Weighted Model Integration with Orthogonal Transformations

David Merrell and **Aws Albarghouthi** and **Loris D'Antoni**
University of Wisconsin—Madison
Department of Computer Sciences
{dmerrell, aws, loris}@cs.wisc.edu

Abstract

Weighted model counting and integration (WMC/WMI) are natural problems to which we can reduce many probabilistic inference tasks, e.g., in Bayesian networks, Markov networks, and probabilistic programs. Typically, we are given a first-order formula, where each satisfying assignment is associated with a weight—e.g., a probability of occurrence—and our goal is to compute the total weight of the formula. In this paper, we target *exact inference* techniques for WMI that leverage the power of *satisfiability modulo theories* (SMT) solvers to decompose a first-order formula in *linear real arithmetic* into a set of *hyperrectangular regions* whose weight is easy to compute. We demonstrate the challenges of hyperrectangular decomposition and present a novel technique that utilizes *orthogonal transformations* to transform SMT formulas in order to enable efficient inference. Our evaluation demonstrates our technique's ability to improve the time required to achieve exact probability bounds.

1 Introduction

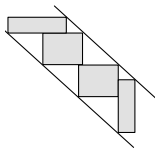
Motivation Weighted model counting (WMC) is a natural problem to which we can reduce many probabilistic inference tasks, for instance, in Bayesian networks and probabilistic programs [Chavira and Darwiche, 2008; Chistikov *et al.*, 2015; Fierens *et al.*, 2015]. In WMC, we are given a propositional formula, where each satisfying assignment has a weight—e.g., a probability of occurrence—and our goal is to compute the total weight of the formula. Recently, there have been a number of works that leverage the power of state-of-the-art *satisfiability* (SAT) solvers as oracles for solving WMC problems [Chakraborty *et al.*, 2014; 2013].

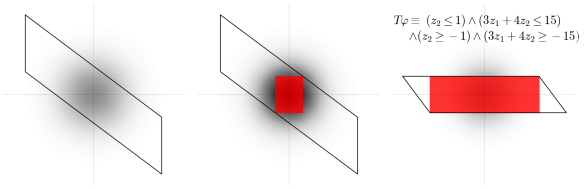
To handle richer domains, researchers from the artificial intelligence, as well as the formal methods, community have also started tackling the *weighted model integration* (WMI) problem [Belle *et al.*, 2015b; 2015a; 2016; Chistikov *et al.*, 2015; 2014], where formulas are over infinite domains, e.g., real arithmetic. The idea is that inference in, for instance, hybrid Markov networks and rich probabilistic programs, can be reduced to WMI problems. Recent works tackling the WMI problem have primarily targeted the first-order theory of linear real arithmetic (LRA), which is well supported in state-of-the-art satisfiability modulo theories (SMT) solvers [Barrett *et al.*, 2009].

Problem setting In this paper, we continue the investigation of WMI algorithms, with an emphasis on a versatile class of techniques for formulas in LRA. Given a formula φ in real arithmetic with k free variables, we view φ as a region in $\mathbb{R}^n$. We assume the value of each free variable x in φ is drawn independently from some probability distribution $\mathcal{D}_x$. Our goal is to compute the probability of satisfaction of φ.

A number of recent works [Albarghouthi *et al.*, 2017a; 2016; Sankaranarayanan *et al.*, 2013; Chistikov *et al.*, 2015] have attacked this problem by decomposing φ into a set of *hyperrectangles* (rectangles in $\mathbb{R}^n$). The idea is that computing the *weight* of a hyperrectangular region is simple, as all dimensions have constant bounds. Of course, there may be infinitely many hyperrectangles in the decomposition of φ. By considering more and more hyperrectangles in φ, we approach the true weight of φ from below, and converge in the limit. Similarly, we can approach the weight of the *negation* of φ ($\neg\varphi$) from below, thus approaching the weight of φ from above. A key feature of this class of techniques is that we can terminate the process at any point and retrieve exact lower and upper bounds on φ's weight. This is often desirable in verification of programs with uncertainty, where we are interested in proving some upper/lower bound on the probability of a *bad* event [Sankaranarayanan *et al.*, 2013; Bouissou *et al.*, 2016; Sampson *et al.*, 2014].

Challenges addressed The foremost challenge in hyperrectangular decomposition is that it is hard to *fill* a slanted region with hyperrectangles; that is, we might require a large number of hyperrectangles, making convergence incredibly slow. (See illustration on the right where gray rectangles are used to fill the region between the two slanted faces.) To tackle this challenge, we apply a *linear transformation* (e.g., a rotation) to the given formula φ in order to make its faces *axis-aligned*. In doing so, we must (*i*) understand how transforming φ affects its weight, and (*ii*) ensure that hyperrectangular decomposition can still be applied to the *transformed* formula. The problem is that, in





(a) (b) (c)

Figure 1: (a) Example formula φ; joint distribution of x_1, x_2 is indicated by shading. (b) WMI by hyperrectangular decomposition converges slowly in this situation. (c) If a transformation were possible, then WMI would converge faster by sampling larger hyperrectangles.

general, transforming a formula may change its weight; may produce irrational coefficients (preventing the use of SMT solvers); or may produce stochastic dependence between its free variables (undermining a key premise of hyperrectangular decomposition).

