Quantum Information and Computation, Vol. 17, No. 7&8 (2017) 0541–0567

QUANTUM CONDITIONAL QUERY COMPLEXITY

IMDAD S. B. SARDHARWALLA
Department of Applied Mathematics and Theoretical Physics
Centre for Mathematical Sciences, University of Cambridge
Wilberforce Road, Cambridge CB3 0WA, U.K.

SERGII STRELCHUK
Department of Applied Mathematics and Theoretical Physics
Centre for Mathematical Sciences, University of Cambridge
Wilberforce Road, Cambridge CB3 0WA, U.K.

RICHARD JOZSA
Department of Applied Mathematics and Theoretical Physics
Centre for Mathematical Sciences, University of Cambridge
Wilberforce Road, Cambridge CB3 0WA, U.K.

Received September 15, 2016
Revised May 3, 2017

We define and study a new type of quantum oracle, the quantum conditional oracle, which provides oracle access to the conditional probabilities associated with an underlying distribution. Amongst other properties, we (a) obtain highly efficient quantum algorithms for identity testing, equivalence testing and uniformity testing of probability distributions; (b) study the power of these oracles for testing properties of boolean functions, and obtain an algorithm for checking whether an n-input m-output boolean function is balanced or ϵ-far from balanced; and (c) give an algorithm, requiring $\tilde{O}(n/\epsilon)$ queries, for testing whether an n-dimensional quantum state is maximally mixed or not.

Keywords: quantum query complexity, boolean functions, quantum oracle, quantum spectrum testing

Communicated by: R Cleve & A Harrow

1 Introduction

One of the fundamental challenges in statistics is to infer information about properties of large datasets as efficiently as possible. This is becoming increasingly important as we collect progressively more data about our world and our lives. Often one would like to determine a certain property of the collected data while having no physical ability to access all of it. This can be formalised as the task of *property testing*: determining whether an object has a certain property, or is 'far' from having that property, ideally minimising the number of inspections of it. There has been an explosive growth in recent years in this field [21, 20, 7], and particularly in the sub-field of *distribution testing*, in which one seeks to learn information

about a data set by drawing samples from an associated probability distribution.

The *classical conditional sampling oracle* (COND) [2, 10, 12] grants access to a distribution D such that one can draw samples not only from D, but also from D_S, the conditional distribution of D restricted to an arbitrary subset S of the domain. Such oracle access reveals a separation between the classical query complexity of identity testing (i.e. whether an unknown distribution D is the same as some known distribution D^*), which takes a constant number of queries, and equivalence testing (i.e. whether two unknown distributions D_1 and D_2 are the same), which requires $\Omega(\sqrt{\log \log N})$ queries, where N is the size of the domain [2]. In this paper we introduce a natural quantum version of the COND oracle (see Definition 2.4 below) and study its computational power.

More specifically, we will consider the PCOND (pairwise-COND) oracle, which only accepts query subsets S of cardinality 2 or N, and introduce the PQCOND (pairwise-QCOND) oracle. While being rather restricted in comparison to the full COND and QCOND oracles, they nevertheless offer significant advantages over the standard sampling oracles. Like the PCOND oracle, it is not immediately clear how to implement the PQCOND oracle in practice. However, results for this oracle may be the first step to understanding general conditional models, which occur in practice often, in more detail.

1.1 Results

Quantum algorithms for property testing problems. We study the following property testing tasks for classical probability distributions and present efficient algorithms for their solution using our PQCOND oracle. We compare our results with previously known bounds for the standard quantum sampling oracle QSAMP and the classical PCOND oracle.

1. *Uniformity Test:* Given a distribution D and a promise that D is either the uniform distribution $\mathcal{A}$ or $|D - \mathcal{A}| \geq \epsilon$, where $|\cdot|$ is the L_1-norm, decide which of the options holds.
2. *Known-distribution Test:* Given a fixed distribution D^* and a promise that either $D = D^*$ or $|D - D^*| \geq \epsilon$, decide which of the options holds.
3. *Unknown-distribution Test:* Given two distributions $D^{(1)}$ and $D^{(2)}$ and a promise that either $D^{(1)} = D^{(2)}$ or $|D^{(1)} - D^{(2)}| \geq \epsilon$, decide which of the options holds.
4. *Distance from uniformity:* Given a distribution D and the uniform distribution $\mathcal{A}$, estimate $\hat{d} = |D - \mathcal{A}|$.

The query complexities for the above problems are listed in Table 1, with our new results given in the last column. The notation $\tilde{O}(f(N,\epsilon))$ denotes $O(f(N,\epsilon)\log^k f(N,\epsilon))$ for some k, i.e. logarithmic factors are hidden.

Testing properties of boolean functions. A slight modification of the PQCOND oracle will allow for the testing of properties of boolean functions.

Given $f : \{0,1\}^n \to \{0,1\}^m$ with $n \geq m$, define $F_i := |\{x \in \{0,1\}^n : f(x) = i\}|/2^n$ for $i \in \{0,1\}^m$. The function f is promised to be either:

Task	Standard quantum oracle (QSAMP)	PCOND oracle [10]	PQCOND oracle [this work]
Uniformity Test	$O\left(\frac{N^{1/3}}{\epsilon^{4/3}}\right)$ [9]	$\tilde{O}\left(\frac{1}{\epsilon^2}\right), \Omega\left(\frac{1}{\epsilon^2}\right)$	$\tilde{O}\left(\frac{1}{\epsilon}\right)$
Known-distribution Test	$\tilde{O}\left(\frac{N^{1/3}}{\epsilon^5}\right)$ [14]	$\tilde{O}\left[\left(\frac{\log N}{\epsilon}\right)^4\right]$	$\tilde{O}\left[\left(\frac{\log N}{\epsilon}\right)^3\right]$
Unknown-distribution Test	$O\left(\frac{N^{1/2}}{\epsilon^{3/2}}\right)$ [25]	$\tilde{O}\left[\left(\frac{\log^2 N}{\epsilon^7}\right)^3\right]$	$\tilde{O}\left[\left(\frac{\log^2 N}{\epsilon^7}\right)^2\right]$
Distance from uniformity	$O\left(\frac{N^{1/2}}{\epsilon^{3/2}}\right)$ [25]	$\tilde{O}\left(\frac{1}{\epsilon^{20}}\right)$	$\tilde{O}\left(\frac{1}{\epsilon^{13}}\right)$

Table 1: Query complexity for property testing problems using three different access models: the standard quantum oracle (QSAMP), the PCOND oracle, and our PQCOND oracle.

- a balanced function, i.e. $F_i = \frac{1}{2^m} \quad \forall i \in \{0,1\}^m$; or
- ϵ-far from balanced, i.e. $\sum_{i\in\{0,1\}^m} |F_i - \frac{1}{2^m}| \geq \epsilon$.

Provided we have PQCOND access to f, we present a quantum algorithm that decides which of these is the case using $\tilde{O}(1/\epsilon)$ queries.

Testing Mixedness [26] of a Quantum State. The problem of deciding if a quantum state ρ of dimension n is maximally-mixed or ϵ-far from it is a natural question within the framework of Quantum Spectrum Testing. In the standard model, where one measures a number of copies of ρ, it is found that $\Theta(n/\epsilon^2)$ measurements are needed [27]. It is of interest to study whether the PQCOND model, a more powerful model, can improve the complexity of solving this problem, without trivialising it.

Formally, the Mixedness problem is stated as follows: Given access to copies of a quantum state ρ of dimension n and a constant $\epsilon > 0$, it is promised that one of the following holds:

- $\|\rho - \mathbb{1}/n\|_1 = 0$, i.e. ρ is the maximally-mixed state; or
- $\|\rho - \mathbb{1}/n\|_1 \geq \epsilon$, i.e. ρ is ϵ-far from the maximally-mixed state,

where $\|\cdot\|_1$ is the trace norm.[a] Decide which is the case.

We present a quantum algorithm to decide the above problem that uses $\tilde{O}(n/\epsilon)$ PQCOND queries.

[a] For an $(n \times n)$ matrix A, $\|A\|_1 = \mathrm{Tr}\sqrt{AA^\dagger} = \sum_{i\in[n]} a_i$, where the a_i are the singular values of A.

1.2 Motivation

The conditional access model is versatile and well-suited to a wide range of practical applications, one example of which is mentioned below.

Lottery machine. A *gravity pick lottery machine* works as follows: N balls, numbered $1, \ldots, N$, are dropped into a spinning machine, and after a few moments a ball is released. One might wish to determine whether or not such a machine is fair, i.e. whether or not a ball is released uniformly at random. A distribution testing algorithm would correctly decide between the following options (assuming that one is guaranteed to be true) with high probability:

- The lottery machine is fair and outputs i with probability $1/N$;
- The lottery machine is ϵ-far from uniform.

In this example, access to a COND oracle is equivalent to being able to choose which balls are allowed into the spinner. Classically, it is known that $\Theta(N^{1/2}/\epsilon^4)$ queries [5] to the SAMP oracle are required to determine whether or not a distribution generated by such a lottery machine is uniform. However, given access to the corresponding quantum oracle, QSAMP, only $O(N^{1/3}/\epsilon^{4/3})$ queries are required [9]. The COND oracle requires $\Omega(1/\epsilon^2)$ queries, and using the PQCOND oracle we can achieve this with $\tilde{O}(1/\epsilon)$ queries.

Many related problems, such as determining whether or not two known/unknown distributions are identical, have been extensively studied in the classical [6, 31, 10, 11, 19, 22, 15, 14] and quantum [9, 26, 25] literature, and near-optimal bounds have often been placed on the number of queries required to solve the respective problems.

1.3 Outline

In Section 2 we introduce notation and define our quantum conditional oracles. In Section 3 we prove our main technical tool—the QCOMPARE function—which efficiently compares conditional probabilities of a distribution. In Section 4 we apply it to obtain new, efficient query complexity bounds for property testing of probability distributions. In Section 5 we test properties of boolean functions, before presenting a test for the Mixedness of a quantum state in Section 6.

