The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Polynomial-Time Probabilistic Reasoning with Partial Observations via Implicit Learning in Probability Logics

Brendan Juba[*]
Washington University in St. Louis
bjuba@wustl.edu

Abstract

Standard approaches to probabilistic reasoning require that one possesses an explicit model of the distribution in question. But, the empirical learning of models of probability distributions from partial observations is a problem for which efficient algorithms are generally not known. In this work we consider the use of bounded-degree fragments of the "sum-of-squares" logic as a probability logic. Prior work has shown that we can decide refutability for such fragments in polynomial-time. We propose to use such fragments to decide queries about whether a given probability distribution satisfies a given system of constraints and bounds on expected values. We show that in answering such queries, such constraints and bounds can be implicitly learned from partial observations in polynomial-time as well. It is known that this logic is capable of deriving many bounds that are useful in probabilistic analysis. We show here that it furthermore captures key polynomial-time fragments of resolution. Thus, these fragments are also quite expressive.

Introduction

Most scientific reasoning is probabilistic. It is quite rare for a conclusion to hold categorically. Informal conclusions such as "smoking causes cancer" correspond to formally probabilistic claims that the rate of incidence of cancer is higher in a population of smokers than another in which participants are forbidden from smoking. Likewise, our knowledge of the specific cases that comprise a study is almost necessarily incomplete. We are often interested in latent variables, such as whether or not a patient has a specific disease, that we seek to infer based on some observed attributes, e.g., the manifested symptoms. Unfortunately, if we do not already understand the processes that connect the observed attributes to the latent factors of interest, e.g., in the sense of possessing a probabilistic graphical model (Pearl 1988) of the system, the situation is quite challenging.

Indeed, the currently dominant approach to reasoning from data in such problems, as embodied for example by Markov Logic Networks (Richardson and Domingos 2006), is to first learn such a probabilistic graphical model, and then apply weighted model counting on translations of this model (Gogate and Domingos 2011; Domingos and Webb 2012). The problem lies in the first stage, in the "structure learning" problem. When the examples are not fully specified, existing approaches based on Expectation-Maximization ("E-M") do not scale well (Koller and Friedman 2009, Section 19.4). Although there are a variety of proposals for learning, for example, models with bounded tree-width (Narasimhan and Bilmes 2004; Sesh Kumar and Bach 2013), the networks encountered in practice often do not feature low tree-width (Chavira and Darwiche 2008). In a similar spirit, Poon and Domingos (2011) proposed sum-product networks as an alternative model for probability distributions. While sum-product networks provide some advantages over standard graphical models, ultimately just as with standard graphical models, structure learning has been accomplished by using the E-M meta-algorithm over the model likelihood. Thus, it falls prey to the same issues.

In this work we will consider techniques for reasoning under partial information that bypass the structure learning problem and indeed the whole framework of probabilistic graphical models. The structure learning problem under partial information remains out of reach, so what we achieve will necessarily be incomparable to what can be achieved given such a graphical model. Instead, we extend probability logics, e.g., as proposed by Nilsson (1986) or the generalization to expectation proposed by Halpern and Pucella (2007), in two ways: First, we observe that these logics that only consider linear inequalities can be strengthened to logics that consider polynomial inequalities on the support and moments. For this, we will observe that the "sum-of-squares" (a.k.a., "Positivstellensatz refutation") logic introduced by Grigoriev and Vorobjov (2001) can be interpreted as a probability logic, following work by Lasserre (2001) on deciding whether or not a probability distribution can be consistent with a given set of moments (aka, "moment problems"). We note that this logic has a rather powerful polynomial-time fragment: in addition to being able to derive and use a wide variety of standard analytic inequalities and represent conditional probability bounds as considered by Grosof (1986), *we also show that these fragments can simulate some of the strongest fragments of resolution known to be decidable in polynomial time*. And second, *we show how learning from partial examples can be integrated into answering queries in such a logic using implicit learning* (Juba 2013).

[*]Supported by NSF Award CCF-1718380.

Relationship to Other Work

The second extension will distinguish our approach both from Nilsson's logic, which does not consider how the input probabilistic bounds are obtained, and from approaches such as reasoning directly about the maximum entropy distribution that is consistent with the given constraints (Bacchus et al. 1996). Meanwhile, it is of course distinguished from the work on PAC-semantics and learning to reason (Khardon and Roth 1997; Valiant 2000; Juba 2013; Michael 2014) in that it can estimate general probabilities and expected values, and is not restricted to only drawing conclusions about what is true with high probability.

The qualitatively most similar relative to our approach is probabilistic logic programming (De Raedt and Kersting 2008), and in particular, Bayesian logic programming from interpretations (Kersting and De Raedt 2008). In this latter problem, we suppose that an unknown distribution is specified by some Bayesian logic program, and we would like to synthesize it from a collection of observations drawn from the distribution. In order for this to be feasible, it must be that the distribution has a nice description as a Bayesian logic program and so on. The distinction between our approach and Bayesian logic programming is roughly that in the setting we will consider, the distribution will not be restricted, but we will therefore necessarily give up hope of being able to generate a complete, compact description of the distribution (which in general may require more than 2^{2^n} bits to represent even if it is supported on n propositional attributes). Instead, we will only be able to infer some relationships between the moments and support of the distribution, depending on how much information the examples provide. Similarly, Khot et al. (2015) proposed to learn explicit models of the data distribution, and to use these models to perform inference in cases where the data has been masked at random. In addition to not relying on the ability to represent the distribution by such models, our approach does not assume that the data-hiding process is independent of the data distribution. But, our inferences will be relatively limited.

Our work relies on the close connection between fragments of the sum-of-squares logic and hierarchies of semidefinite program relaxations of (probability distributions supported on) systems of polynomial inequalities pioneered independently by Lasserre (2001), Parrilo (2000), Nesterov (2000), and Shor (1987). These works proposed the use of these semidefinite programming hierarchies as a way of relaxing polynomial optimization problems, which captures many discrete optimization problems, especially those using 0-1 integer optimization. These techniques have been quite successful at giving new algorithms for many machine learning problems, including many state-of-the-art algorithms (Barak, Kelner, and Steurer 2015; Rakhlin and Sridharan 2015; Hopkins, Shi, and Steurer 2015; Barak and Moitra 2016; Ma, Shi, and Steurer 2016; Arora et al. 2017). The difference is that these approaches take the view that we are seeking to find a specific representation by solving a semidefinite program relaxation in which the distributions are over candidate solutions to the program. Of particular interest in this vein is work by Hopkins and Steurer (2017), who propose this approach as a way of solving general kinds of maximum likelihood/maximum a posteriori (MAP) inference problems, and demonstrate it on a community detection problem. Hopkins and Steurer's work is most notable for making the connection between such an approach and Bayesian inference quite explicit. The main distinction here is that we are interested in deciding queries about the distribution under partial information, whereas Hopkins and Steurer are interested in obtaining explicit estimates of latent parameters. Meanwhile, Erdogdu et al. (2017) showed that such semidefinite programming techniques with a carefully engineered solver can similarly obtain high-quality MAP estimates in Markov random fields faster than (generalized) belief propagation in practice. We don't consider MAP inference here, but we also don't restrict ourselves to Markov random fields.

We stress, however, that Lasserre (2001) in particular viewed these programs as a means to decide "moment problems," i.e., whether or not there may exist a probability distribution consistent with a given set of moments. This is likewise how we use these programs. The main distinction between our work and Lasserre's is two-fold. First, Lasserre was primarily interested in establishing that the hierarchy of relaxations contain some tight relaxation, i.e., for each system, for some sufficiently high degree, the program can only have actual probability distributions as its feasible solutions. By contrast, like the subsequent work in the algorithms community, we are interested in the power of the sum-of-squares programs at fixed, finite levels of the hierarchy for probabilistic inference. Second, we are interested in how to learn additional constraints by incorporating empirical estimates of the moments from partial observations into the programs.

The Sum-of-Squares Probability Logic

For motivation, we will first briefly recall Nilsson's probability logic (Nilsson 1986), based on linear programming, or perhaps more accurately, the reconstruction of Nilsson's work by Fagin et al. (1990), and the generalization of this to reasoning about expected value by Halpern and Pucella (2007). We then recall the sum-of-squares formulation for a system of polynomial inequalities and the connection to these logics via the "pseudo-expectation" view introduced by Barak et al. (2011; 2014) (relaxing Lasserre (2001)).

