Beyond Parity Constraints: Fourier Analysis of Hash Functions for Inference

Tudor Achim TACHIM@CS.STANFORD.EDU
Stanford University, 353 Serra Mall, Stanford, CA 94305

Ashish Sabharwal ASHISHS@ALLENAI.ORG
Allen Institute for Artificial Intelligence, 2157 N Northlake Way, Seattle, WA 98103

Stefano Ermon ERMON@CS.STANFORD.EDU
Stanford University, 353 Serra Mall, Stanford, CA 94305

Abstract

Random projections have played an important role in scaling up machine learning and data mining algorithms. Recently they have also been applied to probabilistic inference to estimate properties of high-dimensional distributions; however, they all rely on the same class of projections based on universal hashing. We provide a general framework to analyze random projections which relates their statistical properties to their Fourier spectrum, which is a well-studied area of theoretical computer science. Using this framework we introduce two new classes of hash functions for probabilistic inference and model counting that show promising performance on synthetic and real-world benchmarks.

1 Introduction

Probabilistic inference is a fundamental task in machine learning and Bayesian statistics (Koller & Friedman, 2009). Inference problems like computing expectations and normalization constants of probabilistic models involve the computation of high-dimensional integrals, which are generally intractable to compute exactly. Instead, these integrals are approximated using two main classes of algorithms: sampling methods like Markov Chain Monte Carlo (Jerrum & Sinclair, 1997), which explore the intractable distribution using local moves, and variational inference (Wainwright & Jordan, 2008; Jordan et al., 1999), which uses tractable probability distributions to approximate the original, complex distribution. These approaches are widely used in practice, but rarely provide general formal guarantees about the quality of their approximations.

More recently, a new class of approximate inference algorithms has been introduced (Gomes et al., 2006a; 2007; 2006b; Chakraborty et al., 2013a;b; Ermon et al., 2014; Ivrii et al., 2015; Achlioptas & Jiang, 2015; Belle et al., 2015; Zhao et al., 2016; Asteris & Dimakis, 2016; Hsu et al., 2016) based on projecting complex, high-dimensional distributions into lower dimensional spaces where their properties are easier to analyze. For the approach to work, the random projection needs to satisfy two properties. First, it has to "preserve" the key properties of the original distribution, at least probabilistically. Second, the resulting lower-dimensional distribution should be easier to analyze. Currently, all these algorithms are based on hash functions implemented using random (low-density) parity check codes (Zhao et al., 2016; Asteris & Dimakis, 2016). These hash functions have excellent statistical properties, in the sense that to a large extent the projection preserves the properties of the original distribution with high probability. Parity-based projections, however, are highly oscillatory and have a non-convex nature, leading to probability distributions that although lower-dimensional, can still be difficult to analyze.

A natural question is whether there exists other classes of hash functions that can be employed for approximate inference. A wide variety of random projections are known for traditional machine learning applications, and their properties are well understood (Achlioptas et al., 2002; Blum, 2005; Rajaraman & Ullman, 2011). However, current analyses in the context of statistical inference are ad-hoc and rely on specific properties of parity constraints, and are thus unsuitable for studying new candidate hash functions.

In this paper we address this theoretical gap by showing a connection between the Fourier spectra of arbitrary (discrete) functions and the statistical properties of their corresponding hash families. Because the Fourier spec-

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

tra of many Boolean functions are well studied in theoretical computer science, learning theory and computational social choice (ODonnell, 2003), this theoretical bridge allows us to quickly make predictions about the statistical performance of a wide class of functions besides parity constraints. Using this bridge, we introduce two new hash functions for use in probabilistic inference and model counting algorithms (Ermon et al., 2013c; Chakraborty et al., 2013b; Gomes et al., 2006a; 2007) and show empirically that these functions perform very well on real-world and synthetic examples.

2 Preliminaries

2.1 Inference by Hashing

There are many recent algorithms that rely on universal hash functions for approximate inference, including WISH (Ermon et al., 2013c), ApproxMC (Chakraborty et al., 2013b), XORSample (Gomes et al., 2006a), and randomly-projected variational inference (Hsu et al., 2016). These methods all build on the original probabilistic reduction between #P and NP developed by Valiant & Vazirani (1986), Stockmeyer (1985), and Sipser (1983).

These methods aim to solve the problem of discrete integration; namely, estimating $\sum_{x\in\mathcal{X}} w(x)$ where w is a non-negative weight function and $\mathcal{X} = \{0,1\}^n$ is a high dimensional set, e.g., corresponding to all possible assignments to n binary variables. If $w(x)$ represents the unnormalized mass of a probability distribution, then the sum is the partition function of that distribution. Non-binary graphical models can be represented in this framework by introducing additional variables and factors corresponding to the binary encoding of the state of each variable. If $w(x)$ is a binary-valued function indicating whether or not the variable configuration x satisfies some constraints, then estimating the sum corresponds to the model counting problem (Chakraborty et al., 2013b).

The key idea behind hashing approaches for discrete integration is to sidestep the exponentially large sums by analyzing the properties of $\mathcal{X}$ when projected randomly onto lower-dimensional spaces and using these low-dimensional projections to approximate the original sum. The functions we select randomly for the projections come from universal hash families, which we review briefly below (Vadhan, 2011; Goldreich, 2011):

Definition 1. *A set of functions $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ is ϵ-SU (Strongly Universal) if, when h is chosen uniformly at random from $\mathcal{H}$, the following conditions hold: (i) for all $x \in \{0,1\}^n$, $h(x)$ is distributed uniformly in $\{0,1\}^m$ and (ii) for all $x_1 \neq x_2 \in \{0,1\}^n$ and $y_1, y_2 \in \{0,1\}^m$ we have that $P[h(x_1) = y_1, h(x_2) = y_2] \leq \epsilon/2^m$.*

The family of fully-independent hash functions, which act independently on any input point in S, is $1/2^m$-strongly universal, but requires $m2^n$ bits to specify. Pairwise independent hash functions where $h(x_1)$ is independent from $h(x_2)$ for any (x_1, x_2) pair, can be specified more compactly, requiring only $O(nm)$ bits, and are also $1/2^m$-strongly universal. They are based on parity constraints and are of the form $h_{A,b}(x) = Ax + b \mod 2$ where A and b are selected uniformly at random from $\{0,1\}^{m\times n}$ and $\{0,1\}^{m\times 1}$ respectively.