2 Preliminaries and Notation

Let D be a probability distribution over a finite set $[N] \coloneqq \{0, 1, \ldots, N-1\}$, where $D(i) \geq 0$ is the weight of the element $i \in [N]$. Furthermore, if $S \subseteq [N]$, then $D(S) = \sum_{i \in S} D(i)$ is the weight of the set S. If $D(S) > 0$, define D_S to be the conditional distribution, i.e. $D_S(i) \coloneqq D(i)/D(S)$ if $i \in S$ and $D_S(i) = 0$ if $i \notin S$.

Below, we recall the definitions of the classical and quantum sampling oracles, and subsequently define the classical and quantum conditional sampling oracles.

Definition 2.1 (Classical Sampling Oracle [10])**.** *Given a probability distribution D over $[N]$, we define the* classical sampling oracle SAMP_D *as follows: each time* SAMP_D *is queried, it returns a single $i \in [N]$, where the probability that element i is returned is $D(i)$.*

Definition 2.2 (Quantum Sampling Oracle [9]). *Given a probability distribution D over $[N]$, let $T \in \mathbb{N}$ be some specified integer, and assume that D can be represented by a mapping $O_D : [T] \to [N]$ such that for any $i \in [N]$, $D(i)$ is proportional to the number of elements in the pre-image of i, i.e. $D(i) = |\{t \in [T] : O_D(t) = i\}|/T$. In other words, O_D labels the elements of $[T]$ by $i \in [N]$, and the $D(i)$ are the frequencies of these labels, and are thus all rational with denominator T.*

Then each query to the quantum sampling oracle QSAMP_D *applies the unitary operation U_D, described by its action on basis states:*

$$U_D |t\rangle |\beta\rangle = |t\rangle |\beta + O_D(t) \mod N\rangle .$$

In particular,

$$U_D |t\rangle |0\rangle = |t\rangle |O_D(t)\rangle .$$

As an example, note that querying with a uniformly random $t \in [T]$ in the first register will result in $i \in [N]$ in the second register with probability $D(i)$.

Definition 2.3 (Classical Conditional Sampling Oracle [10]). *Given a probability distribution D over $[N]$ and a set $S \subseteq [N]$ such that $D(S) > 0$, we define the* classical conditional sampling oracle COND_D *as follows: each time COND_D is queried with query set S, it returns a single $i \in [N]$, where the probability that element i is returned is $D_S(i)$.*

We are now ready to define a new *quantum conditional sampling oracle*, a quantum version of COND_D.

Definition 2.4 (Quantum Conditional Sampling Oracle). *Given a probability distribution D over $[N]$, let $T \in \mathbb{N}$ be some specified integer, and assume that there exists a mapping $O_D : \mathcal{P}([N]) \times [T] \to [N]$, where $\mathcal{P}([N])$ is the power set of $[N]$, such that for any $S \subseteq [N]$ with $D(S) > 0$ and any $i \in [N]$, $D_S(i) = |\{t \in [T] : O_D(S, t) = i\}|/T$.*

Then each query to the quantum conditional sampling oracle QCOND_D *applies the unitary operation U_D, defined below.*

U_D acts on 3 registers:

- *The first consists of N qubits, whose computational basis states label the 2^N possible query sets S;*
- *The second consists of $\log T$ qubits that describe an element of $[T]$; and*
- *The third consists of $\log N$ qubits to store the output, an element of $[N]$.*

The action of the oracle on basis states is

$$U_D |S\rangle |t\rangle |\beta\rangle = |S\rangle |t\rangle |\beta + O_D(S, t) \mod N\rangle .$$

In particular,

$$U_D |S\rangle |t\rangle |0\rangle = |S\rangle |t\rangle |O_D(S, t)\rangle .$$

Remark: Note that querying QCOND_D with query set $S = [N]$ is equivalent to a query to QSAMP_D.

Remark: In the above definition we have made two key assumptions: the first is that the $D(i)$ are rational values; and the second is that O_D exists, which requires that the values $D_S(i)$ are consistent for different subsets $S \subseteq N$. These are strong promises on D and perhaps rather restrictive. However, by making T sufficiently large (requiring only $\log T$ qubits), we can approximate any probability distribution closely enough that the algorithms discussed in the remainder of this paper can still be applied (since the value of T does not affect the number of queries).

The PCOND_D oracle, described in [10], only accepts query subsets S of cardinality 2 or N. This effectively allows us access to distributions formed from the relative probabilities of pairs of elements. Below we define its quantum analogue, the PQCOND_D oracle.

Definition 2.5 (Pairwise Conditional Sampling Oracle)**.** *The PQCOND_D oracle is equivalent to the QCOND_D oracle, with the added requirement that the query set S must satisfy $|S| = 2$ or N, i.e. the distribution can only be conditioned over pairs of elements or the whole set.*

3 Efficient comparison of conditional probabilities

In this section we improve an algorithm given in [9] by reducing the number of queries it makes to the standard quantum oracle. We subsequently use this result to prove our main technical tool, the QCOMPARE algorithm, which compares conditional probabilities of a distribution, and is central to our improved property testers.

The following lemma, proved in [24], provides a general method for improving the dependence between the number of queries made by an algorithm and its success probability.

Lemma 3.1 (Powering lemma [25, 24])**.** *Let* ALG *be an algorithm (quantum or classical) that aims to estimate a quantity $\mu \in \mathbb{R}$, with its output $\tilde{\mu}$ satisfying $\mathbb{P}[|\tilde{\mu} - \mu| \leq \epsilon] \geq 1 - \gamma$, where $\gamma < \frac{1}{2}$ is fixed.*

Then, for any $\delta > 0$, it suffices to repeat ALG *$O(\log(1/\delta))$ times and take the median to obtain an estimate μ' such that $\mathbb{P}[|\mu' - \mu| \leq \epsilon] \geq 1 - \delta$.*

Applying this lemma to Theorem 5 of [9] gives an exponential improvement, from $1/\delta$ to $\log(1/\delta)$, in the dependence on the success probability given there. This is summarised in the theorem below.

Theorem 3.2. *There exists a quantum algorithm* ADDESTPROB(D, S, M) *that takes as input a distribution D over $[N]$, a set $S \subset [N]$ and an integer M. The algorithm makes exactly M queries to the QSAMP_D oracle and outputs $\tilde{D}(S)$, an approximation to $D(S)$, such that $\mathbb{P}[|\tilde{D}(S) - D(S)| \leq \epsilon] \geq 1 - \delta$ for all $\epsilon > 0$ and $\delta \in (0, 1]$ satisfying*

$$M \geq c \log(1/\delta) \max\left(\frac{\sqrt{D(S)}}{\epsilon}, \frac{1}{\sqrt{\epsilon}}\right),$$

where $c = O(1)$ is some constant.

A multiplicative version Theorem 3.2 follows straightforwardly:

Theorem 3.3. *There exists a quantum algorithm* MULESTPROB(D, S, M) *that takes as input a distribution D over $[N]$, a set $S \subset [N]$ and an integer M. The algorithm makes exactly M queries to the* QSAMP_D *oracle and outputs $\tilde{D}(S)$, an approximation to $D(S)$, such that $\mathbb{P}[\tilde{D}(S) \in [1-\epsilon, 1+\epsilon]D(S)] \geq 1-\delta$ for all $\epsilon, \delta \in (0,1]$ satisfying*

$$M \geq \frac{c \log(1/\delta)}{\epsilon \sqrt{D(S)}},$$

where $c = O(1)$ is some constant.

Access to the QCOND_D oracle effectively gives us access to the oracle QSAMP_{D_S} for any $S \subseteq [N]$, and this allows us to produce stronger versions of Theorems 3.2 and 3.3:

Theorem 3.4. *There exists a quantum algorithm* ADDESTPROBQCOND(D, S, R, M) *that takes as input a distribution D over $[N]$, a set $S \subseteq [N]$ with $D(S) > 0$, a subset $R \subset S$ and an integer M. The algorithm makes exactly M queries to the* QCOND_D *oracle and outputs $\tilde{D}_S(R)$, an approximation to $D_S(R)$, such that $\mathbb{P}[|\tilde{D}_S(R) - D_S(R)| \leq \epsilon] \geq 1-\delta$ for all $\epsilon > 0$ and $\delta \in (0,1]$ satisfying*

$$M \geq c \log(1/\delta) \max\left(\frac{\sqrt{D_S(R)}}{\epsilon}, \frac{1}{\sqrt{\epsilon}}\right),$$

where $c = O(1)$ is some constant.

Theorem 3.5. *There exists a quantum algorithm* MULESTPROBQCOND(D, S, R, M) *that takes as input a distribution D over $[N]$, a set $S \subseteq [N]$ with $D(S) > 0$, a subset $R \subset S$ and an integer M. The algorithm makes exactly M queries to the* QCOND_D *oracle and outputs $\tilde{D}_S(R)$, an approximation to $D_S(R)$, such that $\mathbb{P}[\tilde{D}_S(R) \in [1-\epsilon, 1+\epsilon]D_S(R)] \geq 1-\delta$ for all $\epsilon, \delta \in (0,1]$ satisfying*

$$M \geq \frac{c \log(1/\delta)}{\epsilon \sqrt{D_S(R)}},$$

where $c = O(1)$ is some constant.

3.1 The QCOMPARE *algorithm*

An important routine used in many classical distribution testing protocols (see [10]) is the COMPARE function, which outputs an estimate of the ratio $r_{X,Y} := D(Y)/D(X)$ of the weights of two disjoint subsets $X, Y \subset [N]$ over D. As stated in Section 3.1 of [10], if X and Y are disjoint, $D(X \cup Y) > 0$, and $1/K \leq r_{X,Y} \leq K$ for some chosen integer $K \geq 1$, COMPARE outputs $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$ with probability at least $1-\delta$ using only $O(K \log(1/\delta)/\eta^2)$ COND_D queries. Surprisingly, the number of queries is independent of N, the size of the domain of the distribution.

Here we introduce a procedure called QCOMPARE, constructed from calls to ADDESTPROBQCOND and MULESTPROBQCOND, that uses the QCOND_D oracle and subsequent quantum operations to perform a similar function to COMPARE, achieving the same success probability and bound on the error with $O(\sqrt{K} \log(1/\delta)/\eta)$ queries.

