LP-Based Weighted Model Integration over Non-Linear Real Arithmetic

S. Akshay[1], **Supratik Chakraborty**[1], **Soroush Farokhnia**[2], **Amir Goharshady**[3], **Harshit Jitendra Motwani**[4], **Đorđe Žikelić**[5]
[1]Indian Institute of Technology Bombay,
[2]Hong Kong University of Science and Technology,
[3]University of Oxford,
[4]Max Planck Institute for Software Systems,
[5]Singapore Management University
{akshayss,supratik}@cse.iitb.ac.in, amir.goharshady@cs.ox.ac.uk, hmotwani@mpi-sws.org, sfarokhnia@connect.ust.hk, dzikelic@smu.edu.sg

Abstract

Weighted model integration (WMI) is a relatively recent formalism that has received significant interest as a technique for solving probabilistic inference tasks with complicated weight functions. Existing methods and tools are mostly focused on linear and polynomial functions and provide limited support for WMI of rational or radical functions, which naturally arise in several applications. In this work, we present a novel method for approximate WMI, which provides more effective support for the wide class of semi-algebraic functions that includes rational and radical functions, with literals defined over non-linear real arithmetic. Our algorithm leverages Farkas' lemma and Handelman's theorem from real algebraic geometry to reduce WMI to solving a number of linear programming (LP) instances. The algorithm provides formal guarantees on the error bound of the obtained approximation and can reduce it to any user-defined value ϵ. Furthermore, our approach is perfectly parallelizable. Finally, we present extensive experimental results, demonstrating the superior performance of our method on a range of WMI tasks for rational and radical functions when compared to state-of-the-art tools for WMI, in terms of both applicability and tightness.

1 Introduction

Probabilistic inference tasks are of central importance in many applications. In the discrete setting, weighted model counting (WMC) [Chavira and Darwiche, 2008] is a popular and effective paradigm for solving such problems in a wide variety of models. In WMC, the probabilistic inference task is reduced to computing the weighted sum of satisfying assignments of Boolean formula. Recent tools have taken advantage of the vast advances made in SAT-solvers to solve WMC effectively [Chakraborty *et al.*, 2016]. However, many probabilistic inference tasks are defined over richer domains, e.g, continuous settings, where the summation needs to be replaced by integration, and satisfiability needs to be checked for formulas involving, e.g, linear and non-linear constraints over reals. This has led to the definition of the problem of weighted model integration (WMI) [Belle *et al.*, 2015], which applies in the continuous and even hybrid settings. In WMI, the weighted sum over models is replaced by an integral (or a Boolean combination of integrals) of weight functions defined over the satisfying assignments for SMT (satisfiability modulo theory)-formulas over infinite domains. This often necessitates the use of SMT-solvers over various theories, which are richer but less scalable than SAT-solvers.

In the last 10 years, a rich line of research has emerged around WMI (e.g, [Morettin *et al.*, 2017; Dos Martires *et al.*, 2019; Morettin *et al.*, 2021; Abboud *et al.*, 2022; Spallitta *et al.*, 2024]) with multiple techniques developed for increasingly more powerful weight functions ranging from linear functions to even polynomial functions, and integrating over increasingly more complex sets, from intervals to polytopes and even conjunctions of polynomials. WMI methods have also been investigated for applications involving probabilistic inference in probabilistic programs [Sankaranarayanan *et al.*, 2013; Albarghouthi *et al.*, 2017; Gehr *et al.*, 2016; Beutner *et al.*, 2022], where integration over domains like polytopes and semi-algebraic sets is often needed. The existing approaches can be broadly divided into two groups: exact and approximate. Exact approaches are either SMT-solver based [Morettin *et al.*, 2017] or knowledge-compilation based [Dos Martires *et al.*, 2019]. In both cases, given #*P*-hardness of even the exact WMC problem, these approaches are often restrict to theories such as LRA (linear real arithmetic). Approximate approaches are often further divided into statistical and numerically approximate, where the former refers to Monte-Carlo methods which are highly scalable and expressive, but provide statistical guarantees or even no guarnatees at all [Spallitta *et al.*, 2024; Dos Martires *et al.*, 2019]. The latter are approximate but with formal guarantees on the error bound, and hence provide a tradeoff between the exact and statistical approaches. We adopt this style of guarantees in this paper.

Our contributions. In this work, our goal is to develop a numerically approximate (henceforth referred to as just approximate) approach for WMI which allows rich weight functions beyond linear and polynomial functions, as well as integration over domains that are more complex than previously considered. Our main contribution is a new approximate method for the WMI problem that can handle integration over general semi-algebraic sets, i.e., Boolean combinations of polynomials, and general semi-algebraic weight functions.

Two important classes of semi-algebraic weight functions that are of particular interest in this work are *rational functions* and *radical functions*, whose integration commonly arises in physics, motion modelling and finance. For instance, the computation of inverse Fourier transforms often involves integration over rational functions, whereas in electrostatics and quantum physics Legendre polynomials are utilized to approximate spherical harmonics [Riley *et al.*, 2006]. In motion modelling, Lévy-flights provide a good probabilistic model for how humans and animals (e.g. sharks) perform blind rapid search for some target region, e.g. food or shelter [Sims *et al.*, 2008; Humphries *et al.*, 2010]. In mathematical finance, pricing of certain assets are often modelled using Lévy-flights with heavy-tailed Cauchy distribution guiding the jums in prices (see also Example 3) [Tankov, 2003; Bouchaud and Potters, 2003]. Many computational inference problems in these examples reduce to integrating rational or radical functions.