Returning to our original goal of estimating $\sum_{x\in\mathcal{X}} w(x)$, we rely on the key idea that the sum can be estimated by splitting $\mathcal{X}$ using h drawn from a universal hash family $\mathcal{H}$ and considering the properties of a random cell $h^{-1}(y)\cap\mathcal{X}$. For example, suppose $w(x)$ is an indicator function for a set S, e.g., defined as a set of constraints as in a model counting application. A randomized algorithm $\mathcal{A}$ for estimating the size of a set $S \subseteq \{0,1\}^n$ with high probability is to draw T hash functions $h_{1,m}, \cdots, h_{T,m}$ from universal hash families $\mathcal{H}^m = \{h : \{0,1\}^n \to \{0,1\}^m\}$ for increasing m and returning 2^{m-1} as soon as the intersection $h^{-1}_{t,m}(y_{t,m}) \cap S$ is empty for more than half of the T trials. The reason this algorithm works is that by uniformity (property 1 in Definition 1), in expectation the cell $h^{-1}_{t,m}(y_{t,m}) \cap S$ contains $|S|/2^m$ elements (the expectation is over the choice of hash functions). Furthermore, by the second property in Definition 1, the variance of the number of elements of S that show up in a cell $h^{-1}(y)$ is controlled, which guarantees concentration about the mean over the T trials.

When designing new hash families for approximate inference, then, our goal is to bound the probability that the r.v.

$$X(h, S, y) = |h^{-1}(y) \cap S| \tag{1}$$

deviates from its mean, $\mu(h, S, y)$, when h is drawn from the family $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ and y is any vector in $\{0,1\}^m$. When $\mathcal{H}$ is universal we note that $\mu = |S|/2^m$. To be useful for hashing-based approaches for approximate counting X needs to concentrate about its mean; in particular, we say that $\mathcal{H}$ is *weakly (k,δ)-concentrated* if $\Pr[X \geq \mu + \sqrt{k}] \leq 1/\delta$ and $\Pr[X \leq \mu - \sqrt{k}] \leq 1/\delta$. Ermon et al. (2014) showed that the randomized algorithm above, using $T \geq 8\ln(8n)$ trials, computes $|S|$ within a factor of 4 with probability at least 3/4 when X is *weakly $(\mu^2, 4)$-concentrated*.

It can be seen from standard concentration inequalities that if h is chosen from a family of $1/2^m$-SU hash functions, then X will be *weakly $(\mu^2, 4)$-concentrated*. Interestingly, Ermon et al. (2014) showed that this condition can be relaxed to the notion of *average* universality:

Definition 2. *We say that a hash family $\mathcal{H} = \{h : \{0,1\}^n \to \{0,1\}^m\}$ is (ϵ, q)-AU (Average Universal) if the following two conditions hold when h is chosen uniformly at random from $\mathcal{H}$:*

1. *Uniformity: for all $x \in \{0,1\}^n$, $h(x)$ is distributed uniformly in $\{0,1\}^m$.*
2. *Average universality: for all $S \subseteq \{0,1\}^n$ with $|Q| = q$ and $y_1, y_2 \in \{0,1\}^m$, $\sum_{\substack{x_1,x_2 \in Q \\ x_1 \neq x_2}} P[h(x_1) = y_1, h(x_2) = y_2] \leq q(q-1)\epsilon/2^m$.*

Proposition 1 ((ϵ, q)-AU and Concentration)**.** *If $\mathcal{H}$ is an (ϵ, q)-AU hash family, then when $|S| = q$ and h is drawn from $\mathcal{H}$, $X(h, S, y)$ is weakly (μ^2, δ) concentrated if $\epsilon \leq (\mu/(\delta - 1) + \mu - 1)/(|S| - 1)$. As a corollary, if $\epsilon \leq \frac{31}{5(4|S|-1)}$ then the randomized algorithm $\mathcal{A}$ for estimating the size of a set S will output $|S|$ within a factor of 4 with probability at least 3/4 using $T \geq 8\ln(8n)$ trials.*

Importantly, it was shown that average universal hash families can be implemented using low-density parity check codes (Ermon et al., 2014; Zhao et al., 2016; Asteris & Dimakis, 2016). The significance of this result is that it shows the existence of other hash function families that can be used for approximate inference besides the traditional pairwise independent ones. These alternative hash families (equivalently, random projections) have major practical advantages. Intuitively, they are random enough to preserve the properties of the original set $\mathcal{X}$ being hashed. However, their behavior is more predictable so that the projected model is easier to analyze. The practical gains can be huge (Ivrii et al., 2015; Achlioptas & Jiang, 2015; Zhao et al., 2016).

A natural question is whether there exists other classes of hash functions with good statistical properties. In particular, our goal is to design new hash families that are more tractable to optimize over than parity functions, but retain useful statistical properties. The chief difficulty in analyzing new families is that the quantity $\Pr[h(x_1) = y_1, h(x_2) = y_2]$, used to bound ϵ, depends on both the underlying hash function and on the "shape" or geometry of the set Q in part 2 of Definition 2. We address this via a hash family construction for arbitrary boolean function "templates" that lets bound the joint probability in terms of the Fourier spectrum of the underlying template.

2.2 Fourier Spectra of Boolean Functions

We review the basics of Fourier analysis of boolean functions. For ease of notation we will consider boolean functions defined as $f : \{-1,1\}^n \to \{-1,1\}$ with -1 representing true.

Consider an inner product over boolean functions:

$$\langle f, g \rangle = \mathbf{E}[fg] = 2^{-n} \sum_{x \in \{-1,1\}^n} f(x)g(x)$$

Next we define the functions $\chi_S(x)$ to be the parity function $\chi_S(x) = \prod_{i \in S} x_i$, for all subsets $S \subseteq [n]$. Since $\mathbf{E}[\chi_S] = 0$ for $S \neq \emptyset$ and $S \neq S' \Rightarrow \mathbf{E}[\chi_S \chi'_S] = 0$, $(\chi_S)_{S \subseteq [n]}$ is an orthonormal basis for the space of functions $f : \{-1,1\}^n \to \{-1,1\}$. Formally, we have the representation $f = \sum_{S \subseteq [n]} \hat{f}(S)\chi_S$, where $\hat{f}(S) = \langle \chi_S, f \rangle$. Since $\langle f, f \rangle = 1$ for any f, by Parseval's identitiy we have $\sum_{S \subseteq [n]} \hat{f}(S)^2 = 1$ (ODonnell, 2003).