Algorithm 1 QCOMPARE$(D, X, Y, \eta, K, \delta)$

Input: QCOND access to a probability distribution D over $[N]$, disjoint subsets $X, Y \subset [N]$ such that $D(X \cup Y) > 0$, 'range' parameter $K \geq 1$, 'distance' parameter $\eta \in (0, \frac{3}{8K})$, and 'failure probability' parameter $\delta \in (0, 1]$.

1. Set $M = O\left(\frac{\sqrt{K}\log(1/\delta)}{\eta}\right)$.
2. Set $\tilde{w}_+(X) =$ ADDESTPROBQCOND$(D, X \cup Y, X, M)$.
3. Set $\tilde{w}_+(Y) =$ ADDESTPROBQCOND$(D, X \cup Y, Y, M)$.
4. Set $\tilde{w}_\times(X) =$ MULESTPROBQCOND$(D, X \cup Y, X, M)$.
5. Set $\tilde{w}_\times(Y) =$ MULESTPROBQCOND$(D, X \cup Y, Y, M)$.
6. Check that $\tilde{w}_+(X) \leq \frac{3K}{3K+1} - \frac{\eta}{3}$. If the check fails, return Low and exit.
7. Check that $\tilde{w}_+(Y) \leq \frac{3K}{3K+1} - \frac{\eta}{3}$. If the check fails, return High and exit.
8. Return $\tilde{r}_{X,Y} = \frac{\tilde{w}_\times(Y)}{\tilde{w}_\times(X)}$.

Theorem 3.6. *Given the input as described,* QCOMPARE *(Algorithm 1) outputs* Low, High, *or a value* $\tilde{r}_{X,Y} > 0$, *and satisfies the following:*

1. *If* $1/K \leq r_{X,Y} \leq K$, *then with probability at least* $1 - \delta$ *the procedure outputs a value* $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$;
2. *If* $r_{X,Y} > K$ *then with probability at least* $1 - \delta$ *the procedure outputs either* High *or a value* $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$;
3. *If* $r_{X,Y} < 1/K$ *then with probability at least* $1 - \delta$ *the procedure outputs either* Low *or a value* $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$.

The procedure performs $O\left(\frac{\sqrt{K}\log(1/\delta)}{\eta}\right)$ QCOND_D *queries on the set* $X \cup Y$ *via use of* ADDESTPROBQCOND *and* MULESTPROBQCOND.

The proof of this theorem is given in Section A.1.

4 Property testing of probability distributions

We now apply our results to obtain new algorithms for a number of property testing problems.

Corollary 4.1. *Let* $\mathcal{A}^{(N)}$ *be the uniform distribution on* $[N]$ *(i.e.* $\mathcal{A}^{(N)}(i) = 1/N, i \in [N]$*). Given* PQCOND *access to a probability distribution* D *over* $[N]$, *there exists an algorithm that uses* $\tilde{O}(1/\epsilon)$ PQCOND_D *queries and decides with probability at least* $2/3$ *whether*

- $|D - \mathcal{A}^{(N)}| = 0$ *(i.e.* $D = \mathcal{A}^{(N)}$*) (the algorithm outputs* Equal*), or*
- $|D - \mathcal{A}^{(N)}| \geq \epsilon$ *(the algorithm outputs* Far*),*

provided that it is guaranteed that one of these is true. Here $|\cdot|$ *is the* L_1*-norm.*[b]

The intuition behind Corollary 4.1 as well as a simpler, but slightly weaker, algorithm requiring $\tilde{O}(1/\epsilon^4)$ queries to the PCOND oracle is presented in Appendix B.

[b] For two distributions $D^{(1)}$ and $D^{(2)}$ over $[N]$, $|D^{(1)} - D^{(2)}| = \sum_{i \in [N]} |D^{(1)}(i) - D^{(2)}(i)|$.

Proof. We replace the calls to COMPARE with the corresponding calls to QCOMPARE in Algorithm 4 of [10]. For this method, calls to QCOMPARE only require conditioning over pairs of elements, and hence the PQCOND_D oracle may be used instead of QCOND_D. □

Remark: The corresponding classical algorithm (Algorithm 4 in [10]) makes $O\left(\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)$ calls to both the COMPARE procedure (with distance parameter $\tilde{O}(\epsilon^c), c \in [0,1]$, range parameter 2, and failure probability parameter $\mathrm{poly}(\epsilon)$) and the SAMP_D oracle, and in total uses $\tilde{O}(1/\epsilon^2)$ PCOND_D queries. The authors also show (Section 4.2 of [10]) that any classical algorithm making COND_D queries must use $\Omega(1/\epsilon^2)$ queries to solve this problem with bounded probability. Thus the above quantum algorithm is quadratically more efficient than any classical COND algorithm.

Corollary 4.2. *Given the full specification of a probability distribution D^* (i.e. a* known *distribution) and* PQCOND *access to a probability distribution D, both over $[N]$, there exists an algorithm that uses $\tilde{O}\left(\frac{\log^3 N}{\epsilon^3}\right)$ PQCOND_D queries and decides with probability at least $2/3$ whether*

- $|D - D^*| = 0$ *(i.e. $D = D^*$), or*
- $|D - D^*| \geq \epsilon$,

provided that it is guaranteed that one of these is true.

Proof. We replace the calls to COMPARE with the corresponding calls to QCOMPARE in Algorithm 5 of [10]. □

Remark: The corresponding classical algorithm (Algorithm 5 in [10]) makes $\tilde{O}\left(\frac{\log^2(N/\epsilon)}{\epsilon^2}\right)$ calls to both the COMPARE procedure (with distance parameter $\tilde{O}\left(\frac{\epsilon}{\log(N/\epsilon)}\right)$, range parameter 2, and failure probability parameter $\tilde{O}\left(\frac{\epsilon^2}{\log^2(N/\epsilon)}\right)$) and the SAMP_D oracle, and in total uses $\tilde{O}\left(\frac{\log^4 N}{\epsilon^4}\right)$ PCOND_D queries.

Corollary 4.3. *Given* PQCOND *access to probability distributions $D^{(1)}$ and $D^{(2)}$ over $[N]$, there exists an algorithm that decides, with probability at least $2/3$, whether*

- $|D^{(1)} - D^{(2)}| = 0$ *(i.e. $D^{(1)} = D^{(2)}$), or*
- $|D^{(1)} - D^{(2)}| \geq \epsilon$,

provided that it is guaranteed that one of these is true. The algorithm uses $\tilde{O}\left(\frac{\log^4 N}{\epsilon^{14}}\right)$ $\mathsf{PQCOND}_{D^{(1)}}$ and $\mathsf{PQCOND}_{D^{(2)}}$ queries.

Proof. We replace the calls to COMPARE with the corresponding calls to QCOMPARE in Algorithm 9 of [10]. □

Remark: The corresponding classical algorithm (Algorithm 9 in [10]) makes $\tilde{O}\left(\frac{\log^2 N}{\epsilon^7}\right)$ calls to the $\mathrm{COMPARE}_{D^{(1)}}$ and $\mathrm{COMPARE}_{D^{(2)}}$ procedures (crucially with distance parameter $\tilde{O}\left(\frac{\epsilon^7}{\log^2 N}\right)$,

range parameter 4, and failure probability parameter $\tilde{O}\left(\frac{\epsilon^7}{\log^2 N}\right)$) and the $\mathsf{SAMP}_{D^{(1)}}$ and $\mathsf{SAMP}_{D^{(2)}}$ oracles, and in total uses $\tilde{O}\left(\frac{\log^6 N}{\epsilon^{21}}\right)$ $\mathsf{PCOND}_{D^{(1)}}$ and $\mathsf{PCOND}_{D^{(2)}}$ queries.

Corollary 4.4. *Given* PQCOND *access to a probability distribution D over $[N]$, there exists an algorithm that uses $\tilde{O}(1/\epsilon^{13})$ queries and outputs a value $\hat{d}$ such that $|\hat{d} - |D - \mathcal{A}^{(N)}|| = O(\epsilon)$.*

Proof. We replace the calls to COMPARE with the corresponding calls to QCOMPARE in Algorithm 11 of [10]. In addition, we trivially replace all queries to the SAMP_D oracle with queries to PQCOND_D with query set $[N]$. □

Remark: The corresponding classical algorithm (Algorithm 11 in [10]) uses $\tilde{O}(1/\epsilon^{20})$ queries.

5 Property testing of Boolean functions

The results in Section 4 can be applied to test properties of Boolean functions. One challenge in the field of cryptography is determining whether or not a given boolean function is 'balanced'. We present an algorithm to solve this problem with a constant number of PQCOND queries.

Consider a function $f : \{0,1\}^n \to \{0,1\}^m$, for $n, m \in \mathbb{N}$ with $n \geq m$. If $m = 1$, we might consider the following problem:

Problem 5.1 (Constant-balanced problem)**.** *Given $f : \{0,1\}^n \to \{0,1\}$, decide whether*

- *f is a balanced function, i.e. $|\{x \in \{0,1\}^n : f(x) = 0\}|/2^n = |\{x \in \{0,1\}^n : f(x) = 1\}|/2^n = \frac{1}{2}$, or*
- *f is a constant function, i.e. $f(x) = 0 \ \ \forall x \in \{0,1\}^n$ or $f(x) = 1 \ \ \forall x \in \{0,1\}^n$,*

provided that it is guaranteed that f satisfies one of these conditions.

With standard quantum oracle access to f, this problem can be solved exactly with one query, through use of the Deutsch-Jozsa algorithm [16, 18]. Consider the following extension of this problem:

Problem 5.2. *Given $f : \{0,1\}^n \to \{0,1\}$, write $F_i := |\{x \in \{0,1\}^n : f(x) = i\}|/2^n$. Decide whether*

- *f is a balanced function, i.e. $F_0 = F_1 = \frac{1}{2}$, or*
- *f is ϵ-far from balanced, i.e. $\left|F_0 - \frac{1}{2}\right| + \left|F_1 - \frac{1}{2}\right| = 2\left|F_0 - \frac{1}{2}\right| \geq \epsilon$,*

provided that it is guaranteed that f satisfies one of these conditions.

This problem can be solved classically with bounded probability by querying f $O(1/\epsilon^2)$ times to estimate F_0 to error $\epsilon/3$ (Theorem 2.8 in [5], proved by using a simple Chernoff bound in combination with the Chebyshev inequality).

