6-16-2022

On Approximating Total Variation Distance

Arnab Bhattacharyya

Sutanu Gayen

Kuldeep S. Meel

Dimitrios Myrisiotis

A. Pavan

See next page for additional authors

Authors

Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, Dimitrios Myrisiotis, A. Pavan, and N. V. Vinodchandran

On Approximating Total Variation Distance

Arnab Bhattacharyya
School of Computing
National University of Singapore
arnabb@nus.edu.sg

Sutanu Gayen
CSE Department
Indian Institute of Technology Kanpur
sutanu@cse.iitk.ac.in

Kuldeep S. Meel
School of Computing
National University of Singapore
meel@comp.nus.edu.sg

Dimitrios Myrisiotis
School of Computing
National University of Singapore
dimitris@nus.edu.sg

A. Pavan
Department of Computer Science
Iowa State University
pavan@iastate.edu

N. V. Vinodchandran
School of Computing
University of Nebraska-Lincoln
vinod@cse.unl.edu

June 16, 2022

Abstract

Total variation distance (TV distance) is a fundamental notion of distance between probability distributions. In this work, we introduce and study the computational problem of determining the TV distance between two product distributions over the domain $\{0,1\}^n$. We establish the following results.

1. Exact computation of TV distance between two product distributions is #P-complete. This is in stark contrast with other distance measures such as KL, Chi-square, and Hellinger which tensorize over the marginals.
2. Given two product distributions P and Q with marginals of P being at least $1/2$ and marginals of Q being at most the respective marginals of P, there exists a fully polynomial-time randomized approximation scheme (FPRAS) for computing the TV distance between P and Q. In particular, this leads to an efficient approximation scheme for the interesting case when P is an arbitrary product distribution and Q is the uniform distribution.

We pose the question of characterizing the complexity of approximating the TV distance between two arbitrary product distributions as a basic open problem in computational statistics.

1 Introduction

An overarching theme in modern machine learning is the use of probability distributions to describe data. Datasets are often modeled by high-dimensional distributions with additional structure reflecting correlations among the features. In this context, a basic problem is *distance approximation*: given two distributions P and Q, compute $\rho(P,Q)$ for a distance measure ρ. For example, P and Q could be the the outputs of two unsupervised learning algorithms, and one could ask how much they differ. As another example, a key component of generative adversarial

networks [GPAM+14, ACB17] is the discriminant which approximates the distance between the model and the true distributions.

An important notion of distance between probability distributions is the *total variation (TV) distance* or the *statistical difference*. Given two distributions P and Q on a finite domain $\mathcal{D}$, their TV distance is defined as follows:

$$d_{\mathrm{TV}}(P,Q) = \frac{1}{2}\sum_{x\in\mathcal{D}} |P(x) - Q(x)| = \max_{S\subseteq\mathcal{D}} \{P(S) - Q(S)\}.$$

The total variation distance arguably is the most fundamental distance measure between two probability distributions. It has a physical interpretation due to the second equality above: TV distance between two distributions is the maximum bias of any event with respect to two distributions. It satisfies many mathematically desirable properties: it is bounded, is a metric, is invariant with respect to bijections, and it satisfies the data processing inequality; that makes it desirable to work with. Because of these reasons the total variation distance is one of the main distance measures in a wide range of areas including probability and statistics, machine learning, and information theory. In theoretical computer science, the notion plays an important role in several areas, e.g., property testing [Can22], approximate counting [Jer03], density estimation [DL01], pseudorandomness [Yao82, NW94] and cryptography [GM84].

In this work, we study total variation distance from a *computational* perspective. Given two distributions P and Q over a finite domain $\mathcal{D}$, how hard is it to compute $d_{\mathrm{TV}}(P,Q)$? If P and Q are explicitly specified by the evaluation of their mass functions at all points of the domain $\mathcal{D}$, the problem becomes trivial as d_{TV} can obviously be computed in $O(|\mathcal{D}|)$ time. The computational problem becomes interesting for high-dimensional structured distributions with succinct representations.

Arguably, the simplest model for high-dimensional distributions is a product of Bernoulli trials.[1] Such a product distribution P over $\{0,1\}^n$ can be described by n parameters $p_1,\ldots,p_n$ where each $p_i \in [0,1]$ is the probability that the i'th coordinate equals 1. Moreover, such a model serves as a great testing ground for various intuitions regarding computational statistics, due to its ubiquity and simplicity. In this work, we focus on the computational problem of finding the total variation distance between two product distributions P and Q, given their descriptions $\langle p_1,\ldots,p_n\rangle$ and $\langle q_1,\ldots,q_n\rangle$. We hope that our work will be a first step towards a better understanding of the computational nature of TV distance (and other distance measures) in more general settings that are of practical relevance.

1.1 Our contributions

We investigate the computational problem DistProduct of computing the total variation distance between two product distributions formally defined as follows.

Definition 1.1. DistProduct: Given two probability distributions P and Q over $\{0,1\}^n$ that are products of Bernoulli distributions presented by their marginal probabilities, compute $d_{\mathrm{TV}}(P,Q)$.

Our first technical result is that DistProduct is #P-complete and hence it is unlikely to be tractable since a Turing machine with access to #P-oracle contains the entire polynomial hierarchy [Tod91].

Theorem 1.2. DistProduct *is* #P*-complete.*

[1] In the rest of the paper we use the term *product distributions* to refer to products of Bernoulli trials.

The #P-completeness of DistProduct is in stark contrast to certain other distance measures such as Hellinger, Chi-square and KL which *tensorize* over marginals the in the sense that they are easily expressible in terms of those of the one-dimensional marginals. Thus, the hardness result formalizes the intuition that total variation distances are typically hard to compute in practice.

Theorem 1.2 explains the computational hardness of exact computation of TV distance. Often, in many practical applications, approximation suffices. While, for many #P-hard problems, even computing an approximation is hard (for example for problems such as counting solutions of CNF formula, where the problem of checking the number of solutions is nonzero or not is NP-hard), there are important counting problems for which this is not the case. Well known examples include counting number of solutions of a DNF formula, counting the number of perfect matchings in a graph, and counting the number of subsets that satisfy a Knapsack constraint.

This leads us to the following significant open question.

Question 1.3. *What is the computational complexity of the problem of approximating the TV distance between two product distributions to within* $1 \pm \varepsilon$ *relative error?*

Question 1.3 addresses the status of a basic problem in computational statistics and we feel that the lack of prior work on it is surprising. Indeed, in our view, the identification of Question 1.3 as an open problem is one of the main conceptual contributions of this work.

We provide some partial answers to the above question. We show that under certain structural assumption about the input distributions, there is an efficient fully polynomial randomized approximation scheme (FPRAS) for DistProduct.

Theorem 1.4. *There is a randomized polynomial-time algorithm that on input* ε, δ *and probability vectors* $\langle p_i, \cdots, p_n \rangle$ *and* $\langle q_i, \cdots, q_n \rangle$ *that defines two binary product distributions* P *and* Q*, where* $\frac{1}{2} \leq p_i < 1$ *and* $0 < q_i \leq p_i$*, outputs a number* v *so that* $(1-\varepsilon)d_{\mathrm{TV}}(P,Q) \leq v \leq (1+\varepsilon)d_{\mathrm{TV}}(P,Q)$ *with probability* $\geq 1-\delta$*.*

An interesting corollary of this result that the distance between an arbitrary product distribution and the uniform distribution can be efficiently approximated.[2]

Theorem 1.4 is quite surprising in our view. If P and Q were uniform distributions over two unit-volume convex bodies in $\mathbb{R}^n$, then $d_{\mathrm{TV}}(P,Q)$ corresponds to the volume of the symmetric difference between P and Q. The symmetric difference between two convex bodies is non-convex, so there is no reason to expect a polynomial time algorithm for this problem [Vem21]. The additional structure in Theorem 1.4 of product distributions over $\{0,1\}^n$ enables the existence of an efficient algorithm.

Using similar techniques, we can also design FPRASs for certain other settings such as when the number of distinct q_i's. In particular, we establish the following theorem.

Theorem 1.5. *There is a randomized polynomial-time algorithm that on input* ε, δ *and probability vectors* $\langle p_i, \cdots, p_n \rangle$ *and* $\langle q_i, \cdots, q_n \rangle$*, where* p_i *are arbitrary and* q_i *takes values from a set of constant cardinality, outputs a number* v *so that* $(1-\varepsilon)d_{\mathrm{TV}}(P,Q) \leq v \leq (1+\varepsilon)d_{\mathrm{TV}}(P,Q)$ *with probability* $\geq 1-\delta$*.*

While we are unable to settle Question 1.3, we establish a hardness result for approximating TV distance for the class of distributions that can be modeled by sparse *Bayesian networks*. Thus if we slightly generalize product distributions, then TV distance approximation is hard.

[2] While Theorem 1.4 needs p_i to be at least $1/2$, since $q_i s$ are $1/2$ for the uniform case, marginal Bernoulli distributions of an arbitrary product distribution can be flipped if necessary to get the required condition.

Definition 1.6. NeqBayes: Given two probability distributions P and Q that are defined by Bayes nets, decide whether $d_{\mathrm{TV}}(P, Q) \neq 0$ or not.

Theorem 1.7. NeqBayes *is* NP*-complete even over Bayes nets distributions of in-degree 2.*

The above theorem implies that it is NP-hard to multiplicatively approximate TV distance between two Bayes net distributions.