Contributions We make the following key contributions:

- We argue that hyperrectangular decomposition on LRA formulas can benefit from linear transformations, but show that this is only true for formulas with Gaussian free variables. Additionally, we show the desirable properties of *orthogonal transformations* in this setting. (Section 3)
- We show how to construct orthogonal transformation matrices with rational entries, using *Givens rotations* and *Pythagorean triples*. We also provide a method to obtain suitable transformations for formulas. (Section 4)
- We implement and evaluate our approach using the Z3 SMT solver; on representative benchmarks, our approach often dramatically improves performance of weighted model integration tasks. (Section 5)

2 Motivation and Illustration

As a motivating example, suppose we have the independent random variables $x_1 \sim \mathcal{N}(0,1)$ and $x_2 \sim \mathcal{N}(0,1)$. Let φ be a first-order formula in the theory of *linear real arithmetic* (LRA) over variables x_1 and x_2:

$$\varphi \equiv (3x_1 + 4x_2 \leqslant 5) \wedge (x_1 \leqslant 3) \\ \wedge (3x_1 + 4x_2 \geqslant -5) \wedge (x_1 \geqslant -3)$$

Intuitively, the satisfying assignments of φ can be viewed as a region in $\mathbb{R}^2$, as illustrated in Figure 1(a), where φ is the region inside the parallelogram. (For simplicity, we illustrate our approach on a convex polyhedron; however, the class of formulas we consider, LRA, admits formulas with disjunctions, i.e., unions of polyhedra.)

Hyperrectangular decomposition Our goal is to compute the probability that φ is satisfied. Formally, we want to evaluate the following integral: $\int_\varphi p(x_1, x_2) dx_1 dx_2$, where $p(x_1, x_2)$ is the joint probability density function (PDF) of x_1 and x_2.

A number of recent techniques use SMT solvers to decompose the region φ into an infinite set of non-overlapping rectangles $H_1, H_2, \ldots$, such that $\bigvee_{i=1}^{\infty} H_i \equiv \varphi$. The idea is that computing the weight of a hyperrectangular region is easy, as all dimensions have constant upper and lower bounds. Intuitively, the weight of φ is the sum of weights of all H_i: $\sum_{i=1}^{\infty} \left(\int_{H_i} p(x_1, x_2) dx_1 dx_2 \right)$. Thus, recent techniques [Sankaranarayanan *et al.*, 2013; Albarghouthi *et al.*, 2017a] iteratively compute the weight of more and more hyperrectangles H_i, converging to the weight of φ in the limit.

For the purposes of our work here, we are primarily concerned with the *quality* of the hyperrectangles that the algorithm considers. Intuitively, the larger the hyperrectangles, the faster the convergence. In the case of our formula φ, the largest possible hyperrectangle in terms of weight is shown in Figure 1(b). It is easy to see that this rectangle covers a relatively small subset of φ. This is due to the two slanted faces of φ, which force us to find hyperrectangles that only *touch* the faces with two of their corners.

Applying linear transformations Ideally, we would decompose φ into a series of hyperrectangles where most of the weight is concentrated in a small number of rectangles, thus accelerating convergence to the true weight of φ. To do so, we observe that we can apply a linear transformation T to the region φ, resulting in a new region $T\varphi$ that admits a better hyperrectangular decomposition. In our example, we might choose T to be a *rotation* φ that aligns the two slanted faces with an axis. The resulting formula $T\varphi$ is shown in Figure 1(c). The red region is the rectangle with the largest weight that fits in $T\varphi$. Observe how this rectangle covers the densest region—around the origin.

In this illustrative example, we assumed that we can rotate the formula such that some of its faces are axis-aligned. This is not always possible. It may be that the "best" transformation matrix contains irrational entries, leading to irrational constants in the transformed formula. Meanwhile, SMT solvers expect formulas to have rational constants. Later, we will show how to construct transformation matrices that make formulas better-conditioned for hyperrectangular decomposition, without introducing irrational constants.

3 Problem Definition

We now formalize our weighted model integration problem.

Formulas We consider formulas in the *quantified linear real arithmetic* (LRA) fragment of first-order logic. LRA is supported by state-of-the-art SMT solvers like Z3 [De Moura and Bjørner, 2008].

A *formula* φ is constructed using the following grammar:

$$\varphi := c_1 x_1 + \ldots + c_n x_n \leqslant c \mid \varphi \wedge \varphi \\ \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists x.\, \varphi \mid \forall x.\, \varphi$$

where $c_i, c \in \mathbb{R}$. We use the column vector $\boldsymbol{x}$ to denote the variables that are *free* in φ (i.e., not bound by a $\exists$ or $\forall$ quantifier). Note that LRA is closed under quantifier elimination: for every formula φ, there is an equivalent LRA formula φ' that is quantifier free.

We will use $\varphi[\boldsymbol{x} \mapsto \boldsymbol{x'}]$ to denote φ with occurrences of variables $\boldsymbol{x}$ replaced by respective elements of $\boldsymbol{x'}$.

Weighted model integration Given a formula φ with k free variables $\boldsymbol{x}^{\mathrm{T}} = [x_1, \ldots, x_k]$, we assume $\boldsymbol{x}$ is a vector of independently distributed random variables.