Probability Logics Based on Linear Programming

In the propositional case of Nilsson's logic, we have a variable $p(\varphi)$ representing the likelihood of each propositional formula φ under consideration. Naturally, $p(\varphi) \in [0, 1]$. We have a number of constraints that we know hold in general among such variables,such as $p(\neg\varphi) + p(\varphi) = 1$. Or, more generally, $p(\psi \wedge \varphi) + p(\neg\psi \wedge \varphi) = p(\varphi)$, where $p(\top) = 1$ and for any equivalent formulas ψ and φ, $p(\psi) = p(\varphi)$. In Nilsson's logic, the basic relationships between these variables are given by a system of *linear inequalities*, of the form $\sum_\varphi a(\varphi)p(\varphi) \geq c$, where the coefficients $a(\varphi), c \in \mathbb{R}$. We can encode all of the above relationships as linear inequalities, as well as relationships such as $p(\psi) \leq p(\varphi)$ if $\psi \models \varphi$.

Now, given such a system of linear inequalities,

$$\sum_{\varphi} a^{(1)}(\varphi)p(\varphi) \geq c^{(1)} \cdots \sum_{\varphi} a^{(m)}(\varphi)p(\varphi) \geq c^{(m)}$$

for any $\alpha^{(1)}, \ldots, \alpha^{(m)} \in \mathbb{R}^+$, the linear combination

$$\sum_{i=1}^{m} \alpha^{(i)} \sum_{\varphi} a^{(i)}(\varphi)p(\varphi) \geq \sum_{i=1}^{m} \alpha^{(i)} c^{(i)}$$

is also true. Nilsson's logic also allows us to infer this.

Now, given that Nilsson's logic includes propositional reasoning in its axioms, it is naturally NP-hard and Fagin et al. (1990) show that when we restrict our attention to measurable probabilities, it is in fact NP-complete. We will consider a weaker starting point that is tractable. All of the initial constraints can be written as a system of linear inequalities; suppose that some subset of such inequalities is given. Now, given an additional linear inequality constraint, the question of whether or not it is consistent with our initial subset is merely linear feasibility. And, it follows from Farkas' lemma (Boyd and Vandenberghe 2004, p.263, e.g.) that the system is infeasible if and only if we can derive $0 \geq 1$ from the linear inequality inference rule. So, the linear combination inference rule gives refutation-completeness for this (admittedly weak) logic, and algorithms for linear programming give a method for deciding feasibility in polynomial time. By binary search on lower and upper bounds for the $p(\varphi)$, we can search for the tightest upper and lower bounds entailed by this system by adding the inequalities one at a time and testing for feasibility.

Indeed, essentially the same line of reasoning carries over to the generalization of Nilsson's logic to reasoning about expectations by Halpern and Pucella (2007), in which we replace the $p(\varphi)$ variables with $e(x_i)$ variables indicating the expected value of some random variable x_i. We remark that in Halpern and Pucella's logic, one can treat the original propositional variables φ as random variables, and recover Nilsson's logic. The main defect to the use of linear programming for inference is again that we drop most of the powerful propositional reasoning capabilities of the logic, reducing them to a handful of chosen inequalities.

Example 1. *We now give an example of these logics. Let's consider a medical domain where we wish to reason about the level of some protein X. We can represent the expression level of X with the positive real quantity $Xlevel$. Let's suppose X is said to be "elevated" when its expression level is greater than 10, and we have a Boolean indicator $highX$ for this event. Finally, suppose we know that the X level is elevated for at least 10% of the population. Now, can it be that the average expression level of X is less than 1? No: the average is at least* $10 \cdot .1 + 0 \cdot .9 = 1$. *A proof of this, in the style of Halpern and Pucella is to first use the* additivity *axiom to infer* $e(Xlevel) = e(Xlevel \cdot highX) + e(Xlevel \cdot (1 - highX))$. *We then use the* distributivity *axiom to infer* $e(Xlevel \cdot highX) \geq e(10highX)$ *(since* $Xlevel \cdot highX \geq 10highX$ *always) and* $e(Xlevel \cdot (1 - highX)) \geq e(0)$ *(since* $Xlevel \cdot (1 - highX) \geq 0$ *always). By the* homogeneity *axiom,* $e(10highX) \geq 10e(highX)$. *Similarly, by the* total probability *axiom,* $e(0) = 0$. *We are given* $e(highX) \geq .1$, *so by the* linear inequality *axioms, we can thus obtain* $e(Xlevel) \geq 10 \cdot .1 + 0 = 1$.

The Sum-of-Squares Semidefinite Programs

We will adopt the perspective of Halpern and Pucella, as our logic will be most naturally viewed as a logic of expected value. Let us now consider, for some fixed degree parameter $d \in \mathbb{N}$, a set of variables corresponding to the expected values of all monomials over our random variables $x_1, \ldots, x_n$ of total degree at most d, which we denote $e(x^{\vec{\alpha}})$, with the interpretation that $x^{\vec{\alpha}} = \prod_{i=1}^{n} x_i^{\alpha_i}$ where $\sum_{i=1}^{n} \alpha_i \leq d$ ($\vec{\alpha}$ is thus the vector of exponents of the monomial). We will enforce $e(1) = 1$ (for the "empty monomial" 1).

We will consider both discrete, propositional variables and bounded, continuous-valued variables. We will use the following standard set of constraints for propositional variables: for each propositional variable, we will include variables x_i and $\bar{x}_i$ encoding the variable x_i and its negation, related by the *complementarity axiom*, $x_i - \bar{x}_i + 1 = 0$. Each will also be constrained by the *Boolean axiom*, $x_i^2 - x_i = 0$ (and $\bar{x}_i^2 - \bar{x}_i = 0$). For other, continuous variables x_i, we will usually assume that there is an upper and lower bound on the range $B_i, L_i \in \mathbb{R}$, and we will include explicit bounding inequalities: for each monomial $x^{\vec{\alpha}}$ of total degree up to d, we give an upper and lower bound, $B_{\vec{\alpha}}$ and $L_{\vec{\alpha}}$, on the monomial as follows: we compute the largest and smallest magnitude positive and negative values consistent with the bounds on the range of each variable in the monomial. These may be computed in linear time by a simple dynamic programming algorithm that iteratively considers only the bounds for the product of the first k attributes. We finally take the largest of these values for $B_{\vec{\alpha}}$ and the smallest for $L_{\vec{\alpha}}$. We then include $B_{\vec{\alpha}} - x^{\vec{\alpha}} \geq 0$, and $x^{\vec{\alpha}} - L_{\vec{\alpha}} \geq 0$ in our constraints. Any additional constraints on the distribution's support in the background knowledge and query system will be added to the system of defining polynomial constraints.

Now, consider the matrix in which rows and columns are indexed by the monomials of degree up to $d/2$ in some standard way, and the $(\vec{\alpha}, \vec{\beta})$ entry of this matrix is the variable $e(x^{\vec{\alpha}+\vec{\beta}})$. That is, for the vector $\vec{v}$ of variables indexed in the same order, the matrix is $\mathbb{E}[\vec{v}\vec{v}^{\top}]$, which is certainly positive semidefinite. We refer to this as a *moment matrix*. Using a semidefinite program, we can strengthen our original linear program formulation by adding the constraint that this moment matrix must be positive semidefinite.

We can interpret this as follows. We note that for any real vector $\vec{p}$ again indexed by the monomials, $\vec{p}^{\top}\vec{v}$ gives the expected value of the polynomial with coefficients given by $\vec{p}$: $\sum_{\vec{\alpha}} p_{\vec{\alpha}} e(x^{\vec{\alpha}}) = \mathbb{E}[\sum_{\vec{\alpha}} p_{\vec{\alpha}} x^{\vec{\alpha}}]$. So, the positive semidefiniteness of this moment matrix corresponds to requiring that in any feasible solution, the square of any polynomial has a nonnegative expected value.

