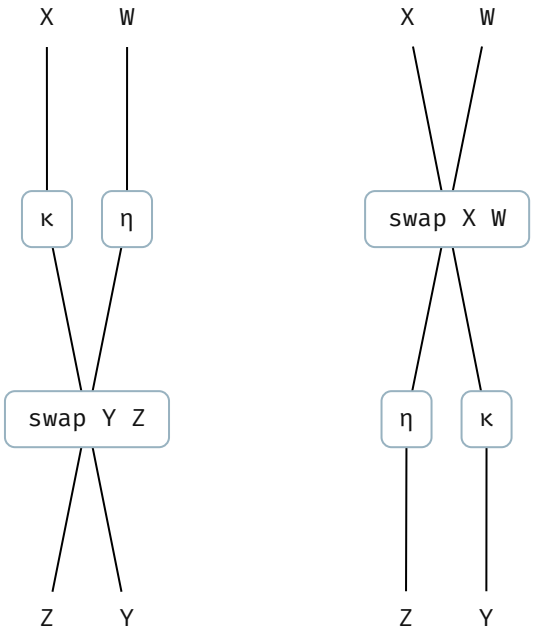
LEAN

```
lemma swap_parallelComp :
    swap Y Z ∘ₖ (κ ∥ₖ η) = η ∥ₖ κ ∘ₖ swap X W := by
  kernel_hom
  /- ⊢ (κ ⊗ₘ η) ≫ (β_ Y Z).hom = (β_ X W).hom ≫ (η ⊗ₘ κ) -/
  monoidal
```

[17] yuma-mizuno.github.io/coherence-tactics

The `kernel_hom` tactic reformulates the proof goal into a logically equivalent statement wherein kernels are represented as morphisms of the category `SFinKer`. This reformulation grants access to the full suite of categorical tactics and proof procedures. We provide multiple convenience wrappers that apply `kernel_hom` automatically, including `kernel_monoidal`, which successively invokes `kernel_hom` followed by `monoidal`, and the `kernel_diagram` command, which renders the string diagram corresponding to a kernel equality directly within the editor interface, to be read from top to bottom

```
1 #kernel_diagram swap_parallelComp
```
LƎ∀N



Although the `kernel_diagram` command cannot directly prove kernel equalities, it provides a visual representation that clarifies the structure of such equalities, thereby guiding the proof strategy. This visualization is particularly valuable for complex equalities involving multiple kernel compositions and parallel applications, where the corresponding string diagram highlights the underlying relationships.

The `kernel_hom` tactic represents a substantial LƎ∀N project, spanning over 1500 lines of code. Its implementation required both an exhaustive characterization of the equivalence between the functional and categorical representations of kernels, and a careful treatment of universe levels — a non-trivial aspect of LƎ∀N's type system. While the latter is arguably the most technically interesting part of the implementation, it falls outside the scope of this thesis; we refer the reader to the project documentation for further details[18].

Despite the development independent of optimization algorithm formalization, the `kernel_hom` tactic provides substantial practical benefits: by shifting from the functional representation with explicit integrals to categorical morphisms, kernel equalities that ordinarily require dozens of lines of formal reasoning can

[18] gaetanserre.fr/KernelHom

be discharged in a single line. In addition to this substantial reduction in proof complexity, this tactic allows to leverage all existing and future categorical tools developed in `Mathlib`, particularly automated results generation. Therefore, while this work is recent and has not yet been applied to concrete formalization projects, it is expected to play a crucial role in the future development of probability theory in `L∃∀N`, and thus in the formalization of global optimization algorithms.

3 —

Formalization of global optimization algorithms

Abstract. We now present the formalization of the unifying framework for iterative stochastic optimization algorithms introduced in Chapter B.1, along with the formal verification the measure-theoretic properties that underpin it. This formalization is crucial for establishing a rigorous foundation for the analysis of such algorithms, and for mechanizing the proofs of their convergence properties. To validate the practical applicability of this framework, we formalize the proof of Proposition B.1.2, presented in Section B.1.3. Moreover, we demonstrate that concrete algorithms, specifically LIPO, RANKOPT, and CMA-ES, can be formally instantiated within this framework, following the constructions outlined in Section B.1.1. This instantiation process involves defining the initial measure and kernel sequence for each algorithm, and verifying that they satisfy the necessary properties to fit within the abstract framework. Finally, we discuss the technical difficulties encountered in formalizing particle-based methods. We outline the broad contours of what a full formalisation of these methods would entail and identify the main challenges that need to be overcome to achieve this objective.

This section presents the L∃∀N formalization of the framework introduced in Definition B.1.16, together with the formal verification of the measure-theoretic properties (see Theorems B.1.16 and B.1.17). We then validate the practical applicability of this framework by formalizing the proof of Proposition B.1.2, presented in Section B.1.3. This mechanization serves as a case study demonstrating the utility of our abstract framework for rigorous analysis of optimization algorithms. We then instantiate this framework with concrete algorithms, following the constructions outlined in Section B.1.1 and discuss the technical difficulties encountered in formalizing particle-based methods.

The formalization follows the hierarchy described in Chapter B.1. The `Algorithm` structure encodes Definition B.1.16, so that every algorithm is an object of a single type. A family is a function that builds an `Algorithm` from what is specific to the

family. For instance, `Decision` builds a decision-based algorithm from an initial measure and decision sets, assumed measurable and of positive measure. LIPO and RANKOPT are then obtained by applying `Decision` to their own decision sets (see Section B.3.1). The McKean-Vlasov-based family would be built in the same way, from the drift and the diffusion of the SDE, once stochastic calculus is available in `Mathlib` (see Section B.3.3). Finally, PRS and CMA-ES are defined directly as objects of type `Algorithm`.

Results are then proved at the most general level where they hold. The trajectory measure and the two properties of Theorems B.1.16 and B.1.17 are formalized once for every `Algorithm`, and so is Proposition B.1.2 for every `Algorithm` on a compact metric space. The proof that the kernels of a decision-based algorithm are Markov kernels is written once, in `Decision`. Each of these results holds for every algorithm built below its level, without a new proof. Modules of a family, such as the trust region of Section A.2.4, are not formalized yet. They would fit the same scheme, as functions that take an algorithm of the family and return an `Algorithm`, with properties proved once for the whole family.

The formal code is available in two open-source repositories: the `LeanGO`[19] project contains both the abstract-framework formalization and the concrete-algorithm implementations; the `LipoCons`[20] project provides the mechanization of Proposition B.1.2.

Formalization of Definition B.1.16. We formalize the abstract algorithm definition via type abbreviations for iteration history and a L∃∀N structure encoding the initial measure and kernel sequence.

L∃∀N
```
abbrev iter (Ω : Type*) (n : ℕ) := Iic n → Ω

abbrev prod_iter_image (Ω β : Type*) (n : ℕ) :=
  (iter Ω n) × (iter β n)
```

Given two types Ω and β, the abbreviation `iter Ω n` encodes Ω^{n+1}, representing an $(n+1)$-step iteration history. Similarly, `prod_iter_image Ω β n` represents the joint history of points and their evaluations: $\Omega^{n+1} \times \beta^{n+1}$.

L∃∀N
```
structure Algorithm (Ω β : Type*) [MeasurableSpace Ω]
    [MeasurableSpace β] where
  /-- The initial smeasure over the search space `Ω`. -/
  ν : Measure Ω
  /-- The initial measure `ν` must be a probability
  measure. -/
  [prob_measure : IsProbabilityMeasure ν]
  /-- The kernels that defines how to sample the next point
  based on the history of points and evaluations. -/
  kernel_iter (n : ℕ) : Kernel (prod_iter_image Ω β n) Ω
  /-- The kernels must satisfy be Markov kernels. -/
```

[19] gaetanserre.fr/LeanGO
[20] gaetanserre.fr/LipoCons

```
12   [markov_kernel (n : ℕ) : IsMarkovKernel (kernel_iter n)]
```

Here, Ω is the search space, β is the evaluation space (typically $\mathbb{R}$), and ν is the initial measure. The type class instances ensure that ν is a probability measure and each `kernel_iter n` is a Markov kernel.

We construct the trajectory measure $\mu_\infty^{(V)}$ using the existing Ionescu–Tulcea formalization in LƎ∀N (Marion, 2026)

LƎ∀N
```
def measure (A : Algorithm Ω β) {V : Ω → β}
    (hV : Measurable V) : Measure (ℕ → Ω) :=
  (Kernel.traj (A.iter_comap hV) 0).avg A.v_mequiv
```

The term `A.iter_comap hf` implements the pullback kernels from Definition B.1.17 and `Kernel.traj` corresponds to the infinite product of kernels produced by the Ionescu-Tulcea theorem, a kernel of type $\Omega \rightsquigarrow (\mathbb{N} \to \Omega)$. The `.avg` operation averages this kernel with respect to the initial measure to yield a probability measure on $\mathbb{N} \to \Omega$ (for more details on the average operation, see the $\overline{S}$ operator in Definition B.A.28).

The finite-dimensional marginals are defined by pushing forward the trajectory measure along the function that truncates an infinite sequence to its first $n+1$ elements, `frestrictLe`

LƎ∀N
```
def fin_measure (A : Algorithm Ω β) (n : ℕ) {V : Ω → β}
    (hV : Measurable V) : Measure (iter Ω n) :=
  (A.measure hV).map (frestrictLe n)
```

For any object `A` of type `Algorithm Ω β`, the `measure` and `fin_measure` fields are available and corresponds to the measures $\mu_\infty^{(V)}$ and $\left(\mu_n^{(V)}\right)_{n\in\mathbb{N}}$ defined in Chapter B.1

LƎ∀N
```
#check A.measure hV
#check A.fin_measure n hV
```

```
A.measure hV : Measure (ℕ → Ω)
A.fin_measure n hV : Measure (iter Ω n)
```

Note that no assumptions are made to the spaces Ω and β, and the function V except measurability. This is allowed by the high level of generality of the Ionescu-Tulcea theorem, and is particularly suitable for the purpose of formalizing global optimization algorithms, as no prior assumptions are made on the functions being optimized or the search space.

Formalization of the measures' properties. We formalize the two key measure properties (Theorems B.1.16 and B.1.17) on which the proof of Proposition B.1.2 depends.

The LƎ∀N statement of Theorem B.1.17 is as follows

```
theorem fin_measure_mono {n m : ℕ} {s : Set (iter Ω n)}
    (hs : MeasurableSet s) {e : Set (iter Ω m)}
    (he : MeasurableSet e) (hmn : n ≤ m)
    (hse : e ⊆ {u | subTuple hmn u ∈ s}) {V : Ω → β}
    (hf : Measurable V) :
    A.fin_measure m hV e ≤ A.fin_measure n hV s := by ...
```
LEAN

where `subTuple hmn` takes a tuple of size `m + 1` and returns its first `n + 1` elements. The LEAN proof essentially follows the same steps as the one presented in Chapter B.1.

The LEAN statement of Theorem B.1.16 is as follows

```
theorem eq_restrict {f g : Ω → β} (hf : Measurable f)
    (hg : Measurable g) {s : Set Ω} (hs : MeasurableSet s)
    (h : s.EqOn f g) (n : ℕ) :
    (A.fin_measure n hf).restrict (rect s n) =
    (A.fin_measure n hg).restrict (rect s n) := by ...
```
LEAN

where `rect s n` is corresponds to the $n+1$ cartesian product of s with itself. Analogously, the LEAN proof mirrors the proof presented in Chapter B.1, although the decomposition of the product of kernels is carried out with considerably greater technical precision.

Formalization of Proposition Proposition B.1.2. With the framework and its measure-theoretic properties fully formalized, we present a formal proof of the sketch in Section B.1.3. The LEAN statement reads

```
variable {Ω : Type*} [MeasurableSpace Ω]
  [PseudoMetricSpace Ω] [CompactSpace Ω]
  [Nonempty Ω] [OpensMeasurableSpace Ω]

theorem sample_iff_consistent (A : Algorithm Ω ℝ) :
    (∀ {V : Ω → ℝ}, (hV : Lipschitz V) →
      sample_whole_space A hV.continuous)
    ↔
    (∀ {V : Ω → ℝ}, (hV : Lipschitz V) →
      is_consistent A hV) := by ...
```
LEAN

Here, `sample_whole_space` and `is_consistent` encode properties **i.** and **ii.** of Proposition B.1.2, respectively. Convergence is expressed via `fin_measure`.

The formalization leverages type classes to encode the mathematical structure of the search space, enabling work in a general setting without fixing a concrete type. The hypotheses match those of Proposition B.1.2 as the search space only needs to be a compact metric space, which includes compact subsets of $\mathbb{R}^d$. This effort, comprising approximately 1000 lines of code, establishes that any object fitting into the `Algorithm` structure satisfies the equivalence in Proposition B.1.2. The next step is then to demonstrate that concrete algorithms can be represented as objects of this type.

A canonical example of a concrete algorithm is the *Pure Random Search* (PRS), which samples points independently from a fixed distribution μ. This algorithm is particularly elegant to formalize within our framework: since all sampled points follow the same distribution irrespective of the iteration history, the transition kernels remain constant across all iterations. This translates directly into the following formalization

```
def PRS : Algorithm α β where
  V := μ
  kernel_iter _ := Kernel.const _ μ
```
LEAN

Notably, each kernel discards its input (the iteration history, represented by the wildcard _) and unconditionally returns the fixed distribution μ. While PRS provides an accessible introduction to instantiating our framework, practical algorithms typically exhibit more intricate behavior. To illustrate the expressiveness of our framework, we examine decision-based approaches, specifically LIPO and RANKOPT.