We can view the formula φ as a region in $\mathbb{R}^k$. We define the *weight* of φ as follows: $\mathrm{WMI}(\varphi) = \int_\varphi p(\boldsymbol{x}) dx_1 \ldots dx_k$, where $p(\boldsymbol{x})$ is the joint probability density function of $\boldsymbol{x}$.

Hyperrectangular decomposition We describe the algorithm we shall use for weighted model integration. The details of the algorithm are not important for our exposition—only the high-level idea. We refer the reader to past works for details [Sankaranarayanan *et al.*, 2013; Albarghouthi *et al.*, 2017a; 2016].

We will use H for an LRA formula defining a hyperrectangular region in $\mathbb{R}^k$; i.e., H is of the form $\bigwedge_{i=1}^{k} l_i \leqslant x_i \leqslant u_i$, where $l_i, u_i \in \mathbb{R}$. A key point is that $\text{WMI}(H)$ is easily computed using the *cumulative density functions* (CDFs) $F_1, \ldots, F_k$ of $x_1, \ldots, x_k$, since the upper and lower bounds of each dimension are constant, and the x_i are independent: $\text{WMI}(H) = \prod_i (F_i(u_i) - F_i(l_i))$.

A *hyperrectangular decomposition* of a formula φ is an infinite sequence of hyperrectangles $H_1, H_2, \ldots$ such that (*i*) $\bigvee_i H_i \equiv \varphi$ and (*ii*) $H_i \wedge H_j \equiv \mathit{false}$ for any $i \neq j$.

Recent approaches [Albarghouthi *et al.*, 2017a; Sankaranarayanan *et al.*, 2013] use hyperrectangular decomposition to compute the weight of φ, by summing the hyperrectangles' individual weights.

At any point during this summation, the sum is a lower bound on $\text{WMI}(\varphi)$, and convergence to the actual weight is guaranteed in the limit.[1] In order to additionally compute an upper bound on $\text{WMI}(\varphi)$, one could simply invoke this process on $\neg\varphi$, since we know that $\text{WMI}(\varphi) + \text{WMI}(\neg\varphi) = 1$.

Permissible distributions Note that hyperrectangular decomposition requires the formula's free variables to be independently distributed. The following theorem, credited to Skitovitch and Darmois, characterizes the class of distributions that remain independent under linear transformation [Skitovitch, 1953; Darmois, 1953].

Theorem 1. *Let $L_1 = \sum_i \alpha_i x_i$ and $L_2 = \sum_i \beta_i x_i$, with x_i independent real random variables and $\alpha_i, \beta_i \in \mathbb{R}$. If L_1 and L_2 are independent, then all x_j with nonzero α_j, β_j must be Gaussian.*

In short: nontrivial linear combinations of independent random variables are independent *only if those random variables are Gaussian*. It follows that only formulas with *Gaussian* free variables can be subjected to a linear transformation, and then integrated via hyperrectangular decomposition.

Canonical forms Without further loss of generality, we assume that all of our weighted model integration problems are over formulas φ where $\boldsymbol{x} \sim \mathcal{N}_k(0, I)$, where I is the identity matrix.

Note that we can transform any formula φ with variables $\boldsymbol{x} \sim \mathcal{N}_k(\mu, \Sigma)$ into a *canonical form* φ' with variables $\boldsymbol{x'} \sim \mathcal{N}_k(0, I)$, such that $\text{WMI}(\varphi) = \text{WMI}(\varphi')$.

To perform this transformation, we exploit the following well-known property of multivariate Gaussians: There exists a matrix $C \in \mathbb{R}^{k \times k}$ such that if $\boldsymbol{z} = C^{-1}(\boldsymbol{x} - \mu)$, then $\boldsymbol{z} \sim \mathcal{N}_k(0, I)$. The following theorem characterizes correctness of this transformation. The idea is that we can replace the variables $\boldsymbol{x}$ of φ with the new variables $\boldsymbol{z}$, creating a new formula φ' whose free variables are $\boldsymbol{z}$.

[1] In the case φ is decomposable into finitely many hyperrectangles, then the algorithm can converge in finitely many steps.

Theorem 2. *Let φ have free variables $\boldsymbol{x} \sim \mathcal{N}_k(\mu, \Sigma)$. Let C be a matrix such that $\boldsymbol{z} = C^{-1}(\boldsymbol{x} - \mu)$ and $\boldsymbol{z} \sim \mathcal{N}_k(0, I)$, where $\boldsymbol{z}$ is a vector of fresh (unused) variables. Let $\varphi' \equiv \exists \boldsymbol{x}.\, \varphi \wedge \boldsymbol{z} = C^{-1}(\boldsymbol{x} - \mu)$. Then, $\text{WMI}(\varphi') = \text{WMI}(\varphi)$.*

Henceforth, we shall assume that all weighted model integration problems are over formulas in canonical form.

Example 1. Consider the simple formula $\varphi \equiv x > 0$, where $x \sim \mathcal{N}(10, 20)$. We can construct the canonical form $\varphi' \equiv \exists x.\, z = (x - 10)/20 \wedge x > 0$, where $z \sim \mathcal{N}(0, 1)$. We have $\text{WMI}(\varphi) = \text{WMI}(\varphi')$. ■

Orthogonal transformations While theorem 1 shows that only Gaussian distributions can retain independence under linear transformations, it does not characterize the linear transformations which make that possible. We show that *orthogonal transformations*—rotations and reflections—preserve independence in our setting.