Another family of constraints that we will use is the following. Suppose that we know the polynomial constraint $g(\vec{x}) \geq 0$ holds for all $\vec{x}$ in the support of the distribution. Then surely, $\mathbb{E}[p(\vec{x})^2 g(\vec{x})] \geq 0$ for any polynomial $p(\vec{x})$. We can capture such a constraint as follows: again, we consider a matrix in which the rows and columns are indexed by the monomials $\vec{\alpha}$ up to degree $(d-d')/2$ where $g(\vec{x})$ has total degree d', such that in position $(\vec{\alpha},\vec{\beta})$ we have $\sum_{\vec{\gamma}} g_{\vec{\gamma}} e(x^{\vec{\alpha}+\vec{\beta}+\vec{\gamma}})$. If we then assert that this *"localizing matrix"* for g is positive semidefinite, it indeed imposes the desired constraint on the possible values for the $e(x^{\vec{\alpha}})$.

Finally, similarly, if $h(\vec{x})=0$ holds for all $\vec{x}$ in the distribution's support, then $\mathbb{E}[p(\vec{x})h(\vec{x})]=0$ for all polynomials p. We can add a linear constraint $\sum_{\vec{\gamma}} h_{\gamma} x^{\vec{\alpha}+\vec{\gamma}}=0$ for all $x^{\vec{\alpha}}$ of degree at most $d-d'$ when h has total degree d'.

The resulting semidefinite program given by this system of moment matrix and localizing matrix constraints is referred to as the *degree-d sum-of-squares relaxation* of the system of polynomial inequalities (Shor 1987; Nesterov 2000; Parrilo 2000; Lasserre 2001). We can test the feasibility of this system in polynomial time using semidefinite program solvers. As we will recall next, analogous to our earlier consequence of Farkas' Lemma, the feasibility of the resulting semidefinite programs is captured by a simple algebraic logic under some general conditions—in particular given we included explicit bounds on the range of the random variables among the defining polynomial inequalities.

Sum-of-Squares Refutations for Probabilities

For polynomials $g_1,\dots,g_r$ and $h_1,\dots,h_s$ in $\mathbb{R}[x_1,\dots,x_n]$, consider the set of $\vec{x}\in\mathbb{R}^n$ satisfying $g_j(\vec{x})\geq 0$ (for $j=1,\dots,r$) and $h_k(\vec{x})=0$ (for $k=1,\dots,s$). A *sum-of-squares* is, as the name suggests, a polynomial $\sigma(\vec{x})=\sum_{\ell=1}^{t} q_\ell(\vec{x})^2$ for polynomials $q_1,\dots,q_t\in\mathbb{R}[x_1,\dots,x_n]$. It is easy to see that a sum-of-squares σ must be non-negative for every $\vec{x}\in\mathbb{R}^n$. Thus, if we can find sums-of-squares $\sigma_0,\sigma_1,\dots,\sigma_r$ and polynomials $u_1,\dots,u_s$ such that

$$\sigma_0(\vec{x})+\sum_{j=1}^{r}\sigma_j(\vec{x})g_j(\vec{x})+\sum_{k=1}^{s}u_k(\vec{x})h_k(\vec{x})=-1 \qquad (1)$$

the set defined by the inequalities $g_1(\vec{x})\geq 0,\dots,g_r(\vec{x})\geq 0$ and $h_1(\vec{x})=0,\dots,h_s(\vec{x})=0$ must be empty (or else we would reach a contradiction). The *sum-of-squares* proof system of Grigoriev and Vorobjov (2001) is to show that such systems are unsatisfiable by finding such polynomials:

Definition 2 (Sum-of-squares refutation: syntax)**.** *A* sum-of-squares refutation *of a system of equalities $h_1(\vec{x})=0,\dots,h_s(\vec{x})=0$ and inequalities $g_1(\vec{x})\geq 0,\dots,g_r(\vec{x})\geq 0$ consists of $\sigma_0,\dots,\sigma_r$ and $u_1,\dots,u_s$ satisfying Equation 1. The* degree *of the sum-of-squares refutation is degree of the resulting formal expression, while the* size *is the number of monomials in $\sigma_0,\dots,\sigma_r$ and $u_1,\dots,u_s$. The system is* explicitly compact *if the variables are either all explicitly Boolean (the constraint $x_i^2-x_i=0$ is included) or bounded ($x_i^2\leq B_i$ is included for some $B_i\geq 0$).*

This is a formal language of polynomial expressions. The semantics of these expressions are given by assigning an expectation operator: we map each monomial to an expected value, and extend these to polynomial expressions by linearity. Now, noting that the program capturing the existence of a sum-of-squares refutation is the dual to the sum-of-squares relaxation, one obtains the following theorem:

Theorem 3 (Soundness (Shor 1987; Nesterov 2000; Parrilo 2000; Lasserre 2001))**.** *Let $g_1(\vec{x})\geq 0,\dots,g_r(\vec{x})\geq 0$, $h_1(\vec{x})=0,\dots,h_s(\vec{x})=0$ be a system of constraints that is explicitly compact. Then either there is a degree-d sum-of-squares refutation or there is a solution to the degree-d sum-of-squares relaxation.*

Note that this means that if there is a sum-of-squares refutation, then there cannot be any probability distributions consistent with the given support constraints, or else these would give a feasible solution to the semidefinite program we developed in the previous section. Thus, for any degree, sum-of-squares is a *sound*, but possibly incomplete refutation system for expectation operators, and hence for the associated probability distributions.

We can furthermore add prior knowledge of constraints on the moments of the distribution to the semidefinite program, and read off an extended kind of sum-of-squares proof from the dual: If we add the constraint $e(x^{\vec{\alpha}})\leq\gamma$ to the semidefinite program, this corresponds to adding nonnegative multiples of the polynomial $(\gamma-x^{\vec{\alpha}})$. A sum-of-squares derivation of -1 yields a contradiction now by considering the expected value of the derived expression for a distribution additionally satisfying the moment constraints: the expression is equal to -1, but $\mathbb{E}[\gamma-x^{\alpha}]\geq 0$ by definition and the rest of the expression is again nonnegative on the distribution's support. Indeed, for any polynomial $p(\vec{x})$, we can incorporate knowledge that $\mathbb{E}[p(\vec{x})]\leq\gamma$ by adding nonnegative multiples of $(\gamma-p(\vec{x}))$ in the dual.

Example 4. *Let's return to our medical example and see how sum-of-squares can capture the same style of reasoning and infer the same bound on $\mathbb{E}[Xlevel]$. We first express that $highX$ is Boolean with the constraint $highX^2-highX=0$ and that $Xlevel$ is nonnegative with $Xlevel\geq 0$. Next, we can encode the relationship between $Xlevel$ and $highX$ with the following pair of inequalities: $highX(Xlevel-10)\geq 0$ and $(1-highX)(10-Xlevel)\geq 0$. Then when $highX=1$, $Xlevel\geq 10$ and when $highX=0$, $10\geq Xlevel$, as desired. (In general, multiplying by a Boolean variable has this effect of making a constraint conditional.) Finally, we include the bound $\mathbb{E}[highX]\geq .1$. Now, we can use these constraints to infer the expression $Xlevel-1$ is nonnegative as follows. In general, one builds up these expressions by a series of substitutions. Starting from the total probability expression from the Halpern-Pucella style proof, we first substitute the conditional bounds for the expression $e(Xlevel\cdot highX)+e(Xlevel\cdot(1-highX))$, i.e., $(Xlevel-10)highX+Xlevel(1-highX)$. The first part is given, but we can obtain the second by multiplying $Xlevel\geq 0$ by a sum-of-squares expression, and using the fact that $highX$ is Boolean to rewrite the result like so: $Xlevel(1-highX)=(1-highX)^2Xlevel+(-Xlevel)(highX^2-highX)$. Now, to use our bound on $\Pr[highX]\geq .1$, we add $10(highX-.1)$ to the expression. Overall, we thus obtain the following sum-of-squares expression, which simplifies to $Xlevel-1$:*

$$10(highX-.1)+1\cdot highX(Xlevel-10)+$$
$$(-Xlevel)(highX^2-highX)+(1-highX)^2Xlevel.$$

Since this is an expression of degree 3, Theorem 3 says that the degree-3 sum-of-squares program will detect infeasibility with the constraint $\mathbb{E}[Xlevel] \leq 1 - \delta$ for any $\delta > 0$.