Our new method reduces the approximate WMI problem to the problem of computing an approximation of the volume of a semi-algebraic set. Several works in mathematics literature have considered the problem of computing approximate volumes of basic semi-algebraic sets, e.g., [Lairez *et al.*, 2019] uses Picard-Fuchs equation and then numerically solve it for computing the volume of basic semi-algebraic sets, while [Henrion *et al.*, 2009] uses semi-definite programming relaxations, but these are harder to implement. Our approach instead leverages results from optimization such as Farkas' lemma and real-algebraic geometry such as Handelman's theorem to reduce the approximate WMI problem to solving a number of linear programming (LP) instances. This gives us a computationally tractable approach. Moreover, for any desired error bound $\epsilon > 0$, we prove that our method is guaranteed to return an ϵ-tight approximation of the WMI value, by tuning method parameters at the cost of increasing the number of needed LP-calls. Thus, our contributions are as follows:

1. We develop a novel algorithm for the approximate WMI problem. Our algorithm supports integration of general semi-algebraic weight functions over general semi-algebraic sets. This makes our method effectively applicable to a large new class of weight functions, such as rational functions and radical functions.
2. Our method provides formal guarantees on the tightness of the approximation error. In particular, for any error bound $\epsilon > 0$ provided by the user, our method produces an ϵ-tight approximation of the WMI value.
3. We implement our algorithm and integrated our tool (WMI-LP) with the state-of-the-art WMI framework of [Spallitta *et al.*, 2024]. Our extensive experiments demonstrate that our approach outperforms state-of-the-art exact and approximate tools in handling a wider variety of polynomial, rational, and radical functions. It also provides consistently tighter error bounds, highlighting its robustness and accuracy across benchmarks.

Related Works. We already mentioned several related works on WMI. [Morettin *et al.*, 2017] is an exact method for WMI which uses predicate abstraction, but is restricted to LRA and Boolean formulas. In particular, it uses MathSAT [Cimatti *et al.*, 2013] for SMT reasoning and LattE Integrale [De Loera *et al.*, 2004] for finding integrals and can handle polynomial weight functions over LRA formulas. On the other hand, [Dos Martires *et al.*, 2019] uses knowledge compilation for WMI for NRA and Boolean formulas and can handle probability functions as weight functions over NRA formulas. It provides support for both exact and approximate WMI. The exact WMI method Symbo depends on the PSI solver [Gehr *et al.*, 2016], which has proof rules for computing exact integrals. Their approximate method Sampo depends upon Monte Carlo sampling, therefore does not provide guarantees. WMI has also been explored for applications involving probabilistic inference in probabilistic programs [Sankaranarayanan *et al.*, 2013; Albarghouthi *et al.*, 2017; Gehr *et al.*, 2016; Beutner *et al.*, 2022]. Finally, [Spallitta *et al.*, 2024] is the most recent and relevant work, that we build upon, which provides an option of using exact or sampling-based integrators for WMI. In particular, we have integrated our WMI-LP tool with their tool and provide an option to use our method to compute the integration in their algorithm. Its exact method cannot handle weight functions which are rational functions or radicals, whereas the sampling-based method can handle rational functions but does not provide any guarantees on approximation tightness. Moreover, the approximate method cannot handle radical functions. Beyond the earlier mentioned works, volume approximation has been considered in applications in computer graphics (e.g., [Rom and Brakhage, 2011]). The idea of gridding, which we also use is common and has been found in several previous works (e.g., [Dehnert *et al.*, 2015; Akshay *et al.*, 2024b; Akshay *et al.*, 2024a]). The use of Handleman's theorem and Farkas' lemma has been done for quantifier elimination, invariant generation etc, but not for volume computation and not over general semi-algebraic sets [Chatterjee *et al.*, 2025; Colón *et al.*, 2003].

2 Preliminaries

In this section, we introduce the necessary preliminaries on semi-algebraic sets and functions, and formally define the problem that we consider in this work.

Semi-algebraic Sets. A set $S \subseteq \mathbb{R}^n$ is said to be *semi-algebraic*, if it is the satisfiability set of a logical predicate defined in terms of a boolean combination of finitely many polynomial inequalities over $\mathbb{R}^n$. Formally, S is semi-algebraic if it can be expressed as

$$S = \left\{ \mathbf{x} \in \mathbb{R}^n \mid \varphi(\mathbf{x}) \right\},$$

where ϕ is some logical predicate over n real-valued variables that can be generated by the grammar

$$\begin{array}{lll} \varphi & := & \ell \mid \varphi \wedge \varphi \mid \neg\varphi \\ \ell & := & f \bowtie 0, \text{ where } \bowtie \ \in \{>, \geq\}, \ f \in \mathbb{R}[\mathbb{V}], \end{array} \quad (1)$$

with $\mathbb{V} = \{x_1, x_2, \ldots, x_n\}$ a set of n real-valued variables and $\mathbb{R}[\mathbb{V}]$ denoting the set of all polynomial functions over $\mathbb{V}$.

Semi-algebraic Functions. A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be *semi-algebraic*, if its graph $\Gamma(f) := \{(\mathbf{x}, y) \in \mathbb{R}^{n+1} \mid \mathbf{x} \in \mathbb{R}^n, y = f(\mathbf{x})\}$ is a semi-algebraic set in $\mathbb{R}^{n+1}$.

Example 1 (Rational and radical functions)**.** *Two classes of semi-algebraic functions that will be of particular interest in this work are rational and radical functions. Rational functions are defined as fractions of polynomials, i.e.* $f(x) = \frac{p(x)}{q(x)}$, *where* $p(x)$ *and* $q(x) \neq 0$ *are polynomials in* $\mathbb{R}[\mathbb{V}]$.

Radical functions extend this definition and are expressed in the form $r(x) = \sqrt[n]{f(x)}$, *where* $f(x)$ *is a rational function as defined above, and* $n \in \mathbb{N}$ *is a positive integer.*

Graphs of both rational and radical functions are semi-algebraic, and hence, these functions are semi-algebraic.

Weighted Model Integration. Weighted Model Integration (WMI) generalizes Weighted Model Counting (WMC) to hybrid domains involving both Boolean and real variables by integrating a weight function $w(\mathbf{x})$ over a semi-algebraic set S defined by logical formulas. In WMC, the goal is to compute the weighted sum of satisfying assignments for a propositional formula, where each Boolean literal is assigned a non-negative weight. WMI extends this by integrating over real-valued variables in addition to summing over Boolean assignments. Classically, WMI assumes a factorization of the weight function w, enabling the integral to be computed as a sum over disjoint Boolean assignments. Formally, the WMI of a formula ϕ can be expressed as

$$\mathrm{WMI}(\phi, w) = \sum_{\mathbf{b} \in \mathcal{B}(\phi)} \prod_{\ell \in \mathbf{b}} w(\ell) \int_{S(\mathbf{b})} w(\mathbf{x})\, d\mathbf{x},$$

where $\mathcal{B}(\phi)$ denotes the set of truth assignments to the Boolean atoms of ϕ, $S(\mathbf{b})$ is the semi-algebraic region defined by the real-valued constraints under $\mathbf{b}$, and ℓ iterates over the literals in $\mathbf{b}$. Since existing WMI and WMC tools efficiently handle the Boolean part of ϕ, we focus exclusively on the integral component of the problem. In this work, we extend WMI to support a broader class of weight functions than prior methods, enabling more expressive and flexible modeling for hybrid domains.