3.1 Formalization of decision-based algorithms

We present an LEAN interface for decision-based approaches, as presented in Chapter A.2, based on the `Algorithm` structure defined above. We then instanciate this interface for the LIPO and RANKOPT algorithms, following the constructions outlined in Section B.1.1,

Interface for decision-based approaches. Recall that decision-based approaches rely on an initial probability measure μ (often the uniform distribution on the search space) and a sequence of decision rules $D_n : \Omega^{n+1} \times \beta^{n+1} \to [0,1]$. At each iteration n, a candidate point x is sampled from μ and accepted with probability $D_n(\mathbf{C}_n, V(\mathbf{C}_n), x)$.

Consider the special case of binary decision rules, where D_n is $\{0,1\}$-valued. In this setting, acceptance is deterministic: a candidate point is accepted if and only if $D_n(\mathbf{C}_n, V(\mathbf{C}_n), x) = 1$. The transition kernels then take the form

$$\kappa_n(e_V(\mathbf{C}_n)) = \mu\|_{\{x \mid D_n(\mathbf{C}_n, V(\mathbf{C}_n), x)=1\}},$$

where $\mu\|_A$ denotes the restriction and renormalization of μ to a set A. This restricted measure is a probability measure provided that A has positive measure under μ.

We formalize decision-based approaches through `Decision`, a specialization of the `Algorithm` structure. Several conditions are essential: measurability properties of the decision rules and the positive measure of the acceptance set. Under these conditions, `Decision` correctly encodes the transition kernels described above.

```
/-- The interface for decision-based optimization
algorithms. -/
```
LEAN

```
def Decision : Algorithm α β where
  ν := μ
  kernel_iter _ :=
    decision_kernel μ measurableSet_decision_prod
  markov_kernel n :=
    ⟨fun data ↦ cond_isProbabilityMeasure (h n data)⟩
```

We do not show the full code nor the signature of the assumptions for the sake of readability.

Formalization of LIPO. We now formalize LIPO within this framework. Recall the decision rules defined in Eq. 12. The acceptance criterion is formalized through the set of candidate points satisfying

```
def potential_max : Set α :=
  {x | Tuple.max (fun i ↦ data.2 i) ≤
    Tuple.min (fun i ↦ data.2 i + κ * dist (data.1 i) x)}
```

Here, `data` records the iteration history, with `data.1 i` and `data.2 i` denoting the i-th point and its evaluation, respectively. We establish that LIPO satisfies the `Decision` interface through a utility lemma ensuring measurability conditions of this acceptance set.

The LIPO formalization then becomes

```
def LIPO : Algorithm α ℝ := Decision μ
  (fun _ ↦ measurableSet_potential_max_prod κ)
  h -- decision set has positive measure
```

where `μ` is the initial measure, `measurableSet_potential_max_prod` ensures measurability under mild assumptions on the search space, and `h` is an assumption that asserts that the decision set has positive measure. This assumption is not trivial to verify, as it depends on the structure of the search space and the choice of μ, and is left to the user to verify in concrete cases.

Once formalized, we can directly access the trajectory measure, finite-dimensional marginals, and the consistency equivalence for LIPO, provided the search space satisfies the necessary structure (compact metric space, etc.)

```
#check (LIPO μ κ h).measure hV
#check (LIPO μ κ h).fin_measure n hV
#check sample_iff_consistent (LIPO μ κ h)
```

```
(LIPO μ κ h).measure hV : Measure (ℕ → α)
(LIPO μ κ h).fin_measure n hV : Measure (iter α n)
sample_iff_consistent (LIPO μ κ h) : ...
```

Formalization of RankOpt. We formalize RANKOPT using the same approach as LIPO, but with a different decision rule. Unlike LIPO, the evaluation space for RANKOPT is more general than $\mathbb{R}$. Recall that the decision rules of RANKOPT are defined in

Eq. 14. Given a set of ranking rules $\mathcal{R}$, the set of points satisfying the decision rule of RANKOPT is formalized as follows

```
def potential_max (ℛ : Set (RankRule α)) : Set α :=
    {x | ∃ (r : ℛ), ranking_loss data r = 0 ∧
      0 ≤ (r.1.1 x (argmax_f data)).1}
```

where `RankRule` is the type of functions from $\alpha \times \alpha$ to $\{-1, 0, 1\}$ that are jointly measurable, and `ranking_loss` corresponds to Eq. 15.

Given some mild assumptions on the search space and the evaluation space, we can then show that RANKOPT satisfies the `Decision` interface.

```
def RankOpt : Algorithm α β := Decision μ
  (fun _ ↦ measurableSet_potential_max_prod hℛ)
  h -- decision set has positive measure
```

where `measurableSet_potential_max_prod` is the RANKOPT equivalent of LIPO's utility lemma on the measurability of the set of points satisfying the decision rule, and `h` encodes the assumption that this set has positive measure.

Thus, the decision-based family is defined once, by `Decision`. LIPO and RANKOPT only give their decision sets, and the proof that their kernels are Markov kernels comes from `Decision`. This is the theoretical counterpart of the `Decision` base class of GLOBE, from which decision-based algorithms inherit in code (see Chapter A.4).

The current `Decision` interface restricts to binary decision rules, which suffices for LIPO and RANKOPT. We plan to extend it to continuous acceptance probabilities to enable formalization of ADALIPO and ADARANKOPT.

3.2 Formalization of CMA-ES

We present a formalization of the CMA-ES algorithm within our framework, that encompasses any mean and covariance update rules, as long as they are measurable w.r.t. the full iteration history.

As explained in Section B.1.1, to instantiate CMA-ES within our framework, the search space must be of the form $\mathbb{R}^{d \times \lambda}$ to represent the batches of candidate points sampled at each iteration. The formulation relies on measurable functions `mean` and `covar` that compute the mean and covariance matrix of the next batch of candidate points from the iteration history. The transition kernels are then defined as follows

```
def CMAKernel (n : ℕ) :
    Kernel (prod_iter_image (ℝ_ d lam) (ℝ_ d lam) n)
      (ℝ_ d lam) where
  toFun data := Measure.pi (fun _ ↦ multivariateGaussian
      (mean n data) (covar n data))
  measurable' := by ...
```

where `ℝ_ d lam` denotes the type of λ-tuples of d-dimensional real vectors, `multivariateGaussian` is the multivariate Gaussian distribution on $\mathbb{R}^d$, `mean n data` and `covar n data` compute the mean and covariance matrix from the iteration history at iteration n, and `Measure.pi` constructs the product measure of λ independent multivariate Gaussians. The measurability condition relies solely on the measurability of `mean` and `covar`, a standard assumption satisfied by all practical implementations.

The resulting formalization of CMA-ES is given by

```
def CMA_ES : Algorithm (ℝ_ d lam) (ℝ_ d lam) where
  v := Measure.pi (fun _ ↦ multivariateGaussian m S)
  kernel_iter := CMAKernel d lam hmean hcovar
  markov_kernel n :=
    ⟨fun a ⇒ by simp [CMAKernel]; infer_instance⟩
```

where `m` and `S` are the initial mean and covariance matrix, and `hmean` and `hcovar` are the measurability assumptions on the `mean` and `covar` functions.

This formalization accommodates arbitrary measurable mean and covariance update rules. The framework encompasses the classical CMA-ES update rules described in Section A.1.1 alongside alternative update rules proposed in the literature, including those in Akimoto et al. (2014); Jastrebski & Arnold (2006); Loshchilov et al. (2018); Ros & Hansen (2008).

3.3 On the formalization of McKean-Vlasov-based approaches

As explained in Section B.1.1, McKean-Vlasov-based approaches can conceptually be formalized within our framework, as the solution of such SDEs can be represented as a family of non-linear Markov kernels (see Appendix A). However, this formalization is technically challenging, as it requires a precise mechanization of stochastic calculus. While differential calculus is well-developed in `Mathlib` (see for example Gouëzel (2026)), stochastic calculus remains in its early stages: the tools necessary for merely stating an SDE are not yet fully in place, and those needed to derive properties of their solutions, such as transition kernels, are even less so. The formalization of McKean-Vlasov-based approaches is therefore currently out of reach, but is expected to become feasible as the probability theory library in `Mathlib` matures. However, when the diffusion term is zero, as for SBS, the particles follow an ordinary differential equation that requires no stochastic calculus, and the formalization of such algorithms within our framework is potentially feasible today.

A significant step in this direction is the Brownian motion formalization project[21] led by Rémy Degenne, which provides formalizations of the Kolmogorov extension theorem, the Kolmogorov–Chentsov continuity theorem, and the construction of

[21] github.com/RemyDegenne/brownian-motion

a Brownian motion on the nonnegative reals — all essential prerequisites for a formalization of SDEs. The project is ongoing; forthcoming steps include a formalization of Itô's lemma, which would provide a first means of writing SDEs in LΞ∀N. We plan to contribute to this effort in the future to facilitate the extension of our formalization to McKean-Vlasov-based approaches.

In the following, we outline how such a formalization would proceed, assuming the necessary SDE infrastructure is available. The first step would be to state the McKean-Vlasov SDE that defines the algorithm. It would take a similar form

LΞ∀N
```
structure McKeanVlasov (Ω : Type*) where
  drift : Ω → Measure Ω → Ω
  diffusion : Ω → Measure Ω → Matrix d d Ω
```

where $\Omega \subseteq \mathbb{R}^d$. Then, through a type class, we would encode the fact that some process $(X_t)_{t\geq 0}$ is a solution of this SDE such as

LΞ∀N
```
class IsMcKeanVlasovSolution (X : ℝ≥0 → Ω)
    (mkv : McKeanVlasov Ω) where
  is_solution : ...
```

A theorem would establish the existence and uniqueness of a solution to the McKean-Vlasov SDE, given suitable regularity assumptions on the drift and diffusion. This type class would then provide an API to access the transition kernels of the process, such as

LΞ∀N
```
def IsMcKeanVlasovSolution.transition_kernels (X : ℝ≥0 → Ω)
    (mkv : McKeanVlasov Ω) [IsMcKeanVlasovSolution X mkv] :
    DMarkovFamily ℝ≥0 Ω where ...
```

where `DMarkovFamily` is the type of families of Markov kernels depending on the law of the process, as defined in Definition B.A.28 (see Appendix A).

One could then use these kernels to instantiate the `Algorithm` structure, and thus to access all tools developed in the previous sections, such as the trajectory measure.

3.4 Discussion and perspectives

In this chapter, we introduced `LeanGO`, a LΞ∀N library providing a formalization of a unifying framework for iterative stochastic optimization algorithms, together with a formal verification of the underlying measure-theoretic properties, as well as the formalization of several representative algorithms within this framework. This contribution constitutes the first LΞ∀N library dedicated to the formalization of global optimization algorithms, thereby establishing a foundation for future developments in this direction.

LeanGO follows the same hierarchy as GLOBE. In GLOBE, every algorithm inherits from the Optimizer class, and a family is a base class, such as Decision (see Chapter A.4). In LeanGO, every algorithm is an object of the type Algorithm, and a family is a special case of this type, such as Decision (see Section B.3.1). As every algorithm of GLOBE can be used through the Optimizer interface, the same benchmark runs on all of them. In the same way, a theorem proved for any Algorithm holds for every algorithm of LeanGO. Within a family, a plugin written once for the Decision class works for every decision-based algorithm of GLOBE. Similarly, a theorem proved once for Decision holds for LIPO and RANKOPT, without a new proof. Thus, GLOBE factors the code shared by algorithms, and LeanGO factors their proofs. The McKean-Vlasov-based family has no counterpart in LeanGO yet, since Mathlib does not have the stochastic calculus needed to state its SDEs (see Section B.3.3).

Future work will proceed along several complementary axes. First, we plan to align and integrate our formalization efforts with the *Lean Machine Learning* project, a recent initiative aimed at developing a comprehensive L∃∀N ecosystem for machine learning algorithms. This library already includes formalizations of bandit algorithms, and we intend to extend it to encompass global optimization methods as well. By merging these efforts, we aim to establish a unified formal framework for machine learning and optimization within L∃∀N.

Second, we will focus on formalizing classical theoretical results for optimization algorithms, including convergence rate analyses, in order to demonstrate the practical relevance of the proposed framework. We will also address the formalization of more complex algorithms, such as the BAYESOPT algorithm, which may require the development of additional formal tools to capture their specific structural properties.

Then, we will contribute to the development of stochastic calculus within Mathlib, with the goal of enabling the formalization of McKean–Vlasov–based approaches. Although this represents a substantial undertaking, it is a necessary step toward fully realizing the potential of our framework for the formal verification of global optimization algorithms.

Finally, we will investigate the development of L∃∀N tooling aimed at bridging the gap between concrete algorithm implementations and their formal counterparts within our framework. Such tools would enable users to experiment with algorithms in a practical setting while retaining the ability to formally verify their properties within the same environment, thereby reducing the overhead associated with embedding implementations into an abstract formalism. As GLOBE and LeanGO share the same hierarchy, a first target is to connect them. Each algorithm of GLOBE would then come with its formal counterpart and its proofs. Although this direction remains largely exploratory at this stage, it constitutes a natural extension of the present work toward making formalization more accessible and practically usable.

Appendix

A —

Markov processes and generators

Abstract. This section provides a comprehensive exposition of the theory of Markov processes, semigroups, Feller processes, generators, and their connections to stochastic differential equations, drawing on established results from Chaintron & Diez (2022); Del Moral (2004); Del Moral (2013); Le Gall (2016). A distinctive feature is the parallel LEAN formalization available in the repository[22]. Each major definition and result concerning families of Markov kernels, their operators, and Feller semigroups is accompanied by its formal counterpart, ensuring rigorous mechanized verification of the underlying theory. To our knowledge, this provides the first such comprehensive formalization in a proof assistant, establishing a solid foundation for formal analysis of stochastic processes and their applications in stochastic algorithms.