Putinar's Positivstellensatz (Putinar 1993) asserts that such proofs exist in general, i.e., the system is *complete* if one considers expressions of sufficiently large degree.

Theorem 5 (Completeness, a corollary of Putinar 1993)**.** *There exists a probability distribution with expected values $\{e(x^{\vec{\alpha}})\}_{\vec{\alpha}\in\mathbb{N}^n}$ supported on a set given by an explicitly compact system $g_1(\vec{x}) \geq 0, \ldots, g_r(\vec{x}) \geq 0$, $h_1(\vec{x}) = 0, \ldots, h_s(\vec{x}) = 0$ iff every moment matrix is positive semidefinite, every localizing matrix for each g_j is postive semidefininte, and every localizing matrix for each h_k is zero (i.e., solutions exist for all degrees d).*

Polynomial-time solvability. We can decide the existence of such refutations in polynomial time (in the number of attributes and size of the coefficients in the proofs) by finding solutions to the corresponding "dual" semidefinite program. The sum-of-squares refutation that we are supposing to exist is itself simply a way of interpreting such solutions to this "dual" program (Lasserre (2010, Section 4.2) gives a nice review), and indeed we observe that it suffices to find such an expression that attains a negative value. So, it suffices to show that we can find a dual solution with negative value, given that there exists a dual solution in which the coefficients have polynomial size and each primal variable has explicit (polynomial-magnitude) bounds on its range.

Technically, it is only known how to solve semidefinite programs when a solution is known to lie in a ball of exponential radius, up to a polynomial number of bits of numerical precision using algorithms such as the ellipsoid method (Grötschel, Lovász, and Schrijver 2012), and in general this may cause difficulty for finding sum-of-squares refutations (O'Donnell 2017; Raghavendra and Weitz 2017). Nevertheless, our assumptions on the bounds on the size of the coefficients and ranges of the attributes ensure that, indeed a solution to polynomial accuracy suffices. Note that the assumed bound on the coefficients expressly rules out the problems observed by O'Donnell for the applications in optimization, so the only issue is the additive error suffered by the ellipsoid solutions. The degree-d program has at most n^d coefficient variables, and if we know that each monomial is explicitly bounded to have magnitude at most S, then so long as each coordinate is estimated to within accuracy $1/2n^dS$, then substituting our bounds for the remaining variables (using these explicit bounds) suffices to give us a dual expression that has value at most $-1/2$, which is still adequate for a refutation.

Indeed, our bound on the proof's coefficients ensures that the positive semidefinite matrix variable of the dual program has a polynomially bounded trace. This matrix variable is obtained from $\sum_i \vec{p}_i\vec{p}_i^\top$ if $\sum_{\vec{\alpha}} p_{i,\vec{\alpha}}x^{\vec{\alpha}}$ ($i = 1, 2, \ldots$) are the polynomials used in the proof, which has trace $\sum_i\sum_{\vec{\alpha}} p_{i,\vec{\alpha}}^2$. Together with the bound on the input coefficients, the range of values each expression in the dual program may take is thus constrained, and hence its *width* is polynomially bounded. Thus, as we tolerate a polynomial additive error, if we only use inequality constraints then even the fast method of Arora et al. (2005) suffices.

The Expressive Strength of Sum-of-Squares

We now briefly review the surprising strength of these proofs, and establish some new results on their Boolean reasoning ability. As with the logics of Nilsson (1986) and Halpern and Pucella (2007), by searching over feasible inequality constraints, we can use sum-of-squares queries to compute upper and lower bounds on probabilities (of Boolean expressions) and more generally expectations of polynomials of real-valued variables, thus bounding the variance and so on. Along the lines of Grosof's extension to Nilsson's logic (Grosof 1986), we can express bounds (and thus also queries) on conditional probabilities $\Pr[x_a|x_b] \leq p$ by rewriting: as $\Pr[x_a \wedge x_b] = \mathbb{E}[x_ax_b]$, it is equivalent to $\mathbb{E}[x_ax_b] \leq \mathbb{E}[px_b]$, or $\mathbb{E}[x_ax_b - px_b] \leq 0$. The key question is how tight these bounds are.

Much of the excitement about sum-of-squares in algorithms stems from the power of even low-degree (e.g., degree-4) sum-of-squares proofs to capture a surprisingly strong fragment of probabilistic reasoning. In particular, Barak et al. (2012) show that such sum-of-squares proofs exist for key analytic inequalities such as the Cauchy-Schwarz and Hölder inequalities and hypercontractivity bounds. O'Donnell and Zhou (2013) and Kauers et al. (2014) further established that low-degree sum-of-squares can derive reverse hypercontractivity bounds, and thus also the KKL Theorem (Kahn, Kalai, and Linial 1988) and an invariance principle (generalizing the central limit theorem) (Mossel, O'Donnell, and Oleszkiewicz 2010). These theorems were used previously to analyze instances of optimization problems that were hard for existing algorithms. These works then observed that the sum-of-squares relaxation can actually then derive tight bounds that solve these very same instances since sum-of-squares refutations capture all of these constraints. We thus observe that sum-of-squares is quite powerful for probabilistic analysis.

We now turn to consider the Boolean reasoning ability of sum-of-squares. Note that the logics of both Nilsson and Halpern-Pucella included all propositional tautologies in their axioms, and this Boolean reasoning is an important part of the logics. Since sum-of-squares does not directly include a set of axioms to handle Boolean reasoning, we need to demonstrate that it is capable of a significant set of inferences, ideally simulating known tractable fragments of other Boolean logics. Indeed, we now observe that sum-of-squares simulates a variety of fragments of systems for which polynomial-time algorithms were already known.

Notably, Berkholz (2018) has shown that degree-$2d$ sum-of-squares simulates the degree-d fragment of Polynomial Calculus when we include the Boolean axioms $x_i^2 - x_i = 0$ for all variables x_i. Polynomial calculus is a system that allows us to derive new polynomial equations by taking linear combinations of previous lines, or by multiplying a previous line by an indeterminate (variable). In particular, if we include a variable for each literal and the complementarity axiom $x_i + \bar{x}_i - 1 = 0$, the degree-d fragment easily simulates *width-d* resolution (Alekhnovich et al. 2002), where recall the width is the maximum number of literals in any clause. This is accomplished by representing the clauses as a system of single monomial constraints of degree at most d: for the clause $\ell_1 \vee \cdots \vee \ell_k$, we have $\bar{\ell}_1 \cdots \bar{\ell}_k = 0$ is an equivalent constraint of degree at most d.

We further show that sum-of-squares simulates space-bounded treelike resolution without the bounded-width requirement. Although the encoding used in the above works cannot express wide clausal constraints on the support, we can instead use a constraint $\sum_i \ell_i \geq 1$ on the support to encode the clause $\bigvee_i \ell_i$ (with each ℓ_i also constrained by $\ell_i^2 - \ell_i = 0$). We will use the following characterization of space-bounded treelike resolution given by Ansótegui et al. (2008): For a *partial assignment* to propositional variables $x_1, \ldots, x_n$, $\rho \in \{0, 1, *\}^n$ (where $*$ means "unassigned") the partial evaluation or restriction of a formula φ by ρ, denoted $\varphi|_\rho$, is obtained by plugging in ρ_i for x_i when $\rho_i \neq *$, and simplifying the Boolean connectives in the natural way: we delete true inputs to ANDs, false inputs to ORs, and replace ANDs and ORs respectively with false and true if they have, respectively, a false or true input. (Negations are simply evaluated if their input is.) Recall that *unit propagation* on a set of clauses $\mathcal{C}$ and a partial assignment ρ is the inference rule that, if some clause $C \in \mathcal{C}$ simplifies to a single literal ℓ under partial evaluation by ρ then we can infer the setting that satisfies ℓ and add it to ρ. If we ever obtain an empty clause, $C|_\rho = \bot$, then this is a *unit propagation refutation*. Ansótegui et al. (2008) essentially observe that unit propagation captures "clause space 1" treelike resolution refutations, and can be generalized to space s by allowing $s-1$ levels of the following *failed literal* rule: inductively, supposing we have defined the system with $s-1$ levels (where 0 levels is unit propagation), s levels of the failed literal rule is as follows: if, when we add an assignment $x_i = b$ to our partial assignment ρ, we can obtain a level-$s-1$ refutation, then we can infer $x_i = 1-b$, and add this to ρ instead. A level s refutation then occurs if we can eventually obtain a level-$s-1$ refutation without asserting any additional literals (thus, eventually, a unit propagation refutation). Ansótegui et al. found that instances of resolution that are easy in practice are refutable with relatively small s. Note that unit propagation alone ($s = 1$) already simulates chaining, e.g. in Horn KBs.