More precisely, our goal in weighted model integration (WMI) is to compute or approximate the value of the integral of a weight function over some given set. Given a weight function $w : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ and a set $S \subseteq \mathbb{R}^n$, we write

$$\mathrm{WMI}(S, w) = \int_S w(\mathbf{x})\, d\mathbf{x}$$

to denote the Lebesgue integral of the function w over the set S. For the integral to be mathematically well defined, we assume that both w and S are Borel-measurable and that w is a non-negative function.

Problem Statement. We now formally define our problem, which is concerned with approximate WMI of semi-algebraic weight functions over semi-algebraic sets.

Let $\mathbb{V} = \{x_1, x_2, \ldots, x_n\}$ be a set of n real-valued variables. Suppose that we are given a semi-algebraic set $S = \{\mathbf{x} \in \mathbb{R}^n \mid \varphi(\mathbf{x})\}$ where the predicate ϕ is defined according to the grammar in eq. (1), and a non-negative semi-algebraic weight function $w : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$. In addition, suppose that we are given an error bound $\epsilon > 0$.

The goal of our WMI problem is to compute an ϵ-tight approximation of the value of the integral $\mathrm{WMI}(S, w)$, i.e. we want to compute a value $M \in \mathbb{R}$ such that

$$\mathrm{WMI}(S, w) - \epsilon \leq M \leq \mathrm{WMI}(S, w) + \epsilon.$$

Assumption: Boundedness. In addition to assuming that w and S are semi-algebraic, our WMI algorithm will also assume that the set $S \subseteq \mathbb{R}^n$ is *bounded*. In particular, for each variable $x_i \in \mathbb{V}$, we assume that we are given an interval bound $[a_i, b_i]$ on the values that x_i can attain over S. Hence, we have $S \subseteq \prod_{i=1}^n [a_i, b_i]$.

Assumption: Formula of Weight Function. We assume the predicates for the weight function's graph $\Gamma(w)$ and for the semi-algebraic set $\{(\mathbf{x}, y) \in \mathbb{R}^{n+1} \mid w(\mathbf{x}) > y\}$ are provided in the form that conforms to the grammar in eq. (1) as the description of the weight function in the input to our algorithm. This is needed in order to allow for automated reasoning about the weight function w.

Example 2. *To illustrate our problem, we present examples of two special cases of weight functions that give rise to classical and well known computational problems. If* $w(\mathbf{x}) = 1$ *for all* $\mathbf{x} \in \mathbb{R}^n$, *then our WMI problem becomes the problem of approximating the volume of a bounded set* S.

If $w(\mathbf{x})$ *is a probability density function of some probability distribution, then our WMI problem becomes the probabilistic inference problem of approximating the probability* $\mathbb{P}[\mathbf{X} \in S]$, *where* $\mathbf{X}$ *is a random variable over* $\mathbb{R}^n$ *whose probability density function is* $w(\mathbf{x})$.

In both cases, we are interested in computing ϵ*-tight approximations, where* $\epsilon > 0$ *is a precision parameter that can be provided by the user.*

Example 3 (Lévy Flight in Financial Modeling)**.** *In finance, asset prices often experience rare but significant jumps, modeled as Lévy flights. These dynamics can be represented as:*

$$P_{t+1} = P_t + \Delta P_t, \quad \Delta P_t \sim \mathit{Cauchy}(\mu, \gamma),$$

where the heavy-tailed Cauchy distribution captures extreme price movements. An event of interest, such as the price exceeding a threshold K, is expressed as a predicate $\psi(P_T)$.

Computing the probability $\mathbb{P}[\psi(P_T)]$ *reduces to a weighted model integration problem over a rational weight function on a semi-algebraic domain, since the probability density function of the Cauchy*(μ, γ) *probability distribution is a rational function. See the extended version [Akshay* et al.*, 2025a] for more details.*

3 Algorithm

We now present our algorithm for solving the WMI problem for semi-algebraic functions and sets, that was defined in the previous section. Our algorithm proceeds in three steps:

1. ***Step 1. Reduction to Volume Computation.*** We reduce the problem of computing an ϵ-tight approximation of WMI(S, w) to the problem of computing an ϵ-tight approximation of WMI$(S', 1)$, where $S' \subseteq \mathbb{R}^{n+1}$ is a suitably defined semi-algebraic set that depends on S and w and which satisfies WMI$(S', 1) =$ WMI(S, w).
2. ***Step 2. Computation of Bounds on S'.*** The new semi-algebraic set S' is defined over $n + 1$ variables $x_1, \ldots, x_n$ and a fresh variable y introduced in the construction of S' (see details below). In this step, we compute an interval bound on the values that y can attain over S'.
3. ***Step 3. Volume Approximation:.*** Finally, we compute an ϵ-tight approximation of volume WMI$(S', 1) =$ WMI(S, w), by using Algorithm 1.

Step 1. Reduction to Volume Computation. We define the semi-algebraic set S' for which WMI$(S', 1) =$ WMI(S, w) as follows. Let y be a fresh variable distinct from all variables in $\mathbb{V} = \{x_1, \ldots, x_n\}$. We define a predicate ψ over the variables $\{x_1, \ldots, x_n, y\}$ via

$$\psi = y \geq 0 \wedge w(x_1, x_2, \cdots, x_n) \geq y \wedge \varphi, \tag{2}$$

where recall φ is a predicate over $\{x_1, \ldots, x_n\}$ for which $S = \{\mathbf{x} \in \mathbb{R}^n \mid \varphi(\mathbf{x})\}$. Notice that ψ conforms to the grammar in eq. (1), since ϕ conforms to the grammar in eq. (1) and we also assumed that the predicate $w(\mathbf{x}) \leq y = \neg(w(\mathbf{x}) > y)$ is provided as an algorithm input in the form conforming to the grammar in eq. (1). Hence, the set $S' = \{(\mathbf{x}, y) \in \mathbb{R}^{n+1} \mid \psi(\mathbf{x}, y)\}$ is semi-algebraic. By using the properties of Lebesgue integration, it follows that

$$\text{WMI}(S', 1) = \text{WMI}(S, w),$$

as desired. For a formal proof of this claim, we refer the reader to the extended version [Akshay *et al.*, 2025a].