The Fourier spectrum of a function f, namely the weights corresponding to different parity basis functions, is very closely related to its *noise sensitivity* (ODonnell, 2003):

Definition 3. *The* noise sensitivity *of $f : \{-1,1\}^n \to \{-1,1\}$, $\mathrm{NS}_\epsilon(f)$, is the probability that $f(x) \neq f(y)$ when x is drawn uniformly from $\{-1,1\}^n$ and y from $N_\epsilon(x)$, where $N_\epsilon(x)$ is the distribution obtained by flipping each bit of x independently with probability ϵ.*

The noise sensitivity of a function is intimately connected with its Fourier spectrum:

$$\mathrm{NS}_\epsilon(f) = \frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} (1 - 2\epsilon)^{|S|} \hat{f}(S)^2 \qquad (2)$$

For proofs and more background, see ODonnell (2003).

Intuitively, the noise sensitivity at small ϵ tells us how correlated the value of f is across pairs of points that are close in Hamming distance, i.e., how much f "mixes". From eq. (2), we can see that NS_ϵ increases with ϵ for $\epsilon \in [0, 1/2]$, and $\mathrm{NS}_{1/2}(f) = 1/2$ for any f.

Furthermore, by inspection of equation (2), the more concentrated the Fourier spectrum is on large sets S (recall Parseval's identity), the more noise sensitive f is. As a corollary, we can see immediately that for any ϵ the parity function over k variables, XOR_k, is the most noise sensitive function over k variables and that

$$\mathrm{NS}_\epsilon(\mathrm{XOR}_k) = \frac{1}{2} - \frac{1}{2}(1 - 2\epsilon)^k$$

This follows from equation (2) noting that the spectrum of XOR_k has a single non-zero coefficient.

Intuitively, the noise sensitivity of a function f determines how well f splits up points on average across $\{-1,1\}^n$, where the randomness is over the points in the space. Our goal is to find *families* of hash functions that split up all sets of points S, where the randomness is over the choice of functions in the family (see Definition 1). One might hope that the Fourier spectrum of f can be related closely to the variance properties of hash families based on f; indeed, in the next section we show that this is the case.

3 Hashing with Noise Sensitive Functions

We begin by defining a notion of *templated* hash families. For simplicity, we first consider functions with a fixed input size (i.e., those with a fixed sized support). Our results,

however, easily generalize to variable-length hash families, such as the short XORs used by Ermon et al. (2014), as discussed in Section 3.1.

Let $[n]$ and $[-n]$ denote $\{1,\dots,n\}$ and $\{-1,\dots,-n\}$, resp. $\sigma : [n] \to [n] \cup [-n]$ is a *signed permutation* if $(|\sigma(1)|,\dots,|\sigma(n)|)$ is a permutation of $[n]$. With some abuse of notation, for boolean variables $\{x_1,\dots,x_n\}$, we use $-x_i$ to denote the logical negation of x_i and $\sigma(x_i)$ to denote the possibly negated variable that x_i is mapped to: $\mathrm{sign}(\sigma(i))\, x_{|\sigma(i)|}$.

Definition 4 (Templated Hash Family)**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function over n variables. Let $\mathcal{O}$ be the set of all signed permutations of $[n]$; $|\mathcal{O}| = n!\,2^n$. A templated hash family $\mathcal{H}_f = \{h : \{0,1\}^n \to \{0,1\}\}$ is a family of hash functions defined as:*

$$\mathcal{H}_f = \{f(\sigma(x_1),\sigma(x_2),\dots,\sigma(x_n)) \mid \sigma \in \mathcal{O}\}$$

Specifying a function in $\mathcal{H}_f$ requires a number of random bits that is linear in n. Also, if f has a support of size k, i.e., it mentions only k of the n variables, then $\mathcal{H}_f$ effectively has $(n-k)!\,2^{n-k}$ copies of each function. Our results apply also to the subset of $\mathcal{H}_f$ with only the $\binom{n}{k}k!\,2^k$ distinct functions.

Hashing into $\{0,1\}^m$ can be done by choosing m hash functions independently from $\mathcal{H}_f$. Hence, w.l.o.g., we focus our analysis to the case of $m = 1$. Because we want to control the variance of $\sum_{x\in S} 1\{h(x) = 1\}$ for arbitrary sets S, our goal is to bound $\sum_{x_1\neq x_2\in S} \Pr[h(x_1) = 1, h(x_2) = 1]$ when h is drawn uniformly at random from $\mathcal{H}_f$ (cf. Definition 2). We start by observing that this is bounded above by $\sum_{x_1\neq x_2\in S} \Pr[h(x_1) = h(x_2)]$. If all hash functions $h \in \mathcal{H}_f$ are balanced, that is $\Pr[h(x) = 1] = 1/2$ when x is chosen uniformly at random, then $\Pr[h(x_1) = 1, h(x_2) = 1] = (1 - \Pr[h(x_1) \neq h(x_2)])/2$ when h is selected uniformly at random from $\mathcal{H}_f$. To start analyzing this sum we introduce a slight variant of NS_ϵ, referred to as $\mathrm{NS}'_w(f)$, which measures the noise sensitivity of f at *fixed* distance w. We use the notation $N_w(x)$ for the set of all configurations obtained by flipping exactly w bits of x, i.e., $N_w(x) = \{y \mid d_H(x,y) = w\}$.

Definition 5 (Fixed-Distance Noise Sensitivity)**.** *The* fixed-distance noise sensitivity $\mathrm{NS}'_w(f)$ *is the probability that $f(x) \neq f(y)$ when x is drawn uniformly from $\{0,1\}^n$ and y uniformly from $N_w(x)$.*

The following relates the noise sensitivity of a function f with that of all functions in $\mathcal{H}_f$.