Theorem 6. *The degree-$s+1$ sum-of-squares relaxation of the linear inequality encoding of a CNF is infeasible if there exists a level-s refutation.*

Proof. By induction on s: First, for level-0, we argue that degree-1 sum-of-squares detects unit propagation refutations among the input system. We argue this by induction on the number of steps of unit propagation inference. In 0 steps, if there is an empty clause in the input, we have the constraint $-1 \geq 0$ in the input system, which is a trivial contradiction. Now, after i steps, suppose we have $\ell_i \leq 0$ (or $\ell_i = 0$) for $i \neq i^*$ in some clause. Then we can derive $\ell_{i^*} \geq 1$ in degree 1 (and hence, $\bar{\ell}_{i^*} \leq 0$) from $\sum_i \ell_i - 1 + \sum_{i \neq i^*} -\ell_i = \ell_{i^*} - 1$. Observe that these linear combinations do not increase the degree. Thus, finally, if we derived $\ell_i \leq 0$ for all literals in some clause, we can similarly derive $-1 \geq 0$ to obtain our contradiction in degree-1.

Now, given that the program detects level-$s-1$ refutations in degree s, we argue that it detects level s refutations in degree $s+1$: indeed, we will argue that if $\bar{\ell}$ can be inferred by an application of the failed literal rule and if there is a feasible solution to the degree $s+1$ sum-of-squares relaxation, that for all monomials x^α, $e(\ell x^{\vec{\alpha}}) = 0$. Indeed, suppose not. Consider the Cholesky factorization of the (positive semidefinite) moment matrix into $UU^\top$. We know that we must have that for the row indexed by ℓ, $\vec{u}_\ell \neq 0$, so $e(\ell) = e(\ell^2) > 0$. It follows then that we can perform the following "*conditioning operation*" (from Karlin et al. 2011): for the minor of the moment and localizing matrices consisting of all index monomials with a factor of ℓ, we know that this is a positive semidefinite minor. If we rescale each by $1/e(\ell^2)$, then indeed it further satisfies the constraint that the entry we obtained from the (ℓ, ℓ) position takes value 1. We thus see that this is a solution to the sum-of-squares program of degree s in which moreover it may be shown that by the localizing constraints for the Boolean axiom on ℓ, for any monomial $x^{\vec{\alpha}}$, $e(\ell x^{\vec{\alpha}}) = e(x^{\vec{\alpha}})$. More generally by the localizing constraints on the complementarity axiom, $e(\bar{\ell}) = 0$ in this solution, as is any monomial containing $\bar{\ell}$. Thus, we obtain a solution in which $\ell = 1$ and $\bar{\ell} = 0$. But since we are supposing that we can infer $\bar{\ell}$ by an application of the level-s failed literal rule, this means that there is a level-$s-1$ refutation of this system. By IH, therefore, this system must be infeasible, a contradiction. It must be then that indeed $e(\ell x^{\vec{\alpha}}) = 0$ for all monomials $x^{\vec{\alpha}}$ as claimed and therefore also $e(\bar{\ell} x^{\vec{\alpha}}) = e(x^{\vec{\alpha}})$; in particular, then, $e(\bar{\ell}) = e(1) = 1$ in any feasible solution. Finally if some input clause simplifies to the empty clause, we again obtain that $-1 \cdot e(1) \geq 0$ in contradiction to our constraint that $e(1) = 1$. Thus, if there is a level-s refutation, the degree $s+1$ sum-of-squares relaxation must be infeasible. □

Weaknesses. So we see how we can use sum-of-squares to compute estimates on probabilities and bounds on expected values, using the complexity of deriving the bound rather than the form of the distribution as our "bias." Compared to, e.g., graphical models, this approach circumvents some fundamental difficulties, but has some other inherent weaknesses. The main weakness is that we cannot express prior knowledge about the independence of random variables in our distribution. Indeed, this would lead to quadratic (or worse) constraints in the "primal" optimization problem, preventing us from solving it in polynomial time. Only if some probabilities are given explicitly can we assert independence with respect to those events, e.g., $\mathbb{E}[x_a] = p$ and $\mathbb{E}[x_a x_b] = p\mathbb{E}[x_b]$ for $p \in [0, 1]$. A second weakness is that in our tractable fragments, we can only infer values for marginal distributions that refer to at most $O(1)$ variables at a time. (That is, we must "sum out" all but $O(1)$ variables.) We may have constraints on more variables as long as these are expressed by a low-degree polynomial, such as our linear inequality encodings of clauses, but this only captures a limited family of the possible constraints.

Implicit Learning From Partial Examples

We now suppose that we have access to *partial examples* drawn from the distribution D we are seeking to reason about. We would like to use these examples to empirically learn additional constraints that the distribution satisfies. We will develop a notion of implicit learning of constraints for use in answering queries in our probability logic, analogous to Juba (2013) for Boolean logics.

Masking and Testability

We will use a model of learning under partial information adapted from prior works (Michael 2010; Rubin 1976). In such models, the distribution over partial examples is produced as follows. First, a "ground truth" example $\vec{x}$ is drawn from the distribution D. Then there is a second random process M that independently selects a function m that takes complete examples $\vec{x}$ and produces a *partial example* $\vec{\rho}$ by replacing any number of coordinates of $\vec{x}$ with the value $*$ meaning "unknown" or "missing." We denote the distribution on partial examples by $M(D)$. Note that the choice of which coordinates to hide may depend on all of the actual values of $\vec{x}$. M, like D, is arbitrary in general. Indeed, $m(\vec{x})$ may even hide the entire example. Thus, we can only hope to learn constraints from the partial examples $\vec{\rho}$ it produces that are revealed by M to be obeyed with high probability.

A family of such constraints is as follows. Again, we suppose that each ith attribute is known to have some upper and lower bound B_i and L_i, and that the corresponding constraints are included in the program. For a given partial example $\vec{\rho}$, there also exists an upper and lower bound, $B_{\vec{\alpha}}$ and $L_{\vec{\alpha}}$, on the monomial $x^{\vec{\alpha}}$ given $\vec{\rho}$ as follows: first, for each $x_i^{\alpha_i}$, if $\rho_i \neq *$, then we substitute $\rho_i^{\alpha_i}$ for ρ_i. Then we compute the largest and smallest magnitude positive and negative values consistent with the bounds on the range of each variable (if any) in the monomial as before.

Definition 7. *We will say that a polynomial inequality* $\sum_{\vec{\alpha}} p_{\vec{\alpha}} x^{\vec{\alpha}} \geq 0$ *is* witnessed *under $\vec{\rho}$ if, when we plug in the upper bound for each $x^{\vec{\alpha}}$ under $\vec{\rho}$ for every $p_{\vec{\alpha}} < 0$ and the lower bound when $p_{\vec{\alpha}} > 0$, the inequality is still satisfied. We will say that a system of polynomial constraints $p_1(\vec{x}) \geq 0, \ldots, p_\ell(\vec{x}) \geq 0$ is* $(1-\gamma)$-testable *with respect to M and D if with probability at least $(1-\gamma)$ over $\vec{\rho}$ drawn from M and D, every $p_i(\vec{x}) \geq 0$ in the system is witnessed under $\vec{\rho}$.*

We will show that knowledge encoded by all such testable systems of constraints can be efficiently learned implicitly, even though there may be many such constraints, as long as γ is small enough. In support of the empirical estimation of such expressions, we will need the following two definitions.