1.2 Related work

While the problem of multiplicatively approximating total variation distance between distributions appears to have escaped much attention from the algorithms community, the problem of additively approximating TV distance for succinctly represented high dimensional distributions have been discovered to be fundamentally related to cryptography. Sahai and Vadhan [SV03] in a seminal work established that the problem of deciding whether given two distributions samplable by Boolean circuits are close or far apart in TV distance captures the computational difficulty of the complexity class SZK (Statistical Zero Knowledge). The class SZK is fundamental to cryptography and is believed to be computationally hard. Subsequent works captured variations of this theme and provided completeness results for other Zero Knowledge classes using similar computational problems. In particular, Goldreich et al. [GSV99] showed that the problem of deciding whether a distribution samplable by a Boolean circuit is close or far from the uniform distribution is complete for the complexity class NISZK (Non Interactive Statistical Zero Knowledge).

Complementing the above-mentioned hardness results, it has also been recently observed (see [CDKS18, Appendix C.2] and [BGMV20]) that if we restrict the distribution so that in addition to sampling, we can also compute the probability mass at a given point, then an additive approximation of TV distance becomes feasible. In particular, this implies efficient algorithms for *additively* approximating the TV distance between structured high dimensional distributions such as Bayesian networks, Ising models, and multivariate Gaussians. More generally, there has been a large body of works on algorithms for *testing* (which is closely related to additive approximation in certain settings) distributions in TV distance from the theoretical computer science community for the last 20 years; see [Can22] and the references therein.

Other related works include that of Cortes et al. [CMR07], Lyngsø et al. [LP02] and Kiefer [Kie18]. Their works focused on finding the complexity of computing the TV distance between two hidden Markov models (equivalently labelled Markov chains) culminating in the results that it is undecidable whether the distance is more than a threshold and it is #P-hard to additively approximate it. It is well-known that differential privacy with zero privacy budget captures TV distance. Murtagh and Vadhan [MV18] considered the problem of optimal composition of several differentially private protocols which corresponds to TV distance computation between binary product distributions as a special case. They showed computing this optimal composition is #P-hard (using a hard instance that corresponds to non-zero privacy budgets), and gave an approximation algorithm for this problem using Dyer's approximation algorithm for #Knapsack [Dye03].

From a technical standpoint, our work uses results from the literature on approximation algorithms for the problem #Knapsack (the problem of counting the number of sets that satisfy Knapsack constraint). Dyer et al. [DFK+93] gave a subexponential-time approximation algorithm for #Knapsack. Later, Morris and Sinclair [MS04] designed a fully polynomial randomized approximation scheme for it. Subsequently, Dyer [Dye03] re-proved their result by utilizing simpler techniques which we also use in this paper. More recently, Stefankovic, Vempala, and Vigoda [SVV12] and independently Gopalan, Klivans, and Meka [GKM10] gave *deterministic* fully polynomial-time approximation schemes for #Knapsack.

1.3 Technical overview

We briefly explain here some of the technical ingredients of our main results, Theorem 1.2 and Theorem 1.4.

Proof sketch of Theorem 1.4. This theorem is proved in two steps. First, we introduce a problem #PMFEQUALS and show that it is#P-hard by a reduction from #SUBSETSUM. #PMFEQUALS is the following problem: Given a product distribution with parameters $p_1, \ldots, p_n$, and a number v: count the number of strings $x \in \{0,1\}^n$ such that $P(x) = v$ ($|\{x \mid P(x) = v\}|$). In the second step, we reduce #PMFEQUALS to the problem of computing TV distance between product distributions. The core of our reduction, we construct distributions $P', Q', \hat{P}, \hat{Q}$ such that #PMFEQUALS is a function of $d_{\mathrm{TV}}(P', Q')$ and $d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)$. We elaborate on this idea below.

Let $p_1, \ldots, p_n$ and v be the numbers in an arbitrary instance of #PMFEQUALS. In our proof, we distinguish between two cases depending on whether $v < 2^{-n}$ or $v \geq 2^{-n}$. Let us assume for now that $v < 2^{-n}$ (the other case is similar).

We define distributions $\hat{P}$ and $\hat{Q}$ over $\{0,1\}^{n+1}$ where $\hat{P} = \otimes_i \mathrm{Bern}(\hat{p}_i)$ and $\hat{Q} = \otimes_i \mathrm{Bern}(\hat{q}_i)$ as follows: $\hat{p}_i := p_i$ for $i \in [n]$ and $\hat{p}_{n+1} := 1$; and $\hat{q}_i := 1/2$ for $i \in [n]$ and $\hat{q}_{n+1} := v2^n$. We next define two additional product distributions P' and Q' over $\{0,1\}^{n+2}$. The new distributions P' and Q' are such that $p'_i := p_i$ for $i \in [n]$, $p'_{n+1} := 1$, and $p'_{n+2} := \frac{1}{2} + \beta$ and $q'_i := \frac{1}{2}$ for $i \in [n]$, $q'_{n+1} := v2^n$, and $q'_{n+2} := \frac{1}{2} - \beta$. where β is as an appropriately chosen quantity (see Claim 3.5).

We then express the value of an instance of #PMFEQUALS as a function of $d_{\mathrm{TV}}\left(P', Q'\right)$ and $d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)$. In particular, we establish that

$$|\{x \mid P(x) = v\}| = \frac{1}{2\beta v}\left(d_{\mathrm{TV}}\left(P', Q'\right) - d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)\right).$$

This proves that computing TV distance is #P-hard in this case.

Proof sketch of Theorem 1.4. This theorem is established in two steps. First we reduce the problem of computing TV distance of product distributions to a "#KNAPSACK-like" problem. We then design an approximation algorithm for the latter problem.

We will first sketch the ideas behind the first redution. This is done by an algebraic manipulation of the expression that gives TV distance, combined with a sum reorganization idea of de Colnet and Meel [dCM19]. Using this manipulation we arrive at a #KNAPSACK-like instance.

Consider the following derivation of TV distance:

$$\begin{aligned} d_{\mathrm{TV}}(P, Q) &= \sum_{S \subseteq [n]} \max\left(0, \prod_{i \in S} p_i \prod_{i \notin S} (1 - p_i) - \prod_{i \in S} q_i \prod_{i \notin S} (1 - q_i)\right) \\ &= \sum_{S \subseteq [n]} \max\left(0, \prod_{i \in [n]} (1 - p_i) \prod_{i \in S} \frac{p_i}{1 - p_i} - \prod_{i \in [n]} q_i \prod_{i \notin S} \frac{1 - q_i}{q_i}\right) \\ &= \sum_{S \subseteq [n]} \max(0, R_1 \cdot W_S - R_2 \cdot K_S) \end{aligned}$$

where $R_1 := \prod_{i \in [n]} (1 - p_i)$ and $R_2 := \prod_{i \in [n]} q_i$ are constants, and $W_S := \prod_{i \in S} \frac{p_i}{1 - p_i}$, $K_S := \prod_{i \notin S} \frac{1 - q_i}{q_i}$.

If ε is the FPRAS accuracy error, we are able to show that (a normalized) variant of the expression above lies in some set $[1, U]$, which we perceive as $[1, U] = \bigcup_{i=1}^{u} \left[(1 + \varepsilon)^{i-1}, (1 + \varepsilon)^{i+1}\right)$,

for appropriate values of U and u. Then we show that estimating the number of sets $S \subseteq [n]$ such that $Y_S := \max(0, R_1 \cdot W_S - R_2 \cdot K_S)/m$ lies in each of the intervals $\left[(1+\varepsilon)^{i-1}, (1+\varepsilon)^{i+1}\right)$ (here, m is a normalization constant) yields an approximation to the TV distance. Next, by a reorganization idea of de Colnet and Meel [dCM19], we show that counting the number of S for which $Y_S \in \left[(1+\varepsilon)^{j-1}, (1+\varepsilon)^{j}\right)$ is equivalent to counting the number of sets S such that $Y_S \geq (1+\varepsilon)^j$, where $j \in [u]$. Now, by a sequence of calculations, we are able to reformulate this problem as the following #KNAPSACK-like problem.

> Given two sets $\{x_1, \cdots, x_n\}$ and $\{y_1, \cdots, y_n\}$ and constants A and B, find the number of sets S for which $A\exp\left(\sum_{i\in S} x_i\right) + B\exp\left(\sum_{i\in S} x_i + y_i\right) \leq 1$ holds.

When Q is the uniform distribution, then K_S does not depend on S. In this setting, the inequality above becomes $A\exp\left(\sum_{i\in S} x_i\right) + B'\exp\left(\sum_{i\in S} x_i\right) \leq 1$ (for some constant B'), or $\sum_{i\in S} x_i \leq \log(1/(A+B'))$. Finding the number of sets S satisfying this constraint is exactly the #KNAPSACK[3] problem! The work of Dyer [Dye03] (see also [GKM10]) designed an approximation algorithm for the #KNAPSACK problem. Thus, we can estimate the number of solutions for the case when Q is uniform yielding an FPRAS for this special case.