Proposition 2. *For any $h \in \mathcal{H}_f, 0 \le w \le n$, and $\epsilon \le 1/2$, we have $\mathrm{NS}'_w(h) = \mathrm{NS}'_w(f)$ and $\mathrm{NS}_\epsilon(h) = \mathrm{NS}_\epsilon(f)$.*

Proof. Let $h(x_1,\cdots,x_n) = f(\sigma(x_1),\cdots,\sigma(x_n))$. Then

$$\begin{aligned}\mathrm{NS}'_w(h) &= \frac{1}{2^n\binom{n}{w}}\sum_x \sum_{y\in N_w(x)} 1\{h(x)\neq h(y)\}\\ &= \frac{1}{2^n\binom{n}{w}}\sum_x \sum_{y\in N_w(x)} 1\{f(\sigma(x))\neq f(\sigma(y))\}\end{aligned}$$

Since σ bijects $\{0,1\}^n$ to $\{0,1\}^n$ and $d_H(x,y) = d_H(\sigma(x),\sigma(y))$ since σ is a signed permutation, the sum is symmetric about σ and is equal to

$$\frac{1}{2^n\binom{n}{w}}\sum_x \sum_{y\in N_w(x)} 1\{f(x)\neq f(y)\} = \mathrm{NS}'_w(f)$$

Thus $\mathrm{NS}'_w(h) = \mathrm{NS}'_w(f)$. Since $\mathrm{NS}_\epsilon(f) = \sum_w \binom{n}{w}\epsilon^k(1-\epsilon)^{n-k}\mathrm{NS}'_w(f)$, the result for NS_ϵ follows. □

It turns out that $\Pr[h(x_1) = h(x_2)]$ can be expressed very naturally in terms of $\mathrm{NS}'_w(f)$ when $d_H(x_1,x_2) = w$ and h is drawn from $\mathcal{H}_f$. Note, however, the subtlety in the underlying probability spaces: $\Pr[h(x_1) = h(x_2)]$ is a probability over choices of h, whereas $\mathrm{NS}'_w(f)$ refers to a fixed function f but considers randomness over which w bits of a randomly chosen x are flipped. We show next that one can reconcile this difference in underlying probability spaces using a symmetry argument:

Lemma 1 (Noise Sensitivity vs. Clash Probability)**.** *Let $\mathcal{H}_f$ be a templated family of hash functions. Then the clash probability $\Pr_{h\sim\mathcal{H}_f}[h(x) = h(y)]$ when $d_H(x,y) = w$ is closely related to the noise sensitivity of f:*

$$1 - \mathrm{NS}'_w(f) = \Pr_{h\sim\mathcal{H}_f}[h(x) = h(y)] \tag{3}$$

Proof. The proof relies on the symmetry of the hash family construction. If h is drawn from $\mathcal{H}_f$, then

$$\begin{aligned}1 - \mathrm{NS}'_w(h) &= \Pr_{\substack{x'\sim\{0,1\}^n\\ y'\sim N_w(x')}}[h(x') = h(y')]\\ &= \frac{1}{2^n\binom{n}{w}}\sum_{x'} \sum_{y'\in N_w(x')} 1\{h(x') = h(y')\}\\ &= \frac{1}{2^n\binom{n}{w}}\sum_{x'} \sum_{y'\in N_w(x')} 1\{f(x') = f(y')\}\end{aligned}$$

where the last line uses the fact that summing over x' and y' negates the effect of the signed permutation that defines h. Furthermore, for any x', y' with $d_H(x',y') = w$, there are $w!(n-w)!$ signed permutations σ that map x, y to x', y' (recall $d_H(x,y) = w$). These are obtained by choosing all permutations that yield the desired positions of the flipped bits and fixing the sign change to map x to x'. Thus we can

rewrite the sum over x' and y' to be over all possible signed permutations σ:

$$\frac{1}{2^n\binom{n}{w}}\sum_{\sigma}\frac{1\{f(\sigma(x))=f(\sigma(y))\}}{w!(n-w)!}$$
$$=\frac{1}{2^n n!}\sum_{\sigma}1\{f(\sigma(x))=f(\sigma(y))\}$$
$$=\Pr_{h\sim\mathcal{H}_f}[h(x)=h(y)]$$

as desired. □

Using this connection between clash probability and noise sensitivity, we can generalize Theorem 3 of Ermon et al. (2014) from short XORs to any boolean function:

Theorem 1. *Let $f:\{0,1\}^n\to\{0,1\}$ be any Boolean function, and $\mathcal{H}_f$ be the corresponding templated hash family. Let $0\le q\le 2^n$, τ be a permutation of $[n]$ such that $\mathrm{NS}'_{\tau(j)}(f)$ is non-increasing with j, $w^*=\max\left\{w\mid\sum_{j=1}^{w}\binom{n}{\tau(j)}\le q-1\right\}$, and*

$$\epsilon(n,m,q,f)=\frac{1}{q-1}\left(\sum_{w=1}^{w^*}\binom{n}{\tau(w)}\left(1-\mathrm{NS}'_{\tau(w)}(f)\right)^m+\left(q-1-\sum_{w=1}^{w^*}\binom{n}{\tau(w)}\right)\left(1-\mathrm{NS}'_{\tau(w^*+1)}(f)\right)^m\right)$$

Then the family formed by the conjunction of m independent draws from $\mathcal{H}_f$ is a $(\epsilon(n,m,q,f),q)$-AU hash family.

In particular, if f is such that $\mathrm{NS}'_n(f)$ is a non-increasing function of k, we can set τ to the identity permutation and recover the same form as Theorem 3 of Ermon et al. (2014). For general f, one can instead use tools from the theory of noise sensitivity to compute—numerically or analytically—the value of $\mathrm{NS}'_w(f)$ for various k and use that to determine the ordering τ.