A.1 Family of Markov kernels

Let $(X, \mathscr{X})$ and $(Y, \mathscr{Y})$ two measurable spaces. A Markov kernel is a measurable function that associates to each element $x \in X$ a probability measure on $(Y, \mathscr{Y})$.

[22] github.com/gaetanserre/MarkovProcess

Definition B.A.21 – *Markov kernel*

Given two measurable spaces $(X, \mathcal{X})$ and $(Y, \mathcal{Y})$, a Markov kernel is a function $\kappa : X \to \mathcal{P}(Y)$ such that

$$\forall A \in \mathcal{Y}, \kappa(\cdot, A) \text{ is measurable.}$$

Remark

An equivalent way to write the measurability condition is to require κ to be measurable w.r.t. the measurable space of probability measures on Y (Giry, 1982).

For details on the LƎ∀N definition of measurable spaces, measures, and Markov kernels, see Section B.2.2.

As, for any $x \in X$, $\kappa(x)$ is a probability measure on $Y, \mathcal{Y}$, we can associate to κ a linear operator acting on the space of bounded strongly-measurable functions on Y. For any $f \in B(Y)$ and any $x \in X$, this operator is defined by

$$\kappa f \overset{\text{def}}{=} \mathbb{E}_{y \sim \kappa(x)}[f(y)] = \int_Y f(y)\, \mathrm{d}\kappa.$$

By abuse of notation, we also denote this operator by κ. It is important to note that this operator preserves the boundedness and strong measurability of functions (it is even a contraction in the supremum norm). This property allows the composition of multiple operators arising from Markov kernels.

LEAN formalization. The LƎ∀N formalization of this kernel operator is as follows

LƎ∀N

```
def operator {X Y Z : Type*} [MeasurableSpace X]
    [MeasurableSpace Y] [MeasurableSpace Z]
    [NormedAddCommGroup Z] [NormedSpace ℝ Z]
    (κ : Kernel X Y) [IsMarkovKernel κ]
    (f : Y →ₘ Z) :=
  fun x ↦ ∫ y, f y ∂κ x
```

The notation `F →ₘ Z` denotes the type of bounded strongly-measurable functions from Y to Z. The preservation of boundedness and strong measurability by the operator is formalized by showing that `operator κ f` has type `X →ₘ Z` whenever `f` has type `F →ₘ Z`.

Such a kernel can be used to describe the transitions of a stochastic process: at each time $t \geq 0$, one associates a Markov kernel Q from $(X, \mathcal{X})$ to itself such that, for any state $x \in X$, the probability distribution of states reachable from x after time t is given by $Q(x)$.

This usage of Markov kernels motivates the definition of Markov families of kernels.

Definition B.A.22 – *Family of Markov kernels*

Let $(X, \mathscr{X})$ be a measurable space and $(G, +, \mathscr{Z})$ a measurable semigroup. A family of kernels $(Q_t : X \rightsquigarrow X)_{t \in Z}$ is a family of Markov kernels if

i. for any $t \in Z$, Q_t is a Markov kernel;

ii. there exists $\mathbb{0} \in Z$ such that, for any $x \in X$, $Q_e(x) = \delta_x$;

iii. for any $t, s \in Z$, $x \in X$ and $A \in \mathscr{X}$,

$$Q_{t+s}(x, A) = \int_X Q_t(y, A) \, \mathrm{d}Q_s(x).$$

iv. for any $A \in \mathscr{X}$, the function $(t, x) \mapsto Q_t(x, A)$ is measurable w.r.t. the product σ-algebra $\mathscr{Z} \otimes \mathscr{X}$.

In classical literature where $Z \overset{\text{def}}{=} \mathbb{R}_+$, $\mathbb{0} \overset{\text{def}}{=} 0$ is chosen in property **ii.**. Indeed, if $(Q_t)_{t \in \mathbb{R}_+}$ represents the transition of a stochastic process, then Q_0 represents the absence of movement, i.e., the initial state is unchanged.

Property **iii.** is known as the *Chapman-Kolmogorov equation*. It allows to define the composition between kernels such that **iii.** can be rewritten as $Q_{t+s} = Q_t \otimes_k Q_s$.

LEAN formalization. To formalize such a family of kernels in LEAN, we introduce the following structure

LEAN

```
structure ProbKernels (G X : Type*) [AddSemigroup Z]
    [MeasurableSpace X] where
  toFun : Z → Kernel X X
  markov_kernels g : IsMarkovKernel (toFun g)
```

This structure simply defines a family of Markov kernels that are indexed by an additive semigroup. The properties **ii.**, **iii.** and **iv.** are not included in this definition, but are instead formalized in an other structure that inherits from `ProbKernels`

LEAN

```
structure MarkovFamily (G X : Type*) [AddSemigroup Z]
    [MeasurableSpace Z] [Zero Z] [MeasurableSpace X]
    extends ProbKernels Z X where
  -- Property ii)
  kernel_dirac : toFun 0 = Kernel.dirac
  -- Property iii)
  chapman_kolmogorov t s : toFun (t + s) =
    toFun t ∘ₖ toFun s
  -- Property iv)
  measurable : Measurable (fun (t, x) ↦ toFun t x)
```

Note that the Chapman-Kolmogorov equation is formalized using the composition of kernels.

The Chapman-Kolmogorov equation can also be formulated in terms of composition of the associated operators.

Lemma B.A.17

Given a family of Markov kernels $(Q_t)_{t\in Z}$, the Chapman-Kolmogorov equation is equivalent to the following property

$$\forall t, s \in Z, Q_{t+s} = Q_t \overset{\leftarrow}{\circ} Q_s = Q_s \circ Q_t.$$

Proof. ($\Longleftarrow$) For any $x \in X$ and any $A \in \mathcal{X}$, we have

$$\begin{aligned} Q_{t+s}(x, A) &= \int_X \mathbb{1}_A(y) \,\mathrm{d}Q_{t+s}(x) \\ &= Q_{t+s}(\mathbb{1}_A)(x) \\ &= \int_X \int_X \mathbb{1}_A(z) \,\mathrm{d}Q_t(y) \,\mathrm{d}Q_s(x) \\ &= \int_X Q_t(y, A) \,\mathrm{d}Q_s(x). \end{aligned}$$

($\Longrightarrow$) For any function $f \in B(X)$, let $\tilde{f} \overset{\text{def}}{=} f + \|f\|_\infty$. This function is non-negative, bounded and strongly-measurable. Therefore, it can be written as the limit of a non-decreasing sequence of sum of indicator functions of measurable sets, i.e, there exists sequences $\left(a_{n,k}\right)_{n,k\in\mathbb{N}} \in X^{\mathbb{N}}$, $\left(A_{n,k}\right)_{n,k\in\mathbb{N}} \in \mathcal{X}^{\mathbb{N}}$ and a function $\Phi : \mathbb{N} \to \mathbb{N}$ with $\Phi(n) \to \infty$ such that

$$\forall x \in X, \sum_{k=1}^{\Phi(n)} a_{n,k} \mathbb{1}_{A_{n,k}}(x) \underset{n\to\infty}{\nearrow} \tilde{f}(x).$$

Therefore, for any $x \in X$,

$$\sum_{k=1}^{\Phi(n)} a_{n,k} \mathbb{1}_{A_{n,k}}(x) - \|f\|_\infty \to f(x).$$

As this sequence is non-decreasing, we can apply the monotone convergence theorem to obtain

$$\begin{aligned} \forall x \in X, \int_X f(y) \,\mathrm{d}Q_{t+s}(x) &= \lim_{n\to\infty} \int_X \sum_{k=1}^{\Phi(n)} a_{n,k} \mathbb{1}_{A_{n,k}}(y) \,\mathrm{d}Q_{t+s}(x) - \|f\|_\infty \\ &= \lim_{n\to\infty} \sum_{k=1}^{\Phi(n)} a_{n,k} \int_X \mathbb{1}_{A_{n,k}}(y) \,\mathrm{d}Q_{t+s}(x) - \|f\|_\infty \\ &= \lim_{n\to\infty} \sum_{k=1}^{\Phi(n)} a_{n,k} Q_{t+s}(x, A_{n,k}) - \|f\|_\infty \\ &= \lim_{n\to\infty} \sum_{k=1}^{\Phi(n)} a_{n,k} \int_X Q_t(y, A_{n,k}) \,\mathrm{d}Q_s(x) - \|f\|_\infty \\ &= \lim_{n\to\infty} \sum_{k=1}^{\Phi(n)} a_{n,k} \int_X \int_X \mathbb{1}_{A_{n,k}}(z) \,\mathrm{d}Q_t(y) \,\mathrm{d}Q_s(x) - \|f\|_\infty \end{aligned}$$

$$= \int_X \int_X \lim_{n\to\infty} \sum_{k=1}^{\Phi(n)} a_{n,k} \mathbb{1}_{A_{n,k}}(z) \, \mathrm{d}Q_t(y) \, \mathrm{d}Q_s(x) - \|f\|_\infty$$
$$= \int_X \int_X f(z) \, \mathrm{d}Q_t(y) \, \mathrm{d}Q_s(x).$$

□

As the LEAN formalization of the Chapman-Kolmogorov equation is based on the composition of kernels, the previous lemma is stated using this composition. The proof follows the same lines as the one given above but is not included here for the sake of brevity.

LEAN

```
lemma kernel_comp_iff_operator_comp (Q : ProbKernels X Y) :
    (∀ t s, Q (t + s) = Q t ∘ₖ Q s) ↔ ∀ t s,
      (Q (t + s)).operator = (Q s).operator ∘ (Q t).operator
```

Note that this lemma does not require the family of kernels to be a family of Markov kernels as in Definition B.A.22, but only a family of kernels as in the `ProbKernels` structure.

Remark

In general, the composition of kernels is not commutative. However, in classical litteration of Markov processes, $Z \overset{\text{def}}{=} \mathbb{R}_+$. As the addition of real numbers is commutative, the Chapman-Kolmogorov equation implies that $Q_t \otimes_k Q_s = Q_s \otimes_k Q_t$ for any $t, s \in \mathbb{R}_+$.

A.2 Transition monoids

Let $(G, +, \mathscr{Z})$ be a measurable semigroup and $(Q_t)_{t\in Z}$ a family of Markov kernels on $(X, \mathscr{X})$. While Le Gall (2016) introduces $(Q_t)_{t\in Z}$ as a *transition semigroup*, a rigorous characterization of its algebraic structure is often omitted in the literature. We provide explicit proofs that $\left((Q_t)_{t\in Z}, \otimes_k\right)$ and $\left((Q_t)_{t\in Z}, \overleftarrow{\circ}\right)$ are not merely semigroups but monoids.

A.2.1 Kernels monoid

A monoid is an algebraic structure $(G, +, e)$ such that $(G, +)$ is a semigroup and $e \in G$ is a neutral element for $+$, i.e., for any $x \in G$, $x + e = e + x = x$. We start by showing that $\left((Q_t)_{t\in Z}, \otimes_k\right)$ is a monoid.

Kernels semigroup. Let $f : (G, +) \to \left((Q_t)_{t\in Z}, \otimes_k\right)$ be the function that maps each $t \in Z$ to Q_t. This function is a semigroup homomorphism, i.e., for any $t, s \in Z$,

$$f(t + s) = Q_{t+s} = Q_t \otimes_k Q_s = f(t) \otimes_k f(s).$$

Moreover, as f is trivially surjective, $\left((Q_t)_{t\in Z}, \otimes_k\right)$ is indeed a semigroup

$$\forall t, s, r \in Z, Q_t \otimes_k Q_s \otimes_k Q_r = f(t) \otimes_k f(s) \otimes_k f(r)$$

$$
\begin{aligned}
&= f(t+s+r) \\
&= f(t+(s+r)) \\
&= f(t) \otimes_k f(s+r) \\
&= Q_t \otimes_k (Q_s \otimes_k Q_r).
\end{aligned}
$$

Neutral element. We now show that $Q_{\mathbb{0}}$ is a neutral element of $\left((Q_t)_{t\in Z}, \otimes_k\right)$. For any $t \in Z$, we have

$$
\begin{aligned}
\forall x \in X, A \in \mathcal{X}, (Q_t \otimes_k Q_{\mathbb{0}})(x, A) &= Q_{t+\mathbb{0}}(x, A) \\
&= \int_X Q_t(y, A)\, \mathrm{d}Q_{\mathbb{0}}(x) \\
&= \int_X Q_t(y, A)\, \mathrm{d}\delta_x(y) \\
&= Q_t(x, A).
\end{aligned}
$$

Conversely,

$$
\begin{aligned}
(Q_{\mathbb{0}} \otimes_k Q_t)(x, A) &= \int_X \delta_y(A)\, \mathrm{d}Q_t(x) \\
&= \int_X \mathbb{1}_A(y)\, \mathrm{d}Q_t(x) \\
&= Q_t(x, A).
\end{aligned}
$$

This shows that $\left((Q_t)_{t\in Z}, \otimes_k, Q_{\mathbb{0}}\right)$ is a monoid.

LEAN formalization. To formalize the semigroup structure of a family of Markov kernel in L∃∀N, we introduce the `KernelsMonoid` type class that, given a family of kernels indexed by a semigroup, requires that the addition of two indexes corresponds to the composition of the associated kernels and that the kernel associated to $\mathbb{0}$ is the dirac kernel.