For the more general case that we are interested in, the above problem behaves very differently from the #KNAPSACK problem and a host of technical challenges arise when we attempt to build upon the ideas from Dyer [Dye03]. At the heart of Dyer's idea, there is a discretization argument that transforms the standard pseudo-deterministic algorithm for #KNAPSACK into an FPRAS. This discretization argument basically changes the weights of the Knapsack items (x_i and y_i here) so that a respective, discretized, Knapsack problem is solvable in polynomial time. Then, Dyer shows that this set of solutions $\mathcal{S}'$ of the discretized problem extends the set of original solutions $\mathcal{S}$, but this blow-up is modest, that is, $S \subseteq S'$ and $\frac{|\mathcal{S}|}{|\mathcal{S}'|}$ is at most $n+1$. Dyer proves the bound by prescribing a function $f : \mathcal{S}' \to \mathcal{S}$ so that for any $y \in S$, size of the preimage $f^{-1}(y)$ is at most $n+1$ (the mapping removes the index of $S \in \mathcal{S}'$ that induces the heaviest element in the x_i sequence.) Once this bound is established Dyer uses the standard dynamic programming algorithm to sample from $\mathcal{S}'$ and then estimate $|\mathcal{S}|$ by a standard Monte-Carlo argument.

For our case, observe that while $A'\exp\left(\sum_{i\in S} x_i\right) \leq 1$ reduces to #KNAPSACK but having two terms, i.e., $A\exp\left(\sum_{i\in S} x_i\right) + B\exp\left(\sum_{i\in S} x_i + y_i\right) \leq 1$, does not reduce to #KNAPSACK (or its variant). From a technical perspective, Dyer's approach of discretization does not give us the desired mapping between the sets of solutions of the discretized problem and the original problem. Our case, however, displays monotonicity, i.e., if $A\exp\left(\sum_{i\in S} x_i\right) + B\exp\left(\sum_{i\in S} x_i + y_i\right) \leq 1$, then it is also the case that for all $T \subseteq S$, we have $A\exp\left(\sum_{i\in T} x_i\right) + B\exp\left(\sum_{i\in T} x_i + y_i\right) \leq 1$. Accordingly, one naturally wonders whether it is possible to discretize and employ Monte-Carlo sampling?

Our main technical contribution here is that we managed to accomplish exactly this, via a new discretization procedure we call *adaptive discretization*. The primary insight in our approach is to perform discretization that, *in a sense*, depends on each subset $S \subseteq [n]$. Such a discretization can now allow us to bound $\frac{|\mathcal{S}|}{|\mathcal{S}'|}$ by some polynomial. The dependence on the subset S, however, presents a technical hurdle: after all, S is not known a priori and there are exponentially many possible S. The key insight in our approach is to employ discretization that only depends on the $(\max_{i\in S} x_i, \max_{i\in S} y_i)$, and accordingly all S for which $(\max_{i\in S} x_i, \max_{i\in S} y_i)$ are equal belong to the same class. This gives rise to a sampling process that partitions the space $2^{[n]}$ into quadratically many classes which yield solutions of some fixed maximum value in x_i and y_i, and after a class is chosen, we proceed to sample a set S from that class.

[3] Given a sequence of weights $w_1, \cdots, w_n$ and a capacity parameter W, count the number of sets $S \subseteq [n]$ so that $\sum_{i\in S} w_i \leq W$.

1.4 Organization

We sketch some background information in Section 2. The rest of the paper is organized as follows: In Section 3, we prove Theorem 1.2 and Theorem 1.7; Theorem 1.4 and Theorem 1.5 are proved in Section 4.

2 Preliminaries

Finite precision. Since the probability distributions are specified by real-valued parameters, we should be explicit about how to model computations on them using finite, discrete machines. We assume throughout that the real-valued parameters are actually rationals of the form 2^{-k} where k is bounded by a polynomial in the scaling parameter n. Note that the total variation distance between two product distributions with rational parameters is also rational.

Bayes nets. Given a directed acyclic graph (DAG) over n nodes, a probability distribution P over $\{0,1\}^n$ is said to be *Markov with respect to* G if P factorizes according to G; we will also say that P *has structure* G. A DAG G is said to have *degree* d, if every node has at most d parents. Finally, a distribution P over $\{0,1\}^n$ is a *degree-*d *Bayes net* if P is Markov with respect to some degree-d DAG.

FPRAS. We will make use of the following notion of randomized (relative) approximation algorithms.

Definition 2.1. An FPRAS is a randomized algorithm which takes an instance of an optimization (or counting) problem and a parameter $\varepsilon > 0$ and, in time polynomial in n and $1/\varepsilon$ (whereby n is the size of the instance in question), produces a solution that has a high probability of being within a factor $(1 \pm \varepsilon)$ of optimal.

Read-once branching programs and approximate counting. A (W, n)-branching program is a (standard) branching program of width W over n Boolean input variables. A read-once branching program (ROBP) is a branching program whereby each input variable is accessed only once. The following notion of small-space sources was introduced by Kamp et al. [KRVZ06], and will be useful in our work in the context of encoding solutions to Knapsack instances that have fixed Hamming weight.

Definition 2.2 ([KRVZ06])**.** A width w small-space source is described by a (w, n)-branching program D with an additional probability distribution p_v on the outgoing edges associated with vertices $v \in D$. Samples from the source are generated by taking a random walk on D according to p_v's and outputting the labels of the edges traversed.

Gopalan, Klivans, and Meka [GKM10] used the concept of *small-space sources* in counting Knapsack solutions of a fixed Hamming weight.

Theorem 2.3 ([GKM10])**.** *Given a Knapsack instance $(a, b) \in \mathbb{Z}_+^n$ of total weight $W = \sum ia_i + b$, $\varepsilon > 0$ and $r \in [n]$ we can in deterministic time $O(n^3 r(r + \log W)/\varepsilon)$ compute an ε-relative error approximation for the number of solutions to the Knapsack instance of Hamming weight exactly r.*

3 The hardness of computing TV distance

3.1 Products of Bernoulli distributions

We first formally define a the problem of counting the number of points in the sample space that has a certain probability.

Definition 3.1. #PMFEQUALS: Given probability vector $\langle p_1, \ldots, p_n \rangle$ where $p_i \in [0,1]$ and a number v, count the number of $x \in \{0,1\}^n$ such that $P(x) = v$ where P is the product distribution described by $\langle p_1, \ldots, p_n \rangle$.

Definition 3.2. #SUBSETSUM: Given numbers $a_1, \ldots, a_n$, and and a target T. Compute the number of subsets $S \subseteq [n]$ such that $\sum_{i \in S} a_i = T$.

Theorem 3.3. #SUBSETSUM *is* #P*-hard.*

Theorem 3.4. #PMFEQUALS *is* #P*-complete.*

Proof. We will reduce #SUBSETSUM to #PMFEQUALS. The result then will follow from Theorem 3.3. Let $a_1, \ldots, a_n$, and T be the numbers of an arbitrary #SUBSETSUM instance, namely I_S. We will create a #PMFEQUALS instance I_P that has the same number of solutions as I_S.

To this end, let $p_i := \frac{2^{a_i}}{1+2^{a_i}}$ for every i and $v := \frac{2^T}{\prod_{i \in [n]}(1-p_i)}$. Now observe that

$$\begin{aligned}
\sum_{i \in S} a_i = T &\Leftrightarrow \sum_{i \in S} \log \frac{p_i}{1-p_i} = \log \frac{v}{\prod_{i \in [n]} 1-p_i} \\
&\Leftrightarrow \log \prod_{i \in S} \frac{p_i}{1-p_i} = \log \frac{v}{\prod_{i \in [n]} 1-p_i} \\
&\Leftrightarrow \prod_{i \in S} \frac{p_i}{1-p_i} = \frac{v}{\prod_{i \in [n]} 1-p_i} \\
&\Leftrightarrow \prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) = v \\
&\Leftrightarrow P(x) = v,
\end{aligned}$$

where x is such that $x_i = 1$ iff $i \in S$. This completes the proof. □

Theorem 1.2. DISTPRODUCT *is* #P*-complete.*

Proof. Membership in #P, while not trivial, is standard and is given in Theorem 3.8 for the more general case of Bayes net distributions.

For establishing the hardness, we will reduce #PMFEQUALS to DISTPRODUCT. The result then will follow from Theorem 3.4. In what follows, let $p_1, \ldots, p_n$ and v be the numbers in an arbitrary instance of #PMFEQUALS.

We distinguish between two cases depending on whether $v < 2^{-n}$ or $v \geq 2^{-n}$.

($v < 2^{-n}$). First, we construct two distributions $\hat{P} = \mathrm{Bern}(\hat{p}_1) \otimes \cdots \otimes \mathrm{Bern}(\hat{p}_n) \otimes \mathrm{Bern}(\hat{p}_{n+1})$ and $\hat{Q} = \mathrm{Bern}(\hat{q}_1) \otimes \cdots \otimes \mathrm{Bern}(\hat{q}_n) \otimes \mathrm{Bern}(\hat{q}_n)$ over $\{0,1\}^{n+1}$ as follows: $\hat{p}_i := p_i$ for $i \in [n]$ and $\hat{p}_{n+1} := 1$; and $\hat{q}_i := 1/2$ for $i \in [n]$ and $\hat{q}_{n+1} := v2^n$. We have that

$$d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right) = \sum_x \max\left(0, \hat{P}(x) - \hat{Q}(x)\right)$$

$$
\begin{aligned}
&= \sum_x \max\left(0, P(x) - \frac{1}{2^n} v 2^n\right) \\
&= \sum_x \max(0, P(x) - v) \\
&= \sum_{x : P(x) > v} (P(x) - v)\,.
\end{aligned}
$$

We now define two more distributions P' and Q' over $\{0,1\}^{n+2}$, by making use of the following claim.