Now we consider an even more general problem:

Problem 5.3. *Given* $f : \{0,1\}^n \to \{0,1\}^m$, *write* $F_i := |\{x \in \{0,1\}^n : f(x) = i\}|/2^n$. *Decide whether*

- f *is a balanced function, i.e.* $F_i = \frac{1}{2^m} \quad \forall i \in \{0,1\}^m$, *or*
- f *is* ϵ*-far from any balanced function, i.e.* $\sum_{i\in\{0,1\}^m} \left|F_i - \frac{1}{2^m}\right| \geq \epsilon$,

provided that it is guaranteed that f *satisfies one of these conditions.*

By allowing PQCOND access to f, this can be solved in $\tilde{O}(1/\epsilon)$ queries. In what sense do we allow PQCOND access to f? We relate f to a probability distribution by setting $N = 2^m$, $D(i) = F_i$ (i.e. D is the probability distribution formed from the image of f), and using the definition of $D_S(i)$ given at the start of Section 2. The problem is now a question of uniformity testing, and is solved by an application of the algorithm presented in Corollary 4.1. Using the standard quantum oracle QSAMP, this problem requires $\Omega(2^{m/3})$ queries (a lower bound for uniformity testing given in [13]).

The problem does not naturally lend itself to the classical COND model, as our solution makes use of the mapping O_D (see Definition 2.2). Using the standard classical sampling oracle SAMP, this problem requires $\Omega(2^{m/2})$ queries [5].

6 Mixedness Testing

Recall the description of Mixedness testing presented in Section 1.1.

Given an n-dimensional quantum state $\rho \in \mathbb{C}^n \times \mathbb{C}^n$ and a basis $\mathcal{B} = \{|b_i\rangle\}_{i\in[n]}$ where n is even, let $D^{(\rho,\mathcal{B})}_{[n]}$ be the probability distribution over $[n]$ defined by $D^{(\rho,\mathcal{B})}_{[n]}(i) := \mathrm{Tr}(\rho\,|b_i\rangle\langle b_i|) = \langle b_i|\,\rho\,|b_i\rangle$. It is easy to see that for any basis $\mathcal{B}$, $D^{\mathbb{1}/n,\mathcal{B}}_{[n]} = \mathcal{A}^{(n)}$, where $\mathcal{A}^{(n)}$ is the uniform distribution over $[n]$. Then for any state ρ,

- if $\|\rho - \mathbb{1}/n\|_1 = 0$, then $\left|D^{\rho,\mathcal{B}}_{[n]} - \mathcal{A}^{(n)}\right| = 0$ for any basis $\mathcal{B}$;
- if $\|\rho - \mathbb{1}/n\|_1 \geq \epsilon$, perhaps we can choose a basis $\mathcal{B}$ such that $\left|D^{\rho,\mathcal{B}}_{[n]} - \mathcal{A}^{(n)}\right| \geq \nu(\epsilon, n)$, for some function ν.

Corollary 4.1, with distance parameter $\nu(\epsilon, n)$, could then be used to distinguish between these two options.

As the first case above is immediate, we henceforth assume that $\|\rho - \mathbb{1}/n\|_1 \geq \epsilon$. In order to simplify the analysis, we assume that n is even, let $\Delta = \rho - \mathbb{1}/n$, and introduce

$$\delta^{(\mathcal{B})} := \left|D^{\rho,\mathcal{B}}_{[n]} - \mathcal{A}^{(n)}\right| = \sum_{i\in[n]} |\,\langle b_i|\,\Delta\,|b_i\rangle\,| \geq 0.$$

Let $\tilde{\mathcal{B}} = \left\{\left|\tilde{b}_i\right\rangle\right\}_{i\in[n]}$ be the eigenbasis of Δ, and let $d_i := \left\langle\tilde{b}_i\right|\Delta\left|\tilde{b}_i\right\rangle$, $i \in [n]$ be the eigenvalues. Thus, $\Delta = \sum_{i\in[n]} d_i \left|\tilde{b}_i\right\rangle\left\langle\tilde{b}_i\right|$. Note that $\mathrm{Tr}\,\Delta = \sum_{i\in[n]} d_i = 0$, and also $\eta := \|\rho - \mathbb{1}/n\|_1 = \|\Delta\|_1 = \sum_{i\in[n]} |d_i| \geq \epsilon$.

Now suppose we choose a basis $\mathcal{B} = \{|b_i\rangle\}_{i\in[n]}$ uniformly at random, i.e. we choose $W \in \mathcal{U}(n)$ uniformly at random according to the Haar measure, and set $|b_i\rangle = W|\tilde{b}_i\rangle$. Then

$$\delta^{(\mathcal{B})} = \sum_{i\in[n]} |\langle b_i|\Delta|b_i\rangle| = \sum_{i\in[n]} \left|\langle \tilde{b}_i|W^\dagger \Delta W|\tilde{b}_i\rangle\right|. \tag{1}$$

Theorem 8 in [23] provides the bound

$$\mathbb{E}\left(\delta^{(\mathcal{B})}\right) \geq \frac{\|\Delta\|_2}{3}.$$

Since $\|\Delta\|_2 \geq \frac{\|\Delta\|_1}{\sqrt{n}} = \frac{\eta}{\sqrt{n}}$, we see that

$$\mathbb{E}\left(\delta^{(\mathcal{B})}\right) \geq \frac{\eta}{3\sqrt{n}}.$$

Remark: We can provide a tighter bound asymptotically, which is derived in Section C.1.

The following lemma allows us to relate this lower bound on $\mathbb{E}\left(\delta^{(\mathcal{B})}\right)$ to a lower bound on $\mathbb{P}[\delta^{(\mathcal{B})} \geq \lambda]$, for some λ.

Lemma 6.1.

$$\mathbb{P}\left[\delta^{(\mathcal{B})} \geq \lambda\right] \geq \frac{1}{\eta}\left(\mathbb{E}\left(\delta^{(\mathcal{B})}\right) - \lambda\right)$$

Proof. Let $p = p(\mu)$ be the probability density function for $\delta^{(\mathcal{B})}$. As noted in eq. (C.1), $0 \leq \delta^{(\mathcal{B})} \leq \eta$. Thus, for $\lambda \in [0, \eta]$ we can write

$$\begin{aligned}
\mathbb{E}\left(\delta^{(\mathcal{B})}\right) &= \int_0^\eta \mu p(\mu)\, d\mu \\
&= \int_0^\lambda \mu p(\mu)\, d\mu + \int_\lambda^\eta \mu p(\mu)\, d\mu \\
&\leq \int_0^\lambda \lambda p(\mu)\, d\mu + \int_\lambda^\eta \eta p(\mu)\, d\mu \\
&\leq \lambda + \eta \mathbb{P}\left[\delta^{(\mathcal{B})} \geq \lambda\right].
\end{aligned}$$

Rearranging the inequality gives the result. □

Applying this lemma, we deduce

$$\mathbb{P}\left[\delta^{(\mathcal{B})} \geq \lambda\right] \geq \frac{1}{3\sqrt{n}} - \frac{\lambda}{\eta}.$$

Setting $\lambda = \frac{\min(1,\epsilon)}{6\sqrt{n}}$ and recalling that $\epsilon \leq \eta$ gives

$$\mathbb{P}\left[\delta^{(\mathcal{B})} \geq \frac{\min(1,\epsilon)}{6\sqrt{n}}\right] \geq \frac{1}{6\sqrt{n}}.$$

Suppose we repeat this test k times, choosing different bases $\mathcal{B}_1, \ldots, \mathcal{B}_k$ uniformly at random according to the Haar measure on $\mathcal{U}(n)$. We call $\mathcal{B}$ 'good' if $\delta^{(\mathcal{B})} \geq \frac{\min(1,\epsilon)}{6\sqrt{n}}$. Let $K(k)$ represent the event that at least one of $\mathcal{B}_1, \ldots, \mathcal{B}_k$ is 'good'. Then

$$\mathbb{P}[K(k)] \geq 1 - \left(1 - \frac{1}{6\sqrt{n}}\right)^k.$$

Setting $k = 24\sqrt{n}$ gives

$$\mathbb{P}[K(24\sqrt{n})] \geq 1 - \frac{1}{e^4} \geq \frac{49}{50}.$$

6.1 *Executing the algorithm*

Suppose we run the above algorithm l times in total. Then by using a Chernoff bound (eq. (1) in [10]) and considering Corollary 4.1, it follows that

- if the distributions are 'equal', $\mathbb{P}\left[\text{algorithm outputs Equal} \geq \frac{1}{2}l \text{ times}\right] \geq 1 - e^{-l/18}$;
- if the distributions are 'far', $\mathbb{P}\left[\text{algorithm outputs Far} \geq \frac{1}{2}l \text{ times}\right] \geq 1 - e^{-l/18}$.

The full algorithm has been set out below.

Algorithm 2 MAXIMALLYMIXEDSTATETEST(ρ)

Input: PQCOND access to a probability distribution $D_{[n]}^{(\rho,\mathcal{B})}$ over $[n]$ for any $\mathcal{B}$, as described in Section 6, and parameter ϵ. Set $l = 18\log(72\sqrt{n})$.

1. Choose $k = 24\sqrt{n}$ bases $\mathcal{B}_1, \ldots \mathcal{B}_k$ uniformly at random.
2. For each $j = 1, \ldots, k$, run the algorithm given in Corollary 4.1 on the distribution $D_{[n]}^{(\rho,\mathcal{B}_j)}$ l times with distance parameter $\frac{\min(1,\epsilon)}{6\sqrt{n}}$, returning $u_j = 1$ if at least $\frac{1}{2}l$ of the runs return Far, and $u_j = 0$ otherwise.
3. If any u_j is equal to 1, output Far, otherwise output Equal.