Step 2. Computation of Bounds on S'. Next, we derive an interval bound $[a, b]$ on the values that y can attain over S'. Thus, we have $S' \subseteq \prod_{i=1}^{n}[a_i, b_i] \times [a, b] \subseteq \mathbb{R}^{n+1}$. This will be needed later in Algorithm 1 which will compute the ϵ-tight approximation of WMI$(S', 1)$.

Since the predicate ψ in eq. (2) contains the clause $y \geq 0$, we can set a lower bound to $a = 0$. On the other hand, we observe from eq. (2) that b is a correct upper bound if and only if $w(x_1, \ldots, x_n) \leq b$ for all $(x_1, \ldots, x_n) \in S$. Since we know that $S \subseteq \prod_{i=1}^{n}[a_i, b_i]$ by our problem assumptions, it follows that any value of b for which the constraint

$$\bigwedge_i (x_i \geq a_i \wedge x_i \leq b_i) \implies w(x_1, \cdots, x_n) \leq b$$

is satisfied yields a correct upper bound on the value that y can attain over S'. Such value of b can be computed by performing binary search. For each fixed value of b, the check of whether the above system of constraints is satisfied is reduced to solving a number of linear programming (LP) instances, by using Farkas' Lemma and Handelman's theorem. We present the details of this procedure in the extended version [Akshay *et al.*, 2025a] and also provide the statements of Farkas' Lemma [Farkas, 1902] and Handelman's Theorem [Handelman, 1988] in the extended version [Akshay *et al.*, 2025a]. In the special case of w being a rational or a radical function, this procedure can be further optimized by directly reducing the problem to solving LP instances and without the need to perform binary search, see the extended version [Akshay *et al.*, 2025a] for details.

Step 3. Volume Approximation. Finally, we present our algorithm for computing an ϵ-tight approximation of the volume WMI$(S', 1)$ of the semi-algebraic set S' defined in Step 2.

The main idea behind our algorithm is to cover the semi-algebraic set S' with a finite number of hyper-rectangles. The volume of S' is then approximated by summing the volumes of all the hyper-rectangles contained in S'.

Algorithm Overview. Consider the semi-algebraic set S' defined by the predicate ψ in Step 2. We begin with the hyper-rectangle $[a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_n, b_n] \times [a, b]$, which encloses the entire semi-algebraic set. We then recursively divide this hyper-rectangle into smaller hyper-rectangles. After each subdivision, for each hyper-rectangle we determine whether it is contained entirely inside, outside, or intersects the semi-algebraic set S'. Hyper-rectangles completely outside of S' are discarded, while those completely inside are added to the total volume. Hyper-rectangles that intersect S' are further subdivided. This process is continued until the volume of the remaining hyper-rectangles (i.e. those that have been neither discarded nor added to the volume) is less than ϵ. That way, we ensure that the total added volume upon the algorithm termination yields an ϵ-tight approximation of the total volume of S'. To determine whether a hyper-rectangle is inside, outside, or intersects S', we again use Handelman's Theorem and Farkas' Lemma to translate this problem into solving a number of LP instances, which can be solved by using an off-the-shelf LP solver. Finally, the algorithm outputs an ϵ-tight approximation of the volume of the set S'. The pseudocode of our algorithm is shown in Algorithm 1.

Hyper-rectangles. A *hyper-rectangle H* in $\mathbb{R}^{n+1}$ is defined as the solution set of a system of inequalities of the form

$$\psi_H = \begin{cases} l_1 \leq x_1 \leq u_1 \\ l_2 \leq x_2 \leq u_2 \\ \vdots \\ l_{n+1} \leq x_{n+1} \leq u_{n+1} \end{cases} \tag{3}$$

where each $l_1, u_1, \ldots, l_{n+1}, u_{n+1} \in \mathbb{R}$ with $l_i \leq u_i$ for each i. Given a hyper-rectangle H, we denote its *volume* as $\text{vol}(H) = \prod_{i=1}^{n+1}(u_i - l_i)$ and its *diameter* as $\text{diameter}(H) = \max_{i=1}^{n+1}\{u_i - l_i\}$.

Subdivision of Hyper-rectangles. The *subdivision of hyper-rectangle H along the i-th dimension* is the pair of hyper-rectangles $\{H_1, H_2\}$, where H_1 is defined by the system of inequalities $\psi_{H_1} = \psi_H \wedge (x_i \leq \frac{l_i + u_i}{2})$ and H_2 is defined by the system of inequalities $\psi_{H_2} = \psi_H \wedge (x_i \geq \frac{l_i + u_i}{2})$.

Subset Decision Procedure. The SubsetDecision procedure determines whether a hyper-rectangle H is contained entirely inside, outside, or intersects the semi-algebraic set S'. We use Farkas' Lemma and Handelman's Theorem to translate this problem into solving a number of LP instances. In what follows, we present the details of this translation.