Claim 3.5. *There exists a $\beta \in (0,1)$ such that the following hold, for all x:*

- *If $P(x) < v$, then $P(x)\left(\frac{1}{2} + \beta\right) < v\left(\frac{1}{2} - \beta\right)$;*
- *if $P(x) > v$, then $P(x)\left(\frac{1}{2} - \beta\right) > v\left(\frac{1}{2} + \beta\right)$.*

Proof. It is possible to find such a β because we can assume that the numbers $\{P(x)\}_x$ and v are represented with some finite precision of $q(n)$ bits (for a polynomial q). In particular, we can define β to be equal to $2^{-100q(n)}$. Then, we can set the precision of our algorithm to be $p(n) := 100q(n)$ bits; see Section 2. (Claim 3.5) □

The new distributions P' and Q' are such that $p'_i := p_i$ for $i \in [n]$, $p'_{n+1} := 1$, and $p'_{n+2} := \frac{1}{2} + \beta$ and $q'_i := \frac{1}{2}$ for $i \in [n]$, $q'_{n+1} := v2^n$, and $q'_{n+2} := \frac{1}{2} - \beta$. where β is as in Claim 3.5. We have the following claim which implies that #PMFEQUALS is in FP if computing TV distance is in FP. Thus, computing TV distance is #P-hard.

Claim 3.6. *It is the case that*

$$
|\{x \mid P(x) = v\}| = \frac{1}{2\beta v}\left(d_{\mathrm{TV}}(P', Q') - d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)\right).
$$

Proof. We have that $d_{\mathrm{TV}}(P', Q')$ is equal to

$$
\begin{aligned}
&\sum_{x : P(x) \geq v} \max\left(0, P(x)\left(\frac{1}{2} + \beta\right) - v\left(\frac{1}{2} - \beta\right)\right) \\
&\qquad + \sum_{x : P(x) > v} \max\left(0, P(x)\left(\frac{1}{2} - \beta\right) - v\left(\frac{1}{2} + \beta\right)\right) \\
&= 2\beta v\,|\{x \mid P(x) = v\}| + \sum_{x : P(x) > v} (P(x) - v) \\
&= 2\beta v\,|\{x \mid P(x) = v\}| + d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)
\end{aligned}
$$

which gives us the desired claim. (Claim 3.6) □

($v \geq 2^{-n}$). First, let us define distributions $\hat{P} = \mathrm{Bern}(\hat{p}_1) \otimes \cdots \otimes \mathrm{Bern}(\hat{p}_n)$ and $\hat{Q} = \mathrm{Bern}(\hat{q}_1) \otimes \cdots \otimes \mathrm{Bern}(\hat{q}_n)$ as follows: $\hat{p}_i := p_i$ for $i \in [n]$, $\hat{p}_{n+1} := \frac{1}{v2^n}$; and $\hat{q}_i := \frac{1}{2}$ for $i \in [n]$, and $\hat{q}_{n+1} := 1$. We have that

$$
\begin{aligned}
d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right) &= \sum_x \max\left(0, \hat{P}(x) - \hat{Q}(x)\right) \\
&= \sum_x \max\left(0, P(x)\,\frac{1}{v2^n} - \frac{1}{2^n}\right) + \sum_x \max\left(0, P(x)\left(1 - \frac{1}{v2^n}\right)\right)
\end{aligned}
$$

$$= \sum_x \max\left(0, P(x)\,\frac{1}{v2^n} - \frac{1}{2^n}\right) + 1 - \frac{1}{v2^n}.$$

As earlier, we define two more distributions P' and Q', by making use of Claim 3.5 again. The new distributions P' and Q' are such that $p'_i := p_i$ for $i \in [n]$, $p'_{n+1} := \frac{1}{v2^n}$, $p'_{n+2} := \frac{1}{2} + \beta$ and $q'_i := q_i$ for $i \in [n]$, $q'_{n+1} := 1$, and $q'_{n+2} := \frac{1}{2} - \beta$. We have that the following claim which implies that #PMFEQUALS is in FP if computing TV distance is in FP.

Claim 3.7. *It is the case that*

$$|\{x \mid P(x) = v\}| = \frac{2^{n-1}}{\beta}\left(d_{\mathrm{TV}}\left(P', Q'\right) - d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)\right).$$

Proof. We have that

$$\begin{aligned}
d_{\mathrm{TV}}\left(P', Q'\right) &= \sum_x \max\left(0, P'(x) - Q'(x)\right) \\
&= \sum_x \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2} + \beta\right) - \frac{1}{2^n}\left(\frac{1}{2} - \beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2} - \beta\right) - \frac{1}{2^n}\left(\frac{1}{2} + \beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\left(1 - \frac{1}{v2^n}\right)\left(\frac{1}{2} + \beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\left(1 - \frac{1}{v2^n}\right)\left(\frac{1}{2} - \beta\right)\right) \\
&= \sum_x \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2} + \beta\right) - \frac{1}{2^n}\left(\frac{1}{2} - \beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2} - \beta\right) - \frac{1}{2^n}\left(\frac{1}{2} + \beta\right)\right) \\
&\quad + \sum_x \max\left(0, P(x)\left(1 - \frac{1}{v2^n}\right)\right) \\
&= \sum_{x:P(x)\geq v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2} + \beta\right) - \frac{1}{2^n}\left(\frac{1}{2} - \beta\right)\right) \\
&\quad + \sum_{x:P(x)>v} \max\left(0, P(x)\,\frac{1}{v2^n}\left(\frac{1}{2} - \beta\right) - \frac{1}{2^n}\left(\frac{1}{2} + \beta\right)\right) + \left(1 - \frac{1}{v2^n}\right),
\end{aligned}$$

by Claim 3.5, or

$$\begin{aligned}
&= 2\beta\frac{1}{2^n}\,|\{x \mid P(x) = v\}| + \sum_x \max\left(0, P(x)\,\frac{1}{v2^n} - \frac{1}{2^n}\right) + \left(1 - \frac{1}{v2^n}\right) \\
&= \frac{\beta}{2^{n-1}}\,|\{x \mid P(x) = v\}| + d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)
\end{aligned}$$

or

$$|\{x \mid P(x) = v\}| = \frac{2^{n-1}\left(d_{\mathrm{TV}}(P', Q') - d_{\mathrm{TV}}\left(\hat{P}, \hat{Q}\right)\right)}{\beta}.$$ (Claim 3.7) □

This concludes the proof. □

3.2 Distributions defined by Bayes nets

We first consider the DistBayes problem where given two probability distributions P and Q defined by Bayes nets (see Section 2), the objective is to output $d_{\mathrm{TV}}(P,Q)$. The #P-hardness of DistBayes is immediate from Theorem 1.2, but we reprove it because (i) the proof is simpler when the distributions are allowed to have degree at least 2 as Bayes nets, and (ii) the proof can be easily adapted to prove Theorem 1.7.

Theorem 3.8. DistBayes *is* #P*-complete.*

Proof. We separately prove membership in #P and hardness for #P.

Membership in #P. Let P and Q be two Bayes net distributions over a finite alphabet. The goal is to design a nondeterministic machine $\mathcal{N}$ so that the number of accepting paths of $\mathcal{N}$ (normalized by an appropriate quantity) equals $d_{\mathrm{TV}}(P,Q)$. We will assume that the probabilities specified in the CPTs of Bayes nets for P and Q are fractions. Let M be the product of the denominators of all the probabilities in the CPTs. The non-deterministic machine $\mathcal{N}$, first guesses an element i in the sample space of P and Q and computes $|P(i)-Q(i)|$ and accepts on $M\times|P(i)-Q(i)|$. It is easy to see that $d_{\mathrm{TV}}(P,Q) = M\times$ number of accepting paths of $\mathcal{N}$.

Hardness for #P. We will show that the problem of computing the number of satisfying assignments of a CNF formula reduces to DistBayes. Let F be a CNF formula with n variables $x_1,\ldots,x_n$ and m gates $y_1,\ldots,y_m$ where Y is topologically sorted with Y_n being the output gate. We will define two Bayes net distributions both on the same DAG G. Intuitively, G is the digraph of F.

The vertex set of G is split into $\mathcal{X}$ of n nodes with node X_i corresponds to variable x_i, $\mathcal{Y}$ of m nodes with each node Y_i corresponds to gate y_i of F and another node Z. So totally there are $n+m+1$ nodes. There is directed edge from V_i to V_j if the gate/variable corresponding to V_i is an input to V_j. Finally there is an edge from Y_m to Y (Essentially G is the directed tree formed by the formula F).

The distributions P and Q on G are given by CPTs defined as follows: They differ only in the last variable Z: For node X_i, it is a uniform random bit. For each Y_i, the CPT is deterministic: For each of the setting of the parents Y_j, Y_k the variable Y_i takes the value of the gate y_i for that setting of its inputs y_j, y_k. Finally Z in the first distribution P is a random bit and Z in the second distribution Q is defined by the value of Y_n ORed with a random bit.