Suppose we have a formula φ in canonical form. Let $p(\boldsymbol{x})$ denote the joint PDF of its free variables, and let T be a linear transformation. Then we know the joint density $p(\boldsymbol{x}) \propto \exp\left(-\frac{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}}{2}\right)$, and therefore the *transformed* density $p(T\boldsymbol{x}) \propto \exp\left(-\frac{\boldsymbol{x}^{\mathrm{T}}T^{\mathrm{T}}T\boldsymbol{x}}{2}\right)$. If the matrix T is orthogonal, then $T^{\mathrm{T}}T = I$, and $p(T\boldsymbol{x}) = p(\boldsymbol{x})$; hence, the free variables will remain independent.

Orthogonal transformations also have the convenient property of *preserving weight*. If T is an orthogonal matrix, then $|\det T| = 1$. It follows from *integration by substitution of variables* that $\text{WMI}(\varphi) = \text{WMI}(T\varphi)$.

4 WMI with Orthogonal Transformations

As suggested previously, orthogonal transformations can make formulas better-conditioned for WMI. However, this strategy comes with challenges. First, SMT solvers expect formulas with rational coefficients, but orthogonal matrices will generally have irrational entries. A naïve approximation of irrational entries with rationals would generally produce a non-orthogonal transformation that preserves neither independence nor weight. Second, choosing a transformation that maximizes the efficiency of WMI for a formula entails a difficult optimization problem.

In this section, we present (*i*) a technique for constructing orthogonal matrices with rational entries and (*ii*) a technique for choosing a favorable matrix.

4.1 Orthogonal Matrices with Rational Entries

Our method for constructing orthogonal matrices with rational entries relies on *Givens rotations*. A Givens rotation is an (orthogonal) matrix of the form

$$G_{i,j} = \frac{1}{c}\begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 & \\ \vdots & \ddots & \vdots & & \vdots & & \vdots & \\ 0 & \cdots & a & \cdots & -b & \cdots & 0 & i \\ \vdots & & \vdots & \ddots & \vdots & & \vdots & \\ 0 & \cdots & b & \cdots & a & \cdots & 0 & j \\ \vdots & & \vdots & & \vdots & \ddots & \vdots & \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 & \end{bmatrix} \\ \qquad\qquad\quad i \qquad\quad j$$

where $c = \sqrt{a^2 + b^2}$. This matrix rotates vectors in $\mathbb{R}^k$, but only acts on their i^{th} and j^{th} components. Its entries $\frac{a}{c}$ and $\frac{b}{c}$ can be interpreted as $\cos\theta$ and $\sin\theta$, respectively, where θ is the angle swept by the rotation.

Any orthogonal matrix Q can be constructed as a product of Givens rotations (with an additional trivial reflection in the case that Q is a reflection); in this sense, the Givens rotations are primitives of the orthogonal transformations.

Notice that the entries of G_{ij} are rational whenever a, b, and c are integers; i.e., whenever (a, b, c) form a *Pythagorean triple*. The Pythagorean triples (a, b, c) generate a set of unit vectors $[\frac{a}{c}, \frac{b}{c}]^{\mathrm{T}} \in \mathbb{Q}^2$ that is dense on the unit circle in $\mathbb{R}^2$ [Shiu, 1983]; see Figure 2(c). In consequence, any angle θ can be approximated to arbitrary precision by $\arccos\frac{a}{c} = \arcsin\frac{b}{c}$, where (a, b, c) is a Pythagorean triple. We use Shiu's method (1983) for obtaining (a, b, c), given θ and a precision requirement. In practice, we limit our precision in terms of the lengths of integers a, b, and c—formulas with large integers impose higher computational cost on the SMT solver during hyperrectangular decomposition.

Since a Pythagorean triple (a, b, c) can be generated to approximate any angle $\theta \simeq \arccos(\frac{a}{c}) = \arcsin(\frac{b}{c})$ to arbitrary precision, it follows that we can approximate any Givens rotation $G_{ij} \in \mathbb{R}^{k\times k}$ to arbitrary precision with a *rational* Givens rotation $G_{q;ij} \in \mathbb{Q}^{k\times k}$. Since any orthogonal transformation can be constructed from Givens rotations, we can approximate any orthogonal transformation with a rational one. The next section describes our method for choosing a useful transformation and building its approximation from rational Givens rotations.

4.2 Composing a Suitable Transformation

Throughout this section, the following example will guide our discussion:

$$\varphi_{\text{ex}} \equiv (x_1 + x_2 \leqslant 0) \wedge (x_1 + 2x_2 \leqslant 1),$$

with $x_1, x_2 \sim \mathcal{N}(0, 1)$ iid (see Figure 2(a)). Note that in its current form, φ_{ex} would be poorly suited for hyperrectangular decomposition: most of its weight is near its boundary (around the origin), and would be reachable only by the corners of sampled rectangles. It seems possible that an orthogonal transformation would make φ_{ex} better suited for WMI. In this subsection, we outline a methodology for choosing that transformation.