The analysis of this algorithm is separated into two cases:

- $\|\rho - \mathbb{1}/n\|_1 = 0$: The probability that a particular u_j is equal to 1 in Step 2 is less than $e^{-l/18}$. Thus, the probability of the algorithm failing is, by the union bound[c], at most $(24\sqrt{n})\, e^{-l/18} = \frac{1}{3}$, and hence the algorithm outputs Equal with probability at least $\frac{2}{3}$.
- $\|\rho - \mathbb{1}/n\|_1 \geq \epsilon$: Suppose that $\mathcal{B}_j$ is 'good'. Then with probability at least $1 - e^{l/18} \geq \frac{99}{100}$, we get $u_j = 1$, and the algorithm will output Far in Step 3. The probability that one of $\mathcal{B}_1, \ldots, \mathcal{B}_k$ is 'good' is at least $\frac{49}{50}$, and hence the probability that the entire algorithm outputs Far is at least $0.97 \geq \frac{2}{3}$.

[c] For a countable set of events $A_1, A_2, \ldots$, we have that $\mathbb{P}\left[\bigcup_i A_i\right] \leq \sum_i \mathbb{P}[A_i]$.

Each run of the algorithm given in Corollary 4.1 requires $\tilde{O}(6\sqrt{n}/\epsilon)$ PQCOND queries if $\epsilon \leq 1$, and hence in total Algorithm 2 requires

$$\tilde{O}\left(\frac{6\sqrt{n}}{\epsilon}kl\right) = \tilde{O}\left(\frac{n}{\epsilon}\right)$$

PQCOND queries.

7 Discussion

Quantum conditional oracles give us new insights into the kinds of information that are useful for testing properties of distributions. In many circumstances such oracles serve as natural models for accessing information. In addition, they are able to demonstrate separations in query complexity between a number of problems, thereby providing interesting new perspectives on information without trivialising the set-up. We now mention some open questions.

Group testing and pattern matching are further important areas to which our notion of a quantum conditional oracle could be applied. The structure of questions commonly considered there suggest that use of PQCOND would decrease the query complexity dramatically for many practically relevant problems compared to the best known quantum and classical algorithms [1, 17, 3, 8].

In our algorithms, we have made particular use of the PQCOND oracle, the quantum analogue of the PCOND oracle. It is noted in [10] that the unrestricted COND oracle offers significant advantages over the PCOND oracle for many problems, and it is possible that similar improvements could be achieved for some quantum algorithms through use of the unrestricted QCOND oracle.

We believe that a more detailed analysis of the Mixedness problem from Section 6 will yield an algorithm requiring only $\tilde{O}(\sqrt{n}/\epsilon)$ queries. Through a slightly different approach we have proved this up to a small conjecture.

The algorithm that we present for testing Mixedness (Algorithm 2) chooses several bases $\mathcal{B}_1, \ldots, \mathcal{B}_k$ independently and uniformly at random. It remains open, however, whether or not a more adaptive approach to choosing bases will yield an algorithm requiring fewer queries.

Our definition of the spectrum testing problem in Section 6 made use of the trace norm, $\|\cdot\|_1$. One might wonder how the query complexity would be affected if the problem were defined with a different norm, such as the operator norm,[d] $\|\cdot\|_\infty$. Numerical simulations and limited analysis suggest that the probability of picking a 'good' basis $\mathcal{B}$ tends to 1 as $n \to \infty$, and hence that the number of queries required to distinguish between the two options would be independent of n. We leave the proof of this conjecture as an open question.

Acknowledgements

[d] For an $(n \times n)$ matrix A, $\|A\|_\infty = \max_{i \in [n]} a_i$, where the a_i are the singular values of A.

I.S.B.S. thanks EPSRC for financial support. S.S. acknowledges the support of Sidney Sussex College.

References

1. Problem 33: Group Testing - Open Problems in Sublinear Algorithms. *Sublinear.info*.
2. Jayadev Acharya, Clement Canonne, and Gautam Kamath. A Chasm Between Identity and Equivalence Testing with Conditional Queries. Technical Report 156, 2014.
3. Andris Ambainis, Aleksandrs Belovs, Oded Regev, and Ronald de Wolf. Efficient Quantum Algorithms for (Gapped) Group Testing and Junta Testing. *arXiv:1507.03126 [quant-ph]*, July 2015. arXiv: 1507.03126.
4. Kendall E Atkinson. *An introduction to numerical analysis*. John Wiley & Sons, 2008.
5. Tugkan Batu, Eldar Fischer, Lance Fortnow, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 442–451. IEEE, 2001.
6. Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing Closeness of Discrete Distributions. *arXiv:1009.5397 [cs, math, stat]*, September 2010. arXiv: 1009.5397.
7. Eric Blais, Joshua Brody, and Kevin Matulef. Property Testing Lower Bounds via Communication Complexity. *computational complexity*, 21(2):311–358, May 2012.
8. Annalisa De Bonis. Constraining the number of positive responses in adaptive, non-adaptive, and two-stage group testing. *Journal of Combinatorial Optimization*, pages 1–34, September 2015.
9. S. Bravyi, A. W. Harrow, and A. Hassidim. Quantum Algorithms for Testing Properties of Distributions. *IEEE Transactions on Information Theory*, 57(6):3971–3981, June 2011.
10. C. Canonne, D. Ron, and R. Servedio. Testing probability distributions using conditional samples. 44(3):540–616.
11. Clément Canonne and Ronitt Rubinfeld. Testing Probability Distributions Underlying Aggregated Data. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming*, number 8572 in Lecture Notes in Computer Science, pages 283–295. Springer Berlin Heidelberg, July 2014. DOI: 10.1007/978-3-662-43948-7_24.
12. S. Chakraborty, E. Fischer, Y. Goldhirsh, and A. Matsliah. On the Power of Conditional Samples in Distribution Testing. *SIAM Journal on Computing*, pages 1261–1296, January 2016.
13. Sourav Chakraborty, Eldar Fischer, Arie Matsliah, and Ronald de Wolf. Quantum queries for testing distributions, 2009.
14. Sourav Chakraborty, Eldar Fischer, Arie Matsliah, and Ronald de Wolf. New Results on Quantum Property Testing. *arXiv:1005.0523 [quant-ph]*, May 2010. arXiv: 1005.0523.
15. Siu-On Chan, Ilias Diakonikolas, Gregory Valiant, and Paul Valiant. Optimal Algorithms for Testing Closeness of Discrete Distributions. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 1193–1203, Philadelphia, PA, USA, 2014. Society for Industrial and Applied Mathematics.
16. Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 454, pages 339–354. The Royal Society, 1998.
17. A. De Bonis, L. Gasieniec, and U. Vaccaro. Optimal Two-Stage Algorithms for Group Testing Problems. *SIAM Journal on Computing*, 34(5):1253–1270, January 2005.
18. David Deutsch and Richard Jozsa. Rapid Solution of Problems by Quantum Computation. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 439(1907):553–558, December 1992.
19. Ilias Diakonikolas, Daniel M. Kane, and Vladimir Nikishkin. Testing Identity of Structured Distributions. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 1841–1854, Philadelphia, PA, USA, 2015. Society for Industrial and Applied Mathematics.
20. Oded Goldreich. *Property Testing: Current Research and Surveys*. Springer, October 2010. Google-

Books-ID: HIdqCQAAQBAJ.
21. Oded Goldreich, Shari Goldwasser, and Dana Ron. Property Testing and Its Connection to Learning and Approximation. *J. ACM*, 45(4):653–750, July 1998.
22. Sudipto Guha, Andrew McGregor, and Suresh Venkatasubramanian. Streaming and Sublinear Approximation of Entropy and Information Distances. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm*, SODA '06, pages 733–742, Philadelphia, PA, USA, 2006. Society for Industrial and Applied Mathematics.
23. Aram W Harrow, Ashley Montanaro, and Anthony J Short. Limitations on quantum dimensionality reduction. In *International Colloquium on Automata, Languages, and Programming*, pages 86–97. Springer, 2011.
24. Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.
25. Ashley Montanaro. Quantum speedup of monte carlo methods. In *Proc. R. Soc. A*, volume 471, page 20150301. The Royal Society, 2015.
26. Ashley Montanaro and Ronald de Wolf. A Survey of Quantum Property Testing. *arXiv:1310.2035 [quant-ph]*, October 2013. arXiv: 1310.2035.
27. Ryan O'Donnell and John Wright. Quantum Spectrum Testing. *arXiv:1501.05028 [quant-ph]*, January 2015. arXiv: 1501.05028.
28. Herbert Robbins. A remark on stirling's formula. *The American Mathematical Monthly*, 62(1):26–29, 1955.
29. Michelle Schatzman. *Numerical Analysis: A Mathematical Introduction*. Clarendon Press, 2002. Google-Books-ID: 3SuNiR1hzxUC.
30. Stanislav Sýkora. Quantum theory and the bayesian inference problems. *Journal of Statistical Physics*, 11(1):17–27.
31. G. Valiant and P. Valiant. The Power of Linear Estimators. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 403–412, October 2011.

Appendix A Efficient comparison of conditional probabilities

A.1 Proof of Theorem 3.6

We prove this case-by-case. We introduce the shorthand $w(X) \coloneqq D_{X\cup Y}(X) = D(X)/D(X\cup Y)$, $w(Y) \coloneqq D_{X\cup Y}(Y) = D(Y)/D(X\cup Y)$ and note that $r_{X,Y} = w(Y)/w(X)$. In addition, since $w(X)+w(Y)=1$, it is straightforward to show the following inequalities for a constant $T \geq 1$:

$$\begin{aligned}
r_{X,Y} \geq \frac{1}{T} \quad &\Longrightarrow \quad w(X) \leq \frac{T}{T+1},\ w(Y) \geq \frac{1}{T+1} \\
r_{X,Y} \leq \frac{1}{T} \quad &\Longrightarrow \quad w(X) \geq \frac{T}{T+1},\ w(Y) \leq \frac{1}{T+1} \\
r_{X,Y} \geq T \quad &\Longrightarrow \quad w(X) \leq \frac{1}{T+1},\ w(Y) \geq \frac{T}{T+1} \\
r_{X,Y} \leq T \quad &\Longrightarrow \quad w(X) \geq \frac{1}{T+1},\ w(Y) \leq \frac{T}{T+1}
\end{aligned} \tag{A.1}$$

The strict versions of these inequalities also hold true.

1. $\boldsymbol{1/K \leq r_{X,Y} \leq K}$

   In this case we wish our algorithm to output $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$.