Note that for the formula F computes a Boolean function on the input variables: Let f be this function. We will in fact extend f to $\{0,1\}^m$ to also include the values of the intermediate gates. With this notation for any binary string XYZ of length $n+m+1$, both P and Q has a probability 0 if $Y\neq f(X)$. Thus $d_{\mathrm{TV}}(P,Q)$ can be written as

$$\begin{aligned} 2\cdot d_{\mathrm{TV}}(P,Q) &= \sum_{X,f(X),Z} |P-Q| \\ &= \sum_{X\in A} |P-Q| + \sum_{X\in R} |P-Q|. \end{aligned}$$

In the above A is the set of assignments of F that evaluates to True and R is the set of assignments that evaluates to False. We will compute each sum separately.

$$\sum_{X\in A} |P-Q| = \sum_{X\in A, Z=0} |P-Q| + \sum_{X\in A, Z=1} |P-Q|$$

$$= \sum_{X\in A, Z=0} |0 - \frac{1}{2^{n+1}}| + \sum_{X\in A, Z=1} |\frac{1}{2^n} - \frac{1}{2^{n+1}}|$$
$$= \frac{|A|}{2^n}.$$

Now we will consider the second sum.

$$\sum_{X\in R} |P - Q| = \sum_{X\in A, Z=0} |P - Q| + \sum_{X\in R, Z=1} |P - Q|$$
$$= \sum_{X\in A, Z=0} |\frac{1}{2^{n+1}} - \frac{1}{2^{n+1}}| + \sum_{X\in A, Z=1} |\frac{1}{2^{n+1}} - \frac{1}{2^{n+1}}|$$
$$= 0.$$

Thus $d_{\mathrm{TV}}(P, Q) = \frac{|A|}{2^{n+1}}$. □

As a corollary to the above proof we get that NeqBayes is NP-complete.

Theorem 1.7. NeqBayes *is* NP*-complete even over Bayes nets distributions of in-degree 2.*

A natural question is to compute the total variation distance between a Bayes net distribution and the uniform distribution. Notice that checking whether Bayes net distribution is uniform or not is easy as the distribution is uniform if and only if all the conditional probabilities in the CPT are uniform. However we show that exactly computing the distance to the uniform distribution is again #P-complete.

Theorem 3.9. DistBayesUniform *is* #P*-complete.*

Proof. The reduction is similar and we use the same notation. For a formula F the distribution is same as P where the last variable takes OR of Y_n and a random bit. We will write the $d_{\mathrm{TV}}(P, \mathbb{U})$ as an easily computable function of number of satisfying assignments of F.

$$2 \cdot d_{\mathrm{TV}}(P, \mathbb{U}) = \sum_{X,Y,Z} |P - \mathbb{U}| = \underbrace{\sum_{\substack{X,Y,Z \\ Y\neq f(X)}} |P - \mathbb{U}|}_{(1)} + \underbrace{\sum_{\substack{X,Y,Z \\ Y= f(X)}} |P - \mathbb{U}|}_{(2)}.$$

We will evaluate (1) and (2) separately, below. We have

$$\sum_{\substack{X,Y,Z \\ Y\neq f(X)}} |P - \mathbb{U}| = \sum_{\substack{X,Y,Z \\ Y\neq f(X)}} |0 - \frac{1}{2^{n+m+1}}| = \frac{2^{n+1}(2^m - 1)}{2^{n+m+1}} = 1 - \frac{1}{2^m},$$

$$\sum_{\substack{X,Y,Z \\ Y= f(X)}} |P - \mathbb{U}| = \underbrace{\sum_{\substack{X,f(X),Z \\ X\in A}} |P - \mathbb{U}|}_{(3)} + \underbrace{\sum_{\substack{X,f(X),Z \\ X\in R}} |P - \mathbb{U}|}_{(4)},$$

$$\sum_{\substack{X,f(X),Z \\ X\in A}} |P - \mathbb{U}| = \sum_{\substack{X,f(X),0 \\ X\in A}} |P - \mathbb{U}| + \sum_{\substack{X,f(X),1 \\ X\in A}} |P - \mathbb{U}|$$
$$= \sum_{\substack{X,f(X),0 \\ X\in A}} |0 - \frac{1}{2^{n+1+1}}| + \sum_{\substack{X,f(X),1 \\ X\in A}} |\frac{1}{2^n} - \frac{1}{2^{n+m+1}}|$$

$$= \frac{|A|}{2^{n+m+1}} + \frac{|A| \cdot (2^{m+1}-1)}{2^{n+m+1}} = \frac{|A|}{2^n},$$

and

$$\begin{aligned}\sum_{\substack{X,f(X),Z\\X\in R}} |P-\mathbb{U}| &= \sum_{\substack{X,f(X),0\\X\in R}} |P-\mathbb{U}| + \sum_{\substack{X,f(X),1\\X\in R}} |P-\mathbb{U}| \\ &= \sum_{\substack{X,f(X),0\\X\in R}} |\frac{1}{2^{n+1}} - \frac{1}{2^{n+m+1}}| + \sum_{\substack{X,f(X),1\\X\in R}} |\frac{1}{2^{n+1}} - \frac{1}{2^{n+m+1}}| \\ &= \frac{|R| \cdot (2^m-1) \cdot 2}{2^{n+m+1}}.\end{aligned}$$

Thus

$$\begin{aligned}2 \cdot d_{\mathrm{TV}}(P,\mathbb{U}) &= (1)+(3)+(4) \\ &= 1 - \frac{1}{2^m} + \frac{|A|}{2^n} + \frac{|R| \cdot (2^m-1)\cdot 2}{2^{n+m+1}} = 2 \cdot (1 - \frac{1}{2^m} + \frac{|A|}{2^{m+n+1}})\end{aligned}$$

and so $d_{\mathrm{TV}}(P,U) = \frac{|A|}{2^{n+m+1}} + (1-\frac{1}{2^m})$. □

4 Algorithms for approximating the TV distance

4.1 Products of Bernoulli distributions, where $p_i \geq 1/2$ and $q_i \leq p_i$

Theorem 1.4. *There is a randomized polynomial-time algorithm that on input ε, δ and probability vectors $\langle p_i, \cdots, p_n\rangle$ and $\langle q_i, \cdots, q_n\rangle$ that defines two binary product distributions P and Q, where $\frac{1}{2} \leq p_i < 1$ and $0 < q_i \leq p_i$, outputs a number v so that $(1-\varepsilon)d_{\mathrm{TV}}(P,Q) \leq v \leq (1+\varepsilon)d_{\mathrm{TV}}(P,Q)$ with probability $\geq 1-\delta$.*

Proof. Without loss of generality, we focus on indices i such that $p_i \neq q_i$. Moreover, we assume that $q_i \geq 1/3$ without loss of generality.

Setting. As we have already seen in Section 1.3, it is the case that

$$\begin{aligned}d_{\mathrm{TV}}(P,Q) &= \frac{1}{2}\sum_{x\in\{0,1\}^n} |P(x)-Q(x)| \\ &= \sum_{x\in\{0,1\}^n} \max(0, P(x)-Q(x)) \\ &= \sum_{S\subseteq[n]} \max\left(0, \prod_{i\in S} p_i \prod_{i\notin S}(1-p_i) - \prod_{i\in S} q_i \prod_{i\notin S}(1-q_i)\right) \\ &= \sum_{S\subseteq[n]} \max\left(0, \prod_{i\in[n]}(1-p_i)\prod_{i\in S}\frac{p_i}{1-p_i} - \prod_{i\in[n]} q_i \prod_{i\notin S}\frac{1-q_i}{q_i}\right) \\ &= \sum_{S\subseteq[n]} \max(0, R_1 \cdot W_S - R_2 \cdot K_S)\end{aligned}$$

where $R_1 := \prod_{i\in[n]}(1-p_i)$ and $R_2 := \prod_{i\in[n]} q_i$ are constants, and $W_S := \prod_{i\in S}\frac{p_i}{1-p_i}$, $K_S := \prod_{i\notin S}\frac{1-q_i}{q_i}$.

Each non-zero term $\max(0, P(x) - Q(x))$ contributes at least

$$m := \min_i (p_i - q_i) \min\left(\prod_{j \neq i} p_j, \prod_{j \neq i} q_j\right) = \min_i (p_i - q_i) \prod_{j \neq i} q_j \geq \min_i \frac{p_i - q_i}{3^n}.$$

Moreover, each non-zero term of $\max(0, P(x) - Q(x))$ contributes at most $M := 1$. Let $U := M/m$. Then $Y_S := \max(0, R_1 \cdot W_S - R_2 \cdot K_S)/m$ lies in $[1, U]$. We now partition $[1, U]$ in intervals of the form $\left[(1+\varepsilon)^{i-1}, (1+\varepsilon)^{i+1}\right)$, that is,

$$[1, U] = \bigcup_{i=1}^{u} \left[(1+\varepsilon)^{i-1}, (1+\varepsilon)^{i+1}\right),$$

where $u := \lceil \log_{1+\varepsilon} U \rceil$.