Aligning one of the faces of the boundary of φ_{ex} with one of the axes would allow the weight near that face to be efficiently captured. However, note that the faces have differing amounts of weight in their proximity; one could say that the face passing through the origin ($x_1+x_2 \leqslant 0$) has more weight than the other ($x_1 + 2x_2 \leqslant 1$), which passes through regions of lower density.

From these observations, we derive the following methodology for choosing an orthogonal transformation, which we exemplify below for φ_{ex}:

1. On each face of φ_{ex}, numerically integrate the joint PDF of x_1, x_2; this surface integral will be called the *weight* of a face. Let the normal vectors of the faces of φ_{ex} be called $\boldsymbol{a}_1 = [1, 1]^{\mathrm{T}}, \boldsymbol{a}_2 = [1, 2]^{\mathrm{T}}$; numerically, we find their weights to be $w_1 \approx 0.37$ and $w_2 \approx 0.18$.
2. Since $w_1 > w_2$, we construct an orthogonal matrix that aligns $\boldsymbol{a}_1$ with one of the axes. The result is $T = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$.
3. Since T contains irrational entries, we cannot use it to transform φ_{ex}. Instead, we approximate T with *rational* T_q. Our example is in two dimensions, so it suffices for T_q to be a single rational Givens rotation:
$$T_q = \frac{1}{169}\begin{bmatrix} 119 & -120 \\ 120 & 119 \end{bmatrix}.$$
4. Transform φ_{ex} with T_q to yield
$$T_q\varphi_{\text{ex}} \equiv (239x_1 - x_2 \leqslant 0) \wedge \left(\frac{359}{169}x_1 + \frac{118}{169}x_2 \leqslant 1\right),$$
illustrated in Figure 2(b).

In two dimensions it suffices to choose one face of the formula and rotate it into alignment with an axis; this fully defines the rotation. In higher dimensions, orthogonal transformations have more degrees of freedom and it becomes more challenging to fully specify them.

We generalize steps 1–3 in the above procedure to higher dimensions in the following way:

– Construct matrix A, whose columns $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n$ are normal unit vectors for the faces of the formula, ordered by their faces' weights.
– Compute a rational approximation of A's *QR-factorization*; $A = QR$ where Q is orthogonal and R is upper-triangular. Specifically, we use a variant of *QR-factorization by Givens rotations*, a well-known method that composes Q from Givens rotations. Our variant is modified to use only *rational* Givens rotations, and results in a rational approximation Q_q of Q.
– Let $T_q = Q_q$.

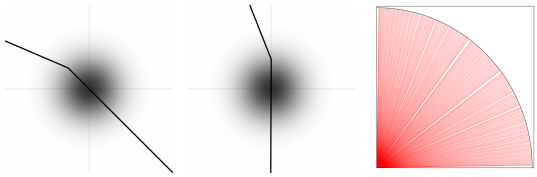
We give a brief description of QR-factorization, and some rationale for using it in this fashion. Every matrix A has a QR-factorization; i.e. $A = QR$ where the columns $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n$ of Q are orthonormal and R is upper-triangular. This factorization is unique, and the columns $\boldsymbol{q}_i$ have the following property:

$$\begin{aligned} \boldsymbol{q}_1 &= \boldsymbol{a}_1, \\ \boldsymbol{q}_2 &= \boldsymbol{a}_2 - \text{ proj}_{\boldsymbol{q}_1}(\boldsymbol{a}_2), \\ &\vdots \\ \boldsymbol{q}_n &= \boldsymbol{a}_n - \text{ proj}_{\boldsymbol{q}_1,\ldots,\boldsymbol{q}_{n-1}}(\boldsymbol{a}_n) \end{aligned}$$

(where normalization is implicit on the RHS).

In effect, Q is orthogonal and transforms faces into alignment with axes, prioritized by the faces' weights. Specifically, $Q^{\mathrm{T}}\boldsymbol{a}_1$ is aligned perfectly with the first axis, $Q^{\mathrm{T}}\boldsymbol{a}_2$ is aligned approximately with the second axis, and so on. The rational Q_q approximates this property, while preserving the formula's rationality.

The computational cost of QR-factorization by Givens rotations is $O(n^3)$. In the context of weighted model integration, this polynomial cost is dwarfed by the worst-case exponential cost incurred by the SMT solver.



(a) (b) (c)

Figure 2: (a) Running example φ_{ex}. (b) Transformed φ_{ex}: $T_q\varphi_{\text{ex}}$. (c) 2,122 rational unit vectors can be constructed in the first quadrant, using integers with $\leqslant 4$ digits.

5 Implementation and Evaluation

Implementation We implemented our orthogonal transformations technique atop an implementation of the algorithm proposed in [Albarghouthi *et al.*, 2017a], which uses the Z3 SMT solver [De Moura and Bjørner, 2008] to perform hyperrectangular decomposition. Constructing a single hyperrectangle involves making a Z3 query. We use the approach presented in Section 4 to construct an orthogonal transformation, after using Redlog [2] to eliminate quantifiers from the formula.