Definition 8. *The* upper bound *of a polynomial expression* $\sum_{\vec{\alpha}} p_{\vec{\alpha}} x^{\vec{\alpha}}$ *is given by substituting the upper bound on $x^{\vec{\alpha}}$ (for the empty partial assignment) for each monomial $x^{\vec{\alpha}}$ if $p_{\vec{\alpha}} > 0$, and substituting the lower bound on $x^{\vec{\alpha}}$ for $p_{\vec{\alpha}} < 0$. The* lower bound *is similarly given by substituting lower bounds on $x^{\vec{\alpha}}$ for $p_{\vec{\alpha}} > 0$ and upper bounds on $x^{\vec{\alpha}}$ for $p_{\vec{\alpha}} < 0$. The* naïve norm *of the polynomial expression is then given by the maximum of its upper bound and the absolute value of its lower bound.*

Definition 9. *For a polynomial $p(\vec{x})$, we will let $p|_{\vec{\rho}}(\vec{x})$ denote the polynomial with ρ_i plugged in for x_i whenever $\rho_i \neq *$, and collecting terms with the same monomial. We refer to this as the* partial evaluation *of p under $\vec{\rho}$.*

Note that both can be computed in linear time in the size of the expression, given our bounds on the individual monomials and the partial assignment, respectively.

Example 10. *As an example, $p(x,y,z) = 3x^2yz - xy^2 + 2z - 1$ partially evaluated at $\rho = (10, *, 2)$ is $p|_\rho(y) = -10y^2 + 600y + 3$. If $-1 \leq y \leq 1$, the lower bound is -607, so $p(x,y,z) \geq 0$ is* not *witnessed under ρ in this case, but if $0 \leq y \leq .1$, then the lower bound is 2.9, so then $p(x,y,z) \geq 0$ would be witnessed under ρ. Note that the substitutions of bounds are done monomial-by-monomial, as otherwise deciding validity of the polynomial inequalities would be intractable in general. Similarly, the naïve norms of $p|_\rho(y)$ are, respectively, 607 and 63.*

Deciding Queries With Implicit Learning

Given a degree bound d, a collection of partial examples $\vec{\rho}^{(1)}, \ldots, \vec{\rho}^{(m)}$, an input system of support constraints, $g_j(\vec{x}) \geq 0$ (for $j = 1, \ldots, r$, that we assume includes the upper and lower bound constraints for each attribute) and $h_k(\vec{x}) = 0$ (for $k = 1, \ldots, s$), and an input system of moment constraints, $p_\ell(\vec{x}) \geq 0$ (for $\ell = 1, \ldots, t$), we write the following semidefinite program to decide if our input bounds and constraints are consistent with the distribution from which the examples were drawn. For each $i = 1, \ldots, m$, we create a set of variables $\vec{x}^{(i)}$ for those variables unfixed in $\vec{\rho}^{(i)}$. We add a degree-d moment matrix constraint for $\vec{x}^{(i)}$, a degree-d localizing matrix constraint for each $g_j|_{\vec{\rho}^{(i)}}(\vec{x}^{(i)})$, and a degree-$d$ localizing matrix constraint for each $h_k|_{\vec{\rho}^{(i)}}(\vec{x}^{(i)})$.[1] Finally, we write for each monomial $x^{\vec{\alpha}}$ of degree up to d, the constraints

$$\frac{1}{m}\sum_{i=1}^{m}(x^{(i)\vec{\alpha}})|_{\vec{\rho}^{(i)}} - \frac{(B_{\vec{\alpha}} - L_{\vec{\alpha}})\sqrt{\ln(\binom{2n}{\leq d}/\delta)}}{\sqrt{2m}} \leq x^{\vec{\alpha}}$$

$$x^{\vec{\alpha}} \leq \frac{1}{m}\sum_{i=1}^{m}(x^{(i)\vec{\alpha}})|_{\vec{\rho}^{(i)}} + \frac{(B_{\vec{\alpha}} - L_{\vec{\alpha}})\sqrt{\ln(\binom{2n}{\leq d}/\delta)}}{\sqrt{2m}}$$

and add the constraints $p_\ell(\vec{x}) \geq 0$. We then accept the system if and only if the semidefinite program is feasible.

We now state our guarantee for this approach. In the theorem statement, for simplicity we use a single common size parameter $S \in \mathbb{R}$.

Theorem 11. *For $m = \Omega(S^2(d \log n + \log \frac{1}{\delta}))$ partial examples drawn from $M(D)$ where for all $\vec{\alpha}$ of degree at most d, $B_{\vec{\alpha}} - L_{\vec{\alpha}} \leq S$, with probability $1 - \delta$:*

- *if D satisfies the input system of bounds and constraints the semidefinite program is feasible*

[1]We implicitly add the constraint that for $i \neq i'$, any monomial containing variables from the variables for examples i and i' gets value 0. But since the moment matrix then has a block-diagonal structure, it is equivalent to simply impose the constraint that each ith block has a positive semidefinite moment matrix.

- *if there is a system of constraints that is $(1-\frac{1}{2S})$-testable under M and D and, together with the input system, completes a sum-of-squares refutation with naïve norm at most S, then the semidefinite program is infeasible.*

Note that for a Boolean system, the naïve norm closely corresponds to the size of the proof (in number of monomials).

Overview. We first describe the proof at a high level for intuition. For the first case, we argue that since the range of the values is bounded by S, the empirical constraints are valid with probability $1-\delta$ overall using Hoeffding's inequality:

Theorem 12 (Hoeffding's inequality)**.** *Let $X_1,\ldots,X_m$ be i.i.d. $[0,1]$-valued random variables. Let $\bar{X}=\frac{1}{m}\sum_{i=1}^m X_i$. Then for $\gamma>0$, $\Pr[\bar{X}-\mathbb{E}[X_i]>\gamma]\leq e^{-2m\gamma^2}$.*

Thus, since we have otherwise assumed D satisfies the given bounds and constraints, the program is feasible in this case. For the second case, we consider the hypothetical sum-of-squares refutation. We plug in each of the m partial examples, and average the resulting expressions; we note that this is still formally equal to -1. We will break the averaged expression into two parts: the part that can be written using the known bounds and constraints by using the empirical inequalities, and the parts using the unknown but testable constraints. If we can show that the combined slack in the empirical ineqalities and the total size of the unknown part can be bounded by $-(1-\Delta)$ for some constant $\Delta>0$, then the known part of the expression can be bounded by $-1+(1-\Delta)=-\Delta$, so scaling the expression by $1/\Delta$ will give -1 and hence a refutation. We can use the empirical moment expressions to derive the moment bounds, where m is sufficiently large that the total error in these expressions is bounded by a constant, less than $1/6$. So it only remains to show how we bound the unknown, testable part of the system by another constant, say $1/2$. For this last part, we note that we have assumed that these constraints are $(1-1/2S)$-testable. This means that the entire expression is bounded by 0 except in a $1/2S$-fraction of the examples, where the size of the remaining expression can be bounded by the naïve norm, which is at most S. Thus, the total size of these non-vanishing expressions can be bounded by $1/2$ as needed. We now give the full proof.

Proof. We first analyze the first case. Since the distribution is assumed to satisfy the support constraints, for any $\vec{\rho}=m(\vec{x})$ drawn from M and D, since $\vec{x}$ is in the support of D, the completion of $\vec{\rho}$ to $\vec{x}$ is an assignment that satisfies all of the support constraints. Moreover, since for each moment, $x^{\vec{\alpha}}$ has $B_{\vec{\alpha}}-L_{\vec{\alpha}}\leq S$, we see that $\frac{1}{S}(x^{\vec{\alpha}}-L_{\vec{\alpha}})$ is a random variable in the range $[0,1]$; therefore, by Hoeffding's inequality, for our choice of m the moment constraints hold with probability $1-\delta/\binom{2n}{\leq d}$ for each $\vec{\alpha}$. So, by a union bound over these moments, with probability $1-\delta$, $\frac{1}{m}\sum_{i=1}^m(x^{(i)})^{\vec{\alpha}}$ (using the ground truth examples) is feasible in these constraints, as therefore is $\frac{1}{m}\sum_{i=1}^m(x^{(i)\vec{\alpha}})|_{\vec{\rho}^{(i)}}$. So our empirical upper and lower bound constraints on $x^{\vec{\alpha}}$ are satisfied. We also know that $\mathbb{E}[x^{\vec{\alpha}}]$, by assumption, satisfies the given polynomial constraints on our moments. Thus the sum-of-squares relaxation is feasible and so our algorithm accepts with probability $1-\delta$ in this case as needed.