L∃∀N

```
class KernelsMonoid {G X : Type*} [AddSemigroup Z] [Zero Z]
    [MeasurableSpace X] (Q : α → Kernel X X) where
  map_add t s : Q (t + s) = Q t ∘ₖ Q s
  dirac_0 : Q 0 = Kernel.dirac
```

Using only these properties, we can show that $\left((Q_t)_{t\in Z}, \otimes_k\right)$ is a monoid. The semigroup structure follows from an additive homomorphism `addHom` : $Z \to \left((Q_t)_{t\in Z}, \otimes_k\right)$, a consequence of the `map_add` property of the `KernelsMonoid` type class.

L∃∀N

```
def addHom : Z →₊ Q.image where
  toFun := fun t ↦ Q.elem t
  map_add' t s := by
    ext : 1
    dsimp only [HAdd.hAdd, Add.add]
    rw [← i.map_add t s]
    rfl
```

The surjectivity of `addHom` establishes the semigroup structure.

```
instance : AddSemigroup Q.image :=
  (addHom_surj Q).addSemigroup (addHom Q) (addHom Q).map_add
```

Setting `Q 0` as the neutral element yields the `AddMonoid` type class.

Then, if $\left((Q_t)_{t\in Z}, \otimes_k\right)$ has the properties of Definition B.A.22, we can show that it satisfies the `KernelsMonoid` type class, and thus is a monoid.

```
instance : KernelsMonoid Q where
  map_add t s := Q.chapman_kolmogorov t s
  dirac_0 := Q.kernel_dirac
```

A.2.2 Operators monoid

We proceed the same way to show that $\left((Q_t)_{t\in Z}, \overleftarrow{\circ}\right)$ is a monoid.

Operators semigroup. Analogously, let $f : (G, +) \to \left((Q_t)_{t\in Z}, \overleftarrow{\circ}\right)$ be the function that maps each $t \in Z$ to Q_t. The Chapman-Kolmogorov equation being equivalent to the property that $Q_{t+s} = Q_t \overleftarrow{\circ} Q_s$, f is a semigroup homomorphism, yielding the semigroup structure of $\left((Q_t)_{t\in Z}, \overleftarrow{\circ}\right)$.

Neutral element. For any $f \in B(X)$, we have

$$\forall x \in X, Q_{\mathbb{0}} f(x) = \int_X f(y)\, \mathrm{d}Q_{\mathbb{0}}(x) = \int_X f(y)\, \mathrm{d}\delta_x = f(x).$$

Thus, for any $t \in Z$ and any $f \in B(X)$,

$$\left(Q_t \overleftarrow{\circ} Q_{\mathbb{0}}\right) f = Q_{\mathbb{0}}(Q_t f) = Q_t f$$

and

$$\left(Q_{\mathbb{0}} \overleftarrow{\circ} Q_t\right) f = Q_t(Q_{\mathbb{0}} f) = Q_t f.$$

Consequently, $\left((Q_t)_{t\in Z}, \overleftarrow{\circ}, Q_{\mathbb{0}}\right)$ is a monoid.

LEAN formalization. We follow the same steps as for the formalization of the kernels monoid. We introduce the `OperatorsMonoid` type class that requires the addition of two indexes to correspond to the reverse composition of the associated operators and that the operator associated to $\mathbb{0}$ is the identity operator.

```
class OperatorsMonoid (Q : Z → (X →ₘ ℝ) → X →ₘ ℝ) where
  map_add t s : Q (t + s) = Q s ∘ Q t
  id_0 f : Q 0 f = f
```

As $\left((Q_t)_{t\in Z}, \overleftarrow{\circ}\right)$ are operators associated to a family of Markov kernels, the `OperatorsMonoid` type class is satisfied

```
instance : OperatorsMonoid (fun a ↦ (Q a).operator) where
  map_add t s := Q.operator_add t s
  id_0 := Q.operator_id
```

where `Q.operator_add` is the right implication of Lemma B.A.17.

A.3 Markov processes

Let $\left(X, \mathscr{X}, (\mathscr{F}_t)_{t\in\overline{\mathbb{R}}_+}, \mathbb{P}\right)$ be a filtered probability space. We define a *Markov process* on this space.

Definition B.A.23 – *Markov process*

Let $(Q_t)_{t\in\overline{\mathbb{R}}_+}$ be a family of Markov kernels on $(X, \mathscr{X})$ as in Definition B.A.22. A Markov process associated to this family is a stochastic process $(X_t)_{t\in\overline{\mathbb{R}}_+}$ that is $\mathscr{F}_t$-adapted and such that, for any $t, s \geq 0$ and any $f \in B(X)$,

$$\mathbb{E}[f(X_{t+s}) \mid \mathscr{F}_t] = Q_s f(X_t).$$

If the filtration is not specified, we consider the natural filtration of the process

$$\mathscr{F}_t^X \overset{\text{def}}{=} \sigma\left((X_s)_{0\leq s\leq t}\right).$$

The existence of such a process is guaranteed by Kolmogorov's extension theorem. We refer to Le Gall (2016) for details on the construction of Markov processes from a family of Markov kernels.

For any $A \in \mathscr{X}$, the classical Markov property is recovered by setting $f \overset{\text{def}}{=} \mathbb{1}_A$

$$\mathbb{P}(X_{t+s} \in A \mid \mathscr{F}_t) = Q_s(X_t, A).$$

Definition B.A.24 – *λ-resolvent*

Let $(Q_t)_{t\in\overline{\mathbb{R}}_+}$ be a family of Markov kernels on $(X, \mathscr{X})$ and $\lambda > 0$. The λ-resolvent of $(Q_t)_{t\in\overline{\mathbb{R}}_+}$ is the linear operator $R_\lambda : B(X) \to B(X)$ defined by

$$\forall x \in X, R_\lambda f(x) \overset{\text{def}}{=} \int_0^\infty e^{-\lambda t} Q_t f(x)\, \mathrm{d}t.$$

It corresponds to the Laplace transform of the function $x \mapsto Q_t f$.

The λ-resolvent of a family of Markov kernels posses several properties that we will not detail here. This operator is used in Theorem B.A.18 to show that the infinitesimal generator of a Feller semigroup can be used to recover the semigroup of kernels itself. For more details on the properties of the λ-resolvent, see Le Gall (2016).

Remark

The property **iv.** of Definition B.A.22 is used as the measurability of $x \mapsto Q_t f$ is necessary to ensure the well-definedness of the Lebesgue integral in the definition of the λ-resolvent.

A.4 Feller processes and their generators

Feller processes are a particular class of Markov processes that are associated to a family of Markov kernels with additional regularity properties. They are particularly relevant in the context of stochastic algorithms as they can be used to model the evolution of the distribution of the iterates of an algorithm. In particular, any solution of a *standard* stochastic differential equation (as opposed to McKean-Vlasov SDEs) with Lipschitz coefficients is a Feller process (Le Gall, 2016).

Definition B.A.25 – *Feller semigroup*

Let $(G, +)$ a semigroup and $(A_t)_{t \in Z}$ a family of linear operators on $B(X)$. Let $t, s \in Z$ and $f \in B(X)$. The family $\left((A_t)_{t \in Z}, \overset{\leftarrow}{\circ}\right)$ is a Feller semigroup if

- **i.** $A_{t+s} = A_t \overset{\leftarrow}{\circ} A_s$;
- **ii.** there exists $\mathbb{0} \in Z$ such that, $A_{\mathbb{0}} f = f$;
- **iii.** $A_t f \in B(X)$;
- **iv.** $\|A_t f\|_\infty \leq \|f\|$;
- **v.** $A_t \mathbf{1} = \mathbf{1}$;
- **vi.** if $f \geq 0$, then $A_t f \geq 0$;
- **vii.** if f is continuous, then $A_t f$ is continuous;
- **viii.** $\|A_t f - f\|_\infty \underset{t \downarrow \mathbb{0}}{\to} 0$.

We observe several key aspects of this definition. First, by the results of Appendix A.2, properties **i.** and **ii.** establish that $\left((A_t)_{t \in Z}, \overset{\leftarrow}{\circ}, A_{\mathbb{0}}\right)$ forms a monoid. Second, property **vii.** characterizes a Feller semigroup as a family of operators acting on $C^0(X)$; certain formulations in the literature restrict to endomorphisms of $C^0(X)$ by omitting this regularity condition, though our definition includes it for greater generality. Third, property **viii.** ensures that the mapping $t \mapsto A_t f$ is uniformly continuous for all $f \in B(X)$. This follows from the inequality: for all $t, s \in Z$,

$$\lim_{t \downarrow \mathbb{0}} \left\| A_{s+t} f - A_s f \right\|_\infty = \lim_{t \downarrow \mathbb{0}} \|A_s (A_t f - f)\|_\infty = 0.$$

Finally, when $(A_t)_{t \in Z}$ denote the operators associated with a family of Markov kernels according to Definition B.A.22, properties **i.** through **vi.** of the present definition are satisfied.

LEAN formalization. The LEAN formalization of a Feller semigroup is made through the `FellerSemigroup` type class that extends the `OperatorsMonoid` type class to include the additional properties of a Feller semigroup.

LEAN

```
class FellerSemigroup {G X : Type*} [AddSemigroup Z]
    [Zero Z] [TopologicalSpace Z] [MeasurableSpace X]
    [TopologicalSpace X] (A : Z → (X →ₘ ℝ) → X →ₘ ℝ)
    extends OperatorsMonoid where
  preserve_mass t : A t 1 = 1
  non_negative t f : 0 ≤ f → 0 ≤ A t f
  contraction t f : ‖A t f‖ₑ ≤ ‖f‖ₑ
  continuous t (f : X →ₘ ℝ) :
    Continuous f → Continuous (A t f)
  strongly_continuous f :
    Tendsto (fun t ↦ ‖A t f - f‖ₑ) (𝓝 0) (𝓝 0)
```

Thus, any family of operators $\left((Q_t)_{t\in Z}, \overleftarrow{\circ}\right)$ associated to a family of Markov kernels such that properties **vii.** and **viii.** of Definition B.A.25 are satisfied is a Feller semigroup, as the `preserve_mass` and `contraction` properties follow from the Markov kernel structure.

Infinitesimal generator. For any Feller semigroup $\left((A_t)_{t\in Z}, \overleftarrow{\circ}\right)$, the infinitesimal generator is defined as the operator that encodes the complete specification of the semigroup. This operator (or its adjoint) solves the Kolmogorov forward and backward equations. The generator arises canonically in the theory of standard stochastic differential equations, where it coincides with the solution of the associated Fokker-Planck equation.

Definition B.A.26 – *Infinitesimal generator*

Let $(G,+)$ a semigroup and $\left((A_t)_{t\in Z}, \overleftarrow{\circ}\right)$ a Feller semigroup on $B(X)$. Let

$$D(L) \stackrel{\text{def}}{=} \left\{ f \in C^0(X) \;\middle|\; \lim_{t\downarrow 0} \frac{A_t f - f}{t} \text{ uniformly converges in } C^0(X) \right\}$$

and, for any $f \in D(L)$, we denote by *infinitesimal generator* of $(A_t)_{t\in Z}$ the operator

$$Lf \stackrel{\text{def}}{=} \lim_{t\downarrow 0} \frac{A_t f - f}{t}.$$

Let L^* be the adjoint of L in the distributional sense, i.e.,

$$\forall \varphi \in D(L), \mu \in \mathcal{P}(X), \langle L\varphi, \mu\rangle = \langle \varphi, L^*\mu\rangle.$$

Given the infinitesimal generator L of a Feller semigroup, the semigroup itself can be recovered via the λ-resolvent.

Theorem B.A.18

Let $(Q_t)_{t\in\mathbb{R}_+}$ be a Feller semigroup. It can be recovered from its infinitesimal generator L and its domain $D(L)$ via the λ-resolvent.

The proof of this result relies on the properties of the λ-resolvent and is detailed in Le Gall (2016).

Let $(X_t)_{t\geq 0}$ be a Feller process with associated family of Markov kernels $(Q_t)_{t\geq 0}$. The infinitesimal generator of the associated Feller semigroup and its adjoint are solution of the Kolmogorov backward and forward equations, respectively

$$\forall \varphi \in D(L), \frac{\mathrm{d}}{\mathrm{d}t}\langle \mu_t, \varphi \rangle = \langle L\varphi, \mu_t \rangle$$

and

$$\partial_t \mu_t = L^* \mu_t,$$

where $\mu_t \overset{\text{def}}{=} \mathcal{L}(X_t)$ is the distribution of X_t.

When $(X_t)_{t\geq 0}$ is the solution of a standard SDE with Lipschitz continuous coefficients, the process is Feller and its infinitesimal generator L takes the explicit form of a second-order differential operator. The Kolmogorov forward equation $\partial_t \mu_t = L^* \mu_t$ then reduces to the classical Fokker-Planck equation, which provides a PDE characterization of the evolution of $\mu_t \overset{\text{def}}{=} \mathcal{L}(X_t)$ in terms of the drift and diffusion coefficients of the SDE. Moreover, Theorem B.A.18 guarantees that the Markov kernels $(Q_t)_{t\geq 0}$ describing the transitions of $(X_t)_{t\geq 0}$ can be fully recovered from L and its domain $D(L)$, so that the SDE, its generator, and its transition kernels are three equivalent descriptions of the same object.

A.5 Family of non-linear Markov kernels

The theory of Markov kernels developed in the preceding sections provides a characterization of classical stochastic processes through their transition probabilities. However, this framework is fundamentally insufficient for McKean-Vlasov SDEs, where the evolution of a particle's distribution depends explicitly on the empirical law of the entire system at that time. Specifically, let $(X_t)_{t\geq 0}$ be a solution to a McKean-Vlasov SDE as in Definition A.3.8. The transition probabilities from X_t to X_s with $s > t$ are not determined solely by the current state X_t; they fundamentally depend on the entire distribution $\mathcal{L}(X_t)$ at time t. This law-dependence makes the classical Markov kernel framework inapplicable, as it would require specifying a single kernel transition for each initial state, whereas the transition here varies across the distribution of initial states.