The significance of Theorem 1 is that it provides a general framework to predict the statistical properties of a family of hash functions $\mathcal{H}_f$ constructed using a "template". The theorem does not pose any restriction on the function f. The noise sensitivity is known for a large class of functions (ODonnell, 2003), and those results can be readily applied in our context. Furthermore, it is possible to evaluate in closed form the noise sensitivity of functions obtained by composing other simpler functions. Similarly, the Fourier spectrum is known for many common functions; given the relationship between noise sensitivity and Fourier spectrum, this information can also be used to predict the performance of the corresponding hash functions. More broadly, Theorem 1 provides useful guidance towards designing effective hash function families.

Having connected the variance expression for (1) with the noise sensitivities at specific hamming distances w of the hash family's underlying function (the NS'_w's), we are now ready to directly connect the variance to the underlying function template's Fourier spectrum.

Proposition 3. *Let $f:\{-1,1\}^n\to\{-1,1\}$. The fixed-distance noise sensitivity of f at a distance $0\le w\le n$ can be computed in closed form in either of two ways:*

$$\mathrm{NS}'_w(f)=\sum_{T\subseteq\sup(f)}p(w,n,|T|)\hat{f}(T)^2$$
$$\mathrm{NS}'_w(f)=\frac{1}{\binom{n}{w}}\frac{\epsilon_w}{(1-\epsilon_w)^n}\sum_{i=1}^{n+1}b_{iw}\mathrm{NS}_{\epsilon_i}(h)$$

where $\sup(f), p(w,n,|T|), \epsilon_w, b_{iw}$ are defined in the Appendix.

The proof is based on the following connection between the Fourier spectrum of f, $\mathrm{NS}_\epsilon(f)$ and $\mathrm{NS}'_w(f)$:

$$\frac{1}{2}-\frac{1}{2}\sum_{T\subseteq[n]}(1-2\epsilon)^{|T|}\hat{f}(T)^2=\mathrm{NS}_\epsilon(f)=$$
$$\sum_{w=0}^{n}\binom{n}{w}\epsilon^w(1-\epsilon)^{n-w}\mathrm{NS}'_w(f)=$$
$$(1-\epsilon)^n\sum_{w=0}^{n}\left(\frac{\epsilon}{1-\epsilon}\right)^w\binom{n}{w}\mathrm{NS}'_w(f)$$

The result formalizes the intuition that as the mass of the Fourier spectrum of f is placed on higher order coefficients, the average noise sensitivities at fixed distances $\mathrm{NS}'_w(f)$ increase, which in turn implies that the joint probabilities $\Pr[h(x_1)=h(x_2)]$ decrease, thus reducing the variance of (1) corresponding to the hash family. The last equation shows that $\mathrm{NS}_\epsilon(f)$ can be computed by evaluating an n-th degree polynomial with coefficients $\binom{n}{w}\mathrm{NS}'_w(f)$ at the point $\epsilon/(1-\epsilon)$. This means that given a closed-form expression for $\mathrm{NS}_\epsilon(f)$, we can recover the coefficients (the $\mathrm{NS}'_w(f)$'s) exactly by evaluating $\mathrm{NS}_\epsilon(f)$ at $\epsilon_0,\ldots,\epsilon_n$ and multiplying by the inverse of the Vandermonde matrix over $\epsilon_0/(1-\epsilon_0),\ldots,\epsilon_n/(1-\epsilon_n)$, which can be evaluated in closed form. The details of the procedure are available in the Appendix.

3.1 Variable-Length Hash Families

We can also extend these results easily to variable-length families of hash functions, as follows.

Definition 6. *$\mathcal{H}_{(f,n,\rho)}$ is a templated hash family with parameters f and ρ, where h is drawn by first choosing the number of variables for the function, $k\in[n]$, with probability $\binom{n}{k}\rho^k(1-\rho)^{n-k}$ and then choosing $h\in\mathcal{H}_{f_k}$ uniformly at random.*

As an example, we see that $\mathcal{H}_{(\mathrm{XOR},n,\rho)}$ recovers the low-density parity hash functions of Ermon et al. (2014). It follows from Lemma 1 that when x and y are at hamming distance w, we have

$$\begin{aligned}&\Pr_{h\sim\mathcal{H}_{(f,n,\rho)}}[h(x)=h(y)]\\&\quad=\sum_{k=0}^{n}\binom{n}{k}\rho^k(1-\rho)^{n-k}\Pr_{h\sim\mathcal{H}_{f_k}}[h(x)=h(y)]\\&\quad=1-\sum_{k=0}^{n}\binom{n}{k}\rho^k(1-\rho)^{n-k}\mathrm{NS}'_w(f_k)\end{aligned}$$

so the technical results of Theorem 1 can be applied directly to this "mixture" of hash families.

4 Discrete Integration and Model Counting

Motivated by the connection between Fourier spectrum and hash family variance, we introduce two new hash functions for use in inference and model counting that show excellent computational efficiency and also reasonable statistical power in practice when used in our templated construction from Definition 4. In particular, they are known to be among the most noise sensitive monotone functions, where a monotone function is one for which flipping an input bit of x to obtain y implies that $f(y)\geq f(x)$. This property is useful for our applications because monotone functions are easier to optimize over. The noise sensitivity of these functions has been extensively studied in computational social choice theory, where they have been used to study the formal properties of voting systems (ODonnell, 2003).

TRIBES functions are disjunctive normal form (DNF) formulas defined by splitting n variables into b groups of size a, and returning true if any of the groups have variables that are all true and false otherwise (Ben-Or & Linial). These groups of a variables are typically called *coalitions*. a and b are chosen so that the probability that a randomly chosen point evaluates to FALSE, $(1-1/2^a)^b$, is as close as possible to $1/2$. Formally, if we write the coalitions as $A_1,\ldots,A_b\subseteq[n]$, then

$$\mathrm{TRIBES}(x_1,\ldots,x_n)=\bigvee_{i=1}^{b}\left(\bigwedge_{j=1}^{a}x_{A_{ij}}\right)$$

Note that if x is selected uniformly at random from $\{0,1\}^n$, $\Pr[\mathrm{TRIBES}(x)=1]=1-(1-1/2^a)^b$. Tribes functions can be represented compactly in conjunctive normal form (CNF) using b extra variables, n clauses of length two, and one clause of length b. This makes TRIBES functions especially well-suited for model counting applications.