Now, for the second case. We assume that there is a set of $(1-1/2S)$-testable support constraints, $\tilde{g}_{\tilde{j}}(\vec{x})\geq 0$ for $\tilde{j}=1,\ldots,\tilde{r}$ and $\tilde{h}_{\tilde{k}}(\vec{x})=0$ for $\tilde{k}=1,\ldots,\tilde{s}$, and moment constraints, $\tilde{p}_{\tilde{\ell}}(\vec{x})\geq 0$ for $\tilde{\ell}=1,\ldots,\tilde{t}$, that complete a degree-d sum-of-squares refutation with naïve norm at most S. Suppose we substitute $x^{(i)\vec{\alpha}}|_{\vec{\rho}^{(i)}}$ for $x^{\vec{\alpha}}$ for each $\vec{\alpha}$ in this proof. Observe that it remains formally identical to -1, the degree does not increase, and the sum-of-squares expressions remain sum-of-squares expressions (on $\vec{x}^{(i)}$).

Now, suppose we average over these m refutations. Again, we obtain an expression formally identical to -1 of the same degree. The final sum-of-squares term $\sigma_0(\vec{x})$ becomes another sum-of-squares, $\frac{1}{m}\sum_{i=1}^m\sigma_0(\vec{x}^{(i)})|_{\vec{\rho}^{(i)}}$, of the same degree, so we can find an analogous expression in our program. For portions of the expression derived from the support constraints, for example the term $\sigma_j(\vec{x})g_j(\vec{x})$, we have $\sum_{i=1}^m(\frac{1}{m}\sigma_j(\vec{x}^{(i)})g_j(\vec{x}^{(i)}))|_{\vec{\rho}^{(i)}}$. So, we see that this can still be written using the support constraints for the individual examples included in our program. The equality support constraints can be similarly transformed.

Next, we consider the moment bounds, which we know can only be multiplied by positive constants. We find that the expressions $\lambda_\ell p_\ell(\vec{x})$, i.e., of the form $\lambda_\ell\sum_{\vec{\alpha}}p_{\vec{\alpha}}x^{\vec{\alpha}}$. We see again that under this substitution we obtain an expression that may be rewritten as $\sum_{i=1}^m\frac{\lambda_\ell}{m}\sum_{\vec{\alpha}}p_{\vec{\alpha}}x^{(i)\vec{\alpha}}|_{\vec{\rho}^{(i)}}$. Now, our program provides us with the moment constraint $p_\ell(\vec{x})$ and the upper and lower empirical bounds on each moment, $\frac{1}{m}\sum_{i=1}^m x^{(i)\vec{\alpha}}|_{\vec{\rho}^{(i)}}+\frac{B_{\vec{\alpha}}\sqrt{\ln(\binom{2n}{\leq d}/\delta)}}{\sqrt{2m}}-x^{\vec{\alpha}}\geq 0$ and $\frac{L_{\vec{\alpha}}\sqrt{\ln(\binom{2n}{\leq d}/\delta)}}{\sqrt{2m}}-\frac{1}{m}\sum_{i=1}^m x^{(i)\vec{\alpha}}|_{\vec{\rho}^{(i)}}+x^{\vec{\alpha}}\geq 0$. By choosing the appropriate bound (the first if $p_{\vec{\alpha}}>0$ and the second if $p_{\vec{\alpha}}<0$), multiplying by $|p_{\vec{\alpha}}|>0$ in each case, and summing these up, we can obtain an expression of the form

$$\frac{1}{m}\sum_{i=1}^m p_\ell(\vec{x}^{(i)})|_{\vec{\rho}^{(i)}}-p_\ell(\vec{x})+\frac{\sum_{\vec{\alpha}}|p_{\ell,\vec{\alpha}}C_{\vec{\alpha}}|\sqrt{\ln\frac{\binom{2n}{\leq d}}{\delta}}}{\sqrt{2m}}$$

where $C_{\vec{\alpha}}$ is at most our bound on $x^{\vec{\alpha}}$. We can add a λ_ℓ-multiple of this to $\lambda_\ell p_\ell(\vec{x})$ to obtain $\frac{1}{m}\sum_{i=1}^m\lambda_\ell p_\ell(\vec{x}^{(i)})|_{\vec{\rho}^{(i)}}$ as in the averaged refutation, plus a bounded error term.

Finally we consider the testable constraints in the expression. Note that without the terms from these constraints, the original sum-of-squares refutation sums to

$$-1-\left(\sum_{\tilde{j}=1}^{\tilde{r}}\tilde{\sigma}_{\tilde{j}}(\vec{x})\tilde{g}_{\tilde{j}}(\vec{x})+\sum_{\tilde{k}=1}^{\tilde{s}}\tilde{u}_{\tilde{k}}(\vec{x})\tilde{h}_{\tilde{k}}(\vec{x})+\sum_{\tilde{\ell}=1}^{\tilde{t}}\lambda_{\tilde{\ell}}\tilde{p}_{\tilde{\ell}}(\vec{x})\right)$$

where of course, the $\tilde{\sigma}_{\tilde{j}}(\vec{x})$ are sums of squares and the $\tilde{u}_{\tilde{k}}(\vec{x})$ are arbitrary polynomials. Now, for our choice of m, with probability $1-\delta$, for all but $\frac{2}{3S}m$ of the partial examples, our constraints are simultaneously witnessed under $\vec{\rho}^{(i)}$. By the definition of witnessing, the lower bounds under each such $\vec{\rho}^{(i)}$ on the corresponding witnessed terms in the assumed sum-of-squares refutation are at least 0, and these can be derived from our support bounds for each $\vec{\rho}^{(i)}$. Thus, using our bounds on monomials, we can bound the averaged formal expression by

$$-1 - \frac{1}{m} \sum_{\substack{\vec{\rho}^{(i)} \text{ not} \\ \text{witnessed}}} \left(\sum_{\tilde{j}=1}^{\tilde{r}} (\tilde{\sigma}_{\tilde{j}}(\vec{x}^{(i)}) \tilde{g}_{\tilde{j}}(\vec{x}^{(i)}))|_{\vec{\rho}^{(i)}} + \sum_{\tilde{k}=1}^{\tilde{t}} (\tilde{u}_{\tilde{k}}(\vec{x}^{(i)}) \tilde{h}_{\tilde{k}}(\vec{x}^{(i)}))|_{\vec{\rho}^{(i)}} + \sum_{\tilde{\ell}=1}^{\tilde{t}} \lambda_{\tilde{\ell}} \tilde{p}_{\tilde{\ell}}(\vec{x}^{(i)})|_{\vec{\rho}^{(i)}} \right) \leq -1 + \frac{2}{3}$$

since the testable portion of the proof has lower bound at least $-S$. Note, moreover, that we can derive the upper bound on the unwitnessed expressions using our bounds on the empirical monomials, obtaining some additional error terms. Summing over all of the error terms, we find that since the overall proof has naïve norm at most S, we can obtain that the total error term is at most $1/6$ by an appropriate choice of m. Thus, overall, we see that we can construct a sum-of-squares derivation of $-1/6$, and hence also of -1 since we can multiply each coefficient by 6 and still obtain a valid sum-of-squares expression for the system. Therefore we detect that the system is infeasible and reject with probability $1 - \delta$ as needed in this case. □

Directions for Future Work

An important shortcoming of our approach is that it is propositional so far. One can make use of "independently quantified expressions" (Valiant 2000) to extend it to some fragments of first-order logic via propositionalization. But, an interesting question is whether or not we can employ richer representations by making use of the symmetries in such instances, along the lines of work on relational linear programming by Kersting et al. (2017). Finally, we required the implicitly learned constraints to be testable with probability $1 - 1/2S$ where S is the naïve norm of the proof. It would be desirable to make use of $1 - \epsilon$-testable constraints for arbitrary ϵ, perhaps only establishing that the portion of the distribution satisfying the query system has total probability at most ϵ.

Acknowledgements

We thank the anonymous reviewers for their detailed comments.

References

Alekhnovich, M.; Ben-Sasson, E.; Razborov, A. A.; and Wigderson, A. 2002. Space complexity in propositional calculus. *SIAM J. Comput.* 31(4):1184–1211.