To address this fundamental limitation, we introduce *non-linear Markov kernels* — probability kernels indexed not merely by states, but by probability measures. These kernels generalize the classical framework by making explicit the dependence of transitions on the distributional structure of the process. In this section, we establish that families of non-linear Markov kernels inherit algebraic structures analogous to that of classical families (see Definition B.A.22), enabling McKean-Vlasov-type processes to be fully characterized through their transition law-kernels. The formal development is motivated by and adapts the framework of

Chaintron & Diez (2022), which draws on the classical theory of McKean-Vlasov SDEs such as Sznitman (2006).

Definition B.A.27 – *Non-linear Markov kernel*

Given two measurable spaces $(X, \mathscr{X})$ and $(Y, \mathscr{Y})$, a non-linear Markov kernel is a function $\kappa : \mathcal{P}(X) \to X \to \mathcal{P}(Y)$ such that

$$\forall A \in \mathscr{Y}, (\mu, x) \mapsto \kappa^\mu(x, A) \text{ is measurable.}$$

In a similar way as for classical Markov kernels, we can associate to a non-linear Markov kernel κ a linear operator acting on the space of bounded strongly-measurable functions on Y. For any $f \in B(Y)$, any $\mu \in \mathcal{P}(X)$ and any $x \in X$, this operator is defined by

$$\kappa^\mu f \overset{\text{def}}{=} \mathbb{E}_{y \sim \kappa^\mu(x)}[f(y)] = \int_Y f(y) \, \mathrm{d}\kappa^\mu(x).$$

Moreover, this operator also preserves the boundedness and strong measurability of functions, i.e. $\kappa^\mu f \in B(Y)$.

Definition B.A.28 – *Family of non-linear Markov kernels*

Let $(X, \mathscr{X})$ be a measurable space and $(G, +, \mathscr{Z})$ a measurable semigroup. A family of non-linear kernels $\left(Q_t^{(\cdot)} : \mathcal{P}(X) \to X \rightsquigarrow X\right)_{t \in Z}$ is a family of Markov kernels if

i. for any $t \in Z$ and any $\mu \in \mathcal{P}(X)$ Q_t^μ is a Markov kernel;

ii. there exists $\mathbb{0} \in Z$ such that, for any $\mu \in \mathcal{P}(X)$ and any $x \in X$, $Q_{\mathbb{0}}^\mu(x) = \delta_x$;

iii. for any $t, s \in Z$, $\mu \in \mathcal{P}(X)$, $x \in X$, and $A \in \mathscr{X}$,

$$Q_{t+s}^\mu(x, A) = \int_X Q_t^{\overline{S}_s(\mu)}(y, A) \, \mathrm{d}Q_s^\mu(x),$$

where, for any $A \in \mathscr{X}$, $\overline{S}_s(\mu, A) \overset{\text{def}}{=} \int_X Q_s^\mu(x) \, \mathrm{d}\mu$. This operator represents the average transition of the distribution μ after time s.

iv. for any $A \in \mathscr{X}$, the function $(t, \mu, x) \mapsto Q_t^\mu(x, A)$ is measurable w.r.t. the product σ-algebra $\mathscr{Z} \otimes \mathfrak{P}(X) \otimes \mathscr{X}$.

The property **iii.** is a non-linear version of the Chapman-Kolmogorov equation, where the transition from x to A after time $t + s$ depends on the initial distribution μ and the average transition of μ after time s. One can use this eqution to define a composition between non-linear kernels such that **iii.** can be rewritten more concisely

$$Q_{t+s}^{(\cdot)} = Q_t^{(\cdot)} \otimes_\mu Q_s^{(\cdot)} \overset{\text{def}}{=} \mu \mapsto \left(Q_t^{\overline{S}_s(\mu)} \otimes_k Q_s^\mu\right).$$

Following the same idea, one can also define a reverse composition between operators associated to non-linear kernels

$$Q_t^{(\cdot)} \overset{\leftarrow}{\circ}_\mu Q_s^{(\cdot)} \overset{\text{def}}{=} \mu \mapsto \left(Q_t^{\overline{S}_s(\mu)} \overset{\leftarrow}{\circ} Q_s^{\mu}\right).$$

LEAN formalization. We proceed similarly to the formalization of a family of Markov kernels, introducing the simple `DProbKernels` structure to represent a family of non-linear Markov kernels indexed by a semigroup.

LEAN

```
structure DProbKernels (G X : Type*) [AddSemigroup Z]
    [MeasurableSpace X] where
  toFun : Z → Measure X → Kernel X X
  markov_kernels' t μ : IsMarkovKernel (toFun t μ)
```

Then, we include properties **ii.**, **iii.** and **iv.** of Definition B.A.28 in the `DMarkovFamily` structure, which extends `DProbKernels` to represent a family of non-linear Markov kernels.

LEAN

```
structure DMarkovFamily (G X : Type*) [AddSemigroup Z]
    [MeasurableSpace Z] [Zero Z] [MeasurableSpace X]
    extends DProbKernels Z X where
  kernel_dirac' μ : toFun 0 μ = Kernel.dirac
  chapman_kolmogorov' t s μ : toFun (t + s) μ =
    (toFun t ◁ (toFun s μ).avg μ) ∘ₖ toFun s μ
  measurable' : Measurable (fun (t, μ, x) ⇒ toFun μ t x)
```

A.6 Non-linear transition monoids

Let $(G, +, \mathcal{Z})$ be a measurable semigroup and $\left(Q_t^{(\cdot)}\right)_{t\in Z}$ a family of non-linear Markov kernels on $(X, \mathcal{X})$. By using the same arguments as in Appendix A.2, one can show that $\left(\left(Q_t^{(\cdot)}\right)_{t\in Z}, \otimes_\mu\right)$ and $\left(\left(Q_t^{(\cdot)}\right)_{t\in Z}, \overset{\leftarrow}{\circ}_\mu\right)$ are monoids. The proofs are omitted here to avoid redundancy but can be found in the GitHub repository of the project.

LEAN formalization. We proceed analogously to the formalization of the transition monoid associated with a family of Markov kernels. To this end, we introduce the `DKernelsMonoid` type class that requires that the addition of two indexes corresponds to the composition of the associated non-linear kernels and that the kernel associated to $\mathbb{0}$ is the dirac kernel.

LEAN

```
class DKernelsMonoid (Q : Z → Measure X → Kernel X X) where
  map_add t s μ : Q (t + s) μ
    = (Q t ◁ (Q s μ).avg μ) ∘ₖ Q s μ
  dirac_0 μ : Q 0 μ = Kernel.dirac
```

This type class provides the structural conditions under which $\left(\left(Q_t^{(\cdot)}\right)_{t\in Z}, \otimes_\mu\right)$ can be endowed with a monoid structure, where Q_0^μ serves as the neutral element. Moreover, any family satisfying the properties set forth in Definition B.A.28 instantiates `DKernelsMonoid`, and consequently inherits this monoid structure.

```
instance : DKernelsMonoid Q where
  map_add t s := Q.chapman_kolmogorov t s
  dirac_0 := Q.kernel_dirac
```

Turning to the monoid structure of $\left(\left(Q_t^{(\cdot)}\right)_{t\in Z}, \overleftarrow{\circ}_\mu\right)$, we introduce the `DOperatorsMonoid` type class, which requires that the addition of two indices correspond to the reverse composition of the associated operators, mediated by an averaging function satisfying the property that averaging over Q_0^μ recovers μ itself. This abstract condition is necessary to establish that Q_0^μ acts as a neutral element with respect to reverse composition, and is satisfied in particular by the operator $\overline{S}_0$ introduced in Definition B.A.28.

```
class DOperatorsMonoid
    (Q : Z → Measure X → (X →ₘ ℝ) → X →ₘ ℝ) where
  avg : Measure X → Q.image → Measure X
  map_add t s μ : Q (t + s) μ =
    Q s μ ∘ (Q t ◁ avg μ ◁ Q.elem s)
  id_0 μ f : Q 0 μ f = f
  avg_0 μ : (avg μ ◁ Q.elem 0) = μ
```

This type class gives rise to a monoid structure for the family of operators, and is instantiated by any family of operators arising from a family of non-linear Markov kernels in the sense of Definition B.A.28.

```
instance : DOperatorsMonoid
    (fun a μ ↦ (Q a μ).operator) where  ...
```

The proof of this instance is omitted here; a complete formalization is available in the project's GitHub repository.

A.7 Non-linear Markov processes

Let $\left(X, \mathscr{X}, (\mathscr{F}_t)_{t\in\overline{\mathbb{R}}_+}, \mathbb{P}\right)$ be a filtered probability space. We define a *non-linear Markov process* on this space.

Definition B.A.29 – *Non-linear Markov process*

Let $\left(Q_t^{(\cdot)}\right)_{t\in\overline{\mathbb{R}}_+}$ be a family of Markov kernels on $(X,\mathscr{X})$ as in Definition B.A.28. A non-linear Markov process associated to this family is a stochastic process $(X_t)_{t\in\overline{\mathbb{R}}_+}$ that is $\mathscr{F}_t$-adapted and such that, for any $t,s\geq 0$ and any $f\in B(X)$,

$$\mathbb{E}[f(X_{t+s}) \mid \mathscr{F}_t] = Q_s^{\mu_t} f(X_t),$$

where $\mu_t \overset{\text{def}}{=} \mathcal{L}(X_t)$.

Remark

When considering a non-linear Markov process, the definition of $\overline{S}_t$ becomes more intuitive. Indeed, suppose that $X_0\sim\mu_0$. Then, for any $t\geq 0$, we have

$$\mathcal{L}(X_t) = \int_X \mathcal{L}\mid \mathcal{X}_0 = x \,\mathrm{d}\mu_0 = \int_X Q_t^{\mu_0}(x)\,\mathrm{d}\mu_0 = \overline{S}_t(\mu_0).$$

One can easily see that $\overline{S}_t(\mu)$ represents the law of X_t, whenever $X_0\sim\mu$. Therefore, the composition $Q_t^{\overline{S}_s(\mu)} \otimes_k Q_s^{\mu}$ represents a "jump" of time s depending on μ, followed by a "jump" of time t depending on the distribution of the state after time s.

A.8 Non-linear Feller processes and their generators

The notion of Feller semigroup and infinitesimal generator can be extended to the non-linear setting. The definitions are similar to the ones given in Definition B.A.25 and Definition B.A.26, but with the composition of kernels and operators replaced by their non-linear counterparts.

Definition B.A.30 – *Non-linear Feller semigroup*

Let $(G,+)$ a semigroup and $\left(A_t^{(\cdot)}\right)_{t\in Z}$ a family of linear operators on $\mathcal{P}(X)\times B(X)$. Let $t,s\in Z$, $\mu\in\mathcal{P}(X)$, and $f\in B(X)$. The family $\left(\left(A_t^{(\cdot)}\right)_{t\in Z}, \overset{\leftarrow}{\circ}_\mu\right)$ is a non-linear Feller semigroup if

- **i.** $A_{t+s}^{(\cdot)} = A_t^{(\cdot)} \overset{\leftarrow}{\circ}_\mu A_s^{(\cdot)}$;
- **ii.** there exists $\mathbb{0}\in Z$ such that, $A_{\mathbb{0}}^{\mu} f = f$;
- **iii.** $A_t^{\mu} f \in B(X)$;
- **iv.** $\left\|A_t^{\mu} f\right\|_\infty \leq \|f\|$;
- **v.** $A_t^{\mu}\mathbf{1} = \mathbf{1}$;
- **vi.** if $f\geq 0$, then $A_t^{\mu} f\geq 0$;
- **vii.** if f is continuous, then $A_t^{\mu} f$ is continuous;
- **viii.** $\left\|A_t^{\mu} f - f\right\|_\infty \underset{t\downarrow\mathbb{0}}{\to} 0$.

Analogously to Definition B.A.25, properties **i.** through **vi.** of the present definition are satisfied by any family of operators $\left(\left(Q_t^{(\cdot)}\right)_{t\in Z}, \overleftarrow{\circ}_\mu\right)$ associated to a family of non-linear Markov kernels.

LEAN formalization. The LEAN formalization of a non-linear Feller semigroup is made through the `DFellerSemigroup` type class that extends the `DOperatorsMonoid` type class to include the additional properties of a non-linear Feller semigroup.

LEAN

```
class DFellerSemigroup {G X : Type*} [AddSemigroup Z]
    [Zero Z] [TopologicalSpace Z] [MeasurableSpace X]
    [TopologicalSpace X]
    (A : Z → Measure X → (X →ₘ ℝ) → X →ₘ ℝ)
    extends DOperatorsMonoid A where
  preserve_mass t μ : A t μ 1 = 1
  non_negative t μ f : 0 ≤ f → 0 ≤ A t μ f
  contraction t μ f : ‖A t μ f‖ₑ ≤ ‖f‖ₑ
  continuous t μ (f : X →ₘ ℝ) :
    Continuous f → Continuous (A t μ f)
  strongly_continuous μ f :
    Tendsto (fun t ↦ ‖A t μ f - f‖ₑ) (𝓝 0) (𝓝 0)
```

Similarly, any family of operators $\left(\left(Q_t^{(\cdot)}\right)_{t\in Z}, \overleftarrow{\circ}_\mu\right)$ associated to a family of non-linear Markov kernels such that properties **vii.** and **viii.** of Definition B.A.30 are satisfied is a non-linear Feller semigroup, as the `preserve_mass` and `contraction` properties follow from the Markov kernel structure.