MAJORITY functions return true if the majority of their inputs are true. Formally,

$$\mathrm{MAJ}(x_1,\ldots,x_n)=\mathrm{sgn}(x_1+\cdots+x_n-n/2)$$

When n is even the MAJORITY template is instantiated with an auxiliary bit drawn uniformly at random that signals in which direction ties should be broken. Using the bijection between x and its complement $\bar{x}$ and the tie-breaking random bit we see that $\Pr[\mathrm{MAJ}(x)=1]=1/2$ when x is selected uniformly at random from $\{0,1\}^n$. Because MAJORITY can be represented as a linear constraint, it is especially well-suited for discrete integration applications which use global optimization toolkits to answer MAP queries (Ermon et al., 2013c).

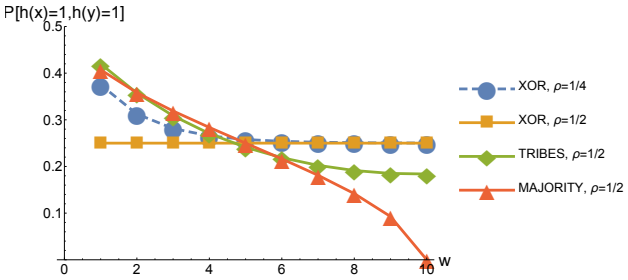

Figure 1. Joint probabilities $\Pr[h(x)=1,h(y)=1]$ at distances $w=d_H(x,y)=1,\ldots,n$, with $n=10$, for XOR, TRIBES, and MAJORITY hash families $\mathcal{H}_{(f,n,\rho)}$ at $\rho=1/2$ and $\rho=1/4$ for XORs.

Figure 1 shows the clash probabilities for several hash families $\mathcal{H}_{(f,n,\rho)}$, using our framework to compute them at fixed distances w based on the known Fourier spectra of the templates. We can see that when the probability ρ of including variables in the hash function is 1/2, the parity hash functions are uncorrelated at all distances. Furthermore, as ρ decreases to 1/4 the correlation increases at small distances but remains small for large w. Finally, both TRIBES and MAJORITY are somewhat correlated at small and large w, with MAJORITY being very highly negatively correlated at $w=n$ because flipping all the bits in x flips the output of the function.

Our new hash functions can be used in a wide array of approximate inference and probabilistic counting techniques such as ApproxMC (Chakraborty et al., 2013b), MBound (Gomes et al., 2006a), WISH (Ermon et al., 2013c;b;a), and others (Zhu & Ermon, 2015; Hadjis & Ermon, 2014). Although they are not noise sensitive enough to give upper bounds with high probability, by substituting them for parity constraints in these methods we show that in practice they obtain lower bounds an order of magnitude faster than short parity constraints, the next best method. We focus on augmenting WISH with the new hash families, but the method can be used in the other algorithms (Ivrii et al., 2015; Achlioptas & Jiang, 2015; Belle et al., 2015) with minor modifications.

WISH is an algorithm for approximately solving the dis-

crete integration problem $\sum_{x\in\{0,1\}^n} w(x)$. In a probabilistic inference context $w(x)$ is interpreted to be the unnormalized probability mass of some distribution; in particular, we use $w(x) = \prod_\alpha \psi_\alpha(x_\alpha)$ or $w(x) = \exp(\theta'\varphi(x))$. The log partition function, $\log Z = \log \sum_{x\in\{0,1\}^n} \exp(\theta'\varphi(x))$, is necessary for computing the log likelihood of evidence which can be used both as a heuristic for stopping the training process early and to compare which of a set of distributions are better models for a fixed dataset.

The original WISH algorithm operates on the same principle as the randomized algorithm $\mathcal{A}$ introduced earlier for estimating the size of a large set S defined implicitly through constraints (Ermon et al., 2013c). WISH uses hash functions drawn from a universal hash family $\mathcal{H}$ to randomly partition the domain of w in a pairwise-independent way and performs a small number of MAP queries on w in this constrained space to estimate specially-chosen quantiles of the weight function with high probability. However, the "density" of the parity functions required are impractical for large problems, and even the looser requirements given in (Ermon et al., 2014) can be intractable for modern combinatorial optimization toolkits. Thus, we focus on leveraging our new hash functions to obtain lower bounds that, while not tight, are still useful in practice and can be found much faster than previous approaches. To this end we introduce THF-WISH-LB (Algorithm 1), a modification of WISH that uses our new hash families for finding lower bounds. Together with this new algorithm we also present two new hash function templates, MAJORITY and TRIBES, which are well suited for discrete integration and model counting, respectively.

The hash families $\mathcal{H}$ used by THF-WISH-LB are parameterized by β, the desired expected value $E_{h\sim\mathcal{H}}[h(x)]$, as well as the boolean function template f and number of variables n. In order for THF-WISH-LB to provide lower bounds on Z, it must be possible to choose hash family parameters as a function of $0 < \beta \leq 1$ so that $E_{h\sim\mathcal{H}}[h(x)] \leq \beta$ for all $x \in \chi$. For example, when using TRIBES we select coalition sizes a, number of groups b, and number of constraints k so that $ab \leq n$ and $(1-(1-1/2^a)^b)^k$ is as tight a lower bound to β as possible. In practice, as n becomes large it is possible to select parameters for the hash families to bring $E_{h\sim\mathcal{H}}[h(x)]$ very close to β.