From eq. (A.1), we immediately have that

$$\frac{1}{K+1} \leq w(X), w(Y) \leq \frac{K}{K+1}. \tag{A.2}$$

Steps 2 and 3 use AddEstProbQCond to estimate $w(X)$ and $w(Y)$ to within additive error $\eta/3$ with probability at least $1-\delta/4$. As stated in Theorem 3.4, this requires

$$O\left(\max\left(\frac{\sqrt{w(X)}}{\eta}, \frac{1}{\sqrt{\eta}}\right)\log(1/\delta)\right) = O\left(\frac{\log(1/\delta)}{\eta}\right)$$

queries to QCOND_D, where the equality is due to the fact that $w(X) \leq 1$, and thus M (defined in Algorithm 1) queries suffice.

Step 4 uses MulEstProbQCond to estimate $w(X)$ to within multiplicative error $\eta/3$ with probability at least $1-\delta/4$. From Theorem 3.5, we clearly require

$$O\left(\frac{\log(1/\delta)}{\eta\sqrt{w(X)}}\right) = O\left(\frac{\sqrt{K}\log(1/\delta)}{\eta}\right)$$

queries to QCOND_D in order to achieve these, where the equality is due to eq. (A.2), and thus M queries suffice. Step 5 requires the same number of queries.

With a combined probability of at least $1-\delta$, Steps 2–5 all pass, and produce the following values:

$$\begin{aligned}
\tilde{w}_+(X) &\in [w(X)-\eta/3, w(X)+\eta/3],\\
\tilde{w}_+(Y) &\in [w(Y)-\eta/3, w(Y)+\eta/3],\\
\tilde{w}_\times(X) &\in [1-\eta/3, 1+\eta/3]w(X),\\
\tilde{w}_\times(Y) &\in [1-\eta/3, 1+\eta/3]w(Y).
\end{aligned}$$

From eq. (A.2), we see that

$$\tilde{w}_+(X), \tilde{w}_+(Y) \leq \frac{K}{K+1} + \frac{\eta}{3} < \frac{3K}{3K+1} - \frac{\eta}{3},$$

where the final inequality is due to the algorithm's requirement that $\frac{\eta}{3} < \frac{1}{8K}$.

Thus, the checks in Steps 6 and 7 pass, and Step 8 gives us

$$\tilde{r}_{X,Y} \in [1-\eta, 1+\eta]r_{X,Y}.$$

2. $\boldsymbol{K < r_{X,Y}}$

   This is split into two sub-cases.

   (a) $\boldsymbol{3K < r_{X,Y}}$

   In this case we wish our algorithm to output High.

From eq. (A.1) we have that

$$w(X) < \frac{1}{3K+1}, \quad w(Y) > \frac{3K}{3K+1}. \tag{A.3}$$

As in Case 1, Steps 2 and 3 allow us to gain

$$\tilde{w}_+(X) \in [w(X) - \eta/3, w(X) + \eta/3],$$
$$\tilde{w}_+(Y) \in [w(Y) - \eta/3, w(Y) + \eta/3],$$

with combined probability at least $1 - \delta/2$. (We henceforth assume that we have gained such values.)

Using eq. (A.3) it is easy to show that $\tilde{w}_+(X) < \frac{3K}{3K+1} - \frac{\eta}{3}$ and that $\tilde{w}_+(Y) > \frac{3K}{3K+1} - \frac{\eta}{3}$. Hence the check in Step 6 passes, but the check in Step 7 fails, and the algorithm outputs High and exits.

(b) $\boldsymbol{K < r_{X,Y} \leq 3K}$

In this case we wish our algorithm to either output High *or output* $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$.

From eq. (A.1), we have that

$$\frac{1}{3K+1} \leq w(X) < \frac{1}{K+1}, \quad \left(\frac{1}{3K+1} <\right) \frac{K}{1+K} < w(Y) \leq \frac{3K}{3K+1}. \tag{A.4}$$

Thus, with $O(\sqrt{K}\log(1/\delta)/\eta)$ queries, as in Case 1, we gain

$$\tilde{w}_+(X) \in [w(X) - \eta/3, w(X) + \eta/3],$$
$$\tilde{w}_+(Y) \in [w(Y) - \eta/3, w(Y) + \eta/3],$$
$$\tilde{w}_\times(X) \in [1 - \eta/3, 1 + \eta/3] w(X),$$
$$\tilde{w}_\times(Y) \in [1 - \eta/3, 1 + \eta/3] w(Y),$$

with combined probability at least $1 - \delta$. (We henceforth assume that we have gained such values.)

Using eq. (A.4), we see that $\tilde{w}_+(X) < \frac{3K}{3K+1} - \frac{\eta}{3}$, and thus Step 6 will pass.

Assuming the check in Step 7 passes, Step 8 will output $\tilde{r}_{X,Y} \in [1-\eta, 1+\eta] r_{X,Y}$.

However, given the upper bound for $w(Y)$ in eq. (A.4), it is possible to have $\tilde{w}_+(Y) > \frac{3K}{3K+1} - \frac{\eta}{3}$, causing the check in Step 7 to fail and the algorithm to output High.

3. $\boldsymbol{r_{X,Y} < 1/K}$

This is split into two sub-cases.

(a) $\boldsymbol{r_{X,Y} < 1/(3K)}$

This is equivalent to the condition that $3K < r_{Y,X}$, and thus follows the same argument as Case 2a, with X and Y interchanged and an output of Low instead of High.

(b) $\boldsymbol{1/(3K) \leq r_{X,Y} < 1/K}$

This is equivalent to the condition that $K < r_{Y,X} \leq 3K$, and thus follows the same argument as Case 2b, with X and Y interchanged and an output of Low instead of High.

□

Appendix B Property testing of probability distributions

B.1 An illustrative example

This section describes the intuition behind Corollary 4.1, and presents a simpler but slightly weaker algorithm requiring $\tilde{O}(1/\epsilon^4)$ queries to the PCOND *oracle. This algorithm is presented in [10], though here we give a more in-depth derivation.*

Let $\mathcal{A}^{(N)}$ be the uniform distribution on $[N]$ (i.e. $\mathcal{A}^{(N)}(i) = 1/N, i \in [N]$). Given PCOND access to a probability distribution D over $[N]$, we wish to decide (with high probability) whether

- $|D - \mathcal{A}^{(N)}| = 0$ (i.e. $D = \mathcal{A}^{(N)}$), or
- $|D - \mathcal{A}^{(N)}| \geq \epsilon$,

provided that it is guaranteed that one of these is true.

Suppose that the latter option is true, i.e. D is ϵ-far from uniform.

We now partition our domain into two sets: elements of weight at least $1/N$; and elements of weight less than $1/N$. More formally, we define

$$H := \left\{ h \in [N] : D(h) \geq \frac{1}{N} \right\}, \qquad L := \left\{ l \in [N] : D(l) < \frac{1}{N} \right\}$$

Proposition 7.1.

$$\sum_{h \in H} \left(D(h) - \frac{1}{N} \right) = \sum_{l \in L} \left(\frac{1}{N} - D(l) \right) \geq \frac{\epsilon}{2}$$

Proof. First, note that $\sum_{i\in[N]} D(i) = 1$ and thus

$$\begin{aligned}
0 &= \sum_{i\in[N]} \left(D(i) - \frac{1}{N} \right) \\
&= \sum_{h\in H} \left(D(h) - \frac{1}{N} \right) + \sum_{l\in L} \left(D(l) - \frac{1}{N} \right) \\
&= \sum_{h\in H} \left(D(h) - \frac{1}{N} \right) - \sum_{l\in L} \left(\frac{1}{N} - D(l) \right)
\end{aligned}$$

and the equality follows.

Since D is ϵ-far from uniform, we have that

$$\begin{aligned}
\epsilon &\leq \sum_{i\in[N]} \left| D(i) - \frac{1}{N} \right| \\
&= \sum_{h\in H} \left| D(h) - \frac{1}{N} \right| + \sum_{l\in L} \left| D(l) - \frac{1}{N} \right| \\
&= 2 \sum_{h\in H} \left| D(h) - \frac{1}{N} \right| = 2 \sum_{l\in L} \left| \frac{1}{N} - D(l) \right|
\end{aligned}$$

and the inequality follows. □

We define the 'significantly heavy' and 'significantly light' sets

$$\begin{aligned}
H' &:= \left\{ h \in [N] : D(h) \geq \frac{1}{N} + \frac{\epsilon}{4N} \right\} \subseteq H, \\
L' &:= \left\{ l \in [N] : D(h) < \frac{1}{N} - \frac{\epsilon}{4N} \right\} \subseteq L
\end{aligned}$$

Now,

$$\begin{aligned}
\frac{\epsilon}{2} &\leq \sum_{h\in H} \left(D(h) - \frac{1}{N} \right) \\
&= \sum_{h\in H'} \left(D(h) - \frac{1}{N} \right) + \sum_{h\in H\setminus H'} \underbrace{\left(D(h) - \frac{1}{N} \right)}_{<\frac{\epsilon}{4N}} \\
&< D(H') - \frac{|H'|}{N} + \frac{\epsilon}{4N} (\underbrace{|H|}_{\leq N} - |H'|) \\
&\leq D(H') + \frac{\epsilon}{4} - \left(\frac{|H'|}{N} + \frac{\epsilon |H'|}{4N} \right) \\
&\leq D(H') + \frac{\epsilon}{4},
\end{aligned}$$

and hence

$$D(H') > \frac{\epsilon}{2} - \frac{\epsilon}{4} = \frac{\epsilon}{4}.$$

And,

$$\begin{aligned}
\frac{\epsilon}{2} &\leq \sum_{l \in L} \left(\frac{1}{N} - D(l) \right) \\
&= \sum_{l \in L'} \left(\frac{1}{N} - D(l) \right) + \sum_{l \in L \setminus L'} \underbrace{\left(\frac{1}{N} - D(l) \right)}_{\leq \frac{\epsilon}{4N}} \\
&\leq \frac{|L'|}{N} - D(L') + \frac{\epsilon}{4N} (\underbrace{|L|}_{\leq N} - |L'|) \\
&\leq \frac{|L'|}{N} + \frac{\epsilon}{4} - \left(D(L') + \frac{\epsilon |L'|}{4N} \right) \\
&\leq \frac{|L'|}{N} + \frac{\epsilon}{4},
\end{aligned}$$

and thus

$$|L'| \geq \frac{N\epsilon}{4}.$$

We can obtain an element of L' with high probability by sampling from $\mathsf{SAMP}_{\mathcal{A}^{(N)}}$ $O(1/\epsilon)$ times, and we can obtain an element of H' with high probability by sampling from SAMP_D $O(1/\epsilon)$ times. These elements will have a multiplicative difference of at least $\frac{1/N + \epsilon/(4N)}{1/N - \epsilon(4N)} \geq 1 + \frac{\epsilon}{2}$, which can be detected with high probability by using the COMPARE procedure with parameters, say, $\eta = \epsilon/100$ and $K = 2$, requiring $\tilde{O}(1/\epsilon^2)$ PCOND_D queries.