Non-linear infinitesimal generator.

Definition B.A.31 – *Infinitesimal generator*

Let $(G,+)$ a semigroup and $\left(\left(A_t^{(\cdot)}\right)_{t\in Z}, \overleftarrow{\circ}\right)$ a non-linear Feller semigroup on $B(X)$. For any $\mu \in \mathcal{P}(X)$, let

$$D(L^\mu) \stackrel{\text{def}}{=} \left\{ f \in C^0(X) \;\middle|\; \lim_{t\downarrow 0} \frac{A_t^\mu f - f}{t} \text{ uniformly converges in } C^0(X) \right\}$$

and, for any $f \in D(L^\mu)$, we denote by *non-linear infinitesimal generator* of $\left(A_t^{(\cdot)}\right)_{t\in Z}$ the operator

$$L^\mu f \stackrel{\text{def}}{=} \lim_{t\downarrow 0} \frac{A_t^\mu f - f}{t}.$$

Let L_μ^* be the adjoint of L^μ in the distributional sense, i.e.,

$$\forall \varphi \in D(L^\mu), \mu \in \mathcal{P}(X), \langle L^\mu \varphi, \mu\rangle = \langle \varphi, L_\mu^* \mu\rangle.$$

Let $(X_t)_{t\geq 0}$ be a non-linear Feller process with associated non-linear family of Markov kernels $\left(Q_t^{(\cdot)}\right)_{t\geq 0}$. It is classical to suppose that the domain of the non-linear generator L^μ does not depend on μ, so that we can simply write $D(L)$ instead of $D(L^\mu)$ (see Chaintron & Diez (2022)). Under this assumption, the non-

linear infinitesimal generator L and its adjoint L^* are solution of the non-linear Kolmogorov backward and forward equations, respectively

$$\forall \varphi \in D(L), \frac{\mathrm{d}}{\mathrm{d}t}\langle \mu_t, \varphi \rangle = \langle L_t^{\mu} \varphi, \mu_t \rangle$$

and

$$\partial_t \mu_t = L_{\mu_t}^* \mu_t.$$

When $(X_t)_{t\geq 0}$ is the solution of a McKean-Vlasov SDE with globally Lipschitz coefficients, by analogy with the linear case, the process is a non-linear Feller process, and the non-linear Kolmogorov forward equation reduces to the McKean-Vlasov-Fokker-Planck equation. In this setting, the coefficients of the generator and the associated PDE depend explicitly on the distribution of the process, reflecting the fundamental distinction between linear and non-linear Markov kernels. The family of non-linear Markov kernels $\left(Q_t^{(\cdot)}\right)_{t\geq 0}$ can be fully recovered from the non-linear generator (see Chaintron & Diez (2022) for details).

For McKean-Vlasov-based algorithms discretized into N particles, transition probabilities depend on the empirical distribution $\hat{\mu}_t$ of the system. However, this non-linear dependency can be eliminated by reformulating the problem at the system level: the state space becomes the product space $\Omega = X^N$, and the classical transition kernels on Ω^{n+1} automatically encode these dependencies through the system's complete history. This reformulation makes McKean-Vlasov-based algorithms directly expressible within the abstract framework of Definition B.1.16.

Bibliography

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631.

Akimoto, Y., Auger, A., & Hansen, N. (2014). Comparison-based natural gradient optimization in high dimension. *Annual Conference on Genetic and Evolutionary Computation*, 373–380.

Al-Dujaili, A., & Suresh, S. (2018). Multi-objective simultaneous optimistic optimization. *Information Sciences*, *424*, 159–174.

Alon, N., Bloom, T. F., Gowers, W., Litt, D., Sawin, W., Shankar, A., Tsimerman, J., Wang, V., & Wood, M. M. (2026). Remarks on the disproof of the unit distance conjecture. *Arxiv Preprint Arxiv:2605.20695*.

Ambrosio, L., Gigli, N., & Savaré, G. (2005). *Gradient flows: in metric spaces and in the space of probability measures*. Springer.

Aronszajn, N. (1950). *Theory of Reproducing Kernels*. Transactions of the American Mathematical Society.

Asgeirsson, D., Brasca, R., Kuhn, N., Majno, F. A. E. N. M., Capriglio, D., & Topaz, A. (2024). Categorical Foundations of Formalized Condensed Mathematics. *The Journal of Symbolic Logic*, 1–28.

Asuncion, A., Newman, D., & others. (2007). *UCI machine learning repository*. Irvine, CA, USA.

Awasthi, P., Mao, A., Mohri, M., & Zhong, Y. (2024). DC-programming for neural network optimizations. *Journal of Global Optimization*, 1–17.

Azencott, R. (1989). *Simulated annealing*. Séminaire Bourbaki.

Barco, V. del, Infanti, G., Rivas, E., & Schwahn, P. (2026). Formalizing a classification theorem for low-dimensional solvable lie algebras in lean. In *Lecture Notes in Computer Science* (pp. 317–339). Springer Nature Switzerland.

Berlinet, A., & Thomas-Agnan, C. (2011). *Reproducing kernel Hilbert spaces in probability and statistics*. Springer.

Bianchi, P., Dragomir, R.-A., & Priser, V. (2025). Consensus-Based Optimization Beyond Finite-Time Analysis. *Arxiv Preprint Arxiv:2509.12907*.

Billingsley, P. (1999). *Convergence of Probability Measures*. Wiley.

Biscani, F., & Izzo, D. (2020). A parallel global multiobjective framework for optimization: pagmo. *Journal of Open Source Software*, *5*(53), 2338. https://doi.org/10.21105/joss.02338

Bobbin, M. P., Sharlin, S., Feyzishendi, P., Dang, A. H., Wraback, C. M., & Josephson, T. R. (2024). Formalizing chemical physics using the Lean theorem prover. *Digital Discovery*, *3*(2), 264–280.

Boltzmann, L. (1970). Weitere studien über das wärmegleichgewicht unter gasmolekülen. In *Kinetische Theorie II: Irreversible Prozesse Einführung und Originaltexte* (pp. 115–225). Springer.

Boneh, A., & Golan, A. (1979). Constraints' redundancy and feasible region boundedness by random feasible point generator (RFPG). *Third European Congress on Operations Research (EURO III), Amsterdam*.

Borghi, G., Herty, M., & Pareschi, L. (2023). An adaptive consensus based method for multi-objective optimization with uniform Pareto front approximation. *Applied Mathematics & Optimization*, *88*(2), 58.

Braun, C. V., Lange, R. T., & Toussaint, M. (2024). Stein variational evolution strategies. *Arxiv Preprint Arxiv:2410.10390*.

Brent, R. P. (1973). *Algorithms for Minimisation without Derivatives*. Prentice Hall.

Brochu, E., Cora, V. M., & De Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Arxiv Preprint Arxiv:1012.2599*.

Bungert, L., Roith, T., & Wacker, P. (2025). Polarized consensus-based dynamics for optimization and sampling. *Mathematical Programming*, *211*(1), 125–155.

Cappé, O., Douc, R., Guillin, A., Marin, J.-M., & Robert, C. P. (2008). Adaptive importance sampling in general mixture classes. *Statistics and Computing*, *18*(4), 447–459.

Carathéodory, C. (1918). *Vorlesungen über reelle Funktionen*. B. G. Teubner-Verlag.

Carlier, R. (2026). Unbiasing symmetric monoidal categories in Lean. *Arxiv Preprint Arxiv:2603.00896*.

Carmona, R., & Delarue, F. (2018). *Probabilistic Theory of Mean Field Games with Applications* (Vol. 2). Springer.

Carmona, R., Delarue, F., & others. (2018). *Probabilistic theory of mean field games with applications I-II* (Vol. 3). Springer.

Carneiro, M., & Riehl, E. (2025). Formalizing Colimits in $\mathcal{C}$at. In Y. Forster & C. Keller (Eds.), *International Conference on Interactive Theorem Proving* (Vol. 352, p. 20:1–20:19). Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

Carrillo, J. A., Choi, Y.-P., Totzeck, C., & Tse, O. (2018). An analytical framework for consensus-based global optimization method. *Mathematical Models and Methods in Applied Sciences*, *28*(06), 1037–1066.

Carrillo, J. A., Jin, S., Li, L., & Zhu, Y. (2021). A consensus-based global optimization method for high dimensional machine learning problems. *ESAIM: Control, Optimisation and Calculus of Variations*, *27*, S5.

Chaintron, L.-P., & Diez, A. (2022). Propagation of chaos: A review of models, methods and applications. I. Models and methods. *Kinetic and Related Models*, *15*(6), 895.

Chao, Y., Sun, X., Riedl, K., Khadiv, M., & Etesami, J. (2026). Convergence of Consensus-Based Particle Methods for Nonconvex Bi-Level Optimization. *Arxiv Preprint Arxiv:2605.19667*.

Cho, K., & Jacobs, B. (2019). Disintegration and Bayesian inversion via string diagrams. *Mathematical Structures in Computer Science*, *29*(7), 938–971.

Clémençon, S., Lugosi, G., & Vayatis, N. (2008). Ranking and Empirical Minimization of U-statistics. *The Annals of Statistics*, *36*(2).

Conn, A. R., Scheinberg, K., & Vicente, L. N. (2009). *Introduction to derivative-free optimization*. SIAM.

The Coq Development Team. (2024). *The Coq Proof Assistant*. Zenodo.

Coquand, T., & Huet, G. (1988). The calculus of constructions. *Information and Computation*, *76*(2), 95–120.

Cuchiero, C., Di Persio, L., Guida, F., & Svaluto-Ferro, S. (2024). Measure-valued affine and polynomial diffusions. *Stochastic Processes and Their Applications*, *175*, 104392.

Da Prato, G., & Zabczyk, J. (2014). *Stochastic equations in infinite dimensions* (Vol. 152). Cambridge University Press.

Dawson, D. A., & others. (1994). *Measure-valued processes, stochastic partial differential equations, and interacting systems* (Vol. 5). American Mathematical Soc.

Degenne, R. (2025). Markov kernels in Mathlib's probability library. *Preprint Arxiv:2510.04070*.

Del Moral, P. (2004). Feynman-kac formulae. In *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications* (pp. 47–93). Springer.

Del Moral, P. (2013). Mean field simulation for Monte Carlo integration. *Monographs on Statistics and Applied Probability*, *126*(26), 6.

Delarue, F., & Hammersley, W. R. (2025). Rearranged stochastic heat equation. *Probability Theory and Related Fields*, *191*(1), 41–102.

Delarue, F., & Vasileiadis, A. (2025). Exploration noise for learning linear-quadratic mean field games. *Mathematics of Operations Research*, *50*(3), 1762–1831.

Delarue, F., Tanré, E., & Maillet, R. (2024). Ergodicity of some stochastic Fokker-Planck equations with additive common noise. *Preprint Arxiv:2405.09950*.

Doorn, F. van, Massot, P., & Nash, O. (2023). Formalising the h-principle and sphere eversion. *Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs*, 121–134.

Duncan, A., Nüsken, N., & Szpruch, L. (2023a). On the geometry of Stein variational gradient descent. *Journal of Machine Learning Research*, *24*(56), 1–39.

Duncan, A., Nüsken, N., & Szpruch, L. (2023b). On the geometry of Stein variational gradient descent. *Journal of Machine Learning Research*, *24*(56), 1–39.

Durmus, A., & Moulines, É. (2017). Nonasymptotic convergence analysis for the unadjusted Langevin algorithm. *The Annals of Applied Probability*, *27*(3), 1551–1587.

El Yadari, M., El Motaki, S., Yahyaouy, A., Makany, P., El Fazazy, K., Gualous, H., & Le Masson, S. (2024). Taxonomy of optimization algorithms combined with CNN for optimal placement of virtual machines within physical machines in data centers. *Energy Informatics*, *7*(1), 107.

Erny, X. (2025). Generators of measure-valued jump diffusions and convergence rate of diffusive mean-field models. *Electronic Journal of Probability*, *30*, 1–44.

Estrade, A., & Fournier, J. (2025). Gaussian Fields Through Geometrical Properties. In *Stochastic Geometry: Percolation, Tesselations, Gaussian Fields and Point Processes* (pp. 81–131). Springer.

Etxeberria, R. (1999). Global optimization using Bayesian networks. *II Symposium on Artificial Intelligence*, 332–339.

Evans, L. C. (2025). *Measure theory and fine properties of functions*. Chapman, Hall/CRC.

Fang, J., Liu, W., Chen, L., Lauria, S., Miron, A., & Liu, X. (2023). *A survey of algorithms, applications and trends for particle swarm optimization.*

Folland, G. B. (1999). *Real analysis: modern techniques and their applications*. John Wiley & Sons.

Fornasier, M., Klock, T., & Riedl, K. (2022). Convergence of anisotropic consensus-based optimization in mean-field law. *International Conference on the Applications of Evolutionary Computation (Part of Evostar)*, 738–754.

Fornasier, M., Klock, T., & Riedl, K. (2024). Consensus-Based Optimization Methods Converge Globally. *SIAM Journal on Optimization*, *34*(3), 2973–3004.

Fornasier, M., Pareschi, L., Huang, H., & Sünnen, P. (2021). Consensus-based optimization on the sphere: Convergence to global minimizers and machine learning. *Journal of Machine Learning Research*, *22*(237), 1–55.

Fourati, F., Alouini, M.-S., & Aggarwal, V. (2026). Ecpv2: Fast, efficient, and scalable global optimization of lipschitz functions. *AAAI Conference on Artificial Intelligence*, *40*(43), 36909–36918.