Evaluation Our primary goal in evaluation is to understand the effect of applying orthogonal transformations on SMT formulas prior to performing weighted model integration. To this end, we considered two classes of benchmarks: (*i*) First, we generated random benchmarks of quantifier-free formulas of increasing size, where the task is to compute the weight of the formula, where variables are independently and identically distributed. Our primary finding is that transformations can dramatically speed up convergence, particularly as we increase the number of dimensions. (*ii*) Second, we adapted random-walk benchmarks from the programming languages literature [Chatterjee *et al.*, 2016]. Given a probabilistic program simulating a random walk in $\mathbb{R}^n$, the goal is to compute the probability of landing in a specific region of $\mathbb{R}^n$ after k steps. We encoded these probabilistic programs as SMT formulas, using the methods of Chistikov *et al.* [2015].

Results on synthetic problems For the synthetic benchmarks, we randomly generated formulas using two parameters: (*i*) the number of free variables (3, 5, or 7), and (*ii*) the number of conjuncts in the formula (1, 2, or 3). For each combination of number of free variables and conjuncts, we generated 40 benchmarks. The benchmarks were conjunctions of linear inequalities. The coefficients of each inequality were generated as uniformly distributed unit vectors in 3-, 5-, or 7-dimensional space; the formulas can be understood as conjunctions of randomly oriented halfspaces. We let each formula φ run for 100 seconds, where we, in parallel, ran $\text{WMI}(\varphi)$ and $\text{WMI}(\neg\varphi)$. This is with the exception of the 7-variable/3-conjunct benchmarks, which tended to be too large for the hyperrectangular decomposition method.

Figure 4 shows our results on the synthetic benchmarks. The x-axes represent the sum of $\text{WMI}(\varphi) + \text{WMI}(\neg\varphi)$ after 100 seconds for the case *without* orthogonal transformations; the y-axes represent results *with* orthogonal transformations. Recall that the longer our WMI algorithm runs, the closer it arrives to the actual volume. The closer the value is to 1 the better. We observe that for lower dimensions (3), orthogonal transformations do not provide a significant improvement in computed volume; on benchmarks with 3 free variables and 1 conjunct, an orthogonal transformation gave only a 17% increase to the weight on average. However, as we consider larger dimensions, the effect of orthogonal transformations becomes more pronounced; on average, orthogonal transformations gave a 130% increase in weighted volume for the 5-variable/2-conjunct benchmarks and a 1500% increase for the 7-variable/1-conjunct benchmarks. This dependence on dimension results from the corners of hyperrectangles becoming increasingly prominent in higher dimensions.

For a closer picture of the difference in performance with and without orthogonal transformations, consider Figure 5. Here, we pick two representative benchmarks and plot the value of $1-(\text{WMI}(\varphi)+\text{WMI}(\neg\varphi))$, as computed by the algorithm over the course of 100 seconds (lower value is better). Our results show that WMI with transformations consistently provides better bounds for the duration of execution. Thus, at any point in execution, we can halt weighted model integration and derive better exact bounds in the presence of orthogonal transformations.

Results on random-walk problems For the random walk problems, we generated a set of benchmarks named $n\texttt{d}$-$k\texttt{s}$-t, where n is the space $\mathbb{R}^n$ in which the random walk takes place; k is the number of steps considered; and t is the type of walk: `simp` is a program without conditionals, while `b` and `b2` are more complex walks involving branching.

The goal is to compute the probability of an event φ encoding the position of the walk after k steps. Using the technique of Chistikov *et al.*, we encoded programs as quantified LRA formulas. The sizes of the generated formulas are shown in the table in Figure 3. The largest formula we considered has 90 atomic predicates of the form $\sum_i c_i x_i \leqslant c$, 15 free variables, and 67 existentially quantified variables.

The bar chart in Figure 3 shows the lower bound on the probability after 100s of execution. We notice that orthogonal transformations yielded consistently better results, with the exception of 2 benchmarks. Our results here display a similar pattern to the synthetic benchmarks: the effects of orthogonal transformations are more pronounced for larger dimensions.

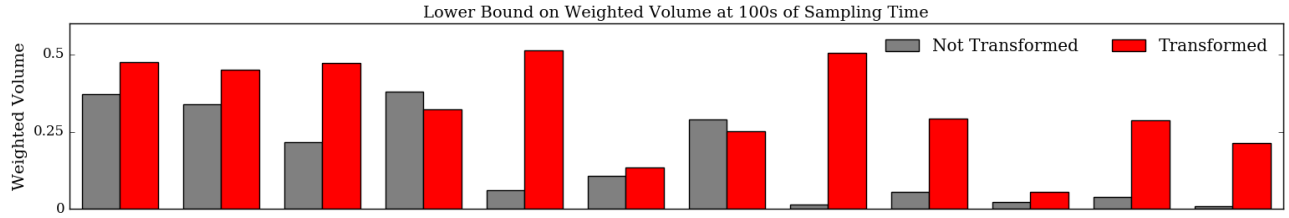
6 Related Work

We now compare our work with existing WMI techniques.

Exact weighted model integration Our work is closely related to exact model integration for formulas in SMT theories. To our knowledge, the first such technique is due to [Ma *et al.*, 2009], where the goal was to compute the volume of a formula φ in LRA. There, the authors iteratively made calls to an off-the-shelf tool—LattE [De Loera *et al.*, 2012]—for computing the volume of a polytope. This technique decomposes the formula into the set of polytopes (effectively, its DNF form). Recent work by Belle et al. [Belle *et al.*, 2015a; 2016] also utilizes LattE as a backend tool, but generalizes the problem to piecewise-polynomial weight specifications. In their setting, weights defined by Gaussian distributions are approximated using piecewise-polynomial functions.