Theorem 2. *Assume a small fixed accuracy parameter δ and constant $\alpha \leq 0.0042$, and a template hash family $\mathcal{H}^\beta_{(f,n,\rho)}$ whose parameters, including ρ, can be selected as a function of $0 < \beta \leq 1$ so that $E_{h\sim\mathcal{H}}[h(x)] \leq \beta$. When $T \geq \lceil \ln(n/\delta)/\alpha \rceil$, THF-WISH-LB$(w, n, T, \mathcal{H}^\beta_{(f,n,\rho)})$ will not exceed $\log Z + \log 16$ with probability $1-\delta$.*

Proof. The proof closely follows that of Theorem 1 of (Ermon et al., 2013c) by applying Markov's inequality to the expectation $\mathbf{E}_{h\sim\mathcal{H}}[|h^{-1}(1) \cap S|]$, which is upper bounded by βS, followed by Chernoff bounds to bound the concentration of the median of $T \geq \lceil \ln(n/\delta)/\alpha \rceil$ trials about the mean μ. □

Algorithm 1 THF-WISH-LB$(w(x), n, T, \mathcal{H}^\beta_{(f,n,\rho)})$

for $i = 0, \cdots, n$ **do**
  for $t = 0, \cdots, T$ **do**
    Choose hash family parameters for $\mathcal{H}$ so that $E_{h\sim\mathcal{H}}[h(x)] \leq 2^{-i}$.
    Sample hash function $h_{i,t}(x) \sim \mathcal{H}$.
    $w_{i,t} \leftarrow \max_x w(x)$ subject to the constraint $h_{i,t}(x) = 1$.
  end for
  $M_i \leftarrow$ Median$(w_{i,1}, \ldots, w_{i,T})$
end for
return $\log\left(M_0 + \sum_{i=0}^{n-1} M_{i+1} 2^i\right)$

5 Experimental Evaluation

5.1 Partition Functions

We compare our partition function lower bounds with Sparse XORs (Ermon et al., 2014) on grid Ising models. These grid models are created on M^2 binary variables taking values in $\{-1, 1\}$ with unary potentials on each variable and binary potentials between all pairs of neighbors. The unary potentials are $\psi_i(x_i) = \exp(fx_i)$ and the binary potentials are $\psi_{i,j}(x_i, x_j) = \exp(w_{ij}x_ix_j)$. We consider mixed field Ising models where the coupling strength $w_{i,j}$ is drawn from $[-w, w]$ and the unary potentials f are in $\{0.1, 1\}$. Estimating the log-partition function in these models is challenging because the distribution becomes highly concentrated and multimodal as the coupling strength is increased, which presents a challenge for classic approaches like mean field and belief propagation.

We compare our method to SPARSE-WISH (Ermon et al., 2014) and Mean Field and Belief Propagation implemented in the libDAI inference algorithm library (Mooij, 2010), which also implements the Junction Tree algorithm we use as ground truth. Because our goal is to find computationally efficient hash families that still provide useful lower bounds on the partition function, we limit each MAP query in the inner loop of THF-WISH-LB and SPARSE-WISH to five minutes. Because the theoretically-motivated density of variables in the sparse parity constraints used by SPARSE-WISH was still too high for the solver to find informative solutions within five minutes, we relaxed the constraint density to 5% of the variables for all runs. The results of the partition function estimates for grids of varying coupling strengths are presented in Figure 2. We see that MAJORITY-WISH recovers the true log partition function almost exactly and performs as well as

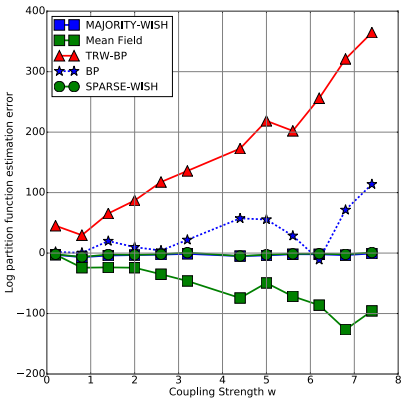

Figure 2. Results on grid Ising models of size 15×15 for different coupling strengths, with field strength of $f = 0.1$

SPARSE-WISH at all coupling strengths. Furthermore, *all* instances of MAJORITY-WISH for both field strengths terminated within three seconds and in 1.2 seconds on average, whereas the average time for the SPARSE-WISH MAP queries was 30 seconds and 60 seconds for field strengths 0.1 and 1, respectively.

5.2 Model Counting

Parity constraints are also difficult to reason about in the model counting setting, where the goal is to estimate the size of the set of solutions satisfying a set of constraints (Gomes et al., 2006a; 2007; 2006b; Chakraborty et al., 2013b). This is the canonical #-P complete counting problem. We focus on counting solutions to real-world CNF formulas that encode problems in a wide range of domains (latin squares, Langford's problem, logistic planning, and hardware verification). These problems are defined over hundreds to thousands of variables, and the theoretically-required density of parity constraints needed to find provable bounds on the model count are hopelessly intractable for modern SAT solvers.

This suggests a natural application for TRIBES functions, which are known to be among the most noise sensitive *monotone* functions, and consequently easier to optimize over. When parity constraints were chosen using the theoretically-motivated densities from (Ermon et al., 2014) we were unable to certify satisfiability for any of the problems within reasonable time frames, so we used the largest number of variables per constraint that reliably terminated within 15 minutes; in practice, this generated extremely sparse constraints on the order of 3 to 4 variables per constraint on average.

In Table 1 we compare the lower bounds obtained by THF-WISH-LB when using TRIBES functions and when using very low-density parity constraints. We highlight in bold the lower bounds and running times that are significantly better than the competing method's. Although there are some instances for which parity constraints obtain better lower bounds, the TRIBES-constrained lower bounds are largely comparable with the parity lower bounds, whereas the runtime t_{trib} is competitive with the XOR running times in all cases and 1 to 2 orders of magnitude faster on a large subset of the benchmark.

Table 1. Approximate lower bounds across several model counting problems. GT is ground truth (if known), LB_{trib} are the tribes lower bounds, LB_{xor} are the XOR lower bounds, t_{trib} is the average time (s) per tribes instance, and t_{xor} is the average time (s) per XOR instance. All counts are in $\log_{10}$. Further problem details are given in Table 2 in the Appendix.