Fourati, F., Kharrat, S., Aggarwal, V., & Alouini, M.-S. (2025). Every Call is Precious: Global Optimization of Black-Box Functions with Unknown Lipschitz Constants. In Y. Li, S. Mandt, S. Agrawal, & E. Khan (Eds.), *International Conference on Artificial Intelligence and Statistics* (Vol. 258, pp. 5176–5184). PMLR.

Franceschi, L., Donini, M., Perrone, V., Klein, A., Archambeau, C., Seeger, M., Pontil, M., & Frasconi, P. (2024). Hyperparameter optimization in machine learning. *Preprint Arxiv:2410.22854*.

Fremlin, D. H. (2003). *Broad foundations*. Torres Fremlin.

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

Gallego, V., & Insua, D. R. (2018). Stochastic gradient MCMC with repulsive forces. *Preprint Arxiv:1812.00071*.

Gao, X., Gürbüzbalaban, M., & Zhu, L. (2020). Breaking reversibility accelerates langevin dynamics for non-convex optimization. *International Conference on Neural Information Processing Systems*.

Gaviano, M., Kvasov, D. E., Lera, D., & Sergeyev, Y. D. (2003). Algorithm 829: Software for generation of classes of test functions with known local and global minima for global optimization. *ACM Transactions on Mathematical Software (TOMS)*, *29*(4), 469–480.

Gerber, N. J., Hoffmann, F., & Vaes, U. (2025). Mean-field limits for consensus-based optimization and sampling. *ESAIM: Control, Optimisation and Calculus of Variations*, *31*, 74.

Germain, P., & Monmarché, P. (2025). Stochastic moments dynamics: a flexible finite-dimensional random perturbation of Wasserstein gradient descent. *Preprint Arxiv:2505.07448*.

Giry, M. (1982). A categorical approach to probability theory. In *Categorical Aspects of Topology and Analysis* (pp. 68–85). Springer Berlin Heidelberg.

Gomez, J. (2019). Stochastic global optimization algorithms: A systematic formal approach. *Information Sciences*, *472*, 53–76.

Gonthier, G., & others. (2008). Formal proof–the four-color theorem. *Notices of the AMS*, *55*(11), 1382–1393.

Gonthier, G., Asperti, A., Avigad, J., Bertot, Y., Cohen, C., Garillot, F., Le Roux, S., Mahboubi, A., O'Connor, R., Ould Biha, S., & others. (2013). A machine-checked proof of the odd order theorem. *International Conference on Interactive Theorem Proving*, 163–179.

Gorham, J., & Mackey, L. (2015). Measuring sample quality with Stein's method. *International Conference on Neural Information Processing*, 226–234.

Gouëzel, S. (2026). Higher order differential calculus in mathlib. *ACM SIGPLAN International Conference on Certified Programs and Proofs*, 1–14.

Gowers, W. T., Green, B., Manners, F., & Tao, T. (2025). On a conjecture of Marton. *Annals of Mathematics*, *201*(2), 515–549.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research*, *13*, 723–773.

Hájek, P. (2001). *Metamathematics of fuzzy logic* (Vol. 4). Springer Science & Business Media.

Hales, T., Adams, M., Bauer, G., Dang, T. D., Harrison, J., Hoang, L. T., Kaliszyk, C., Magron, V., McLaughlin, S., Nguyen, T. T., & others. (2017). A formal proof of the Kepler conjecture. *Forum of Mathematics, Pi*, *5*, e2.

Han, J. M., & Doorn, F. van. (2020). A formal proof of the independence of the continuum hypothesis. *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, 353–366.

Han, J., & Liu, Q. (2018). Stein variational gradient descent without gradient. *International Conference on Machine Learning*, 1900–1908.

Hansen, N. (2006). The CMA evolution strategy: a comparing review. *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms*, 75–102.

Hansen, N. (2023). The CMA Evolution Strategy: A Tutorial. *Preprint Arxiv:1604.00772*.

Hansen, N., & Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. *IEEE International Conference on Evolutionary Computation*, 312–317.

Hansen, N., & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, *9*(2), 159–195.

Hansen, N., Müller, S. D., & Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, *11*(1), 1–18.

Hartman, P. (2002). *Ordinary differential equations*. SIAM.

Hooke, R., & Jeeves, T. A. (1961). "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the ACM (JACM)*, *8*(2), 212–229.

Houssein, E. H., Gad, A. G., Hussain, K., & Suganthan, P. N. (2021). Major advances in particle swarm optimization: theory, analysis, and application. *Swarm and Evolutionary Computation*, *63*, 100868.

Houssein, E. H., Saeed, M. K., Hu, G., & Al-Sayed, M. M. (2024). Metaheuristics for Solving Global and Engineering Optimization Problems: Review, Applications, Open Issues and Challenges: EH Houssein et al. *Archives of Computational Methods in Engineering*, *31*(8), 4485–4519.

Hwang, C.-R. (1980). Laplace's method revisited: weak convergence of probability measures. *The Annals of Probability*, 1177–1182.

Jastrebski, G. A., & Arnold, D. V. (2006). Improving evolution strategies through active covariance matrix adaptation. *2006 IEEE International Conference on Evolutionary Computation*, 2814–2821.

Johnson, S. G. (2007). *The NLopt nonlinear-optimization package*.

Jones, D. R. (2001). Direct global optimization algorithm. *Encyclopedia of Optimization*, 431–440.

Jones, D. R., Perttunen, C. D., & Stuckman, B. E. (1993). Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, *79*(1), 157–181.

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, *13*(4), 455–492.

Kallenberg, O. (2021). *Foundations of Modern Probability* (3rd ed.). Springer Nature.

Kandasamy, K., Schneider, J., & Póczos, B. (2015). High dimensional Bayesian optimisation and bandits via additive models. *International Conference on Machine Learning*, 295–304.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *International Conference on Neural Networks*, *4*, 1942–1948.

Khalil, H. K. (2002). *Nonlinear systems*. Prentice Hall.

Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680.

Kitagawa, G. (1993). A Monte Carlo filtering and smoothing method for non-Gaussian nonlinear state space models. *Proceedings of the 2nd US-Japan Joint Seminar on Statistical Time Series Analysis*, *110*.

Kolmogorov, A. (1933). *Grundbegriffe der wahrscheinlichkeitsrechnung*.

Korba, A., Aubin-Frankowski, P.-C., Majewski, S., & Ablin, P. (2021). Kernel stein discrepancy descent. *International Conference on Machine Learning*, 5719–5730.

Korba, A., Salim, A., Arbel, M., Luise, G., & Gretton, A. (2020). A non-asymptotic analysis for Stein variational gradient descent. *Advances in Neural Information Processing Systems*, *33*, 4672–4682.

Larranaga, P., & Bielza, C. (2025). Estimation of distribution algorithms. In *Handbook of Heuristics* (pp. 583–598). Springer.

Le Gall, J.-F. (2016). *Brownian motion, martingales, and stochastic calculus* (Vol. 274). Springer.

Lee, J., Lee, I.-H., Joung, I., Lee, J., & Brooks, B. R. (2017). Finding multiple reaction pathways via global optimization of action. *Nature Communications*, *8*(1), 15443.

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., & Stoica, I. (2018). Tune: A Research Platform for Distributed Model Selection and Training. *Preprint Arxiv:1807.05118*.

Liu, Q. (2017). Stein variational gradient descent as gradient flow. *Advances in Neural Information Processing Systems*, *30*.

Liu, Q., & Wang, D. (2016). Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in Neural Information Processing Systems*, *29*.

Liu, Q., Lee, J., & Jordan, M. (2016). A kernelized Stein discrepancy for goodness-of-fit tests. *International Conference on Machine Learning*, 276–284.

Loshchilov, I., Glasmachers, T., & Beyer, H.-G. (2018). Large scale black-box optimization by limited-memory matrix adaptation. *IEEE Transactions on Evolutionary Computation*, *23*(2), 353–358.

Lototsky, S. V., Rozovsky, B. L., & others. (2017). *Stochastic partial differential equations* (Vol. 11). Springer.

Lozano, J. A. (2006). *Towards a new evolutionary computation: advances on estimation of distribution algorithms* (Vol. 192). Springer Science & Business Media.

Lu, J., Lu, Y., & Nolen, J. (2019). Scaling limit of the Stein variational gradient descent: The mean field regime. *SIAM Journal on Mathematical Analysis*, *51*(2), 648–671.

MacKay, D. J., & others. (1998). Introduction to Gaussian processes. *NATO ASI Series F Computer and Systems Sciences*, *168*, 133–166.

Maillet, R. (2023). A Note on the Long-Time Behaviour of Stochastic McKean–Vlasov Equations with Common Noise. *Preprint Arxiv:2306.16130*.

Malherbe, C., & Vayatis, N. (2017a). Global optimization of Lipschitz functions. *International Conference on Machine Learning*, 2314–2323.

Malherbe, C., & Vayatis, N. (2017b). Global optimization of Lipschitz functions. *Preprint Arxiv:1703.02628*.

Malherbe, C., Contal, E., & Vayatis, N. (2016). A ranking approach to global optimization. *International Conference on Machine Learning*, 1539–1547.

Malherbe, C., Serré, G., Kalogeratos, A., & Vayatis, N. (2026). *A Ranking Approach to Global Optimization*. https://hal.science/hal-05616863

Marion, E. (2026). A Formalization of the Ionescu-Tulcea Theorem in Mathlib. *Preprint Arxiv:2506.18616*.

Martinez-Cantin, R. (2014). BayesOpt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits. *Journal of Machine Learning Research*, *15*(115), 3915–3919.

The mathlib Community. (2020, January). The lean mathematical library. *ACM SIGPLAN International Conference on Certified Programs and Proofs*.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61.

Močkus, J. (1974). On Bayesian methods for seeking the extremum. *IFIP Technical Conference on Optimization Techniques*, 400–404.

Monmarché, P., & Reygner, J. (2025). Local convergence rates for Wasserstein gradient flows and McKean–Vlasov equations with multiple stationary solutions. *Probability Theory and Related Fields*, 1–59.

Morgan, C., McIver, A., & Seidel, K. (1996). Probabilistic predicate transformers. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, *18*(3), 325–353.

Moss, S., & Perrone, P. (2023). A category-theoretic proof of the ergodic decomposition theorem. *Ergodic Theory and Dynamical Systems*, *43*(12), 4166–4192.

Moura, L. de, & Ullrich, S. (2021). The lean 4 theorem prover and programming language. *International Conference on Automated Deduction*, 625–635.

Mühlenbein, H. (1997). The equation for response to selection and its use for prediction. *Evolutionary Computation*, *5*(3), 303–346.

Mühlenbein, H., & Mahnig, T. (1999). FDA-A scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, *7*(4), 353–376.

Mühlenbein, H., & Paass, G. (1996). From recombination of genes to the estimation of distributions I. Binary parameters. *International Conference on Parallel Problem Solving from Nature*, 178–187.

Munos, R. (2011). Optimistic optimization of a deterministic function without the knowledge of its smoothness. *Advances in Neural Information Processing Systems*, *24*.

Nelder, J. A., & Mead, R. (1965). A simplex method for function minimization. *The Computer Journal*, *7*(4), 308–313.

Nipkow, T., Wenzel, M., & Paulson, L. C. (2002). *Isabelle/HOL: a proof assistant for higher-order logic*. Springer.

Nogueira, F. (2014). *Bayesian Optimization: Open source constrained global optimization tool for Python*. https://github.com/bayesian-optimization/BayesianOptimization

Nüsken, N., & Renger, D. R. M. (2023). Stein variational gradient descent: Many-particle and long-time asymptotics. *Foundations of Data Science*, *5*(3), 286–320.

OpenAI. (2026). *Planar Point Sets with Many Unit Distances*.

Paulin-Mohring, C. (2015). Introduction to the Calculus of Inductive Constructions. In B. W. Paleo & D. Delahaye (Eds.), *All about Proofs, Proofs for All* (Vol. 55). College Publications.

Paulin-Mohring, C. (2015). Introduction to the calculus of inductive constructions. *All About Proofs, Proofs for All*, *55*.

Pelikan, M., & Mühlenbein, H. (1999). The bivariate marginal distribution algorithm. In *Advances in soft computing: Engineering design and manufacturing* (pp. 521–535). Springer.

Pelikan, M., Goldberg, D., & Cantu-Paz, E. (1999). BOA: The Bayesian optimization algorithm. *Genetic and Evolutionary Computation Conference*, 525–532.

Pinnau, R., Totzeck, C., Tse, O., & Martin, S. (2017). A consensus-based model for global optimization and its mean-field limit. *Mathematical Models and Methods in Applied Sciences*, *27*(01), 183–204.

Pintér, J. D. (1991). Global optimization in action. *Scientific American*, *264*, 54–63.

Powell, M. J. (2009). The BOBYQA algorithm for bound constrained optimization without derivatives. *Cambridge NA Report Na2009/06, University of Cambridge, Cambridge*, *26*(26–46), 1.

Powell, W. B. (2019). A unified framework for stochastic optimization. *European Journal of Operational Research*, *275*(3), 795–821.

Price, W. L. (1983). Global optimization by controlled random search. *Journal of Optimization Theory and Applications*, *40*(3), 333–348.

Prigogine, I., & Balescu, R. (1957). Sur la theorie moleculaire du mouvement brownien. *Physica*, *23*(1–5), 555–568.

Rapin, J., & Teytaud, O. (2018). *Nevergrad - A gradient-free optimization platform*. GitHub.

Revuz, D., & Yor, M. (2013). *Continuous martingales and Brownian motion* (Vol. 293). Springer.