In comparison with the aforementioned works, our ap-

[2] `http://www.redlog.eu/`



Benchmark	1d-2s-simp	1d-2s-b2	1d-3s-simp	1d-3s-b2	2d-2s-simp	2d-2s-b	2d-2s-b2	2d-3s-simp	3d-2s-simp	3d-2s-b	3d-2s-b2	3d-3s-simp
# Predicates	14	36	18	50	25	61	55	31	36	90	76	45
# Free Vars	3	5	4	7	6	10	8	8	9	15	11	8
# Quant. Vars	12	27	16	36	23	46	43	29	34	67	61	43

Figure 3: Random walk problems.

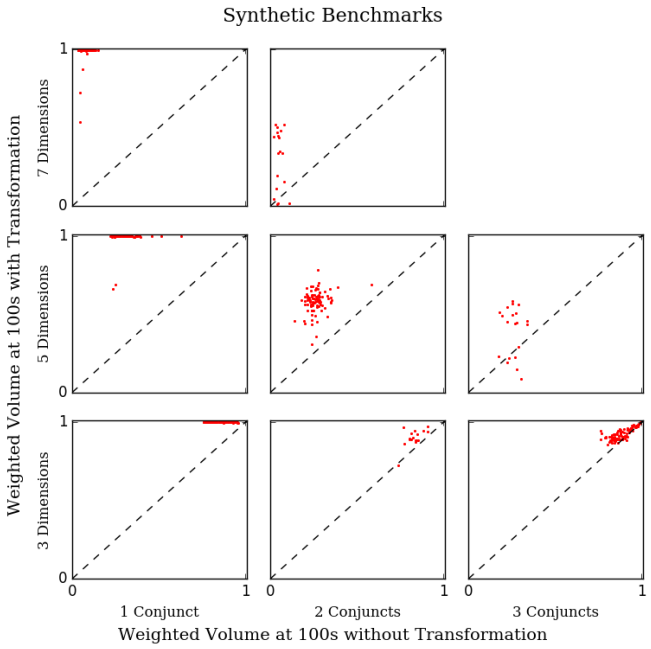


Figure 4: Results of tests on synthetic formulas. Each scatter plot compares the probability accumulated by transformed and original WMI tasks for a class of formulas.

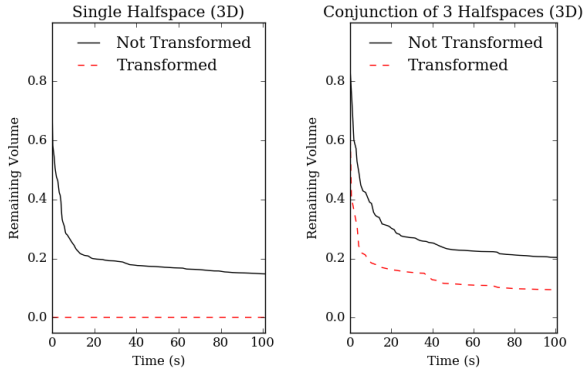


Figure 5: Plots of WMI's progress over representative benchmarks; the plots show remaining volume (less is better).

proach is not restricted to volume computation or piecewise-polynomial weights. Hyperrectangular decomposition can integrate arbitrary weight functions, although the orthogonal transformation technique described in this paper must be restricted to a formula's Gaussian variables.

Recent work introduced probabilistic inference modulo theories [Braz *et al.*, 2016]. The instantiations of their DPLL-like algorithm have only been over discrete bounded domains. In contrast, our work targets weighted model integration over the unbounded, dense domain of real arithmetic.

Approximate weighted model integration Weighted model counting is a #P-complete problem. Recently, there has been interest in providing (ϵ, δ)-guarantees in model counting using polynomially many calls to an NP oracle—a SAT or SMT solver. A number of works have studied the propositional counting setting, e.g., [Chakraborty *et al.*, 2014; 2013; Ermon *et al.*, 2014; 2013].

In the weighted model integration setting, recent work has shown how to use SMT solvers to approximate volumes of LRA formulas [Belle *et al.*, 2015b; Chistikov *et al.*, 2015]. In comparison to these works, our goal is to provide an exact (non-probabilistic) guarantee. Specifically, we provide exact upper and lower bounds on the desired probability by decomposing an LRA formula into hyperrectangles. The hyperrectangular decomposition idea is also exploited by Chistikov *et al.* in the approximate setting.

7 Conclusion

We addressed weighted model integration (WMI) algorithms that decompose a linear arithmetic formula into a set of disjoint hyperrectangular regions. We argued that the hyperrectangular decomposition approach benefits from applying transformations on formulas, and showed how to to construct orthogonal transformations over rationals, preserving the independence of Gaussian variables without introducing irrational coefficients. We implemented and evaluated our approach, demonstrating the substantial improvements available through orthogonal transformations. As a result, downstream applications such as repair and synthesis [Albarghouthi *et al.*, 2017b] are improved.

Acknowledgements We thank Jin-Yi Cai for suggesting exploring rotations to improve performance. This material is based upon work supported by the National Science Foundation under Grant numbers 1566015 and 1652140.