Let the number of sets S such that Y_S is in $\left[1, (1+\varepsilon)^{i}\right)$ be n_i, and let the average contribution in the range $\left[(1+\varepsilon)^{i-1}, (1+\varepsilon)^{i}\right)$ be B_i. We have

$$\frac{d_{\mathrm{TV}}(P, Q)}{m} = n_1 B_1 + (n_2 - n_1) B_2 + \cdots + (n_u - n_{u-1}) B_u. \tag{1}$$

Since $(1+\varepsilon)^{i-1} \leq B_i \leq (1+\varepsilon)^{i}$, the following is a $(1+\varepsilon)$-factor approximation of the RHS of Equation (1):

$$d := n_1 (1+\varepsilon) + (n_2 - n_1)(1+\varepsilon)^2 + \cdots + (n_u - n_{u-1})(1+\varepsilon)^u. \tag{2}$$

By a reorganization trick [dCM19], Equation (2) becomes

$$d = \left((1+\varepsilon)^u - (1+\varepsilon)^{u-1}\right)(n_u - n_{u-1}) + \cdots + (1+\varepsilon) n_u,$$

see Figure 4.1.

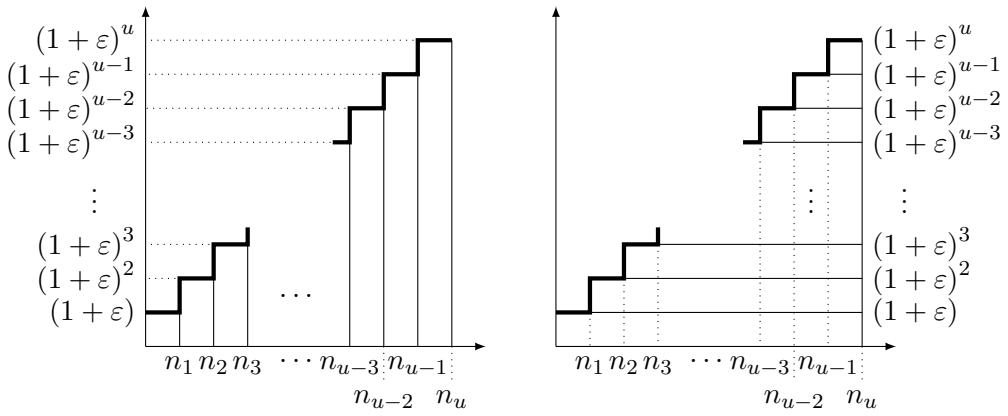

Figure 4.1: The sum reorganization trick.

Therefore, it would suffice to estimate $n_u - n_j$ for all $1 \leq j \leq u-1$. (Note that $n_u = 2^n$.) Thus, $t_j := n_u - n_j$ counts the sets S for which Y_S is at least $(1+\varepsilon)^j$.

Moreover, Y_S can be also written as $Y_S = \max\left(0, \exp\left(b_1 + \sum_{i \in S} x_i\right) - \exp\left(b_2 + \sum_{i \notin S} y_i\right)\right)/m$ if we set $x_i := \ln \frac{p_i}{1-p_i}$, $y_i := \ln \frac{1-q_i}{q_i}$, and $b_k := \ln R_k$. Note that $x_i \geq 0$ and $x_i + y_i \geq 0$, by the choices of p_i and q_i.

As we mentioned above, what is left is to count the number of sets S such that $Y_S \geq (1+\varepsilon)^j$. Let $A := (1+\varepsilon)^j\, m$. Equivalently, we want to count the number of sets S such that

$$\exp\left(b_1 + \sum_{i\in S} x_i\right) - \exp\left(b_2 + \sum_{i\notin S} y_i\right) \geq A,$$

$$\text{or} \qquad \exp\left(b_2 - b_1 + \sum_{i\notin S} y_i - \sum_{i\in S} x_i\right) + A\exp\left(-b_1 - \sum_{i\in S} x_i\right) \leq 1.$$

Observe that $\exp\left(\sum_{i\in[n]} x_i - \sum_{i\in S} x_i\right) / \exp\left(\sum_{i\in[n]} x_i\right) = B\exp\left(\sum_{i\notin S} x_i\right)$ by setting $B := 1/\exp\left(\sum_{i\in[n]} x_i\right)$. For that matter, we may focus on counting the number of sets S such that

$$A\exp\left(\sum_{i\notin S} x_i\right) + B\exp\left(\sum_{i\notin S} x_i + y_i\right) \leq 1$$

whereby A and B are appropriately updated to absorb the expressions containing the constants b_1 and b_2. Let $\mathcal{S}_0$ be the collection of such sets S, and let $\mathcal{S}$ be the collection of sets S such that

$$A\exp\left(\sum_{i\in S} x_i\right) + B\exp\left(\sum_{i\in S} x_i + y_i\right) \leq 1.$$

Note that $|\mathcal{S}| = |\mathcal{S}_0|$. From now on we shall focus on estimating $|\mathcal{S}|$.

Adaptive discretization. A core idea in our work is to rely on adaptive discretization wherein the discretization depends on the choice of the subset S of $[n]$. Since S is unknown a priori, it is unclear how such a discretization may work. To this end, our key insight is to design a discretization scheme that only depends on $\max_{i\in S} x_i$ and $\max_{i\in S} x_i + y_i$. To this end, let $W_1 := \max_{i\in S} x_i$ and $W_2 := \max_{i\in S} x_i + y_i$. We elaborate on this idea below.

Let $z_i := x_i + y_i$,

$$x'_i := \left\lfloor \frac{10n}{W_1} x_i \right\rfloor, \qquad \text{and} \qquad z'_i := \left\lfloor \frac{10n}{W_2} z_i \right\rfloor,$$

and so $x_i = \frac{W_1}{10n}\,(x'_i + \delta_i)$ and $z_i = \frac{W_2}{10n}\,(z'_i + \lambda_i)$ whereby δ_i and λ_i are rounding errors and $0 \leq \delta_i, \lambda_i \leq 1$.

Let $\mathcal{S}'$ be the set of solutions of the discretized problem, that is, the collection of sets S such that

$$A\exp\left(\frac{W_1}{10n}\sum_{i\in S} x'_i\right) + B\exp\left(\frac{W_2}{10n}\sum_{i\in S} z'_i\right) \leq 1.$$

We will be estimating the size of $\frac{|\mathcal{S}|}{|\mathcal{S}'|}$ by sampling from $\mathcal{S}'$ and checking the membership of the sample in $\mathcal{S}$.

Comparing $|\mathcal{S}|$ to $|\mathcal{S}'|$. Note that

$$A\exp\left(\sum_{i\in S} x_i\right) \geq A\exp\left(\frac{W_1}{10n}\sum_{i\in S} x'_i\right),$$

$$B\exp\left(\sum_{i\in S} x_i + y_i\right) \geq B\exp\left(\frac{W_2}{10n}\sum_{i\in S} x'_i + y'_i\right) = B\exp\left(\frac{W_2}{10n}\sum_{i\in S} z'_i\right),$$

and so

$$A\exp\left(\frac{W_1}{10n}\sum_{i\in S}x'_i\right)+B\exp\left(\frac{W_2}{10n}\sum_{i\in S}z'_i\right)$$
$$\leq A\exp\left(\sum_{i\in S}x_i\right)+B\exp\left(\sum_{i\in S}x_i+y_i\right)\leq 1.$$

which means that any solution S to the original problem is also a solution of the scaled problem.

We will now show that $|\mathcal{S}'|\leq \text{poly}(n)\,|\mathcal{S}|$. To this end, we will define a function $f:\mathcal{S}'\to\mathcal{S}$ and show that for every $S\in\mathcal{S}$ there at most $\text{poly}(n)$ sets in $\mathcal{S}'$ mapping (under f) to it. Our function f is defined as follows:

> For $S\in\mathcal{S}$, let $f(S):=S\in\mathcal{S}$. Consider $S\in\mathcal{S}'\setminus\mathcal{S}$; let be ℓ_1 the index of the heaviest element x_i (where $i\in S$) and ℓ_2 be the index of the heaviest element y_i (where $i\in S$). Let now $f(S):=S\setminus(\{\ell_1\}\cup\{\ell_2\})$.

We will show that if $S\in\mathcal{S}'\setminus\mathcal{S}$, then $f(S)\in\mathcal{S}$.

Claim 4.1. *If $S\in\mathcal{S}'\setminus\mathcal{S}$, then*

$$A\exp\left(\sum_{i\in f(S)}x_i\right)+B\exp\left(\sum_{i\in f(S)}x_i+y_i\right)\leq 1$$

for $S\in\mathcal{S}'\setminus\mathcal{S}$.

Proof. First observe that since $S\in\mathcal{S}'\setminus\mathcal{S}$, we know that

$$A\exp\left(\frac{W_1}{10n}\sum_{i\in S}x'_i\right)+B\exp\left(\frac{W_2}{10n}\sum_{i\in S}z'_i\right)\leq 1.$$

Note that $x_{\ell_1}-\frac{W_1|S|}{10n}=W_1-\frac{W_1|S|}{10n}>0$ and for that matter

$$\begin{aligned}A\exp\left(\frac{W_1}{10n}\sum_{i\in S}x'_i\right)&\geq A\exp\left(\frac{W_1}{10n}\sum_{i\in S}\left(\frac{10n}{W_1}x_i-1\right)\right)\\&=A\exp\left(\sum_{i\in S}x_i-\frac{W_1}{10n}\right)\\&\geq A\exp\left(\sum_{i\in S\setminus\{\ell_1\}}x_i\right)\\&\geq A\exp\left(\sum_{i\in f(S)}x_i\right).\end{aligned}$$

Also, $x_{\ell_2}+y_{\ell_2}-\frac{W_2|S|}{10n}=W_2-\frac{W_2|S|}{10n}>0$ and, therefore,

$$B\exp\left(\frac{W_2}{10n}\sum_{i\in S}z'_i\right)\geq B\exp\left(\frac{W_2}{10n}\sum_{i\in S}\left(\frac{10n}{W_2}z_i-1\right)\right)$$

$$
\begin{aligned}
&= B \exp\left(\sum_{i \in S} z_i - \frac{W_2}{10n}\right) \\
&\geq B \exp\left(\sum_{i \in S \setminus \{\ell_2\}} z_i\right) \\
&\geq B \exp\left(\sum_{i \in f(S)} x_i + y_i\right)
\end{aligned}
$$

similarly as above.