INSTANCE	GT	LB_{trib}	LB_{xor}	t_{trib}	t_{xor}
LANG12	5	3	4	5	0
LANG15	7	6	6	0	0
LANG16	8	6	6	10	1
LANG19	11	6	7	1	1
LANG20	12	9	9	7	1
LANG23	15	10	10	1	1
LANG24	—	10	10	1	1
LANG27	—	9	11	1	1
LANG28	—	12	10	1	1
LS8	11	10	10	0	1
LS9	17	13	16	0	1
LS10	24	18	19	**1**	209
LS11	33	25	24	**28**	623
LS12	—	32	29	**34**	658
LS13	—	33	34	**3**	74
LS14	—	36	34	**12**	761
LS15	—	**39**	34	**51**	829
20RDR45	—	29	29	**1**	523
23RDR45	—	**27**	10	**7**	800
2BITMAX6	97	25	25	20	3
9SYMML	—	18	**19**	8	1
APEX75	—	40	**59**	17	24
FCLQ18	—	30	**44**	24	8
FCLQ20	—	35	**45**	19	9
VDAGRRCSW9	—	70	**99**	32	151

6 Conclusion

This paper presented two main contributions: (i) a procedure for creating compact hash families based on arbitrary boolean function "templates", and (ii) a theoretical connection between the template function's Fourier spectrum and the corresponding hash family's key statistical properties. Guided by this framework, we introduced two useful new hash families, TRIBES and MAJORITY, and showed that they yield comparable accuracy with significant improvements in runtime on both discrete integration and model counting problems.

Acknowledgements

We thank Ryan Williams for helpful discussions. This work was partially supported by the Future of Life Institute (grant 2015-143902) and Ford Motor Company (123213).

References

Achlioptas, Dimitris and Jiang, Pei. Stochastic integration via error-correcting codes. In *Proc. Uncertainty in Artificial Intelligence*, 2015.

Achlioptas, Dimitris, Mcsherry, Frank, and Schölkopf, Bernhard. Sampling techniques for kernel methods. In *Advances in Neural Information Processing Systems*, pp. 335–342, 2002.

Asteris, Megasthenis and Dimakis, Alexandros G. LDPC codes for discrete integration. Technical report, UT Austin, 2016.

Belle, Vaishak, Van den Broeck, Guy, and Passerini, Andrea. Hashing-based approximate probabilistic inference in hybrid domains. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence (UAI)*, 2015.

Ben-Or, Michael and Linial, Nathan. Collective coin flipping.

Blum, Avrim. Random projection, margins, kernels, and feature-selection. In *Proceedings of the 2005 international conference on Subspace, Latent Structure and Feature Selection*, pp. 52–68. Springer-Verlag, 2005.

Chakraborty, Supratik, Meel, Kuldeep, and Vardi, Moshe. A scalable and nearly uniform generator of SAT witnesses. In *Proc. of the 25th International Conference on Computer Aided Verification (CAV)*, 2013a.

Chakraborty, Supratik, Meel, Kuldeep, and Vardi, Moshe. A scalable approximate model counter. In *Proc. of the 19th International Conference on Principles and Practice of Constraint Programming (CP)*, pp. 200–216, 2013b.

Ermon, Stefano, Gomes, Carla P., Sabharwal, Ashish, and Selman, Bart. Embed and project: Discrete sampling with universal hashing. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2085–2093, 2013a.

Ermon, Stefano, Gomes, Carla P., Sabharwal, Ashish, and Selman, Bart. Optimization with parity constraints: From binary codes to discrete integration. In *Proc. of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013b.

Ermon, Stefano, Gomes, Carla P., Sabharwal, Ashish, and Selman, Bart. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proc. of the 30th International Conference on Machine Learning (ICML)*, 2013c.

Ermon, Stefano, Gomes, Carla P., Sabharwal, Ashish, and Selman, Bart. Low-density parity constraints for hashing-based discrete integration. In *Proc. of the 31st International Conference on Machine Learning (ICML)*, pp. 271–279, 2014.

Goldreich, Oded. Randomized methods in computation. *Lecture Notes*, 2011.

Gomes, Carla P., Sabharwal, Ashish, and Selman, Bart. Near-uniform sampling of combinatorial spaces using XOR constraints. In *Advances in Neural Information Processing Systems (NIPS)*, 2006a.

Gomes, Carla P., Sabharwal, Ashish, and Selman, Bart. Model counting: A new strategy for obtaining good bounds. In *Proc. of the 21st National Conference on Artificial Intelligence (AAAI)*, pp. 54–61, 2006b.

Gomes, Carla P., van Hoeve, Willem Jan, Sabharwal, Ashish, and Selman, Bart. Counting CSP solutions using generalized XOR constraints. In *Proc. of the 22nd National Conference on Artificial Intelligence (AAAI)*, 2007.

Hadjis, Stefan and Ermon, Stefano. Importance sampling over sets: A new probabilistic inference scheme. In *UAI*, 2014.

Hsu, Lun-Kai, Achim, Tudor, and Ermon, Stefano. Tight variational bounds via random projections and I-projections. In *Proc. of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

Ivrii, Alexander, Malik, Sharad, Meel, Kuldeep S, and Vardi, Moshe Y. On computing minimal independent support and its applications to sampling and counting. *Constraints*, pp. 1–18, 2015.

Jerrum, Mark and Sinclair, Alistair. The markov chain monte carlo method: An approach to approximate counting and integration. In *Approximation Algorithms for NP-hard Problems*, pp. 482–520. PWS Publishing, Boston, MA, 1997.

Jordan, Michael I., Ghahramani, Zoubin, Jaakkola, Tommi, and Saul, L.K. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Koller, Daphne and Friedman, Nir. *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.

Mooij, Joris M. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, 2010.

ODonnell, Ryan William. *Computational applications of noise sensitivity*. PhD thesis, Massachusetts Instt. of Tech., 2003.

Rajaraman, Anand and Ullman, Jeffrey David. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA, 2011. ISBN 1107015359, 9781107015357.

Sipser, Michael. A complexity theoretic approach to randomness. In *Proc. of the 15th ACM Symposium on Theory of Computing (STOC)*, pp. 330–335, 1983.

Stockmeyer, Larry. On approximation algorithms for #P. *SIAM Journal on Computing*, 14(4):849–861, 1985.

Vadhan, Salil. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 2011.

Valiant, Leslie and Vazirani, Vijay. NP is as easy as detecting unique solutions. *Theoretical Computer Sci.*, 47:85–93, 1986.

Wainwright, Martin J. and Jordan, Michael I. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Zhao, Shengjia, Chaturapruek, Sorathan, Sabharwal, Ashish, and Ermon, Stefano. Closing the gap between short and long xors for model counting. In *Proceedings of AAAI*, 2016.

Zhu, Michael and Ermon, Stefano. A hybrid approach for probabilistic inference using random projections. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 2039–2047, 2015.