References

[Albarghouthi *et al.*, 2016] Aws Albarghouthi, Loris D'Antoni, Samuel Drews, and Aditya Nori. Fairness as a program property. *FATML*, November 2016.

[Albarghouthi *et al.*, 2017a] Aws Albarghouthi, Loris D'Antoni, Samuel Drews, , and Aditya Nori. Quantifying program bias. *arXiv:1702.05437*, 2017.

[Albarghouthi *et al.*, 2017b] Aws Albarghouthi, Loris D'Antoni, and Samuel Drews. Repairing decision-making programs under uncertainty. In *Computer Aided Verification*, 2017. Forthcoming.

[Barrett *et al.*, 2009] Clark W Barrett, Roberto Sebastiani, Sanjit A Seshia, and Cesare Tinelli. Satisfiability modulo theories. *Handbook of satisfiability*, 185:825–885, 2009.

[Belle *et al.*, 2015a] Vaishak Belle, Andrea Passerini, and Guy Van den Broeck. Probabilistic inference in hybrid domains by weighted model integration. In *IJCAI*, pages 2770–2776, 2015.

[Belle *et al.*, 2015b] Vaishak Belle, Guy Van den Broeck, and Andrea Passerini. Hashing-based approximate probabilistic inference in hybrid domains. In *UAI*, 2015.

[Belle *et al.*, 2016] Vaishak Belle, Guy Van den Broeck, and Andrea Passerini. Component caching in hybrid domains with piecewise polynomial densities. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 3369–3375, 2016.

[Bouissou *et al.*, 2016] Olivier Bouissou, Eric Goubault, Sylvie Putot, Aleksandar Chakarov, and Sriram Sankaranarayanan. Uncertainty propagation using probabilistic affine forms and concentration of measure inequalities. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 225–243. Springer, 2016.

[Braz *et al.*, 2016] Rodrigo de Salvo Braz, Ciaran O'Reilly, Vibhav Gogate, and Rina Dechter. Probabilistic inference modulo theories. In *IJCAI*, 2016.

[Chakraborty *et al.*, 2013] Supratik Chakraborty, Kuldeep S Meel, and Moshe Y Vardi. A scalable approximate model counter. In *International Conference on Principles and Practice of Constraint Programming*, pages 200–216. Springer, 2013.

[Chakraborty *et al.*, 2014] Supratik Chakraborty, Daniel Fremont, Kuldeep Meel, Sanjit Seshia, and Moshe Vardi. Distribution-aware sampling and weighted model counting for sat. In *AAAI*, 2014.

[Chatterjee *et al.*, 2016] Krishnendu Chatterjee, Hongfei Fu, Petr Novotný, and Rouzbeh Hasheminezhad. Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs. In *POPL*. ACM, 2016.

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

[Chistikov *et al.*, 2014] Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in SMT and value estimation for probabilistic programs. *CoRR*, abs/1411.0659, 2014.

[Chistikov *et al.*, 2015] Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in SMT and value estimation for probabilistic programs. In *TACAS*, 2015.

[Darmois, 1953] G. Darmois. Analyse général des liaisons stochastiques. *Review of the International Statistical Institute*, 21(1):2–8, 1953.

[De Loera *et al.*, 2012] JA De Loera, Brandon Dutra, Matthias Koeppe, Stanislav Moreinis, Gregory Pinto, and Jianqiu Wu. Software for exact integration of polynomials over polyhedra. *ACM Communications in Computer Algebra*, 45(3/4):169–172, 2012.

[De Moura and Bjørner, 2008] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[Ermon *et al.*, 2013] Stefano Ermon, Carla P Gomes, Ashish Sabharwal, and Bart Selman. Embed and project: Discrete sampling with universal hashing. In *Advances in Neural Information Processing Systems*, pages 2085–2093, 2013.

[Ermon *et al.*, 2014] Stefano Ermon, Carla P Gomes, Ashish Sabharwal, and Bart Selman. Low-density parity constraints for hashing-based discrete integration. In *ICML*, pages 271–279, 2014.

[Fierens *et al.*, 2015] Daan Fierens, Guy Van den Broeck, Joris Renkens, Dimitar Shterionov, Bernd Gutmann, Ingo Thon, Gerda Janssens, and Luc De Raedt. Inference and learning in probabilistic logic programs using weighted boolean formulas. *Theory and Practice of Logic Programming*, 15(03):358–401, 2015.

[Ma *et al.*, 2009] Feifei Ma, Sheng Liu, and Jian Zhang. Volume computation for boolean combination of linear arithmetic constraints. In *International Conference on Automated Deduction*, pages 453–468. Springer, 2009.

[Sampson *et al.*, 2014] Adrian Sampson, Pavel Panchekha, Todd Mytkowicz, Kathryn S McKinley, Dan Grossman, and Luis Ceze. Expressing and verifying probabilistic assertions. In *PLDI*, volume 49, pages 112–122. ACM, 2014.

[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 *PLDI*, pages 447–458, 2013.

[Shiu, 1983] P. Shiu. The shapes and sizes of pythagorean triples. *The Mathematical Gazette*, 67(439):33–38, March 1983.

[Skitovitch, 1953] V.P. Skitovitch. On one property of the normal distribution. *Doklady AN SSSR*, 89:217–219, 1953.