Taken together, the considerations above yield the desired

$$
\begin{aligned}
&A \exp\left(\sum_{i \in S} x_i\right) + B \exp\left(\sum_{i \in S} x_i + y_i\right) \\
&\qquad \leq A \exp\left(\frac{W_1}{10n} \sum_{i \in S} x'_i\right) + B \exp\left(\frac{W_2}{10n} \sum_{i \in S} z'_i\right) \leq 1.
\end{aligned}
$$

□

What is left now is to show that for every $S \in \mathcal{S}$ there at most $\mathrm{poly}(n)$ sets in $\mathcal{S}'$ mapping (under f) to it. To do this, we revisit the definition of f, whereby a set $S \in \mathcal{S}$ is mapped to S, and a set $S \in \mathcal{S}' \setminus \mathcal{S}$ is mapped to $S \setminus (\{\ell_1\} \cup \{\ell_2\})$, where ℓ_1 is the index of the heaviest element x_i (where $i \in S$) and ℓ_2 is the index of the heaviest element $x_i + y_i$ (where $i \in S$). Then there are at most $(n+1)^2$ sets mapped to S under f.

Sampling. We first assume every (x_i, y_i) tuple is distinct for simplicity and discuss the general case later. We define $F(\ell, A, B)$ to be equal to the solutions of the scaled problem, that is, the number of sets $S \subseteq [\ell]$ such that $S \in \mathcal{S}'$ and use this dynamic programming idea to compute $F(\ell, A, B)$:

$$
F(0, A, B) = \begin{cases} 1, & \text{if } A + B \leq 1, \\ 0, & \text{otherwise;} \end{cases} \qquad F(\ell, A, B) = F(\ell - 1, A, B) + F\left(\ell - 1, A', B'\right),
$$

where $A' := A \exp\left(\frac{W_1}{10n} x'_\ell\right)$ and $B' := B \exp\left(\frac{W_2}{10n} (x'_\ell + y'_\ell)\right)$. What is left is to find a way of sampling sets $S \in \mathcal{S}$.

We use the following procedure to generate a sample S from $\mathcal{S}'$:

1. Sort $\{(x_i, y_i)\}_{i \in [n]}$ according to $\{x_i\}_{i \in [n]}$.
2. For (ℓ_1, ℓ) ranging in $[n] \times [n]$:
   (a) $W_1 \leftarrow x_{\ell_1}$, $W_2 \leftarrow x_\ell + y_\ell$.
   (b) If $x_{\ell_1} + y_{\ell_1} > W_2$, then abort (we assign probability zero to the pair (W_1, W_2)).
   (c) Find $\ell_2 \leq \ell_1$ such that $x_{\ell_2} + y_{\ell_2} = W_2$; if no such ℓ_2 exists, then abort (we assign probability zero to the pair (W_1, W_2)).
   (d) Let Σ be the subset of $\{1, \ldots, \ell_1\}$ such that $\max_{i \in \Sigma} x_i + y_i \leq W_2$.
   (e) Set the probability to sample (W_1, W_2) proportional to $N_{(\ell_1, \ell)}$, where $N_{(\ell_1, \ell)}$ is the number of subsets $S \in \mathcal{S}'$ of Σ such that $W_1 = \max_{i \in S} x_i$ and $W_2 = \max_{i \in S} x_i + y_i$ (otherwise, we assign probability zero to the pair (W_1, W_2)).
   Note that $N_{(\ell_1, \ell)}$ is computable in polynomial time using dynamic programming.

3. Sample $(W_1, W_2) \propto N_{(\ell_1,\ell)}$.

4. We will sample $S \subseteq \Sigma$ such that $\max_S x_i = W_1$, $\max_S x_i + y_i = W_2$, and $S \in \mathcal{S}'$. That is:

   (a) Set $S := \emptyset$ and $\ell := \ell_0$ (whereby we assume that $\Sigma = \{i_1, \ldots, i_{\ell_0}\}$).

   (b) Set $A_{\ell_0} := A$ and $B_{\ell_0} := B$.

   (c) While $\ell > 0$ do:

      i. With probability $F(i_{\ell-1}, A_{\ell-1}, B_{\ell-1}) \,/ F(i_\ell, A_\ell, B_\ell)$ do:

         A. Set $S := S \cup \{i_\ell\}$;

         B. Set $A_{\ell-1} := A_\ell \exp\left(\frac{W_1}{10n} x'_{i_\ell}\right)$;

         C. Set $B_{\ell-1} := B_\ell \exp\left(\frac{W_2}{10n}\left(x'_{i_\ell} + y'_{i_\ell}\right)\right)$.

      ii. $\ell := \ell - 1$.

   (d) Return S.

Note that $|\mathcal{S}'| = \sum_{\ell_1,\ell} N_{(\ell_1,\ell)}$. Now, if there is a set of repeated pairs $R = \{i, j, \ldots\}$ that give rise to the same $(W_1, W_2) = (x_i, x_i + y_i) = (x_j, x_j + y_j)$, then we need to count all sets T that use (W_1, W_2) as the maximum pair, one item from R is included in T, and T must satisfy the scaled constraint. Here we use a dynamic programming idea for constructing two tables of values $F_1(\ell, A, B)$ and $F_2(\ell, A, B)$ that count the sets of solutions which are subsets of Σ and $\Sigma \setminus R$ respectively. During our recursive sampling, we look at the differences $F_1(\ell, A, B) - F_2(\ell, A, B)$ for various choices of ℓ, A, B which count the set of solutions having at least one item from R.

More generally, the indices for the repetitions of W_1 and W_2 could be non-singleton and different. For such a pair (W_1, W_2), we first filter out all items having x_i values more than W_1. We further filter out all items having y_i values more than W_2. Let Σ, R_1 and R_2 be the subsets of all items, items having x_i values equal to W_1, and items having $x_i + y_i$ values equal to W_2 respectively after this process. If either of R_1 or R_2 is empty, this (W_1, W_2) pair is invalid.

Let $F_1(\ell, A, B), F_2(\ell, A, B), F_3(\ell, A, B), F_4(\ell, A, B)$ count the set of solutions of the discretized constraint which are subsets of $\Sigma \setminus R_1, \Sigma \setminus R_2, \Sigma \setminus (R_1 \cup R_2), \Sigma$ respectively for various values of ℓ, A, B; which we can count using dynamic programming. Then $(F_1 + F_2 - F_3)$ counts the set of solutions which do not use (W_1, W_2) as the maximum pair. It follows that $F_4 - (F_1 + F_2 - F_3)$ counts exactly the set of solutions which use (W_1, W_2) as the maximum pair. Sampling from the later set can be carried out using these counts.

Counting. We rely on the standard Monte Carlo 0–1 estimation technique. To this end, we maintain a counter, say count. We first sample S and if $S \in \mathcal{S}$, then we increment count by 1. We repeat the whole procedure $t = \mathsf{poly}(n, 1/\varepsilon, 1/\delta)$, where δ is the confidence error of the FPRAS and we estimate $|\mathcal{S}|$ as $\frac{\mathsf{count}}{t} \cdot |\mathcal{S}'|$.

Running time. The running time is polynomial, as (a) the possible values for j are $\ln(M/m) \lesssim n \min_i \ln \frac{1}{p_i - q_i}$, (b) Item 1 can be done in time $O(n \log n)$, (c) Item 2 is executed n^2 times, (d) Item 2e can be done in time $O(n^5)$ (since the choices of (A, B) are $O(n^4)$, and the choices of ℓ are $O(n)$), and (e) is executed $O(n)$ times. □

4.2 Products of Bernoulli distributions, for $O(1)$ distinct q_i's

We will consider the case where P is an arbitrary distribution and Q has $\leq k$ distinct parameters. Without loss of generality, let $Q = \bigotimes_i \mathrm{Bern}(q_i) = \mathrm{Bern}(a_1)^{n_1} \otimes \cdots \otimes \mathrm{Bern}(a_k)^{n_k}$ such that $n_1 + \cdots + n_k = n$. In this section we give an FPRAS for $d_{\mathrm{TV}}(P, Q)$ when k is a constant.

For simplicity of exposition, we will first show the result for the case when $Q = \mathrm{Bern}(a)^n$. Our approach is to reduce this problem to #KNAPSACK with a fixed Hamming weight. The latter problem has been solved in [GKM10] by using small space sources, see Theorem 2.3. One issue arises if there is an i such that $p_i < 1/2$. In this case, we switch 0 and 1 in such coordinates to obtain $\mathrm{Bern}(1-p_i)$ and $\mathrm{Bern}(2/3)$. However, this still can be reduced to #KNAPSACK with two fixed Hamming weights. Moreover, such strings can also be randomly sampled by a small space source.