Ansótegui, C.; Bonet, M. L.; Levy, J.; and Manyá, F. 2008. Measuring the hardness of SAT instances. In *Proc. AAAI'08*, 222–228.

Arora, S.; Ge, R.; Ma, T.; and Risteski, A. 2017. Provable learning of noisy-OR networks. In *Proc. 49th STOC*, 1057–1066.

Arora, S.; Hazan, E.; and Kale, S. 2005. Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. In *Proc. 46th FOCS*, 339–348.

Bacchus, F.; Grove, A. J.; Halpern, J. Y.; and Koller, D. 1996. From statistical knowledge bases to degrees of belief. *Artificial intelligence* 87(1-2):75–143.

Barak, B., and Moitra, A. 2016. Noisy tensor completion via the sum-of-squares hierarchy. In *Proc. 29th COLT*, volume 49 of *JMLR W&CP*. 1–29.

Barak, B.; Brandão, F. G. S. L.; Harrow, A. W.; Kelner, J.; Steurer, D.; and Zhou, Y. 2012. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proc. 44th STOC*, 307–326. Extended version available as arXiv:1205.4484.

Barak, B.; Kelner, J.; and Steurer, D. 2014. Rounding sum of squares relaxations. In *Proc. 46th STOC*, 31–40.

Barak, B.; Kelner, J.; and Steurer, D. 2015. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proc. 47th STOC*, 143–151.

Barak, B.; Raghavendra, P.; and Steurer, D. 2011. Rounding semidefinite programming hierarchies via global correlation. In *Proc. 52nd FOCS*, 472–481.

Berkholz, C. 2018. The relation between polynomial calculus, Sherali-Adams, and sum-of-squares proofs. In *Proc. 35th STACS*, LIPIcs, 11:1–11:14.

Boyd, S., and Vandenberghe, L. 2004. *Convex optimization*. Cambridge university press.

Chavira, M., and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6–7):772–799.

De Raedt, L., and Kersting, K. 2008. Probabilistic inductive logic programming. In De Raedt, L.; Frasconi, P.; Kersting, K.; and Muggleton, S., eds., *Probabilistic Inductive Logic Programming: Theory and Applications*, volume 4911 of *LNCS*. Springer. 1–27.

Domingos, P., and Webb, W. A. 2012. A tractable first-order probabilistic logic. In *Proc. 26th AAAI*, 1902–1909.

Erdogdu, M. A.; Deshpande, Y.; and Montanari, A. 2017. Inference in graphical models via semidefinite programming hierarchies. In *Proc. 31st NIPS*, 417–425.

Fagin, R.; Halpern, J. Y.; and Megiddo, N. 1990. A logic for reasoning about probabilities. *Information and computation* 87(1-2):78–128.

Gogate, V., and Domingos, P. 2011. Probabilistic theorem proving. In *Proc. 27th UAI*, 256–265.

Grigoriev, D., and Vorobjov, N. 2001. Complexity of null- and positivstellensatz proofs. *Ann. Pure and Applied Logic* 113(1):153–160.

Grosof, B. N. 1986. An inequality paradigm for probabilistic knowledge: the logic of conditional probability intervals. In *Machine Intelligence and Pattern Recognition*, volume 4. Elsevier. 259–275.

Grötschel, M.; Lovász, L.; and Schrijver, A. 2012. *Geometric algorithms and combinatorial optimization*. Springer, 2nd edition.

Halpern, J. Y., and Pucella, R. 2007. Characterizing and reasoning about probabilistic and non-probabilistic expectation. *J. ACM* 54(3):15.

Hopkins, S. B., and Steurer, D. 2017. Efficient Bayesian estimation from few samples: community detection and related problems. In *Proc. 58th FOCS*, 379–390.

Hopkins, S.; Shi, J.; and Steurer, D. 2015. Tensor principal component analysis via sum-of-square proofs. In *Proc. 28th COLT*, volume 40 of *JMLR W&CP*. 956–1006.

Juba, B. 2013. Implicit learning of common sense for reasoning. In *Proc. 23rd IJCAI*, 939–946.

Kahn, J.; Kalai, G.; and Linial, N. 1988. The influence of variables on Boolean functions. In *Proc. 29th FOCS*, 68–80.

Karlin, A. R.; Mathieu, C.; and Nguyen, C. T. 2011. Integrality gaps of linear and semi-definite programming relaxations for knapsack. In *International Conference on Integer Programming and Combinatorial Optimization*, 301–314. Springer.

Kauers, M.; O'Donnell, R.; Tan, L.-Y.; and Zhou, Y. 2014. Hypercontractive inequalities via SOS, and the Frankl-Rödl graph. In *Proc. 25th SODA*, 1644–1658.

Kersting, K., and De Raedt, L. 2008. Basic principles of learning bayesian logic programs. In De Raedt, L.; Frasconi, P.; Kersting, K.; and Muggleton, S., eds., *Probabilistic Inductive Logic Programming: Theory and Applications*, volume 4911 of *LNCS*. Springer. 189–221.

Kersting, K.; Mladenov, M.; and Tokmakov, P. 2017. Relational linear programming. *Artificial Intelligence* 244:188–216.

Khardon, R., and Roth, D. 1997. Learning to reason. *J. ACM* 44(5):697–725.

Khot, T.; Natarajan, S.; Kersting, K.; and Shavlik, J. 2015. Gradient-based boosting for statistical relational learning: the Markov logic network and missing data cases. *Mach. Learn.* 100:75–100.

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models*. Cambridge, MA: MIT Press.

Lasserre, J. B. 2001. Global optimzation with polynomials and the problem of moments. *SIAM J. Optimization* 11(3):796–817.

Lasserre, J. B. 2010. *Moments, Positive Polynomials, and Their Applications*. London: Imperial College Press.

Ma, T.; Shi, J.; and Steurer, D. 2016. Polynomial-time tensor decompositions with sum-of-squares. In *Proc. 57th FOCS*, 438–446.

Michael, L. 2010. Partial observability and learnability. *Artificial Intelligence* 174(11):639–669.

Michael, L. 2014. Simultaneous learning and prediction. In *Proc. 14th KR*, 348–357.

Mossel, E.; O'Donnell, R.; and Oleszkiewicz, K. 2010. Noise stability of functions with low influences: invariance and optimality. *Ann. Math.* 171(1):295–341.

Narasimhan, M., and Bilmes, J. A. 2004. PAC-learning bounded tree-width graphical models. In *Proc. 20th UAI*, 410–417.

Nesterov, Y. 2000. Squared functional systems and optimization problems. *High performance optimization* 13:405–440.

Nilsson, N. J. 1986. Probabilistic logic. *Artificial Intelligence* 28:71–87.

O'Donnell, R., and Zhou, Y. 2013. Approximability and proof complexity. In *Proc. 24th SODA*, 1537–1556.

O'Donnell, R. 2017. SOS is not obviously automatizable, even approximately. In *Proc. 8th Innovations in Theoretical Computer Science*, volume 67 of *LIPIcs*.

Parrilo, P. A. 2000. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. Ph.D. Dissertation, California Institute of Technology.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Poon, H., and Domingos, P. 2011. Sum-product networks: a new deep architecture. In *Proc. 27th UAI*, 337–346.

Putinar, M. 1993. Positive polynomials on compact semialgebraic sets. *Indiana U. Math. J.* 42:969–984.

Raghavendra, P., and Weitz, B. 2017. On the bit complexity of sum-of-squares proofs. In *Proc. 44th International Colloquium on Automata, Languages, and Programming*, volume 80 of *LIPIcs*.

Rakhlin, A., and Sridharan, K. 2015. Hierarchies of relaxations of online prediction problems with evolving constraints. In *Proc. 28th COLT*, volume 40 of *JMLR W&CP*. 1–23.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Mach. Learn.* 62:107–136.

Rubin, D. B. 1976. Inference and missing data. *Biometrika* 63(3):581–592.

Sesh Kumar, K. S., and Bach, F. 2013. Convex relaxations for learning bounded treewidth decomposable graphs. In *Proc. 30th ICML*, 525–533.

Shor, N. 1987. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics and Systems Analysis* 23(5):695–700.

Valiant, L. G. 2000. Robust logics. *Artificial Intelligence* 117:231–253.