Consider the semi-algebraic set $S' \subseteq \mathbb{R}^{n+1}$ defined by the predicate ψ in eq. (2), and let $H \subseteq \mathbb{R}^{n+1}$ be a hyper-rectangle

Algorithm 1 Volume Approximation of Semi-algebraic Sets

procedure VolumeApproximation($\mathbb{V}, S', \psi, \epsilon, [a_i, b_i], [a, b], d$)
  $H_0 \leftarrow [a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_n, b_n] \times [a, b]$
  $VolumeSum \leftarrow 0$
  $HyperRCover \leftarrow \{H_0\}$
  Error $\leftarrow$ Volume($HyperRCover$)
  while $HyperRCover \neq \emptyset \wedge$ Error $\geq \epsilon$ **do**
    Choose $H \in HyperRCover$
    $HyperRCover \leftarrow HyperRCover \setminus \{H\}$
    $result \leftarrow$ SubsetDecision(H, S)
    if $result$ = Outside **then**
      Discard H
      Error $\leftarrow$ Error $-$ Volume(H)
    else if $result$ = Inside **then**
      $VolumeSum \leftarrow VolumeSum + Volume(H)$
      Error $\leftarrow$ Error $-$ Volume(H)
    else
      $H_1, H_2 \leftarrow$ Subdivide(H)
      $HyperRCover \leftarrow HyperRCover \cup \{H_1, H_2\}$
    end if
  end while
   return $VolumeSum$
end procedure

defined by the system of inequalities ψ_H as in eq. (3). Observe that H is contained entirely inside S' if and only if

$$\forall (\mathbf{x}, y) \in \mathbb{R}^{n+1}. (\mathbf{x}, y) \in \psi_H \Longrightarrow \psi(\mathbf{x}, y), \quad (4)$$

that H is contained entirely outside S' if and only if

$$\forall (\mathbf{x}, y) \in \mathbb{R}^{n+1}. (\mathbf{x}, y) \in \psi_H \Longrightarrow \neg\psi(\mathbf{x}, y), \quad (5)$$

and that H intersects S' if and only if neither of the above is satisfied. Hence, in order to determine whether H is contained entirely inside, outside, or intersects S', it suffices to determine if either of the above two formulas is valid. In what follows, we show how SubsetDecision determines the validity of eq. (4). The validity of eq. (5) is determined analogously.

To determine the validity of eq. (4), recall that S' is semi-algebraic hence ψ can be expressed as a boolean combination of polynomial inequalities over variables $x_1, \ldots, x_n, y$, i.e.

$$\psi = \bigwedge_{i=1}^{p} \bigvee_{j=1}^{q} p_{i,j}(x_1, \ldots, x_n, y) \geq 0,$$

where each $p_{i,j}$ is a polynomial over $x_1, \ldots, x_n, y$. Hence, eq. (4) is valid if and only if

$$\forall (\mathbf{x}, y) \in \mathbb{R}^{n+1}. (\mathbf{x}, y) \in \psi_H \Longrightarrow \bigvee_{j=1}^{q} p_{i,j}(x_1, \ldots, x_n, y) \geq 0$$

holds for each $1 \leq i \leq p$. On the other hand, to show validity of the above, it suffices to show that

$$\forall (\mathbf{x}, y) \in \mathbb{R}^{n+1}. (\mathbf{x}, y) \in \psi_H \Longrightarrow p_{i,j}(x_1, \ldots, x_n, y) \geq 0 \quad (6)$$

holds for at least one $1 \leq j \leq q$. In the extended version [Akshay *et al.*, 2025a], we show how SubsetDecision uses Farkas' Lemma (when the polynomial degree of $p_{i,j}$ is 1) or Handelman's Theorem (when the polynomial degree of $p_{i,j}$ is at least 2) to reduce the problem of determining validity of eq. (6) to solving an LP instance.

SubsetDecision checks if for each $1 \leq i \leq p$ there exists $1 \leq j \leq q$ such that eq. (6) is valid. If the answer is positive, SubsetDecision concludes that H is contained entirely inside S' and returns "Inside". An analogous check of validity of eq. (5) is performed and if the answer is positive then SubsetDecision concludes that H is contained entirely outside S' and returns "Outside". Finally, if neither eq. (4) nor eq. (5) are shown to be valid, SubsetDecision returns "Unknown" which indicates that further subdivision is needed.

Algorithm Termination and Correctness. The following theorem shows that Algorithm 1 is guaranteed to terminate and to return a correct ϵ-tight approximation on the volume $\mathrm{WMI}(S', 1) = \mathrm{WMI}(S, w)$, as desired.

Theorem 1 (Termination and correctness, Proof in the extended version [Akshay *et al.*, 2025a])**.** *Given a bounded semi-algebraic set S with bounds $S \subseteq \prod_{i=1}^{n}[a_i, b_i]$, a non-negative semi-algebraic weight function w and a precision bound $\epsilon > 0$, Algorithm 1 is guaranteed to terminate and returns a value M such that*

$$WMI(S, w) - \epsilon \leq M \leq WMI(S, w) + \epsilon.$$

4 Experimental Results

Implementation. We implemented our approach in Python 3 and used SymPy [Meurer *et al.*, 2017] and NumPy [Harris *et al.*, 2020] for symbolic computations. We also employed Gurobi [Gurobi Optimization, LLC, 2023] to solve the resulting LP instances. Moreover, we integrated our tool (WMI-LP) into the state-of-the-art WMI framework of [Spallitta *et al.*, 2024], thus enabling its direct and user-friendly application not only for volume computation but also to WMI instances. Both versions of the tool [Akshay *et al.*, 2025b], i.e. standalone or integrated with [Spallitta *et al.*, 2024], are free and open-source software and publicly available at GitHub[1].

Machine. All experiments were performed on an Intel Xeon Gold 5115 CPU (2.40GHz, 16 cores) running Ubuntu 20.04 with 64 GB of RAM.

Baselines. We compared our approach against the two integration solvers available in the WMI framework of [Spallitta *et al.*, 2024]. This includes LattE [De Loera *et al.*, 2004], which is a symbolic integrator, and VolEsti [Chalkis and Fisikopoulos, 2020], which is sampling-based. Additionally, we also compared against state-of-the-art probabilistic inference tools, namely PSI [Gehr *et al.*, 2016] and GuBPI [Beutner *et al.*, 2022]. Moreover, we also compared our tool against Mathematica [Inc., 2024a]. We did not include WolframAlpha [Inc., 2024b] in our evaluation, as it does not provide direct control over precision or error bounds, and it runs on cloud infrastructure with unspecified computational resources. This makes it unsuitable for a fair, apples-to-apples comparison, unlike Mathematica, which was executed locally on the same machine as our tool. In all experiments, we set $\epsilon = 0.1$ and enforce a time limit of 1 hour per instance for all baselines.

Input Instances. To showcase the merits and limitations of each approach, we considered two families of functions as our benchmarks:

[1]Our tool is available at: https://github.com/destrat18/wmilp.