Since there will be $O(1/\epsilon^2)$ pairs to test, and each use of COMPARE requires $\tilde{O}(1/\epsilon^2)$ queries, the overall sample complexity of the algorithm will be $\tilde{O}(1/\epsilon^4)$.

Appendix C Mixedness Testing

C.1 A tighter asymptotic bound on $\mathbb{E}\left(\delta^{(\mathcal{B})}\right)$

This section provides a derivation of the asymptotic bound $\mathbb{E}\left(\delta^{(\mathcal{B})}\right) \gtrsim \frac{\eta}{\sqrt{2\pi n}}$.

From eq. (1), we have

$$\delta^{(\mathcal{B})} = \sum_{i \in [n]} \left| \sum_{j \in [n]} |W_{ji}|^2 d_j \right| \leq \sum_{j \in [n]} \left(\sum_{i \in [n]} |W_{ji}|^2 \right) |d_j| = \eta. \tag{C.1}$$

by the triangle inequality.

Let $v_j^{(i)} = |W_{ji}|^2$, introduce the vector $V^{(i)} = (v_0^{(i)}, \ldots, v_{n-1}^{(i)})$, and write $d = (d_0, \ldots, d_{n-1})$. Then

$$\delta^{(\mathcal{B})} = \sum_{i \in [n]} |V^{(i)} \cdot d|.$$

We now make use of Sykora's theorem [30], which states that if W is chosen uniformly at random according to the Haar measure on $\mathcal{U}(n)$, then the vector $V^{(i)}$, for any i, is uniformly distributed over the probability simplex

$$T_n = \{(v_0, \ldots, v_{n-1}) : v_i \in [0,1], \textstyle\sum_{i \in [n]} v_i = 1\}.$$

Since all of the $V^{(i)}$'s have the same distribution, we see that

$$\mathbb{E}\left(\delta^{(\mathcal{B})}\right) = n\mathbb{E}(|V \cdot d|),$$

where V is a generic $V^{(i)}$.

We now write $\mathbb{E}(|V \cdot d|)$ as an integral over the probability simplex T_n. We have

$$\mathbb{E}(f(V)) = \int_{T_n} f(V) dV \coloneqq (n-1)! \int_{v_0=0}^{1} \cdots \int_{v_{n-1}=0}^{1} \delta(1 - \textstyle\sum_{i \in [n]} v_i) f(V)\, dv_0 \cdots dv_{n-1}$$

where $dV = (n-1)!\, \delta(1 - \sum_{i \in [n]} v_i)\, dv_0 \cdots dv_{n-1}$ is the normalised measure on T_n, defined so that $\mathbb{E}(1) = 1$.

Note that the integral expression for $\mathbb{E}(|V \cdot d|) = \mathbb{E}(|v_0 d_0 + \cdots + v_{n-1} d_{n-1}|)$ is completely symmetric in the v_i's (and hence in the d_i's). Thus, if σ is a permutation on $[n]$, we have that

$$\mathbb{E}(|v_0 d_0 + \cdots + v_{n-1} d_{n-1}|) = \mathbb{E}(|v_0 d_{\sigma(0)} + \cdots + v_{n-1} d_{\sigma(n-1)}|).$$

Using this observation, we can write

$$\begin{aligned}
&\mathbb{E}(|v_0 d_0 + \cdots + v_{n-1} d_{n-1}|) \\
&= \frac{1}{n} \Big[\mathbb{E}(|v_0 d_{\sigma(0)} + \cdots + v_{n-1} d_{\sigma(n-1)}|) + \mathbb{E}(|v_0 d_{\sigma(1)} + \cdots + v_{n-1} d_{\sigma(0)}|) \\
&\qquad + \mathbb{E}(|v_0 d_{\sigma(2)} + \cdots + v_{n-1} d_{\sigma(1)}|) + \cdots + \mathbb{E}(|v_0 d_{\sigma(n-1)} + \cdots + v_{n-1} d_{\sigma(n-2)}|)\Big] \\
&= \frac{1}{n} \Big[\mathbb{E}(|v_0 d_{\sigma(0)} + \cdots + v_{n-1} d_{\sigma(n-1)}|) + \mathbb{E}(|-v_0 d_{\sigma(1)} - \cdots - v_{n-1} d_{\sigma(0)}|) \qquad \text{(C.2)} \\
&\qquad + \mathbb{E}(|v_0 d_{\sigma(2)} + \cdots + v_{n-1} d_{\sigma(1)}|) + \cdots + \mathbb{E}(|-v_0 d_{\sigma(n-1)} - \cdots - v_{n-1} d_{\sigma(n-2)}|)\Big] \\
&\geq \frac{1}{n} \mathbb{E}\Big[|v_0(d_{\sigma(0)} - d_{\sigma(1)} + \cdots - d_{\sigma(n-1)}) + v_1(d_{\sigma(1)} - d_{\sigma(2)} + \cdots - d_{\sigma(0)}) \qquad \text{(C.3)} \\
&\qquad + v_2(d_{\sigma(2)} - d_{\sigma(3)} + \cdots - d_{\sigma(1)}) + \cdots v_{n-1}(d_{\sigma(n-1)} - d_{\sigma(0)} + \cdots - d_{\sigma(n-2)})|\Big] \\
&= \frac{1}{n} |d_{\sigma(0)} - d_{\sigma(1)} + d_{\sigma(2)} - \cdots - d_{\sigma(n-1)}| \;\; \mathbb{E}(|v_0 - v_1 + v_2 - \cdots - v_{n-1}|),
\end{aligned}$$

where in eq. (C.2) minus signs are added inside every other expectation (note that n is even), and eq. (C.3) is derived using the triangle inequality.

Since σ was an arbitrary permutation, we can instead write

$$\mathbb{E}(|V \cdot d|) \geq \frac{1}{n} \left[\max_{\sigma \in Sym([n])} |d_{\sigma(0)} - d_{\sigma(1)} + d_{\sigma(2)} - \cdots - d_{\sigma(n-1)}| \right] \ \mathbb{E}(|v_0 - v_1 + v_2 - \cdots - v_{n-1}|),$$

where $Sym([n])$ is the symmetric group on $[n]$, and hence

$$\mathbb{E}\left(\delta^{(\mathcal{B})}\right) \geq M^{(d)} E_n,$$

where

$$M^{(d)} := \max_{\sigma \in Sym([n])} |d_{\sigma(0)} - d_{\sigma(1)} + d_{\sigma(2)} - \cdots - d_{\sigma(n-1)}|, \tag{C.4}$$

$$E_n := \mathbb{E}(|v_0 - v_1 + v_2 - \cdots - v_{n-1}|). \tag{C.5}$$

Evaluation of $M^{(d)}$ and E_n is carried out below in Sections C.1.1 and C.1.2, where we find that $M^{(d)} \geq \frac{1}{2}\eta$ and $E_n \sim \sqrt{\frac{2}{\pi}} \frac{1}{\sqrt{n}}$ for large n. Hence

$$\mathbb{E}\left(\delta^{(\mathcal{B})}\right) \gtrsim \frac{\eta}{\sqrt{2\pi n}}.$$

C.1.1 Evaluating $M^{(d)}$

This section provides a lower bound for the quantity $M^{(d)}$, as defined in eq. (C.4).

Let D^+ be the set of non-negative d_i's, labelled such that $d_0^+ \geq d_1^+ \geq \cdots$, and similarly let D^- be the set of negative d_i's, labelled such that $d_0^- \leq d_1^- \leq \cdots$. w.l.o.g. suppose $|D^-| \geq |D^+|$.

Let $|D^+| = \frac{n}{2} - k$, where $k \leq \frac{n}{2}$. Thus $|D^-| = \frac{n}{2} + k$. Note that $\sum_i d_i^+ = -\sum_i d_i^- = \frac{1}{2}\eta$.

We now define σ so that the following statements are true:

- $d_{\sigma(1)} = d_0^-, d_{\sigma(3)} = d_1^-, \ldots, d_{\sigma(n-1)} = d_{\frac{n}{2}-1}^-$;
- $d_{\sigma(0)} = d_0^+, d_{\sigma(2)} = d_1^+, \ldots, d_{\sigma(n-2k-2)} = d_{\frac{n}{2}-k-1}^+$;
- $d_{\sigma(n-2k)}, d_{\sigma(n-2k+2)}, \ldots, d_{\sigma(n-2)}$ can be filled with the remaining members of D^-.

Then

- $d_{\sigma(0)} + d_{\sigma(2)} + \cdots + d_{\sigma(n-2k-2)} = \frac{1}{2}\eta$;
- $d_0^-, \ldots, d_{\frac{n}{2}-1}^- \leq d_{\frac{n}{2}-1}^- \implies -d_{\sigma(1)} - d_{\sigma(3)} - \cdots - d_{\sigma(n-1)} \geq -\frac{n}{2} d_{\frac{n}{2}-1}^-$;
- $d_{\frac{n}{2}}^-, \ldots, d_{\frac{n}{2}+k-1}^- \geq d_{\frac{n}{2}-1}^- \implies d_{\sigma(n-2k)} + d_{\sigma(n-2k+2)} + \cdots + d_{\sigma(n-2)} \geq k d_{\frac{n}{2}-1}^-$.