Riedl, K. (2024). Leveraging memory effects and gradient information in consensus-based optimisation: On global convergence in mean-field law. *European Journal of Applied Mathematics*, *35*(4), 483–514.

Rinnooy Kan, A. H. G., Timmer, G. T., Rinnooy Kan, A. H. G., & Timmer, G. T. (1985). *The Multi Level Single Linkage Method for Unconstrained and Constrained Global Optimization*.

Rios, L. M., & Sahinidis, N. V. (2013). Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, *56*(3), 1247–1293.

Riou, J. (2025). Formalization of derived categories in Lean/mathlib. *Annals of Formalized Mathematics*, *1*(13609).

Roberts, G. O., & Tweedie, R. L. (1996). Exponential Convergence of Langevin Distributions and Their Discrete Approximations. *Bernoulli*, *2*(4), 341–363.

Ros, R., & Hansen, N. (2008). A simple modification in CMA-ES achieving linear time and space complexity. *International Conference on Parallel Problem Solving from Nature*, 296–305.

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, *1*(2), 127–190.

Rudi, A., Marteau-Ferey, U., & Bach, F. (2025). Finding global minima via kernel approximations: A. Rudi et al. *Mathematical Programming*, *209*(1), 703–784.

Schauder, J. (1930). Der fixpunktsatz in funktionalraümen. *Studia Mathematica*, *2*(1), 171–180.

Serré, G., Beja-Battais, P., Chirrane, S., Kalogeratos, A., & Vayatis, N. (2024). Lipo+: Frugal global optimization for lipschitz functions. *Hellenic Conference on Artificial Intelligence*, 1–6.

Serré, G., Germain, P., Gruffaz, S., & Kalogeratos, A. (2026c). *Enhancing Exploration in Global Optimization by Noise Injection in the Probability Measures Space*.

Serré, G., Kalogeratos, A., & Vayatis, N. (2025). Stein Boltzmann Sampling: A Variational Approach for Global Optimization. In Y. Li, S. Mandt, S. Agrawal, & E. Khan (Eds.), *International Conference on Artificial Intelligence and Statistics* (Vol. 258, pp. 757–765). PMLR.

Serré, G., Kalogeratos, A., & Vayatis, N. (2026b). *A Unifying Framework for Global Optimization: From Theory to Formalization*.

Serré, G., Kalogeratos, A., & Vayatis, N. (2026a). *GLOBe: A Modular Global Optimization library*. https://hal.science/hal-05623801

Smith, R. L. (1984). Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, *32*(6), 1296–1308.

Sothanaphan, N. (2026). Resolution of Erdős Problem #728: a writeup of Aristotle's Lean proof. *Arxiv Preprint Arxiv:2601.07421*.

Spendley, W., Hext, G. R., & Himsworth, F. R. (1962). Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics*, *4*(4), 441–461.

Sriperumbudur, B. K., Fukumizu, K., & Lanckriet, G. R. (2011). Universality, Characteristic Kernels and RKHS Embedding of Measures. *Journal of Machine Learning Research*, *12*(7).

Stein, C. (1972). A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. *Berkeley Symposium on Mathematical Statistics and Probability*, *6*, 583–603.

Steinwart, I. (2001). On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, *2*(Nov), 67–93.

Steinwart, I., & Christmann, A. (2008). *Support vector machines*. Springer Science & Business Media.

Sun, L., Karagulyan, A., & Richtarik, P. (2023). Convergence of stein variational gradient descent under a weaker smoothness condition. *International Conference on Artificial Intelligence and Statistics*, 3693–3717.

Surjanovic, S., & Bingham, D. (2024). *Virtual Library of Simulation Experiments: Test Functions and Datasets*.

Sznitman, A.-S. (2006). Topics in propagation of chaos. In *Ecole d'été de probabilités de Saint-Flour XIX—1989* (pp. 165–251). Springer.

Tao, T. (2022). Analysis I. *Texts and Readings in Mathematics*.

Trouvé, A. (1996). Rough large deviation estimates for the optimal convergence speed exponent of generalized simulated annealing algorithms. *Annales De L'ihp Probabilités Et Statistiques*, *32*(3), 299–348.

Tugaut, J. (2014). Phase Transitions of McKean–Vlasov Processes in Double-Wells Landscape. *Stochastics*, *86*(2), 257–284.

Tulcea, C. I. (1949). Mesures dans les espaces produits. *Atti Accad. Naz. Lincei Rend*, *7*, 208–211.

Valko, M., Carpentier, A., & Munos, R. (2013). Stochastic simultaneous optimistic optimization. *International Conference on Machine Learning*, 19–27.

Vie, A., Kleinnijenhuis, A. M., & Farmer, D. J. (2020). Qualities, challenges and future of genetic algorithms: a literature review. *Arxiv Preprint Arxiv:2011.05277*.

Villani, C. (2003). *Topics in Optimal Transportation*. American Mathematical Society.

Villani, C. (2009). *Optimal Transport*. Springer Berlin Heidelberg.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., & others. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272.

Wang, W.-chuan, Tian, W.-can, Xu, D.-mei, & Zang, H.-fei. (2024). Arctic puffin optimization: A bio-inspired metaheuristic algorithm for solving engineering design optimization. *Advances in Engineering Software*, *195*, 103694.

Wang, Z., Hutter, F., Zoghi, M., Matheson, D., & De Feitas, N. (2016). Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, *55*, 361–387.

Welling, M., & Teh, Y. W. (2011). Bayesian learning via stochastic gradient Langevin dynamics. *International Conference on Machine Learning*, 681–688.

Wieser, E., & Song, U. (2022). Formalizing Geometric Algebra in Lean: E. Wieser, U. Song. *Advances in Applied Clifford Algebras*, *32*(3), 28.

Winfield, D. (1970). Functions and functional optimization by interpolation in data tables. *Phd. Cambridge: Harvard University*.

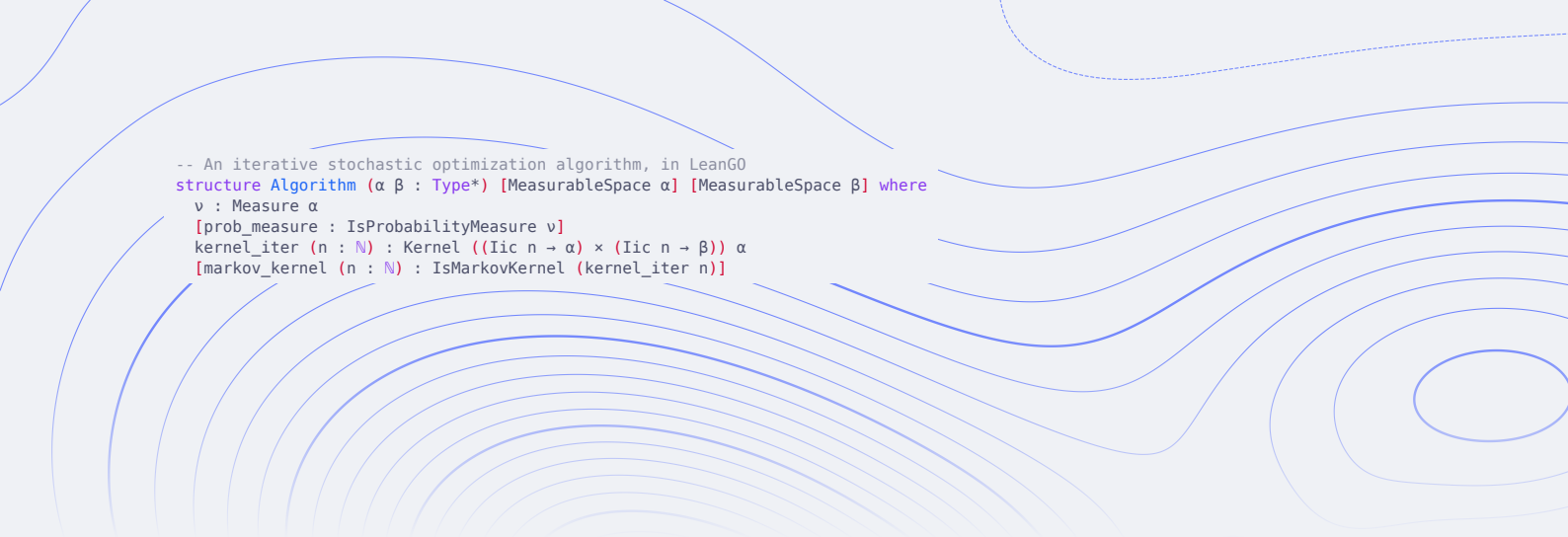
Wolpert, D. H., & Macready, W. G. (2002). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82.

Zabinsky, Z. B. (2003). *Stochastic adaptive search for global optimization* (Vol. 72). Springer Science & Business Media.

Zabinsky, Z. B., & Smith, R. L. (1992). Pure adaptive search in global optimization. *Mathematical Programming*, *53*(1), 323–338.

Zhan, Z.-H., Shi, L., Tan, K. C., & Zhang, J. (2022). A survey on evolutionary computation for complex continuous optimization. *Artificial Intelligence Review*, *55*(1), 59–110.

Zhou, D.-X. (2008). Derivative reproducing properties for kernel methods in learning theory. *Journal of Computational and Applied Mathematics*, *220*(1–2), 456–463.



Methodological advances and formalization in global optimization

Keywords: Global optimization, McKean-Vlasov, Unification, Formal methods

Global optimization algorithms are numerous and diverse, yet beneath their surface lie shared mathematical structures that can be identified and exploited. This thesis advances the field by recognizing these underlying structures, organizing algorithms into families, developing methodological advances rooted in this perspective, and establishing formal foundations through rigorous mathematical frameworks.

Among global optimization algorithms, two families stand out and deserve distinguishing: decision-based approaches and McKean-Vlasov-based approaches. These families share deep mathematical structures that enable reasoning and result transfer across algorithms. Within this family framework, this thesis develops two major methodological contributions: **i)**, the introduction of *Stein Boltzmann Sampling*, a novel McKean-Vlasov-based algorithm; **ii)** a modular approach to adding common noise in McKean-Vlasov-based algorithm, improving exploration while preserving mean-field stochasticity. The common noise approach exemplifies the power of family reasoning: it is a single module applicable uniformly across an entire algorithmic family. This thesis also validates this family-centric perspective through a Python/C++ library that exploits shared structure to maximize code reuse and deploy plugins applicable to all algorithms within a family.

This family-centric reasoning naturally motivates investigating what unifies optimization algorithms at a larger scale. The second part develops an abstract mathematical framework unifying all optimization algorithms, including those from the two distinguished families, within a common language. This unified framework enables reasoning about all algorithms simultaneously. One crucial application of this framework is establishing rigorous foundations for global optimization research through formal methods. Specifically, this thesis develops contributions in LEAN that formalize key results on Markov kernels, and implements both the unifying framework and concrete optimization algorithms, building a practical formal library. This enables both rigorous reasoning about optimization algorithm theory and reusable formal developments, opening possibilities for mechanized verification and integration with emerging tools like large language models.

Avancées méthodologiques et formalisation en optimisation globale

Mots-clés : Optimisation globale, McKean-Vlasov, Unification, Méthodes formelles

Les algorithmes d'optimisation globale sont nombreux et variés, mais sous leur surface résident des structures mathématiques partagées qui peuvent être identifiées et exploitées. Cette thèse fait progresser le domaine en reconnaissant ces structures sous-jacentes, en organisant les algorithmes en familles, en développant des avancées méthodologiques enracinées dans cette perspective, et en établissant des fondations formelles par le biais de cadres mathématiques rigoureux.

Parmi les algorithmes d'optimisation globale, deux familles se distinguent particulièrement: les approches à base de décision et les approches basées sur les dynamiques de McKean-Vlasov. Ces familles partagent des structures mathématiques profondes qui permettent le raisonnement et le transfert de résultats entre algorithmes. Dans ce cadre familial, cette thèse développe deux contributions méthodologiques majeures: **i)** l'introduction de *Stein Boltzmann Sampling*, un nouvel algorithme basé sur des dynamiques de McKean-Vlasov; **ii)** une approche modulaire pour ajouter du bruit commun dans les dynamiques de McKean-Vlasov, améliorant l'exploration tout en préservant la stochasticité en limite champ-moyen. L'approche du bruit commun exemplifie la puissance du raisonnement par famille: c'est un module unique applicable uniformément à toute une famille d'algorithmes. Cette thèse valide également cette perspective centrée sur les familles par une bibliothèque Python/C++ qui exploite la structure partagée pour maximiser la réutilisation du code et déployer des modules applicables à tous les algorithmes au sein d'une famille.

Ce raisonnement centré sur les familles motive naturellement l'étude de ce qui unifie les algorithmes d'optimisation à une échelle plus large. La seconde partie développe un cadre mathématique abstrait unifiant tous les algorithmes d'optimisation, incluant ceux des deux familles distinguées, dans un langage commun. Ce cadre unifié permet le raisonnement sur tous les algorithmes simultanément. Une application cruciale de ce cadre est l'établissement de fondations rigoureuses pour la recherche en optimisation globale par les méthodes formelles. Spécifiquement, cette thèse développe des contributions en LEAN qui formalisent des résultats clés sur les noyaux de Markov, et implémente à la fois le cadre unificateur et des algorithmes d'optimisation concrets, construisant une bibliothèque formelle fonctionnelle. Cela permet à la fois le raisonnement rigoureux sur la théorie des algorithmes d'optimisation et les développements formels réutilisables, ouvrant des possibilités pour la vérification mécanisée et l'intégration avec des outils émergents comme les modèles de langage.