	WMI-LP		GuBPI		VolEsti		LattE	PSI	Mathematica
	Lower	**Upper**	**Lower**	**Upper**	**Lower**	**Upper**			
A12	5.352712	5.452689	0.891150	5.404662	5.155940	5.597974	ER	SY	5.401414
A17	31.511295	31.611293	0.000000	31.569187	29.321834	32.967666	ER	SY	31.562522
A22	4.045379	4.145252	0.000000	4.093632	3.763386	4.315086	ER	SY	4.093190
A30	7.981809	8.081765	0.000000	8.032287	7.591300	8.382684	ER	SY	8.031472
A41	1.536057	1.636048	1.206099	1.975841	1.567089	1.681303	ER	TO	SY
A49	3.846735	3.946729	0.023396	4.013448	3.871012	4.206076	ER	ER	SY
A53	1.628985	1.728967	1.311641	2.054783	1.604541	1.755675	ER	TO	SY
B11	8.588160	8.688141	0.000000	8.638746	N/S	N/S	N/S	SY	CX
B17	7.498986	7.598938	0.000000	7.548972	N/S	N/S	N/S	SY	CX
B18	7.882580	7.982558	0.000000	7.930250	N/S	N/S	N/S	7.930080	7.930080
B24	4.918883	5.018882	1.059239	4.972304	N/S	N/S	N/S	SY	4.968923
B32	1.996137	2.096055	2.042692	2.045879	N/S	N/S	N/S	SY	ER
B35	0.680046	0.778197	0.726561	0.726887	N/S	N/S	N/S	SY	0.726724
B38	1.159555	1.259449	1.209101	1.210712	N/S	N/S	N/S	SY	1.209906

Table 1: A selection of the experimental results. See the extended version [Akshay *et al.*, 2025a] for the complete table. N/S indicates input not supported, TO indicates a timeout, CX a complex-valued result, SY a symbolic result, and ER an execution error.

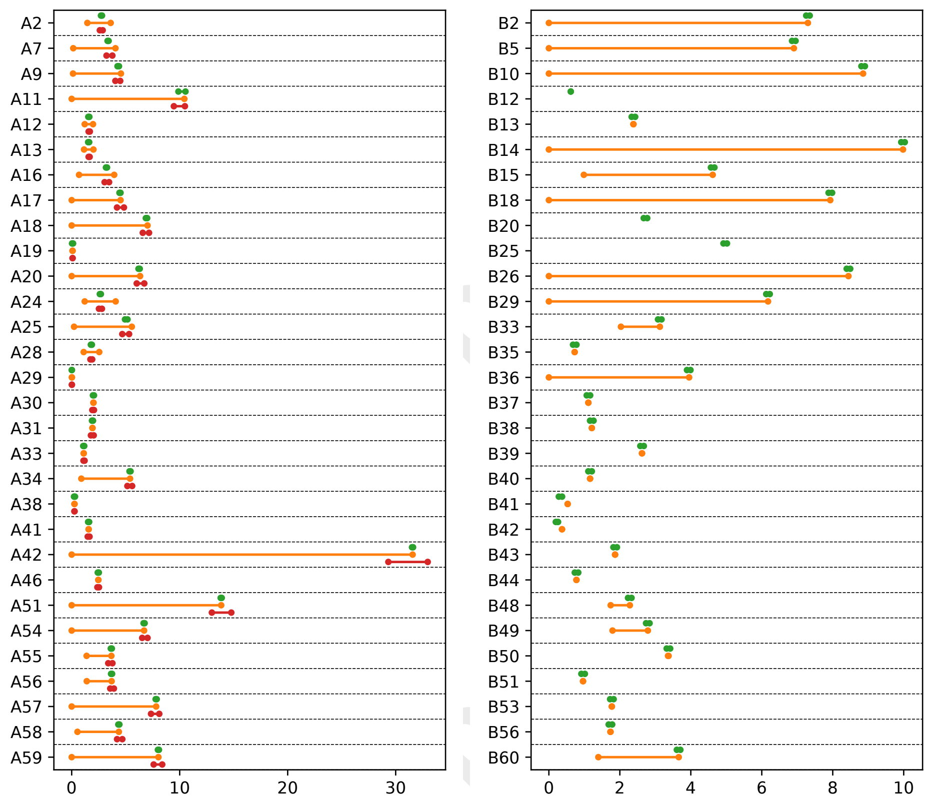

Figure 1: Comparison of the intervals obtained by our approach (WMI-LP) (green), GuBPI (orange), and VolEsti (red) for a selection of benchmarks. See the extended version [Akshay *et al.*, 2025a] for the entire set of results.

(A) ***Randomly-generated Rational Functions.*** To ensure unbiased evaluation and demonstrate the robustness of our method across diverse and arbitrary cases, we generated a set of 60 rational functions, half with one variable and the other half with two variables. The polynomials used in these rational functions are of degree two or three and their coefficients were picked randomly.

(B) ***Randomly-generated Radical Functions.*** This benchmark family is similar to the previous case, except that we consider radical functions instead of mere rational functions. This set also contains 60 benchmarks, half of which are square roots of polynomials. As in the previous case, the coefficients are chosen randomly.

Details of Benchmarks. The extended version [Akshay *et al.*, 2025a] contains details of all the functions used as benchmarks.

Summary of Results. Table 1 reports the experimental results over a selection of the benchmarks. Due to the page restrictions, a complete table of results, showing the performance of our approach and every baseline over every benchmark function is available in the extended version [Akshay *et al.*, 2025a]. We now highlight a couple of important findings.