Proposition 4.2. *There is an FPRAS for estimating $d_{\mathrm{TV}}(P,Q)$ where P is an arbitrary distribution and $Q = \mathrm{Bern}(a)^n$ for an $0 \le a \le 1$.*

Proof. We have

$$\begin{aligned} d_{\mathrm{TV}}(P,Q) &= \sum_{S\subseteq[n]} \max\left(\prod_{i\in S} p_i \prod_{i\notin S}(1-p_i) - (a)^{|S|}(1-a)^{n-|S|}, 0\right) \\ &= \prod_{i\in[n]}(1-p_i) \sum_{S\subseteq[n]} \left(\prod_{i\in S}\left(\frac{p_i}{1-p_i}\right) - \frac{1}{\prod_{i\in[n]}(1-p_i)}(1-a)^n\left(\frac{1}{2}\right)^{|S|}\right) \\ &= A \sum_{S\subseteq[n]} \max\left(\prod_{i\in S} w_i - B\left(\frac{1}{2}\right)^{|S|}\right). \end{aligned}$$

From our earlier discussion, it suffices to approximately count sets S such that $\prod_{i\in S} w_i \le B\left(\frac{1}{2}\right)^{|S|} + C = D$ for some C, D. Firstly assume, $w_i \ge 1$ for every i, we take a logarithm to reduce this to a #KNAPSACK instance for every fixed $|S| = 1, 2, \ldots, n$. As mentioned earlier, the latter problems can be solved by a result from [GKM10] (see Theorem 2.3). Finally, we take the sum of these counts to solve our problem.

If some $w_i < 1$, we switch 0 and 1 in those coordinates to get $\mathrm{Bern}(1-p_i)$ and $\mathrm{Bern}(2/3)$. Then in Q, the first m coin biases are $1/3$ and the last $(n-m)$ coin biases are $2/3$ wlog. In that case, as before:

$$d_{\mathrm{TV}}(P,Q) = A \sum_{S\subseteq[n]} \max\left(\prod_{i\in S} w_i - B\prod_{i\in S} v_i\right),$$

where $w_i > 1$ for every i and $v_i = \frac{q_i}{1-q_i}$. Now for every S, Hamming weight for the first m places are s_1 and last $n-m$ places are s_2. Therefore, it suffices to solve a #KNAPSACK instance with the constraints as follows:

- $\prod_{i\in S} w_i \le D + (1/3)^{s_1}(2/3)^{m-s_1}(2/3)^{s_2}(1/3)^{n-m-s_2}$;
- Hamming weight of S on first m bits is s_1;
- Hamming weight of S on last $n-m$ bits is s_2.

The last two constraints can be encoded in a small space source to generate random strings. Therefore the same paper [GKM10] gives an algorithm for the above #KNAPSACK problem, as we vary over every fixing of s_1, s_2. We can finally sum over all the disjoint possibilities of s_1 and s_2 to get our final answer. □

Now we return to the case where Q has k distinct parameters, i.e. without loss of generality, $Q = \otimes_i \mathrm{Bern}(q_i) = \mathrm{Bern}(a_1)^{n_1} \otimes \cdots \otimes \mathrm{Bern}(a_k)^{n_k}$ such that $n_1 + \cdots + n_k = n$.

Proposition 4.3 (Theorem 1.5 restated)**.** *There is an FPRAS for $d_{\mathrm{TV}}(P,Q)$ where $Q = \bigotimes_i \mathrm{Bern}(q_i) = \mathrm{Bern}(a_1)^{n_1} \otimes \cdots \otimes \mathrm{Bern}(a_k)^{n_k}$ such that $n_1 + \cdots + n_k = n$.*

Proof. First, assume that $p_i \geq 1/2$ for all i. We have

$$d_{\mathrm{TV}}(P,Q) = \sum_{S \subseteq [n]} \max\left(\prod_{i \in S} p_i \prod_{i \notin S} (1-p_i) - a_1^{n_{11}}(1-a_1)^{n_{10}} \cdots a_k^{n_{k1}}(1-a_k)^{n_{k0}}, 0 \right),$$

where $n_i = n_{i0} + n_{i1}$ such that n_{i0} and n_{i1} denote the counts of 0's and 1's respectively in the group of a_i parameters in S. It suffices to count sets such that the last expression is at most A for different values of A. We perform this count as follows. We partition the 2^n values of S into subsets corresponding to every possibility of n_{i1} and n_{i0}s between 0 through n_i. Hence there are at most n^k many partitions. For each partition, we solve a KNAPSACK instance with the following constraints:

- Each n_{i1} and n_{i0} correspond to a fixed possibility determined by S;
- $\prod_{i \in S} \frac{p_i}{1-p_i} \leq B$ for some B determined by n_{i1}s and n_{i0}s.

Each of the constraints in the first item can be sampled by a small space source of width at most n^k and therefore each partition can be approximately counted in polynomial time whenever $k = O(1)$. Our final answer consists of the sum over all the partitions, which will still be $(1 \pm \varepsilon)$-approximate.

Now, if there are any $p_i < 1/2$, we work with $(1-p_i)$ and $(1-q_i)$ at that particular coordinate. Essentially, this effectively doubles the number of Bernoulli parameters to $2k$ and the resulting algorithm is still polynomial time for $k = O(1)$. □

Acknowledgements

The works of AB and KSM were supported in part by National Research Foundation Singapore under its NRF Fellowship Programme (NRF-NRFFAI-2019-0002, NRF-NRFFAI1-2019-0004), Ministry of Education Singapore Tier 2 grants (MOE-T2EP20121-0011, MOE2019-T2-1-152), NUS startup grant (R-252-000-A33-133), and Amazon Faculty Research Awards. SG was supported by an Initiation grant of IIT Kanpur. AP was supported in part by NSF CCF-2130536 and NSF HDR:TRIPODS-1934884 awards. NVV was supported in part by NSF CCF-2130608 and NSF HDR:TRIPODS-1934884 awards. The work was done in part while AB and DM were visiting the Simons Institute for the Theory of Computing.

References

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[BGMV20] Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, and N. V. Vinodchandran. Efficient distance approximation for structured high-dimensional distributions via learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[Can22] Clément Canonne. Topics and techniques in distribution testing. Preprint at https://ccanonne.github.io/files/misc/main-survey-fnt.pdf, 2022.

[CDKS18] Yu Cheng, Ilias Diakonikolas, Daniel Kane, and Alistair Stewart. Robust learning of fixed-structure bayesian networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[CMR07] Corinna Cortes, Mehryar Mohri, and Ashish Rastogi. L_p distance and equivalence of probabilistic automata. *Int. J. Found. Comput. Sci.*, 18(4):761–779, 2007.

[dCM19] Alexis de Colnet and Kuldeep S. Meel. Dual hashing-based algorithms for discrete integration. In *Principles and Practice of Constraint Programming - 25th International Conference, CP 2019, Stamford, CT, USA, September 30 - October 4, 2019, Proceedings*, volume 11802 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2019.

[DFK+93] Martin E. Dyer, Alan M. Frieze, Ravi Kannan, Ajai Kapoor, Ljubomir Perkovic, and Umesh V. Vazirani. A mildly exponential time algorithm for approximating the number of solutions to a multidimensional knapsack problem. *Comb. Probab. Comput.*, 2:271–284, 1993.

[DL01] Luc Devroye and Gábor Lugosi. *Combinatorial methods in density estimation*. Springer Science & Business Media, 2001.

[Dye03] Martin E. Dyer. Approximate counting by dynamic programming. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 693–699. ACM, 2003.

[GKM10] Parikshit Gopalan, Adam R. Klivans, and Raghu Meka. Polynomial-time approximation schemes for knapsack and related counting problems using branching programs. *Electron. Colloquium Comput. Complex.*, page 133, 2010.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[GSV99] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. Can statistical zero knowledge be made non-interactive? or on the relationship of SZK and NISZK. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*, pages 467–484. Springer, 1999.

[Jer03] Mark Jerrum. Counting, sampling and integrating: algorithms and complexity. In *Lectures in Mathematics – ETH Zürich*. Birkhauser, Berlin, 2003.

[Kie18] Stefan Kiefer. On computing the total variation distance of hidden markov models. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 130:1–130:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[KRVZ06] Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 691–700. ACM, 2006.

[LP02] Rune B. Lyngsø and Christian N. S. Pedersen. The consensus string problem and the complexity of comparing hidden markov models. *J. Comput. Syst. Sci.*, 65(3):545–569, 2002.

[MS04] Ben Morris and Alistair Sinclair. Random walks on truncated cubes and sampling 0-1 knapsack solutions. *SIAM J. Comput.*, 34(1):195–226, 2004.

[MV18] Jack Murtagh and Salil P. Vadhan. The complexity of computing the optimal composition of differential privacy. *Theory Comput.*, 14(1):1–35, 2018.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[SV03] Amit Sahai and Salil P. Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003.

[SVV12] Daniel Stefankovic, Santosh S. Vempala, and Eric Vigoda. A deterministic polynomial-time approximation scheme for counting knapsack solutions. *SIAM J. Comput.*, 41(2):356–366, 2012.

[Tod91] Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991.

[Vem21] Santosh Vempala. Personal communication, 2021.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982.