Hence

$$|d_{\sigma(0)} - d_{\sigma(1)} + d_{\sigma(2)} - \cdots - d_{\sigma(n)}| \geq \left|\frac{1}{2}\eta + \left(k - \frac{n}{2}\right) d^-_{\frac{n}{2}-1}\right| \geq \frac{1}{2}\eta,$$

where the final inequality follows since $k \leq \frac{n}{2}$ and $d^-_{\frac{n}{2}-1} < 0$.

Thus $M^{(d)} \geq \frac{1}{2}\eta$.

C.1.2 Evaluating E_n

This section provides a lower bound for the quantity E_n, as defined in eq. (C.5).

To evaluate E_n we will use the Hermite-Genocchi Theorem (Theorem 3.3 in [4]), which relates integrals over the probability simplex to associated divided differences.

The divided difference of n points $(x_0, f(x_0)), \ldots, (x_{n-1}, f(x_{n-1}))$ is defined by

$$f[x_0, \ldots, x_{n-1}] := \sum_{j\in[n]} \frac{f(x_j)}{\prod_{k\neq j}(x_j - x_k)}, \tag{C.6}$$

where limits are taken if any of the x_j are equal. It can be shown that for repeated points (see Exercise 4.6.6 in [29])

$$f[\underbrace{x_0, \ldots, x_0}_{(r_0+1) \text{ times}}, \underbrace{x_1, \ldots, x_1}_{(r_1+1) \text{ times}}, x_2, \ldots, x_{n-1}] = \frac{1}{r_0! r_1!} \frac{\partial^{r_0+r_1}}{\partial x_0^{r_0} \partial x_1^{r_1}} f[x_0, x_1, x_2, \ldots, x_{n-1}], \tag{C.7}$$

where $x_0, \ldots, x_{n-1} \in \mathbb{R}$ are distinct.

Now, the Hermite-Genocchi Theorem states that

$$f[x_0, \ldots, x_{n-1}] = \frac{1}{(n-1)!} \int_{T_n} f^{(n-1)}(v_0 x_0 + \cdots v_{n-1} x_{n-1}) \; dV,$$

where we recall that $dV = (n-1)! \; \delta(1 - \sum_{i\in[n]} v_i) \; dv_0 \cdots dv_{n-1}$.

In order to evaluate E_n, we set $f^{(n-1)}(\xi) = (n-1)!|\xi|$. Thus

$$f(\xi) = \begin{cases} \frac{1}{n}\xi^n & \xi \geq 0 \\ -\frac{1}{n}\xi^n & \xi < 0 \end{cases}$$

and $E_n = f[1, -1, 1, -1, \ldots, 1, -1]$.

Let $m = \frac{n}{2} - 1$ (i.e. $n = 2m + 2$). Then by eq. (C.7) we have that

$$E_{2m+2} = \frac{1}{m!^2} \; \partial_0^m \partial_1^m f[x_0, x_1]|_{x_0=-1, x_1=1},$$

where we have used the notation $\partial_i \equiv \frac{\partial}{\partial x_i}$.

In the neighbourhood of $x_0 = -1, x_1 = 1$, we have (by eq. (C.6))

$$f[x_0, x_1] = -\frac{1}{2m+2} \frac{x_0^{2m+2} + x_1^{2m+2}}{x_0 - x_1},$$

and thus

$$E_{2m+2} = -\frac{1}{2m+2}\frac{1}{m!^2}\ A|_{x_0=-1,x_1=1}\,, \tag{C.8}$$

where

$$A = \partial_0^m \partial_1^m \left(\frac{x_0^{2m+2} + x_1^{2m+2}}{x_0 - x_1} \right).$$

We see that

$$\begin{aligned} A &= \partial_1^m \partial_0^m \left(\frac{x_0^{2m+2}}{x_0 - x_1} \right) - \partial_0^m \partial_1^m \left(\frac{x_1^{2m+2}}{x_1 - x_0} \right) \\ &= \partial_1^m \partial_0^m \left(\frac{x_0^{2m+2}}{x_0 - x_1} \right) - (\textit{same term with } x_0 \textit{ and } x_1 \textit{ interchanged}). \end{aligned} \tag{C.9}$$

We use the Leibniz product rule[e] to deduce that

$$\partial_0^m \left(x_0^n \left(\frac{1}{x_0 - x_1} \right) \right) = \sum_{k=0}^{m} \binom{m}{k} \left[\frac{(2m+2)!}{(2m+2-k)!} x_0^{2m+2-k} \right] \left[\frac{(-1)^{m-k}}{(x_0 - x_1)^{m+1-k}} (m-k)! \right],$$

and hence that the first term in eq. (C.9) is

$$\begin{aligned} &\partial_1^m \partial_0^m \left(x_0^n \left(\frac{1}{x_0 - x_1} \right) \right) \\ &= \sum_{k=0}^{m} \binom{m}{k} \left[\frac{(2m+2)!}{(2m+2-k)!} x_0^{2m+2-k} \right] \left[\frac{(-1)^{m-k}}{(x_0 - x_1)^{2m+1-k}} (2m-k)! \right] \\ &= (2m+2)!(-1)^m \sum_{k=0}^{m} \binom{m}{k} \frac{(-1)^k (2m-k)!}{(2m+2-k)!} \frac{x_0^{2m+2-k}}{(x_0 - x_1)^{2m+1-k}} \\ &= (2m+2)!(-1)^m (x_0 - x_1) \sum_{k=0}^{m} \binom{m}{k} \frac{(-1)^k}{(2m+2-k)(2m+1-k)} \left(\frac{x_0}{x_0 - x_1} \right)^{2m+2-k}. \end{aligned}$$

Substituting this into eq. (C.9) and setting $x_0 = -1, x_1 = 1$ gives

$$A|_{x_0=-1,x_1=1} = -4(2m+2)!(-1)^m \sum_{k=0}^{m} \binom{m}{k} \frac{(-1)^k}{(2m+2-k)(2m+1-k)} \left(\frac{1}{2} \right)^{2m+2-k}.$$

Now set

$$B = (-1)^m \sum_{k=0}^{m} \binom{m}{k} \frac{(-1)^k}{(2m+2-k)(2m+1-k)} \gamma^{2m+2-k}$$

so that

$$A|_{x_0=-1,x_1=1} = -4(2m+2)!B|_{\gamma=\frac{1}{2}}. \tag{C.10}$$

[e] $(uv)^{(m)} = \sum_{k=0}^{m} \binom{m}{k} u^{(k)} v^{(m-k)}$

Next, note that

$$\frac{\partial^2 B}{\partial \gamma^2} = (-1)^m \sum_{k=0}^{m} \binom{m}{k} (-1)^k \gamma^{2m-k} = \gamma^m \sum_{k=0}^{m} \binom{m}{k} (-\gamma)^{m-k} = \gamma^m (1-\gamma)^m,$$

and thus

$$\begin{aligned} B|_{\gamma=\frac{1}{2}} &= \int_{z=0}^{\frac{1}{2}} \int_{\alpha=0}^{z} \alpha^m (1-\alpha)^m \ d\alpha \, dz + C \\ &= \int_{z=0}^{\frac{1}{2}} B_z(m+1, m+1) \ dz + C, \end{aligned}$$

where $B_z(p,q) = \int_0^z \alpha^{p-1}(1-\alpha)^{q-1} \ d\alpha$ is the incomplete Beta function. By setting $m = 0$ it is easy to deduce that $C = 0$.

Now, the indefinite integral of the incomplete Beta function is

$$\int B_z(p,q) \ dz = zB_z(p,q) - B_z(p+1,q),$$

and hence we deduce that

$$\begin{aligned} B|_{\gamma=\frac{1}{2}} &= \frac{1}{2} B_{\frac{1}{2}}(m+1, m+1) - B_{\frac{1}{2}}(m+2, m+1) \\ &= \int_0^{\frac{1}{2}} \alpha^m (1-\alpha)^m d\alpha - \int_0^{\frac{1}{2}} \alpha^{m+1} (1-\alpha)^m d\alpha \\ &= \frac{1}{2} \left[\int_0^{\frac{1}{2}} \alpha^m (1-\alpha)^m \underbrace{(1-2\alpha)}_{=(1-\alpha)-\alpha} \ d\alpha \right] \\ &= \frac{1}{2} \int_0^{\frac{1}{2}} (\alpha^m (1-\alpha)^{m+1} - \alpha^{m+1}(1-\alpha)^m) \ d\alpha \\ &= \frac{1}{2(m+1)} \int_0^{\frac{1}{2}} \frac{d(\alpha^{m+1}(1-\alpha)^{m+1})}{d\alpha} \ d\alpha \\ &= \frac{1}{2(m+1)} [\alpha^{m+1}(1-\alpha)^{m+1}]_0^{1/2} \\ &= \frac{1}{2^{2m+3}(m+1)}. \end{aligned}$$

Substituting this into eq. (C.10) and subsequently into eq. (C.8), we get

$$\begin{aligned} E_{2m+2} &= -\frac{1}{2m+2} \frac{1}{m!^2} \cdot -4(2m+2)! \cdot \frac{1}{2^{2m+3}(m+1)} \\ &= \frac{(2m+1)!}{2^{2m+1} m!^2 (m+1)} \\ &= \frac{2m+1}{m+1} \cdot \frac{(2m)!}{m!^2} \cdot \frac{1}{2^{2m+1}}. \end{aligned} \tag{C.11}$$

Stirling's formula [28] tells us that for large m

$$m! \sim \sqrt{2\pi} m^{m+\frac{1}{2}} e^{-m},$$

and thus

$$\frac{(2m)!}{m!^2} \sim \frac{\sqrt{2\pi}(2m)^{2m+\frac{1}{2}} e^{-2m}}{2\pi m^{2m+1} e^{-2m}} = \frac{2^{2m}}{\sqrt{m\pi}}.$$

For large m, $\frac{2m+1}{m+1} \sim 2$, and thus eq. (C.11) tells us that

$$E_{2m+2} \sim \frac{1}{\sqrt{m\pi}}.$$

Replacing m with $\frac{n}{2} - 1$, we deduce that

$$E_n \sim \sqrt{\frac{2}{\pi}} \frac{1}{\sqrt{n}}.$$