- ***Supported Functions.*** As evident in Table 1, our approach is robust in handling a wide variety of polynomial, rational and radical functions. It successfully finds bounds for every benchmark function. In contrast, LattE could not solve any of the benchmarks, i.e. 0 out of 120, and VolEsti could only support benchmarks in (A), failing on all benchmarks in (B). GuBPI solved all instances in (A) and 53 of the 60 instances in (B), whereas PSI solved 2 out of 60 instances in (A) and 18 out of 60 in (B). Mathematica was able to solve 30 out of 60 instances in (A) and 31 out of 60 in (B). However, for 28 instances in (A) and 6 in (B), it returned symbolic expressions instead of numerical bounds, which were not usable for further downstream tasks. Additionally, it produced complex-valued results for 1 instance in (A) and 20 instances in (B). For the remaining cases, Mathematica encountered execution errors. Thus, our LP-based WMI approach is applicable to functions that were beyond the scope of all previous state-of-the-art tools.
- ***Errors.*** As demonstrated in Section 3, our approach is able to guarantee any desired bound ϵ on the error. We used $\epsilon = 0.1$ in the experiments. Among previous approaches, LattE and PSI are exact but, as mentioned above, can only handle a small portion of the benchmarks. In contrast, VolEsti and GuBPI are approximate but applicable to more instances. Figure 1 provides a comparison between the error bounds obtained by our approach (green), GuBPI (orange) and VolEsti (red) for a selected benchmarks. The results for each benchmark and each approach is available in the extended version [Akshay *et al.*, 2025a], where we have visualized the interval between the established lower and upper-bounds. This figure illustrates our approach's ability to consistently guarantee small errors over all benchmarks. In constrast, the previous methods almost always provide looser bounds. Moreover, their performance is not consistent. While they can find relatively tight bounds on some benchmarks, they provide extremely loose approximations on others. It is also noteworthy that VolEsti fails on all benchmarks of family (B). Finally, even though GuBPI is able to handle the vast majority of the benchmarks, Figure 1 shows that it often has errors that are orders of magnitude larger than those of our approach.

In summary, our experimental results demonstrate that our LP-based WMI approach is able to handle functions that were beyond the reach of previous state-of-the-art tools. Previous exact approaches are applicable only to a small portion of the benchmarks, for which they are able to synthesize exact solutions. Unfortunately, they fail to find an answer in a vast majority of the cases. Conversely, previous approximate approaches are generally able to handle more instances but with considerably larger error than ours. Thus, our LP-based WMI approach significantly improves the state-of-the-art in WMI for semi-algebraic sets and functions in terms of both applicability and accuracy.

Conclusion

In this work, we presented a novel algorithm for computing Weighted Model Integration (WMI) for a class of semi-algebraic functions, including rational and radical functions, with literals defined over non-linear real arithmetic. Our algorithm leverages Farkas' lemma and Handelman's theorem from real algebraic geometry to reduce the WMI problem to solving a number of linear programming (LP) instances, providing a computationally efficient and tractable approach. The algorithm guarantees a formal bound on the approximation error, ensuring that for any user-specified error $\epsilon > 0$, the returned approximation is ϵ-tight. Moreover, the algorithm is parallelizable, enabling further scalability.

We also provided experimental results demonstrating the superior performance of our algorithm compared to state-of-the-art tools in WMI and probabilistic inference for rational and radical functions. Specifically, our method achieves tighter error bounds and solves a larger number of instances. Additionally, we integrated our tool (WMI-LP) with the state-of-the-art WMI framework to extend its support for rational and radical functions.

Acknowledgments

The research was supported by the Asian Universities Alliance Scholars Award Program (AUASAP), which financed a visit by S. Akshay to HKUST and another visit by A.K. Goharshady to IIT Bombay. The authors are grateful to the Schloss Dagstuhl – Leibniz Center for Informatics; this collaboration started at the Dagstuhl Seminar 23241: "Scalable Analysis of Probabilistic Models and Programs". This work was also partially supported by the SBI Foundation Hub for Data Science & Analytics, IIT Bombay. Author names are ordered alphabetically.

References

[Abboud *et al.*, 2022] Ralph Abboud, İsmail İlkan Ceylan, and Radoslav Dimitrov. Approximate weighted model integration on DNF structures. *Artif. Intell.*, 311:103753, 2022.

[Akshay *et al.*, 2024a] S Akshay, Supratik Chakraborty, Amir Kafshdar Goharshady, R Govind, Harshit J Motwani, and Sai Teja Varanasi. Automated synthesis of decision lists for polynomial specifications over integers. In *Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR 2024)*, 2024.

[Akshay *et al.*, 2024b] S Akshay, Supratik Chakraborty, Amir Kafshdar Goharshady, R Govind, Harshit Jitendra Motwani, and Sai Teja Varanasi. Practical approximate quantifier elimination for non-linear real arithmetic. In *International Symposium on Formal Methods*, pages 111–130. Springer, 2024.

[Akshay *et al.*, 2025a] S. Akshay, Supratik Chakraborty, Soroush Farokhnia, Amir Goharshady, Harshit Jitendra Motwani, and Djordje Zikelic. Lp-based weighted model integration over non-linear real arithmetic. extended version. (hal-05071513), 2025.

[Akshay *et al.*, 2025b] S. Akshay, Supratik Chakraborty, Soroush Farokhnia, Amir Goharshady, Harshit Jitendra Motwani, and Djordje Zikelic. Tool - WMI-LP: Weighted model integration using linear programming. https://github.com/destrat18/wmilp, 2025.

[Albarghouthi *et al.*, 2017] Aws Albarghouthi, Loris D'Antoni, Samuel Drews, and Aditya V Nori. Fairsquare: probabilistic verification of program fairness. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA):1–30, 2017.

[Belle *et al.*, 2015] Vaishak Belle, Andrea Passerini, and Guy Van den Broeck. Probabilistic inference in hybrid domains by weighted model integration. In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 2770–2776. AAAI Press, 2015.

[Beutner *et al.*, 2022] Raven Beutner, C-H Luke Ong, and Fabian Zaiser. Guaranteed bounds for posterior inference in universal probabilistic programming. In *Proceedings of the 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 536–551, 2022.

[Bouchaud and Potters, 2003] Jean-Philippe Bouchaud and Marc Potters. *Theory of financial risk and derivative pricing: from statistical physics to risk management*. Cambridge university press, 2003.

[Chakraborty *et al.*, 2016] Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic SAT calls. In Subbarao Kambhampati, editor, *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 3569–3576. IJCAI/AAAI Press, 2016.

[Chalkis and Fisikopoulos, 2020] Apostolos Chalkis and Vissarion Fisikopoulos. volesti: Volume approximation and sampling for convex polytopes in r. *arXiv preprint arXiv:2007.01578*, 2020.

[Chatterjee *et al.*, 2025] Krishnendu Chatterjee, Ehsan Kafshdar Goharshady, Mehrdad Karrabi, Harshit J Motwani, Maximilian Seeliger, and Djordje Zikelic. Quantified linear and polynomial arithmetic satisfiability via template-based skolemization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 11158–11166, 2025.

[Chavira and Darwiche, 2008] Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artif. Intell.*, 172(6-7):772–799, 2008.

[Cimatti *et al.*, 2013] Alessandro Cimatti, Alberto Griggio, Bastiaan Joost Schaafsma, and Roberto Sebastiani. The mathsat5 smt solver. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 93–107. Springer, 2013.

[Colón *et al.*, 2003] Michael A Colón, Sriram Sankaranarayanan, and Henny B Sipma. Linear invariant generation using non-linear constraint solving. In *Computer Aided Verification: 15th International Conference, CAV 2003, Boulder, CO, USA, July 8-12, 2003. Proceedings 15*, pages 420–432. Springer, 2003.

[De Loera *et al.*, 2004] Jesús A De Loera, Raymond Hemmecke, Jeremiah Tauzer, and Ruriko Yoshida. Effective lattice point counting in rational convex polytopes. *Journal of symbolic computation*, 38(4):1273–1302, 2004.

[Dehnert *et al.*, 2015] Christian Dehnert, Sebastian Junges, Nils Jansen, Florian Corzilius, Matthias Volk, Harold Bruintjes, Joost-Pieter Katoen, and Erika Ábrahám. Prophesy: A probabilistic parameter synthesis tool. In *Computer Aided Verification: 27th International Conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part I 27*, pages 214–231. Springer, 2015.

[Dos Martires *et al.*, 2019] Pedro Zuidberg Dos Martires, Anton Dries, and Luc De Raedt. Exact and approximate weighted model integration with probability density functions using knowledge compilation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7825–7833, 2019.

[Farkas, 1902] Julius Farkas. Theory of simple inequalities. *Journal for pure and applied mathematics (Crelles Journal)*, 1902(124):1–27, 1902.

[Gehr *et al.*, 2016] Timon Gehr, Sasa Misailovic, and Martin Vechev. Psi: Exact symbolic inference for probabilistic programs. In *Computer Aided Verification: 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17-23, 2016, Proceedings, Part I 28*, pages 62–83. Springer, 2016.

[Gurobi Optimization, LLC, 2023] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.

[Handelman, 1988] David Handelman. Representing polynomials by positive linear functions on compact convex polyhedra. *Pacific Journal of Mathematics*, 132(1):35–62, 1988.

[Harris *et al.*, 2020] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 2020.

[Henrion *et al.*, 2009] Didier Henrion, Jean B Lasserre, and Carlo Savorgnan. Approximate volume and integration for basic semialgebraic sets. *SIAM review*, 51(4):722–743, 2009.

[Humphries *et al.*, 2010] Nicolas E Humphries, Nuno Queiroz, Jennifer RM Dyer, Nicolas G Pade, Michael K Musyl, Kurt M Schaefer, Daniel W Fuller, Juerg M Brunnschweiler, Thomas K Doyle, Jonathan DR Houghton, et al. Environmental context explains lévy and brownian movement patterns of marine predators. *Nature*, 465(7301):1066–1069, 2010.

[Inc., 2024a] Wolfram Research Inc. Mathematica, version 14.2. https://www.wolfram.com/mathematica, 2024. Accessed: 2025-05-20.

[Inc., 2024b] Wolfram Research Inc. Wolfram alpha, version 14.2. https://www.wolfram.com/wolfram-alpha-notebook-edition, 2024. Accessed: 2025-05-20.

[Lairez *et al.*, 2019] Pierre Lairez, Marc Mezzarobba, and Mohab Safey El Din. Computing the volume of compact semi-algebraic sets. In *Proceedings of the 2019 on International Symposium on Symbolic and Algebraic Computation*, pages 259–266, 2019.

[Meurer *et al.*, 2017] Aaron Meurer, Christopher P Smith, Mateusz Paprocki, Ondrej Certik, Sergey B Kirpichev, Matthew Rocklin, AMiT Kumar, Sergiu Ivanov, Jason K Moore, Sartaj Singh, et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, 2017.

[Morettin *et al.*, 2017] Paolo Morettin, Andrea Passerini, and Roberto Sebastiani. Efficient weighted model integration via smt-based predicate abstraction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 720–728. AAAI Press, 2017.

[Morettin *et al.*, 2021] Paolo Morettin, Pedro Zuidberg Dos Martires, Samuel Kolb, and Andrea Passerini. Hybrid probabilistic inference with logical and algebraic constraints: a survey. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 4533–4542. ijcai.org, 2021.

[Riley *et al.*, 2006] Kenneth Franklin Riley, Michael Paul Hobson, and Stephen John Bence. *Mathematical methods for physics and engineering: a comprehensive guide*. Cambridge university press, 2006.

[Rom and Brakhage, 2011] Michael Rom and Karl-Heinz Brakhage. *Volume mesh generation for numerical flow simulations using Catmull-Clark and surface approximation methods*. Inst. für Geometrie und Praktische Mathematik Aachen, Germany, 2011.

[Sankaranarayanan *et al.*, 2013] Sriram Sankaranarayanan, Aleksandar Chakarov, and Sumit Gulwani. Static analysis for probabilistic programs: inferring whole program properties from finitely many paths. In *Proceedings of the 34th ACM SIGPLAN conference on Programming language design and implementation*, pages 447–458, 2013.

[Sims *et al.*, 2008] David W Sims, Emily J Southall, Nicolas E Humphries, Graeme C Hays, Corey JA Bradshaw, Jonathan W Pitchford, Alex James, Mohammed Z Ahmed, Andrew S Brierley, Mark A Hindell, et al. Scaling laws of marine predator search behaviour. *Nature*, 451(7182):1098–1102, 2008.

[Spallitta *et al.*, 2024] Giuseppe Spallitta, Gabriele Masina, Paolo Morettin, Andrea Passerini, and Roberto Sebastiani. Enhancing smt-based weighted model integration by structure awareness. *Artificial Intelligence*, 328:104067, 2024.

[Tankov, 2003] Peter Tankov. *Financial modelling with jump processes*. Chapman and Hall/CRC, 2003.